

PUBLIC
SAP HANA Platform 2.0 SPS 08
Document Version: 1.0 – 2024-11-20

SAP HANA Security Guide for SAP HANA Platform

© 2026 SAP SE or an SAP affiliate company. All rights reserved.

Content

1 SAP HANA Security Guide. 7

2 Security Information by Guide. 9

3 SAP HANA Security Patches. 13

4 SAP HANA Overview. 15

4.1 The SAP HANA Database. 15

4.2 SAP HANA XS and Development Infrastructure. 16

SAP HANA XS, Advanced Model. 16

SAP HANA XS, Classic Model. 17

4.3 Technical System Landscape. 18

Overview of SAP HANA Security Functions. 19

Database Isolation. 23

Security Considerations After Updating a Single-Container System. 25

4.4 SAP HANA Implementation Scenarios. 26

SAP HANA as a Data Mart. 26

SAP HANA in a Classic 3-tier Architecture. 28

SAP HANA as Technical Infrastructure for Native Application Development. 30

5 SAP HANA Network and Communication Security. 34

5.1 Communication Channels. 34

5.2 Network Security. 36

5.3 Securing Data Communication. 40

Secure Communication Between SAP HANA and JDBC/ODBC Clients. 42

Secure Communication Between SAP HANA and an LDAP Directory Server. 59

Secure Communication Between SAP HANA XS Classic and HTTP Clients. 62

Secure Internal Communication. 64

6 SAP HANA User Management. 73

6.1 User Types. 73

6.2 User Groups. 76

Connect Restrictions. 80

SQL Statements and Authorization for User Group Administration (Reference). 83

6.3 User Administration Tools. 86

6.4 Predefined Database Users. 89

Deactivate the SYSTEM User. 97

6.5 Predefined Operating System Users. 98

7 SAP HANA Authentication and Single Sign-On. 100
7.1 User Authentication Mechanisms. 100
7.2 Logon Checks. 104
7.3 Troubleshooting Authentication Problems. 105
7.4 Password Policy. 105
Password Policy Configuration Options. 107
Password Exclude List. 114
_SYS_PASSWORD_BLACKLIST. 115
7.5 Single Sign-On Integration. 116
Single Sign-On Using Kerberos. 116
Single Sign-On Using SAML 2.0. 118
Single Sign-On Using SAP Logon and Assertion Tickets. 122
Single Sign-On Using JSON Web Tokens. 123
7.6 X.509 Certificate-Based User Authentication. 127
7.7 LDAP User Authentication. 130

8 SAP HANA Authorization. 135
8.1 Privileges. 136
System Privileges. 139
Object Privileges. 146
Analytic Privileges. 154
Package Privileges. 175
Application Privileges. 176
Prerequisites for Granting and Revoking Privileges and Roles. 178
8.2 Database Roles. 182
Predefined Database (Catalog) Roles. 183
Catalog Roles and Design-Time Roles Compared. 188
SAP HANA DI Roles. 191
Repository Roles. 194
8.3 Authorization in the Repository of the SAP HANA Database. 199
Developer Authorization in the Repository. 200
_SYS_REPO Authorization in the Repository. 201
Granting and Revoking Privileges on Activated Repository Objects. 202
8.4 Cross-Database Authorization in Tenant Databases. 204
8.5 LDAP Group Authorization. 206
LDAP Group Authorization for Existing Users. 207
8.6 Shared Business Authorizations in SAP HANA. 210
GENERATE_STRUCTURED_PRIVILEGE_PFCG_CONDITION (SYS). 212

9 SAP HANA Data Masking. 215
9.1 Masking Definition. 217
9.2 Authorization in Masked Tables and Views. 218

9.3 Analysis of Effective Mask Expressions. . . . 224
9.4 Example: Masking Data with DEFAULT Mask Mode. . . . 224
9.5 Example: Masking Data in Object Hierarchy with SESSION USER Mask Mode 225
9.6 Example: Masking Data in a View with Structured Privilege Check. . . . 227
9.7 Example: Masking Data in View Hierarchy with Structured Privilege Check. . . . 228

10 SAP HANA Data Anonymization. . . . 231

11 Data Storage Security in SAP HANA. . . . 233
11.1 Data Security in the File System. . . . 233
11.2 Server-Side Data Encryption Services. . . . 234
Server-Side Secure Stores. . . . 235
Encryption Key Management. . . . 239
Data and Log Volume Encryption. . . . 241
Backup Encryption. . . . 243
Encryption Configuration Control. . . . 247
Internal Application Encryption Service. . . . 249
Root Key Backup. . . . 252
Local Secure Store (LSS). . . . 253
11.3 Client-Side Data Security. . . . 272
Client-Side Data Encryption. . . . 273
Protection of Data in SAP HANA Studio Workspaces. . . . 274
11.4 Cryptographic Service Provider. . . . 275

12 Auditing Activity in SAP HANA. . . . 277
12.1 Audit Policies. . . . 278
Audit Policies for Tenant Databases. . . . 282
Actions Audited by Default Audit Policy. . . . 283
12.2 Audit Trails. . . . 284
Audit Trail Layout for Trail Target CSV and SYSLOG. . . . 287
Audit Trail Layout for Trail Target Database Table. . . . 290
12.3 Auditing Configuration and Audit Policy Management. . . . 294
System Properties for Configuring Auditing. . . . 295
12.4 Best Practices and Recommendations for Creating Audit Policies. . . . 298

13 Certificate Management in SAP HANA. . . . 303
13.1 Certificate Management in the Database. . . . 303
In-Database Certificate Management Workflow. . . . 305
Client Certificates. . . . 306
Certificate Collections 307
SQL Statements and Authorization for In-Database Certificate Management (Reference). . . . 311
13.2 Certificate Management in the File System. . . . 318

14 Data Protection and Privacy in SAP HANA. . . . 321

14.1 Deletion of Personal Data. 324

15 Security Risks of Trace, Dump, and Captured Workload Files. 326

16 Security of Further SAP HANA Components and Capabilities. 328

16.1 Security Aspects of SAP HANA Platform Lifecycle Management. 329

16.2 Security of SAP HANA Content. 330

16.3 Security Aspects of SAP HANA Smart Data Access. 331

16.4 Security Aspects of SAP HANA R Integration. 332

16.5 Security for SAP HANA Replication Technologies. 333

17 Security for SAP HANA Extended Application Services, Advanced Model. 336

17.1 Technical System Landscape of SAP HANA XS Advanced. 338

Application Server Components. 341

Users and Clients. 343

17.2 User Administration and Authentication in SAP HANA XS Advanced. 344

User Management. 344

Predefined XS Advanced Users. 347

Predefined Database Roles for XS Advanced. 353

User Authentication. 356

User Administration Tools. 357

17.3 Authorization in SAP HANA XS Advanced. 357

Organizations and Spaces. 358

Scopes, Attributes, and Role Collections. 364

Controller Role Model. 367

Authorization Management Tools 371

17.4 Network and Communication Security with SAP HANA XS Advanced. 374

Security Areas. 374

Public Endpoints. 375

Single-Host Scenario. 377

Multiple-Host Scenario. 378

XS Advanced Certificate Management. 380

17.5 Data Storage Security. 382

System Component Storage. 383

Application Storage. 384

17.6 Security-Relevant Logging and Tracing. 385

Audited Operations. 385

Audit Trails. 385

Application Auditing. 386

17.7 Data Protection and Privacy in SAP HANA XS Advanced. 387

Processing of Personal Data in Platform-Controlled Artifacts 390

Processing of Personal Data in Standard XS Advanced Applications and Services. 394

17.8 Security Aspects of SAP Web IDE for SAP HANA . . . 394
User Authorization and Authentication . . . 396
Known Security-Related Issues . . . 398

18 Security for SAP HANA Deployment Infrastructure (HDI) . . . 400

19 SAP HANA Security Reference Information . . . 407
19.1 SQLScript Security Procedures . . . 407
GENERATE_STRUCTURED_PRIVILEGE_PFCG_CONDITION (SYS) . . . 407
GET_INSUFFICIENT_PRIVILEGE_ERROR_DETAILS . . . 409
IS_VALID_PASSWORD (SYS) . . . 411
IS_VALID_USER_NAME (SYS) . . . 412
19.2 Security Reference for Tenant Databases . . . 414
Restricted Features in Tenant Databases . . . 414
Default Blacklisted System Properties in Tenant Databases . . . 417
19.3 Unpermitted Characters in User Names . . . 419
19.4 Components Delivered as SAP HANA Content . . . 420
Administration . . . 420
Application Lifecycle Management . . . 426
Runtime Libraries . . . 429
Configuration . . . 430
Supportability and Development . . . 431
User Interface . . . 434
Documentation . . . 437

1 SAP HANA Security Guide

The SAP HANA Security Guide is the entry point for all information relating to the secure operation and configuration of SAP HANA.

> **Note**
>
> This guide does not cover security-relevant information of some additional capabilities that may be installed in the SAP HANA system, such as SAP HANA accelerator for SAP ASE or SAP HANA streaming analytics. For more information, see Security of Further SAP HANA Components and Capabilities [page 328].

Why is Security Necessary?

Protecting sensitive information is one of the most important priorities for you as an SAP HANA customer. You need to meet ever increasing cyber-security challenges, keep your systems secure, and stay on top of the compliance and regulatory requirements of today's digital world. SAP HANA allows you to securely run and operate SAP HANA in a variety of environments and to implement your specific compliance, security, and regulatory requirements.

1.1 Important Configurations

> **Caution**
>
> SAP HANA has many configuration settings that allow you to customize your system specifically for your implementation scenario and system environment. Some of these settings are particularly important for the security of your system, and misconfiguration could leave your system vulnerable. For this reason, a security checklist of critical configuration settings is available. See *SAP HANA Security Checklists and Recommendations* on SAP Help Portal.
>
> We recommend that you check your system for critical configurations and the latest security patches. Specifically, we recommend verifying that:
>
> - The master keys of the following stores have been changed:
> - The secure store in the file system (SSFS) of the instance
> - The SSFS used by the system public key infrastructure (PKI)
> - The SAP HANA secure user store (`hdbuserstore`) of the SAP HANA client
> - Critical privileges are only assigned to trusted users and critical privilege combinations are avoided if possible.
> - The network configuration of your SAP HANA system is set up to protect internal SAP HANA communication channels.

- The latest security patches are applied for the SAP HANA system as well as the underlying operating system.

For more information about how to check critical settings and how to find information on recommended settings, see *SAP HANA Security Checklists and Recommendations* on SAP Help Portal.

For more information about keeping your system up to date by installing the latest security patches, see the section on security patches.

Related Information

SAP HANA Security Patches [page 13]
SAP HANA Security Checklists and Recommendations

2 Security Information by Guide

Find security-related information by guide

In addition to this *SAP HANA Security Guide*, several other documents in the SAP HANA documentation set provide task- and tool-oriented security information for specific roles and lifecycle phases. Security-related reference documentation is also available.

SAP HANA Security Guide

This guide is the entry point for all information relating to the secure operation and configuration of an on-premise deployment of SAP HANA. Use it to understand security concepts and features of the SAP HANA database and the SAP HANA extended application services, advanced model.

For an overview of the security features of the SAP HANA database, see Overview of SAP HANA Security Functions [page 19].

> **Note**
>
> Security information for some SAP HANA components, as well as additional capabilities that may be installed in your SAP HANA system is available in other documents. See Security of Further SAP HANA Components and Capabilities [page 328].

SAP HANA Security Checklists and Recommendations

This document contains information and recommendations on critical settings with a security impact. It covers:

- SAP HANA database
- SAP HANA XS advanced

Open SAP HANA Security Checklists and Recommendations

SAP HANA Administration

The *SAP HANA Administration Guide* is the entry point for all information related to the ongoing operation and maintenance of the SAP HANA platform. Use it together with *SAP HANA Administration with SAP HANA Cockpit* to learn how to perform the following security-related administration tasks:

- Monitoring critical security settings
- Creating and provisioning database users
- Configuring auditing and creating audit policies
- Configuring data-at-rest encryption and managing encryption keys
- Managing in-database certificates and certificate collections

To help you integrate SAP HANA securely into your network environment, refer to the section on network administration with detailed information on ports and connections, and host name resolution.

Open the SAP HANA Administration Guide

Open SAP HANA Administration with SAP HANA Cockpit: Security and User Management View

SAP HANA Developer Guide

This guide describes the complete application-development process for SAP HANA XS advanced, including aspects of application security, for example:

- Understanding user identity, authentication, and authorization
- Defining the authentication and authorization models
- Protecting applications from Web-based attacks

Open the SAP HANA Developer Guide

SAP HANA References

The following SAP HANA references contain essential information for administrators and developers with a security focus:

- SAP HANA SQL Reference Guide for SAP HANA Platform
- SAP HANA Data Anonymization Guide
- SAP HANA Client Interface Programming Reference
- SAP HANA Client-Side Data Encryption Guide

> **Note**
>
> The topics listed above for each guide are not intended to be exhaustive but representative.

> **→ Tip**
>
> For a high-level overview of all security capabilities in the SAP HANA platform, as well as links to security-related blog posts, videos, and white papers, visit http://sap.com/hanasecurity.

Target Audiences

Document		Content Type
SAP HANA Security Guide	Technology consultants, security consultants, system administrators	Concept and overview
SAP HANA Security Checklists and Recommendations	System administrators	Reference
SAP HANA Administration Guide	System administrators	Concept and overview, task- and role-oriented
SAP HANA Administration with SAP HANA Cockpit	System administrators	Task- and role-oriented
SAP HANA Developer Guide for XS Advanced Model	Database developers, application programmers and client UI developers working in the SAP HANA XS advanced model using the SAP Web IDE for SAP HANA	Task- and role-oriented
SAP HANA SQL and System Views Reference	Technology consultants, security consultants, system administrators	Reference
SAP HANA Data Anonymization Guide	Developers	Reference, task- and role-oriented

Document		Content Type
SAP HANA Client-Side Data Encryption Guide	System administrators	Concept and overview, task- and role-oriented
SAP HANA Client Interface Programming Reference	Developers	Reference

Additional Documentation Resources

Further SAP HANA Guides

For more information about the SAP HANA landscape, including installation and administration, see SAP HANA Platform on SAP Help Portal.

Important SAP Notes

Important SAP Notes that apply to SAP HANA security are listed in the table below. In addition, SAP publishes information related to security corrections and improvements through SAP security notes. For more information about security notes, see the section on security patches.

> **Note**
>
> SAP supports that customers install additional tools on the SAP HANA appliance within defined boundaries. It is the responsibility of the customer to ensure that the network channels used by those tools are appropriately protected. For detailed information, see the SAP Notes listed below. For SAP HANA deployments that use the SAP HANA tailored data center integration model, the regulations are less restrictive compared to the appliance delivery model. The listed SAP notes can give guidance of the options available for securing SAP HANA.

SAP Note	Title
2380229	SAP HANA 2.0: Central Note
1730928	Using external software in an SAP HANA applianceFor more information about specific topics, see the quick links in the table below.
1730929	Using external tools in an SAP HANA appliance
1730930	Using anti-virus software in an SAP HANA appliance
1730996	Non-recommended external software and software versions
1730997	Non-recommended versions of anti-virus software
1730998	Non-recommended versions of backup tools
1730999	Configuration changes in SAP HANA appliance
1731000	Non-recommended configuration changes

Other Information

Content	SAP Service Marketplace or SDN Quick Link
SAP Notes	https://support.sap.com/notes http://support.sap.com/securitynotes
Released platforms	https://apps.support.sap.com/sap/support/pam
SAP Solution Manager community	https://go.sap.com/community/topic/solution-manager.html
SAP NetWeaver community	https://go.sap.com/community/topic/netweaver.html
SAP HANA in-memory computing community	https://go.sap.com/community/topic/hana.html

Related Information

SAP HANA Security Patches [page 13]
SAP HANA Security Checklists and Recommendations
SAP HANA Administration Guide
SAP HANA Developer Guide for XS Advanced Model (SAP Web IDE)
SAP HANA SQL Reference Guide for SAP HANA Platform
SAP HANA Client Interface Programming Reference

3 SAP HANA Security Patches

To ensure the security of SAP HANA, it's important that you keep your systems up to date by installing the latest SAP HANA revision and monitoring SAP security notes.

SAP HANA Revisions

Security-related code improvements and corrections for SAP HANA are shipped with SAP HANA revisions. SAP publishes information related to security corrections and improvements through SAP security notes. In general, security notes contain information about both the affected SAP HANA application areas and specific measures that protect against the exploitation of potential weaknesses. Additional security measures are also documented here. SAP security notes are released as part of the monthly SAP Security Patch Day.

We recommend that you regularly review new security notes for SAP HANA application areas and decide whether they are relevant in the context of your systems and environment.

For more information about SAP security notes and the SAP Security Patch Day, see SAP Support Portal at http://support.sap.com/securitynotes.

> **Note**
>
> To get full access to SAP Support Portal, you need an authorized user ID.

For a list of all SAP HANA application areas, see the *SAP HANA Master Guide*.

For more information about updating SAP HANA to a new revision, see the *SAP HANA Server Installation and Update Guide*.

Operating System Patches

Install security patches for your operating (OS) system as soon as they become available. If a security patch impacts SAP HANA operation, SAP will publish an SAP Note where this fact is stated. It is up to you to decide whether to install such patches.

If your SAP HANA system runs on SUSE Linux Enterprise Server 11.x for SAP Applications, see SAP Note 1944799.

If your SAP HANA system runs on Red Hat Enterprise Linux (RHEL) 6.x, see SAP Note 2009879.

Related Information

SAP HANA Application Areas

Updating the SAP HANA System

SAP Note 1944799

SAP Note 2009879

4 SAP HANA Overview

SAP HANA is an in-memory platform for doing real-time analytics and for developing and deploying real-time applications. For on-premise deployment, SAP HANA comes either pre-installed on certified hardware provided by an SAP hardware partner (appliance delivery model) or must be installed on certified hardware by a certified administrator (tailored data center integration model).

However, SAP HANA is more than a database management system. It is also a comprehensive platform for the development and execution of native data-intensive applications that run efficiently in SAP HANA, taking advantage of its in-memory architecture and parallel execution capabilities.

4.1 The SAP HANA Database

At the core of SAP HANA is the high-performance, in-memory SAP HANA database.

SAP HANA is an in-memory platform that combines an ACID-compliant database with advanced data processing, application services, and flexible data integration services. The SAP HANA database can act as a standard SQL-based relational database. In this role, it can serve as either the data provider for classical transactional applications (OLTP) and/or as the data source for analytical requests (OLAP). Database functionality is accessed through an SQL interface.

Standard Database Interfaces

SAP HANA provides standard database interfaces such as JDBC and ODBC and supports standard SQL with SAP HANA-specific extensions.

Data Provisioning

Several data provisioning mechanisms are available for getting data from different sources into SAP HANA. For example, in a data mart or analytics scenario, data is replicated into SAP HANA from source systems using one of the supported replication technologies). For applications that use SAP HANA as their primary database (such as SAP S/4HANA), data is created directly in SAP HANA.

Data Recovery

Although the SAP HANA database holds the bulk of its data in memory for maximum performance, it still uses persistent storage to support system restart and recovery. There's minimal delay and no loss of data in the

event of failure. For example, after a power failure, the database can be restarted like any disk-based database and returned to its most recent consistent state. In addition, SAP HANA provides functions for backup and recovery, as well as high availability (disaster recovery and fault recovery).

Related Information

Security for SAP HANA Replication Technologies [page 333]

4.2 SAP HANA XS and Development Infrastructure

SAP HANA includes the SAP HANA extended application services (SAP HANA XS), a layer on top of SAP HANA that provides the platform for running SAP HANA-based Web applications.

SAP HANA XS, Advanced Model

Available since SAP HANA 1.0 SPS 11, the SAP HANA XS advanced model represents an evolution of the application server architecture within SAP HANA by building upon the strengths (and expanding the scope) of SAP HANA extended application services (XS), classic model.

The SAP HANA XS advanced platform supports several programming languages and execution environments, such as Java, and Node.js. The SAP HANA XS advanced application runtimes are invoked over HTTP and communicate with the SAP HANA database via SQL.

The database part of an SAP HANA XS advanced application (for example the definitions of tables, views, and procedures) is deployed using the SAP HANA deployment infrastructure (SAP HANA DI, or HDI). HDI is a service layer of the SAP HANA database that simplifies the consistent deployment of SAP HANA database objects. It supports isolated deployment containers, which can be used, for example, to deploy several instances of the same application on the same SAP HANA database.

SAP Web IDE for SAP HANA is the browser-based development environment for SAP HANA-based applications. It can be used to develop all layers of an application, including UI, XS advanced server applications, and SAP HANA database content. It is based on SAP HANA XS advanced and HDI, and uses Git for source code management.

> → Recommendation
>
> SAP recommends that customers and partners who want to develop new applications use SAP HANA XS advanced model. If you want to migrate existing XS classic applications to run in the new XS advanced run-time environment, SAP recommends that you first check the features available with the installed version of XS advanced; if the XS advanced features match the requirements of the XS classic application you want to migrate, then you can start the migration process. For more information, see the *SAP HANA XS Advanced Migration Guide*.

Downloading XS Advanced from SAP Marketplace

SAP HANA Extended Application Services, advanced model, is available not only on the SAP HANA media but also as a separate component on SAP Marketplace. Users with the required S-User ID can download the latest version of XS advanced component in the package `SAP EXTENDED APP SERVICES 1` from the following location:

Service Marketplace > *Software Downloads [Downloads]* > *SUPPORT PACKAGES & PATCHES* > *By Alphabetical Index (A-Z)* > *H* > *SAP HANA PLATFORM EDITION*:

- *SAP HANA PLATFORM EDITION 2.0* > *SAP EXTENDED APP SERVICES 1*

> → Tip
>
> SAP HANA Extended Application Services, advanced model, is backwards compatible; you can provide access to new features by installing the latest version of the XS advanced component even on older versions of SAP HANA. To download the package `SAP EXTENDED APP SERVICES 1`, see *SAP Software Download Center* in *Related Information* below.

SAP HANA XS, Classic Model

SAP HANA XS classic is the original implementation of SAP HANA XS. The classic XS server is fully integrated into the SAP HANA database and provides application server functions. Accessible through HTTP, the XS server can deliver data through Open Data Protocol (OData) calls and HTML user interfaces. For creating new structures and programs, for example modeling database structures, analytical queries, reports and procedures, as well as developing applications, SAP HANA provides a development environment. This development environment is integrated into the SAP HANA studio and the SAP HANA Web-based Development Workbench. Design-time artifacts, such as custom applications, roles, and application content, are managed in SAP HANA's built-in repository. Design-time objects can be transported from development systems to test and production systems.

> ⓘ Note
>
> SAP HANA XS, classic and the SAP HANA repository are deprecated as of SAP HANA 2.0 SPS 02. For more information, see SAP Note 2465027.

Related Information

SAP HANA as Technical Infrastructure for Native Application Development [page 30]
Security for SAP HANA Extended Application Services, Advanced Model [page 336]
Security Aspects of SAP Web IDE for SAP HANA [page 394]
SAP HANA XS Advanced Migration Guide
SAP Note 2465027
SAP Software Download Center (Logon required)

4.3 Technical System Landscape

An SAP HANA system comprises multiple isolated databases and may consist of one host or a cluster of several hosts (scale-out system).

An SAP HANA system, identified by a single system ID (SID), contains one or more tenant databases and one system database. Databases are identified by a SID and a database name. From the administration perspective, there is a distinction between tasks performed at system level and those performed at database level. Database clients, such as the SAP HANA cockpit, connect to specific databases.

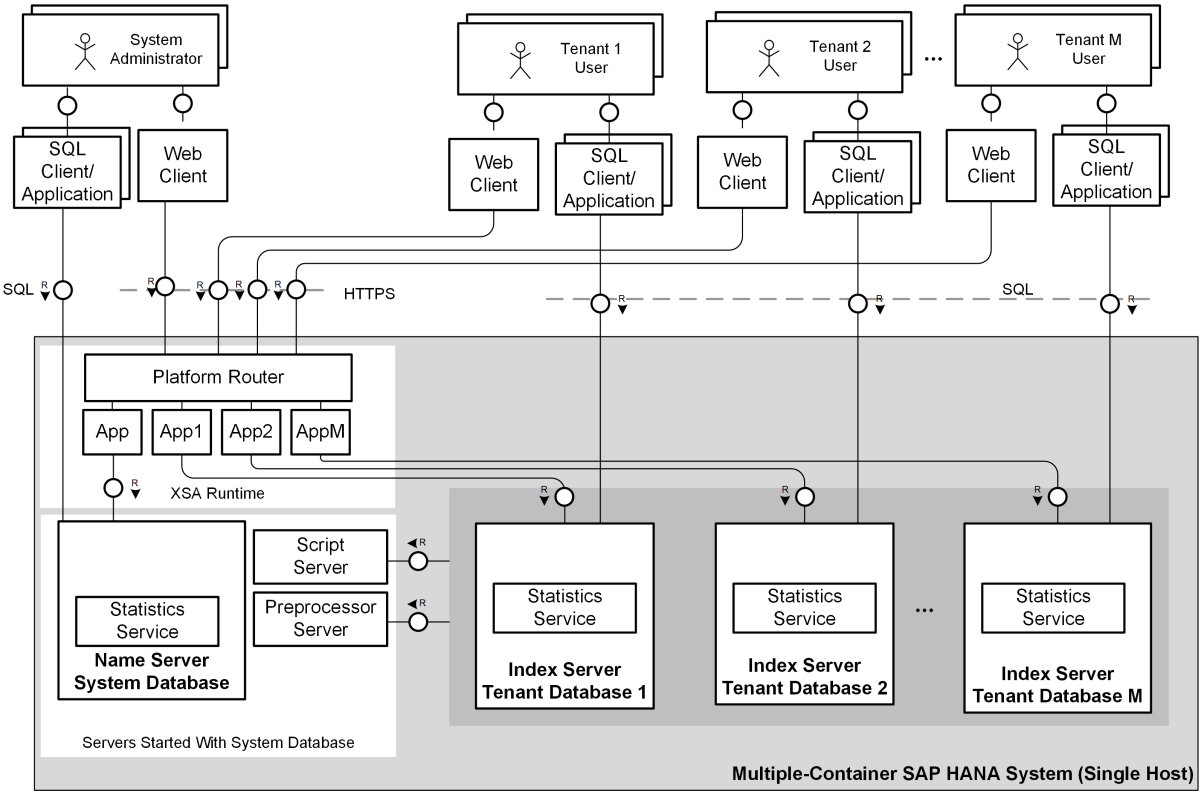
All the databases in a system share the same installation of database system software, the same computing resources, and the same system administration. However, each database is self-contained and fully isolated with its own set of database users, database catalog, persistence, and so on.

The System Database

The system database, which is created during installation, is used for central system administration, for example the creation of tenant databases and global system configuration. The system database stores overall system landscape information, including knowledge of the tenant databases that exist in the system. However, it doesn't own database-related topology information, that is, information about the location of tables and table partitions in databases. Database-related topology information is stored in the relevant tenant database catalog.

Server Architecture

An example of the basic architecture of a single-host SAP HANA system with three tenant databases is shown below. For more information about system architecture, see the *SAP HANA Administration Guide*.



SAP HANA System with Tenant Databases

Related Information

SAP HANA System Architecture Overview

4.3.1 Overview of SAP HANA Security Functions

SAP HANA provides a range of security features and functions at the database and system level to ensure secure access control and secure system setup and configuration.

Security Features of the SAP HANA Database

The following table provides an overview of standard security features in the SAP HANA database. For more detailed information, see the relevant section in this guide.

Security Feature	Description
User and role management	Every tenant database has its own database users and roles, including a tenant database-specific superuser SYSTEM. Depending on the isolation level of the system, there may be only one operating system (OS) user (the default `<sid>adm` user), or one OS user for each tenant database, which must be created.
Authentication and SSO	The SAP HANA database supports a number of authentication mechanisms, including database user name/password, SAML bearer tokens, JSON Web tokens, Kerberos, and LDAP directory server name and password. Whether a per-database configuration is possible depends on the authentication mechanism and the user client: • Authentication by database user name and password is database specific. • For Kerberos-based authentication, a per-database configuration is not possible. Databases users in all databases must be mapped to users in the same Key Distribution Center. • For SAML and JWT-based authentication, a per-database configuration is possible for JDBC/ODBC client access. Different trust stores (containing different certificates) can be configured for individual databases. For this purpose, we recommend using certificates and certificate collections (also referred to as personal security environments or PSEs) stored in the database as opposed to the file system. • For LDAP-based authentication, a per-database configuration is possible. Connections to different LDAP directory servers can be set up by creating separate LDAP providers in each database. To secure communication between the SAP HANA database and the LDAP server (including the transmission of passwords), different trust stores (containing different certificates) can be configured for individual databases using in-memory certificates and certificate collections. **Note** LDAP-based authentication is only possible for users if authentication using their local SAP HANA password is disabled. • Database-specific trust stores cannot be configured for HTTP client access through SAP HANA Extended Services, classic model (SAP HANA XS classic). Therefore, user authentication based on SAML assertions and X.509 certificates cannot be database specific.
Authorization	SAP HANA's standard authorization mechanisms are applied to users at the database level with the following additions: • In the system database, the system privilege DATABASE ADMIN exists to allow system administrators to perform certain tasks on tenant databases (for example, stop a tenant database or back up a tenant database). • A cross-database authorization mechanism exists to support read-only queries between tenant databases. This is made possible through the association of a user in one tenant database with a user in another database. Cross-database access is disabled by default and must be enabled and configured by a system administrator before such user mappings can be set up.

Security Feature	Description
Encryption of data communication in the network	Secure communication based on the Transport Layer Security (TLS)/Secure Sockets Layer (SSL) protocol can be configured separately for external communication between individual databases and JDBC/ODBC clients. Separate key and trust stores must be available and configured for each database. For this purpose, we recommend using certificates and certificate collections stored in the database as opposed to the file system. For communication with HTTP clients, a per-database configuration of TLS/SSL keys and certificates is also possible.
Data-at-rest encryption in the persistence layer	Data and log volume encryption, as well as data and log backup encryption can enabled for the system database and tenant databases individually. Data and log volume encryption ensures that anyone who can access the data and log volumes on disk using operating system commands cannot see the actual data and redo log entries. Backup encryption prevents unauthorized parties from reading the content of backups.
Masking and anonymization	Data masking is an additional layer of access control that can be applied to tables and views. A column mask protects sensitive or confidential data in a particular column of a table or view by transforming the data in such a way that it is only visible partially or rendered completely meaningless for an unprivileged user, while still appearing real and consistent. Anonymization allows you to gain statistically valid insights from your data while protecting the privacy of individuals. Unlike masking and pseudonymization, anonymization methods (also called privacy-enhancing methods) provide a more structured approach to modifying data for privacy protection. The quality of such anonymized or privacy-enhanced data is still sufficient for meaningful analysis. Data anonymization capabilities are integrated into calculation views. Masking and anonymization are both applied at the database level.
Auditing	Actions performed in every tenant database and the system database can be audited individually. To ensure the privacy of tenant database audit trails, they are by default written to a database table that is local to the database being audited. By default, tenant database administrators cannot change the audit trail targets for their database. The system administrator can , but this is not recommended. If the audit trail target for tenant databases is changed to the syslog, audit entries contain a field *Database Name* so that it is possible to differentiate entries from different tenant databases.

Additional System-Level Security Features

The following table provides an overview of additional feature to ensure the secure setup and configuration of the SAP HANA system .

Security Feature	Description
Database isolation	To maximize system security by preventing cross-tenant attacks through operating system mechanisms, it is possible to configure the system for high isolation. In high isolation mode, the processes of individual tenant databases must run under dedicated OS users belonging to dedicated OS groups, instead of all processes running under the single default OS user `<sid>adm` (low isolation). Data in the file system is protected using file and directory permissions.
Configuration change blocklist	To ensure the stability and performance of the overall system or for security reasons, it may be necessary to prevent certain system properties from being changed by tenant database administrators, for example, properties related to resource management. A configuration change blocklist (`multidb.ini`) is available for this purpose. This blocklist contains several critical properties by default. You can customize the default configuration as well as add further properties by editing the file, for example, in the SAP HANA cockpit.
Restricted features	To safeguard and/or customize your system, certain features of the SAP HANA database can be disabled in tenant databases.

Related Information

SAP HANA User Management [page 73]
SAP HANA Authentication and Single Sign-On [page 100]
Certificate Management in SAP HANA [page 303]
SAP HANA Authorization [page 135]
Securing Data Communication [page 40]
Data and Log Volume Encryption [page 241]
Backup Encryption [page 243]
Auditing Activity in SAP HANA [page 277]
SAP HANA Data Masking [page 215]
SAP HANA Data Anonymization [page 231]
Database Isolation [page 23]
Restricted Features in Tenant Databases [page 414]
Prevent Changes to System Properties in Tenant Databases

4.3.2 Database Isolation

Every tenant database is self-contained and isolated in terms of users, database catalog, repository, logs, and so on. However, to protect against unauthorized access at the operating system (OS) level, it's possible to increase isolation further through OS user separation and authenticated communication within databases.

OS User Separation

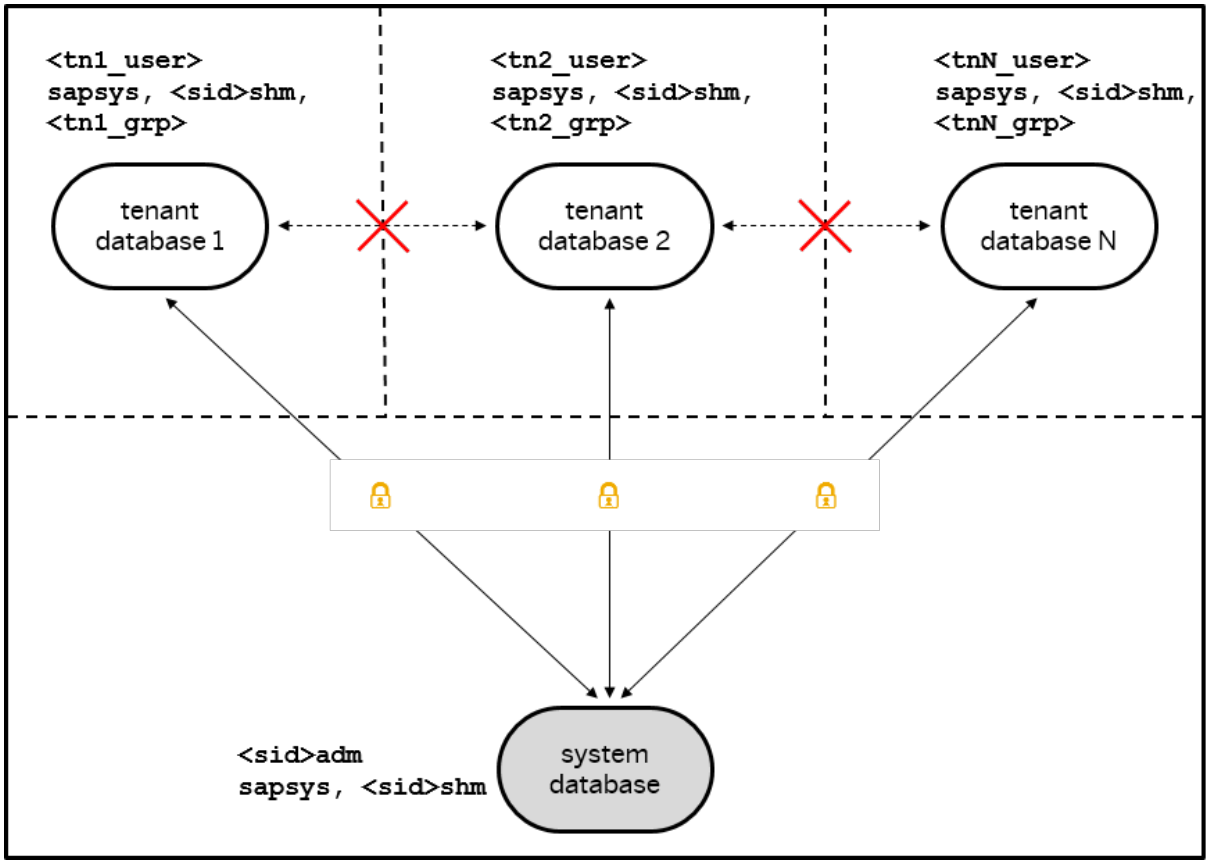
By default, all database processes run under the default OS user `<sid>adm`. If it's important to mitigate against cross-database attacks through OS mechanisms, you can configure the system for high isolation. In this way, the processes of individual tenant databases must run under dedicated OS users belonging to dedicated OS groups, instead of all database processes running under `<sid>adm`. Database-specific data on the file system is then protected using standard OS file and directory permissions.

> ⓘ Note
>
> `<sid>adm` is the OS user for the system database.

Authenticated Communication

In addition, once high isolation has been configured, internal database communication is secured using the Transport Layer Security (TLS)/Secure Sockets Layer (SSL) protocol. Certificate-based authentication is used to ensure that only the processes belonging to the same database can communicate with each other. It's also possible to configure internal communication so that all data communication within databases is encrypted.

> ⓘ Note
>
> If cross-database access is enabled, communication between configured tenant databases is allowed.



High Database Isolation

Configuration

You can specify the isolation level of the system during installation. The default isolation level is low. It's also possible to change the isolation level of an existing system (from low to high or from high to low) at any time. For more information, see *Increase the System Isolation Level* in the *SAP HANA Administration Guide*. Once high isolation has been configured, a dedicated OS user and group must exist for every tenant database. Otherwise, it's not possible to create or start a tenant database.

Internal database communication is secured with the same mechanism used for securing other internal SAP HANA communication channels. Once high isolation has been configured, authenticated communication within databases is enabled without any change required to the default TLS/SSL configuration for internal communication. However, encryption of data communication may need to be configured explicitly.

Related Information

File and Directory Permissions with High Isolation
Secure Internal Communication
Increase the System Isolation Level
SAP HANA Administration Guide

4.3.3 Security Considerations After Updating a Single-Container System

As of SAP HANA 2.0 SPS 01, all SAP HANA systems support tenant databases. If you updated a single-container system, this means that your system now has a system database and one tenant database. You therefore need to review aspects of your operations concept, including security.

Configuration Area	After Update
User administration	You need to set up new users for administration (at least for recovery) in the system database.
Network security	The system database uses additional ports. These ports that might be firewalled for security reasons in the system. If this the case, you need to open these ports so that the system database can be accessed from the SAP HANA cockpit on other hosts. The system database is accessible via SQL port 3`<instance_no>`13.
TLS/SSL configuration for external communication	If TLS/SSL is enabled for both the system database and tenant database, the in-database certificate collection containing the certificates used for trust validation is available only in the tenant database. If you want to use the same certificates, you will need import them into the certificate store of the system database and add them to a certificate collection there. **⚠ Caution** If TLS/SSL is being enforced for client connections (that is, parameter `[communication] sslEnforce` in *global.ini* set to `true`), it will not be possible to establish a connection to the system database. You have to set `sslEnforce` to `false` first. If the certificates used for trust validation are stored in a PSE in the file system, both the tenant database and the system database will have access, so no reconfiguration is required. However, you should validate that sharing the certificate stores for system database and tenant is actually intended.
Database isolation	The default system isolation level is low. It is possible to change the isolation level (from low to high or from high to low) at any time after the update. Once high isolation has been configured, a dedicated OS user and group must exist for every tenant database. Otherwise, it's not possible to create or start a tenant database.
Auditing	Existing audit policies are available in the tenant database database only. You need to create new audit policies for administration tasks in the system database.

For more information about non-security-related aspects, see the update section of the *SAP HANA Server Installation and Update Guide*.

Related Information

Updating a Single-Container System

4.4 SAP HANA Implementation Scenarios

How you implement SAP HANA determines what you need to consider from a security perspective.

For more detailed information, see the section on SAP HANA use cases in the *SAP HANA Master Guide*.

SAP HANA as a Data Mart [page 26]
In a data mart scenario, data is replicated from a source system such as SAP Business Suite into the SAP HANA database. Reporting is then carried out on the data in SAP HANA (for example, using read-only views, dashboards, and so on). Different architectures can be used in this scenario.

SAP HANA in a Classic 3-tier Architecture [page 28]
SAP HANA can be used as a relational database in a classic 3-tier architecture (client, application server, and database).

SAP HANA as Technical Infrastructure for Native Application Development [page 30]
SAP HANA extended application services (SAP HANA XS), advanced model is the default framework for native application development on SAP HANA. The SAP HANA XS, classic model is the deprecated application development framework of SAP HANA.

Related Information

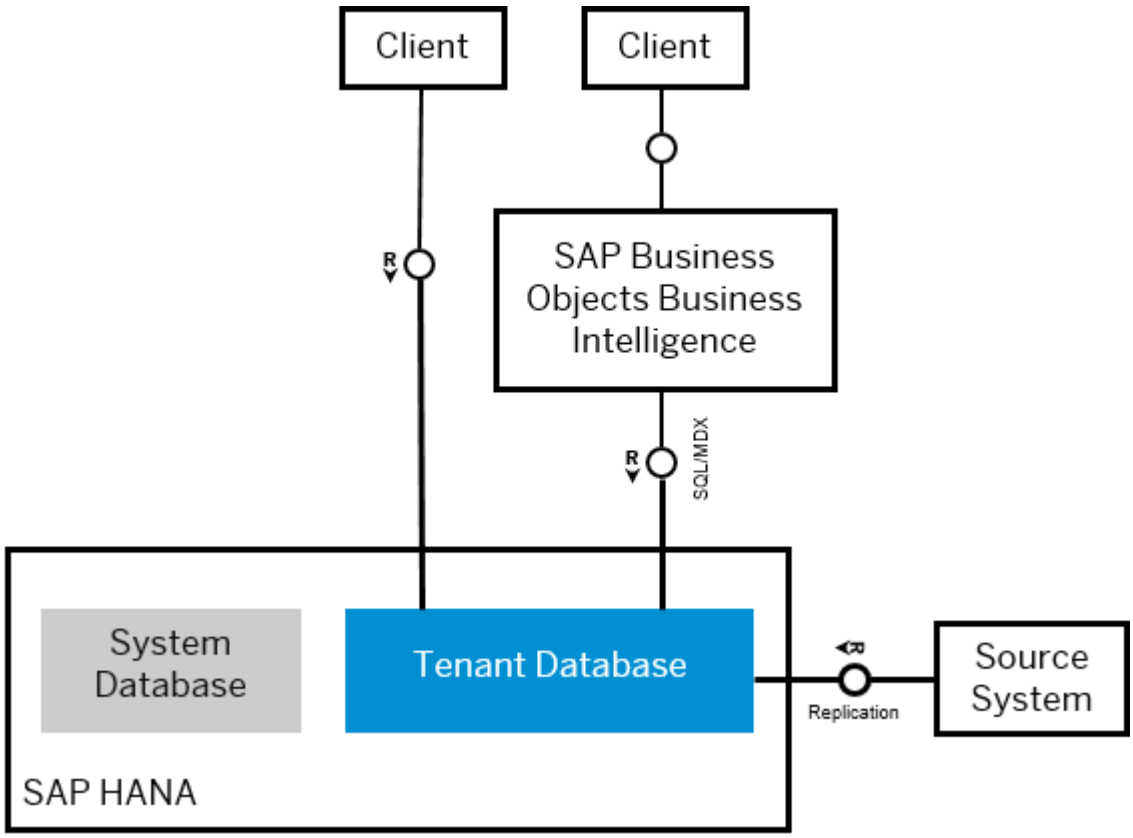
SAP HANA Use Cases

4.4.1 SAP HANA as a Data Mart

In a data mart scenario, data is replicated from a source system such as SAP Business Suite into the SAP HANA database. Reporting is then carried out on the data in SAP HANA (for example, using read-only views, dashboards, and so on). Different architectures can be used in this scenario.

For example, SAP HANA can be integrated into the SAP BusinessObjects Business Intelligence (BI) platform as a relational database. The source data can then be analyzed and reported on by SAP BusinessObjects Business Intelligence Suite products. Alternatively, SAP HANA can be accessed directly by BI clients such as Microsoft Excel. In this case, end-user clients connect directly to the database. These architectures are depicted in the following figure:

SAP HANA as a Data Mart



The implemented architecture determines the extent to which security-related aspects are handled in SAP HANA. However, user and role management in the database layer of SAP HANA is required, at least for technical users and administrators.

The following table outlines the relevance of SAP HANA security-related features in this implementation scenario.

SAP HANA Feature	Relevance in Scenario
User and role management	The extent to which SAP HANA user and role management is required in this scenario depends on your system architecture as follows. • If SAP HANA is integrated into a business intelligence solution (for example, SAP BusinessObjects Business Intelligence platform) only as the reporting database, end users and roles are managed in the relevant application server. User and role management in the database layer of SAP HANA is required only for technical database users and administrators. • If end users connect to the SAP HANA database directly through a SQL client (for example, SAP BusinessObjects Explorer or Microsoft Excel), user and role management in the database layer of SAP HANA is required for both end users and administrators.

SAP HANA Feature	Relevance in Scenario
Authentication and SSO	The extent to which authentication and SSO is handled in SAP HANA depends on your system architecture in the same way as described above. • If SAP HANA is used only as the data store, end-user authentication is handled in the application server. The relevant technical database user accounts are used to authenticate connections to the database. • If end users connect to the SAP HANA database directly through a SQL client, the database user is authenticated. End-user clients support several authentication mechanisms for integration into SSO environments (SAML, Kerberos, SAP logon /assertion tickets).
Authorization	SAP HANA authorization applies to users managed directly in the database.
Encryption of data communication in the network	Secure Sockets Layer (SSL) and Transport Layer Security (TLS) are supported and recommended for network communication where possible.
Encryption in the persistence layer	Data and log volume encryption ensures that anyone who can access the data and log volumes on disk using operating system commands cannot see the actual data and redo log entries.
Auditing	Actions performed in the SAP HANA database can be audited.

Related Information

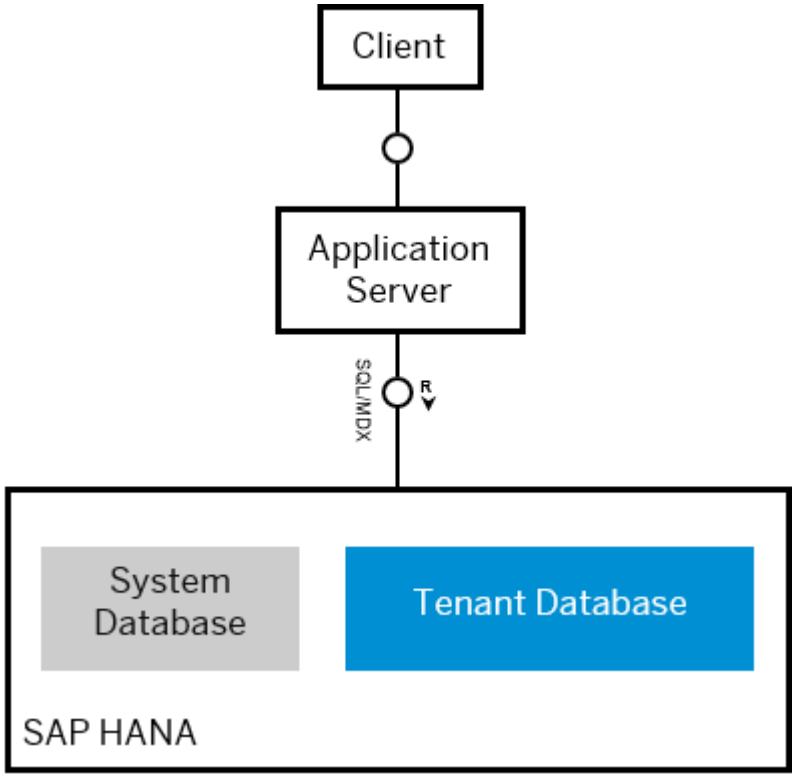
SAP HANA as Data Mart
SAP HANA User Management [page 73]
SAP HANA Authentication and Single Sign-On [page 100]
SAP HANA Authorization [page 135]
Securing Data Communication [page 40]
Data and Log Volume Encryption [page 241]
Auditing Activity in SAP HANA [page 277]

4.4.2 SAP HANA in a Classic 3-tier Architecture

SAP HANA can be used as a relational database in a classic 3-tier architecture (client, application server, and database).

This architecture is depicted in the following figure:

SAP HANA in 3-tier Architecture



In this architecture, security-related features, such as authentication, authorization, encryption, and auditing, are located and enforced primarily in the application server layer. The database is used as a data store only. Applications connect to the database using a technical user, and direct access to the database is only possible for database administrators. End users do not have direct access to either the database itself or the database server on which it's running.

As a consequence, security in the database layer is mainly focused on securing administrative access to the database. Typical examples of this architecture are the SAP S/4HANA and SAP BW. When SAP HANA is used as a database in these scenarios, the same security approach applies, and specific SAP HANA security features are mainly needed to control access of administrators to the database.

The following table outlines the relevance of SAP HANA security-related features in this implementation scenario.

SAP HANA Feature	Relevance in Scenario
User and role management	End users and roles are managed in the application server layer. For example, SAP S/4HANA applications use the user management and authentication mechanisms of the SAP NetWeaver platform, and in particular, SAP NetWeaver Application Server. User and role management in the database layer of SAP HANA is required only for technical database users and administrators.

SAP HANA Feature	Relevance in Scenario
Authentication and SSO	End-user authentication is handled in the application server layer. The relevant technical database users are used to authenticate connections to the database. Administrators with direct access to the database must be authenticated in the database. Administration clients that access the database through SQL (for example, the SAP HANA studio and the `hdsql` command line tool) support the authentication mechanisms Kerberos and SAP logon/assertion tickets for integration into SSO environments.
Authorization	SAP HANA authorization applies only to technical and administrative database users managed in the database.
Encryption of data communication in the network	Secure Sockets Layer (SSL) and Transport Layer Security (TLS) are supported and recommended for network communication where possible.
Encryption in the persistence layer	Data and log volume encryption ensures that anyone who can access the data and log volumes on disk using operating system commands cannot see the actual data and redo log entries.
Auditing	Actions performed in the SAP HANA database can be audited.

Related Information

SAP HANA User Management [page 73]
SAP HANA Authentication and Single Sign-On [page 100]
SAP HANA Authorization [page 135]
Securing Data Communication [page 40]
Data and Log Volume Encryption [page 241]
Auditing Activity in SAP HANA [page 277]

4.4.3 SAP HANA as Technical Infrastructure for Native Application Development

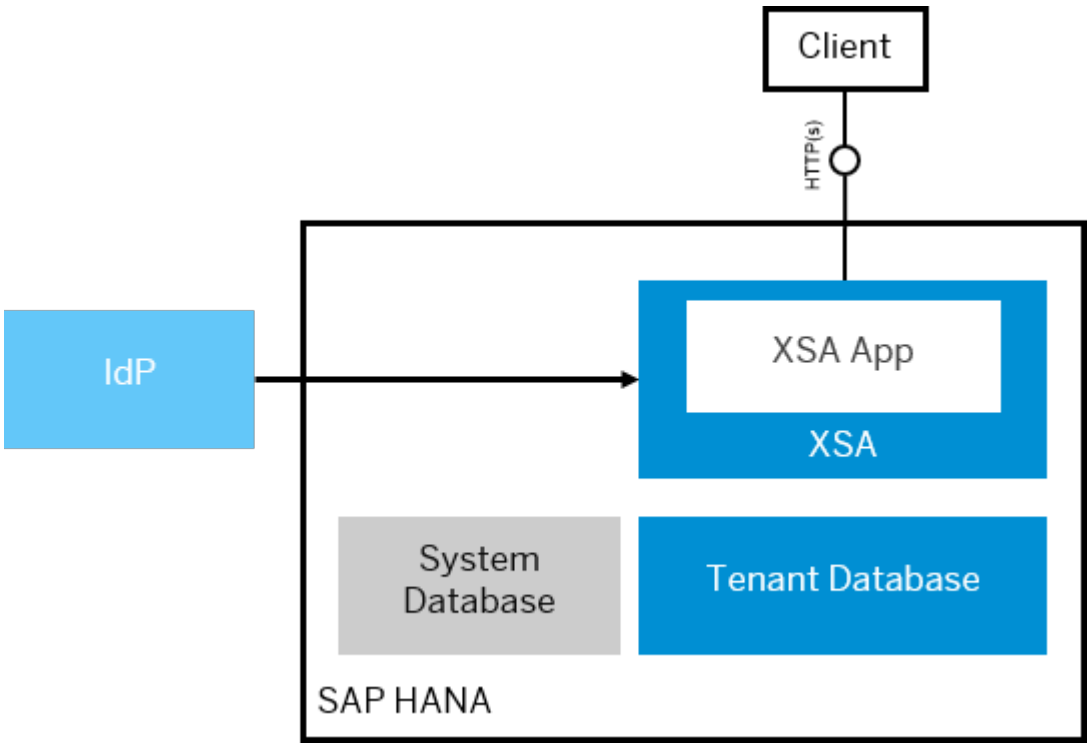
SAP HANA extended application services (SAP HANA XS), advanced model is the default framework for native application development on SAP HANA. The SAP HANA XS, classic model is the deprecated application development framework of SAP HANA.

SAP HANA XS Advanced Model

The SAP XS advanced model provides a comprehensive platform for the development and execution of native data-intensive applications. XS advanced supports a variety of programming languages, such as SQLScript for

execution on the data layer, or Java and node.js for execution in the application server runtime. End-user clients access applications developed on XS advanced via HTTP(S), while the application run-times communicate with the SAP HANA database via SQL.

The architecture of SAP HANA XS advanced (simplified) is depicted in the following figure:



SAP HANA as Technical Infrastructure for Native Applications, XS Advanced

Security and XS Advanced

XS advanced provides deployment flexibility and specific security options.

With XS advanced, the application and database layer can be decoupled. You can either install XS advanced directly on the SAP HANA server, or install it on a separate host to the SAP HANA database. This enables you to scale XS advanced independently of the database, as you can have many more XS advanced hosts than SAP HANA database hosts. You can also install XS advanced in a separate network from SAP HANA itself, which makes it possible to put XS advanced applications into a different network zone and have a firewall between the application and database layers.

XS advanced applications are strictly isolated from each other. They are deployed in dedicated containers via the SAP HANA deployment infrastructure (HDI) at the database layer. At the application layer, you can use dedicated operation system users per application.

For detailed information about the security architecture of SAP HANA XS, advanced model, see the corresponding section in this guide.

For more information about secure application development, see the section *Maintaining Application Security in XS Advanced* in the *SAP HANA Developer Guide for SAP HANA XS Advanced Model*.

> → Recommendation
>
> SAP recommends that customers and partners who want to develop new applications use SAP HANA XS advanced model. If you want to migrate existing XS classic applications to run in the new XS advanced

run-time environment, SAP recommends that you first check the features available with the installed version of XS advanced; if the XS advanced features match the requirements of the XS classic application you want to migrate, then you can start the migration process. For more information, see the *SAP HANA XS Advanced Migration Guide*.

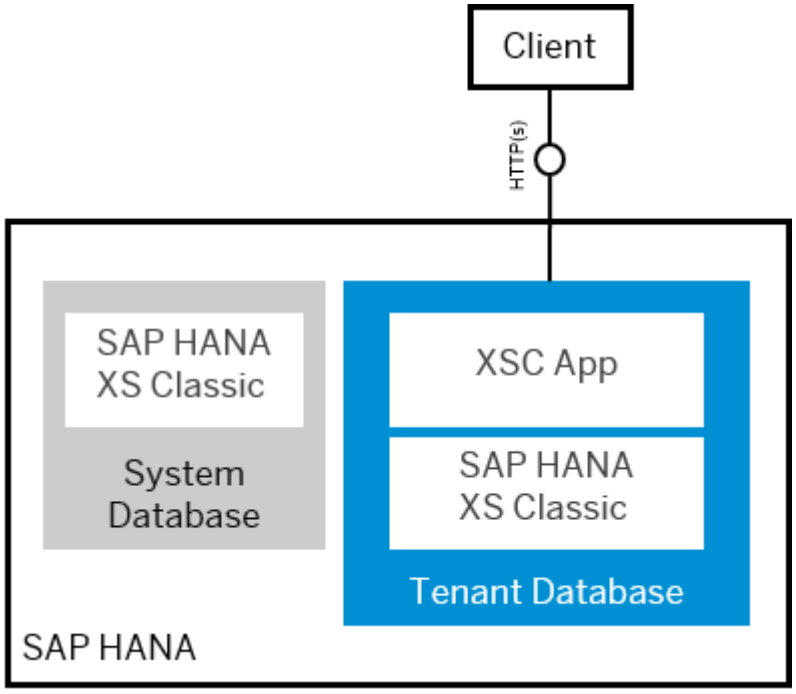
SAP HANA XS Classic Model

The SAP HANA XS classic model embeds a full-featured application server, Web server, and development environment within SAP HANA. Applications can be developed and deployed directly on SAP HANA XS, which exposes them to end users through a web interface.

> ⓘ Note
>
> SAP HANA XS classic and the SAP HANA repository are deprecated as of SAP HANA 2.0 SPS 02. For more information, see SAP Note 2465027.

The architecture of SAP HANA XS classic is depicted in the following figure:

SAP HANA as Technical Infrastructure for Native Applications, XS Classic



Classic native SAP HANA applications rely on the security-related features of SAP HANA. In particular, users of native SAP HANA applications must always have a corresponding user in the SAP HANA database.

The following table outlines the relevance of SAP HANA security-related features in this implementation scenario.

SAP HANA Feature	Relevance in Scenario
User and role management	User and roles are managed fully in SAP HANA.
Authentication and SSO	The database user is used to authenticate not only users connecting to the database through the SQL interface, but also to HTTP clients that connect to SAP HANA XS. Several mechanisms are supported for the integration of HTTP access through SAP HANA XS into SSO environments, including SAML, X.509 client certificates, Kerberos with Simple and Protected GSSAPI Negotiation Mechanism (SPNEGO), and SAP logon/assertion tickets.
Authorization	User access to native SAP HANA applications and applications functions is determined by the privileges granted to the database user.
Encryption of data communication in the network	SSL/TLS are supported and recommended for network communication where possible. The SAP Web Dispatcher can be configured to use HTTPS to secure connections between HTTP client applications and SAP HANA.
Encryption in the persistence layer	Data and log volume encryption ensures that anyone who can access the data and log volumes on disk using operating system commands cannot see the actual data and redo log entries.
Auditing	Actions performed in the SAP HANA database can be audited.

Secure Application Development

For more security information about the following aspects related to SAP HANA XS application development, see the section on *Maintaining Application Security* in the *SAP HANA Developer Guide for SAP HANA XS Classic Model*.

Related Information

Security for SAP HANA Extended Application Services, Advanced Model [page 336]
SAP HANA User Management [page 73]
SAP HANA Authentication and Single Sign-On [page 100]
SAP HANA Authorization [page 135]
Securing Data Communication [page 40]
Data and Log Volume Encryption [page 241]
Auditing Activity in SAP HANA [page 277]
Maintaining Application Security in XS Advanced
Maintaining Application Security
SAP HANA XS Advanced Migration Guide
SAP Note 2465027

5 SAP HANA Network and Communication Security

Several mechanisms are possible for securing network communication in the SAP HANA landscape.

SAP HANA supports encrypted communication for network communication channels. We recommend using encrypted channels in all cases where your network isn't protected by other security measures against attacks such as eavesdropping, for example, when your network is accessed from public networks. Alternatively, use virtual private network (VPN) tunnels to transfer encrypted information.

For more information about network administration, see the *SAP HANA Administration Guide*.

Related Information

Network Administration

5.1 Communication Channels

The network communication channels used by SAP HANA can be categorized into those used for database clients connecting to SAP HANA and those used for internal database communication. SAP recommends using encrypted communication channels where possible.

The following is an overview of the network communication channels used by SAP HANA.

To support the different SAP HANA scenarios and set-ups, SAP HANA has different types of network communication channels:

- Channels used for external access to SAP HANA functionality by end-user clients, administration clients, application servers, and for data provisioning through SQL or HTTP
- Channels used for SAP HANA internal communication within the database, between hosts in multiple-host systems, and between systems in system-replication scenarios

> **Note**
>
> SAP HANA internal communication has sometimes been unofficially referred to as TREXNet communication. However, the term TREXNet is not valid in the context of SAP HANA.

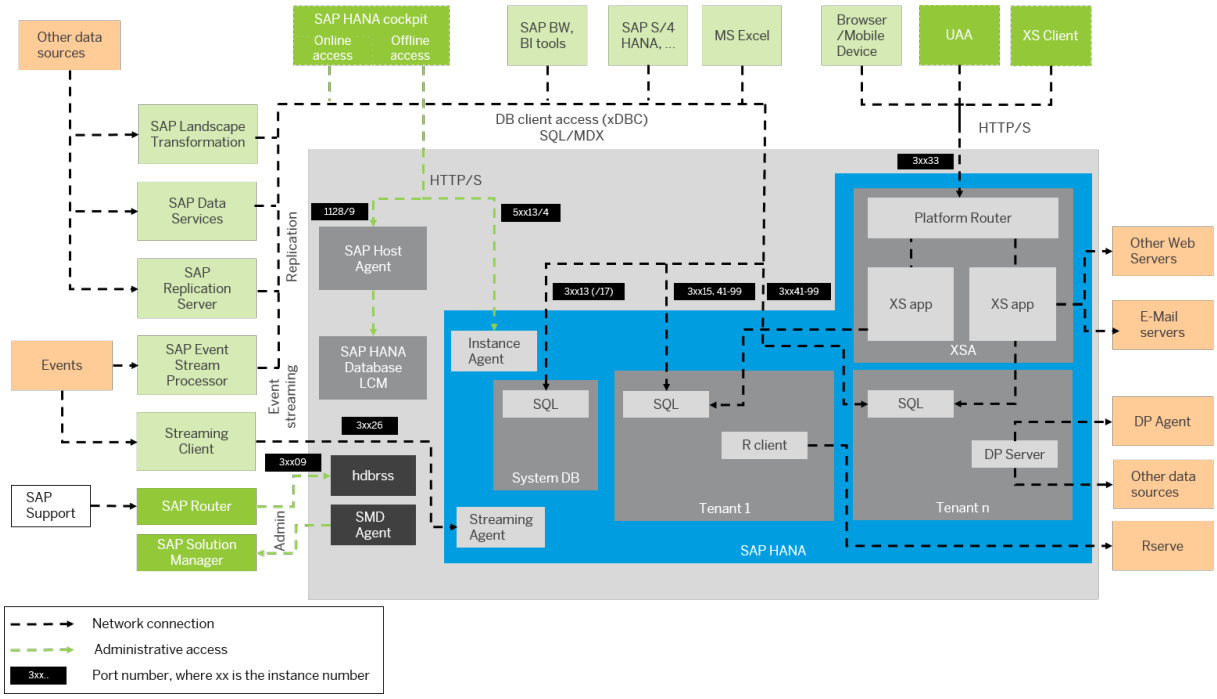
The connections between SAP HANA and external components and applications come under these categories:

- Connections for administrative purposes
- Connections for data provisioning
- Connections from database clients that access the SQL/MDX interface of the SAP HANA database
- Connections from HTTP/S clients

- Outbound connections

You can see an example of what these connections look like in the figure below. Network connections are depicted by dotted arrows. The direction of each arrow indicates which component is the initiator and which component is the listener. Administrative access to and from SAP HANA is depicted by the green dashed arrows. Port numbers are shown with a black background. The "xx" in the port numbers stands for the number of your SAP HANA instance.

The figure below shows all the network channels used by SAP HANA. For the purposes of illustration, a single-host installation with two tenant databases is depicted. However, the connections shown apply equally to a distributed scenario.



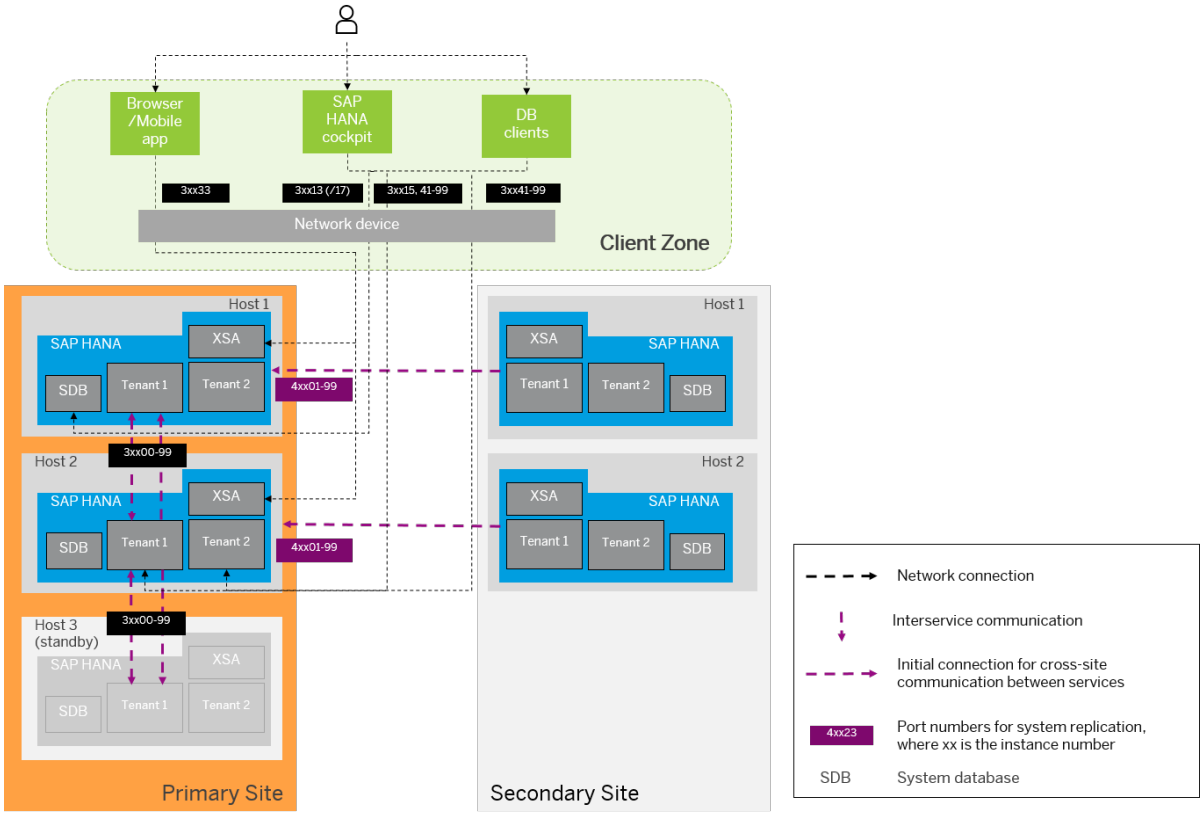
Connections Between SAP HANA and External Components

> ### Note
>
> Some components depicted in the figure are supported on Intel-based hardware platforms only (for example, SAP HANA Streaming Analytics). Refer to the Product Availability Matrix (PAM).

In addition, the different components of SAP HANA, as well as the hosts in a distributed scenario, communicate with each other over internal SAP HANA connections. These connections are also used in system replication scenarios for communication between a primary site and secondary site(s) to ensure high availability in the event of a data center failure.

The following figure shows an example of a distributed SAP HANA system with two active hosts and an extra standby host, both fully replicated to a secondary site to provide full disaster recovery support.

SAP HANA Internal Connections



Related Information

Securing Data Communication [page 40]
Network Administration
Product Availability Matrix

5.2 Network Security

To integrate SAP HANA securely into your network environment, several general recommendations apply.

The components of an SAP HANA landscape communicate through different network communication channels. It is recommended security practice to have a well-defined network topology to control and limit network access to SAP HANA to only those communication channels needed for your scenario, and to apply appropriate additional security measures, such as encryption, where necessary. This can be achieved through different means, such as separate network zones and network firewalls, and through the configuration options provided by SAP HANA (for example, encryption). The exact setup depends on your environment, your implementation scenario, and your security requirements and policies.

The detailed network set-up and recommendations are described in network administration section of the *SAP HANA Administration Guide*. This section contains some additional security-relevant information.

> ⚠ Caution
>
> It is strongly recommended that you apply the measures described in this section to protect access to the SAP HANA database's internal communication channels and to mitigate the risk of unauthorized access to these services.

Network Zones

- We recommend that you operate the different components of the SAP HANA platform in separate network zones.
  To prevent unauthorized access to the SAP HANA environment and the SAP HANA database through the network, use network firewall technology to create network zones for the different components and to restrictively filter the traffic between these zones implementing a "minimum required communication" approach. The relevant network zones depend on your specific application scenario and your network infrastructure. For more information about network zones, see the *SAP HANA Administration Guide*.
- We recommend that you operate SAP HANA in a protected data-center environment.Allow only dedicated authorized network traffic from other network zones (for example, user access from the client network zone) to follow these rules:
  - Clients accessing external standard database functionality, for example by SQL, have only access to the database client access port.
  - Clients (for example, browser applications) accessing the SAP HANA environment through the HTTP access feature of SAP HANA Extended Application Services, classic model (XS classic), for example SAP HANA UI Toolkit for Info Access, have only access to the SAP HANA XS ports.
  - Some administrative functions (for example, starting and stopping the SAP HANA instance) have access to the administrative ports.
  - XS classic exposes some administrative applications (for example, administration of Security Assertion Markup Language (SAML) for user authentication). We recommend using URL filtering (for example, reverse proxy) to control the exposure of different applications to different network zones.

Internal Communication

Database internal communication channels are only used for the following:

- Communication within the database
- Communication between hosts in distributed (multiple-host) scenarios
- Communication between multiple sites in system replication (high-availability) scenarios
- Communication between the SAP HANA database and server components, such as extended storage (SAP HANA dynamic tiering)

Note the following network security considerations for single-host, multiple-host, and system replication (high-availability) scenarios.

Single-Host Scenario

In a single-host scenario, access to the network ports for database internal communication from other network hosts is blocked by default. We recommend that you do not change this setting. The internal communication ports are bound to `localhost`.

> **Note**
>
> In single-host scenarios, the same communication channels are used for communication between the different processes on a single host, and the internal IP addresses/ports are by default bound to the `localhost` interface. Note that this does not apply to dynamic tiering. The dynamic tiering service is bound to all interfaces, although the internal communication between SAP HANA database and dynamic tiering uses the `localhost` interface.

Multiple-Host Scenario

In a distributed scenario (that is, one instance of SAP HANA on multiple hosts), internal network communication takes place between the hosts at one site via internal. Certified SAP HANA hosts contain either dedicated or virtualized network interfaces that are configured as part of a private network using separate IP addresses and ports.

We recommend operating all hosts in a dedicated sub-network.

To prevent unauthorized access to the database via the internal communication channels in distributed systems, we recommend that you prevent access to these network channels and ports from outside the system. There are a number of ways to isolate internal network ports from the client network:

- Using the SAP HANA configuration option to route communication between the hosts of a distributed environment onto a specified network and binding those internal network services exclusively to the network interface (**recommended option**)
  For more information about configuring inter-service communication, see the *SAP HANA Administration Guide*.

  > **Note**
  >
  > In system replication scenarios, you can use this feature in the presence of a secondary site. However, note that additional ports used for communication between primary and secondary sites are opened on the network interface. These ports need to be protected.
- Using operating system commands (for example, `iptables`), and/or network device configuration
- Using network firewall functions to block access to internal ports in specific network zones

If your setup does not permit isolating internal network communication, consider using encryption to protect the internal communication. For more information, see the section on securing internal communication.

System Replication Scenario

In a system replication scenario, you can protect the channels used in the following ways:

- Configuring SAP HANA to use exclusively a separate network dedicated to system replication for communication between primary and secondary site
- Configuring secure communication using the TLS/SSL protocol for encryption and mutual authentication between sites
- Specifying the IP addresses allowed to connect to system replication ports

Additional Measures for Securing Internal Communication

We recommend that you protect internal communication further by applying additional mechanisms. This may include filtering access to the relevant ports and channels by firewalls, implementing network separation, or applying additional protection at the network level (for example, VPN, IPSec).

We recommend routing the connection between the sites over a special site-to-site high-speed network, which typically already implements security measures such as separation from other network access and encryption or authentication between sites. The details of security measures and additional network security measures needed will depend on your specific environment. For more information about network administration, see the *SAP HANA Administration Guide*

SAP HANA Extended Application Services, Advanced Model

Security mechanisms are applied to protect the communication paths used by the SAP HANA XS advanced server infrastructure. SAP provides network topology recommendations to restrict access at the network level. For more information, see the section on SAP HANA XS advanced security.

Data Replication Technologies

Additional network configurations may be required depending on the implemented data replication technology. For more information, see the section on security for SAP HANA replication technologies.

Related Information

Secure Internal Communication [page 64]
Network Administration
Network Zones
Host Name Resolution for System Replication
Configuring SAP HANA Inter-Service Communication
Security for SAP HANA Extended Application Services, Advanced Model [page 336]
Security for SAP HANA Replication Technologies [page 333]

5.3 Securing Data Communication

SAP HANA supports encrypted communication for client-server (external) communication and internal communication.

Communication Channels That Can Be Secured

Communication between the following components can be secured using the Transport Layer Security (TLS)/ Secure Sockets Layer (SSL) protocol.

> **Note**
>
> An alert occurs if TLS/SSL is not configured. To confirm TLS/SSL configuration see column PURPOSE of the PSES System View.

External Channels

- SAP HANA and clients via ODBC or JDBC connections, including the SAP HANA cockpit and SAP HANA studio
- SAP HANA XS classic/advanced server and clients via HTTP

  > **Note**
  >
  > For JDBC and ODBC client connections, user passwords are always transmitted in encrypted hashed form during the user authentication process, never in plain text. For HTTP connections via SAP HANA XS classic, HTTPS must be configured. In SSO environments, we recommend using encrypted communication channels for **all** client connections.
- SAP HANA lifecycle manager and the SAP HANA cockpit and SAP HANA studio
- SAP start service (`sapstartsrv`) and the SAP HANA cockpit and SAP HANA studio
- SAP HANA lifecycle manager and SAP Service Marketplace
- SAP HANA cockpit and SAP Host Agent
- SAP HANA lifecycle manager and SAP Host Agent
- SAP HANA and an LDAP directory server
- SAP HANA and the Rserve server
- SAP HANA and data providers

Internal Channels

- Internal database communication
- Internal communication between hosts in a distributed (multiple-host) SAP HANA system
- Internal communication between systems at the different sites in a system replication (high availability) scenario
- Internal communication between the SAP HANA XS advanced server and the SAP HANA database
- Internal communication between the SAP HANA database and server components, such as extended storage (SAP HANA dynamic tiering).

Trust and Key Stores for Securing Communication

Different trust and key stores exist for internal communication and external communication. Depending on your implementation, these will either be in the form of:

- In-database **certificate collections** (recommended)

  > ⓘ Note
  >
  > Only supported for external communication channels.

- Personal security environments (**PSEs**) stored in the file system

A certificate collection (or PSE) is a secure location where the public information (public-key certificates) and private information (private keys) of the SAP HANA server are stored. A certificate collection may also contain the public information (public-key certificates) of trusted communication partners or root certificates from trusted Certification Authorities. Certificate collections can be created and managed as database objects directly in the SAP HANA database.

> ⚠ Caution
>
> We recommend creating certificate collections for individual purposes in the database directly, rather than using trust stores (PSE) in the file system. By default, the same PSE in the file system is shared by all databases for all external communication channels (including HTTP) and certificate-based authentication. Different PSEs must be explicitly configured for tenant databases.

Internal Communication

The keys and certificates in the certificate collection for internal communication are used to secure the following:

- Communication between database services
- Communication between hosts in a multiple-host system
- Communication between hosts and sites in a system replication scenario
- Communication between the SAP HANA database and additional server components, such as an extended storage server (SAP HANA dynamic tiering) or a streaming analytics server (SAP HANA Streaming Analytics).

External Communication

Certificates used for external communication (for example, JDBC and HTTP client access) are typically signed by an externally available Certification Authority (CA) because the CA certificates need to be integrated in the relevant clients.

For information about certificate handling, see the section on certificate management.

Related Information

External Communication

Secure Communication Between SAP HANA and JDBC/ODBC Clients [page 42]
Secure Communication Between SAP HANA and an LDAP Directory Server [page 59]

Secure Communication Between SAP HANA XS Classic and HTTP Clients [page 62]
Security Aspects of SAP HANA Platform Lifecycle Management [page 329]
Security Aspects of SAP HANA R Integration [page 332]
Security for SAP HANA Replication Technologies [page 333]

Internal Communication

Secure Internal Communication [page 64]
Secure Internal Communication Between Sites in System Replication Scenarios [page 68]
Database Isolation [page 23]

Certificate Collections/PSEs

Certificate Management in SAP HANA [page 303]
PSES System View

5.3.1 Secure Communication Between SAP HANA and JDBC/ ODBC Clients

You can use the Transport Layer Security (TLS)/Secure Sockets Layer (SSL) protocol to secure communication between the SAP HANA database and clients that access the SQL interface of the database.

> **Note**
>
> For SAP HANA 2.0 SPS 08 and later servers, TLS 1.3 is necessary for secure connections. The client versions that support TLS 1.3 include:
>
> - SQLDBC
> - Version 2.18.22 and later with CommonCryptoLib 8.5.48 and later
> - Version 2.18.22 and later with OpenSSL 1.1.1 and later
> - Version 2.23.24 and later with MSCrypto for Windows 11 / Windows Server 2022
> - JDBC
> - Version 2.18.2 and later with a JDK 8 version released on or after July 2022 (SAP JVM 8.1.089 or Oracle JDK 8u341) or JDK 11 and later

By default, enabling TLS/SSL for client-server communication provides the following:

- Server certificate validation (server authentication)
  The database identifies itself to the client when the connection is established. This reduces the risk of man-in-the-middle attacks and fake servers gaining information from clients.
- Data encryption
  In addition to server authentication, the data being transferred between the client and the server is encrypted, which provides integrity and privacy protection. An eavesdropper cannot access or manipulate the data.

TLS/SSL must be configured on both the SAP HANA database (server) and the client. During installation, the server-side configuration is completed automatically through the creation of a dedicated public key infrastructure for external communication (client PKI).

To validate the identity of the client connecting to SAP HANA, you can also enable client certificate validation (client authentication).

> **Note**
>
> The client PKI can only be used for server authentication.

> **→ Remember**
>
> Secure communication between the SAP HANA database and HTTP clients (HTTPS) via the SAP HANA XS server classic must be configured separately. For more information, see *Secure Communication Between SAP HANA XS Classic and HTTP Clients*.

Enforced TLS/SSL for Client Connections

To force all clients communicating with the SAP HANA database through the SQL interface to use a secured connection, you can set the parameter `sslEnforce` in the `communication` section of the `global.ini` configuration file to `true`. The database subsequently refuses SQL connection attempts that do not use SSL.

> **Note**
>
> Communication properties are in the default configuration change blocklist (`multidb.ini`). This means that they cannot initially be changed in tenant databases. They must be changed from the system database. If appropriate for your scenario, you can remove these properties from the change blocklist. SAP HANA deployment scenarios are described in the *SAP HANA Master Guide*. For more information about how to edit the change blocklist, see the *SAP HANA Administration Guide*.

Related Information

Default Blacklisted System Properties in Tenant Databases [page 417]
Prevent Changes to System Properties in Tenant Databases
Configure HTTPS (SSL) for Client Application Access
Secure Communication Between SAP HANA XS Classic and HTTP Clients [page 62]
Mutual Authentication (SAP HANA Platform)

5.3.1.1 TLS/SSL Configuration on the SAP HANA Server

To secure communication between the SAP HANA database and clients that access the SQL interface of the database, TLS/SSL must be configured on both server and client side. Automatic configuration on the server is supported with the client public key infrastructure (PKI).

> **Note**
>
> SAP HANA's automatic TLS certificate configuration (Client PKI) includes ready-to-use TLS support with an instance-specific certificate authority. During installation, SAP HANA automatically creates self-signed server certificates using this internal certificate authority. For production use, you should replace these

default certificates with certificates signed by your organization's trusted Certificate Authority. For more information, see *Managing Client Certificates and Public Keys*.

Automatic Configuration with Client PKI

A dedicated PKI for external communication is automatically created and enabled during system installation. Every host on which a database server and optional component server is running, as well as every tenant database in the system, are integrated into this PKI.

The client PKI comprises a set of tenant-specific certificate authorities (CAs) and host-specific X.509 certificates signed by these CAs, as well as private keys, all of which are stored in database in certificate collections. All certificates use strong encryption and signature algorithms, that is SHA-256 with RSA and a key length of 4096 bits.

Host Certificates

Host certificates are used to prove the server's authenticity. The host certificates include all known host names of the SAP HANA servers in the subject alternative names (SAN) field.

The host certificates of a database (`_SYS_CLIENTPKI_HOST_CERT`) are stored in the database certificate store and assigned to the certificate collection `_SYS_CLIENTPKI` with purpose `SSL`.

> **Note**
>
> As long as the client PKI is enabled, it is not possible to set the purpose `SSL` for any other certificate collection.

Host certificates are short lived (180 days). They are automatically renewed 32 days before expiry, after a restart, and after a host has been added or removed. You can also generate new host certificates using the statement `ALTER SYSTEM CLIENTPKI UPDATE CERTIFICATES`.

Certificate Authorities (CA)

Host certificates are signed by the tenant-specific CA.

The root CA certificate of the database must be provided to the client to establish a secure connection. Since the client PKI is enabled by default, to obtain the root CA certificate, you must log on to the server from a client that does not (initially) require server certificate validation, if TLS/SLS is used to connect. Alternatively, you can log on the local host.

The root CA certificate of a database (`_SYS_CLIENTPKI_ROOT_CERT`) exists in the database certificate store and is assigned to the certificate collections `_SYS_CLIENTPKI` and `_SYS_CLIENTPKI_ROOT_CA`. Since the database must be online to export a root CA certificate from the certificate store, the root CA certificate is also available on the file system in PEM format at $SECUDIR (`clientpki_<dbname>.cer`).

Root CA certificates are long lived (10 years). You can generate a new root CA certificate, as well as new host certificates signed with the new root CA certificate for every host name, using the statement `ALTER SYSTEM CLIENTPKI UPDATE ROOT CA`.

For more information about providing the root CA certificate to the client, see *Server Certificate Authentication* in the *SAP HANA Client Interface Programming Reference*.

Replacing Default Server Certificates with Trusted Certificates

SAP HANA automatically creates self-signed server certificates during installation using an internal certificate authority. For production use, you should replace these default certificates with certificates signed by your organization's trusted Certificate Authority.

Before you can replace the default certificates, you must generate appropriate X.509 certificates using third-party certificate management tools. The certificate generation process varies depending on whether you're working with database-stored certificates (managed through the SAP HANA cockpit) or file system PSEs (stored in personal security environments).

For more information on generating X.509 certificates for PSEs, see SAP Note 1718944.

Once you have obtained the appropriate certificates from your trusted Certificate Authority, you can use the SAP HANA cockpit to import and configure these certificates to replace the default self-signed certificates. The cockpit provides a complete workflow for managing certificates stored in the database.

For detailed instructions on certificate management procedures, see *Managing Client Certificates and Public Keys*.

Enabling the Client PKI After an Update

The client PKI is not enabled after an update. To enable the client PKI and trigger the creation of the associated certificates, private keys and certificate collections, set the parameter `[communication] sslclientpki` in the `global.ini` configuration file to ON.

> ⚠ Caution
>
> If there is already a certificate collection in the database with the purpose `SSL` when you enable the client PKI, the purpose is removed from this collection and set for `_SYS_CLIENTPKI`.

Disabling the Client PKI

If you do not want to use the client PKI, you can disable it by setting the parameter `[communication] sslclientpki` in the `global.ini` configuration file to OFF.

This results in the deletion of all host certificates and all client PKI certificate collections (`_SYS_CLIENTPKI`, `_SYS_CLIENTPKI_ROOT_CA`). The root CA certificate, however, is not dropped. If this long-living certificate has already been distributed to clients, then it will not be necessary to distribute a new one if the client PKI is re-enabled in the future. The root CA certificate can be dropped by executing `ALTER SYSTEM CLIENTPKI DROP ROOT CA`.

> ⓘ Note
>
> You must disable the client PKI and manually configure TLS/SSL in the following cases:
>
> - You require mutual authentication (server and client authentication). The client PKI can only be used for server authentication.
> - You plan to configure system replication for the SAP HANA system.

Manual Configuration

If you want to set up and manage your own PKI for client-server communication, you can manually configure the server for TLS/SSL. The following general prerequisites must be met:

- The client PKI is disabled.
  The parameter `[communication] sslclientpki` in the `global.ini` configuration file must be OFF. In existing installations, this is automatically the case after an update. In a new installation, you must set the parameter to OFF manually.
- The SAP Cryptographic Library CommonCryptoLib is available in the SAP HANA system.
  CommonCryptoLib (`libsapcrypto.so`) is installed by default as part of SAP HANA installation at $DIR_EXECUTABLE.

  > ⓘ Note
  >
  > If you are using trust and key stores located in the file system instead of in the database, OpenSSL is also supported and installed by default as part of the operating system installation. However, OpenSSL is deprecated so you must migrate to CommonCryptoLib. For more information, see SAP Note 2093286.

- The SAP HANA database possesses a public and private key pair, and a public-key certificate.
  All databases (system database and tenant databases) can have their own key pair and public key certificate. In distributed SAP HANA systems, every host must have its own key pair and public key certificate.
  You can use the tools provided with OpenSSL to create server certificates. If you are using CommonCryptoLib, you can also use the SAP Web Dispatcher administration tool or the SAPGENPSE tool, both of which are delivered with SAP HANA. For more information about the SAP Web Dispatcher administration tool, see SAP Note 2009483.

  > ⓘ Note
  >
  > Unless you are using SAPGENPSE, do not password protect the keystore file that contains the server's private key. When using the SAP Web Dispatcher administration tool to create a personal security environment (PSE) for the server, do not specify a PIN. With SAPGENPSE, you can also set a PIN later using the `seclogin` command as follows:
  >
  > ```
  > sapgenpse seclogin -p <PSE path and file name> -x <PIN> -O <sid>adm
  > ```
  >
  > The PIN is stored in the credentials file `cred_v2`in the $SECUDIR directory.

  > ⚠ Caution
  >
  > If your server's keys are compromised, you must replace the key and the certificate.

Configuration

The properties for configuring TLS/SSL on the server for external communication are available in the `communication` section of the `global.ini` configuration file. In general, it's not necessary to configure any of these properties explicitly. The default configuration can be used.

> ⓘ Note
>
> Communication properties are in the default configuration change blocklist (`multidb.ini`). This means that they cannot initially be changed in tenant databases. They must be changed from the system

database. If appropriate for your scenario, you can remove these properties from the change blocklist. SAP HANA deployment scenarios are described in the *SAP HANA Master Guide*. For more information about how to edit the change blocklist, see the *SAP HANA Administration Guide*.

However, you do need to create a certificate collection. You do this as follows:

1. Create a certificate collection in the database.
   In multiple-host (distributed) systems, you can create a certificate collection for each host.
2. Add the server's public key certificate(s) and private key(s).
3. Add the public key certificates of trusted communication partners.
4. Set the purpose SSL for the certificate collection.
   In multiple-host systems, you can assign the purpose SSL to the certificate collection for each host.

→ Tip

For connections to **tenant databases**, certificate validation may not work due to how SAP HANA handles host name resolution. If this is the case, change the value of the parameter `[public_hostname_resolution] use_default_route` in the `global.ini` file on the SYSTEM layer from **`ip`** to **`fqdn`**. SAP HANA then uses the fully qualified domain name and certificate validation for secured JDBC/ODBC connections is allowed.

File System-Based Trust and Key Store

While we recommend using certificate collections that exist in the database to store certificates and keys, it is possible to use stores located in the file system and configured in the `global.ini` file.

If files located in the file system are being used, they are shared by default. It is still possible to configure different trust and key stores for tenant databases for every database in the `global.ini` file. However, bear the following points in mind:

- If different trust and key stores are not explicitly configured for tenant databases, the same ones will be used for all external communication channels (including HTTP) for all databases.

  ⚠ Caution

  If you have configured in tenant databases or the system database single sign-on mechanisms that rely on trust stores located in the file system (such as SAP logon and assertion tickets or SAML) and the trust stores are shared, users of one tenant database may be able to log on to other databases in the system.

- Only the system administrator can configure separate trust and key stores for tenant databases by changing the relevant properties in the `global.ini` file. This is because tenant database administrators are prevented from changing any communication properties. They are in the default configuration change blacklist (`multidb.ini`).

For more information about certificate collections in the database and PSEs in the file system, see the section on certificate management.

Related Information

ALTER SYSTEM CLIENTPKI { UPDATE / DROP } (System Management)

Server Certificate Authentication (SAP HANA Client Interface Programming Reference)
Server-Side TLS/SSL Configuration Properties for External Communication (JDBC/ODBC) [page 48]
Default Blacklisted System Properties in Tenant Databases [page 417]
Prevent Changes to System Properties in Tenant Databases
Certificate Management in SAP HANA [page 303]
Mapping Host Names for Database Client Access

Related SAP Notes

SAP Note 2093286
SAP Note 1718944
SAP Note 1848999
SAP Note 2009483
SAP Note 2009878
Managing Client Certificates and Public Keys

5.3.1.2 Server-Side TLS/SSL Configuration Properties for External Communication (JDBC/ODBC)

The parameters for configuring TLS/SSL for external communication on the SAP HANA server are available in the `communication` section of the `global.ini` configuration file.

The following table lists the configuration properties that can be used to configure TLS/SSL on the server. In general, it is not necessary to configure any of the parameters explicitly. The default configuration can be used.

> **Note**
>
> Communication properties are in the default configuration change blocklist (`multidb.ini`). This means that they cannot initially be changed in tenant databases. They must be changed from the system database. If appropriate for your scenario, you can remove these properties from the change blocklist. SAP HANA deployment scenarios are described in the *SAP HANA Master Guide*. For more information about how to edit the change blocklist, see the *SAP HANA Administration Guide*.

Parameter	Value	Default	Description
`sslMinProtocolVersion`	{SSL30,TLS10,TLS11,TLS12}	TLS12	The minimum available TLS/SSL protocol version **Note** `sslMinProtocolVersion` must be less than or equal to `sslMaxProtocolVersion`.
`sslMaxProtocolVersion`	{TLS10,TLS11,TLS12, TLS13, MAX}	TLS13	The maximum available TLS/SSL protocol version

Parameter	Value	Default	Description
`sslValidateCertificate`	`<Boolean value>`	false	If set to **`true`**, the certificate of the communication partner is validated.
`sslCreateSelfSignedCertificate`	`<Boolean value>`	false	If set to **`true`**, a self-signed certificate is created if the keystore cannot be found.
`sslBlindCAResponse`	`<Boolean value>`	on	If set to **`on`**, a client may send a certificate in response to a client certificate request from the server even if it contains an empty Certificate Authority list If set to **`off`**, the client cannot respond to such a certificate request; the connection is refused.
`sslclientpki`	`<Boolean value>`	on (after installation), off (after upgrade)	Availability of an automatically generated public key infrastructure (PKI) to secure external communication See *TLS/SSL Configuration on the SAP HANA Server*.
`sslEnforce`	`<Boolean value>`	off	If set to on, only TLS-secured connections from clients will be accepted. If set to off, both TLS-secured and insecure connections will be accepted.

Additionally, the parameter `sslCipherSuites` can be used to specify the encryption algorithms available for TLS/SSL connections. Its value depends on the cryptographic service provider used. The default values are **`PFS:HIGH::EC_HIGH:+EC_OPT`** (CommonCryptoLib) and **`ALL:!ADH:!LOW:!EXP:!NULL:@STRENGTH`** (OpenSSL*). For more information, see the documentation of the cryptographic library.

Parameters for Configuring Trust and Key Stores in the File System

The following parameters are used to configure trust and key stores located in the file system. In general, it's not necessary to configure a cryptographic provider nor any of the parameters explicitly. The default configuration can be used.

> ⚠ Caution
>
> We recommend creating certificate collections for individual purposes in the database directly, rather than using trust stores (PSE) in the file system. By default, the same PSE in the file system is shared by all databases for all external communication channels (including HTTP) and certificate-based authentication. Different PSEs must be explicitly configured for tenant databases.

In general, if certificate collections with the purpose *SAML*, *X509*, *JWT*, or *SSL/TLS* exist in the database, the parameters below are ignored. If you explicitly want the in-database collection for one of these purposes to be ignored, configure the parameter `skip_in_memory_pse_store_for_purposes`.

Parameter	Value	Default	Description
`sslCryptoProvider`	{commoncrypto | sapcrypto | openssl}	1. commoncrypto 2. openssl*	Cryptographic provider used for TLS/SSL connection **Note** If you specify a value for this parameter, you must also explicitly specify paths in both the `sslKeyStore` and `sslTrustStore` parameters to avoid configuration issues.
`sslKeyStore`	`file`	• `$SECUDIR/sapsrv.pse` (CommonCryptoLib) • `$HOME/.ssl/key.pem` (OpenSSL*)	Path to the keystore file that contains the server's private key You must specify an absolute path to the keystore file if using OpenSSL*. **Note** If you specify a value for this parameter, you must also explicitly specify a cryptographic provider in the `sslCryptoProvider` parameter to avoid configuration issues.

Parameter	Value	Default	Description
`sslTrustStore`	`file`	• `$SECUDIR/sapsrv.pse` (CommonCryptoLib) • `$HOME/.ssl/trust.pem (OpenSSL*)`	Path to trust store file that contains the server's public certificate You must specify an absolute path to the keystore file if using OpenSSL*. **Note** If you specify a value for this parameter, you must also explicitly specify a cryptographic provider in the `sslCryptoProvider` parameter.
`skip_in_memory_pse_store_for_purposes`	{SSL,SAML,SAP LOGON, X509, JWT}	empty	Purposes that exclusively use a trust and key store (PSE) located in the file system. For each purpose listed, the in-memory certificate collection is skipped and a trust and key store in the file system is used instead. For purposes SSL, SAML, X509, and JWT the trust and key store specified in `sslTrustStore` and `sslKeyStore` are used instead. **Note** The PSE for SAP logon and assertion tickets can be specified with the parameter `[authentication] saplogontickettruststore` in the `indexserver.ini` file (default `saplogon.pse`).

Note

*OpenSSL is deprecated. If you are using OpenSSL, migrate to CommonCryptoLib.

For more information, see SAP Note 2093286 (Migration from OpenSSL to CommonCryptoLib (SAPCrypto)).

Related Information

Default Blacklisted System Properties in Tenant Databases [page 417]
Prevent Changes to System Properties in Tenant Databases
Certificate Management in SAP HANA [page 303]
Single Sign-On Using SAP Logon and Assertion Tickets [page 122]
SAP Note 2093286

5.3.1.3 TLS/SSL Configuration on the Client

You can use TLS/SSL to secure communication between the SAP HANA database and clients that access the SQL interface of the database. TLS/SSL must be configured on both the server and the client.

Configuring a Secure Connection

The client-side configuration required to secure client-to-server communication depends on whether the client communicates with the server via an SQLDBC-based or a JDBC-based connection.

SQLDBC Connections

For SQLDBC-based connections, the configuration properties and their names are the same as the server parameters. The `ENCRYPT` property is used to initiate an TLS/SSL-secured connection. You set the properties according to the client operating system.

> **Note**
>
> The connection parameters for ODBC-based connections can also be used to configure TLS/SSL for connections from ABAP applications to SAP HANA using the SAP Database Shared Library (DBSL). To pass the connection parameters to the DBSL, use the following profile parameter:
>
> ```
> dbs/hdb/connect_property = param1, param2, ...., paramN
> ```
>
> The connection parameters are used for both the primary ABAP connection and secondary connections.

JDBC Connections

For clients connecting via the JDBC interface, TLS/SSL is configured at the Java virtual machine (JVM) level using system properties. There are several ways of configuring these properties. For more information, see the Java Platform documentation.

SAP HANA cockpit and database explorer

Connections from the SAP HANA cockpit and SAP HANA database explorer use JDBC. Secure communication can be enabled and configured when you register the database as a resource in the cockpit or database explorer. The server certificate or root certificate must be imported as trusted certificates into the cockpit or database explorer.

Related Information

SAP HANA Client Interface Programming Reference
Implement Mutual Authentication (SAP HANA Platform)
Security Considerations for SAP HANA Cockpit
Security in the SAP HANA Database Explorer

5.3.1.4 Client-Side TLS/SSL Connection Properties (ODBC)

For ODBC-based connections, the configuration properties and their names are the same as the server parameters with the addition of the `encrypt` property, which initiates a TLS/SSL-secured connection.

The following table lists the configuration properties that are used to configure SSL for ODBC client access.

For more information about other ODBC connection properties and how to set them, see the *SAP HANA Client Interfaces Programming Reference*.

Property	Value	Default	Description
`{ cseKeyStorePassword | clientside_encryption_keystore_password }`	`<string>`		Provides the password for the local key store. This option is required when using client-side encryption. Passwords must be at least eight characters and have at least one upper case character, one lower case character, and one number.
`encrypt`	Boolean	FALSE	Enables or disables TLS/SSL encryption. The server chooses the highest available.

Property	Value	Default	Description
`sslCryptoProvider`	{ commoncrypto | openssl | mscrypto }	1. commoncrypto 2. openssl/mscrypto	Specifies the cryptographic library provider used for TLS/SSL communication. If you specify a value for this property, then you must also explicitly specify paths in both the `sslKeyStore` and `sslTrustStore` properties to avoid configuration issues. If CommonCryptoLib is not available, OpenSSL is used by default in Linux environments, and msCrypto in Microsoft Windows environments. OpenSSL is not available on Microsoft Windows. **Note** Check the `client` folder of the installation package to see if `COMMONCRYPTOLIB.TGZ` is present. If not, you can download CommonCryptoLib separately. For instructions on how to download CommonCryptoLib, see "Download and Install SAP Common Crypto Library" in the *SAP HANA Client Installation and Update Guide*.
`sslHostNameInCertificate`	`<string>`	Empty	Specifies the host name used to verify server's identity. The host name specified here verifies the identity of the server instead of the host name with which the connection was established. For example, in a single-host system, if a connection is established from a client on the same host as the server, then a mismatch would arise between the host named in the certificate (actual host name) and the host used to establish the connection (localhost). If you specify ***** as the host name, then the server's host name is not validated. Other wildcards are not permitted.

Property	Value	Default	Description
`sslKeyStore`	`<file>` | `<PEM-encoded-identity>` msCrypto: MY | Root | Trust | CA	`$SECUDIR/sapcli.pse` (CommonCryptoLib) `$HOME/.ssl/key.pem` (OpenSSL) MY (msCrypto)	Specifies the path to the keystore file that contains the client's identity. An identity is used for mutual TLS authentication and consists of the client's private key, the client's certificate, and, optionally, the certificate of the signing authority that signed the client's certificate. If you are using CommonCryptoLib, then the PSE file must also contain the trust store (for example, the server's public certificates) and the sslTrustStore property should be empty. If you are using CommonCryptoLib, but not doing mutual TLS authentication, then the PSE file contains only the trust store. If you are using CommonCryptoLib, then use the SAPGENPSE tool (installed with CommonCryptoLib) to create the client PSE (`sapcli.pse`) and generate the client's keys. You must also import the server's public certificates into `sapcli.pse`. Typically, this is the root certificate or the certificate of the certification authority that signed the server's public certificates. See *SAP Note 1718944 (SAP HANA DB: Securing External SQL Communication (CommonCryptoLib))*. If you are using OpenSSL, use OpenSSL tools to create the required keystore file. Alternatively, the PEM-encoded client identity can be provided directly as a string. Use the SAPGENPSE tool to export the contents of a PSE file in PEM format. When you provide the client identity as a string, use the sslTrustStore property to provide the PEM-encoded trust store as a string.

Property	Value	Default	Description
`sslTrustStore`	`<file>` | `<PEM-encoded-trust-store>` msCrypto: MY | Root | Trust | CA	`$HOME/.ssl/trust.pem` (OpenSSL) sapcli.pse (CommonCryptoLib) MY (msCrypto)	Specifies the path to a trust store file that contains the server's public certificates if using OpenSSL. Typically, the trust store contains the root certificate or the certificate of the certification authority that signed the server's public certificates. If you are using the cryptographic library CommonCryptoLib or msCrypto, leave this property empty. Alternatively, the PEM-encoded client identity can be provided directly as a string. Use the SAPGENPSE tool to export the contents of a PSE file in PEM format. When you provide the client identity as a string, use the sslTrustStore property to provide the PEM-encoded trust store as a string.
`sslSNIHostname`	`<string>`	Empty	Specifies the name of the host that is attempting to connect at the start of the TLS handshaking process.
`sslValidateCertificate`	Boolean	TRUE	Specifies whether to validate the server's certificate.

Related Information

Connect to SAP HANA via ODBC
ODBC Connection Properties
SAP Note 1718944

5.3.1.5 Client-Side TLS/SSL Connection Properties (JDBC)

For clients connecting via the JDBC interface, TLS/SSL is configured using connection properties.

The following table lists the connection properties that can be used to configure TLS/SSL for JDBC client access.

For more information about other JDBC connection properties and how to set them, see the *SAP HANA Client Interfaces Programming Reference*.

Property	Value	Default	Description
`cseKeyStorePassword`	`<string>`	Empty	Provides the password for the local key store. This option is required when using client-side encryption. s must be at least eight characters and have at least one upper case character, one lower case character, and one number.
`encrypt`	Boolean	FALSE	Enables or disables TLS encryption.
`hostNameInCertificate`	`<string>`	Empty	Specifies the host name used to verify server's identity. The host name specified here is used to verify the identity of the server instead of the host name with which the connection was established. For example, in a single-host system, if a connection is established from a client on the same host as the server, a mismatch would arise between the host named in the certificate (actual host name) and the host used to establish the connection (localhost). **Note** If you specify `*` as the host name, this property has no effect. Other wildcards are not permitted.
`keyStore`	`<file>` | `<store name>`	`<VM default>`	Specifies the location of the Java keystore.
`keyStorePassword`	`<password>`	`<VM default>`	Specifies the password used to access the private key from the keystore file. **Note** This property is not used for SAP HANA studio connections.
`keyStoreType`	`<JKS>` | `<PKCS12>`	`<VM default>`	Specifies the Java keystore file format.
`sniHostname`	`<string>`	Empty	Specifies the name of the host that is attempting to connect at the start of the TLS handshaking process.

Property	Value	Default	Description
`sslKeyStore`	`<string>`	Empty	Specifies an alternative to the `keyStore` connection property. This property allows you to specify the contents of the keystore file as a string. The certificates and private keys must be PEM-encoded, and the private keys must be in PKCS8 format.
`sslTrustStore`	`<string>`	Empty	Specifies an alternative to the `trustStore` connection property. This property allows you to specify the contents of the truststore file as a string. The certificates must be PEM-encoded.
`trustStore`	`<file>` | `<store name>`	`<VM default>`	Specifies the path to the trust store file that contains the server's public certificate(s). Typically, the trust store contains the root certificate or the certificate of the certification authority that signed the server's certificate(s).
`trustStorePassword`	`<password>`	`<VM default>`	Specifies the password used to access the trust store file.
`trustStoreType`	`<JKS>`	`<VM default>`	Specifies the file format of the trust store file.
`validateCertificate`	Boolean	TRUE	If set to true, specifies that the server's certificate is validated.

Related Information

Connect to SAP HANA via JDBC
JDBC Connection Properties

5.3.1.6 SOCKS Proxy Communication Protocol

SOCKS proxy is a communication protocol that uses a proxy server to exchange network packets between a client and server.

SOCKS proxy is an insecure network communication protocol.

Connect to a SOCKS proxy server by using either SAP HANA HDBSQL or the ODBC driver.

SAP HANA hdbsql	Specify the `-proxyhost <hostname>` option. You can also specify additional, optional SOCKS proxy hdbsql options.
ODBC driver	Specify the `PROXY_HOST <hostname>` connection property. You can also specify additional, optional SOCKS proxy ODBC connection properties.

Related Information

SAP HANA HDBSQL Options
ODBC Connection Properties

5.3.2 Secure Communication Between SAP HANA and an LDAP Directory Server

You can use the Transport Layer Security (TLS)/Secure Sockets Layer (SSL) protocol or Secure LDAP protocol (LDAPS) to secure communication between SAP HANA and an LDAP directory server.

The Lightweight Directory Access Protocol (LDAP) is an application protocol for accessing directory services. If you use an LDAP-compliant directory server to manage users and their access to resources, you can leverage LDAP-based authentication to access SAP HANA and LDAP group membership to authorize users. With LDAP-based authentication, users can also be automatically provisioned in SAP HANA. For more information, see the sections on LDAP authentication and LDAP group authorization.

> ⚠ Caution
>
> If the LDAP server is being used for user authentication, communication between SAP HANA and the LDAP server must be secured to protect the transmission of user passwords.

The connection between the SAP HANA database and an LDAP server can be secured in one of two ways:

- **LDAP with STARTTLS**
  The LDAP connection is initiated as unsecured. However, STARTTLS upgrades the connection to encrypted during the connection. It allows the LDAP server to handle secured and unsecured connections using the same port (port 389).

  > ⓘ Note
  >
  > The connection is secured before user credentials are transmitted.

  To use LDAP with STARTTLS, the URLs specified in the LDAP provider must start with “ldap://”. The LDAP provider in SAP HANA must be configured using the CREATE | ALTER LDAP PROVIDER statement with SSL ON.
- **LDAP over TLS/SSL**
  Traditionally secured LDAP connections are handled on a separate port (port 636) and require the use of URLs that start with "ldaps://". With this approach, the TLS handshake is completed before any LDAP protocol messages are exchanged.

To use LDAP over TLS/SSL, the URLs specified in the LDAP provider must start with "ldaps://". The LDAP provider in SAP HANA must be configured using the CREATE | ALTER LDAP PROVIDER statement with SSL OFF.

> **Note**
>
> TLS/SSL-secured communication uses the SAP cryptographic library, CommonCryptoLib. For more information, see SAP Note 1848999.

To force all connections between the SAP HANA database and an LDAP server to use either LDAP with STARTTLS or LDAP over TLS/SSL, you can set the parameter `sslenforce` in the `ldap` section of the `global.ini` configuration file to true. Unsecured connections are refused.

When using the TLS/SSL protocol, the trust store used to authenticate communication must be a certificate collection in the SAP HANA database with the purpose *LDAP*. The certificate of the Certificate Authority (CA) used to sign the certificate used by the LDAP server must be available in this certificate collection. For more information, see the section on certificate management.

Properties for configuring communication with the LDAP server (including TLS/SSL configuration) are available in the `ldap` section of the `global.ini` configuration file. For more information, see the section on LDAP configuration properties.

Related Information

LDAP Group Authorization for Existing Users [page 207]
LDAP User Authentication [page 130]
Communication Configuration Properties for LDAP [page 60]
Certificate Management in SAP HANA [page 303]
CREATE LDAP PROVIDER Statement (Access Control)
SAP Note 1848999
SAP Note 2438641

5.3.2.1 Communication Configuration Properties for LDAP

Properties for configuring LDAP communication

Communication

The parameters for configuring communication between the SAP HANA database and an LDAP server are available in the `ldap` section of the `global.ini` configuration file. In general, it's not necessary to configure any of the following properties explicitly. The default configuration can be used.

TLS/SSL Properties

Parameter	Value	Default	Description
`sslMinProtocolVersion`	{SSL30,TLS10,TLS11,TLS12,TLS13} **Note** The parameter values for TLS versions are 'tls10', 'tls11', 'tls12' and 'tls13', and these should not be confused with the formal notation such as 'TLS1.2' and 'TLS1.3'.	TLS12	The minimum available TLS/SSL protocol version
`sslMaxProtocolVersion`	{TLS10,TLS11,TLS12,MAX} **Note** The parameter values for TLS versions are 'tls10', 'tls11', 'tls12' and 'tls13', and these should not be confused with the formal notation such as 'TLS1.2' and 'TLS1.3'.	MAX	The maximum available TLS/SSL protocol version **Note** The MAX value is internally mapped to TLS12.
`sslCipherSuites`	String value	PFS:HIGH::EC_HIGH:+EC_OPT	The encryption algorithms available for TLS/SSL connections **Note** CommonCryptoLib is used.
`sslenforce`	`<Boolean value>`	false	Indicates whether the connection between the SAP HANA database and an LDAP server must be secured using either LDAP with STARTTLS or LDAP over TLS/SSL. Unsecured connections are refused. For more information, see *Secure Communication Between SAP HANA and an LDAP Directory Server*.

Other Communication Properties

Parameter	Value	Default	Description
`timeout`	Timeout in seconds	0	Timeout for LDAP operations in seconds The default value of 0 means there is no timeout.

> **Note**
>
> SAP HANA does not support referral chasing and aliases are never dereferenced.

Related Information

Secure Communication Between SAP HANA and an LDAP Directory Server [page 59]
Recommendations for Network Configuration

5.3.3 Secure Communication Between SAP HANA XS Classic and HTTP Clients

You can use the Transport Layer Security (TLS)/Secure Sockets Layer (SSL) protocol to secure communication between the SAP HANA XS classic server and HTTP clients.

The SAP HANA XS classic server allows Web-based applications to access SAP HANA via HTTP. The internal Web Dispatcher of the SAP HANA system manages these incoming HTTP requests.

Therefore, to secure communication between the SAP HANA system and HTTP clients, you must configure the internal SAP Web Dispatcher to use TLS/SSL for inbound application requests. You can do this using the SAP HANA Web Dispatcher Administration tool.

A per-database configuration of TLS/SSL keys and certificates is possible. It is also possible to configure HTTPS on the basis of a single "wildcard" server certificate that covers all databases.

> **Caution**
>
> Do not use a wildcard server certificate if strict isolation between tenant databases is required. If authentication relies on a wildcard certificate and a shared trust store, users of one tenant database will be able to log on to other databases in the system.

For more information, see the *SAP HANA Administration Guide*.

Related Information

Configure HTTPS (SSL) for Client Application Access
Configure HTTP(S) Access to Tenant Databases via SAP HANA XS Classic
Network and Communication Security with SAP HANA XS Advanced [page 374]

5.3.3.1 HTTP Access Log

To monitor all HTTP(s) requests processed in an SAP HANA system, you can set up the internal Web Dispatcher to write a standardized HTTP log for each request.

To configure the Web Dispatcher to log all HTTP(s) requests, you add the property `icm/http/logging_0` to the `[profile]` section of the `webdispatcher.ini` configuration file, specifying the following value:

```
PREFIX=/, LOGFILE=$(DIR_INSTANCE)/trace/access_log-%y-%m-%d, MAXSIZEKB=10000,
SWITCHTF=day, LOGFORMAT=SAP
```

This will generate access log files in the following directory: `/usr/sap/<sid>/HDB<instance>/<host>/trace/access_log-<timestamp>`.

> **Example**
>
> Sample log file entry: `[26/Nov/2014:13:42:04 +0200] 10.18.209.126 BOB - "GET /sap/xse/test/InsertComment.xsjs HTTP/1.1" 200 5 245`

The last three numbers are the HTTP response code, the size in bytes, and the response time in milliseconds. For more information about logging and alternative log formats, see the Internet Communication Manager (ICM) documentation on SAP Help Portal.

You can configure the `webdispatcher.ini` configuration file and view log files in the SAP HANA cockpit.

Related Information

View Diagnostic Files in the SAP HANA Database Explorer
View Diagnosis Files in SAP HANA Studio
Configuring SAP HANA System Properties (INI Files)
Traces and Trace Configuration for Internal Web Dispatcher

5.3.4 Secure Internal Communication

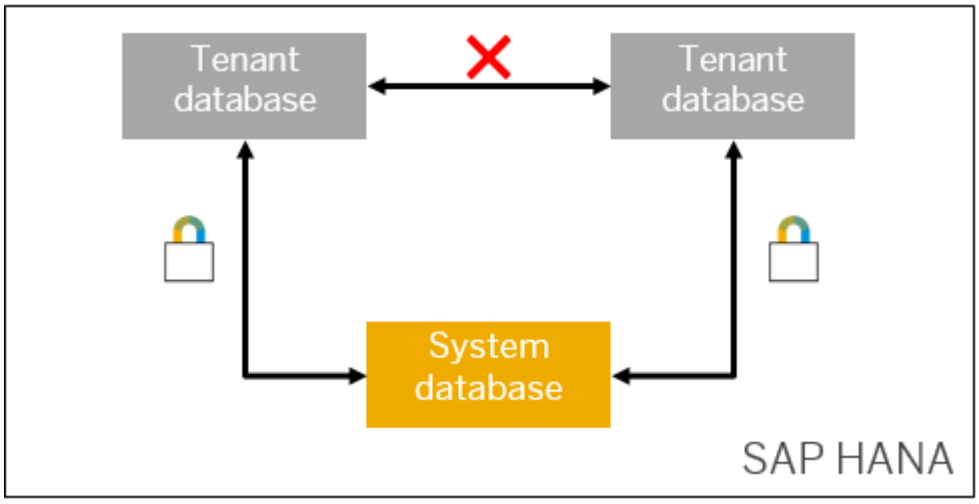
All internal SAP HANA communication can be secured using the Transport Layer Security (TLS)/Secure Sockets Layer (SSL) protocol. A simple public-key infrastructure (PKI) is set up during installation for this purpose.

Internal Communication Channels

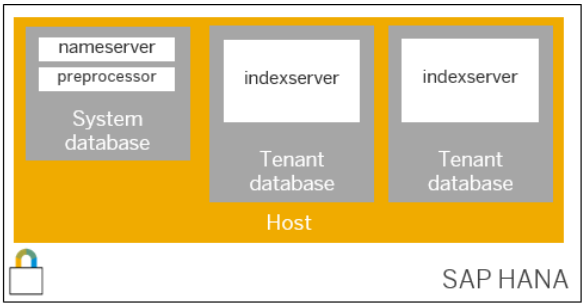
The internal communication channels shown below can be secured using TLS/SSL.

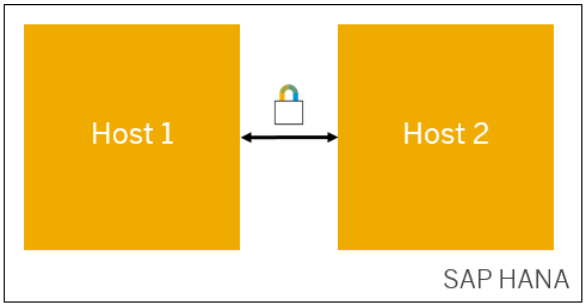
> **Note**
>
> SAP HANA internal communication has sometimes been unofficially referred to as TREXNet communication. However, the term TREXNet is not valid in the context of SAP HANA.

Communication between databases



Communication between the hosts in a multiple-host system and between processes on a single host



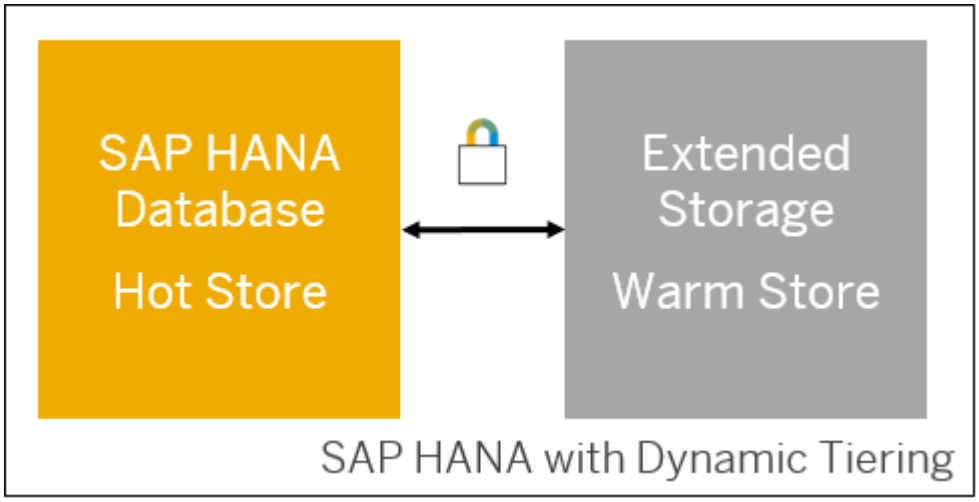


Communication between the sites in a system with system replication enabled



Communication between the SAP HANA database and additional server components

For example: Extended storage server (SAP HANA dynamic tiering) or a streaming analytics server (SAP HANA Streaming Analytics).



System Public Key Infrastructure

A dedicated PKI is created for internal communication automatically during system installation. Every host on which a database server and optional component server is running, as well as every tenant database in the system, are integrated into this PKI, which uses CommonCryptoLib as the cryptographic library.

Each host and database receive a public and private key pair and a public-key certificate for mutual authentication. These certificates are all signed by a dedicated trusted certificate authority (CA) that is unique to the SAP HANA instance. The root personal security environment (PSE) file is stored in the system PKI SSFS (secure store in the file system). All other PSEs are encrypted with an automatically generated random PIN and stored in the file system. Certificates are automatically renewed when they expire.

> ⓘ Note
>
> A unique master key that protects the system PKI SSFS is generated during installation or update. However, if you received your system pre-installed from a hardware or hosting partner, we recommend

that you change it immediately after handover to ensure that it is not known outside of your organization. For more information about how to change the SSFS master keys, see the *SAP HANA Administration Guide*.

To secure internal communication between hosts and sites, you can set up and configure your own PKI, but we recommend you use the system PKI. The system PKI is always used to secure communication within tenant databases and communication with optional server components.

> ### Note
>
> If high isolation is configured for tenant databases, the system PKI **must** also be used to secure communication between hosts.

For more information about migrating to the system PKI from a manually configured PKI, see SAP Note 2175672.

TLS/SSL Configuration Using System PKI

No interaction is required to set up the system PKI, but you may need to explicitly enable TLS/SSL depending on the channel as follows:

Communication Channel	Configuration Required to Enable TLS/SSL
Communication between the processes of individual databases	Configure the system for high isolation. High isolation requires that the processes of individual databases run under dedicated operating system (OS) users in dedicated OS groups. In addition, it enables certificate-based authentication so that only the processes belonging to the same database can communicate with each other. If you also want data communication within databases to be encrypted, you must change the value of the property `[communication] ssl` in the `global.ini` from **`off`** to **`systemPKI`**. If the property `ssl` is not visible (for example, in the SAP HANA studio), add the key **`ssl`** with the value **`systemPKI`** to the section `communication`. → Remember Change (or add) the property in the system database in the SYSTEM layer of the configuration file. Note If cross-database access is enabled, communication between configured tenant databases is allowed. For more information about how to configure a system for high isolation, see the *SAP HANA Administration Guide*.

Communication Channel	Configuration Required to Enable TLS/SSL
Communication between hosts in a multiple-host system and localhost communication	Enable TLS/SSL manually. In the `global.ini` configuration file, change the value of the property `[communication] ssl` to **`systemPKI`**. This configuration ensures that only hosts belonging to the same system can communicate with each other and that all data communication between hosts is encrypted. **Note** In a system that is not configured for high isolation, you can still enable secure communication between hosts. Remember to change the property in the system database in the SYSTEM layer. Enabling secure communication between hosts automatically enables secure communication between processes on the same host without any further configuration. Note the following: • If you are operating a single-host and require secure localhost communication, you must still enable TLS/SSL for inter-host communication as described above. • If you have enabled TLS/SSL for inter-host communication as described above, but do not require secure localhost communication, you can change the value of the property `[communication] ssl_local` from **`on`** to **`off`**.
Communication between sites in a system with system replication enabled	Several steps are required to enable TLS/SSL for the communication channel used for system replication. For more information, see *Secure Internal Communication Between Sites in System Replication Scenarios*.
Communication between the SAP HANA database and additional server components	No configuration required TLS/SSL is automatically enabled and cannot be disabled.

Related Information

Server-Side Secure Stores [page 235]
Change the SSFS Master Keys
Database Isolation [page 23]
Increase the System Isolation Level
Server-Side TLS/SSL Configuration Properties for Internal Communication [page 69]
Secure Internal Communication Between Sites in System Replication Scenarios [page 68]
Legacy Configuration of Secure Internal Communication [page 71]
SAP Note 2175672

5.3.4.1 Secure Internal Communication Between Sites in System Replication Scenarios

Communication between sites in a system replication scenario is always authenticated and internal network communication between primary and secondary systems is secured by default using TLS/SSL.

Communication between the sites in a system replication landscape must be secured. The system PKI (public key infrastructure) that is automatically created during system installation is the default and recommended mechanism for communication. No interaction is required to set up the system PKI. However, you can also set up and configure your own PKI (see *Legacy Configuration of Secure Internal Communication*).

> → Remember
>
> System replication is configured for the system as a whole. This means that the system database and all tenant databases are part of the system replication.

Configuring Authentication Between Sites

To ensure that only configured systems in a system replication landscape can communicate with each other, SAP HANA uses certificate-based authentication based on the system PKI. To establish trust between systems, you must copy the system PKI SSFS data file and key file from the primary system to the same location on the secondary system(s). These files can be found at the following locations:

- `$DIR_INSTANCE/../SYS/global/security/rsecssfs/data/SSFS_<SID>.DAT`
- `$DIR_INSTANCE/../SYS/global/security/rsecssfs/key/SSFS_<SID>.KEY`

Copy the files before you register the secondary system with the primary system.

Configuring TLS/SSL-Secured Communication

In addition to authenticated communication, SSL is enabled by default to secure the communication channels between primary and secondary systems:

- Metadata channel used to transmit metadata (for example, topology information) between the sites
- Data channel used to transmit data between the sites

> ⓘ Note
>
> On SAP HANA systems with dynamic tiering, the same configuration applies. No additional steps are required. However, before you configure communication for dynamic tiering, see SAP Note 2356851.
>
> Be aware that you need additional licenses for SAP HANA options and capabilities. For more information, see Important Disclaimer for Features in SAP HANA [page 440].

Additional Network Security

You can further secure communication between sites by configuring SAP HANA to use exclusively a separate network dedicated to system replication for communication between primary and secondary sites.

It is also recommended that you configure the IP addresses of those hosts that are allowed to connect to the ports required for system replication.

For more information, see also the section on network security and the section on host name resolution in the *SAP HANA Administration Guide*.

Related Information

Legacy Configuration of Secure Internal Communication [page 71]
Server-Side TLS/SSL Configuration Properties for Internal Communication [page 69]
Configure Secure Communication (TLS/SSL) Between Primary and Secondary Sites
Host Name Resolution for System Replication
Network Security [page 36]
Secure Internal Communication [page 64]
SAP Note 2356851

5.3.4.2 Server-Side TLS/SSL Configuration Properties for Internal Communication

The properties for configuring TLS/SSL for internal SAP HANA communication are available in the `communication` section of the `global.ini` configuration file.

The following properties are available for configuring TLS/SSL for internal communication.

> **Note**
>
> Only the system administrator can configure these properties. This is because tenant database administrators are prevented from changing any communication properties. They are in the default configuration change blocklist (`multidb.ini`).

Property	Value	Default	Description
`ssl`	{on | systemPKI | off}	off	Enables TLS/SSL on internal communication channels The following values are possible: • **`off`** (default) With this value, TLS/SSL is disabled. In systems configured for high isolation, but with the default value **`off`**, host and database authentication is enabled but internal communication is not encrypted. • **`systemPKI`** This value enables the use of the system PKI for secure communication between hosts (host authentication and encrypted data communication). This value additionally enables encrypted data communication within databases. For more information, see *Secure Internal Communication*. If you have installed the new runtime environment for application development, SAP HANA Extended Application Services (XS) Advanced Model, the value **`systemPKI`** is set automatically during installation. • **`on`** This value enables the use of a manually configured PKI for secure communication between hosts. Additional properties for trust and key stores apply in this case. For more information, see *Legacy Configuration of Secure Internal Communication*. > **Note** > In a system configured for high isolation, do not set the value of this property to **`on`**. This will result in an error. To change the default value of this property, you must first add it to the `communication` section of the `global.ini` file.
`ssl_local`	`<Boolean value>`	on	Enables TLS/SSL for communication between localhost processes This parameter is only evaluated if `ssl` has the value **`systemPKI`** or **`on`**.

Property	Value	Default	Description
`sslInternalValidateCertificate`	`<Boolean value>`	true	If **`true`**, the certificate of the communication partner is validated In a system configured for high isolation, this parameter is ignored. The certificate of the communication partner is always validated.

Related Information

Secure Internal Communication [page 64]
Legacy Configuration of Secure Internal Communication [page 71]
SAP HANA Extended Application Services, Advanced Model
Default Blacklisted System Properties in Tenant Databases [page 417]

5.3.4.3 Legacy Configuration of Secure Internal Communication

Although it is recommended that you use the system PKI (public key infrastructure) that is automatically created during system installation to secure internal communication channels, you can set up and configure your own PKI. This manually configured PKI is also used if system replication is configured for the system.

TLS/SSL Configuration for Communication Between Hosts

Since a host can both initiate a connection with another host (client role) as well as be the target of a connection initiated by another host (server role), every host in the system requires a public and private key pair, and a public-key certificate (server certificate) with which it can identify itself to other hosts. Each host also needs the certificate or certificates with which it can validate the identity of other hosts. Typically, this is the root certificate or the certificate of the certification authority (CA) that signed the other hosts' certificates.

> **Note**
>
> SAP HANA dynamic tiering does not support legacy configuration (using a manually configured PKI). If you are using SAP HANA dynamic tiering, use the system PKI configuration.

Use CommonCryptoLib as the cryptographic library. It is installed by default as part of SAP HANA server installation.

To manually configure secure communication between hosts:

1. Create a CA for the SAP HANA installation using external tools, for example, the OpenSSL command line tool.

We recommend that you use a dedicated CA to sign all certificates used. We recommend storing your CA certificate in `$DIR_INSTANCE/ca`. This is typically the root certificate.

> → Recommendation
>
> Create one private CA for each SAP HANA host. Do not use public CA for securing internal SAP HANA communication.

2. On every host, create the required server certificates.
   Every host is verified with its fully qualified domain name (FQDN). The common name (CN) must be the FQDN of the host you get by reverse DNS look-up. The other fields describe your organization.
3. Sign the certificates with the CA.
4. On every host, create a local keystore named `sapsrv_internal.pse` in directory `$SECUDIR` and import the private key and certificate, and the CA certificate (or root certificate).
   In the `communication` section of the file `global.ini`, create the property `ssl` with the value **on**.

TLS/SSL Configuration for Cross-Site Communication in System Replication Scenarios

In a system with system replication enabled, communication between sites (metadata and data channels) can be secured using the same configuration described above. For the data communication SSL is enabled by default (property `[system_replication_communication] enable_ssl` in the `global.ini` configuration file). For more information, see *Secure Internal Communication Between Sites in System Replication Scenarios*.

Keystore Configuration

The `[communication] sslInternalKeyStore` parameter in the `global.ini` configuration file specifies the path to the keystore file that contains the certificates for the following internal communication channels:

- Communication between hosts
- Communication between sites in system replication scenarios (data communication channel).

The default value is `$SECUDIR/sapsrv_internal.pse`.

Related Information

Secure Internal Communication Between Sites in System Replication Scenarios [page 68]

6 SAP HANA User Management

SAP HANA database users may be technical users or correspond to real end users. Several tools are available for user management.

Every user who wants to work directly with the SAP HANA database must have a database user with the necessary privileges. Depending on the scenario, the user accessing SAP HANA may either be a technical system user or an individual end user.

After successful logon, the user's authorization to perform the requested operations on the requested objects is verified. This is determined by the privileges that the user has been granted. Privileges can be granted to database users either directly, or indirectly through roles. Several tools are available for provisioning and managing users. For more information about the authorization model of the SAP HANA database, see the section on authorization.

Related Information

SAP HANA Authentication and Single Sign-On [page 100]
SAP HANA Authorization [page 135]

6.1 User Types

It is often necessary to specify different security policies for different types of database user. In the SAP HANA database, we differentiate between database users that correspond to real people and technical database users.

Technically, database users that correspond to real people and technical database users are the same. The only difference between them is conceptual.

Database Users that Correspond to Real People

Every person who needs to work with SAP HANA must have a database user. Depending on your system configuration and scenario, database users can be created by:

- User administrators
- User group administrators
- An LDAP provider used for user authentication and enabled for automatic user creation

> **Note**
>
> In SAP HANA, the user is technically created by the user `SYS` (see CREATOR column in system view USERS).

Database users that correspond to real people are dropped when the person leaves the organization. This means that any database objects that they own are also automatically dropped, and any privileges that they granted are automatically revoked.

> **Note**
>
> Database users created by an LDAP provider must be dropped manually by a user administrator. If the user is dropped on the LDAP server, the corresponding database user in SAP HANA is **not** automatically dropped.

Database users are created with either the `CREATE USER` or `CREATE RESTRICTED USER` statement, or using the *User Management* app.

Standard Users

Standard users correspond to users created with the `CREATE USER` statement. By default they can create objects in their own schema and read data in system views. Read access to system views is granted by the `PUBLIC` role, which is granted to every standard user.

Restricted Users

Restricted users, which are created with the `CREATE RESTRICTED USER` statement, initially have no privileges. Restricted users are intended for provisioning users who access SAP HANA through client applications and who are not intended to have full SQL access via an SQL console. If the privileges required to use the application are encapsulated within an application-specific role, then it is necessary to grant the user only this role. In this way, it can be ensured that users have only those privileges that are essential to their work.

Compared to standard database users, restricted users are initially limited in the following ways:

- They cannot create objects in the database as they are not authorized to create objects in their own database schema.
- They cannot view any data in the database as they are not granted the standard PUBLIC role.
- They are only able to connect to the database using HTTP/HTTPS.
  For restricted users to connect via ODBC or JDBC, access for client connections must be enabled by executing the SQL statement `ALTER USER <user_name> ENABLE CLIENT CONNECT` or enabling the corresponding option for the user in the SAP HANA cockpit.
  For full access to ODBC or JDBC functionality, users also require the predefined role `RESTRICTED_USER_ODBC_ACCESS` or `RESTRICTED_USER_JDBC_ACCESS`.

  > **Note**
  >
  > Disabling ODBC/JDBC access for a user, either a restricted user or a standard user, does not affect the user's authorizations or prevent the user from executing SQL commands via channels other than JDBC/ODBC. If the user has been granted SQL privileges (for example, system privileges and object privileges), he or she is still authorized to perform the corresponding database operations using, for example, a HTTP/HTTPS client.

A user administrator can convert a restricted user into a standard user (or vice versa) as follows:

- Granting (or revoking) the PUBLIC role
  You can do this by editing the user in the SAP HANA cockpit or with the SQL statement `ALTER USER <username> GRANT | REVOKE ROLE PUBLIC`.
- Granting (or revoking) authorization to create objects in the user's own schema
  You can do this by editing the user in the SAP HANA cockpit or with the SQL statement `ALTER USER <username> GRANT | REVOKE CREATE ANY ON OWN SCHEMA`.
- Enabling (or disabling) full SQL
  You can do this by editing the user in the SAP HANA cockpit or with the SQL statement `ALTER USER <user_name> ENABLE CLIENT CONNECT`.

> **Note**
>
> A user is only identified as a restricted user in system view USERS if he doesn't have the PUBLIC role or authorization for his own schema.

Users created by an LDAP provider can be either standard users or restricted users depending on the configuration.

Technical Database Users

Technical database users do not correspond to real people. They are therefore not dropped if a person leaves the organization. This means that they should be used for administrative tasks such as creating objects and granting privileges for a particular application.

Some technical users are available as standard, for example, the users SYS and _SYS_REPO.

Other technical database users are created for application-specific purposes. For example, an application server may log on to the SAP HANA database using a dedicated technical database user.

Technical users are standard users created with the CREATE USER statement.

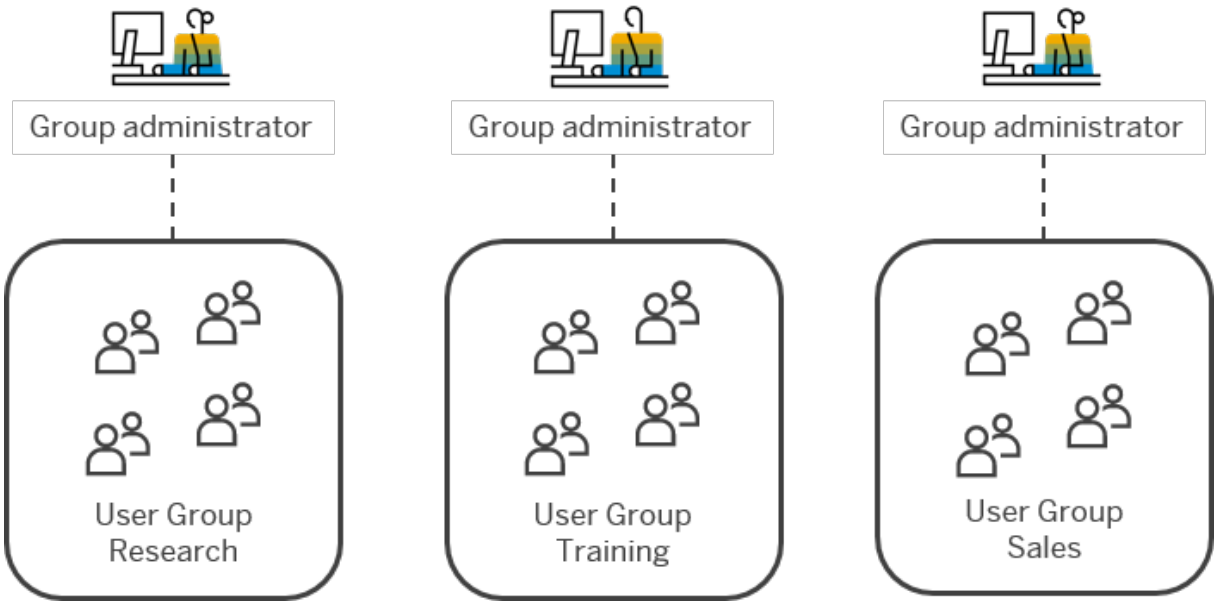
Related Information

CREATE USER Statement (Access Control)
USERS System View
Predefined Database (Catalog) Roles [page 183]

6.2 User Groups

User groups allow you to manage related users together. Group administrators can be assigned to manage individual user groups exclusively and independently of each other.

Overview



User Groups

User groups are an efficient way to manage users:

- Every user group can have its own **dedicated administrator(s)**. In this way, user management tasks can be delegated to several people independently of each other.
- A user group can be configured for **exclusive administration**, which means that only the designated group administrator(s) can manage the users in the group. This could be useful, for example, to protect highly-privileged users or technical users from accidental deletion or manipulation.
- **Group specific-user configuration** for password policy and client connect restrictions settings are possible. For example, you could put the technical users required by connecting applications into their own user group with a customized password policy so that the passwords of these users are extra complex, or define conditions on when users within a user group are allowed to connect to the SAP HANA database. You can also disable access for all users of a user group with a single statement.

> ⓘ Note
>
> User groups do not control data access. They are intended to support a separation of user management duties. A user's authorization (roles and privileges) control data access.

Group Creation and Administration Mode

A global user administrator (that is, a user with system privilege USER ADMIN) creates user groups using the `CREATE USERGROUP` statement. The user administrator can then designate one or more group administrators by granting the object privilege `USERGROUP OPERATOR` on the user group to the relevant user.

> **Note**
>
> A user can be the group administrator of more than one group.

User administrators can also specify the administration "mode" of the user group. This determines who can administer the group:

- Group administrators **and** user administrators (shared administration)
  Not only designated group administrators can modify the group but also any user administrator (that is, any user with system privilege USER ADMIN).
  This is the default administration mode.
- Group administrators only (exclusive administration)
  Only the designated group administrator(s) can modify the group, for example, adding new users to the group or removing users from the group. In this way, groups of users can be managed completely independently of each other.

  > **Note**
  >
  > To add an existing user from the global pool of users to the user group, as well as remove a user from the group (and return it to the global pool of users), the group administrator also needs to be a user administrator, that is have system privilege USER ADMIN.

The user administrator can configure the administration mode when creating the group, or later by editing the group. The authorization mode is configured with the option `ENABLE | DISABLE USER ADMIN` of the `CREATE USERGROUP` and `ALTER USERGROUP` statements as follows:

Statement	Administration Mode
`CREATE USERGROUP <usergroupname>`	Group administrators and user administrators (default if authorization mode is not explicitly specified)
`CREATE USERGROUP <usergroupname> DISABLE USER ADMIN`	Group administrators only (exclusive administration)
`ALTER USERGROUP <usergroupname> DISABLE USER ADMIN`	Group administrators only (exclusive administration)
`ALTER USERGROUP <usergroupname> ENABLE USER ADMIN`	Group administrators and user administrators

> **Example**
>
> 1. User administrator creates a new user group requiring its own exclusive administrator:
>    ```
>    CREATE USERGROUP TechnicalUsers DISABLE USER ADMIN;
>    ```
> 2. User administrator assigns the group administrator:
>    ```
>    GRANT USERGROUP OPERATOR ON USERGROUP TechnicalUsers TO TechnicalUsersAdmin;
>    ```
> 3. Group administrator adds new user to the group:
>    ```
>    CREATE USER sapadm PASSWORD <password> SET USERGROUP TechnicalUsers;
>    ```

For more information on who can perform exactly which operations, see the *Reference: Authorization for User Group Administration*.

Group Membership

User administrators and/or group administrators add new or existing users to a user group with the `SET USERGROUP` option of the `CREATE | ALTER USER` statements.

Statement	Group Membership
`CREATE USER <username> SET USERGROUP <usergroupname>`	Creates new user in a user group
`ALTER USER <username> SET USERGROUP <usergroupname>`	Adds an existing user to a user group
`ALTER USER <username> UNSET USERGROUP`	Removes a user from a user group

A user can belong to only one user group, but a user does not have to be a member of any group. Users who are not in any group are managed as normal by user administrators.

To move a user from one group to another, a user authorized for both user groups simply add the user to the new user group with the `ALTER USER <username> SET USERGROUP <usergroupname>`. This automatically removes the user from the original user group.

User Group Configuration

In addition to grouping users into meaningful categories, user groups also allow you to mass manage certain user settings and parameters. In this way, you can configure all users in a user group not only quickly but differently to users in other groups. Groups can be configured using the `CREATE | ALTER USERGROUP` statement.

Group Setting: Client Connect Restrictions

- Use the `ENABLE | DISABLE CLIENT CONNECT` option to control whether or not the users in a user group can connect to SAP HANA, for example to stop users temporarily from connecting during updates or troubleshooting activities:

  Sample Code

  ```
  ALTER USERGROUP MyUserGroup DISABLE CLIENT CONNECT
  ```

- Define conditions on when users within a user group are allowed to connect to the SAP HANA database. For more information, see *Connect Restrictions*.

Configuring User Group Parameters

There are two steps to configuring users with user groups:

- Configuring the group-specific values of parameters
- Enabling the parameter set to which the configured parameters belong so that they take effect

To set parameter values use the `SET PARAMETER` option to enable parameter sets, use the `ENABLE PARAMETER SET` option. To see the group-specific values of parameters, query the view `USERGROUP_PARAMETERS`.

Password policy is the only set of parameters that can be configured for a user group.

Parameter Set: Password Policy

The users of different user groups may have different requirements when it comes to passwords. For example, you may want the passwords of technical users to be very complex. A group administrator can configure group-specific values for the individual parameters of the password policy.

Sample Code

```
CREATE USERGROUP Training SET PARAMETER 'password_layout' = 'A1a!',
'minimal_password_length' = '16' ENABLE PARAMETER SET 'password policy'
```

Note

If a group-specific value is not explicitly set for a parameter, the value configured in the password policy of the database appears as the user group value in `USERGROUP_PARAMETERS`.

To see which values are currently in effect for a particular user, query the view `M_EFFECTIVE_PASSWORD_POLICY`.

Sample Code

```
SELECT * from "PUBLIC"."M_EFFECTIVE_PASSWORD_POLICY" where USER_NAME =
'<user_name>'
```

Related Information

SAP HANA Authorization [page 135]
Password Policy [page 105]
Password Policy Configuration Options [page 107]
SQL Statements and Authorization for User Group Administration (Reference) [page 83]
CREATE USERGROUP Statement (Access Control)
ALTER USERGROUP Statement (Access Control)
USERGROUPS System View
CREATE USER Statement (Access Control)
ALTER USER Statement (Access Control)
USERGROUP_PARAMETERS System View
M_EFFECTIVE_PASSWORD_POLICY System View

Connect Restrictions [page 80]

6.2.1 Connect Restrictions

Connect restrictions are arrays of access policies that limit the access to the SAP HANA database to users within a user group, such as restricting administrator users to a certain IP range or technical users to specific applications. These restrictions are evaluated during the user group's authentication process, and if the user group has connect restrictions, members of that group can only connect if at least one allows access.

Evaluation of Connect Restrictions

If a user group has connect restrictions, members of the user group can only connect to the SAP HANA database if at least one of the connect restrictions allows access.

> **Note**
>
> A connect restriction can consist of multiple conditions and all must be met to allow access.

Creating a User Group-Specific Connect Restriction

A single connect restriction can contain an array of multiple application,IP, and authentication method conditions.

Examples

Create a connect restriction called RESTRICTION_ONE with conditions for IP addresses in CIDR notation and application. The restriction allows connections for hdbsql applications from the mentioned IP addresses and IP ranges:

> **Sample Code**

```
CREATE USERGROUP Usergroup1 CONNECT RESTRICTION (RESTRICTION_ONE IP
('1.2.3.4', '1.2.3.4/8', '1:2:3:4:5:6:7:8/16') APPLICATION 'hdbsql');
```

Update the connect restriction called RESTRICTION_TWO with conditions for connections from a single IP address, for example from your corporate network:

> **Sample Code**

```
ALTER USERGROUP Usergroup1 ADD CONNECT RESTRICTION (RESTRICTION_TWO IP
('192.28.28.0'));
```

Alter RESTRICTION_TWO again and only allow connections with certificates from the corporate network:

Sample Code

```
ALTER USERGROUP Usergroup1 ALTER CONNECT RESTRICTION (RESTRICTION_TWO IP
('192.28.28.0') AUTHENTICATION ('X509'));
```

Note

You can define multiple connect restrictions for a user group. This means that together RESTRICTION_ONE, RESTRICTION_TWO, and APPLICATION allow connections from IP 1.2.3.4, hdbsql application, or from the corporate network with any application.

By default, a new connection restriction is disabled. Use the ALTER USERGROUP statement to enable the restriction:

Sample Code

```
ALTER USERGROUP Usergroup1 ENABLE CONNECT RESTRICTION RESTRICTION_ONE
, RESTRICTION_TWO;
```

The definition of connect restrictions can be viewed in the system view USERGROUP_CONNECT_RESTRICTIONS.

Deactivating Connection for All Users of a User Group

Use the ENABLE | DISABLE CLIENT CONNECT option to control whether or not the users in a user group can connect to the SAP HANA database, for example to stop users temporarily from connecting during updates or troubleshooting activities.

Note

If you disable client connects for a user group using this feature, no user of that user group can connect, even if connect restrictions would allow it.

Sample Code

```
ALTER USERGROUP MyUserGroup DISABLE CLIENT CONNECT
```

Validating a User Group Specific Connect Restriction

Use the table function PUBLIC.VALIDATE_USERGROUP_CONNECT_RESTRICTION to validate whether a connect attempt would be possible given the active connect restrictions of a user group:

Sample Code

```
select * from
SYS.VALIDATE_USERGROUP_CONNECT_RESTRICTION(application_name=>'hdbsql',
usergroup_name=>'USERGROUP1', client_IP_address=>'1.2.3.4');
```

Use the table function PUBLIC.VALIDATE_USERGROUP_CONNECT_RESTRICTION_DETAILS to display details of each connect restriction if a login were allowed for a specific condition:

Sample Code

```
select * from
SYS.VALIDATE_USERGROUP_CONNECT_RESTRICTION_DETAILS(application_name=>'hdbsql',
 usergroup_name=>'USERGROUP1', client_IP_address=>'1.2.3.4');
```

For more information on both tables, see the *SAP HANA Cloud, SAP HANA Database SQL Reference Guide*.

Authentication/Connect Failures Due to Connect Restrictions

Find details on why a connect restriction does not allow a user connection in the SYS.AUTHENTICATION_ERROR_DETAILS table.

Related Information

CREATE USERGROUP Statement (Access Control)
ALTER USERGROUP Statement (Access Control)

6.2.2 SQL Statements and Authorization for User Group Administration (Reference)

Creating and configuring user groups, and subsequently managing the users in those groups requires different combinations of privileges.

Creating and Configuring User Groups

User administrators create and configure user groups. Group administrators can change the configuration.

To...	You need...	For example...
Create a user group	System privilege USER ADMIN	• `CREATE USERGROUP TechnicalUsers;` • `CREATE USERGROUP Research DISABLE USER ADMIN;`
Change the administration mode of a user group	System privilege USER ADMIN	• `ALTER USERGROUP TechnicalUsers DISABLE USER ADMIN;` • `ALTER USERGROUP Research ENABLE USER ADMIN;`
Make another user the group administrator of a user group	• System privilege USER ADMIN, or • Object privilege USERGROUP OPERATOR on the group with the option to grant it to others	`GRANT USERGROUP OPERATOR ON USERGROUP TechnicalUsers TO TechnicalUsersAdmin;`

To...	You need...	For example...
Configure user parameters and enable/disable usergroup parameter sets	• System privilege USER ADMIN, or • Object privilege USERGROUP OPERATOR on the group **Note** If the user group has been configured for exclusive administration, USERGROUP OPERATOR on the group is required.	• `CREATE USERGROUP Training SET PARAMETER 'password_layout' = 'A1a!', 'minimal_password_length' = '7' ENABLE PARAMETER SET 'password policy';` • `ALTER USERGROUP TechnicalUsers SET PARAMETER 'force_first_password_change' = 'false';` • `ALTER USERGROUP TechnicalUsers ENABLE PARAMETER SET 'password policy';` • `ALTER USERGROUP TechnicalUsers DISABLE PARAMETER SET 'password policy';`
Delete a user group	System privilege USER ADMIN **Note** You cannot delete a user group if there are still users in the group.	`DROP USERGROUP Training;`

Managing Users

Managing users who are not in any user group

User administrators manage users who do not belong to a user group.

To...	You need...	For example...
Create, change, delete a user not in any user group	System privilege USER ADMIN	`CREATE USER Michael PASSWORD <password>;`

Managing users in a user group configured for shared administration (default)

Group administrators and user administrators can manage users in user groups configured with the option `ENABLE USER ADMIN`.

To..	You need...	For example...
Create a user in the user group	• System privilege USER ADMIN, or • Object privilege USERGROUP OPERATOR on the group	`CREATE USER John PASSWORD <password> SET USERGROUP Research;`
Delete a user in the user group	• System privilege USER ADMIN, or • Object privilege USERGROUP OPERATOR on the group	`DROP USER John CASCADE;`
Add an existing user from the global pool of users to the user group	System privilege USER ADMIN	`ALTER USER Julie SET USERGROUP Research;`
Move a user from another user group to the user group	• System privilege USER ADMIN, or • Object privilege USERGROUP OPERATOR on both groups	`ALTER USER Julie SET USERGROUP Training;`
Remove a user from the user group (and return to the global pool of users)	System privilege USER ADMIN	`ALTER USER Julie UNSET USERGROUP;`

Users in a user group configured for exclusive administration

Only group administrators can manage users in user groups configured with the option `DISABLE USER ADMIN`.

To..	You need...	For example...
Create a user in the user group	Object privilege USERGROUP OPERATOR on the user group	`CREATE USER sapsid PASSWORD <password> SET USERGROUP TechnicalUsers;`
Delete a user in the user group	Object privilege USERGROUP OPERATOR on the user group	`DROP USER sapsid CASCADE;`
Add an existing user from the global pool of users to the user group	• System privilege USER ADMIN **and** • Object privilege USERGROUP OPERATOR on the group	`ALTER USER Thomas SET USERGROUP TechnicalUsers;`
Move a user from another user group to the user group	• Object privilege USERGROUP OPERATOR on both groups **Note** If the user's current group is not configured for exclusive administration, object privilege USERGROUP OPERATOR on this group is not required; system privilege USER ADMIN is sufficient.	`ALTER USER Julie SET USERGROUP TechnicalUsers;`
Remove a user from the user group	• System privilege USER ADMIN **and** • Object privilege USERGROUP OPERATOR on the group	`ALTER USER Thomas UNSET USERGROUP;`

Related Information

CREATE USERGROUP Statement (Access Control)
ALTER USERGROUP Statement (Access Control)
CREATE USER Statement (Access Control)
ALTER USER Statement (Access Control)
Password Policy Configuration Options

6.3 User Administration Tools

Depending on your organization and its user provisioning strategy, people with different job functions may be involved in the process of user administration. Different tools are used for different tasks.

Native SAP HANA User Administration

The recommended process for provisioning users in SAP HANA is as follows:

1. Define and create roles.
2. Define and create user groups.
3. Create users in user groups.
4. Grant roles to users.

> **Note**
>
> Creating user groups and assigning users to user groups is an optional step and depends on the requirements in your setup.

Further administration tasks include:

- Deleting users when they leave the organization
- Reactivating users after too many failed logon attempts
- Deactivating users if a security violation has been detected
- Resetting user passwords

The following table provides an overview of who does which of these tasks and the SAP HANA tools available:

Job Function	Task	Environment	Tool
Role designer or creator	Create roles and role hierarchies that reflect the access requirements, job function, and responsibilities of system users	Design time **Note** It is also possible to create roles in runtime on the basis of SQL statements. However, it is recommended that you create design-time roles as they offer more flexibility (for example, they can be transported between systems). For more information, see *Catalog Roles and Design-Time Roles Compared*.	• SAP HANA XS advanced: SAP Web IDE for SAP HANA • SAP HANA XS classic:*Developer Workbench* of the SAP HANA studio or *Editor* tool of the SAP HANA Web-based Development Workbench
Application developer	Create roles for new applications developed on SAP HANA XS classic **Note** SAP HANA XS advanced has the additional concept of application roles and role collections. These are independent of database roles in SAP HANA itself. In the XS advanced context, SAP HANA database roles are used only to control access to database objects (for example, tables, views, and procedures) for XS advanced applications. For more information about the authorization concept of XS advanced.	Design time	• SAP HANA XS classic:*Developer Workbench* of the SAP HANA studio or *Editor* tool of the SAP HANA Web-based Development Workbench
User, user group, or system administrator	Create user groups Create SAP HANA database users in user groups Grant roles to database users Delete, deactivate, and reactivate database users	Runtime	• *User Management* app of the SAP HANA cockpit • *User Group Management* app of the SAP HANA cockpit

Job Function	Task	Environment	Tool
	Reset user passwords		• *Security* tool of the SAP HANA Web-based Development Workbench • SAP HANA hdbsql `hdsql` is useful when using scripts for automated processing.
User or system administrator	Grant roles to database users	Runtime	• *Role Assignment* app of the SAP HANA cockpit • *Security* tool of the SAP HANA Web-based Development Workbench

SAP HANA Lifecycle Management Tool hdblcm(gui)

You can use the SAP HANA lifecycle management tools to perform post-installation steps including changing the passwords of the system database user SYSTEM and operating system administrator `<sid>`adm as part of system rename. For more information, see the *SAP HANA Administration Guide*.

Integration into Other Identity Management Tools

LDAP-Compliant Identity Management Server

The Lightweight Directory Access Protocol (LDAP) is an application protocol for accessing directory services. If you use an LDAP-compliant directory server to manage users and their access to resources, you can leverage LDAP-based authentication to access SAP HANA and LDAP group membership to authorize users. With LDAP-based authentication, users can be automatically provisioned in SAP HANA.

> ⓘ Note
>
> LDAP-based authentication and LDAP automatic user provisioning are not supported for SAP HANA XSC applications; however, SAP HANA XSC supports LDAP-based authorization.

> ⓘ Note
>
> Currently, authorization and authentication with OpenLDAP is only supported for Active Directory.

SAP NetWeaver Identity Management

SAP NetWeaver Identity Management 7.2 Support Package Stack (SPS) 3 and higher contains a connector to the SAP HANA database. With SAP NetWeaver ID Management you can perform several user administration tasks in the SAP HANA database, including:

- Creating and deleting user accounts
- Granting roles

> **Note**
>
> Roles created in runtime are supported as of SAP NetWeaver ID Management SPS 8. Roles created in design time are supported as of SPS 9.

- Setting passwords for users

To use the SAP HANA connector for SAP NetWeaver ID Management, a dedicated SAP HANA database user must be created with the following roles and privileges:

- Standard role `MONITORING`
- System privilege `ROLE ADMIN` and `USER ADMIN`
- Object privilege `EXECUTE` on the procedure `GRANT_ACTIVATED_ROLE`

SAP Access Control

SAP Access Control 10.1 can be used to manage access and risk analysis for SAP HANA-based authorizations.

Related Information

User Administration in SAP HANA
Database Roles [page 182]
Catalog Roles and Design-Time Roles Compared [page 188]
SAP HANA DI Roles [page 191]
Repository Roles [page 194]
Authorization in SAP HANA XS Advanced [page 357]
SAP HANA HDBSQL (Command-Line Reference)
User and Role Management
Renaming a System
SAP Note 1986645
Integration into Other Identity Management Tools
LDAP User Authentication [page 130]
LDAP Group Authorization [page 206]
SAP NetWeaver Identity Management
SAP Access Control

6.4 Predefined Database Users

A number of predefined database users are required for installing, upgrading, and operating the SAP HANA database.

The following tables list the users that are available by default in SAP HANA.

> **Note**
>
> If you have installed the runtime environment for application development, SAP HANA Extended Application Services (XS) Advanced Model, several additional predefined users are available. For more information, see the section *Predefined XS Advanced Users*.

Database Users

SYS

User	SYS
Description	SYS is a technical database user. It is the owner of database objects such as system tables and monitoring views.
Password Specification	Not applicable This is a technical database user. It is not possible to log on with this user.

SYSTEM

User	SYSTEM
Description	The SYSTEM database user is created during the creation of the SAP HANA database. It is the most powerful database user with irrevocable system privileges, such as the ability to create other database users, access system tables, and so on. The SYSTEM user of the **system database** has additional privileges, namely the privileges required for managing tenant databases, for example, creating and dropping databases, changing configuration (*.ini) files of databases, performing database-specific data back-ups, stopping and starting databases. The SYSTEM user does **not** automatically have access to objects created in the SAP HANA repository. > **Caution** > Do not use the SYSTEM user for day-to-day activities. Instead, use this user to create dedicated database users for administrative tasks and to assign privileges to these users. It is recommended that you then deactivate the SYSTEM user. You may temporarily reactivate the SYSTEM user for emergency or bootstrapping tasks. See *Deactivate the SYSTEM User*. > **Note** > The SYSTEM user is not required to update the SAP HANA database system; a lesser-privileged user can be created for this purpose. However, to upgrade SAP support package stacks, SAP enhancement packages and SAP systems using the Software Update Manager (SUM) and to install, migrate, and provision SAP systems using the Software Provisioning Manager (SWPM), the SYSTEM user **is required** and needs to be temporarily reactivated for the duration of the upgrade, installation, migration or provisioning.

Password Specification	The initial password of the `SYSTEM` user of the system database and the first tenant database (if applicable) is specified by your hardware partner or certified administrator during installation. After handover, it is important that you change these passwords. The initial password of the `SYSTEM` user of subsequently created tenant databases is specified at the time of database creation. **Note** If you updated a single-container system to SAP HANA 2.0 SPS 02, the `SYSTEM` user of the tenant database created during the update process has the password of the original `SYSTEM` user. You are required to specify the password of the `SYSTEM` user of the new system database during the update process. A user administrator (that is, a user with the system privilege `USER ADMIN`) can change the `SYSTEM` user password of a database in the SAP HANA cockpit. An administrator in the system database with system privilege `DATABASE ADMIN` can reset the password of the `SYSTEM` user in a tenant database. **Note** It is also possible to change the `SYSTEM` user password of the system database as part of a system rename with SAP HANA lifecycle manager.
More Information	For more information about how to change the `SYSTEM` user password, see the *SAP HANA Administration Guide*.

XSSQLCC_AUTO_USER_<generated_ID>

User	`XSSQLCC_AUTO_USER_<generated_ID>`

Description	In the runtime configuration of an SAP HANA XS application (SAP HANA XS, classic model), a technical user is automatically generated for an SQL connection configuration (SQLCC) if no user is specified. The user is created on activation of the SQLCC and is automatically granted the role specified in the configuration. If the SQLCC is deactivated, the user cannot be used in runtime. With the standard SAP HANA XS application *SAP HANA XS Admin Tools*, available with the deployment of delivery unit `HANA_XS_BASE`, two such users are created: • The technical user used by *User Self-Service Administration* tool to execute tasks associated with user self-service requests, for example, sending e-mails in response to user requests. This user is associated with the SQLCC artifact `sap.hana.xs.selfService.userselfService.xssqlcc` and is assigned the role `sap.hana.xs.selfService.user.roles::USSExecutor`. This user cannot be used to log on to SAP HANA. • The technical user used by the *SAP Web Dispatcher HTTP Tracing* tool to connect to the database for the purpose of executing HTTP tracing of SAP HANA XS applications. This user is associated with the SQLCC artifact `sap.hana.xs.admin.webdispatcher.server.common.httpTracing.xssqlcc` and is assigned the role `sap.hana.xs.admin.roles::WebDispatcherHTTPTracingAdministrator`. This user cannot be used to log on to SAP HANA. **Note** Since the above users don't have human readable names, check the assigned roles to see which user is which.
Password Specification	Password-based logon is disabled by default for an automatically generated SQLCC user. Therefore, a password is not required.
More Information	For more information about the roles mentioned above, see *HANA_XS_BASE* in the reference section of the *SAP HANA Security Guide*. For more information about SQLCCs and the above applications, see the section about maintaining the SAP HANA XS classic model runtime in the *SAP HANA Administration Guide*.

_SYS_AFL

User	_SYS_AFL
Description	_SYS_AFL is a technical user that owns all objects for Application Function Libraries.
Password Specification	Not applicable This is a technical database user. It is not possible to log on with this user.
More Information	For more information, see the *SAP HANA Business Function Library (BFL)* reference.

_SYS_DATA_ANONYMIZATION

User	_SYS_DATA_ANONYMIZATION

Description	`_SYS_DATA_ANONYMIZATION` is a technical user that owns all data-anonymization objects
Password Specification	Not applicable This is a technical database user. It is not possible to log on with this user.
More Information	For more information, see the *Data Anonymization* in the *SAP HANA Security Guide*.

_SYS_EPM

User	`_SYS_EPM`
Description	`_SYS_EPM` is a technical database used by the SAP Performance Management (SAP EPM) application
Password Specification	Not applicable This is a technical database user. It is not possible to log on with this user.

_SYS_PLAN_STABILITY

User	`_SYS_PLAN_STABILITY`
Description	`_SYS_PLAN_STABILITY` is a technical user that owns all (execution) plan-stability objects for Abstract SQL Plans (ASP)
Password Specification	Not applicable This is a technical database user. It is not possible to log on with this user.
More Information	For more information, see `ABSTRACT SQL PLAN` in the *SAP HANA SQL Reference Guide*.

_SYS_REPO

User	`_SYS_REPO`
Description	`_SYS_REPO` is a technical database user used by the SAP HANA repository (SAP HANA XS, classic model). The repository consists of packages that contain design time versions of various objects, such as attribute views, analytic views, calculation views, procedures, analytic privileges, and roles. _SYS_REPO is the owner of all objects in the repository, as well as their activated runtime versions. **Note** Objects in the repository can be modeled on data objects that are not part of design time, such as tables that are used in replication scenarios. _SYS_REPO does **not** automatically have authorization to access these objects. _SYS_REPO must therefore be granted the SELECT privilege (with grant option) on all data objects underneath all objects modeled in the repository. If this privilege is missing, the activated objects will be invalidated. This is true even for objects belonging to schema SYS.
Password Specification	Not applicable This is a technical database user. It is not possible to log on with this user.

_SYS_SQL_ANALYZER

User	`_SYS_SQL_ANALYZER`

Description	_SYS_SQL_ANALYZER is a technical user used by the query performance analysis tool of SAP HANA, the SQL analyzer. This tool can be used to view detailed information on each query and can help you evaluate potential bottlenecks and optimizations for these queries. The SQL analyzer is accessible from the SAP HANA cockpit and SAP Web IDE for SAP HANA.
Password Specification	Not applicable This is a technical database user. It is not possible to log on with this user.
More Information	For more information about the SQL analyzer, see the *SAP HANA Administration Guide*.

_SYS_STATISTICS

User	_SYS_STATISTICS
Description	_SYS_STATISTICS is a technical database user used by the internal monitoring mechanism of the SAP HANA database. It collects information about status, performance, and resource usage from all components of the database and issues alerts if necessary.
Password Specification	Not applicable This is a technical database user. It is not possible to log on with this user.

_SYS_TABLE_REPLICAS

User	_SYS_TABLE_REPLICAS
Description	_SYS_TABLE_REPLICAS is a technical database user required by replication and federation components
Password Specification	Not applicable This is a technical database user. It is not possible to log on with this user.

_SYS_TABLE_REPLICA_DATA

User	_SYS_TABLE_REPLICA_DATA
Description	_SYS_TABLE_REPLICA_DATA is a technical database user required by replication and federation components. This user owns system-internal tables containing replicated data
Password Specification	Not applicable This is a technical database user. It is not possible to log on with this user.

_SYS_TASK

User	_SYS_TASK
Description	_SYS_TASK is a technical database user in SAP HANA smart data integration. This user owns all task framework objects.
Password Specification	Not applicable This is a technical database user. It is not possible to log on with this user.

More Information	For more information, see SAP HANA Smart Data Integration and SAP HANA Smart Data Quality on SAP Help Portal.

_SYS_WORKLOAD_REPLAY

User	`_SYS_WORKLOAD_REPLAY`
Description	`_SYS_WORKLOAD_REPLAY` is a technical database user used by capture and replay capability of the SAP HANA cockpit. This tool allows administrators to capture and replay workloads from an SAP HANA system in order to check the impact of a system change (for example, hardware change). The user `_SYS_WORKLOAD_REPLAY` manages control and preprocessing data. Performance results are also stored in this user's schema (`_SYS_WORKLOAD_REPLAY`), but are only accessible by internal procedures.
Password Specification	Not applicable This is a technical database user. It is not possible to log on with this user.
More Information	For more information about SAP HANA Workload Capture and Replay, see the *SAP HANA Administration Guide*.

_SYS_XB

User	`_SYS_XB`
Description	`_SYS_XB` is a technical user for internal use only.
Password Specification	Not applicable This is a technical database user. It is not possible to log on with this user.

Database Users Related to the SAP HANA Deployment Infrastructure (HDI)

> **Note**
>
> The following users exist only if HDI is enabled in the database.

_SYS_DI*

User	`_SYS_DI`, `_SYS_DI_*_CATALOG`, `_SYS_DI_SU`, `_SYS_DI_TO`

Description	Technical users of the SAP HANA Deployment Infrastructure (HDI): • `_SYS_DI` Owns all HDI SQL-based APIs, for example all API procedures in the _SYS_DI schema and API procedures in containers • `_SYS_DI_*_CATALOG` Technical users used by the HDI to access database system catalog tables and views • `_SYS_DI_SU` Technical superuser of the HDI created at installation time • `_SYS_DI_TO` Owns transaction and connections of all internal HDI transactions
Password Specification	Not applicable These are technical database users. It is not possible to log on with these users.

Technical Users for HDI Schema-Based Containers

The deployment of database objects with SAP HANA Deployment Infrastructure (HDI) is based on a container model where each container corresponds roughly to a database schema. Each schema, and the database objects deployed into the schema, are owned by a dedicated technical database user.

For every container deployed, a new technical database user and schema with the same name as the container are created. Additional schemas and technical users required for metadata and deployment APIs are also created.

These technical database users are used internally by HDI only. As a general rule, these users are created as restricted database users who do not have any privileges by default (not even the role `PUBLIC`), and cannot be used to log on to the database. The only exception to this rule concerns user `S#00` who, from SAP HANA 2.0 SPS 02 revision 20, is granted the role `PUBLIC` by default.

> → Tip
>
> For more information about HDI security, see *SAP HANA Deployment Infrastructure (HDI) Reference for SAP HANA Platform* in *Related Information* below.

Related Information

Predefined XS Advanced Users [page 347]
File and Directory Permissions with High Isolation
Change the SID of an SAP HANA System
Local Secure Store (LSS) [page 253]
Deactivate the SYSTEM User [page 97]
Resetting the SYSTEM User Password
Maintaining the SAP HANA XS Classic Model Run Time
HANA_XS_BASE [page 422]
SQL Connection Details
The Statistics Service
SAP HANA Business Function Library (BFL)

SAP HANA Smart Data Integration and SAP HANA Smart Data Quality
Analyzing Statement Performance
Capturing and Replaying Workloads
Maintaining SAP HDI Containers
SAP HANA Developer Guide for XS Advanced Model (SAP Web IDE)

6.4.1 Deactivate the SYSTEM User

As the most powerful database user, `SYSTEM` is not intended for use in production systems. Use it to create lesser privileged users for particular purposes and then deactivate it.

Prerequisites

You have the system privilege `USER ADMIN`.

Context

It is highly recommended that you do not use `SYSTEM` for day-to-day activities in production environments. Instead, use it to create database users with the minimum privilege set required for their duties (for example, user administration, system administration). Then deactivate `SYSTEM`. You may temporarily reactivate the `SYSTEM` user for emergency or bootstrapping tasks.

> ⓘ Note
>
> The `SYSTEM` user is not required to update the SAP HANA database system; a lesser-privileged user can be created for this purpose. However, to upgrade SAP support package stacks, SAP enhancement packages and SAP systems using the Software Update Manager (SUM) and to install, migrate, and provision SAP systems using the Software Provisioning Manager (SWPM), the `SYSTEM` user **is required** and needs to be temporarily reactivated for the duration of the upgrade, installation, migration or provisioning.

Procedure

Execute the following statement:

> {·} Code Syntax

```
ALTER USER SYSTEM DEACTIVATE USER NOW;
```

Results

The SYSTEM user is deactivated and can no longer **connect** to the SAP HANA database. However, it may appear as though SYSTEM is still active in the system (for example when a procedure that was created by SYSTEM with DEFINER MODE is called).

You can verify that user SYSTEM is in fact deactivated in the USERS system view. For user SYSTEM, check the values in the columns USER_DEACTIVATED, DEACTIVATION_TIME, and LAST_SUCCESSFUL_CONNECT.

> **Note**
>
> You can still use the SYSTEM user as an emergency user even if it has been deactivated. Any user with the system privilege USER ADMIN can reactivate SYSTEM with the statement ALTER USER SYSTEM ACTIVATE USER NOW. To ensure that an administrator does not do this surreptitiously, it is recommended that you create an audit policy monitoring ALTER USER statements. Also change the password of the SYSTEM user after reactivating it.

Related Information

ALTER USER Statement (Access Control)
Create a Lesser-Privileged Database User for Update
Create an Audit Policy

6.5 Predefined Operating System Users

As part of the installation process, the operating system users <sid>adm and <sid>crypt are automatically created.

<sid>adm

User	<sid>adm, where <sid> is the ID of the SAP HANA system
Description	The <sid>adm user is an operating system user and is also referred to as the operating system administrator. This operating system user has unlimited access to all local resources related to SAP systems. This user is not a database user but a user at the operating system level. **Note** In a system configured for high isolation, additional OS users exist for every tenant database. Access to database-specific data is limited accordingly.

Password Specification	The initial password is specified during installation by your hardware partner or certified administrator. After handover, it is important that you change this password. A system administrator can do this at the operating system level. It is also possible as part of a system rename with SAP HANA lifecycle manager.
More Information	For more information about file and directory permissions in systems configured for high isolation, see the *SAP HANA Administration Guide*.

`<sid>`**crypt**

User	`<sid>crypt`, where `<sid>` is the ID of the SAP HANA system
Description	The `<sid>crypt` is an operating system user that is created if the local secure store (LSS) has been installed. The LSS is used to securely store and manage encryption keys, encryption root keys, and other similarly sensitive data, such as security-relevant configuration settings. `<sid>crypt` owns the storage of the encryption keys and encryption con configuration. This allows operating system-level duties to be strictly separated between system administrators and encryption key administrators and, therefore, lowers the level of trust required in a single operating system user. `<sid>crypt` can read the encryption keys but not the encrypted data.
Password Specification	The initial password is specified during installation by your hardware partner or certified administrator. After handover, it is important that you change this password. A system administrator can do this at the operating system level.
More Information	See the section on the local secure store (LSS).

Related Information

Predefined Database Users [page 89]
Database Isolation [page 23]

7 SAP HANA Authentication and Single Sign-On

The identity of database users accessing SAP HANA is verified through a process called authentication. SAP HANA supports several authentication mechanisms that can be used for the integration of SAP HANA into single sign-on environments (SSO). The mechanisms used to authenticate individual users are specified as part of the user definition.

> **Note**
>
> For JDBC and ODBC client connections, user passwords are always transmitted in encrypted hashed form during the user authentication process, never in plain text. For HTTP connections via SAP HANA XS classic, HTTPS must be configured. In SSO environments, we recommend using encrypted communication channels for **all** client connections.

Related Information

SAP HANA Authorization [page 135]
SAP HANA User Management [page 73]

7.1 User Authentication Mechanisms

SAP HANA supports several authentication mechanisms. Mechanisms that are not required can be disabled.

Supported Authentication Mechanisms

Mechanism	Description	Can Be Used for SSO
SAP HANA user name and password	Users accessing the SAP HANA database authenticate themselves by entering their database user name and their local SAP HANA password.	No

Mechanism	Description	Can Be Used for SSO
Kerberos, SPNEGO	A Kerberos authentication provider can be used to authenticate users accessing SAP HANA in the following ways: • Directly from ODBC and JDBC database clients within a network (for example, the SAP HANA studio) • Indirectly from front-end applications such as SAP BusinessObjects applications and other SAP HANA databases using Kerberos delegation • Via HTTP/HTTPS access by means of SAP HANA Extended Services (SAP HANA XS), advanced model and classic model In this case, Kerberos authentication is enabled with Simple and Protected GSSAPI Negotiation Mechanism (SPNEGO). **Note** A user who connects to the database using an external authentication provider must also have a database user known to the database. The external identity is mapped to the identity of an internal database user.	Yes
Security assertion markup language (SAML)	A SAML bearer assertion can be used to authenticate users accessing SAP HANA directly from ODBC/JDBC database clients. SAP HANA can act as a service provider to authenticate users accessing via HTTP/HTTPS by means of SAP HANA XS classic. **Note** A user who connects to the database using an external authentication provider must also have a database user known to the database. The external identity is mapped to the identity of an internal database user.	Yes
X.509 client certificates	Users can be authenticated by client certificates signed by a trusted Certification Authority (CA) for: • HTTP/HTTPS access by means of SAP HANA XS, advanced model and classic model • JDBC/ODBC client access **Note** To implement X.509 client certificates, the user specified in the certificate must already exist in SAP HANA; there is no support for user mapping.	Yes

Mechanism	Description	Can Be Used for SSO
JSON Web Token (JWT)	A JSON Web Token can be used to authenticate users accessing SAP HANA directly from ODBC/JDBC database clients or indirectly through SAP HANA extended application services, advanced model (SAP HANA XS, advanced). **Note** A user who connects to the database using an external authentication provider must also have a database user known to the database. The external identity is mapped to the identity of an internal database user.	Yes
LDAP	A password stored in an LDAP directory server can be used to authenticate users accessing SAP HANA directly from ODBC/JDBC database clients, if authentication using users' local SAP HANA authentication has been disabled. **Note** A user who connects to the database using an external authentication provider must also have a database user known to the database. The external identity and the database user name are the same. If the LDAP provider is enabled to create database users in SAP HANA, the required user is created automatically if it doesn't exist.	No
Logon and assertion tickets	Users can be authenticated by SAP logon or assertion tickets issued to them when they log on to an SAP system that is configured to create tickets (for example, the SAP Web Application Server or Portal). **Note** To implement logon/assertion tickets, the user specified in the logon/assertion ticket must already exist in SAP HANA; there is no support for user mapping.	Yes
Session cookies	Session cookies are not technically an authentication mechanism. However, they reconnect users who have already been authenticated by Kerberos, SAML or JWT and extend the validity period of logon and assertion tickets.	Yes

Note

For ODBC/JDBC access, by default, the SAP HANA database and the client interfaces choose different authentication methods based on the credentials supplied. You can explicitly limit the authentication methods used by providing a set of allowed authentication methods in the `authenticationMethods` parameter as part of the connection string. For more information, see the *SAP HANA Client Interface Programming Reference*.

Isolated Single Sign-On for Tenant Databases

Separate, database-specific authentication is possible for every certificate-based authentication mechanism since it is possible to create different certificate collections for individual purposes directly in every database, and every database can have its own key pair and public key certificate.

For SAML assertions, X.509 certificates (for HTTP access via SAP HANA XS classic model), and logon tickets, it is also possible to use certificate collections (or PSEs) located on the file system. It is still possible to configure different trust and key stores for every database in the `global.ini` file. However, bear the following points in mind:

- If different trust and key stores are not explicitly configured for tenant databases, the same ones will be used for all external communication channels (including HTTP) for all databases.

  > ⚠ Caution
  >
  > If you have configured in tenant databases or the system database single sign-on mechanisms that rely on trust stores located in the file system (such as SAP logon and assertion tickets or SAML) and the trust stores are shared, users of one tenant database may be able to log on to other databases in the system.

- By default, only the system administrator can configure separate trust and key stores for tenant databases by changing the relevant properties in the `global.ini` file. This is because tenant database administrators are prevented from changing any communication properties. They are in the default configuration change blocklist (`multidb.ini`).

For more information about certificate collections in the database and PSEs in the file system, see the section on certificate management.

For Kerberos-based authentication, a per-database configuration is not possible – databases users in all databases must be mapped to users in the same Key Distribution Center.

Disabling Authentication Mechanisms

Authentication methods have to be enabled individually for each user. For additional security, it is possible to disable authentication mechanisms globally for all users. You do this by configuring the parameter `[authentication] authentication_methods` in the `global.ini` configuration file. The value of this parameter specifies all enabled methods as a comma-separated list.

The default value is
`pbkdf2,password,kerberos,spnego,saml,saplogon,x509xs,jwt,sessioncookie,x509,ldap`.

> ⓘ Note
>
> **`pbkdf2`** refers to password authentication using passwords hashed with PBKDF2 (Password-Based Key Derivation Function 2), while **`password`** refers to password authentication using passwords hashed with secure hash algorithm SHA-256. Even if **`password`** is listed as a supported authentication method here, a user will not be able to log on with a SHA-256 hashed password if this hash method has been disabled with the parameter `[authentication] password_hash_methods`.

> **Note**
>
> If you are using SAP HANA dynamic tiering, it is not possible to disable logon and assertion tickets (`saplogon`) as an authentication mechanism.

Changes to the parameter `authentication_methods`are audited by default if auditing is enabled.

Related Information

Password Policy [page 105]
Single Sign-On Integration [page 116]
LDAP User Authentication [page 130]
Certificate Management in SAP HANA [page 303]
Actions Audited by Default Audit Policy [page 283]
Maintaining Single Sign-On for XS Advanced Applications
Maintaining Single Sign-On for XS Classic Applications
JDBC Connection Properties
ODBC Connection Properties

7.2 Logon Checks

Before a user can connect to an SAP HANA instance, the system performs several checks as part of the logon process.

1. The database authenticates the user using the configured mechanism.
   For example, if user name and password authentication is being enforced, the user name and password are verified.
2. The database verifies for connect restrictions on the user.
   Certain connect restrictions can stop the user from connecting to an SAP HANA instance, for example because the user's IP address is not part of the list of allowed addresses.
3. The database verifies that the user's account is within its validity period.
   In the system view USERS, the columns VALID_FROM and VALID_UNTIL must contain effective values for the user in question.
   The validity period is an optional parameter that a user administrator can set during user provisioning.
4. The database verifies that the user account is active.
   In the system view USERS, the column IS_DEACTIVATED must contain the value FALSE for the user in question.
   User accounts may be deactivated explicitly by a user administrator or by the system, for example, due to too many invalid logon attempts.

If all of the above checks are successful, the user is logged on to the SAP HANA database.

Related Information

CONNECT Statement (Session Management)
Connect Restrictions [page 80]

7.3 Troubleshooting Authentication Problems

Troubleshoot authentication errors occurring during connection using a client or SQL statement.

SAP HANA maintains a list of failed authentication attempts and the reasons for authentication failures in the system view SYS.AUTHENTICATION_ERROR_DETAILS. The maximum number of entries and the maximum time an entry is kept in the table can be configured using the ini file parameters `'authentication_error_details_retain_records'` and `'authentication_error_details_retain_duration'` in the `'authentication'` section. Outdated entries will be regularly removed.

> **Note**
>
> On installation, only DBADMIN can access this system view. DBADMIN can also grant the SELECT and DELETE privileges on the system view SYS.AUTHENTICATION_ERROR_DETAILS to other users.

Related Information

CONNECT Statement (Session Management)

7.4 Password Policy

The passwords of database users are subject to certain rules. These are defined in the password policy.

The SAP HANA database comes with a default password policy. You can change this default password policy in line with your organization's security requirements. If you manage users in user groups, you can also configure group-specific password policies.

Database Password Policy

The password policy of the database is defined by parameters in the `password policy` section of the `indexserver.ini` configuration file for tenant databases and the `nameserver.ini` configuration file for the system database.

The database password policy is valid for all database users unless the user is in a user group with its own dedicated password policy.

> ⚠ Caution
>
> Do not change configuration files directly. Such changes cannot be audited.

You can view and change the database password policy on the *Password Policy and Exclude List* page of the SAP HANA cockpit.

You can also query the system view M_PASSWORD_POLICY to see the current database password policy.

User Group Password Policy

If the users of a user group have different password requirements, you can configure group-specific values for the individual parameters of the password policy in the definition of the user groups.

> Example
>
> ```
> CREATE USERGROUP Training SET PARAMETER 'password_layout' = 'A1a!',
> 'minimal_password_length' = '16' ENABLE PARAMETER SET 'password policy';
> ```

> ⓘ Note
>
> If a group-specific value is not explicitly set for a parameter, the value configured in the password policy of the database appears as the user group value in USERGROUP_PARAMETERS.

You can configure the password policy for a user group in the SAP HANA cockpit.

> → Remember
>
> The parameter set `password policy` must be enabled for the user group in order for configured password policy parameters to take effect.

Effective Password Policy

To determine which password policy a user is currently subject to, query the system view M_EFFECTIVE_PASSWORD_POLICY.

> {·} Sample Code
>
> ```
> SELECT * from "PUBLIC"."M_EFFECTIVE_PASSWORD_POLICY" where USER_NAME =
> '<user_name>';
> ```

Related Information

Configure the Database Password Policy and Password Exclude List
Configure a Password Policy for a User Group
User Groups [page 76]
M_PASSWORD_POLICY System View

7.4.1 Password Policy Configuration Options

The password policy is defined by parameters in the `password policy` section of the `indexserver.ini` configuration file (tenant databases) or `nameserver.ini` configuration file (system database). Password policy parameters may also be individually configured in the definition of a user group.

The following sections describe these parameters, which correspond to the configuration options available in the SAP HANA cockpit.

- Minimum Password Length [page 107]
- Lowercase Letters/Uppercase Letters/Numerical Digits/Special Characters Required [page 108]
- Password Change Required on First Logon [page 109]
- Number of Last Used Passwords That Cannot Be Reused [page 109]
- Number of Allowed Failed Logon Attempts [page 110]
- User Lock Time [page 111]
- Minimum Password Lifetime [page 111]
- Maximum Password Lifetime [page 111]
- Lifetime of Initial Password [page 112]
- Maximum Duration of User Inactivity [page 112]
- Notification of Password Expiration [page 113]
- Exempt SYSTEM User from Locking [page 113]
- Detailed Error Information on Failed Logon [page 114]

Minimum Password Length

The minimum number of characters that the password must contain

Parameter	`minimal_password_length`
Default Value	8 (characters)
Additional Information	You must enter a value between 6 and 64.
UI Label	*Minimum Password Length*

Lowercase Letters/Uppercase Letters/Numerical Digits/ Special Characters Required

The character types that the password must contain and how many

Parameter	`password_layout`
Default Value	Aa1, that is, at least one uppercase letter, at least one number, and at least one lowercase letter
Additional Information	The following character types are possible: • Lowercase letter (a-z) • Uppercase letter (A-Z) • Numerical digits (0-9) • Special characters (underscore (_), hyphen (-), and so on) Any character that is not an uppercase letter, a lowercase letter, or a numerical digit is considered a special character. The following formats are supported for passwords: `<password> ::= { <letter> [ { <letter_or_digit> | # | $ }[…] ] | <digit> [ <letter_or_digit> […] ] | <any_quoted_string> }` If configuring this option in the `indexserver.ini` file using the `password_layout` parameter, you can use any specific letters, numbers and special characters, and the characters can be in any order. For example, the default value example could also be represented by **`a1A`**, **`hQ5`**, or **`9fG`**. To enforce the use of at least one of each character type including special characters, you specify **`A1a_`** or **`2Bg?`**. To enforce the use of a specific number of a particular character type, specify the character type multiple times. For example, if passwords must contain at least 3 digits, you could specify the layout with **`a123A`** or **`789fG`**. **Note** Passwords containing special characters other than underscore must be enclosed in double quotes ("). The SAP HANA studio does this automatically. When a password is enclosed in double quotes ("), any Unicode characters may be used. **Caution** The use of passwords enclosed in double quotes (") may cause logon issues depending on the client used. The SAP HANA studio and `hdbsql` support passwords enclosed in double quotes (").
UI Labels	*Lowercase Letters/Uppercase Letters/Numerical Digits/Special Characters Required*

Password Change Required on First Logon

Defines whether users have to change their initial passwords immediately the first time they log on

Parameter	`force_first_password_change`
Default Value	True
Additional Information	If this parameter is set to **`true`**, users can still log on with the initial password but every action they try to perform will return the error message that they must change their password. If this parameter is set to **`false`**, users are not forced to change their initial password immediately the first time they log on. However, if a user does not change the password before the number of days specified in the parameter `maximum_unused_initial_password_lifetime`, then the password still expires and must be reset by a user administrator. A user administrator (that is, a user with the system privilege USER ADMIN) can force a user to change his or her password at any time with the following SQL statement: `ALTER USER <user_name> FORCE PASSWORD CHANGE` A user administrator can override this password policy setting for individual users (for example, technical users) with the following SQL statement: • `CREATE USER <user_name> PASSWORD <password> [NO FORCE_FIRST_PASSWORD_CHANGE]` • `ALTER USER <user_name> PASSWORD <password> [NO FORCE_FIRST_PASSWORD_CHANGE]` **Note** This parameter is only valid for users connecting with their SAP HANA database user name and password. It is not valid for connections established through other authentication mechanisms.
UI Label	*Password Change Required on First Logon*

Number of Last Used Passwords That Cannot Be Reused

The number of last used passwords that the user is not allowed to reuse when changing his or her current password

Parameter	`last_used_passwords`
Default Value	5 (previous passwords)
Additional Information	If you enter the value **`0`**, the user can reuse his or her old password.
UI Label	*Number of Last Used Passwords That Cannot Be Reused*

Number of Allowed Failed Logon Attempts

The maximum number of failed logon attempts that are permitted; the user is locked as soon as this number is reached

Parameter	`maximum_invalid_connect_attempts`
Default Value	6 (failed logon attempts)
Additional Information	You must enter a value of at least **1**. As long as the number of failed logon attempts does not exceed the permitted number of failed attempts, the timestamp of the last failed attempt is recorded in column LAST_INVALID_CONNECT_ATTEMPT of system view SYS.USERS. Once the user is locked (after having exceeded the permitted number of failed logon attempts), this timestamp is no longer updated. The timestamp is used as the starting time for calculating how long the user is locked (`password_lock_time`). A user administrator can reset the number of invalid logon attempts with the following SQL statement: `ALTER USER <user_name> RESET CONNECT ATTEMPTS` The first time a user logs on successfully after an invalid logon attempt, an entry is made in the INVALID_CONNECT_ATTEMPTS system view containing the following information: • The number of invalid logon attempts since the last successful logon • The time of the last successful logon A user administrator can delete information about invalid logon attempts with the following SQL statement: `ALTER USER <user_name> DROP CONNECT ATTEMPTS` **→ Recommendation** Create an audit policy to log activity in the INVALID_CONNECT_ATTEMPTS system view. For example, create an audit policy that logs data query and manipulation statements executed on this view. **Note** If `password_lock_for_system_user` is set to **`false`**, the SYSTEM user will not be locked regardless of the number of failed logon attempts.
UI Label	*Number of Allowed Failed Logon Attempts*

User Lock Time

The number of minutes for which a user is locked after the maximum number of failed logon attempts

Parameter	`password_lock_time`
Default Value	1440 (minutes)
Additional Information	If you enter the value **`0`**, the user is unlocked immediately. This disables the functionality of parameter `maximum_invalid_connect_attempts`. The value in column LAST_INVALID_CONNECT_ATTEMPT of system view SYS.USERS is used as the start time for calculating how long the user is locked. A user administrator can reset the number of invalid logon attempts and re-activate the user account with the following SQL statement: `ALTER USER <user_name> RESET CONNECT ATTEMPTS`. It is also possible to reactivate the user in the user editor of the SAP HANA studio. To lock a user indefinitely, enter the value **`-1`**. On the *Password Policy and Exclude List* page of the SAP HANA cockpit, this corresponds to selecting the *Lock User Indefinitely* checkbox. The user remains locked until reactivated by a user administrator as described above.
UI Label	*User Lock Time*

Minimum Password Lifetime

The minimum number of days that must elapse before a user can change his or her password

Parameter	`minimum_password_lifetime`
Default Value	1 (day)
Additional Information	If you enter the value **`0`**, the password has no minimum lifetime.
UI Label	*Minimum Password Lifetime*

Maximum Password Lifetime

The number of days after which a user's password expires

Parameter	`maximum_password_lifetime`
Default Value	182 (days)

Additional Information	You must enter a value of at least **1**. A user administrator can exclude users from this password check with the following SQL statement: `ALTER USER <user_name> DISABLE PASSWORD LIFETIME`. However, this is recommended for technical users only, not database users that correspond to real people. A user administrator can re-enable the password lifetime check for a user with the following SQL statement: `ALTER USER <user_name> ENABLE PASSWORD LIFETIME`.
UI Label	*Maximum Password Lifetime*

Lifetime of Initial Password

The number of days for which the initial password or any password set by a user administrator for a user is valid

Parameter	`maximum_unused_initial_password_lifetime`
Default Value	7 (days)
Additional Information	You must enter a value of at least **1**. If a user has not logged on using the initial password within the given period of time, the user will be deactivated until their password is reset. **Note** In SAP HANA 1.0 SPS 12 and earlier, this parameter was misspelled as `maximum_unused_inital_password_lifetime`. If this parameter had a user-specified value before upgrade, this value will be set as the value of the parameter `maximum_unused_initial_password_lifetime`. The misspelled parameter is unset and disappears from the custom configuration file.
UI Label	*Lifetime of Initial Password*

Maximum Duration of User Inactivity

The number of days after which a password expires if the user has not logged on

Parameter	`maximum_unused_productive_password_lifetime`
Default Value	365 (days)

Additional Information	You must enter a value of at least **1**. If a user has not logged on within the given period of time using any authentication method, the user will be deactivated until their password is reset.
UI Label	*Maximum Duration of User Inactivity*

Notification of Password Expiration

The number of days before a password is due to expire that the user receives notification

Parameter	`password_expire_warning_time`
Default Value	14 (days)
Additional Information	Notification is transmitted via the database client (ODBC or JDBC) and it is up to the client application to provide this information to the user. If you enter the value **0**, the user does not receive notification that his or her password is due to expire. The system also monitors when user passwords are due to expire and issues a medium priority alert (check 62). This may be useful for technical database users since password expiration results in the user being locked, which may affect application availability. It is recommended that you disable the password lifetime check of technical users so that their password never expires (`ALTER USER <technical_username> DISABLE PASSWORD LIFETIME`).
UI Label	*Notification of Password Expiration*

Exempt SYSTEM User from Locking

Indicates whether or not the user SYSTEM is locked for the specified lock time (`password_lock_time`) after the maximum number of failed logon attempts (`maximum_invalid_connect_attempts`)

Parameter	`password_lock_for_system_user`
Default Value	true
Additional Information	This parameter cannot be configured for a user group.
UI Label	*Exempt SYSTEM User from Locking*

Detailed Error Information on Failed Logon

Indicates the detail level of error information returned when a logon attempt fails

Parameter	`detailed_error_on_connect`
Default Value	false
Additional Information	If set to **`false`**, only the information `authentication failed` is returned. If set to **`true`**, the specific reason for failed logon is returned: • `Invalid user or password` • `User is locked` • `Connect try is outside validity period` • `User is deactivated`
UI Label	*Detailed Error Information on Failed Logon*

Related Information

Configure the Database Password Policy and Password Exclude List
Create an Audit Policy

7.4.2 Password Exclude List

A password exclude list is a list of words that are not allowed as passwords or parts of passwords. A password exclude list can be managed for every database individually.

The password exclude list in SAP HANA is implemented with the table _SYS_PASSWORD_BLACKLIST in the schema _SYS_SECURITY. This table is empty when you create a new instance.

You can enter words in the password exclude list as part of password policy configuration on the *Password Policy and Exclude List* page of the SAP HANA cockpit. Alternatively, you can add words to table directly using the INSERT statement. For more information about the structure of this table, see _SYS_PASSWORD_BLACKLIST.

Sample Code

```
INSERT INTO _SYS_SECURITY._SYS_PASSWORD_BLACKLIST VALUES ('sap', 'TRUE',
'FALSE');
```

> **Note**
>
> Changes to the password exclude list do not affect the existing passwords of users. The modified exclude list applies the next time the user changes his or her password.

> **Note**
>
> Object privileges SELECT, INSERT, and DELETE on the _SYS_PASSWORD_BLACKLIST table are required to edit the password exclude list.

Related Information

Configure Password Policies
_SYS_PASSWORD_BLACKLIST [page 115]

7.4.3 _SYS_PASSWORD_BLACKLIST

Shows the password exclude list

Column Name	Data Type	Description
BLACKLIST_TERM	NVARCHAR (256)	Word or partial word on the exclude list
CHECK_PARTIAL_PASSWORD	VARCHAR (6)	Specifies whether passwords that contain the word are excluded (TRUE) or only passwords that match the word exactly are excluded (FALSE) **Note** Either TRUE or FALSE must be specified during an insert.
CHECK_CASE_SENSITIVE	VARCHAR (6)	Specifies whether the excluded word is case sensitive (TRUE) or case insensitive (FALSE) **Note** Either TRUE or FALSE must be specified during an insert.

> **Sample Code**
>
> Exclude a new word:

```
INSERT INTO _SYS_SECURITY._SYS_PASSWORD_BLACKLIST VALUES ('sap', 'TRUE',
'FALSE');
```

Sample Code

Remove an excluded word::

```
DELETE FROM _SYS_SECURITY._SYS_PASSWORD_BLACKLIST WHERE BLACKLIST_TERM =
'sap';
```

Related Information

Password Exclude List [page 114]

7.5 Single Sign-On Integration

Integrate SAP HANA into single sign-on (SSO) environments.

Single Sign-On Using Kerberos [page 116]
For integration into Kerberos-based SSO scenarios, SAP HANA supports Kerberos version 5 based on Active Directory (Microsoft Windows Server) or Kerberos authentication servers. For HTTP access using SAP HANA Extended Services (SAP HANA XS) advanced and classic, Kerberos authentication is enabled with Simple and Protected GSSAPI Negotiation Mechanism (SPNEGO).

Single Sign-On Using SAML 2.0 [page 118]
SAP HANA supports the Security Assertion Markup Language (SAML) for user authentication in single sign-on environments. SAML is used for authentication purposes only and not for authorization.

Single Sign-On Using SAP Logon and Assertion Tickets [page 122]
Users can be authenticated in SAP HANA by logon or assertion tickets issued to them when they log on to an SAP system configured to create tickets (for example, the SAP Web Application Server or Portal).

Single Sign-On Using JSON Web Tokens [page 123]
SAP HANA supports JSON Web Tokens (JWT) for user authentication in single sign-on environments.

7.5.1 Single Sign-On Using Kerberos

For integration into Kerberos-based SSO scenarios, SAP HANA supports Kerberos version 5 based on Active Directory (Microsoft Windows Server) or Kerberos authentication servers. For HTTP access using SAP HANA Extended Services (SAP HANA XS) advanced and classic, Kerberos authentication is enabled with Simple and Protected GSSAPI Negotiation Mechanism (SPNEGO).

Kerberos is a network authentication protocol that provides authentication for client-server applications across an insecure network connection using secret-key cryptography.

ODBC and JDBC database clients support the Kerberos protocol, for example, the SAP HANA studio. Access from front-end applications (for example, SAP BusinessObjects XI applications) can also be implemented using

Kerberos delegation. Support for constrained delegation and protocol transition is limited to scenarios in which the middle-tier application connects to SAP HANA as the database layer via JDBC.

Kerberos is supported for HTTP access using SAP HANA XS advanced and classic with Simple and Protected GSSAPI Negotiation Mechanism (SPNEGO). It is up to the HTTP client whether it uses Kerberos directly or SPNEGO.

> → Recommendation
>
> To avoid replay attacks, we recommend that you set up secure communication between the individual components of the SAP HANA database and client connections using the secure sockets layer (SSL) protocol when implementing Kerberos authentication, in particular when using Kerberos with insecure encryption algorithms such as RC4.

Configuration

To allow users to log on to the SAP HANA database from a client using Kerberos authentication, the following configuration steps are necessary:

1. Install MIT Kerberos client libraries on the host(s) of the SAP HANA system.
2. Configure the SAP HANA system for Kerberos and/or SPNEGO authentication.
3. Map SAP HANA database users to their external identities stored in the Kerberos key distribution center (KDC).

   > → Remember
   >
   > A per-database configuration is not possible – databases users in all databases must be mapped to users in the same KDC.

For more information about how to set up SSO with SAP HANA using Kerberos and Microsoft Active Directory, see SAP Note 1837331.

In distributed SAP HANA systems that use Kerberos delegation (SSO2DB), application disruptions resulting from expired authentication are avoided though the use of session cookies. This mechanism is active by default but can be disabled in the `indexserver.ini` configuration file with the `[authentication] session_cookie_for_kerberos` property.

Related Information

Configure Kerberos for SAP HANA Database Hosts
SAP HANA XS Application Authentication
SAP Note 1837331
SAP Note 2354473
SAP Note 1813724
SAP Note 2354556

7.5.2 Single Sign-On Using SAML 2.0

SAP HANA supports the Security Assertion Markup Language (SAML) for user authentication in single sign-on environments. SAML is used for authentication purposes only and not for authorization.

SAML provides the mechanism by which the identity of users accessing the SAP HANA database from client applications is authenticated by XML-based assertions issued by a trusted identity provider. The internal database user to which the external identity is mapped is used for authorization checks during the database session.

SAML can be implemented to authenticate users accessing the SAP HANA database from the following client applications:

- Database clients that access the SQL interface of the SAP HANA database directly
  This covers standard ODBC and JDBC database clients.
  In this scenario, a SAML bearer assertion is used to authenticate the user directly. It is the client application's responsibility to retrieve the SAML bearer assertion used for logon. To log on using a SAML bearer assertion, you must set the user name to an empty string and the SAML bearer assertion as the password in your ODBC/JDBC connection properties.

  > ⓘ Note
  >
  > The SAP HANA studio does not support SAML.

- Clients that connect to SAP HANA through the SAP HANA XS classic server via HTTP
  In this scenario, SAP HANA acts as the service provider that authenticates users on the basis of their SAML bearer assertion.

> → Recommendation
>
> To avoid replay attacks, we recommend that you set up secure communication between the individual components of the SAP HANA database and client connections using the Transport Layer Security (TLS) protocol when implementing SAML authentication.

SAML Assertion Specification

SAP HANA supports plain SAML 2.0 assertions as well as unsolicited SAML responses that include an unencrypted SAML assertion. SAML assertions and responses must be signed using XML signatures.

The following features of XML signatures are supported:

- SHA1, SHA256, SHA384, and SHA512 for hash algorithms
- RSA-SHA1, RSA-SHA256, RSA-SHA384 and RSA-SHA512 as signature algorithms

> ⓘ Note
>
> The configuration parameter `[authentication] saml_signature_hash_types` can be used to restrict the signature algorithms used (for example, disable use of SHA1). The default value of this parameter is `sha1,sha256,sha384,sha512`.

The following SAML assertion features are supported:

- Assertion Subject with `NameID`
- Qualified `NameID` with `SPProvidedID` and `SPNameQualifier`
- Validity conditions (`NotBefore`, `NotOnOrAfter`)
- Audience restrictions
  The value of the parameter `[authentication] saml_service_provider_name` determines whether or not an audience restriction will be validated. If the parameter is not set, an SAP HANA system ignores a possible audience restriction tag in a SAML assertion. By default, in SAP HANA tenant databases the parameter is set to the tenant name and the instance ID, for example, 'TENANT01'. For the system database, the parameter is empty by default.

The following properties of a SAML assertion are evaluated to log on the requesting user to SAP HANA:

Property	Note
`saml:Assertion/@Version`	Required entry: 2.0
`saml:Subject/saml:NameID`	Must be specified
`saml:Subject/saml:NameID/@Format`	Optional entry If present, entry can be `urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified` or `"urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress"`
`saml:Subject/saml:NameID/@SPProvidedID`	Must either match an explicit user mapping in the SAP HANA database, or a wildcard mapping must be configured for the user
`saml:Subject/saml:SubjectConfirmation`	Optional If present, entry must be `{{"urn:oasis:names:tc:SAML:2.0:cm:bearer"}}`
`saml:Conditions` • `@NotBefore` • `@NotOnOrAfter` • `AudienceRestriction`	Condition `@NotOnOrAfter` must be set.

SAML Authentication with Automatic User Creation

If you use an LDAP-compliant directory server to manage users and their access to resources, you can leverage this infrastructure to provision the database users required for SAML-authenticated users.

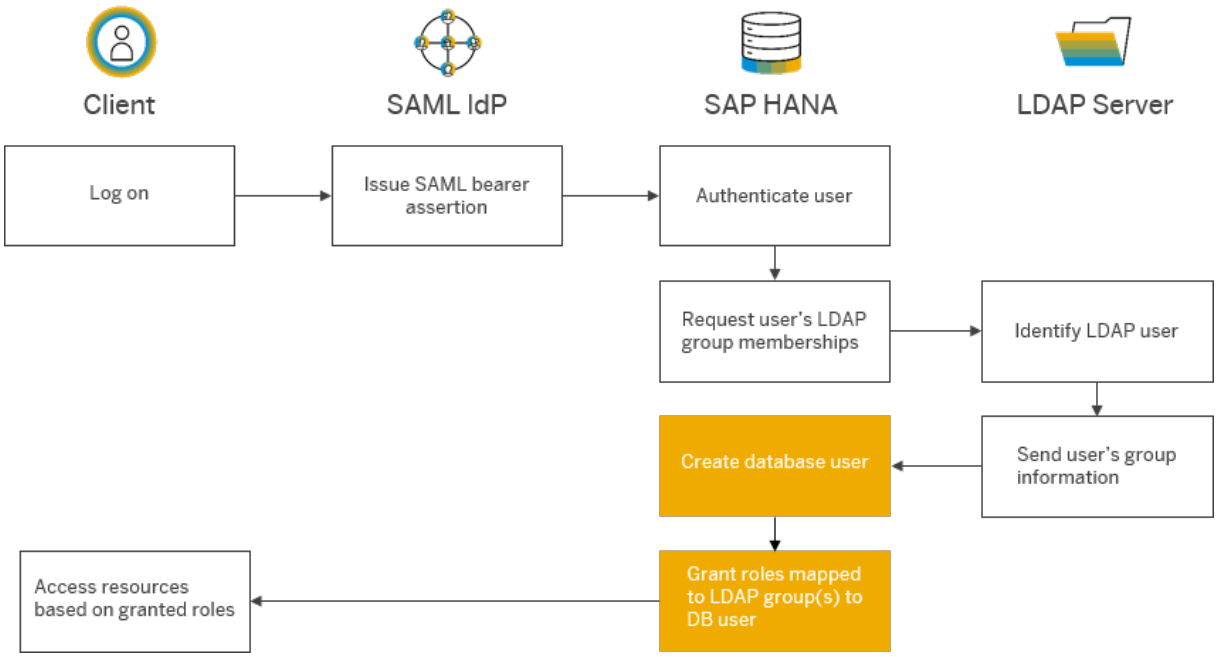
SAML identity providers in SAP HANA may create SAP HANA database users if the following conditions are met:

- LDAP group authorization is configured in the SAP HANA database.
- The SAML identity provider is configured for automatic user creation.

- The SAML-authenticated user exists in your LDAP server and this user is a member of at least one LDAP group mapped to at least one SAP HANA role.

By default, a standard user is created in the specified user group. It may be a restricted user if the SAML provider is configured accordingly. The new database user is automatically configured for SAML authentication and LDAP group authorization (authorization mode `LDAP`).

The following figure illustrates how SAML authentication with automatic user creation works.



SAML Authentication with Automatic User Creation

Configuration for ODBC/JDBC Client Access

To enable logon using SAML bearer assertions, you must configure identity providers and then map them to the required database users. Two types of user mapping are supported:

- SAP HANA-based user mappings
  An external identity is mapped to a database user explicitly in the SAP HANA in the user definition for each identity provider.
  The corresponding assertion subject looks like this:
  ```
  <NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-
  format:unspecified">zgc2VLavgYy4hsohfYPM21</NameID>
  ```

  > **Note**
  > The matching of external identities to database users can be case sensitive or case insensitive. You can configure this when you create the SAML identity provider in SAP HANA.

  Mapping to a database user's e-mail address is also possible. The corresponding assertion subject looks like this:
  ```
  <NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress">
  ```
- Identity provider-based user mappings

The identity provider maps its users to SAP HANA database users and provides this information using the SPProvidedID attribute. This "wildcard" mapping is configured for a database user in the user definition using the keyword `ANY`.
For more information, see the CREATE USER statement in the *SAP HANA SQL and System Views Reference*.
The corresponding assertion subject looks like this:

```
<NameIDFormat="urn:oasis:names:tc:SAML:1.1:nameid- format:unspecified"
SPProvidedID="BILLG">zgc2VLavgYy4hsohfYPM21</NameID>
```

A database user can be both mapped to several providers and mapped to the same provider multiple times with different fixed identities or with keyword ANY.

You can configure SAML identity providers and configure user mapping in the SAP HANA cockpit.

In addition, you must configure the trust store used to validate incoming SAML assertions against certificates signed by a trusted Certification Authority (CA). We recommend creating a certificate collection with the purpose `SAML` that contains the required certificates directly in the database. If necessary, you can create separate collections for different identity providers. It is also possible to use a trust store located on the file system.

> **Note**
>
> If there are no applicable certificate collections with the purpose `SAML` available, the trust store located on the file system is used instead (`sapsrv.pse` by default)

> **Caution**
>
> We recommend creating certificate collections for individual purposes in the database directly, rather than using trust stores (PSE) in the file system. By default, the same PSE in the file system is shared by all databases for all external communication channels (including HTTP) and certificate-based authentication. Different PSEs must be explicitly configured for tenant databases.

Configuration for HTTP Client Access via SAP HANA XS Classic

While you can configure SAML providers for ODBC/JDBC-based SAML authentication using the SAP HANA studio or SQL statements, you should always use the SAP HANA XS Administration Tool to configure SAML providers that will be used for HTTP access via the classic XS server.

You also use the SAP HANA XS Administration Tool to configure an SAP HANA system to act as an SAML service provider. For more information about maintaining SAML providers for HTTP access via the SAP HANA XS classic server, see the *SAP HANA Administration Guide* .

Related Information

CREATE SAML PROVIDER Statement (Access Control)
Add a SAML Identity Provider
CREATE USER Statement (Access Control)

Create a Database User

Certificate Collections [page 307]

Create a Certificate Collection

Maintaining SAML Providers

SAP Note 2284620

7.5.3 Single Sign-On Using SAP Logon and Assertion Tickets

Users can be authenticated in SAP HANA by logon or assertion tickets issued to them when they log on to an SAP system configured to create tickets (for example, the SAP Web Application Server or Portal).

If you want to integrate an SAP HANA system into a landscape that uses logon or assertion tickets for user authentication, you must configure SAP HANA to accept logon/assertion tickets.

Trust Store Configuration

SAP HANA validates incoming logon/assertion tickets against certificates signed by a trusted Certification Authority (CA) stored in a dedicated trust store. This trust store must contain all root certificate(s) used to validate logon/assertion tickets. We recommend creating a certificate collection with the purpose **SAP LOGON** and the required certificates directly in the database.

It is also possible to use a trust store located in the file system. The default location of the trust store in the file system depends on the cryptographic library configured for SSL:

- `$SECUDIR/saplogon.pse` (CommonCryptoLib)

  > **Note**
  >
  > The saplogon.pse trust store is available automatically.

- `$HOME/.ssl/saplogon.pem` (OpenSSL)

  > **Note**
  >
  > **Deprecated:** OpenSSL is deprecated. You must migrate to CommonCryptoLib. For more information, see SAP Note 2093286.

> **Caution**
>
> By default, the same PSE in the file system is shared by all databases. Different PSEs must be explicitly configured for tenant databases.

You can configure the path to the trust store by setting the parameter `[authentication] saplogontickettruststore` in the `indexserver.ini` configuration file.

> **Note**
>
> You must restart SAP HANA after you change this parameter.

> **Note**
>
> This property is in the default configuration change blocklist (`multidb.ini`). This means that it cannot initially be changed in tenant databases. It must be changed from the system database. If appropriate for your scenario, you can remove this property from the change blocklist. For more information about how to edit the change blocklist, see the *SAP HANA Administration Guide*.

User Configuration

The user named in an incoming logon ticket must exist as a database user. The database user also must be configured for authentication using logon/assertion tickets. This can be done on the *User* page of the SAP HANA cockpit or in the user editor of the SAP HANA studio.

For more information about using logon tickets, see the SAP NetWeaver Library on SAP Help Portal.

Related Information

Configure Authentication Using SAP Logon Tickets and Assertions
Default Blacklisted System Properties in Tenant Databases [page 417]
SAP Note 2093286

7.5.4 Single Sign-On Using JSON Web Tokens

SAP HANA supports JSON Web Tokens (JWT) for user authentication in single sign-on environments.

The identity of users accessing the SAP HANA database from client applications can be authenticated by tokens issued by a trusted identity provider. The internal database user to which the external identity is mapped is used for authorization checks during the database session.

JWT can be implemented to authenticate users accessing the SAP HANA database from the following client applications:

- Database clients that access the SQL interface of the SAP HANA database directly
- Clients that connect to SAP HANA through the SAP HANA XS advanced server

> **→ Recommendation**
>
> To avoid replay attacks, we recommend that you set up secure communication between the individual components of the SAP HANA database and client connections using the Transport Layer Security (TLS) protocol when implementing JWT authentication.

JWT Structure

JSON Web Token (JWT) is an open standard. SAP HANA validates tokens according to the IETF standard, with the following restrictions and requirements.

Header

In the header part of the token, the following claims are evaluated:

- "alg" (algorithm)
  This required claim specifies the hashing algorithm used to generate the signature. The following algorithms are supported:
  - RS256
  - ES256
  - ES384
  - ES512
- "kid" (key ID)
  This optional claim contains a hint indicating which public key should be used to validate the signature of the token if keys are managed in a JSON Web Key Set (JWKS). This is the case if the issuer uses public keys without a surrounding certificate to verify tokens.
  If this claim is present in the header, SAP HANA looks for the indicated public key in the trust store used to validate incoming JWTs (certificate collection with purpose JWT). If no matching key is found, SAP HANA tries to validate the token with the other keys and certificates in the certificate collection.
- "typ" (type)
  If present, this claim must have the value "JWT".

The token header therefore looks like this:

```
{
  "alg": "RS256",
  "typ": "JWT",
  "kid": "<key_id_hint>"
}
```

Payload

SAP HANA can evaluate a number of claims in the payload part of the token.

The following standard claims are mandatory:

- "iss" (issuer)
  This claim is required to map the token to an identity provider configured in the SAP HANA database.
- "sub" (subject)
  This claim is required for mapping the database user to the external identity or user name.

SAP HANA supports any number of optional claims. Typical examples of optional claims include the following:

- "origin" (origin)
  This claim provides further information that may be required to map the token to an identity provider in the SAP HANA database (for example, if multiple identity providers with the same issuer exist).
- "aud" (audience)
  This claim contains the intended recipient(s) of the token (for example, application name or ID).

For access from SAP HANA applications, it is also possible to define a proprietary claim to set the XS_APPLICATIONUSER session variable during logon.

If present, the standard claims "nbf" (not before) and "exp" (expiration) are automatically evaluated.

JWT Authentication with Automatic User Creation

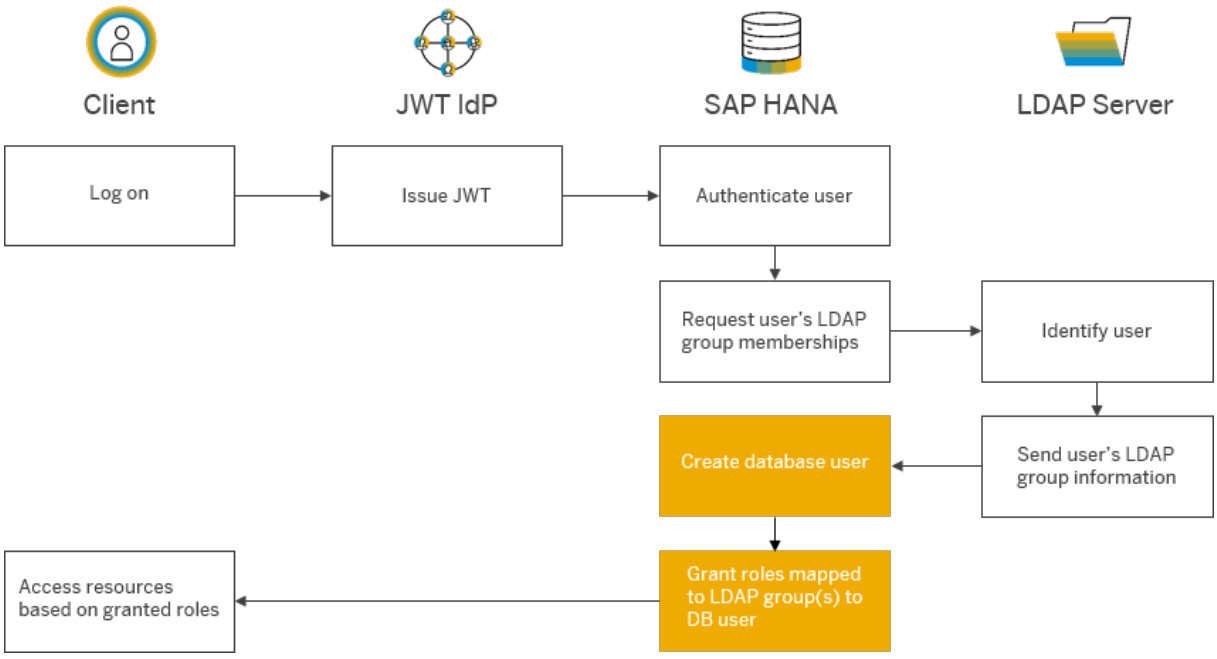
If you use an LDAP-compliant directory server to manage users and their access to resources, you can leverage this infrastructure to provision the database users required for JWT-authenticated users.

JWT identity providers in SAP HANA may create SAP HANA database users if the following conditions are met:

- LDAP group authorization is configured in the SAP HANA database.
- The JWT identity provider is configured for automatic user creation.
- The JWT-authenticated user exists in your LDAP server and this user is a member of at least one LDAP group mapped to at least one SAP HANA role.

By default, a standard user is created in the specified user group. It may be a restricted user if the JWT provider is configured accordingly. The new database user is automatically configured for JWT authentication and LDAP group authorization (authorization mode `LDAP`).

The following figure illustrates how JWT authentication with automatic user creation works.



JWT Authentication with Automatic User Creation

Configuration for ODBC/JDBC Access

Create Identity Providers

To enable the validation of incoming JSON Web Tokens, you must create identity providers in the SAP HANA database. In simple scenarios, an identity provider in SAP HANA corresponds to a single external identity provider (token issuer):

Sample Code

```
CREATE JWT PROVIDER my_jwt_provider WITH ISSUER 'http://example.com:8080/uaa/
oauth/token'
CLAIM 'sub' AS EXTERNAL IDENTITY
```

In some scenarios, multiple identity providers with the same issuer may be required but with different configurations for optional claims in the incoming token (for example, "origin", "aud", "<application user>").

Note

If the issuer is not unique, the information in the claims of an incoming token may not be sufficient to map the token unambiguously to an identity provider in SAP HANA. Therefore, identity providers with the same issuer must have a configured priority. During authentication, the identity provider with the highest priority is checked first. If the claims configured in the identity provider match those in the provided token, the identity provider is used for the authentication. If not, the identity provider with the next lower priority is checked.

Example

The XSUAA is used to authenticate business users to applications deployed in SAP BTP. To authenticate access to SAP HANA, the XSUAA issues a JWT, which must be mapped to an identity provider in SAP HANA for validation. For different applications, you create multiple identity providers in SAP HANA with the issuer XSUAA and use the optional claim configurations to ensure that the correct identity provider is used.

For example, to create an identity provider with a "chained" issuer (for example, a customer identity provider trusted by the XSUAA) and restricted to users from a particular application:

```
CREATE JWT PROVIDER prov_a WITH ISSUER 'http://xsuaa'
CLAIM 'origin' = 'http://customerA'
CLAIM 'aud' HAS MEMBER 'app1'
CLAIM 'sub' AS EXTERNAL IDENTITY
PRIORITY 100
```

To create an identity provider to be mapped if the application user claim is used in the token:

```
CREATE JWT PROVIDER prov_b WITH ISSUER 'http://xsuaa'
CLAIM 'sub' AS EXTERNAL IDENTITY
CLAIM 'appuser' AS APPLICATION USER
PRIORITY 110
```

Since both providers have the same issuer, the priority clause is used, with prov_b having the higher priority.

The claims in the following token match both prov_a and prov_b but since prov_b is checked first, it is used.

```
{ "iss": "http://xsuaa",
  "origin": "http://customerA",
  "sub": "testuser",
  "aud": ["app1", "app2"],
  "appuser": "test"
}
```

For more information about creating identity providers and configuring claims, see the CREATE JWT PROVIDER statement.

Configure User Mapping

Once identity providers exist in SAP HANA, you map users' external identities to the required database users. Two types of user mapping are supported:

- SAP HANA-based user mappings
  Database users are mapped explicitly to their external identities.

  > **Note**
  >
  > The matching of external identities to database users can be case sensitive or case insensitive. You can configure this when you create the JWT identity provider in SAP HANA.

- Identity provider-based user mappings
  The external identity provider maps its users to SAP HANA database users.

A database user can be both mapped to several providers and mapped to the same provider multiple times with different fixed identities or with keyword ANY.

SAP HANA-based user mappings can be configured in the user definition, for example, using the SAP HANA cockpit.

Trust Store for Certificate or Public Key Validation

Finally, you must create and configure the trust store used to validate incoming tokens. To do this, you create a certificate collection with the purpose `JWT` and add the certificates or public keys required for validation. You can create separate collections for different identity providers.

Related Information

https://tools.ietf.org/html/rfc7519
CREATE JWT PROVIDER Statement (Access Control)
Add a JWT Identity Provider
CREATE USER Statement (Access Control)
Create a Database User
Certificate Collections [page 307]
Create a Certificate Collection
User Administration and Authentication in SAP HANA XS Advanced [page 344]

7.6 X.509 Certificate-Based User Authentication

SAP HANA supports X.509 client certificates for user authentication in single sign-on (SSO) environments. In particular, X.509 certificate-based authentication can be used for technical users to secure system-to-system integration.

The identity of users accessing the SAP HANA database from client applications can be authenticated by X.509 client certificates issued by a trusted identity provider. The internal database user to which the external identity is mapped is used for authorization checks during the database session.

X.509 certificates can be used to authenticate users accessing the SAP HANA database from the following client applications:

- Database clients that access the SQL interface of the SAP HANA database directly
  This covers standard ODBC/JDBC database clients using the SAP Common Crypto Library.

  > ### Note
  >
  > The X.509 certificates used for authentication must have a short validity period (short lived) and cannot be revoked. They must also have a minimum RSA private key length of 2048 bits by default. This value can be configured with the parameter `[authentication] x509_rsa_min_key_length` in global.ini.

- Clients that connect to SAP HANA through SAP HANA Extended Application Services via HTTP
  For more information about configuring SSO for SAP HANA XS applications using X.509 certificates, see the *SAP HANA Administration Guide*.

Configuration for ODBC/JDBC Client Access

Create Identity Providers

To enable logon using X.509 certificates, you must first create an identity provider (certificate issuer).

> ### Sample Code
>
> ```
> CREATE X509 PROVIDER MyProvider WITH ISSUER 'CN=DigiCert Global Root CA,
> OU=www.digicert.com, O=DigiCert Inc, C=US';
> ```

> ### Sample Code
>
> ```
> CREATE X509 PROVIDER MyProvider WITH ISSUER 'CN=DigiCert Global Root G5,
> OU=www.digicert.com, O=DigiCert Inc, C=US';
> ```

The ISSUER field of the provider and the issuer distinguished name (DN) in the presented user certificate must be the same. For example, if an intermediate certificate is used to sign user certificates, the issuer DN of the intermediate certificate authority must be specified.

> ### Caution
>
> To avoid issues when commonly used browsers such as Mozilla Firefox and Google Chrome distrust older root certificates, DigiCert has started updating their first generation (G1) CA certificates to G5 certificates. For more information, see https://knowledge.digicert.com/general-information/digicert-root-and-intermediate-ca-certificate-updates-2023 and SAP Note 3327214.

User Mapping

Once the provider exists in SAP HANA, you map the external identities (subject DN) to the required database users. Two types of user mapping are supported:

- SAP HANA-based user mappings
  Database users are mapped explicitly to the subject DN in the X.509 certificate from a given issuer.

Sample Code

```
CREATE USER JOHN WITH IDENTITY 'CN=JOHN, OU=SAP SE, C=DE' FOR X509
PROVIDER MyProvider;
```

- Identity provider-based user mappings
  The identity provider maps its users to SAP HANA database users. The matching of external identities to database users is rule based. This enables the mapping of users from parts of the subject DN of an X.509 certificate from a given issuer to an SAP HANA user name.
  A matching rule is a distinguished name where one attribute has a value of '*'. This attribute contains the user name that needs to be matched during logon. Note that:
  - User name matching is case insensitive.
  - The '*' may be pre- or postfixed, for example, 'user_*'.
  - There may only be exactly one '*' in the rule.
  - If an attribute is matched with a '?', its content will be ignored.

  Example

  The identity provider specifies the matching rules 'CN=*, OU=SAP SE, C=DE' and 'CN=user_*, OU=SAP SE, C=DE'. When the client presents a user certificate with the certificate subject 'CN=JOHN, OU=SAP SE, C=DE' or 'CN=user_john, OU=SAP SE, C=DE', SAP HANA checks whether the user JOHN exists. If the user exists and all other attributes in the certificate match and are in same order, logon is successful. For example, with the same matching rules above, logon with a certificate subject 'CN=JOHN, C=DE, OU=SAP SE' or 'CN=john_user, OU=SAP SE, C=DE' would both fail.

  You can configure one or more matching rules when you create the identity provider in SAP HANA or later with the ALTER X509 PROVIDER command. Matching rules are tried in the same order as defined in the identity provider, and the first one that matches is used.

  Sample Code

  ```
  ALTER X509 PROVIDER MyProvider SET MATCHING RULES 'CN=*, OU=SAP SE, C=DE';
  ```

  Rule-based mapping is configured for a database user in the user definition using the keyword ANY.

  Sample Code

  ```
  CREATE USER JOHN WITH IDENTITY ANY FOR X509 PROVIDER MyProvider;
  ```

A user can be both mapped to several providers and mapped to the same provider multiple times with different fixed subjects or with keyword ANY.

You can configure X.509 identity providers and user mapping in the SAP HANA cockpit.

Trust Store for Certificate Validation

In addition to user mapping, you must configure the trust store used to validate incoming certificates against certificates signed by a trusted Certification Authority (CA). To do this, you create a certificate collection with the purpose X509 that contains the required certificates directly in the database. If necessary, you can create separate collections for different identity providers.

Client-Side Configuration

If the connection between SAP HANA and the client is TLS/SSL secured, by default, the client's TLS certificate is used for authentication if a user is enabled for X.509 authentication and no other credentials have been specified. You can specify a different X.509 certificate for authentication as part of the connection string using the `authenticationX509` connection parameter. You can also explicitly deactivate X.509 authentication for a client connection by using the `authenticationMethods` parameter in the connection string to specify the authentication mechanisms that you do want to use. For more information, see the *SAP HANA Client Interface Programming Reference*.

Related Information

Certificate Collections [page 307]
CREATE X509 PROVIDER (Access Control)
ALTER X509 PROVIDER (Access Control)
DROP X509 PROVIDER (Access Control)
X509_PROVIDERS System View
X509_PROVIDER_RULES System View
X509_USER_MAPPINGS System View
Add an X.509 Identity Provider
JDBC Connection Properties (SAP HANA Client Interface Programming Reference)
ODBC Connection Properties (SAP HANA Client Interface Programming Reference)
Maintaining Single Sign-On for SAP HANA XS Applications
Maintaining Single Sign-On for XS Advanced Applications
SAP Note 3327214
SAP Note 3399573

7.7 LDAP User Authentication

The Lightweight Directory Access Protocol (LDAP) is an application protocol for accessing directory services. If you use an LDAP-compliant directory server to manage users and their passwords, you can leverage LDAP-based authentication to access SAP HANA.

Overview

LDAP authentication can be implemented for users accessing SAP HANA directly via JDBC/ODBC database clients. The user is authenticated against an LDAP directory server using the user name and password provided by the client. The client transmits the password securely to SAP HANA using a hybrid encryption-based protocol that uses a combination of symmetric and asymmetric encryption. The SAP HANA server then decrypts the password and uses it to authenticate the user with the LDAP server. Using LDAP user passwords for authentication eliminates the need to manage user passwords and password policies in the SAP

HANA database. SAP HANA uses OpenLDAP client library version 2.4.46 for LDAP authorization and LDAP authentication.

LDAP-based authentication and LDAP automatic user provisioning are not supported for SAP HANA XSC applications; however, SAP HANA XSC supports LDAP-based authorization.

> **Note**
>
> Users configured for LDAP authentication cannot simultaneously be configured for local password authentication. Local password authentication must be disabled before a user can be configured for LDAP authentication, for example, using `ALTER USER <user_name> DISABLE PASSWORD`.

> **Caution**
>
> To protect the transmission of user passwords between SAP HANA and the LDAP server, you must secure communication between SAP HANA and the LDAP server using the TLS/SSL protocol. You can do this while configuring the connection to the LDAP server in SAP HANA. See the section on secure communication between SAP HANA and an LDAP directory server.

> **Caution**
>
> As part of the authentication process, the SAP HANA server verifies the identity of the client but the client does not verify the identity of the SAP HANA server. For server authentication, you must enable TLS client-server communication. For more information, see the section on secure communication between SAP HANA and JDBC/ODBC clients.

Direct Connections with the SQL CONNECT Statement

When the CONNECT SQL statement is used to connect a user directly from a client program, the statement is independent of the client program. Behavior is solely dependent on the capabilities provided by the SAP HANA server and not client. The existing client network connection to SAP HANA is kept and a new session context is established, after successful authentication, using the specified user and password and the authentication method set for the specified user. The CONNECT SQL statement may be executed on an existing connection to authenticate a user with LDAP authentication even if the client program does not support LDAP authentication.

For example, a technical user MYTECHUSER is created and configured for local password authentication, and user MYLDAPUSER is created and configured for LDAP authentication. If an hdbsql client establishes a connection to SAP HANA using MYTECHUSER and then in that established session, the technical user issues the SQL statement `CONNECT MYLDAPUSER PASSWORD <ldappwd>`, then the current session context becomes user MYLDAPUSER authenticated using LDAP authentication method. The original network connection remains but the current session context is for MYLDAPUSER and the original client program may not directly support LDAP authentication method.

Client Requirements

LDAP-based authentication requires the version of the SAP HANA client available with SAP HANA 2.0 SPS 03. For more information about downloading and installing the SAP HANA client, see the *SAP HANA Client Installation and Update Guide*.

To authenticate with JDBC, the client must use JDK 8 or later and the JDK's policy files must support unlimited strength encryption. Failure to do this will result in the error “Cannot decrypt data: Illegal key size”. To update the policy files:

- For JDK 8 update 150 and earlier, download the Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files for JDK/JRE 8 from Oracle.

- For JDK 8 update 151 and later as well as for JDK 9, edit the file `conf/security/java.security` and uncomment the line `crypto.policy=unlimited`.

Authorization of LDAP-Authenticated Users

The internal database user to be used for subsequent authorization checks in SAP HANA is determined during the logon process. With LDAP authentication, the internal database user name is same as the external identity used to log on. The following situations are possible:

- Database user exists and is configured for LDAP group authorization
  If the database user exists and is configured for LDAP group authorization (authorization mode `LDAP`), it is verified that the authenticated user is a member of at least one LDAP group mapped to at least one SAP HANA role. If this is the case, the user is logged on and the identified roles granted. For more information, see the section on LDAP group authorization for existing users.
- Database user exists and is configured for local authorization
  If the database user exists and is configured for local authorization (authorization mode `LOCAL`), the user is logged on. Privileges and roles must be granted directly to the database user by a user administrator.
- Database user does not exist and the LDAP provider is configured for automatic user creation
  If the database user does not exist and the LDAP provider is enabled to create database users in SAP HANA, the required database user is created. This is described in more detail in the next section.

> ### Note
>
> Currently, authorization and authentication with OpenLDAP is only supported for Active Directory.

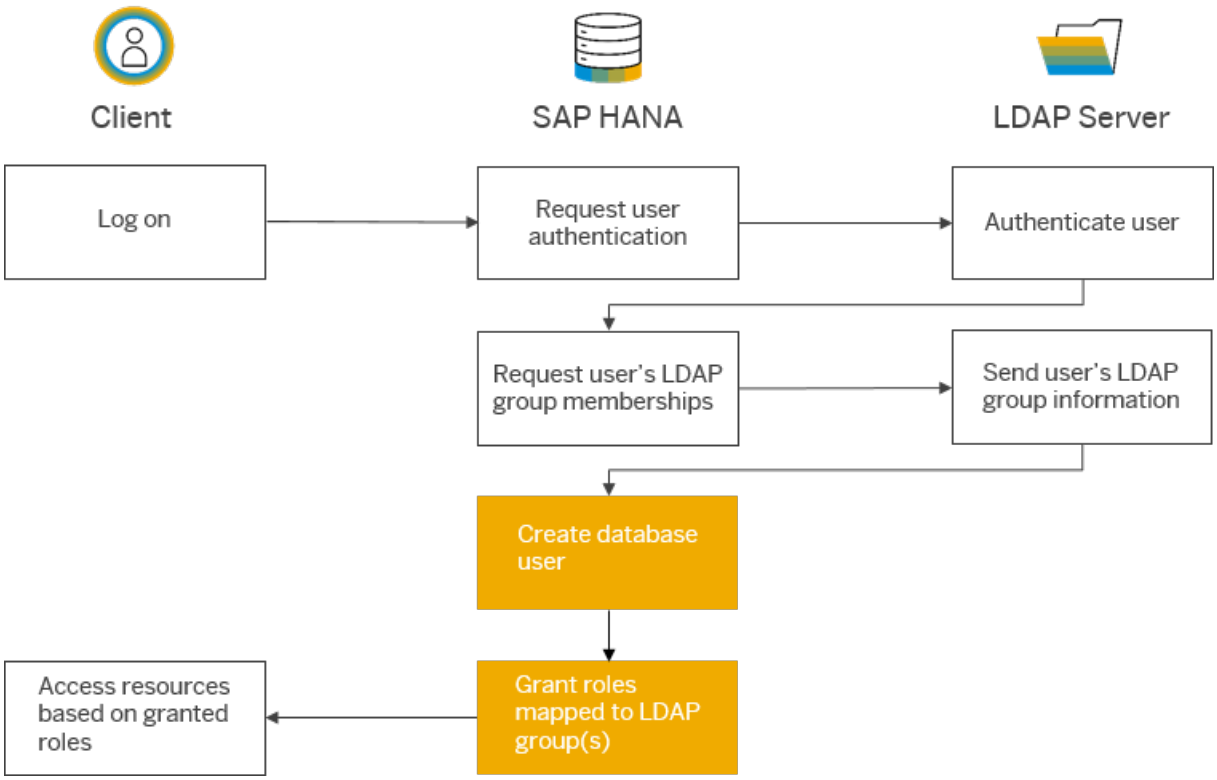
LDAP Authentication with Automatic User Creation

If the database user does not exist and the LDAP provider is enabled to create database users in SAP HANA, it is verified that the authenticated user is a member of at least one LDAP group mapped to at least one SAP HANA role. If this is the case, a database user is created. Otherwise, logon fails.

By default, a standard user is created, but it may be a restricted user if the LDAP provider is configured accordingly. The new database user is automatically configured for LDAP authentication and LDAP group authorization (authorization mode `LDAP`).

> ### Note
>
> Automatic user creation is not supported in the following cases:
>
> - In an active/active (read enabled) scenario from connections to secondary system
> - For users connecting directly using the CONNECT SQL statement

The following figure illustrates how LDAP authentication with automatic user creation works.



LDAP Authentication with Automatic User Creation

> ⓘ Note
>
> If the user on the LDAP server is deleted, the automatically created database user in SAP HANA is **not** deleted. However, this database user will no longer be able to log on. To enable the database user to log on again, either re-create the corresponding user on the LDAP server or change the authentication mechanism of the database user (for example, to user name and password).

Configuration of LDAP Authentication

To enable LDAP authentication, you must create and configure an LDAP provider in SAP HANA.

> ⓘ Note
>
> You can create several LDAP providers but only one can be in use at any time. This is the provider configured as the default.

With LDAP authentication, the external identity used to log on is same as the internal database user name. This user must be configured for LDAP authentication. This is done manually and/or automatically:

- As a user administrator, you can configure a new or existing database user for LDAP authentication in the user definition using the SAP HANA cockpit or the SQL statements CREATE USER or ALTER USER.

  > {·} Sample Code
  >
  > ```
  > CREATE USER julie WITH IDENTITY FOR LDAP PROVIDER
  > ```

> **Note**
>
> A user enabled for LDAP authentication cannot be enabled for local password authentication. Enabling LDAP authentication for a user currently configured for password authentication results in an error. The reverse is also true.

> **Note**
>
> If a user is deactivated on the LDAP server, the user will not be able to authenticate to SAP HANA. If the deactivated user still has an open connection, the session will not be terminated. If the user is deactivated in SAP HANA as well, the open connection can no longer be used.

> **Note**
>
> The user SYSTEM cannot be enabled for LDAP authentication.

- Alternatively or additionally, you can enable the LDAP provider for automatic user creation. You specify this in the provider definition with either the CREATE LDAP PROVIDER or ALTER LDAP PROVIDER statement.

> **Sample Code**
>
> ```
> ALTER LDAP PROVIDER my_ldap_provider ENABLE USER CREATION FOR LDAP
> ```

See also the section on configuring LDAP authentication and authorization in the *SAP HANA Administration Guide*.

Related Information

SAP HANA Client Installation and Update Guide
Secure Communication Between SAP HANA and an LDAP Directory Server [page 59]
Secure Communication Between SAP HANA and JDBC/ODBC Clients [page 42]
LDAP Group Authorization for Existing Users [page 207]
Configure LDAP Authentication and Authorization
CREATE LDAP PROVIDER Statement (Access Control)
ALTER LDAP PROVIDER Statement (Access Control)
CREATE USER Statement (Access Control)
ALTER USER Statement (Access Control)
USERS System View

8 SAP HANA Authorization

When a user accesses the SAP HANA database using a client interface (for example, ODBC, JDBC, or HTTP), his or her ability to perform database operations on database objects is determined by the privileges that he or she has been granted.

All the privileges granted to a user, either directly or indirectly through roles, are combined. This means that whenever a user tries to access an object, the system performs an authorization check on the user, the user's roles, and directly granted privileges. It is not possible to explicitly deny privileges. This means that the system does not need to check all the user's privileges. As soon as all requested privileges have been found, the system skips further checks and grants access.

Related Information

SAP HANA Authentication and Single Sign-On [page 100]
SAP HANA User Management [page 73]

8.1 Privileges

Several privilege types are used in SAP HANA (system, object, analytic, package, and application).

Privilege Type	Applicable To	Target User	Description
System privilege	System, database	Administrators, developers	System privileges control general system activities. They are mainly used for administrative purposes, such as creating schemas, creating and changing users and roles, monitoring and tracing. System privileges are also used to authorize basic repository operations. System privileges granted to users in a particular tenant database authorize operations in that database only. The only exception is the system privileges DATABASE ADMIN, DATABASE STOP, DATABASE START, and DATABASE AUDIT ADMIN. These system privileges can only be granted to users of the system database. They authorize the execution of operations on individual tenant databases. For example, a user with DATABASE ADMIN can create and drop tenant databases, change the database-specific properties in configuration (*.ini) files, and perform database-specific backups.

Privilege Type	Applicable To	Target User	Description
Object privilege	Database objects (schemas, tables, views, procedures and so on)	End users, technical users	Object privileges are used to allow access to and modification of database objects, such as tables and views. Depending on the object type, different actions can be authorized (for example, SELECT, CREATE ANY, ALTER, DROP). Schema privileges are object privileges that are used to allow access to and modification of schemas and the objects that they contain. Source privileges are object privileges that are used to restrict access to and modification of remote data sources, which are connected through SAP HANA smart data access. Object privileges granted to users in a particular database authorize access to and modification of database objects in that database only. That is, unless cross-database access has been enabled for the user. This is made possible through the association of the requesting user with a remote identity on the remote database. For more information, see *Cross-Database Authorization in Tenant Databases* in the *SAP HANA Security Guide*.
Analytic privilege	Analytic views	End users	Analytic privileges are used to allow read access to data in SAP HANA information models (that is, analytic views, attribute views, and calculation views) depending on certain values or combinations of values. Analytic privileges are evaluated during query processing. Analytic privileges granted to users in a particular database authorize access to information models in that database only.

Privilege Type	Applicable To	Target User	Description
Package privilege	Packages in the classic repository of the SAP HANA database	Application and content developers working in the classic SAP HANA repository	Package privileges are used to allow access to and the ability to work in packages in the classic repository of the SAP HANA database. Packages contain design time versions of various objects, such as analytic views, attribute views, calculation views, and analytic privileges. Package privileges granted to users in a particular database authorize access to and the ability to work in packages in the repository of that database only. **Note** With SAP HANA XS advanced, source code and web content are not versioned and stored in the SAP HANA database, so package privileges are not used in this context. For more information, see *Authorization in SAP HANA XS Advanced*.
Package privilege	Packages in the classic repository of the SAP HANA database	Application and content developers working in the classic SAP HANA repository	Package privileges are not relevant in the SAP HANA service for SAP BTP context as the SAP HANA repository is not supported.
Application privilege	SAP HANA XS classic applications	Application end users, technical users (for SQL connection configurations)	Developers of SAP HANA XS classic applications can create application privileges to authorize user and client access to their application. They apply in addition to other privileges, for example, object privileges on tables. Application privileges can be granted directly to users or roles in runtime in the SAP HANA studio. However, it is recommended that you grant application privileges to roles created in the repository in design time. **Note** With SAP HANA XS advanced, application privileges are not used. Application-level authorization is implemented using OAuth and authorization scopes and attributes. For more information, see *Authorization in SAP HANA XS Advanced*.
Application privilege	SAP HANA XS classic applications	Application end users, technical users (for SQL connection configurations)	Application privileges are not relevant in the SAP HANA service for SAP BTP context as SAP HANA XS classic is not supported.

> **Note**
>
> There are no HDI or XS advanced equivalents in the SAP HANA authorization concept for package privileges on repository packages and applications privileges on SAP HANA XS classic applications. For more information about the authorization concept of XS advanced, see the *SAP HANA Security Guide*.

Privileges on Users

An additional privilege type, privileges on users, can be granted to users. Privileges on users are SQL privileges that users can grant on their user. ATTACH DEBUGGER is the only privilege that can be granted on a user.

For example, User A can grant User B the privilege ATTACH DEBUGGER to allow User B debug SQLScript code in User A's session. User A is only user who can grant this privilege. Note that User B also needs the object privilege DEBUG on the relevant SQLScript procedure.

For more information, see the section on debugging procedures in the *SAP HANA Developer Guide*.

Related Information

GRANT
Cross-Database Authorization in Tenant Databases [page 204]
Authorization in SAP HANA XS Advanced [page 357]
Debug an External Session
Recommendations for Database Users, Roles, and Privileges

8.1.1 System Privileges

System privileges control general system activities.

System privileges are mainly used to authorize users to perform administrative actions, including:

- Creating and deleting schemas
- Managing users and roles
- Performing data backups
- Monitoring and tracing
- Managing licenses

System privileges are also used to authorize basic repository operations, for example:

- Importing and exporting content
- Maintaining delivery units (DU)

System privileges granted to users in a particular database authorize operations in that database only. The only exception is the system privileges DATABASE ADMIN, DATABASE STOP, and DATABASE START, DATABASE AUDIT ADMIN. These system privileges can only be granted to users of the system database. They authorize the execution of operations on individual tenant databases. For example, a user with DATABASE ADMIN can create and drop tenant databases, change the database-specific properties in configuration (*.ini) files, and perform database-specific or full-system data backups.

> **Note**
>
> System privileges should always be assigned with caution.

Related Information

System Privileges (Reference) [page 140]
SAP Note 2950209

8.1.1.1 System Privileges (Reference)

System privileges control general system activities.

General System Privileges

System privileges restrict administrative tasks. The following table describes the supported system privileges in an SAP HANA database.

System Privilege	Description
ADAPTER ADMIN	Controls the execution of the following adapter-related statements: CREATE ADAPTER, DROP ADAPTER, and ALTER ADAPTER. It also allows access to the ADAPTERS and ADAPTER_LOCATIONS system views.
AGENT ADMIN	Controls the execution of the following agent-related statements: CREATE AGENT, DROP AGENT, and ALTER AGENT. It also allows access to the AGENTS and ADAPTER_LOCATIONS system views.
ALTER CLIENTSIDE ENCRYPTION KEYPAIR	Authorizes a user to add a new version of a client-side encryption key pair (CKP), or to drop all older versions of the CKP.
AUDIT ADMIN	Controls the execution of the following auditing-related statements: CREATE AUDIT POLICY, DROP AUDIT POLICY, and ALTER AUDIT POLICY, as well as changes to the auditing configuration. It also allows access to the AUDIT_LOG, XSA_AUDIT_LOG, and ALL_AUDIT_LOG system views.

System Privilege	Description
AUDIT OPERATOR	Authorizes the execution of the following statement: ALTER SYSTEM CLEAR AUDIT LOG. It also allows access to the AUDIT_LOG system view.
AUDIT READ	Authorizes read-only access to the rows of the AUDIT_LOG, XSA_AUDIT_LOG, and ALL_AUDIT_LOG system views.
BACKUP ADMIN	Authorizes BACKUP and RECOVERY statements for defining and initiating backup and recovery procedures. It also authorizes changing system configuration options with respect to backup and recovery.
BACKUP OPERATOR	Authorizes the BACKUP statement to initiate a backup.
CATALOG READ	Authorizes unfiltered access to the data in the system views that a user has already been granted the SELECT privilege on. Normally, the content of these views is filtered based on the privileges of the user. CATALOG READ does not allow a user to view system views on which they have not been granted the SELECT privilege.
CERTIFICATE ADMIN	Authorizes the changing of certificates and certificate collections that are stored in the database.
CLIENT PARAMETER ADMIN	Authorizes a user to override the value of the CLIENT parameter for a database connection or to overwrite the value of the $$client$$ parameter in an SQL query.
CREATE CLIENTSIDE ENCRYPTION KEYPAIR	Authorizes a user to create client-side encryption key pairs.
CREATE REMOTE SOURCE	Authorizes the creation of remote data sources by using the CREATE REMOTE SOURCE statement.
CREATE SCENARIO	Controls the creation of calculation scenarios and cubes (calculation database).
CREATE SCHEMA	Authorizes the creation of database schemas using the CREATE SCHEMA statement.
CREATE STRUCTURED PRIVILEGE	Authorizes the creation of structured (analytic privileges). Only the owner of the privilege can further grant or revoke that privilege to other users or roles.
CREDENTIAL ADMIN	Authorizes the use of the statements CREATE CREDENTIAL, ALTER CREDENTIAL, and DROP CREDENTIAL.

System Privilege	Description
DATA ADMIN	Authorizes reading all data in the system views. It also enables execution of Data Definition Language (DDL) statements in the SAP HANA database. A user with this privilege cannot select or change data in stored tables for which they do not have access privileges, but they can drop tables or modify table definitions.
DATABASE ADMIN*	Authorizes all statements related to tenant databases, such as CREATE, DROP, ALTER, RENAME, BACKUP, and RECOVERY.
DATABASE AUDIT ADMIN*	Authorizes a user to create and manage audit policies for a tenant database.
DATABASE BACKUP ADMIN*	Authorizes all statements related to view, create, schedule, and cancel database backups except for recovering or copying a database.
DATABASE BACKUP OPERATOR*	Authorizes a user to view, create, schedule, and cancel database backups.
DATABSE RECOVERY OPERATOR*	Authorizes a user to recover or copy a database.
DATABASE START*	Authorizes a user to start any database in the system and to select from the M_DATABASES view.
DATABASE STOP*	Authorizes a user to stop any database in the system and to select from the M_DATABASES view.
DROP CLIENTSIDE ENCRYPTION KEYPAIR	Authorizes a user to drop other users' client-side encryption key pairs.
ENCRYPTION ROOT KEY ADMIN	Authorizes all statements related to management of root keys: Allows access to the system views pertaining to encryption (for example, ENCRYPTION_ROOT_KEYS, M_ENCRYPTION_OVERVIEW, M_PERSISTENCE_ENCRYPTION_STATUS, M_PERSISTENCE_ENCRYPTION_KEYS, and so on).
EXPORT	Authorizes EXPORT to a file on the SAP HANA server. The user must also have the SELECT privilege on the source tables to be exported.
EXTENDED STORAGE ADMIN	Authorizes the management of SAP HANA dynamic tiering and the creation of extended storage.

System Privilege	Description
IMPORT	Authorizes the import activity in the database using the IMPORT statements. Additional privileges may also be required to be able to execute an IMPORT. See the IMPORT statement for more information.
INIFILE ADMIN	Authorizes making changes to system settings.
LDAP ADMIN	Authorizes the use of the CREATE | ALTER | DROP | VALIDATE LDAP PROVIDER statements.
LICENSE ADMIN	Authorizes the use of the SET SYSTEM LICENSE statement to install a new license.
LOG ADMIN	Authorizes the use of the ALTER SYSTEM LOGGING [ON | OFF] statements to enable or disable the log flush mechanism.
MONITOR ADMIN	Authorizes the use of the ALTER SYSTEM statements for events.
OPTIMIZER ADMIN	Authorizes the use of the ALTER SYSTEM statements concerning SQL PLAN CACHE and ALTER SYSTEM UPDATE STATISTICS statements, which influence the behavior of the query optimizer.
PARTITION ADMIN	Authorizes the use of all non-destructive partitioning operations when altering a table.
RESOURCE ADMIN	Authorizes statements concerning system resources (for example, the ALTER SYSTEM RECLAIM DATAVOLUME and ALTER SYSTEM RESET MONITORING VIEW statements). It also authorizes use of the Kernel Profiler statements, and many of the statements available in the Management Console.
ROLE ADMIN	Authorizes the creation and deletion of roles by using the CREATE ROLE and DROP ROLE statements. It also authorizes the granting and revoking of roles by using the GRANT and REVOKE statements. Activated repository roles, meaning roles whose creator is the predefined user _SYS_REPO, can neither be granted to other roles or users nor dropped directly. Not even users with the ROLE ADMIN privilege can do so. Check the documentation concerning activated objects.
SAVEPOINT ADMIN	Authorizes the execution of a savepoint using the ALTER SYSTEM SAVEPOINT statement.

System Privilege	Description
SCENARIO ADMIN	Authorizes all calculation scenario-related activities (including creation).
SERVICE ADMIN	Authorizes the ALTER SYSTEM [START|CANCEL|RECONFIGURE] statements for administering system services of the database.
SESSION ADMIN	Authorizes the ALTER SYSTEM commands concerning sessions to stop or disconnect a user session or to change session variables.
SSL ADMIN	Authorizes the use of the SET...PURPOSE SSL statement. It also allows access to the PSES system view.
STRUCTUREDPRIVILEGE ADMIN	Authorizes the creation, reactivation, and dropping of structured (analytic) privileges.
SYSTEM REPLICATION ADMIN	Authorizes the use of ALTER SYSTEM statements related to system replication.
TABLE ADMIN	Authorizes the LOAD, UNLOAD and MERGE DELTA statements for tables and table partitions, as well as the ALTER TABLE statement for those clauses that do change the structure of the table and do not allow access to table data either explicitly or implicitly, for example: LOB REORGANIZE, CLEAR COLUMN JOIN DATA STATISTICS, and PRELOAD.
TRACE ADMIN	Authorizes the use of the ALTER SYSTEM statements related to database tracing (including the Kernel Profiler feature) and the changing of trace system settings.
TRUST ADMIN	Authorizes the use of statements to update the trust store.
USER ADMIN	Authorizes the creation and modification of users by using the CREATE | ALTER | DROP USER statements.
VERSION ADMIN	Authorizes the use of the ALTER SYSTEM RECLAIM VERSION SPACE statement of the multi-version concurrency control (MVCC) feature.
WORKLOAD ADMIN	Authorizes execution of the workload class and mapping statements (for example, CREATE | ALTER | DROP WORKLOAD CLASS, and CREATE | ALTER | DROP WORKLOAD MAPPING).
WORKLOAD ANALYZE ADMIN	Used by the Analyze Workload, Capture Workload, and Replay Workload applications when performing workload analysis.

System Privilege	Description
WORKLOAD CAPTURE ADMIN	Authorizes access to the monitoring view M_WORKLOAD_CAPTURES to see the current status of capturing and captured workloads, as well of execution of actions with the WORKLOAD_CAPTURE procedure.
WORKLOAD REPLAY ADMIN	Authorizes access to the monitoring views M_WORKLOAD_REPLAY_PREPROCESSES and M_WORKLOAD_REPLAYS to see current status of preprocessing, preprocessed, replaying, and replayed workloads, as well as the execution of actions with the WORKLOAD_REPLAY procedure.
`<identifier>.<identifier>`	Components of the SAP HANA database can create new system privileges. These privileges use the component-name as the first identifier of the system privilege and the component-privilege-name as the second identifier.

Repository System Privileges

> **Note**
>
> The following privileges authorize actions on individual packages in the SAP HANA repository, used in the SAP HANA Extended Services (SAP HANA XS) classic development model. With SAP HANA XS advanced, source code and web content are no longer versioned and stored in the repository of the SAP HANA database.

System Privilege	Description
REPO.EXPORT	Authorizes the export of delivery units for example
REPO.IMPORT	Authorizes the import of transport archives
REPO.MAINTAIN_DELIVERY_UNITS	Authorizes the maintenance of delivery units (DU, DU vendor and system vendor must be the same
REPO.WORK_IN_FOREIGN_WORKSPACE	Authorizes work in a foreign inactive workspace
REPO.CONFIGURE	Authorize work with SAP HANA Change Recording, which is part of SAP HANA Application Lifecycle Management
REPO.MODIFY_CHANGE	
REPO.MODIFY_OWN_CONTRIBUTION	
REPO.MODIFY_FOREIGN_CONTRIBUTION	

Related Information

GRANT
Developer Authorization in the Repository [page 200]

8.1.2 Object Privileges

Object privileges are SQL privileges that are used to allow access to and modification of database objects.

For each SQL statement type (for example, SELECT, UPDATE, or CALL), a corresponding object privilege exists. If a user wants to execute a particular statement on a simple database object (for example, a table), he or she must have the corresponding object privilege for either the actual object itself, or the schema in which the object is located. This is because the schema is an object type that contains other objects. A user who has object privileges for a schema automatically has the same privileges for all objects currently in the schema and any objects created there in the future.

Object privileges are not only grantable for database catalog objects such as tables, views and procedures. Object privileges can also be granted for non-catalog objects such as development objects in the repository of the SAP HANA database.

Initially, the owner of an object and the owner of the schema in which the object is located are the only users who can access the object and grant object privileges on it to other users.

An object can therefore be accessed only by the following users:

- The owner of the object
- The owner of the schema in which the object is located
- Users to whom the owner of the object has granted privileges
- Users to whom the owner of the parent schema has granted privileges

> ⚠ Caution
>
> The database owner concept stipulates that when a database user is deleted, all objects created by that user and privileges granted to others by that user are also deleted. If the owner of a schema is deleted, all objects in the schema are also deleted even if they are owned by a different user. All privileges on these objects are also deleted.

> ⓘ Note
>
> The owner of a table can change its ownership with the `ALTER TABLE` SQL statement. In this case, the new owner becomes the grantor of all privileges on the table granted by the original owner. The original owner is also automatically granted all privileges for the table with the new owner as grantor. This ensures that the original owner can continue to work with the table as before.

Authorization Check on Objects with Dependencies

The authorization check for objects defined on other objects (that is, stored procedures and views) is more complex. In order to be able to access an object with dependencies, both of the following conditions must be met:

- The user trying to access the object must have the relevant object privilege on the object as described above.
- The user who created the object must have the required privilege on all underlying objects **and** be authorized to grant this privilege to others.

If this second condition is not met, only the owner of the object can access it. He cannot grant privileges on it to any other user. This cannot be circumvented by granting privileges on the parent schema instead. Even if a user has privileges on the schema, he will still not be able to access the object.

> ⓘ Note
>
> This applies to procedures created in DEFINER mode only. This means that the authorization check is run against the privileges of the user who created the object, not the user accessing the object. For procedures created in INVOKER mode, the authorization check is run against the privileges of the accessing user. In this case, the user must have privileges not only on the object itself but on all objects that it uses.

> → Tip
>
> The SAP HANA studio provides a graphical feature, the authorization dependency viewer, to help troubleshoot authorization errors for object types that typically have complex dependency structures: stored procedures and calculation views.

Related Information

GRANT Statement (Access Control)
ALTER TABLE Statement (Data Definition)
Resolve Errors Using the Authorization Dependency Viewer
Object Privileges (Reference) [page 147]
Cross-Database Authorization in Tenant Databases [page 204]

8.1.2.1 Object Privileges (Reference)

Object privileges are used to allow access to and modification of database objects, such as tables and views.

The following table describes the supported object privileges in an SAP HANA database.

Object Privilege	Command Types	Applies to	Privilege Description
ALL PRIVILEGES	DDL & DML	• Schemas • Tables • Views	This privilege is a collection of all Data Definition Language (DDL) and Data Manipulation Language (DML) privileges that the grantor currently possesses and is allowed to grant further. The privilege it grants is specific to the particular object being acted upon. This privilege collection is dynamically evaluated for the given grantor and object.
ALTER	DDL	• Schemas • Tables • Views • Functions/procedures	Authorizes the ALTER statement for the object.
AGENT MESSAGING	DDL	• Schemas • Tables • Views • Functions/procedures	Authorizes the user with which the agent communicates with the data provisioning server using HTTP protocol.
ATTACH DEBUGGER	DDL	• User	Authorizes debugging across different user sessions. For example, userA can grant ATTACH DEBUGGER to userB to allow userB to debug a procedure in userA's session (userB still needs DEBUG privilege on the procedure, however).
CREATE ANY	DDL	• Schemas • Tables • Views • Sequences • Functions/procedures • Remote sources • Graph workspaces • Triggers	Authorizes all CREATE statements for the object.

Object Privilege	Command Types	Applies to	Privilege Description
CREATE OBJECT STRUCTURED PRIVILEGE	DDL	• Schemas • Views	Authorizes creation of structured privileges on the object even if the user does not have the CREATE STRUCTURED PRIVILEGE.
CREATE REMOTE SUBSCRIPTION	DDL	• Remote Source	Authorizes the creation of remote subscriptions executed on this source entry.
CREATE VIRTUAL FUNCTION	DDL	• Remote sources	Authorizes creation of virtual functions (the REFERENCES privilege is also required).
CREATE VIRTUAL PROCEDURE	DDL	• Remote sources	Authorizes creation of virtual procedure to create and run procedures on a remote source.
CREATE VIRTUAL PACKAGE	DDL	• Schemas	Authorizes creation of virtual packages that can be run on remote sources.
CREATE VIRTUAL TABLE	DDL	• Remote sources	Authorizes the creation of proxy tables pointing to remote tables from the source entry.
CREATE TEMPORARY TABLE	DDL	• Schemas	Authorizes the creation of a temporary local table, which can be used as input for procedures, even if the user does not have the CREATE ANY privilege for the schema.
DEBUG	DML	• Schemas • Calculation Views • Functions/procedures	Authorizes debug functionality for the procedure or calculation view or for the procedures and calculation views of a schema.
DEBUG MODIFY	DDL	• Functions/procedures	For internal use only.

Object Privilege	Command Types	Applies to	Privilege Description
DELETE	DML	• Schemas • Tables • Views • Functions/procedures	Authorizes the DELETE and TRUNCATE statements for the object. While DELETE applies to views, it only applies to updatable views (that is, views that do not use a join, do not contain a UNION, and do not use aggregation).
DROP	DDL	• Schemas • Tables • Views • Sequences • Functions/procedures • Remote sources • Graph workspaces	Authorizes the DROP statements for the object.
EXECUTE	DML	• Schemas • Functions/procedures	Authorizes the execution of a SQLScript function or a database procedure by using the CALLS or CALL statement respectively. It also allows a user to execute a virtual function.
INDEX	DDL	• Schemas • Tables	Authorizes the creation, modification, or dropping of indexes for the object.
INSERT	DML	• Schemas • Tables • Views	Authorizes the INSERT statement for the object. The INSERT and UPDATE privilege are both required on the object to allow the REPLACE and UPSERT statements to be used. While INSERT applies to views, it only applies to updatable views (views that do not use a join, do not contain a UNION, and do not use aggregation).

Object Privilege	Command Types	Applies to	Privilege Description
PROCESS REMOTE SUBSCRIPTION EXCEPTION	DDL	• Remote Source • Remote Subscription	Authorizes processing exceptions on this source entry.
LINKED DATABASE	DDL	• Remote source	Authorizes execution of linked database queries.
REFERENCES	DDL	• Schemas • Tables	Authorizes the usage of all tables in this schema or this table in a foreign key definition, or the usage of a personal security environment (PSE). It also allows a user to reference a virtual function package.
REMOTE TABLE ADMIN	DDL	• Remote sources	Authorizes the creation of tables on a remote source object.
SELECT	DML	• Schemas • Tables • Views • Sequences • Graph workspaces	Authorizes the SELECT statement for the object or the usage of a sequence. When selection from system-versioned tables, users must have SELECT on both the table and its associated history table.
SELECT CDS METADATA	DML	• Schemas • Tables	Authorizes access to CDS metadata from the catalog.
SELECT METADATA	DML	• Schemas • Tables	Authorizes access to the complete metadata of all objects in a schema (including procedure and view definitions), including objects that may be located in other schemas.
SQLSCRIPT LOGGING	DML	• Function • Library • Procedure • Schema	Authorizes the collection of logs for a SQLScript object.

Object Privilege	Command Types	Applies to	Privilege Description
TRIGGER	DDL	• Schemas • Tables	Authorizes the CREATE/ALTER/DROP/ENABLE and DISABLE TRIGGER statements for the specified table or the tables in the specified schema.
UNMASKED	DML	• Schemas • Views • Tables	Authorizes access to masked data in user-defined views and tables. This privilege is required to view the original data in views and tables that are defined by using the WITH MASK clause.
UPDATE	DML	• Schemas • Tables • Views	While UPDATE applies to views, it only applies to updatable views (views that do not use a join, do not contain a UNION, and do not use aggregation).
USAGE	DDL	• Client Side Encryption Coulmn Key	Authorizes client side encryption keys.

Object Privilege	Command Types	Applies to	Privilege Description
USERGROUP OPERATOR	DML	• User groups	Authorizes a user to change the settings for a user group, and to add and remove users to/from a user group. Users with the USERGROUP OPERATOR privilege can also create and drop users, but only within the user group they have the USERGROUP OPERATOR privilege on (CREATE USER `<user_name>` SET USERGROUP `<usergroup_name>`). A user can have the USERGROUP OPERATOR privilege on more than one user group, and a user group can have more than one user with the USERGROUP OPERATOR privilege on it. When granting USERGROUP OPERATOR to a user group, you must include the keyword USERGROUP before the name of the user group (for example: `GRANT USERGROUP OPERATOR ON USERGROUP <usergroup> TO <grantee>`). This is slightly differenct syntax than granting USERGROUP OPERATOR to a user.
`<identifier>.<identifier>`	DDL		Components of the SAP HANA database can create new object privileges. These privileges use the component-name as first identifier of the system privilege and the component-privilege-name as the second identifier.

Related Information

GRANT

8.1.3 Analytic Privileges

Analytic privileges grant different users access to different portions of data in the same view based on their business role. Within the definition of an analytic privilege, the conditions that control which data users see is either contained in an XML document or defined using SQL.

Row-Level Access Control

Standard object privileges (`SELECT`, `ALTER`, `DROP`, and so on) implement coarse-grained authorization at object level only. Users either have access to an object, such as a table, view or procedure, or they don't. While this is often sufficient, there are cases when access to data in an object depends on certain values or combinations of values. Analytic privileges are used in the SAP HANA database to provide such fine-grained control at row level of which data individual users can see within the same view.

> **Example**
>
> Sales data for all regions are contained within one analytic view. However, regional sales managers should only see the data for their region. In this case, an analytic privilege could be modeled so that they can all query the view, but only the data that each user is authorized to see is returned.

Creating Analytic Privileges

Although analytic privileges can be created directly as catalog objects in runtime, we recommend creating them as design-time objects that become catalog objects on deployment (database artifact with file suffix `.hdbanalyticprivilege`).

In an SAP HANA XS classic environment, analytic privileges are created in the built-in repository of the SAP HANA database using either the SAP HANA Web Workbench or the SAP HANA studio. In an SAP HANA XS advanced environment, they are created using the SAP Web IDE and deployed using the SAP HANA deployment infrastructure (SAP HANA DI).

> **Note**
>
> HDI supports only analytic privileges deployed using SQL (see below). Furthermore, due to the container-based model of HDI, where each container corresponds to a database schema, analytic privileges created in HDI are schema specific.

XML- Versus SQL-Based Analytic Privileges

Before you implement row-level authorization using analytic privileges, you need to decide which type of analytic privilege is suitable for your scenario. In general, SQL-based analytic privileges allow you to more easily formulate complex filter conditions using sub-queries that might be cumbersome to model using XML-based analytic privileges.

> → Recommendation
>
> SAP recommends the use of SQL-based analytic privileges. Using the *SAP HANA Modeler* perspective of the SAP HANA studio, you can migrate XML-based analytic privileges to SQL-based analytic privileges. For more information, see the *SAP HANA Modeling Guide for SAP HANA Studio*.

> ⓘ Note
>
> As objects created in the repository, XML-based analytic privileges are deprecated as of SAP HANA 2.0 SPS 02. For more information, see SAP Note 2465027.

The following are the main differences between XML-based and SQL-based analytic privileges:

Feature	SQL-Based Analytic Privileges	XML-Based Analytic Privileges
Control of read-only access to SAP HANA information models: • Attribute views • Analytic views • Calculation views	Yes	Yes
Control of read-only access to SQL views	Yes	No
Control of read-only access to database tables	No	No
Design-time modeling using the SAP HANA Web-based Workbench or the *SAP HANA Modeler* perspective of the SAP HANA studio ⓘ Note: This corresponds to development in an SAP HANA XS classic environment using the SAP HANA repository.	Yes	Yes
Design-time modeling using the SAP Web IDE for SAP HANA ⓘ Note: This corresponds to development in an SAP HANA XS advanced environment using HDI.	Yes	No
Transportable	Yes	Yes
HDI support	Yes	No
Complex filtering	Yes	No

Enabling an Authorization Check Based on Analytic Privileges

All column views modeled and activated in the SAP HANA modeler and the SAP HANA Web-based Development Workbench automatically enforce an authorization check based on analytic privileges. XML-based analytic privileges are selected by default, but you can switch to SQL-based analytic privileges.

Column views created using SQL must be explicitly registered for such a check by passing the relevant parameter:

- `REGISTERVIEWFORAPCHECK` for a check based on XML-based analytic privileges
- `STRUCTURED PRIVILEGE CHECK` for a check based on SQL-based analytic privileges

SQL views must always be explicitly registered for an authorization check based on analytic privileges by passing the `STRUCTURED PRIVILEGE CHECK` parameter.

> **Note**
>
> It is not possible to enforce an authorization check on the same view using both XML-based and SQL-based analytic privileges. However, it is possible to build views with different authorization checks on each other.

Related Information

Create Static SQL Analytic Privileges (SAP Web IDE for SAP HANA)
Create Dynamic SQL Analytic Privileges (SAP Web IDE for SAP HANA)
Create Analytic Privileges Using SQL Expressions (SAP Web IDE for SAP HANA)
Create Classical XML-Based Analytic Privileges (SAP HANA Web Workbench)
Create Static SQL Analytic Privileges (SAP HANA Web Workbench)
Create Classical XML-based Analytic Privileges (SAP HANA Studio)
Create SQL Analytic Privileges (SAP HANA Studio)
Convert Classical XML-based Analytic Privileges to SQL-based Analytic Privileges (SAP HANA Studio)
SAP Note 2465027

8.1.3.1 Structure of SQL-Based Analytic Privileges

An analytic privilege consists of a set of restrictions against which user access to a particular attribute view, analytic view, calculation view, or SQL view is verified. In an SQL-based analytic privilege, these restrictions are specified as filter conditions that are fully SQL based.

SQL-based analytic privileges are created using the `CREATE STRUCTURED PRIVILEGE` statement:

```
CREATE STRUCTURED PRIVILEGE <privilege_name> FOR <action> ON <view_name>
<filter_condition>;
```

The `FOR` clause is used restrict the type of access (only the `SELECT` action is supported). The `ON` clause is used to restrict access to one or more views with the same filter attributes.

The `<filter condition>` parameter is used to restrict the data visible to individual users. The following methods of specifying filter conditions are possible:

- Fixed filter (`WHERE`) clause
- Dynamically generated filter (`CONDITION PROVIDER`) clause

Fixed Filter Clauses

A **fixed filter clause** consists of an `WHERE` clause that is specified in the definition of the analytic privilege itself.

You can express fixed filter conditions freely using SQL, including subqueries.

By incorporating SQL functions into the subqueries, in particular `SESSION_USER`, you can define an even more flexible filter condition.

> ### Example
>
> ```
> country IN (SELECT a.country FROM authorizationtable a WHERE SESSION_USER=
> a.user_name)
> ```

> ### Note
>
> A **calculation view** cannot be secured using an SQL-based analytic privilege that contains a complex filter condition if the view is defined on top of analytic and/or attributes views that themselves are secured with an SQL-based analytic privilege with a complex filter condition.

> ### → Remember
>
> If you use a subquery, you (the creating user) must have the required privileges on the database objects (tables and views) involved in the subquery.

Comparative conditions can be nested and combined using AND and OR (with corresponding brackets).

> ### → Tip
>
> To create an analytic privilege that allows either access to all data or no data in a view, set a fixed filter condition such as 1=1 or 1!=1.

Dynamically Generated Filter Clauses

With a dynamically generated filter clause, the `WHERE` clause that specifies the filter condition is generated every time the analytic privilege is evaluated. This is useful in an environment in which the filter clause changes very dynamically. The filter condition is determined by a procedure specified in the `CONDITION PROVIDER` clause, for example:

> ### Sample Code
>
> ```
> CREATE STRUCTURED PRIVILEGE dynamic_ap FOR SELECT ON schema1.v1 CONDITION
> PROVIDER schema2.procedure1;
> ```

Procedures in the `CONDITION PROVIDER` clause must have the following properties:

- They must have the security mode `DEFINER`.
- They must be read-only procedures.
- They must have a predefined signature. Here, the following conditions apply:
  - No input parameter
  - Only one output parameter for the filter condition string of string type NVARCHAR, VARCHAR, CLOB, or NCLOB
    While VARCHAR and NVARCHAR have length limitations of 5000 characters, CLOB and NCLOB can be used to accommodate longer filter strings.
- The procedure may only return conditions expressed with the following operators:
  - =, <=, <, >, >=
  - LIKE
  - BETWEEN
  - IN
  - NOT (...)
  - !=

  A complex filter condition, that is a subquery, may not be returned.

  > → Tip
  >
  > A procedure that returns the filter condition 1=1 or 1>1 can be used to create an analytic privilege that allows access to all data or no data in a view.
- The procedure must be executable by either the user `_SYS_REPO` (default) or the owner of the analytic privilege. This means that:
  - `_SYS_REPO` or the owner of the analytic privilege must be the owner of the procedure, or
  - The owner of the procedure has all privileges on the underlying tables/views with `WITH GRANT OPTION` and has granted the `EXECUTE` privilege on the procedure to `_SYS_REPO` or the owner of the analytic privilege

  The parameter `[authorization] execute_dynamic_analytic_privilege_by_sys_repo_user` in the indexerver.ini configuration file determines which user must be able to execute the procedure.

  > ⚠ Caution
  >
  > Changing the value of this parameter while there are views already secured using analytic privileges with a dynamically generated filter may result in the procedure no longer being executable.
- The procedure must return a valid filter string. In particular, the filter string must not be empty and must represent a valid WHERE condition for the view.

If errors occur in procedure execution or an invalid filter string (empty or not applicable) is returned, the user receives a `Not authorized` error, even if he has the analytic privileges that would grant access.

Related Information

CREATE STRUCTURED PRIVILEGE Statement (Access Control)

8.1.3.1.1 Examples: Securing Views Using SQL-Based Analytic Privileges

Use the CREATE STRUCTURED PRIVILEGE statement to create SQL-based analytic privileges for different scenarios.

Context

The examples provided here take you through the following scenarios:

- Example 1: Securing a column view using an SQL-based analytic privilege with a fixed filter clause [page 159]
- Example 2: Securing an SQL view using an SQL-based analytic privilege with a complex filter clause (subquery) [page 161]
- Example 3: Securing a column view using an SQL-based analytic privilege with a dynamically generated filter clause [page 163]

> **Note**
>
> The analytic privileges in these examples are created using the CREATE STRUCTURED PRIVILEGE statement. Under normal circumstances, you create SQL-based analytic privileges using the SAP HANA Web IDE. They can be granted and revoked only by the actual database user who creates them.

Example 1: Secure a Column View Using an SQL-Based Analytic Privilege with a Fixed Filter Clause

Prerequisites

The database user TABLEOWNER has set up a calculation scenario based on the table SALES_TABLE, which contains the data to be protected.

Context

All sales data is contained in a single view. You want to restrict user access so that sales managers can see only information about the product "car" in the sales region UK and Germany. You want to do this by creating an analytic privilege with a fixed filter clause.

A fixed filter clause consists of an SQL WHERE clause that is specified in the definition of the analytic privilege itself.

→ Tip

In the following procedure, you might find it easier to use the graphical editors to create the calculation view and analytic privilege.

Procedure

1. Create the view containing the sales data:

```
CREATE COLUMN VIEW "TABLEOWNER"."VIEW_SALES" TYPE CALCULATION WITH PARAMETERS
('PARENTCALCINDEXSCHEMA'='TABLEOWNER',
     'PARENTCALCINDEX'='CALCSCEN_SALES',
    'PARENTCALCNODE'='SALES_TABLE',
     'REGISTERVIEWFORAPCHECK'='0') STRUCTURED PRIVILEGE CHECK
;
```

ⓘ Note

You can see above that the authorization check using XML-based analytic privileges is disabled with `'REGISTERVIEWFORAPCHECK'='0'`, while the authorization check using SQL-based analytic privileges is enabled with `STRUCTURED PRIVILEGE CHECK`. Both checks cannot be enabled at the same time.

2. Create the analytic privilege:

```
CREATE STRUCTURED PRIVILEGE AP_SALES_1 FOR SELECT
     ON TABLEOWNER.VIEW_SALES
WHERE REGION IN ('DE','UK')
AND PRODUCT = 'CAR'
;
```

→ Remember

When specifying filters, remember the following:

- You can specify only the SELECT action in the FOR clause.
- You can specify one or more views with the same filter attributes in the ON clause
- You can specify comparative conditions between attributes and constant values using only the following operators:
  - =, <=, <, >, >=
  - LIKE
  - BETWEEN
  - IN
- You can create complex filter conditions by including SQL statements as subqueries inside the WHERE clause. Example 2 illustrates how you do this. But remember: A **calculation view** cannot be secured using an SQL-based analytic privilege that contains a complex filter condition if the view is defined on top of analytic and/or attributes views that themselves are secured with an SQL-based analytic privilege with a complex filter condition.
  Also remember that if you use a subquery, you must have the required privileges on the database objects (tables and views) involved in the subquery.

3. Grant the SELECT privilege on the view TABLEOWNER.VIEW_SALES to the relevant users/roles:

```
GRANT SELECT on TABLEOWNER.VIEW_SALES to <SALES_MANAGERS>;
```

> → Remember
>
> Only the view owner or a user who has the SELECT privilege WITH GRANT OPTION on the view can perform the grant.

4. Grant the analytic privilege to the relevant users/roles:

```
GRANT STRUCTURED PRIVILEGE AP_SALES_1 TO <SALES_MANAGERS>;
```

> → Remember
>
> Owner of the analytic privilege or any user who was granted the analytic privilege with GRANT OPTION can grant it.

Example 2: Secure an SQL View Using an SQL-Based Analytic Privilege with a Complex Filter Clause (Subquery)

Prerequisites

The database user TABLEOWNER has created a table TABLEOWNER.SALES, which contains the data to be protected.

Context

All sales data is contained in a single view. You want to restrict access of user MILLER so that he can see only product information from the year 2008. You want to do this by creating an analytic privilege with a complex filter clause.

With a complex filter clause, the SQL WHERE clause that specifies the filter condition includes an SQL statement, or a subquery. This allows you to create complex filter conditions to control which data individual users see.

> → Tip
>
> In the following procedure, you might find it easier to use the graphical editors to create the calculation view and analytic privilege.

Procedure

1. Create the view containing the sales data which needs to be secured:

```
CREATE VIEW "VIEWOWNER"."ROW_VIEW_SALES_ON_SALES" AS SELECT
     * FROM "TABLEOWNER"."SALES" WITH STRUCTURED PRIVILEGE CHECK
;
```

> → Remember
>
> The user creating the view must have the SELECT privilege WITH GRANT OPTION on the table TABLEOWNER.SALES.

2. Create the table containing user-specific authorization data:

```
CREATE COLUMN TABLE "VIEWOWNER"."AUTHORIZATION_VALUES"("VALUE" VARCHAR(256),
    "USER_NAME" VARCHAR(20));
```

3. Insert authorization information for user MILLER:

```
INSERT INTO "VIEWOWNER"."AUTHORIZATION_VALUES" VALUES('2008', 'MILLER');
```

4. Create the analytic privilege using a subquery as the condition provider:

```
CREATE STRUCTURED PRIVILEGE AP_ROW_VIEW_SALES_ON_SALES FOR SELECT
     ON "VIEWOWNER"."ROW_VIEW_SALES_ON_SALES"
WHERE (CURRENT_DATE BETWEEN 2015-01-01 AND 2015-01-11) AND YEAR IN (SELECT
VALUE FROM VIEWOWNER.AUTHORIZATION_VALUES WHERE USER_NAME = SESSION_USER)
;
```

> → Remember
>
> - Subqueries allow you to create complex filter conditions, but remember: A **calculation view** cannot be secured using an SQL-based analytic privilege that contains a complex filter condition if the view is defined on top of analytic and/or attributes views that themselves are secured with an SQL-based analytic privilege with a complex filter condition.
> - The user creating the analytic privilege must have the SELECT privilege on the objects involved in the subquery, in this case table VIEWOWNER.AUTHORIZATION_VALUES.
> - The session user is the database user who is executing the query to access a secured view. This is therefore the user whose privileges must be checked. For this reason, the table containing the authorization information needs a column to store the user name so that the subquery can filter on this column using the SQL function SESSION_USER.

> ⚠ Caution
>
> While application user can also be used, it is not recommended. The application user is unreliable because it is controlled by the client application. For example, it may set the application user to a technical user or it may not set it at all. In addition, the trustworthiness of the client application cannot be guaranteed.

5. Grant the SELECT privilege on the view VIEWOWNER.ROW_VIEW_SALES_ON_SALES to user MILLER.

```
GRANT SELECT ON "VIEWOWNER"."ROW_VIEW_SALES_ON_SALES" TO MILLER;
```

> → Remember
>
> Only the view owner or a user who has the SELECT privilege WITH GRANT OPTION on the view can perform the grant.

6. Grant the analytic privilege to user MILLER.

```
GRANT STRUCTURED PRIVILEGE AP_ROW_VIEW_SALES_ON_SALES TO MILLER;
```

> → Remember
>
> Owner of the analytic privilege or any user who was granted the analytic privilege with GRANT OPTION can grant it.

Example 3: Secure a Column View Using an SQL-Based Analytic Privilege with a Dynamically Generated Filter Clause

Prerequisites

The database user TABLEOWNER has set up a calculation scenario based on the table SALES_TABLE, which contains the data to be protected.

Context

All sales data is contained in a single view. You want to restrict access of user ADAMS so that he can see only information about cars bought by customer Company A or bikes sold in 2006. You want to do this by creating an analytic privilege with a dynamically generated filter clause.

With a dynamically generated filter clause, the SQL WHERE clause that specifies the filter condition is generated every time the analytic privilege is evaluated. This is useful in an environment in which the filter clause changes very dynamically.

> → Tip
>
> In the following procedure, you might find it easier to use the graphical editors to create the calculation view and analytic privilege.

Procedure

1. Create the view containing the sales data:

```
CREATE COLUMN VIEW "TABLEOWNER"."VIEW_SALES" TYPE CALCULATION WITH PARAMETERS
('PARENTCALCINDEXSCHEMA'='TABLEOWNER',
    'PARENTCALCINDEX'='CALCSCEN_SALES',
    'PARENTCALCNODE'='SALES_TABLE',
    'REGISTERVIEWFORAPCHECK'='0') STRUCTURED PRIVILEGE CHECK
;
```

2. Create a table containing user-specific filter strings:

```
CREATE COLUMN TABLE "AUTHORIZATION"."AUTHORIZATION_FILTERS"("FILTER"
VARCHAR(256),
    "USER_NAME" VARCHAR(20))
;
```

3. Create an authorization filter for user ADAMS:

```
INSERT
INTO "AUTHORIZATION"."AUTHORIZATION_FILTERS" VALUES('(CUSTOMER=''Company A''
AND PRODUCT=''Car'') OR (YEAR=''2006'' AND PRODUCT=''Bike'')',
    'ADAMS')
;
```

> → Remember
>
> Filters containing comparative conditions must be defined as specified in example 1.

4. Create the database procedure that provides the filter clause for the analytic privilege and grant it to object owner of the project:

```
CREATE PROCEDURE "PROCOWNER"."GET_FILTER_FOR_USER"(OUT OUT_FILTER
VARCHAR(5000))
LANGUAGE SQLSCRIPT SQL SECURITY DEFINER READS SQL DATA AS
    v_Filter VARCHAR(5000);
    CURSOR v_Cursor FOR SELECT "FILTER" FROM
"PROCOWNER"."AUTHORIZATION_FILTERS" WHERE "USER_NAME" = SESSION_USER;
BEGIN
    OPEN v_Cursor;
    FETCH v_Cursor INTO v_Filter;
    OUT_FILTER := v_Filter;
    CLOSE v_Cursor;
END;
GRANT EXECUTE ON "PROCOWNER"."GET_FILTER_FOR_USER";
```

> → Remember
>
> When using procedures as the condition provider in an SQL-based analytic privilege, remember the following:
>
> - Procedures must have the following properties:
>   - They must have the security mode DEFINER.
>   - They must be read-only procedures.
>   - A procedure with a predefined signature must be used. The following conditions apply:
>   - No input parameter
>   - Only one output parameter for the filter condition string of string type NVARCHAR, VARCHAR, CLOB, or NCLOB

While VARCHAR and NVARCHAR have length limitations of 5000 characters, CLOB and NCLOB can be used to accommodate longer filter strings.

- The procedure may **not** return a complex filter condition, that is a subquery.
- The procedure must be executable by object owner of the project, that is, either object owner of the project must be the owner of the procedure or the owner of the procedure has all privileges on the underlying tables/views with GRANT OPTION and has granted the EXECUTE privilege on the procedure to the object owner of the project.
- The session user is the database user who is executing the query to access a secured view. This is therefore the user whose privileges must be checked. For this reason, the table or view used in the procedure should contain a column to store the user name so that the procedure can filter on this column using the SQL function SESSION_USER.
- If the filter procedure returns an empty or inapplicable filter string in the output parameter, the user also receives a `Not authorized` error.

5. Create the analytic privilege using the procedure as condition provider:

```
CREATE STRUCTURED PRIVILEGE AP_SALES_2 FOR
SELECT ON "TABLEOWNER"."VIEW_SALES" CONDITION PROVIDER
"AUTHORIZATION"."GET_FILTER_FOR_USER";
```

   On evaluation of the analytic privilege for user ADAMS, the WHERE clause (CUSTOMER='Company A' AND PRODUCT='Car') OR (YEAR='2006' AND PRODUCT='Bike'), as provided by the procedure GET_FILTER_FOR_USER, will be used.

6. Grant the SELECT privilege on the view TABLEOWNER.VIEW_SALES to user ADAMS:

```
GRANT SELECT on TABLEOWNER.VIEW_SALES to ADAMS;
```

   → Remember

   Only the view owner or a user who has the SELECT privilege WITH GRANT OPTION on the view can perform the grant.

7. Grant the analytic privilege to user ADAMS:

```
GRANT STRUCTURED PRIVILEGE AP_SALES_2 TO ADAMS;
```

   → Remember

   Owner of the analytic privilege or any user who was granted the analytic privilege with GRANT OPTION can grant it.

8.1.3.2 Structure of XML-Based Analytic Privileges

An analytic privilege consists of a set of restrictions against which user access to a particular attribute view, analytic view, or calculation view is verified. In an XML-based analytic privilege, these restrictions are specified in an XML document that conforms to a defined XML schema definition (XSD).

> ⓘ Note
>
> As objects created in the repository, XML-based analytic privileges are deprecated as of SAP HANA SPS 02. For more information, see SAP Note 2465027.

Each restriction in an XML-based analytic privilege controls the authorization check on the restricted view using a set of value filters. A value filter defines a check condition that verifies whether or not the values of the view (or view columns) qualify for user access.

The following restriction types can be used to restrict data access:

- View
- Activity
- Validity
- Attribute

The following operators can be used to define value filters in the restrictions.

> ⓘ Note
>
> The activity and validity restrictions support only a subset of these operators.

- IN <list of scalar values>
- CP <pattern with *>
- EQ (=), LE (<=), LT (<), GT (>), GE (>=) <scalar value>
- BT <scalar value as lower limit><scalar value as upper limit>
- IS_NULL
- NOT_NULL

All of the above operators, except IS_NULL and NOT_NULL, accept empty strings (" ") as filter operands. IS_NULL and NOT_NULL do not allow any input value.

The following are examples of how empty strings can be used with the filter operators:

- For the IN operator: IN ("", "A", "B") to filter on these exact values
- As a lower limit in comparison operators, such as:
  - BT ("", “XYZ”), which is equivalent to NOT_NULL AND LE "XYZ""GT "", which is equivalent to NOT_NULL
  - LE "", which is equivalent to EQ ""
  - LT "", which will always return false
  - CP "", which is equivalent to EQ ""

  The filter conditions CP "*" will also return rows with empty-string as values in the corresponding attribute.

View Restriction

This restriction specifies to which column views the analytic privilege applies. It can be a single view, a list of views, or all views. An analytic privilege must have exactly one cube restriction.

> ### Example
>
> IN ("Cube1", "Cube2")

> ### Note
>
> When an analytic view is created in the SAP HANA modeler, automatically generated views are included automatically in the cube restriction.

> ### Note
>
> The SAP HANA modeler uses a special syntax to specify the cube names in the view restriction:
> `_SYS_BIC:<package_hierarchy>/<view_name>`
>
> For example:
>
> ```
> <cubes>
>       <cube name="_SYS_BIC:test.sales/AN_SALES" />
>       <cube name="_SYS_BIC:test.sales/AN_SALES/olap" />
> </cubes>
> ```

Activity Restriction

This restriction specifies the activities that the user is allowed to perform on the restricted views, for example, read data. An analytic privilege must have exactly one activity restriction.

> ### Example
>
> EQ "read", or EQ "edit"

> ### Note
>
> Currently, all analytic privileges created in the SAP HANA modeler are automatically configured to restrict access to READ activity only. This corresponds to SQL SELECT queries. This is due to the fact that the attribute, analytic, and calculation views are read-only views. This restriction is therefore not configurable.

Validity Restriction

This restriction specifies the validity period of the analytic privilege. An analytic privilege must have exactly one validity restriction.

> **Example**
>
> GT 2010/10/01 01:01:00.000

Attribute Restriction

This restriction specifies the value range that the user is permitted to access. Attribute restrictions are applied to the actual attributes of a view. Each attribute restriction is relevant for one attribute, which can contain multiple value filters. Each value filter represents a logical filter condition.

> **Note**
>
> The SAP HANA modeler uses different ways to specify attribute names in the attribute restriction depending on the type of view providing the attribute. In particular, attributes from attribute views are specified using the syntax "<package_hierarchy>/<view_name>$<attribute_name>", while local attributes of analytic views and calculation views are specified using their attribute name only. For example:
>
> ```
> <dimensionAttribute name="test.sales/AT_PRODUCT$PRODUCT_NAME">
>         <restrictions>
>           <valueFilter operator="IN">
>             <value value="Car" />
>             <value value="Bike" />
>           </valueFilter>
>         </restrictions>
>       </dimensionAttribute>
> ```

Value filters for attribute restrictions can be static or dynamic.

- A **static** value filter consists of an operator and either a list of values as the filter operands or a single value as the filter operand. All data types are supported except those for LOB data types (CLOB, BLOB, and NCLOB).
  For example, a value filter (EQ 2006) can be defined for an attribute YEAR in a dimension restriction to filter accessible data using the condition YEAR=2006 for potential users.

  > **Note**
  >
  > Only attributes, not aggregatable facts (for example, measures or key figures) can be used in dimension restrictions for analytic views.

- A **dynamic** value filter consists of an operator and a stored procedure call that determines the operand value at runtime.
  For example, a value filter `(IN (GET_MATERIAL_NUMBER_FOR_CURRENT_USER()))` is defined for the attribute `MATERIAL_NUMBER`. This filter indicates that a user with this analytic privilege is only allowed to access material data with the numbers returned by the procedure `GET_MATERIAL_NUMBER_FOR_CURRENT_USER`.

It is possible to combine static and dynamic value filters as shown in the following example.

> **Example**
>
> ```
> <dimensionAttribute name=" test.sales/AT_PRODUCT$PRODUCT_NAME ">
>         <restrictions>
> ```

```
            <valueFilter operator="CP"> <value value="ELECTRO*"/> </
valueFilter>
            <valueFilter operator="IN"> <procedureCall
schema="PROCEDURE_OWNER" procedure="DETERMINE_AUTHORIZED_PRODUCT_FOR_USER" />
</valueFilter >
        </restrictions>
      </dimensionAttribute>
      <dimensionAttribute name=" test.sales/AT_TIME$YEAR ">
        <restrictions>
           <valueFilter operator="EQ"> <value value="2012"/> </valueFilter>
           <valueFilter operator="IN"> <procedureCall
schema="PROCEDURE_OWNER" procedure="DETERMINE_AUTHORIZED_YEAR_FOR_USER" /> </
valueFilter >
        </restrictions>
```

An analytic privilege can have multiple attribute restrictions, but it must have at least one attribute restriction. An attribute restriction must have at least one value filter. Therefore, if you want to permit access to the whole content of a restricted view, then the attribute restriction must specify all attributes.

Similarly, if you want to permit access to the whole content of the view with the corresponding attribute, then the value filter must specify all values.

The SAP HANA modeler automatically implements these two cases if you do not select either an attribute restriction or a value filter.

Example

Specification of all attributes:

```
<dimensionAttributes>
      <allDimensionAttributes/ >
    </dimensionAttributes>
```

Example

Specification of all values of an attribute:

```
<dimensionAttributes>
      <dimensionAttribute name="PRODUCT">
        <all />
      </dimensionAttribute>
    </dimensionAttributes>
```

Logical Combination of Restrictions and Filter Conditions

The result of user queries on restricted views is filtered according to the conditions specified by the analytic privileges granted to the user as follows:

- Multiple analytic privileges are combined with the logical operator OR.
- Within one analytic privilege, all attribute restrictions are combined with the logical operator AND.
- Within one attribute restriction, all value filters on the attribute are combined with the logical operator OR.

Example

You create two analytic privileges AP1 and AP2. AP1 has the following attribute restrictions:

- Restriction R11 restricting the attribute Year with the value filters `(EQ 2006)` and `(BT 2008, 2010)`
- Restriction R12 restricting the attribute Country with the value filter `(IN ("USA", "Germany"))`

Given that multiple value filters are combined with the logical operator OR and multiple attribute restrictions are combined with the logical operator AND, AP1 generates the condition:

```
((Year = 2006) OR (Year BT 2008 and 2010)) AND (Country IN ("USA", "Germany"))
```

AP2 has the following restriction:

Restriction R21 restricting the attribute Country with the value filter `(EQ "France")`

AP2 generates the condition:

```
(Country = "France")
```

Any query of a user who has been granted both AP1 and AP2 will therefore be appended with the following WHERE clause:

```
((Year = 2006) OR (Year BT 2008 and 2010)) AND (Country IN ("USA", "Germany"))) OR
(Country = "France")
```

Related Information

SAP Note 2465027

8.1.3.2.1 Dynamic Value Filters in the Attribute Restriction of XML-Based Analytic Privileges

The attribute restriction of an XML-based analytic privilege specifies the value range that the user is permitted to access using value filters. In addition to static scalar values, stored procedures can be used to define filters.

By using storing procedures to define filters, you can have user-specific filter conditions be determined dynamically in runtime, for example, by querying specified tables or views. As a result, the same analytic privilege can be applied to many users, while the filter values for authorization can be updated and changed independently in the relevant database tables. In addition, application developers have full control not only to design and manage such filter conditions, but also to design the logic for obtaining the relevant filter values for the individual user at runtime.

Procedures used to define filter conditions must have the following properties:

- They must have the security mode DEFINER.
- They must be read-only procedures.
- A procedure with a predefined signature must be used. The following conditions apply:
  - No input parameter

    - Only 1 output parameter as table type with one single column for the IN operator
    - Only 1 output parameter of a scalar type for all unary operators, such as EQUAL
    - Only 2 output parameters of a scalar type for the binary operator BETWEEN
- Only the following data types are supported as the scalar types and the data type of the column in the table type:
    - Date/Time types DATE, TIME, SECONDDATE, and TIMESTAMP
    - Numeric types TINYINT, SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, and DOUBLE
    - Character string types VARCHAR and NVARCHAR
    - Binary type VARBINARY

NULL as Operand for Filter Operators

In static value filters, it is not possible to specify NULL as the operand of the operator. The operators IS_NULL or NOT_NULL must be used instead. In dynamic value filters where a procedure is used to determine a filter condition, NULL or valid values may be returned. The following behavior applies in the evaluation of such cases during the authorization check of a user query:

Filter conditions of operators with NULL as the operand are disregarded, in particular the following:

- EQ NULL, GT NULL, LT NULL, LE NULL, and CP NULL
- BT NULL and NULL

If no valid filter conditions remain (that is, they have all been disregarded because they contain the NULL operand), the user query is rejected with a "Not authorized" error.

> ### Example
>
> Dynamic analytic privilege 1 generates the filter condition (Year >= NULL) and dynamic analytic privilege 2 generates the condition (Country EQ NULL). The query of a user assigned these analytic privileges (combined with the logical operator OR) will return a "Not authorized" error.

> ### Example
>
> Dynamic analytic privilege 1 generates the filter condition (Year >= NULL) and dynamic analytic privilege 2 generates the condition (Country EQ NULL AND Currency = "USD"). The query of a user assigned these analytic privileges (combined with the logical operator OR) will be filtered with the filter Currency = 'USD'.

In addition, a user query is not authorized in the following cases even if further applicable analytic privileges have been granted to the user.

- The BT operator has as input operands a valid scalar value and NULL, for example, BT 2002 and NULL or BT NULL and 2002
- The IN operator has as input operand NULL among the value list, for example, IN (12, 13, NULL)

Permitting Access to All Values

If you want to allow the user to see all the values of a particular attribute, instead of filtering for certain values, the procedure must return "*" and '' '' (empty string) as the operand for the CP and GT operators respectively. These are the only operators that support the specification of all values.

Implementation Considerations

When the procedure is executed as part of the authorization check in runtime, note the following:

- The user who must be authorized is the database user who executes the query accessing a secured view. This is the session user. The database table or view used in the procedure must therefore contain a column to store the user name of the session user. The procedure can then filter by this column using the SQL function SESSION_USER. This table or view should only be accessible to the procedure owner.

  > ⚠ Caution
  >
  > Do not map the executing user to the application user. The application user is unreliable because it is controlled by the client application. For example, it may set the application user to a technical user or it may not set it at all. In addition, the trustworthiness of the client application cannot be guaranteed.

- The user executing the procedure is the _SYS_REPO user. In the case of procedures activated in the SAP HANA modeler, _SYS_REPO is the owner of the procedures. For procedures created in SQL, the EXECUTE privilege on the procedure must be granted to the _SYS_REPO user.
- If the procedure fails to execute, the user's query stops processing and a "Not authorized" error is returned. The root cause can be investigated in the error trace file of the indexserver, `indexserver_alert_<host>.trc`.

When designing and implementing procedures as filter for dynamic analytic privileges, bear the following in mind:

- To avoid a recursive analytic privilege check, the procedures should only select from database tables or views that are not subject to an authorization check based on analytic privileges. In particular, views activated in the SAP HANA modeler are to be avoided completely as they are automatically registered for the analytic privilege check.
- The execution of procedures in analytic privileges slows down query processing compared to analytic privileges containing only static filters. Therefore, procedures used in analytic privileges must be designed carefully.

8.1.3.3 Runtime Authorization Check of Analytic Privileges

When a user requests access to data stored in an attribute, analytic, calculation, or SQL views, an authorization check based on analytic privileges is performed and the data returned to the user is filtered accordingly. The `EFFECTIVE_STRUCTURED_PRIVILEGES` system view can help you to troubleshoot authorization problems.

Access to a view and the way in which results are filtered depend on whether the view is independent or associated with other views (dependent views).

Independent Views

The authorization check for a view that is not defined on another column view is as follows:

1. The user's authorization to access the view is checked.
   A user can access the view if **both** of the following prerequisites are met:
   - The user has been granted the `SELECT` privilege on the view or the schema in which it is located.

     > **Note**
     >
     > The user does not require `SELECT` on the underlying base tables or views of the view.

   - The user has been granted an analytic privilege that is applicable to the view.
     Applicable analytic privileges are those that meet **all** of the following criteria:

XML-Based Analytic Privilege	SQL-Based Analytic Privilege
A view restriction that includes the accessed view	An ON clause that includes the accessed view.
A validity restriction that applies now	If the filter condition specifies a validity period (for example, `WHERE (CURRENT_TIME BETWEEN ... AND ....) AND <actual filter>)`), it must accomodate the time when the user executes the query.
An action in the activity restriction that covers the action requested by the query **Note** All analytic privileges created and activated in the SAP HANA modeler and SAP HANA Web-based Development Workbench fulfill this condition. The only action supported is read access (`SELECT`).	An action in the FOR clause that covers the action requested by the query **Note** All analytic privileges created and activated in the SAP HANA Web-based Development Workbench fulfill this condition. The only action supported is read access (`SELECT`).
An attribute restriction that includes some of the view's attributes	A filter condition that applies to the view **Note** When the analytic privilege is created, the filter is checked immediately to ensure that it applies to the view. If it doesn't, creation will fail. However, if the view definition subsequently changes, or if a dynamically generated filter condition returns a filter string that is not executable with the view, the authorization check will fail and access is rejected.

   If the user has the `SELECT` privilege on the view but no applicable analytic privileges, the user's request is rejected with a `Not authorized` error. The same is true if the user has an applicable analytic privilege but doesn't have the `SELECT` privilege on the view.

2. The value filters specified in the dimension restrictions (XML-based) or filter condition (SQL-based) are evaluated and the appropriate data is returned to the user. Multiple analytic privileges are combined with the logical operator OR.
   For more information about how multiple attribute restrictions and/or multiple value filters in XML-based analytic privileges are combined, see *XML-Based Analytic Privileges*.

Dependent Views

The authorization check for a view that is defined on other column views is more complex. Note the following behavior.

Calculation and SQL views

- Individual views in the hierarchy are filtered according to their respective analytic privileges, which use the logical OR combination.
- The filtered result of the calculation view is derived from the filtered result of its underlying views. This corresponds to a logic AND combination of the filters generated by the analytic privileges for the individual views.

Result filtering on the view is then performed as follows:

- The user has been granted the `SELECT` privilege on the view or the schema that contains the view.
- The user has been granted analytic privileges that apply to the view itself **and** all the other column views in the hierarchy that are registered for a structured privilege check.

Calculation views and SQL views can be defined by selecting data from other views. This can lead to a complex view hierarchy that requires careful design of row-level authorization.

Analytic views

An analytic view can also be defined on attribute views, but this does **not** represent a view dependency or hierarchy with respect to authorization check and result filtering. If you reference an attribute view in an analytic view, analytic privileges defined on the attribute view are not applied.

This represents a view hierarchy for which the prerequisites described above for calculation views also apply.

- Currency or unit conversions
- Calculated attributes
- Calculated measures that use attributes, calculated attributes, or input parameters in their formulas

If an analytic view designed in the SAP HANA modeler contains one of the elements listed below, it will automatically be activated with a calculation view on top. The name of this calculation view is the name of the analytic view with the suffix `/olap`.

Troubleshooting Failed Authorization

Using the `EFFECTIVE_STRUCTURED_PRIVILEGES` system view, you can quickly see:

- Which analytic privileges apply to a particular view, including the dynamic filter conditions that apply (if relevant)
- Which filter is being applied to which view in the view hierarchy (for views with dependencies)
- Whether or not a particular user is authorized to access the view

Query `EFFECTIVE_STRUCTURED_PRIVILEGES` as follows:

```
SELECT * from "PUBLIC"."EFFECTIVE_STRUCTURED_PRIVILEGES" where ROOT_SCHEMA_NAME
= '<schema>' AND ROOT_OBJECT_NAME = '<OBJECT>' AND USER_NAME = '<USER>';
```

Related Information

Structure of XML-Based Analytic Privileges [page 166]
EFFECTIVE_STRUCTURED_PRIVILEGES System View

8.1.4 Package Privileges

Package privileges authorize actions on individual packages in the classic SAP HANA repository.

> ⓘ Note
>
> With SAP HANA XS advanced, source code and web content are not versioned and stored in the SAP HANA database, so package privileges are not used in this context.

Privileges granted on a repository package are implicitly assigned to the design-time objects in the package, as well as to all sub-packages. Users are only allowed to maintain objects in a repository package if they have the necessary privileges for the package in which they want to perform an operation, for example to read or write to an object in that package. To be able perform operations in all packages, a user must have privileges on the root package `.REPO_PACKAGE_ROOT`.

> → Recommendation
>
> We recommend that package privileges be granted on a single package or a small number of specific packages belonging to your organization, rather than on the complete repository.

If the user authorization check establishes that a user does not have the necessary privileges to perform the requested operation in a specific package, the authorization check is repeated on the parent package and recursively up the package hierarchy to the root level of the repository. If the user does not have the necessary privileges for any of the packages in the hierarchy chain, the authorization check fails and the user is not permitted to perform the requested operation.

In the context of repository package authorizations, there is a distinction between native packages and imported packages.

Privileges for Native Repository Packages

A native repository package is created in the current SAP HANA system and expected to be edited in the current system. To perform application-development tasks on **native** packages in the SAP HANA repository, developers typically need the privileges listed in the following table:

Package Privilege	Description
REPO . READ	Read access to the selected package and design-time objects (both native and imported)

Package Privilege	Description
REPO.EDIT_NATIVE_OBJECTS	Authorization to modify design-time objects in packages originating in the system the user is working in
REPO.ACTIVATE_NATIVE_OBJECTS	Authorization to activate/reactivate design-time objects in packages originating in the system the user is working in
REPO.MAINTAIN_NATIVE_PACKAGES	Authorization to update or delete native packages, or create sub-packages of packages originating in the system in which the user is working

Privileges for Imported Repository Packages

An imported repository package is created in a remote SAP HANA system and imported into the current system. To perform application-development tasks on **imported** packages in the SAP HANA repository, developers need the privileges listed in the following table:

> ⓘ Note
>
> It is not recommended to work on imported packages. Imported packages should only be modified in exceptional cases, for example, to carry out emergency repairs.

Package Privilege	Description
REPO.READ	Read access to the selected package and design-time objects (both native and imported)
REPO.EDIT_IMPORTED_OBJECTS	Authorization to modify design-time objects in packages originating in a system other than the one in which the user is currently working
REPO.ACTIVATE_IMPORTED_OBJECTS	Authorization to activate (or reactivate) design-time objects in packages originating in a system other than the one in which the user is currently working
REPO.MAINTAIN_IMPORTED_PACKAGES	Authorization to update or delete packages, or create sub-packages of packages, which originated in a system other than the one in which the user is currently working

8.1.5 Application Privileges

In SAP HANA XS classic, application privileges define the authorization level required for access to an SAP HANA XS classic application, for example, to start the application or view particular functions and screens.

> ⓘ Note
>
> With SAP HANA XS advanced, application privileges are not used. Application-level authorization is implemented using OAuth and authorization scopes and attributes.

Application privileges can be assigned to an individual user or to a group of users, for example, in a role. The role can also be used to assign system, object, package, and analytic privileges. ,You can use application privileges to provide different levels of access to the same application, for example, to provide advanced maintenance functions for administrators and view-only capabilities to normal users.

If you want to define application-specific privileges, you need to understand and maintain the relevant sections in the following design-time artifacts:

- Application-privileges file (`.xsprivileges`)
- Application-access file (`.xsaccess`)
- Role-definition file (`<RoleName>.hdbrole`)

Application privileges can be assigned to users individually or by means of a user **role**, for example, with the "*application privilege*" keyword in a role-definition file (`<RoleName>.hdbrole`) as illustrated in the following code. You store the roles as design-time artifacts within the application package structure they are intended for, for example, `acme.com.hana.xs.app1.roles`.

```
role acme.com.hana.xs.app1.roles::Display
{
    application privilege: acme.com.hana.xs.appl::Display;
    application privilege: acme.com.hana.xs.appl::View;
    catalog schema "ACME_XS_APP1": SELECT;
    package acme.com.hana.xs.app1: REPO.READ;
    package ".REPO_PACKAGE_ROOT" : REPO.READ;
    catalog sql object "_SYS_REPO"."PRODUCTS": SELECT;
    catalog sql object "_SYS_REPO"."PRODUCT_INSTANCES": SELECT;
    catalog sql object "_SYS_REPO"."DELIVERY_UNITS": SELECT;
    catalog sql object "_SYS_REPO"."PACKAGE_CATALOG": SELECT;
    catalog sql object "ACME_XS_APPL"."acme.com.hana.xs.appl.db::SYSTEM_STATE":
SELECT, INSERT, UPDATE, DELETE;
}
```

The application privileges referenced in the role definition (for example, `Display` and `View`) are actually defined in an application-specific `.xsprivileges` file, as illustrated in the following example, which also contains entries for additional privileges that are not explained here.

> **Note**
>
> The `.xsprivileges` file must reside in the package of the application to which the privileges apply.

The package where the `.xsprivileges` resides defines the scope of the application privileges; the privileges specified in the `.xsprivileges` file can only be used in the package where the `.xsprivileges` resides (or any sub-packages). This is checked during activation of the `.xsaccess` file and at runtime in the by the XS JavaScript API `$.session.(has|assert)AppPrivilege()`.

```
{
  "privileges" : [
   { "name" : "View", "description" : "View Product Details" },
   { "name" : "Configure", "description" : "Configure Product Details" },
   { "name" : "Display", "description" : "View Transport Details" },
   { "name" : "Administrator", "description" : "Configure/Run Everything" },
   { "name" : "ExecuteTransport", "description" : "Run Transports"},
   { "name" : "Transport", "description" : "Transports"}
  ]
}
```

The privileges are **authorized** for use with an application by inserting the *authorization* keyword into the corresponding `.xsaccess` file, as illustrated in the following example. Like the `.xsprivileges` file,

the `.xsaccess` file must reside either in the root package of the application to which the privilege authorizations apply or the specific subpackage which requires the specified authorizations.

> **Note**
>
> If a privilege is inserted into the `.xsaccess` file as an authorization requirement, a user must have this privilege to access the application package where the `.xsaccess` file resides. If there is more than one privilege, the user must have at least one of these privileges to access the content of the package.

```
{
    "prevent_xsrf": true,
    "exposed": true,
    "authentication": {
        "method": "Form"
    },
    "authorization": [
       "acme.com.hana.xs.appl::Display",
       "acme.com.hana.xs.appl::Transport"
     ]
}
```

8.1.6 Prerequisites for Granting and Revoking Privileges and Roles

To be able to grant and revoke privileges and roles to and from users and roles, several prerequisites must be met.

The following table lists the prerequisites that a user must meet to grant privileges and roles to another user (or role).

Prerequisites for Granting Privileges

To grant...	The granting user needs...
A system privilege	The system/object privilege being granted and be authorized to grant it to other users and roles
An object privilege on an object that exists only in runtime	
An object privilege on objects deployed to an HDI container	One of the following: • The privilege being granted and be authorized to grant it to other users and roles • Privileges to execute GRANT_CONTAINER_SCHEMA_PRIVILEGES in the container's API schema, or, if the user is a container group administrator, privileges to execute GRANT_CONTAINER_SCHEMA_PRIVILEGES in the container group's API schema. **→ Recommendation** It is recommended that you grant and revoke privileges on objects deployed to an HDI container using a role deployed to the container.

To grant...	The granting user needs...
An analytic privilege deployed to an HDI container	The privilege being granted and be authorized to grant it to other users and roles **→ Recommendation** It is recommended that you grant and revoke analytic privileges deployed to an HDI container using a role deployed to the container.
An object privilege on an activated object created in the repository, such as a calculation view (SAP HANA XS classic model)	The object privilege EXECUTE on the procedure GRANT_PRIVILEGE_ON_ACTIVATED_CONTENT
An object privilege on a schema containing activated objects created in the repository, such as a calculation view (SAP HANA XS classic model)	The object privilege EXECUTE on the procedure GRANT_SCHEMA_PRIVILEGE_ON_ACTIVATED_CONTENT
A package privilege (SAP HANA XS classic model)	The package privilege being granted and be authorized to grant it to other users and roles
An analytic privilege created in the repository (SAP HANA XS classic model)	The object privilege EXECUTE on the procedure GRANT_ACTIVATED_ANALYTICAL_PRIVILEGE
An application privilege (SAP HANA XS classic model)	The object privilege EXECUTE on the procedure GRANT_APPLICATION_PRIVILEGE
Privilege on user ATTACH DEBUGGER	To be the user on which ATTACH DEBUGGER is granted
A role created in runtime	Either: • The role being granted and be authorized to grant it to other users and roles, or • The system privilege ROLE ADMIN
A role deployed to an HDI container	One of the following: • The role being granted and be authorized to grant it to other users and roles, or • The system privilege ROLE ADMIN • Privileges to execute GRANT_CONTAINER_SCHEMA_ROLES in the container's API schema, or, if the user is a container group administrator, privileges to execute GRANT_CONTAINER_SCHEMA_ROLES in the container group's API schema.
A role created in the repository (SAP HANA XS classic model)	The object privilege EXECUTE on the procedure GRANT_ACTIVATED_ROLE

Prerequisites for Revoking Privileges

To revoke ...	The revoking user needs...
A system privilege	To be the user who granted the privilege
An object privilege on an object that exists only in runtime	

To revoke ...	The revoking user needs...
An object privilege on objects deployed to an HDI container	One of the following: • Be the user who granted the privilege • Privileges to execute REVOKE_CONTAINER_SCHEMA_PRIVILEGES in the container's API schema, or, if the user is a container group administrator, privileges to execute REVOKE_CONTAINER_SCHEMA_PRIVILEGES in the container group's API schema. **→ Recommendation** It is recommended that you grant and revoke privileges on objects deployed to an HDI container using a role deployed to the container.
An analytic privilege deployed to an HDI container	Be the user who granted the privilege **→ Recommendation** It is recommended that you grant and revoke analytic privileges deployed to an HDI container using a role deployed to the container.
An object privilege on an activated object created in the repository, such as a calculation view (SAP HANA XS classic model)	The object privilege EXECUTE on the procedure REVOKE_PRIVILEGE_ON_ACTIVATED_CONTENT
An object privilege on schema containing activated objects created in the repository, such as a calculation view (SAP HANA XS classic model)	The object privilege EXECUTE on the procedure REVOKE_SCHEMA_PRIVILEGE_ON_ACTIVATED_CONTENT
A package privilege (SAP HANA XS classic model)	The user who granted the privilege
An analytic privilege created in the repository (SAP HANA XS classic model)	The object privilege EXECUTE on the procedure REVOKE_ACTIVATED_ANALYTICAL_PRIVILEGE
An application privilege (SAP HANA XS classic model)	The object privilege EXECUTE on the procedure REVOKE_APPLICATION_PRIVILEGE
Privilege on user ATTACH DEBUGGER	To be the user on which ATTACH DEBUGGER is granted
A role created in runtime	• To be the user who granted the role, or • The system privilege ROLE ADMIN **Note** With the exception of roles granted by technical user _SYS_REPO, a user with ROLE ADMIN **cannot** revoke roles granted by technical users SYS and _SYS*.
A role created in the repository	The object privilege EXECUTE on the procedure REVOKE_ACTIVATED_ROLE

To revoke ...	The revoking user needs...
A role deployed to an HDI container	One of the following: • Be the user who granted the role • The system privilege ROLE ADMIN • Privileges to execute REVOKE_CONTAINER_SCHEMA_ROLES in the container's API schema, or, if the user is a container group administrator, privileges to execute REVOKE_CONTAINER_SCHEMA_ROLES in the container group's API schema.

Authorization of User _SYS_REPO

If you are implementing your authorization concept using roles and you are creating roles in the repository of the SAP HANA database, the technical user _SYS_REPO is the granting and revoking user. _SYS_REPO **automatically** meets all of the above prerequisites with the exception of those for granting/revoking objects privileges on objects that exist only in runtime. These privileges must be explicitly granted to _SYS_REPO. For more information about roles as repository objects, see the *SAP HANA Security Guide*.

How are HDI roles granted and revoked?

In the SAP HANA deployment infrastructure (HDI), there are two types of user that can grant or revoke roles.

Roles created in an HDI container can be granted to (or revoked from) other roles and users either by an administrator of the container or an administrator of the container group that the container belongs to.

Preferably, roles created in an HDI container are granted or revoked by a container administrator.

For more information about granting and revoking access to objects deployed to an SAP HDI container, see the *SAP HANA Deployment Infrastructure (HDI) Reference*.

Related Information

Create and Authorize a User
Create and Authorize a Restricted User
Repository Roles
SAP HANA Security Guide

8.2 Database Roles

A database role is a collection of privileges that can be granted to either a database user or another role in runtime.

A role typically contains the privileges required for a particular function or task, for example:

- Business end users reading reports using client tools such as Microsoft Excel
- Modelers creating models and reports
- Database administrators operating and maintaining the database and its users

Privileges can be granted directly to users of the SAP HANA database. However, roles are the standard mechanism of granting privileges as they allow you to implement complex, reusable authorization concepts that can be modeled on business roles.

Creation of Roles

Roles in the SAP HANA database can exist as runtime objects only (catalog roles), or as design-time objects that become catalog objects on deployment (database artifact with file suffix `.hdbrole`).

In an SAP HANA XS classic environment, database roles are created in the built-in repository of the SAP HANA database using either the SAP HANA Web Workbench or the SAP HANA studio. These are also referred to as repository roles. In an SAP HANA XS advanced environment, design-time roles are created using the SAP Web IDE and deployed using SAP HANA deployment infrastructure (SAP HANA DI, or HDI).

Design-time roles are created using the SAP Web IDE and deployed using SAP HANA deployment infrastructure (SAP HANA DI, or HDI).

> ⓘ Note
>
> Due to the container-based model of HDI where each container corresponds to a database schema, HDI roles, once deployed, are schema specific.

SAP HANA XS advanced has the additional concept of application roles and role collections. These are independent of database roles in SAP HANA itself. In the XS advanced context, SAP HANA database roles are used only to control access to database objects (for example, tables, views, and procedures) for XS advanced applications. For more information about the authorization concept of XS advanced, see the *SAP HANA Security Guide*.

Role Structure

A role can contain any number of the following privileges:

- **System privileges** for general system authorization, in particular administration activities
- **Object privileges** (for example, SELECT, INSERT, UPDATE) on database objects (for example, schemas, tables, views, procedures, and sequences)

- **Analytic privileges** on SAP HANA information models
- **Package privileges** on repository packages (for example, `REPO.READ`, `REPO.EDIT_NATIVE_OBJECTS`, `REPO.ACTIVATE_NATIVE_OBJECTS`)
- **Application privileges** for enabling access to SAP HANA-based applications developed in an SAP HANA XS classic environment

> ### Note
>
> There are no HDI or XS advanced equivalents in the SAP HANA authorization concept for package privileges on repository packages and applications privileges on SAP HANA XS classic applications. For more information about the authorization concept of XS advanced, see the *SAP HANA Security Guide*.

A role can also contain other roles.

Roles Best Practices

For best performance of role operations, in particular, granting and revoking, keep the following basic rules in mind:

- Create roles with the smallest possible set of privileges for the smallest possible group of users who can share a role (principle of least privilege).
- Avoid granting object privileges at the schema level to a role if only a few objects in the schema are relevant for intended users.
- Avoid creating and maintaining all roles as a single user. Use several role administrator users instead.

Related Information

Create a Database Role
Roles (.hdbrole and .hdbroleconfig)
Authorization in SAP HANA XS Advanced [page 357]
Create a Design-Time Role
Change a Design-Time Role

8.2.1 Predefined Database (Catalog) Roles

Several catalog roles are available by default in the SAP HANA database.

Several predefined catalog roles are delivered with the SAP HANA database. You should not use these roles directly, but instead use them as templates for creating your own roles.

The table below lists the catalog roles delivered with the SAP HANA database.

> ### Note
>
> These roles do not exist in the SAP HANA repository.

Role	Description
CONTENT_ADMIN	This role contains all the privileges required for using the information modeler in the SAP HANA studio, as well the additional authorization to grant these privileges to other users. It also contains system privileges for working with imported objects in the SAP HANA repository. You can use this role as a template for creating roles for content administrators. **⚠ Caution** The CONTENT_ADMIN role is very privileged and should not be granted to users, particularly in production systems. The CONTENT_ADMIN role should only be used as a template.
MODELING	This role contains all the privileges required for using the information modeler in the SAP HANA studio. It therefore provides a modeler with the database authorization required to create all kinds of views and analytic privileges. **⚠ Caution** The MODELING role contains the predefined analytic privilege _SYS_BI_CP_ALL. This analytic privilege potentially allows a user to access all the data in activated views that are protected by XML-based analytic privileges, regardless of any other analytic privileges that apply. Although the user must also have the SELECT object privilege on the views to actually be able to access data, the _SYS_BI_CP_ALL analytic privilege should not be granted to users, particularly in production systems. For this reason, the MODELING role should only be used as a template.
MONITORING	This role contains privileges for full read-only access to all metadata, the current system status in system and monitoring views, and the data collected by the statistics server.
PUBLIC	This role contains privileges for filtered read-only access to the system views. Only objects for which the users have access rights are visible. By default, this role is granted to every user, except restricted users.
BI_META_DATA_CONSUMER (_SYS_BI)	This role gives read access to the SAP HANA analytic catalog (BIMC views) and nothing else. As analytic clients need to query metadata from these tables at runtime, the role can be used to give end users of analytic clients the required privileges. In contrast to the role MODELING or CONTENT_ADMIN, this role doesn't give full access to the schemas _SYS_BI or _SYS_BIC. Therefore, at least SELECT privileges to the relevant views must also be granted.
BI_META_DATA_CONSUMER_HDI (_SYS_BI)	This role is similar to BI_META_DATA_CONSUMER but only for _HDI BIMC views. These are a bit faster but only read translated texts for SAP HANA deployment infrastructure (HDI) based objects. They don't include translated texts for repository objects. **ⓘ Note** As clients will still query the original BIMC views, this role cannot be used for end users consuming analytic clients.

Role	Description
CREATE_INTERMEDIATE_CALCULATION_VIEW (_SYS_BI)	This role contains only the EXECUTE privilege for a procedure that is called to create temporary calculation views used for data preview on inner nodes. Modelers who want to invoke the data preview on an inner calculation view node from the calculation view editor require this EXECUTE privilege. This privilege is also contained in role MODELING. However, since MODELING is a very powerful role that grants users too many privileges, it should not be granted to end users. See also SAP Note 2299622.
RESTRICTED_USER_JDBC_ACCESS	This role contains the privileges required by restricted database users to connect to SAP HANA through the JDBC client interface. This role is intended to be used in conjunction with application-specific roles. It is recommended that the privileges required to use an application are encapsulated within an application-specific role, which is then granted to restricted database users. By including the RESTRICTED_USER_JDBC_ACCESS role in the application-specific role, it can be ensured that application users have only those privileges that are essential to their work.
RESTRICTED_USER_ODBC_ACCESS	This role contains the privileges required by restricted database users to connect to SAP HANA through the ODBC client interface. This role is intended to be used in conjunction with application-specific roles. It is recommended that the privileges required to use an application are encapsulated within an application-specific role, which is then granted to restricted database users. By including the RESTRICTED_USER_ODBC_ACCESS role in the application-specific role, it can be ensured that application users have only those privileges that are essential to their work.

Role	Description
SAP_INTERNAL_HANA_SUPPORT	This role contains system privileges (for example, CATALOG READ) and object privileges (for example, SELECT on SYS schema) that allow access to certain low-level internal system views needed by SAP HANA development support in support situations. All access is read only. This role does not allow access to any customer data. The definition of the low-level internal system views to which this role allows access is not part of the stable end-user interface and might change from revision to revision. To avoid administrators and end users accidentally accessing these internal system views in applications or scripts, this role is therefore subject to several usage restrictions (listed below) and should be granted only to SAP HANA development support users for the their support activities. In detail, this role contains privileges for read-only access to all metadata, the current system status, and the data of the statistics server. Additionally, it contains the privileges for accessing low-level internal system views. Without the SAP_INTERNAL_HANA_SUPPORT role, this information can be selected only by the SYSTEM user. To avoid accidental use of this role in day-to-day activities, the following restrictions apply to the SAP_INTERNAL_HANA_SUPPORT role: • It cannot be granted to the SYSTEM user. • It can only be granted to a limited number of users at the same time. The maximum number of users to which the role can be granted can be configured with the parameter `internal_support_user_limit` in the authorization section of the `indexserver.ini` configuration file. The default value is 1. • It cannot be granted to another role. • It cannot be granted further object privileges. • It can be granted only further system privileges. • With every upgrade of the SAP HANA database, it is reset to its default privileges. To ensure that system administrators are aware that the SAP_INTERNAL_HANA_SUPPORT role is currently granted to one or more users in a system, an information alert is issued every hour by default. This behavior can configured with check 63.
• AFL__SYS_AFL_AFLPAL_EXECUTE • AFL__SYS_AFL_AFLPAL_EXECUTE_WITH_GRANT_OPTION • AFL__SYS_AFL_AFLBFL_EXECUTE • AFL__SYS_AFL_AFLBFL_EXECUTE_WITH_GRANT_OPTION	Predefined roles for application function libraries (AFL): Predictive Analysis Library (PAL) and Business Function Library (BFL) For more information, see the SAP HANA Predictive Analysis Library (PAL) Reference and the SAP HANA Business Function Library (BFL) Reference

Predefined Database Roles for SAP HDI

> **Note**
>
> These roles only exist if HDI is enabled in the database.

Role	Description
`_SYS_DI_OO_DEFAULTS`	This role contains the set of default privileges that are granted to all HDI container object-owner users (`<container>`#OO users). SAP HANA Deployment Infrastructure (HDI) uses this role internally to grant default privileges instead of using the `PUBLIC` role. It contains only privileges to `SYS` views where additional security checks apply. The role contains `SELECT` privileges on the views: `SYS.DUMMY`, `SYS.PROCEDURES`, `SYS.PROCEDURE_PARAMETERS`, `SYS.TABLES`, `SYS.TABLE_COLUMNS`. This role is not intended to be granted to database users. **Note** Do **not** extend this role in a production system.
`_SYS_DI#<container_group>._SYS_DI_OO_DEFAULTS`	This role contains the set of default privileges that are granted to all HDI container object-owner users (`<container>`#OO users) who are part of the container group `<container_group>`. The role does not contain any privileges by default. Privileges granted to a container group-specific role will automatically be available to all HDI container object-owner users of the specified container group. **Note** This role is not intended to be granted to database users.

Related Information

Predefined Repository Roles [page 198]
Predefined Database Roles for XS Advanced [page 353]
SAP HANA Business Function Library (BFL)
SAP HANA Predictive Analysis Library (PAL)
Alert Configuration
SAP Note 2299622

8.2.2 Catalog Roles and Design-Time Roles Compared

It is possible to create roles as pure runtime objects that follow classic SQL principles or as design-time objects.

In SAP HANA XS classic, database roles are created in the built-in repository of the SAP HANA database using either the SAP HANA Web Workbench or the SAP HANA studio. In SAP HANA XS advanced, design-time roles are created using the SAP Web IDE and deployed using SAP HANA deployment infrastructure (SAP HANA DI, or HDI).

> **Note**
>
> SAP HANA XS classic and the SAP HANA repository are deprecated as of SAP HANA 2.0 SPS 02. For more information, see SAP Note 2465027.

Catalog roles make sense in scenarios where user and role provisioning is carried out solely using a higher-level application that connects to SAP HANA through a technical user such as SAP Identity Management.

The following table summarizes the differences between catalog roles and design-time roles:

Feature	Catalog Roles	Design-Time Repository Roles	Design-Time HDI Roles
Transportability	Roles cannot be transported between systems. They can only be created in runtime by users with the system privilege `ROLE ADMIN`.	Roles can be transported between systems using several transport options: • SAP HANA Application Lifecycle Manager • The change and transport system (CTS+) of the SAP NetWeaver ABAP application server • SAP HANA Transport Container (HTC)	Roles can be transported between systems using the following transport option: • The change and transport system (CTS+) of the SAP NetWeaver ABAP application server
Version management	No version management is possible.	The repository provides the basis for versioning. As repository objects, roles are stored in specific repository tables inside the database. This eliminates the need for an external version control system.	Roles are developed as design-time objects within a project stored in the GIT repository. The complete role history is therefore available in GIT.

Feature	Catalog Roles	Design-Time Repository Roles	Design-Time HDI Roles
Ownership	Roles are owned by the database user who creates them. If the creating user is dropped, any roles created in the user's own schema are also dropped.	The technical user _SYS_REPO is the owner of all roles created in the repository, not the database user who creates them. Therefore, roles are not directly associated with the creating user. To create a role, a database user needs only the privileges required to work in the repository.	Roles are owned by the object owner of the container (technical user `<container>#00`) where role development is taking place. Roles are not directly associated with the creating user. To create a role, a developer needs only the authorization required to work in the relevant space using the SAP Web IDE for SAP HANA. If roles for different purposes are developed in different containers, then roles are not all owned by the same technical user (as is the case with repository roles).

Feature	Catalog Roles	Design-Time Repository Roles	Design-Time HDI Roles
Grant and revoke process	Roles created at runtime are granted directly by the database user using the SQL `GRANT` and `REVOKE` statements. Users with the `ROLE ADMIN` system privilege: • Can create roles and grant any role to any other user or role. Users granted a role with the `ADMIN OPTION`: • Can grant the same role to any other user or role. Any user can grant privileges to a role. If any of these privileges are revoked from the granting user, they are automatically revoked from the role. Roles can be revoked by: • The granting user. • Any user with the `ROLE ADMIN` system privilege. • Users granted the role with `ADMIN OPTION` can revoke this role from any user. **Note** Exception: A user with `ROLE ADMIN` cannot revoke roles granted by the technical users SYS and _SYS*, except for those granted by _SYS_REPO. If the granting user (not necessarily the role creator) is dropped, all roles	Roles are granted and revoked using built-in procedures. Any administrator with the EXECUTE privilege on these can grant and revoke roles.	Roles are granted and revoked using database procedures provided in either the HDI container's or container group's API schema. Any container or container group administrator with the EXECUTE privilege on these procedures can grant and revoke roles. Any user with the system privilege ROLE ADMIN can also grant and revoke roles.

Feature	Catalog Roles	Design-Time Repository Roles	Design-Time HDI Roles
	that they granted are revoked.		

In general, it is recommended that you model roles as design-time objects for the following reasons:

- Unlike roles created in runtime, roles created as design-time objects can be transported between systems. This is important for application development as it means that developers can model roles as part of their application's security concept and then ship these roles or role templates with the application. Being able to transport roles is also advantageous for modelers implementing complex access control on analytic content. They can model roles in a test system and then transport them into a production system. This avoids unnecessary duplication of effort.
- Roles created as design-time objects are not directly associated with a database user. They are created by technical users and granted through the execution of stored procedures. Any user with access to these procedures can grant and revoke a role. Roles created in runtime are granted directly by the database user and can be revoked only by the granting user or a user with the system privilege ROLE ADMIN. Additionally, if the database user is deleted, all roles that he or she granted are revoked. As database users correspond to real people, this could impact the implementation of your authorization concept, for example, if an employee leaves the organization or is on vacation.

Related Information

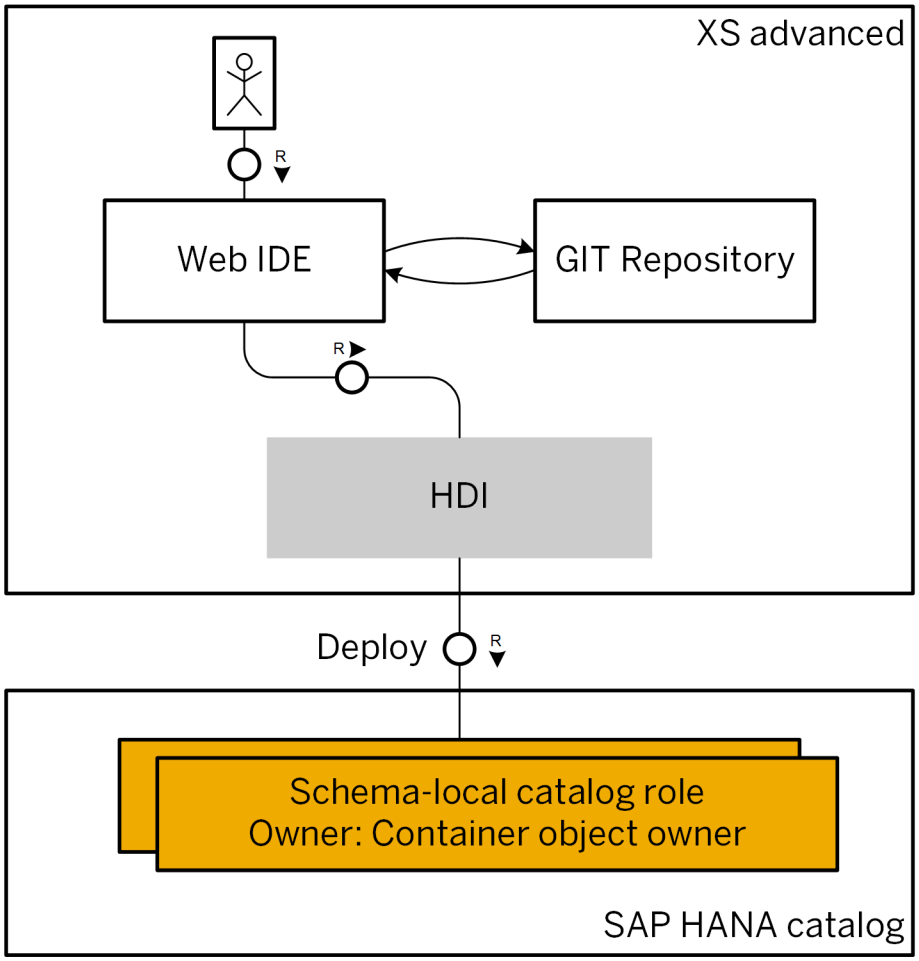
SAP HANA DI Roles [page 191]
Repository Roles [page 194]
SAP Note 2465027

8.2.3 SAP HANA DI Roles

In an SAP HANA XS advanced environment, role developers use the SAP Web IDE for SAP HANA to create the design-time version of roles within a project stored in the GIT repository. When the developer deploys the project, a runtime version of the roles is created in an HDI container (schema) within the SAP HANA database.

Overview

SAP HANA XS advanced provides a comprehensive platform for the development and execution of native data-intensive applications that run efficiently in SAP HANA. It succeeds SAP HANA XS classic as the default application programming model for SAP HANA. This changes the way in which database objects, including roles, are developed. Design-time objects are created within a project stored in the GIT repository. When the developer deploys the project, a runtime version of the objects is created in an HDI container in the SAP HANA database.



Role Development in the SAP HANA Deployment Infrastructure (HDI)

What does the Controller-based model of HDI mean for role development?

In the SAP HANA XS advanced model, development resources may be isolated by leveraging the concept of organizations and spaces. This makes it possible to keep security-related development, like the development of roles, separate from application development. In this way, application developers can't create roles with privileges for database administration for example.

Organizations and spaces also make it possible to develop roles with different purposes separately, for example, roles for database administration and roles for a specific functional area.

For more information about the Controller-based model, see the section on organizations and spaces.

Authorizations Needed for Role Development

The Container Object Owner

According to the authorization concept of the SAP HANA database, a user can only grant a privilege to a user directly or indirectly in a role if the following prerequisites are met:

- The user has the privilege him- or herself
- The user is authorized to grant the privilege to others (`WITH ADMIN OPTION` or `WITH GRANT OPTION`)

A user is also authorized to grant object privileges on objects that he or she owns.

The deployment of database objects with the HDI is based on a container model where each container corresponds roughly to a database schema. Each schema, and the database objects deployed into the schema, are owned by a dedicated technical database user called the **container object owner** (`<container>#OO`). However, this user does not have any "external" privileges, for example system privileges or object privileges on objects in a container in a different space to the role-development container or in an external schema. These privileges must be granted to the container object owner explicitly. If the container object owner does not have all privileges, role deployment will fail with a "missing authorization" error.

There are a number of ways of granting the required privileges to the container object owner. These range from simply granting the privileges to the container object owner directly to more sophisticated methods involving a user-provided service. The various alternatives are described in detail in the document *Best Practices and Recommendations for Developing Roles in SAP HANA* (see Related Information).

> **Note**
>
> The container object owner must always be granted privileges on external objects with the additional parameters `WITH ADMIN OPTION` or `WITH GRANT OPTION`.

The Role Developer

The role developer must have the roles and role collections required to access SAP Web IDE for SAP HANA and to develop in the relevant space and organization.

For more information about the authorization required for developing, see the *SAP HANA Developer Guide for XS Advanced Model*.

Granting and Revoking HDI Roles

HDI roles can be granted and revoked in the following ways:

- With the `GRANT_CONTAINER_SCHEMA_ROLES` and `REVOKE_CONTAINER_SCHEMA_ROLES` procedures of either the container's or the container group's API schema.
  The container administrator and the container group administrator are authorized to execute these procedures. For more information about these procedures and administrator roles, see the *SAP HANA Cloud, SAP HANA Database Deployment Infrastructure Reference*.
- With the `GRANT` and `REVOKE` statements by a user with system privilege `ROLE ADMIN`
  For example `GRANT <role_schema_name>.<role_name> TO <user_name>`, where `<role_schema_name>` is the HDI container name where the role was created.

Roles can be granted and revoked using the SAP HANA cockpit.

Dropping HDI Roles

It is not possible to drop a HDI role by dropping the runtime version of the role using the SQL statement `DROP ROLE`. The role must be undeployed within the container.

Auditing HDI Roles

When role objects are deployed for the first time, a runtime version of the role is created in the corresponding schema in the database using SQL. The same is true if roles are changed or undeployed. You can therefore audit activity related to HDI roles with audit actions `CREATE ROLE`, `ALTER ROLE`, and `DROP ROLE`.

Related Information

SAP HANA Developer Guide for XS Advanced Model (SAP Web IDE)
Organizations and Spaces [page 358]
SAP HDI Administrator Roles
Assign Roles to Database Users
Grant a User a Role from the SAP HDI Container's Schema
Revoke a Role from the SAP HDI Container's Schema
Auditing Activity in SAP HANA [page 277]
Best Practices and Recommendations for Developing Roles in SAP HANA (theoretical part)
Best Practices and Recommendations for Developing Roles in SAP HANA (example project)

8.2.4 Repository Roles

In an SAP HANA XS classic environment, role developers create database roles as design-time objects in the built-in repository of the SAP HANA database using either the SAP HANA Web Workbench or the SAP HANA studio.

> **Note**
>
> SAP HANA XS classic and the SAP HANA repository are deprecated as of SAP HANA 2.0 SPS 02. For more information, see SAP Note 2465027.

Roles created in the repository differ from roles created directly as runtime objects using SQL in several ways.

What authorization does a user need to grant privileges to a role?

According to the authorization concept of the SAP HANA database, a user can only grant a privilege to a user directly or indirectly in a role if the following prerequisites are met:

- The user has the privilege him- or herself
- The user is authorized to grant the privilege to others (`WITH ADMIN OPTION` or `WITH GRANT OPTION`)

A user is also authorized to grant object privileges on objects that he or she owns.

The technical user `_SYS_REPO` is the owner of all objects in the repository, as well as the runtime objects that are created on activation. This means that when you create a role as a repository object, you can grant the following privileges:

- Privileges that have been granted to the technical user `_SYS_REPO` and that `_SYS_REPO` can grant further This is automatically the case for system privileges, package privileges, analytic privileges, and application privileges. Therefore, all system privileges, package privileges, analytic privileges, and application privileges can always be granted in design-time roles.
- Privileges on objects that `_SYS_REPO` owns
  `_SYS_REPO` owns all activated objects. Object privileges on non-activated runtime objects must be explicitly granted to `_SYS_REPO`.

  > ### Note
  > This is true even for objects belonging to schema `SYS`.

  It is recommended that you use a technical user to do this to ensure that privileges are not dropped when the granting user is dropped (for example, because the person leaves the company).

The following table summarizes the situation described above:

Privilege	Action Necessary to Grant in Repository Role
System privilege	None
Package privilege	None
Analytic privilege	None
Application privilege	None
SQL object on activated object (for example, attribute view, analytic view)	None
SQL object privilege on runtime object (for example, replicated table)	Grant privilege to user `_SYS_REPO` with `WITH GRANT OPTION`

> ### Note
> Technically speaking, only the user `_SYS_REPO` needs the privileges being granted in a role, not the database user who creates the role. However, users creating roles in the SAP HANA Web-based Development Workbench must at least be able to **select** the privileges they want to grant to the role. For this, they need either the system privilege `CATALOG READ` or the actual privilege to be granted.

What about the WITH ADMIN OPTION and WITH GRANT OPTION parameters?

When you create a role using SQL (that is, as a runtime object), you can grant privileges with the additional parameters `WITH ADMIN OPTION` or `WITH GRANT OPTION`. This allows a user who is granted the role to grant the privileges contained within the role to other users and roles. However, if you are implementing your authorization concept with privileges encapsulated within roles created in design time, then you do not **want** users to grant privileges using SQL statements. For this reason, it is not possible to pass the parameters `WITH ADMIN OPTION` or `WITH GRANT OPTION` with privileges when you model roles as repository objects.

Similarly, when you grant an activated role to a user, it is not possible to allow the user to grant the role further (`WITH ADMIN OPTION` is not available).

How are repository roles granted and revoked?

It is not possible to grant and revoke activated design-time roles using the `GRANT` and `REVOKE` SQL statements. Instead, roles are granted and revoked through the execution of the procedures `GRANT_ACTIVATED_ROLE` and `REVOKE_ACTIVATED_ROLE`. Therefore, to be able to grant or revoke a role, a user must have the object privilege EXECUTE on these procedures.

How are repository roles dropped?

It is not possible to drop the runtime version of a role created in the repository using the SQL statement `DROP ROLE`. To drop a repository role, you must delete it in the repository and activate the change. The activation process deletes the runtime version of the role.

Can changes to repository roles be audited?

The auditing feature of the SAP HANA database allows you to monitor and record selected actions performed in your database system. One action that is typically audited is changes to user authorization. If you are using roles created in the repository to grant privileges to users, then you audit the creation of runtime roles through activation with the audit action `ACTIVATE REPOSITORY CONTENT`.

Related Information

SAP Note 2465027

8.2.4.1 Repository Roles in the Lifecycle of SAP HANA-Based Applications

Roles are an integral part of developing SAP HANA XS classic applications and their lifecycle. Developers create application-specific objects, including roles, in the repository of the development system. Content administrators transport applications as delivery units to the production system, where they are activated. User administrators grant activated roles to end users.

In application development scenarios, roles are developed like other application-specific artifacts and managed as part of overall application lifecycle management. Roles developed as part of the application encapsulate the privileges required by different user groups to use the application.

The following is a high-level overview of how applications, including application-specific roles, are developed and deployed:

1. Developers build the application by creating the required objects, including roles, in the repository of the development system.
   All development objects, including roles, are stored in packages. Packages that belong to the same application are grouped together into a delivery unit (DU). DUs are the mechanism by which design-time objects in the repository are transported between two systems. They ensure that application-specific objects are transported consistently together within a system landscape.
2. Developers activate their development objects in the development system initially for testing purposes and finally to make them ready for transport.
   The activation process makes the design-time objects available in runtime. In many cases, the runtime objects created are catalog objects, such as schemas, tables, views, and roles.
3. The content administrator transports the application DU from the development system to the production system. This activates the DU and creates runtime objects in the production system.
   Content can be exported and imported using:
   - SAP HANA Application Lifecycle Manager
   - The change and transport system (CTS+) of the SAP NetWeaver ABAP application server
   - HANA Transport Container (HTC)

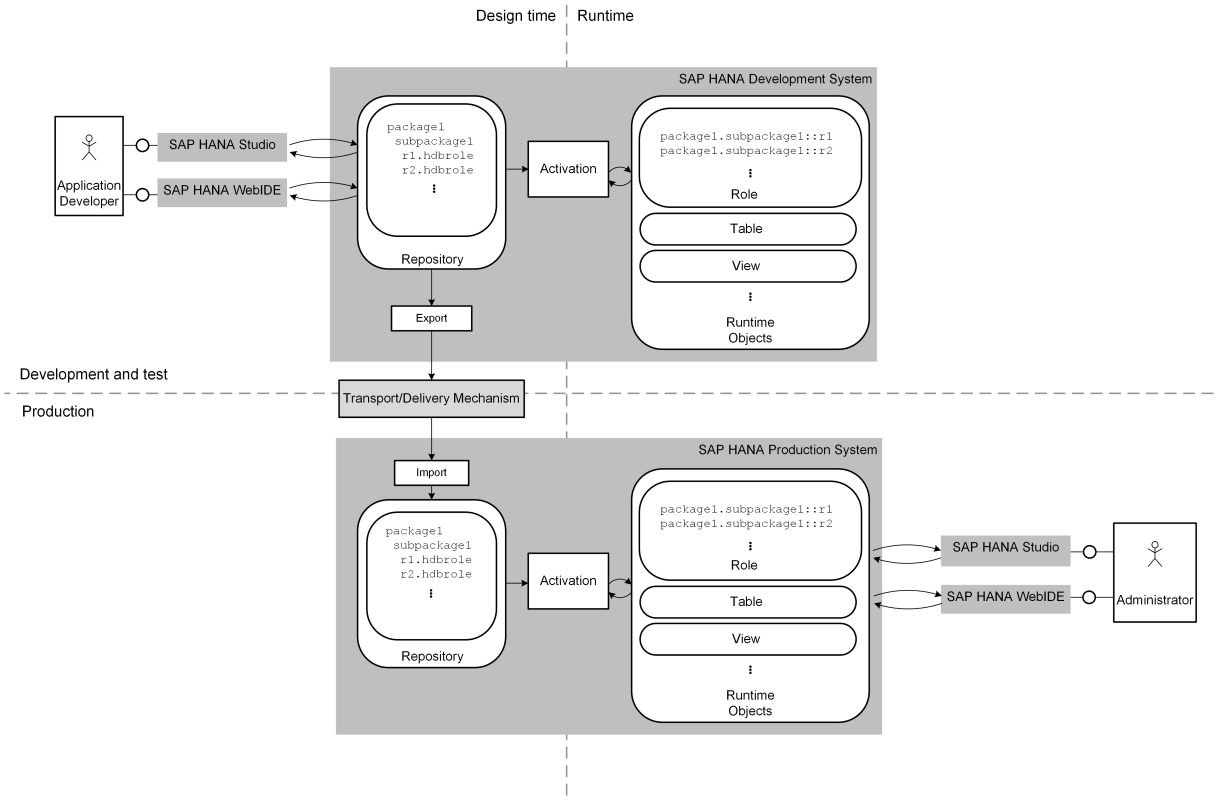
   The transport option used depends on the scenario. For example, CTS+ can be used to transport SAP HANA content in ABAP system landscapes where a CTS transport landscape is already in place.

Once roles are available as runtime objects in the production system, the user administrator can grant them to end users.

> ⚠ Caution
>
> The design-time version of a role in the repository and its activated runtime version should always contain the same privileges. In particular, additional privileges should not be granted to the activated runtime version of a role created in the repository. If a repository role is changed in runtime, the next time the role is activated in the repository, any changes made to the role in runtime will be reverted. It is therefore important that the activated runtime version of a role is not changed in runtime. Although it is not possible to change the activated runtime version of a repository role in the SAP HANA studio, there is no mechanism of preventing a user from doing this at the SQL level.

The following figure illustrates the process described above:

Lifecycle of Repository Roles



Related Information

SAP HANA Application Lifecycle Management
Change and Transport System (Including CTS Plug-In)
How to transport ABAP for SAP HANA applications with HTC

8.2.4.2 Predefined Repository Roles

SAP HANA is delivered with SAP HANA content, a set of pre-installed software components implemented as SAP HANA Web applications, libraries, and configuration data, and developed on the SAP HANA XS, classic model. The privileges required to use a software component delivered as SAP HANA content are contained within repository roles delivered with the component itself.

For more information about the repository roles delivered with SAP HANA content, see *Components Delivered as SAP HANA Content*.

> → Recommendation
>
> As repository roles delivered with SAP HANA can change when a new version of the package is deployed, either do not use them directly but instead as a template for creating your own roles, or have a regular review process in place to verify that they still contain only privileges that are in line with your organization's security policy. Furthermore, if repository package privileges are granted by a role, we

recommend that these privileges be restricted to your organization's packages rather than the complete repository. To do this, for each package privilege (`REPO.*`) that occurs in a role template and is granted on `.REPO_PACKAGE_ROOT`, check whether the privilege can and should be granted to a single package or a small number of specific packages rather than the full repository.

> **Note**
>
> No user has any predefined repository roles initially, except the user `_SYS_REPO` (as the owner of all repository content).

Related Information

Components Delivered as SAP HANA Content [page 420]
Security of SAP HANA Content [page 330]
Package Privileges [page 175]

8.3 Authorization in the Repository of the SAP HANA Database

The authorization concept of SAP HANA applies in the repository of the SAP HANA database.

With the SAP HANA Extended Services (SAP HANA XS) classic development model, developers of SAP HANA-based applications use the built-in repository for storing, versioning, and delivering design-time artifacts such as views, procedures, tables, roles, CDS entities, and Web content exposed via SAP HANA XS classic. The repository provides the basis for concepts like namespaces (through packages), versioning, transport in system landscapes, and software component delivery from SAP or independent software vendors to customers.

Developer Authorization in the Repository [page 200]
To ensure that the process of application development using the SAP HANA Extended Services (SAP HANA XS) classic model is secure, it is important that developers have access to only those repository objects that they actually need to work with.

_SYS_REPO Authorization in the Repository [page 201]
The technical user `_SYS_REPO` is the owner of all objects in the repository, as well as their activated runtime versions. `_SYS_REPO` must be explicitly authorized for objects that are not created in the repository but on which repository objects are modeled.

Granting and Revoking Privileges on Activated Repository Objects [page 202]
Only the _SYS_REPO user has privileges on objects in the repository. Therefore, only this user can grant privileges on them. Since no user can log on as _SYS_REPO, stored procedures are used to grant privileges instead.

8.3.1 Developer Authorization in the Repository

To ensure that the process of application development using the SAP HANA Extended Services (SAP HANA XS) classic model is secure, it is important that developers have access to only those repository objects that they actually need to work with.

The repository of the SAP HANA database consists of packages that contain design-time versions of various objects, such as attribute views, analytic views, calculation views, procedures, analytic privileges, roles, and so on. All repository methods that provide read or write access to content are secured with authorization checks. To allow developers to work with packages in the repository, they must have the required package, system, and object privileges.

The following table explains the privileges that developers require to work in the repository:

Privilege Type	Privileges Required
Package privileges	The SAP HANA repository is structured hierarchically with packages assigned to other packages as sub-packages. If you grant privileges to a user for a package, the user is also automatically authorized for all corresponding sub-packages. In the SAP HANA repository, a distinction is made between native and imported packages. Native packages are packages that were created in the current system and should therefore be edited in the current system. Imported packages from another system should not be edited, except by newly imported updates. An imported package should only be manually edited in exceptional cases. Developers should be granted the following privileges for native packages: • `REPO.READ` • `REPO.EDIT_NATIVE_OBJECTS` • `REPO.ACTIVATE_NATIVE_OBJECTS` • `REPO.MAINTAIN_NATIVE_PACKAGES` Developers should only be granted the following privileges for imported packages in exceptional cases: • `REPO.EDIT_IMPORTED_OBJECTS` • `REPO.ACTIVATE_IMPORTED_OBJECTS` • `REPO.MAINTAIN_IMPORTED_PACKAGES`
System privileges	Developers require the following system privileges to be able to work in the repository: • `REPO.EXPORT` • `REPO.IMPORT` • `REPO.MAINTAIN_DELIVERY_UNITS` • `REPO.WORK_IN_FOREIGN_WORKSPACE` • `REPO.MODIFY_CHANGE`, `REPO.MODIFY_OWN_CONTRIBUTION`, and `REPO.MODIFY_FOREIGN_CONTRIBUTION` These privileges authorize the user to work with SAP HANA Change Recording, which is part of SAP HANA Application Lifecycle Management.

Privilege Type	Privileges Required
Object privileges	To be able to access the repository in the SAP HANA studio or another client, developers need the `EXECUTE` privilege on the database procedure `SYS.REPOSITORY_REST`.

Authorization for SAP HANA Web-based Developer Workbench

If developers are using the SAP HANA Web-based Development Workbench, the privileges required for building and testing development artifacts as well tool access are bundled into the following roles:

- `sap.hana.xs.ide.roles::EditorDeveloper`
- `sap.hana.xs.debugger::Debugger`

For more information, see *SAP HANA Web-Based Development Workbench* in the *SAP HANA Developer Guide (For Web Workbench)*.

Authorization for SAP HANA Application Lifecycle Management

SAP HANA Application Lifecycle Management is a Web-based tool that runs in SAP HANA XS classic. Application developers use this tool to create products, delivery units, packages, and basic application components, while administrators use it to set up the transport of delivery units, start and monitor transports, and upload or download delivery unit archives.

These tasks require different combinations of various privileges. Dedicated roles are available and can be granted to users based on their function (for example, `sap.hana.xs.lm.roles::Administrator`). For more information, see *SAP HANA Application Lifecycle Management*.

Related Information

Package Privileges
System Privileges (Reference) [page 140]
SAP HANA Web-Based Development Workbench
SAP HANA Application Lifecycle Management

8.3.2 _SYS_REPO Authorization in the Repository

The technical user _SYS_REPO is the owner of all objects in the repository, as well as their activated runtime versions. _SYS_REPO must be explicitly authorized for objects that are not created in the repository but on which repository objects are modeled.

The repository of the SAP HANA database consists of packages that contain design-time versions of various objects, such as attribute views, analytic views, calculation views, procedures, analytic privileges, roles, and so

on. Design-time objects must be activated to become runtime objects so that they can be used by end users of SAP HANA and the SAP HANA database.

Inside the repository, only the technical user _SYS_REPO is used. Therefore, this user is the owner of the objects created in the repository and initially is the only user with privileges on these objects. This includes the following objects:

- All tables in the repository schema (_SYS_REPO)
- All activated objects such as procedures, views, analytic privileges, and roles

> **Note**
>
> This does not apply in the case of objects that have been activated using the data preview on intermediate nodes in calculation models. These objects are activated and owned by the user who does the data preview.

Objects in the repository can be modeled on data objects that are not part of design time, such as tables that are used in replication scenarios. _SYS_REPO does not automatically have authorization to access these objects. _SYS_REPO must therefore be granted the SELECT privilege (with grant option) on all data objects behind all objects modeled in the repository. If this privilege is missing, the activated objects will be invalidated. This is true even for objects belonging to schema SYS.

8.3.3 Granting and Revoking Privileges on Activated Repository Objects

Only the _SYS_REPO user has privileges on objects in the repository. Therefore, only this user can grant privileges on them. Since no user can log on as _SYS_REPO, stored procedures are used to grant privileges instead.

Using stored procedures and a technical user for privilege management is beneficial for the following reasons compared to the standard SQL mechanism using the GRANT and REVOKE statements:

- To be able to grant a privilege, a user must have the privilege and be authorized to grant it further. This is not the case when procedures are used. Any user who has the EXECUTE privilege on the relevant grant procedure can grant privileges.

  > **Note**
  >
  > Any user with the system privilege ROLE ADMIN can also grant a role using the GRANT statement.

- If a user grants a privilege or role using the GRANT statement, the privilege or role is automatically revoked when the grantor is dropped or loses the granted privileges.
- Only the grantor can revoke the privilege. With the stored procedures approach, any user with the EXECUTE privilege on the relevant revoke procedure can revoke a granted privilege, regardless of the grantor. If the grantor is dropped, none of the privileges that he or she granted are revoked.

  > **Note**
  >
  > Any user with the system privilege ROLE ADMIN can revoke a role using the REVOKE statement regardless of the grantor.

When the SAP HANA cockpit is used for privilege management, it automatically chooses the suitable method for granting and revoking privileges and roles. So if privileges on activated objects are being granted or revoked, the procedures are used.

> ⚠ Caution
>
> Users who can change and activate objects as well as grant privileges on activated objects have access to all SAP HANA content.

Related Information

Stored Procedures Used to Grant/Revoke Privileges on Activated Repository Objects [page 203]

8.3.3.1 Stored Procedures Used to Grant/Revoke Privileges on Activated Repository Objects

Stored procedures, which exist in the _SYS_REPO schema, are used to grant and revoke privileges on activated modeled objects, analytic privileges, application privileges, and roles.

> ⓘ Note
>
> Public synonyms of these procedures exist. Therefore, these procedures can be called without specifying the schema _SYS_REPO.

Activated Object Type	Procedure Call for Grant and Revoke
Modeled objects, such as calculation views	`CALL GRANT_PRIVILEGE_ON_ACTIVATED_CONTENT ('<object_privilege>','<object>','<user or role>')`
	`CALL REVOKE_PRIVILEGE_ON_ACTIVATED_CONTENT ('<object_privilege>','<object>','<user or role>')`
Schema containing modeled objects	`CALL GRANT_SCHEMA_PRIVILEGE_ON_ACTIVATED_CONTENT ('<analytic_privilege>','<user or role>')`
	`CALL REVOKE_SCHEMA_PRIVILEGE_ON_ACTIVATED_CONTENT ('<analytic_privilege>','<user or role>')`
Analytic privilege	`CALL GRANT_ACTIVATED_ANALYTICAL_PRIVILEGE ('<analytic_privilege>','<user or role>')`
	`CALL REVOKE_ACTIVATED_ANALYTICAL_PRIVILEGE ('<analytic_privilege>','<user or role>')`
Application privilege	`CALL GRANT_APPLICATION_PRIVILEGE ('<application_privilege>','<user or role>')`

Activated Object Type	Procedure Call for Grant and Revoke
	`CALL REVOKE_APPLICATION_PRIVILEGE ('<application_privilege>','<user or role>')`
Role	`CALL GRANT_ACTIVATED_ROLE ('<role>','<user or role>')`
	`CALL REVOKE_ACTIVATED_ROLE ('<role>','<user or role>')`

> **Note**
>
> Object names that are not simple identifiers must be enclosed between double quotes, for example:
>
> `CALL GRANT_APPLICATION_PRIVILEGE ('"com.acme.myApp::Execute"','User')`
>
> This does not apply to the procedures GRANT_ACTIVATED_ROLE and REVOKE_ACTIVATED_ROLE. The role being granted or revoked must not be enclosed in double quotes, for example:
>
> `CALL GRANT_ACTIVATED_ROLE ('acme.com.data::MyUserRole','User')`
>
> For all procedures, the user or role to whom/from whom a privilege or role is being granted/revoked must not be enclosed between double quotes.

8.4 Cross-Database Authorization in Tenant Databases

Read-only queries between tenant databases are possible through the association of the requesting user with a remote identity on the remote database(s). Cross-database access is not enabled by default and must be configured before such user mappings can be set up.

Every tenant database is self-contained with its own isolated set of database users and isolated database catalog. However, to support for example cross-application reporting, cross-database SELECT queries are possible. This means that database objects such as tables and views can be local to one database but be read by users from other databases in the same system.

A user in one database can run a query that references objects in another database if the user is associated with a sufficiently privileged user in the remote database. This associated user is called a remote identity. This is the user who executes the query (or part of the query) in the remote database and therefore the user whose authorization is checked.

For more information about which object types on remote databases can be accessed using this mechanism and which local object types can access remote database objects, see *Cross-Database Access* in the *SAP HANA Administration Guide*.

Example

Assume that we have a system with 2 tenant databases: DB1 and DB2.

USER2 in DB2 wants to query the table SCHEMA1.TABLE1 in DB1, for example, `SELECT * FROM DB1.SCHEMA1.TABLE1`.

This can be achieved as follows:

1. The administrator of DB1 creates a user in DB1 with a remote identity in DB2:

```
CREATE USER USER1 WITH REMOTE IDENTITY USER2 AT DATABASE DB2
```

2. The administrator of DB1 grants user USER1 the privileges required to read the table SCHEMA1.TABLE1:

```
GRANT SELECT ON SCHEMA1.TABLE1 TO USER1 [WITH GRANT OPTION]
```

Now, USER2 in DB2 can select from SCHEMA1.TABLE1 in DB1.

For more information about the syntax notation, see *CREATE USER* in the *SAP HANA SQL and System Views Reference*.

Things to Note About Remote Identities

- A user can be the remote identity for only one user in another database.
- An existing user can be assigned a remote identity using the ALTER USER statement.
- The association between a user and a remote identity is unidirectional. In the above example, USER2 can access SCHEMA1.TABLE1 in DB1 as USER1, but USER1 cannot access objects in DB2 as USER2.
- Only the SELECT privileges of the user in the remote database are considered during a cross-database query. Any other privileges the remote user may have are ignored.
- Before users with remote identities can be created, an administrator must enable cross-database access for the system in the system database and specify which databases can communicate with one another. For more information, see *Enable and Configure Cross-Database Access* in the *SAP HANA Administration Guide*.
  Users receive a `Not authorized` error if they attempt a cross-database operation that is not supported by the current configuration.

System Views for Monitoring Cross-Database Authorization

The following system views contain information about cross-database authorization in a tenant database:

- USERS (SYS)
  The column HAS_REMOTE_USERS indicates whether or not a particular user in the local database has remote identities in other databases.
- REMOTE_USERS (SYS)
  This system view shows which local users can be used by users on other databases for cross-database query execution.

> **Note**
>
> The system views EFFECTIVE_PRIVILEGES and ACCESSIBLE_VIEWS **do not** include privileges that a user has through a remote identity.

Related Information

Cross-Database Access
CREATE USER Statement (Access Control)
ALTER USER Statement (Access Control)
USERS System View
REMOTE_USERS System View
Enable and Configure Cross-Database Access

8.5 LDAP Group Authorization

The Lightweight Directory Access Protocol (LDAP) is an application protocol for accessing directory services. If you use an LDAP-compliant directory server to manage users and their access to resources, you can leverage LDAP group membership to authorize users.

LDAP groups can be mapped to SAP HANA roles. This allows SAP HANA to determine which roles to assign to users based on their membership in one or more LDAP groups, either directly or indirectly through nested groups. Users' access to requested resources is then determined by the privileges defined in the SAP HANA roles.

LDAP group membership can be used to authorize existing SAP HANA users. If you're also using the LDAP server for user authentication in SAP HANA, the required user can be created automatically. For more information, see the section on LDAP user authentication.

SAP HANA uses OpenLDAP client library version 2.4.46 for LDAP authorization and LDAP authentication.

Implementation Considerations

- LDAP group authorization is not supported for cross-database queries between tenant databases. The remote identity of a user must be authorized using the standard mechanism.
- LDAP group authorization is not supported for SAP HANA Extended Application Services, advanced users, that is application users, including those that are created in SAP HANA when SAP HANA is used as identity provider. In addition, the use of SAP HANA XS advanced roles and role collections is not affected by LDAP authorization.
- In an active/active (read enabled) scenario, users are always authenticated on the primary system even if they are connecting to the secondary system, either directly or on the basis of statement routing. This means that roles are granted to the connecting user (based on LDAP group membership) on the primary system. The role assignment is then propagated on the secondary system. However, the user may not immediately have all roles on the secondary system due to a lag in propagation.
- LDAP group authorization can be disabled in tenant databases if it is not required. If any existing users are configured for LDAP group authorization when the feature is disabled, they won't be able to log on. However, the authorization mode of these users can be changed to the standard mechanism.

Related Information

LDAP User Authentication [page 130]
Restricted Features in Tenant Databases [page 414]

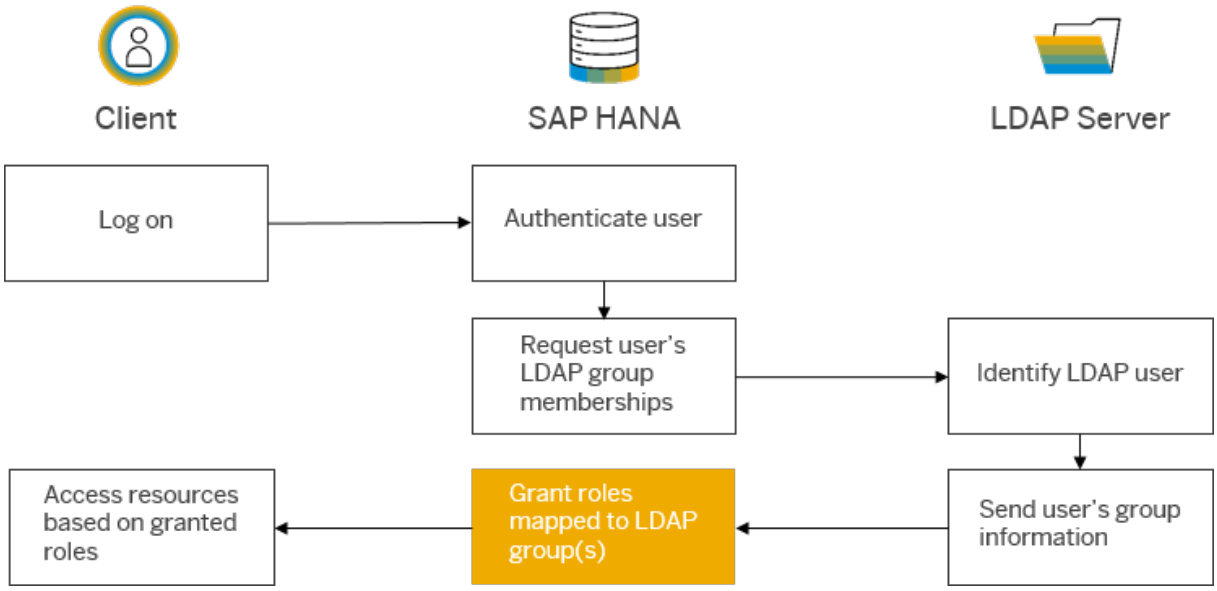
8.5.1 LDAP Group Authorization for Existing Users

Use LDAP group membership to authorize existing SAP HANA database users.

- Overview [page 207]
- Implementing LDAP Group Authorization [page 208]
- Role Reuse Duration [page 209]

Overview

The following figures illustrates how an SAP HANA user is authorized on the basis of his or her LDAP group membership.



Authorization of User Configured for LDAP Group Authorization

> **Note**
>
> If the user is not a member of any LDAP groups, or the groups of which he or she is a member are not mapped to any SAP HANA roles, user logon fails even if authentication was successful.

> **Note**
>
> SAP HANA supports several user authentication mechanisms, including LDAP authentication. When LDAP authentication is used, the required database user can be created automatically and authorized based

on LDAP group membership. For more information about how this works, see the section on LDAP user authentication.

Implementing LDAP Group Authorization

To implement LDAP group authorization, the following high-level steps are required.

Map LDAP groups to SAP HANA roles

LDAP groups can be mapped to SAP HANA catalog roles using the `CREATE ROLE` or `ALTER ROLE` statements. A role that has an LDAP group mapping can be granted to users and other roles as usual. If the role is deleted, it is also revoked from users as usual. Mappings of LDAP groups to this role are also deleted.

Example

```
CREATE ROLE Securities_DBA LDAP GROUP
“cn=Securities_DBA,OU=Application,OU=Groups,ou=DatabaseAdmins,cn=Users,o=verylarg
ebank.com”
```

Note

LDAP group mappings cannot be defined in the definition of design-time roles (database artifact with file suffix .hdbrole). This includes both roles created in the built-in repository of the SAP HANA database (repository roles), and those created using the SAP Web IDE and deployed using SAP HANA deployment infrastructure (SAP HANA DI). However, for roles deployed using SAP HANA DI, it is possible to map LDAP groups to the activated catalog role.

Configure connection to LDAP server in SAP HANA

You set up the LDAP server connection by creating an LDAP provider in the SAP HANA database with the CREATE LDAP PROVIDER or ALTER LDAP PROVIDER statements.

Access to the LDAP server takes place using an LDAP server user with permission to perform searches as specified by the user look-up URL. The credential of this user is stored in the secure internal credential store.

Communication between SAP HANA and the LDAP server can be secured using the TLS/SSL protocol or Secure LDAP protocol (LDAPS). For more information, see the section on secure communication between SAP HANA and an LDAP directory server.

Configure SAP HANA users for LDAP group authorization

To be granted roles on the basis of LDAP group membership, a user must be configured for authorization mode `LDAP`. You can configure the authorization mode for a user in the SAP HANA cockpit or using the `CREATE USER` or `ALTER USER` statement:

Example

```
CREATE USER USER1 PASSWORD <password> AUTHORIZATION LDAP
```

If the LDAP server is also being used for user authentication, the LDAP provider in SAP HANA can configured for automatic user creation. In this way, the required database user is automatically created with authorization mode `LDAP`.

A user with authorization mode `LDAP` is granted roles exclusively based on their LDAP group membership. It is not possible to grant such a user other roles or privileges directly. The default user authorization mode is `LOCAL`. This means that the user must be granted roles and privileges by a user administrator.

> → Tip
>
> To see which authorization mode is configured for a user, refer to the `AUTHORIZATION_MODE` column of the `USERS` system view.

When the authorization mode of a user is switched, the following changes happen:

From LOCAL to LDAP	From LDAP to LOCAL
• Roles based on LDAP group membership are determined and granted the next time the user logs on. • Previously granted roles and privileges are revoked, except roles that were granted by user SYS, for example the PUBLIC role, as well as the CREATE ANY privilege on user's own schema.	• Roles that are granted based on LDAP group membership are revoked. • The time of the last successful LDAP authorization refresh is set to NULL in the system view `LDAP_USERS`.

For more information about configuring LDAP authorization, see the *SAP HANA Administration Guide*.

Role Reuse Duration

To avoid users' LDAP group membership having to be re-evaluated and roles assigned every time they log on, role assignments are stored during the first connection and reused for subsequent connections for a specified period of time, referred to as reuse duration

When this reuse duration expires, LDAP group membership is re-evaluated and roles are reassigned during a subsequent user connection. In this way, high-frequency short-duration connections are efficiently handled.

If the user is logged on in multiple concurrent sessions when the reuse duration expires, LDAP group membership is re-evaluated and roles reassigned for a subsequent user session. Remaining user sessions reuse the role information obtained by the new session.

The default reuse duration is 240 minutes (4 hours) and can be configured using the system property `[authorization] ldap_authorization_role_reuse_duration` in the `indexserver.ini` configuration file.

- If `ldap_authorization_role_reuse_duration` is set to 0, users' LDAP group membership is re-evaluated and roles are granted on every logon.
- If `ldap_authorization_role_reuse_duration` is set to -1, users' LDAP group membership is never re-evaluated, initially granted roles are never revoked.

> ✤ Example
>
> If the reuse duration is set to 60 minutes and the user logs on for the first time at 9:00 am, any new sessions for the user do not have to contact the LDAP server to re-evaluate role assignments until 10:00 am.

Related Information

LDAP User Authentication [page 130]
Configure LDAP Authentication and Authorization
Certificate Management in SAP HANA [page 303]
Secure Internal Credential Store [page 250]
Secure Communication Between SAP HANA and an LDAP Directory Server [page 59]
CREATE ROLE Statement (Access Control)
ALTER ROLE Statement (Access Control)
ALTER LDAP PROVIDER Statement (Access Control)
CREATE LDAP PROVIDER Statement (Access Control)
CREATE USER Statement (Access Control)
ALTER USER Statement (Access Control)

8.6 Shared Business Authorizations in SAP HANA

The basic layer of authorization for ABAP-based SAP applications such as S/4 HANA is provided by "authorization objects" in the SAP NetWeaver Application Server for ABAP. It is possible to create analytic privileges in SAP HANA that reuse these authorizations for read access.

Analytic privileges that reuse ABAP authorization objects can be used to secure access to data in SAP HANA that is accessed either directly or through SAP smart data access. When SAP HANA reuses ABAP authorization objects, it is directly accessing user data in the tables maintained by the ABAP system. Therefore, you can also use the mechanism described here for replication scenarios in which you replicate data from an ABAP system (including user and authorization information) into an SAP HANA system.

> **Note**
>
> When you reuse ABAP authorization objects in SAP HANA, access to views in SAP HANA is based on the authorizations as they are maintained in the ABAP system. This means that the names of the database user in SAP HANA executing the query and the name of the user on the ABAP server have to match.

Being able to create analytic privileges based on ABAP authorizations means that you can provide consistent access to your SAP business data from both SAP applications and new applications built using the SAP HANA extended application services, advanced model. It also reduces the effort of maintaining your authorization model.

Securing a View with Shared Business Authorizations

To secure a view using an analytic privilege that reuses ABAP authorization objects, take the following high-level steps.

1. In the ABAP system, determine the relevant ABAP authorization objects for your scenario.

→ Tip

Objects are bundled into roles. You could look at the roles relevant for your scenario and then see what objects are in them.

2. In SAP HANA, create a view registered for an authorization check based on analytic privileges (STRUCTURED PRIVILEGE CHECK parameter).

{·} Sample Code

```
CREATE VIEW TABLEOWNER.SALES_VIEW as (select * from SALES_DATA) WITH
STRUCTURED PRIVILEGE CHECK;
```

3. Create a helper procedure that calls the built-in procedure `SYS.GENERATE_STRUCTURED_PRIVILEGE_PFCG_CONDITION`.
   The input parameters passed to this procedure generate the filter condition for restricting access based on the specified authorization objects for the current user and client. For more information, see *GENERATE_STRUCTURED_PRIVILEGE_PFCG_CONDITION (SYS)*.

ⓘ Note

The procedure must be executable by _SYS_REPO.

{·} Sample Code

```
CREATE PROCEDURE TABLEOWNER.SALES_CONDITION_PROVIDER(OUT result_condition
NCLOB) LANGUAGE SQLSCRIPT SQL SECURITY DEFINER READS SQL DATA AS
BEGIN
CALL GENERATE_STRUCTURED_PRIVILEGE_PFCG_CONDITION
(
'A_TEST_SCHEMA',
'CHECKID1 AND NOT CHECKID2',
'{
      "data":
      {
          "CHECKID1":
            {
                  "authobj":"OBJ1",
                  "filter":[{"key":"ACTVT","valueList":["03"]}],
                  "mappings":[{"fieldName":"TEST01",
"mappedName":"LIFECYCLE_STATUS"}]
            },
    "CHECKID2":
            {
                  "authobj":"OBJ2",
                  "filter":[{"key":"ACTVT","valueList":["03"]}],
                  "mappings":[{"fieldName":"TEST02",
"mappedName":"BILLING_STATUS"}]
            }
      }
}',
result_condition);
END;
GRANT EXECUTE ON TABLEOWNER.SALES_CONDITION_PROVIDER_NAME TO _SYS_REPO;
```

4. Create an SQL-based analytic privilege, using the helper procedure to specify the filter condition (CONDITION PROVIDER clause).

Sample Code

```
CREATE STRUCTURED PRIVILEGE SALES_AP FOR SELECT ON TABLEOWNER.SALES_VIEW
CONDITION PROVIDER TABLEOWNER.SALES_CONDITION_PROVIDER;
```

5. Include the analytic privilege in a role and assign the role to end users.

Result

When a user requests access to data stored in the views, an authorization check based on the analytic privilege is performed and the data returned to the user is filtered according to their ABAP authorizations. The ABAP authorizations to be used to authorize access are determined by the value of the XS_APPLICATIONUSER session context variable. Therefore, the value of XS_APPLICATIONUSER must match a user name in the ABAP authorization data. By default, XS_APPLICATIONUSER is set to the name of the database user that has opened the connection to SAP HANA. In SAP HANA XS advanced scenarios, XS_APPLICATIONUSER is automatically set to the name of the business user who has logged in to the XSA application.

Related Information

ABAP Authorization Concept
GENERATE_STRUCTURED_PRIVILEGE_PFCG_CONDITION (SYS) [page 212]
Structure of SQL-Based Analytic Privileges [page 156]

8.6.1 GENERATE_STRUCTURED_PRIVILEGE_PFCG_CONDITION (SYS)

SQLScript procedure to generate a filter condition based on ABAP authorization objects that can be used in the CONDITION PROVIDER clause of an analytic privilege

Procedure Call

```
CALL SYS.GENERATE_STRUCTURED_PRIVILEGE_PFCG_CONDITION(<parameter_list>)
```

Input Parameters

- Schema in which the SAP authorization tables UST12 and USRBF2 reside
- An expression that contains at least one CHECKID

> **Note**
>
> Multiple CHECKIDs may be concatenated. Expression can contain AND, OR and NOT Boolean operators.

- A JSON-formatted string that contains the information required to translate each ABAP authorization object referenced. For each CHECKID, the following must be specified:
  - `authobj`
    This specifies the name of the ABAP authorization object to which the CHECKID is mapped. This may contain an empty value if the CHECKID itself is the name of the ABAP authorization object.
  - `filter`
    This allows you to specify fixed conditions that the user's ABAP authorizations need to match. It is a list of required fields (`key`) and their values (`values`).
  - `mappings`
    This specifies the mapping of FIELD value (`fieldName`) in the ABAP authorization object to column name (`mappedName`) of the target view to be protected.

Output Parameters

An NCLOB that contains the SQL filter condition

> **Note**
>
> The procedure returns 1>1 if the user has no permissions and 1=1 if the user has full access.

Examples

> **Example**
>
> Input:

```
CALL SYS.GENERATE_STRUCTURED_PRIVILEGE_PFCG_CONDITION(
'A_TEST_SCHEMA',
'CHECKID1',
'{"data":
    {
        "CHECKID1":
        {
            "authobj":"OBJ1",
            "filter":[{"key":"ACTVT","valueList":["03"]}],
            "mappings":[{"fieldName":"SACMTSOID", "mappedName":"SO_ID"},
{"fieldName":"SACMTSOLCS", "mappedName":"LIFECYCLE_STATUS"}]
        }
    }
}',
?)
```

Related Information

Shared Business Authorizations in SAP HANA [page 210]

9 SAP HANA Data Masking

Data masking provides an additional layer of access control that can be applied to tables and views.

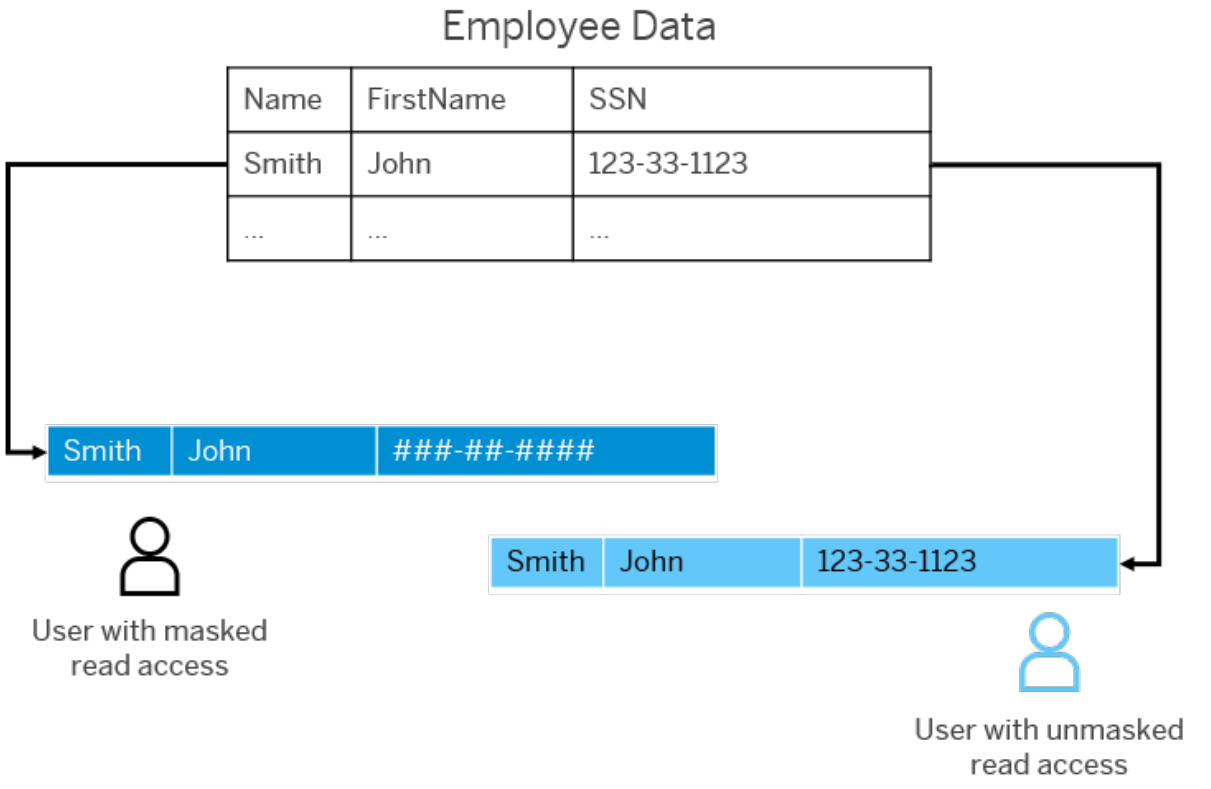
A column mask protects sensitive or confidential data in a particular column of a table or view by transforming the data in such a way that it is only visible partially or rendered completely meaningless for an unprivileged user, while still appearing real and consistent.

> **Note**
>
> Data masking scrambles data during execution only. Data is not encrypted at the physical (disk) layer.

Unlike the all-or-nothing approach of authorization in which an unprivileged user gets a "not authorized" error, a user who is not privileged to see masked data does not get an error, but instead sees only the masked data.

> **Note**
>
> A user still needs the SELECT privilege on a table or view with data masking. Only a user who also has the UNMASKED object privilege for the table or view is able to see the original data.

Masking is useful in situations where strict data protection regulations apply and a second layer of protection is required on top of authorization to safeguard specific data. For example, a highly-privileged administrator has access to company employee data, but instead of seeing the actual content of the column containing social security numbers, he sees masked values (for example, ###-##-####).



Supported Objects

You can specify a mask for columns in rowstore and columnstore tables, as well as in SQL and calculations views.

Masking supports the following string data types:

- VARCHAR
- NVARCHAR
- SHORTEXT
- ALPHANUM

It is also possible to nest views containing masked columns.

Unsupported Objects

Adding mask expressions for the following types of views or tables and/or columns results in a `feature not supported` error:

- Unsupported view types:
  - Join and OLAP views
  - Hierarchy views
  - Parametrized SQL views
  - Transient column views
- Unsupported table types:
  - Temporary tables
  - Extended storage (Dynamic Tiering) and proxy tables
  - Replica tables
  - Virtual tables
  - Document store (DocStore) tables
- Unsupported column types:
  - Columns encrypted using with client-side encryption
  - Calculated and constant columns
- Unsupported object hierarchies:
  - Join and OLAP views cannot be defined on top of tables with masked columns

Implementation Considerations

When masking is used, correlations between masked and unmasked data may allow a user to infer the masked value. For example, when table partitioning is used, and if a masked column is used for range partitioning, a user with the SELECT object privilege on a table or view may be able to use the row ID to determine in which partition a particular row is located.

Related Information

SAP HANA Data Anonymization [page 231]

9.1 Masking Definition

Masking is defined as part of the table or view. The mask mode specifies how access to masked data is determined in object hierarchies, and the mask expression specifies how the data in a column should be returned to unauthorized users.

Column masking is defined directly in the table or view using the CREATE TABLE and CREATE VIEW statements, for example:

```
CREATE VIEW <view_name> (<column_name_list>) AS <subquery>
WITH [DEFAULT | SESSION USER]
MASK (<column_name> USING '<mask_expression>');
```

For column masking, you define:

- **Mask mode**
  When data in masked columns is accessed through **object hierarchies**, the mask mode specifies how the user accessing the data is determined for the authorization check. There are two mask modes: DEFAULT and SESSION USER.
  With DEFAULT mask mode, the privileges of the owner of the next higher-level object with definer security mode are checked. With SESSION USER mask mode, the session user is always subject to the authorization check.
  If you do not set a mask mode, DEFAULT mode is automatically applied.
  For more detailed information about how these mask modes impact the authorization process, see *Authorization in Masked Tables and Views*.
- **Mask expression**
  The mask expression specifies how the data in a specific column should be returned to unauthorized users. It can be specified using any SQL expression that SAP HANA supports, such as SQL functions or user-defined functions. However, it must return data of the same type and length as the original data. This ensures that masked data appears real and consistent.

For example, a new SQL table with DEFAULT mask mode and column mask as a constant expression can be created as follows:

```
CREATE TABLE T(FirstName, Name, MaskedSSN, MaskedCreditcard)
WITH DEFAULT
MASK (MaskedSSN USING 'XXXX-XX-' || RIGHT(MaskedSSN, 4), MaskedCreditcard USING
'XXXX-XXXX-XXXX-XXXX');
```

A view with SESSION USER mask mode can be created as follows:

```
CREATE VIEW V(FirstName, Name, MaskedSSN, MaskedCreditcard) AS SELECT FirstName,
Name, SSN, Creditcard FROM T
WITH SESSION USER
MASK (MaskedSSN USING 'XXXX-XX-' || RIGHT(MaskedSSN, 4), MaskedCreditcard USING
'XXXX-XXXX-XXXX-XXXX');
```

For more information about the syntax of the MASK clause, see the relevant statements in the *SAP HANA SQL and Systems View Reference*.

> **Note**
>
> Table or view creation may fail with the error message below if SAP HANA cannot calculate the length of the resulting mask expression or if it exceeds the column length:`SAP DBTech JDBC: [5735]: Masking: invalid mask expression: exception 4075006: mask expression <masking expression> returns a value potentially too long for the column <column_name>`
>
> In this case, limit the mask expression using other string functions such as:
>
> - SUBSTRING (`<mask_expression>`, 0, `<column_length>`)
> - CAST (`<mask_expression>` as `<column_data_type>`)

> **Note**
>
> You can add masked columns to existing tables and views (for example, to apply masking in an existing application), as well as modify and delete masking definitions using the ALTER TABLE and ALTER VIEW statements. You can add a mask to a column in a calculation view using the SAP Web IDE for SAP HANA.

Related Information

CREATE VIEW Statement (Data Definition)
CREATE TABLE Statement (Data Definition)
ALTER TABLE Statement (Data Definition)
ALTER VIEW Statement (Data Definition)
Mask Column Values in Client Tools (SAP HANA Modeling Guide for SAP Web IDE for SAP HANA)

9.2 Authorization in Masked Tables and Views

The object privilege UNMASKED controls the visibility of unmasked data. In object hierarchies, the mask mode of the object with masked columns determines whose authorization must be checked.

Access to tables or views containing masked columns is controlled by the usual object privileges, for example, SELECT for read access. Data in masked columns is returned in line with the user's privileges, but it is always masked, regardless of query type.

The ability to see the real, unmasked data is controlled by the object privilege UNMASKED.

> **Note**
>
> Not having the UNMASKED privilege does not result in an authorization error, assuming the user is otherwise sufficiently authorized.

As an object privilege, UNMASKED can initially be granted only by the object owner or the owner of the schema that contains the object. For more information, see the section on object privileges.

The following table illustrates how different combinations of the `SELECT` and `UNMASKED` privileges control what a user sees on simple read access to a masked object:

		SELECT privilege	
		Not granted	Granted
UNMASKED privilege	Not granted	Not authorized	###-##-####
	Granted	Not authorized	123-33-1123

Authorization for Simple Read Access

For access to masked data in object hierarchies, the mask mode of the object with masked columns and the authorization of the accessing user determine visibility of unmasked data.

Authorization with DEFAULT Mask Mode

If a masked view or table is defined with DEFAULT mask mode, the SQL security mode (definer or invoker) of higher-level objects determines the user who is actually accessing the masked columns.

About the SQL Security Mode of Complex Objects

Complex SQL objects, such as procedures, functions, or views, can be defined on top of other objects in either "definer" or "invoker" security mode. The security mode determines how the authorization to the underlying objects is checked. A **definer object** uses the privileges of its owner to access the underlying objects. By contrast, an **invoker object** simply provides access through to its underlying objects and expects the querying user to provide all required privileges on the underlying objects.

Examples of definer and invoker objects:

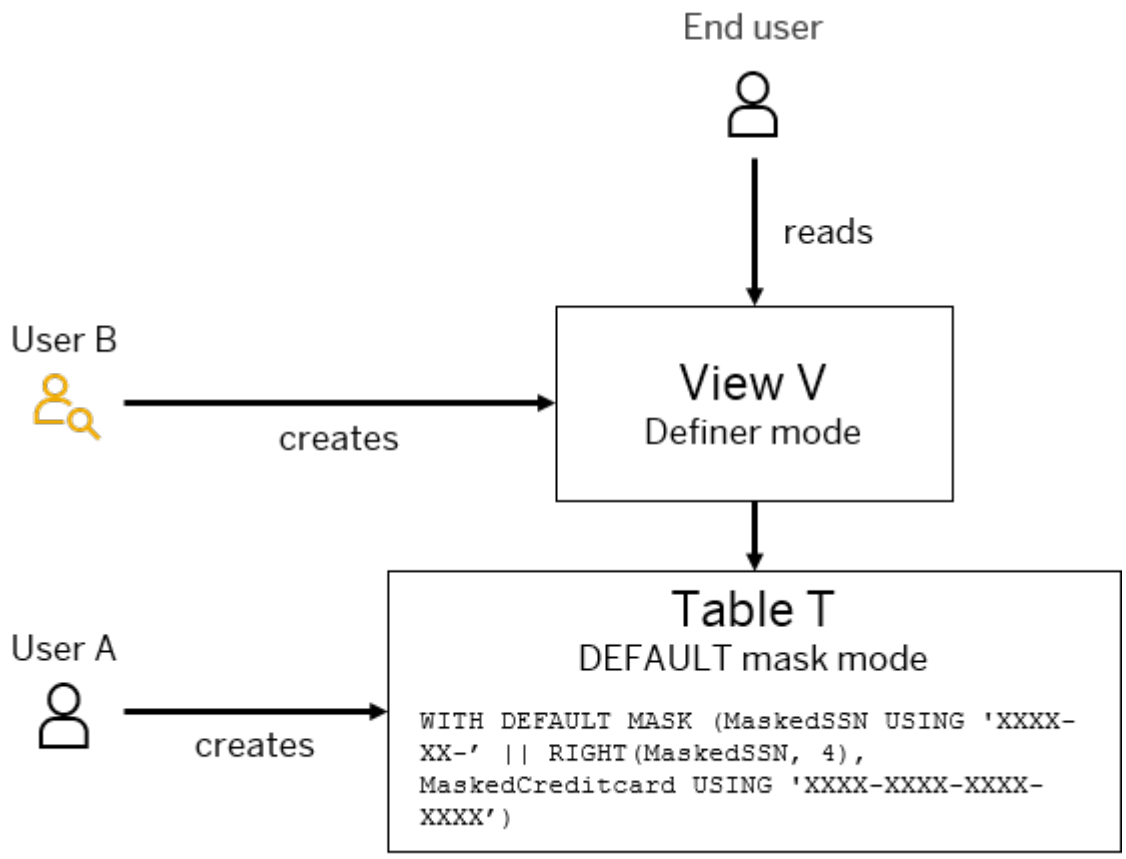
- **Row views** (in other words, views created with the `CREATE VIEW` SQL statement) are considered definer objects as the owner is used to access the underlying objects.
- **SQLScript procedures** can be defined with either definer or invoker security mode.
- **User-defined functions** can be defined with either definer or invoker security mode.

For the syntax of the statements to create these object types, see the *SAP HANA SQL Reference*.

Masking with Definer-Dependent Objects

The owner of a definer object is the user who accesses the underlying objects. This means that if an underlying object contains columns masked using the DEFAULT mask mode, the owner of the definer object is the user who is checked for the UNMASKED privilege. This can be illustrated with the following example.

User A creates a table T with masked columns using the DEFAULT mode, and user B creates a view V on top of T. The end user has read access on V, in other words: the end user has SELECT privilege on V and user B has SELECT WITH GRANT OPTION on T.



Authorization for Object Hierarchy with Definer Dependencies

Whether the end user sees masked or unmasked data from T depends entirely on user B as the **owner of the definer object**:

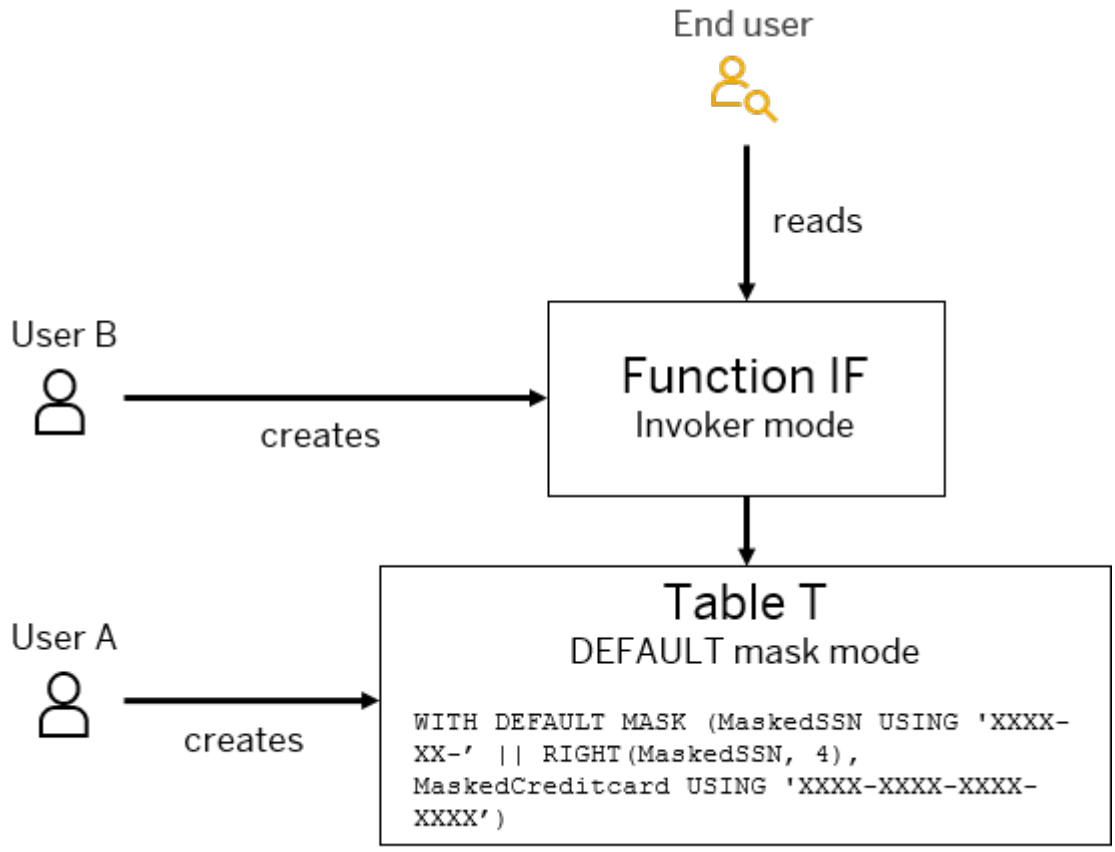
- If user B does not have the UNMASKED privilege on T, data from T is masked with the T mask expression, processed within V and then returned to the end-user.
- If user B does have the UNMASKED privilege on T, unmasked data from T is processed in V and returned to the end-user.

> ⓘ Note
>
> All views, regardless of type, are considered definer objects for the purposes of determining authorization on masked columns. This means that view owners will be checked for the UNMASKED privilege as shown in the example above if they are built on top of tables or other views with DEFAULT mask mode.

Masking with Invoker Dependent Objects

Invoker objects simply provide access through to the underlying objects. This means that if an underlying object contains columns masked using the DEFAULT mask mode, the owner of the invoker object is not considered. Instead, the owner of the next higher-level definer object, if found, or the end user is checked for the UNMASKED privilege. This can be illustrated with the following example.

User A creates a table T with masked columns using DEFAULT masking mode, and user B creates an invoker function IF on top of T. In order to successfully access IF, the end user needs not only EXECUTE privilege on IF, but also as SELECT privilege on T since IF is an invoker object.



Authorization for Object Hierarchy with Invoker Dependencies

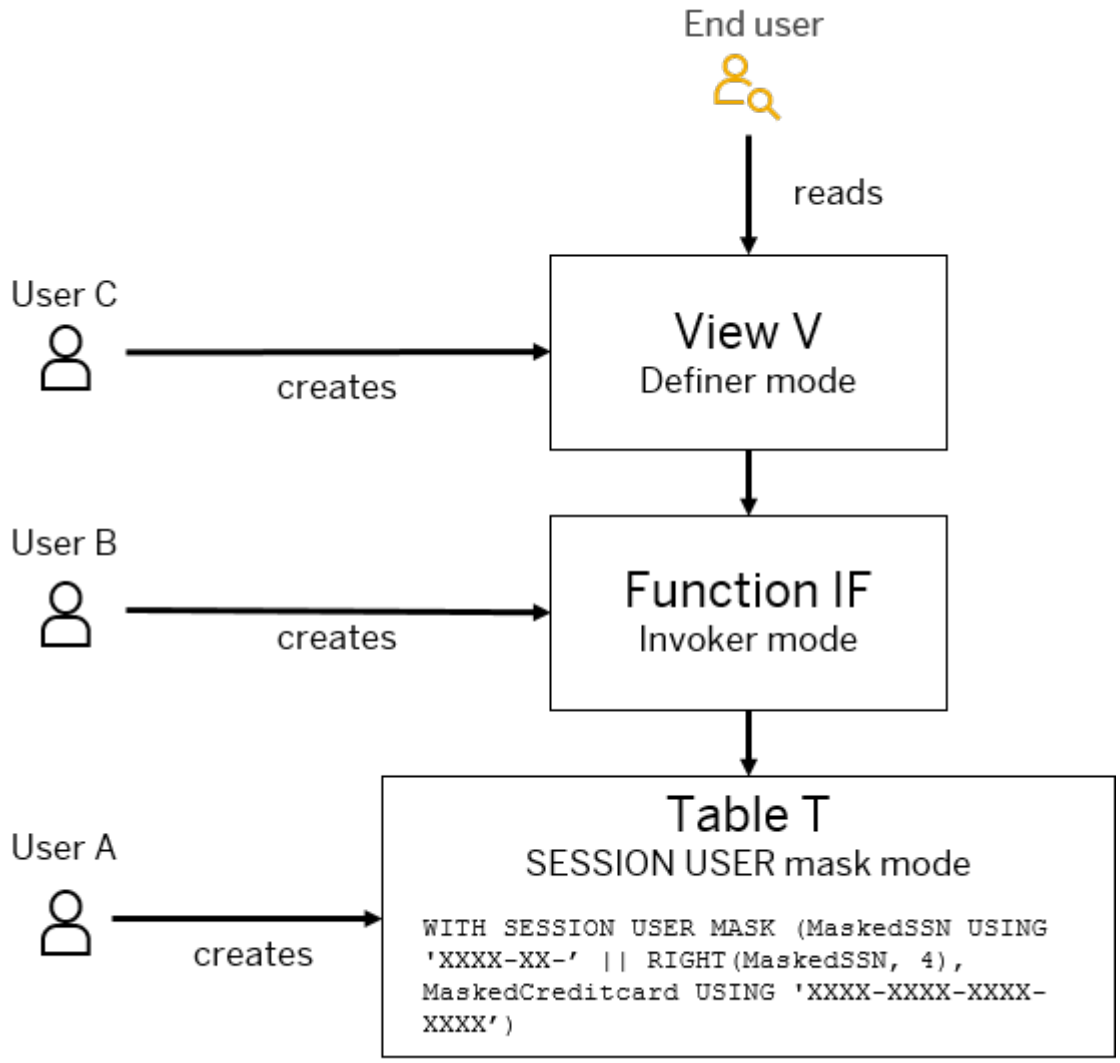
How data in the masked column in T is processed within IF and returned to the end user depends on the whether or not the **end user** has the UNMASKED privilege on T as follows:

- If the end user does not have the UNMASKED privilege on T, data from T is masked with the T mask expression, processed within IF and then returned to the end-user.
- If the end user does have the UNMASKED privilege on T, full data from T is processed in IF and returned to the end-user.

SESSION USER Mask Mode

If a masked view is defined with the SESSION USER mask mode, the session user is always considered the user who accesses the masked columns. This is the user who is checked for the UNMASKED privilege regardless of the security mode of higher-level objects built on top of the object with masked columns. This can be illustrated with the following example.

User A creates a table T with masked columns using SESSION USER masking mode, user B creates an invoker function IF on top of T, and user C creates a definer view V on top of IF. The end user is already able to select from V.



Authorization for Object Hierarchy with SESSION USER Mask Mode

Due to the fact that T uses SESSION USER mask mode, how data in the masked column in T is obtained and further processed by IF and V depends on whether or not the end user has the UNMASKED privilege on T as follows.

- If the end user does not have the UNMASKED privilege on T, data from T is masked with the T mask expression, processed within IF and then returned to the end-user.
- If the end user does have the UNMASKED privilege on T, full data from T is processed in IF and returned to the end-user.

Implementation Considerations

Mask Mode Impact

The mask mode does not always affect the data returned to the end user. In the following situations, the result will be the same regardless of the mask mode:

- Masking is applied to the top-level object in an object hierarchy
  In this case, regardless of the mask mode or security mode of the top-level object, the users accessing the top-level object are checked for the UNMASKED privilege.
- Only procedures and user-defined functions with invoker mode are built on top of an object with masked columns
  The end users accessing any objects in such an object hierarchy are checked for the UNMASKED privilege.

Combining Masking and Analytic Privileges

- Combinations within a single object
  It is not possible to use different mask modes within a single table or a view.
  It is possible to create a view containing masked columns that is also secured with analytic privileges. In this case, the end user must have an applicable analytic privilege in addition to the SELECT privilege. If this is the case, the authorization check first filters the data based on the conditions specified in the analytic privilege, and then masks any data in masked columns. This means that a user who is authorized to see unmasked data still only sees the data that he or she is authorized to see as specified in the analytic privilege.
- Combinations within an object hierarchy
  It is possible to build objects with different mask modes on top of each other in deeply nested hierarchies. However, it is important to validate the results for unauthorized users carefully. While it is not a problem for users with authorization on the complete object hierarchy, different push-down decisions of query optimization might result in different results with masked data for unauthorized users. In particular, if aggregation is executed on top of masked data, wrong results could be returned unnoticed as no error occurs.
  When building analytic privileges on top of masked data, please keep in mind that the way in which you define your mask might influence whether or not a row is filtered in your analytic privilege.

Related Information

CREATE TABLE Statement (Data Definition)
CREATE VIEW Statement (Data Definition)
CREATE PROCEDURE Statement (Procedural)
CREATE FUNCTION Statement (Procedural)
Analytic Privileges [page 154]
Object Privileges [page 146]

9.3 Analysis of Effective Mask Expressions

The system view SYS.EFFECTIVE_MASK_EXPRESSIONS can be used to analyze which mask expressions are currently being applied, for example, if a query on an object is returning unexpected masked data. An administration user with the CATALOG READ system privilege can analyze unexpected query results for other users.

The system view EFFECTIVE_MASK_EXPRESSIONS resolves the complete object hierarchy for a given object and displays the following information for each level of dependency, that is, each pair of dependent object and underlying object:

- The mask expression if the underlying object defines a mask expression of the column of interest
- The actual user accessing the underlying object according to the mask mode of the underlying object and security mode of the dependent object
- If the mask expression is being enforced depending on whether or not the actual accessing user has the UNMASKED privilege on the object with masked columns

> **Note**
>
> Information on masking can also be traced for analysis and debugging purposes using trace component `DATAMASKING` level `INFO` for the `indexserver.ini`.

Related Information

EFFECTIVE_MASK_EXPRESSIONS System View

9.4 Example: Masking Data with DEFAULT Mask Mode

This example shows how to mask data in a view with DEFAULT mask mode.

Procedure

1. User `data_owner` creates a table (`credit_tab`) containing credit card data:

```
CREATE TABLE credit_tab (Name varchar(20), CREDIT_CARD varchar(19));
INSERT INTO credit_tab values ('John', '1111-1111-1111-1111');
INSERT INTO credit_tab values ('James', '2222-2222-2222-2222');
```

2. User `data_owner` creates a view on the credit card table (`credit_view`), masking the credit card number column and specifying DEFAULT mask mode:

```
CREATE VIEW credit_view AS SELECT * FROM credit_tab
```

```
WITH DEFAULT MASK (CREDIT_CARD USING LEFT(CREDIT_CARD,4) || '-XXXX-XXXX-' ||
RIGHT(CREDIT_CARD,4));
```

> → Tip
>
> To mask the same data in different views, you can create a reusable SQL function to generate the mask expression. This is shown in *Example: Masking Data in View Hierarchy with Structured Privilege Check*.

3. User `data_owner` grants user `end_user` access to the view:

```
GRANT SELECT ON credit_view TO end_user;
```

Results

When user `end_user` selects from the credit card view, the data is masked as follows:

```
 ;NAME ;CREDIT_CARD
1;John ;1111-XXXX-XXXX-1111
2;James;2222-XXXX-XXXX-2222
```

9.5 Example: Masking Data in Object Hierarchy with SESSION USER Mask Mode

This example shows the behavior of a user-defined invoker function built on top of a view with a masked column.

Procedure

1. User `data_owner1` creates the credit card table (`credit_tab`), masking the credit card number column:

```
CREATE TABLE credit_tab (Name varchar(20), CREDIT_CARD varchar(19));
INSERT INTO credit_tab values ('John', '1111-1111-1111-1111');
INSERT INTO credit_tab values ('James', '2222-2222-2222-2222');
ALTER TABLE CREDIT_TAB ADD SESSION USER MASK (CREDIT_CARD USING
LEFT(CREDIT_CARD,4) || '-XXXX-XXXX-' || RIGHT(CREDIT_CARD,4));
```

> → Tip
>
> To mask the same data in different views, you can create a reusable SQL function to generate the mask expression. This is shown in *Example: Masking Data in View Hierarchy with Structured Privilege Check*.

2. User `data_owner1` grants select authorization on `credit_tab` to user `data_owner2`:

```
GRANT SELECT ON credit_tab TO data_owner2 WITH GRANT OPTION;
```

3. User `data_owner2` creates a table function on `credit_tab` with definer security mode and a SQL view on top of the function:

```
CREATE FUNCTION credit_function
RETURNS TABLE(Name varchar(20), CREDIT_CARD varchar(19))
LANGUAGE SQLSCRIPT SQL SECURITY DEFINER
AS
BEGIN
    RETURN SELECT * FROM data_owner1.credit_tab;
END;
CREATE VIEW credit_view AS SELECT * FROM credit_function();
```

4. User `data_owner2` grants select authorization to user `end_user`:

```
GRANT SELECT ON credit_view TO end_user
```

Results

When user `end_user` selects from the `credit_view`, the data is masked according the mask expression defined in `data_owner1.credit_tab`.

User `end_user` can see full (unmasked) data only after `data_owner1` grants the UNMASKED privilege on the table `credit_tab`.

Masking Explanation with SYS.EFFECTIVE_MASK_EXPRESSIONS

The effectively enforced mask expressions can be analyzed with the help of the system view SYS.EFFECTIVE_MASK_EXPRESSION. Using a user with the system privilege CATALOG READ, the system view can be queried for our scenario as follows:

```
SELECT
    EFFECTIVE_USER_NAME,
    DEPENDENT_OBJECT_NAME,
    UNDERLYING_OBJECT_NAME,
    UNDERLYING_COLUMN_NAME,
    MASK_EXPRESSION,
    MASK_EXPRESSION_STATUS
FROM SYS.EFFECTIVE_MASK_EXPRESSIONS
WHERE USER_NAME = 'END_USER'
    AND ROOT_SCHEMA_NAME = 'DATA_OWNER2'
    AND ROOT_OBJECT_NAME = 'CREDIT_VIEW'
    AND ROOT_COLUMN_NAME = 'CREDIT_CARD';
```

In first case above, the user `end_user` does not have UNMASKED privilege on the table `credit_tab`, so we see the following result:

Output Code

```
 ;EFFECTIVE_USER_NAME    ;DEPENDENT_OBJECT_NAME    ;UNDERLYING_OBJECT_NAME
    ;UNDERLYING_COLUMN_NAME        ;MASK_EXPRESSION                        ;MA
SK_EXPRESSION_STATUS
1;END_USER        ;            ;CREDIT_VIEW           ;CREDIT_CARD
 ;                           ;
2;DATA_OWNER2        ;CREDIT_VIEW        ;CREDIT_FUNCTION        ;
    ;                                ;
```

```
3;END_USER        ;CREDIT_FUNCTION    ;CREDIT_TAB            ;CREDIT_CARD
       ;LEFT(CREDIT_CARD,4) || '-XXXX-XXXX-' ||
RIGHT(CREDIT_CARD,4)    ;SESSION USER APPLIED
```

In particular, due to the setting of SESSION USER mask mode in `credit_tab`, the effective user is end_user. This is the user who needs the UNMASKED privilege on the table in order to see full data, otherwise the masked expression shown in the column MASK_EXPRESSION is enforced as indicated by the column MASK_EXPRESSION_STATUS.

If end_user is granted the UNMASKED privilege on the credit_tab, we see the following result. The MASK_EXPRESSION_STATUS column indicates that the mask expression is not being enforced.

Output Code

```
 ;EFFECTIVE_USER_NAME    ;DEPENDENT_OBJECT_NAME    ;UNDERLYING_OBJECT_NAME
    ;UNDERLYING_COLUMN_NAME        ;MASK_EXPRESSION                        ;MA
SK_EXPRESSION_STATUS
1;END_USER        ;            ;CREDIT_VIEW           ;CREDIT_CARD
 ;                            ;
2;DATA_OWNER2        ;CREDIT_VIEW       ;CREDIT_FUNCTION        ;
    ;                          ;
3;END_USER        ;CREDIT_FUNCTION    ;CREDIT_TAB            ;CREDIT_CARD
       ;LEFT(CREDIT_CARD,4) || '-XXXX-XXXX-' ||
RIGHT(CREDIT_CARD,4)    ;SESSION USER NOT APPLIED
```

9.6 Example: Masking Data in a View with Structured Privilege Check

This example shows the behavior of a view that contains a masked column and that is also secured with an analytic privilege.

Prerequisites

User `data_owner` has the system privilege STRUCTUREDPRIVILEGE ADMIN.

Procedure

1. User `data_owner` creates a table (`credit_tab`) containing credit card data:

```
CREATE TABLE credit_tab (Name varchar(20), CREDIT_CARD varchar(19));
INSERT INTO credit_tab values ('John', '1111-1111-1111-1111');
INSERT INTO credit_tab values ('James', '2222-2222-2222-2222');
```

2. User `data_owner` creates a view (`credit_view`) on the credit card table, masking the credit card number column and further securing the view with a structured privilege check:

```
CREATE VIEW credit_view AS SELECT * FROM credit_tab
WITH DEFAULT MASK (CREDIT_CARD USING LEFT(CREDIT_CARD,4) || '-XXXX-XXXX-' ||
RIGHT(CREDIT_CARD,4))
    STRUCTURED PRIVILEGE CHECK;
```

> → Tip
>
> To mask the same data in different views, you can create a reusable SQL function to generate the mask expression. This is shown in *Example: Masking Data in View Hierarchy with Structured Privilege Check*.

3. User `data_owner` grants select authorization to user `end_user`:

```
GRANT SELECT ON credit_view TO end_user;
```

4. User `mask_owner` creates an analytic privilege for `credit_view` that allows access to the row containing the credit number 1111-1111-1111-1111:

```
CREATE STRUCTURED PRIVILEGE credit_ap for SELECT ON data_owner.credit_view
WHERE CREDIT_CARD = '1111-1111-1111-1111';
```

5. User `mask_owner` grants the analytic privilege `credit_ap` to user `end_user`:

```
GRANT STRUCTURED PRIVILEGE credit_ap TO end_user;
```

Results

When user `end_user` selects from the credit card view, the data is first filtered in line with the analytic privilege and then masked:

```
 ;NAME ;CREDIT_CARD
1;John ;1111-XXXX-XXXX-1111
```

9.7 Example: Masking Data in View Hierarchy with Structured Privilege Check

This example shows the behavior of a view secured with an analytic privilege built on top of a view with a masked column.

Prerequisites

User `mask_owner` has the system privilege STRUCTUREDPRIVILEGE ADMIN.

Procedure

1. User `data_owner1` creates a table (`credit_tab`) containing credit card data:

```
CREATE TABLE credit_tab (Name varchar(20), CREDIT_CARD varchar(19));
INSERT INTO credit_tab values ('John', '1111-1111-1111-1111');
INSERT INTO credit_tab values ('James', '2222-2222-2222-2222');
```

2. User `mask_owner` defines a mask as a SQL function and grants user `data_owner1` authorization to execute:

```
CREATE FUNCTION credit_mask(input varchar(19)) RETURNS output VARCHAR(19)
LANGUAGE SQLSCRIPT
AS
BEGIN
    output = LEFT(:input,4) || '-XXXX-XXXX-' || RIGHT(:input,4);
END;
GRANT EXECUTE ON credit_mask TO data_owner1;
```

> ### Note
>
> For illustration purposes, this example uses an SQL function to generate the mask expression. For performance reasons, define simple mask expression directly as part of the table or view definition if possible.

3. User `data_owner1` creates a view on the credit card table (`credit_view_base`), masking the credit card number column using the SQL function:

```
CREATE VIEW credit_view_base AS SELECT * FROM credit_tab
WITH MASK (CREDIT_CARD USING mask_owner.credit_mask(credit_card));
```

4. User `data_owner1` grants select authorization on `credit_view_base` to user `data_owner2`:

```
GRANT SELECT ON credit_view_base TO data_owner2;
```

5. User `data_owner2` creates a top-level view on the base view, securing it with a structured privilege check:

```
CREATE VIEW credit_view_top AS SELECT * from data_owner1.credit_view_base
WITH STRUCTURED PRIVILEGE CHECK;
```

> ### Note
>
> User `data_owner2` does not have the `UNMASKED` privilege on `credit_view_base`.

6. User `data_owner2` grants select authorization to user `end_user`:

```
GRANT SELECT ON credit_view_top TO end_user;
```

7. User `mask_owner` creates an analytic privilege for the top-level view `credit_view_top` that allows access to the row containing the credit number 1111-1111-1111-1111:

```
CREATE STRUCTURED PRIVILEGE credit_ap for SELECT ON
data_owner.credit_view_top WHERE CREDIT_CARD = '1111-1111-1111-1111';
```

8. User `mask_owner` grants the analytic privilege `credit_ap` to user `end_user`:

```
GRANT STRUCTURED PRIVILEGE credit_ap TO end_user;
```

Results

When user `end_user` selects from the credit card view, the data is masked first and then filtered in line with the analytic privilege. This returns a empty result set.

10 SAP HANA Data Anonymization

Anonymization methods available in SAP HANA allow you to gain statistically valid insights from your data while protecting the privacy of individuals.

Why Anonymize?

In a data-driven world, a growing amount of business data contains personal or sensitive information. If this data is to be used by applications for statistical analysis, it must be protected to ensure privacy. Trivial modifications to the data like replacing information that directly identifies an individual such as name or social security number (pseudonymization) or simply removing the information is not enough. Re-identification is still possible, for example if additional information is obtained (referred to as a linkage attack).

Unlike masking and pseudonymization, anonymization methods (also called privacy-enhancing methods) provide a more structured approach to modifying data for privacy protection. The quality of such anonymized or privacy-enhanced data is still sufficient for meaningful analysis. Several anonymization methods exist.

SAP HANA supports the methods *k*-anonymity, *l*-diversity, and differential privacy. Which method provides the most appropriate level of privacy depends on your data and the potential attack scenarios and attackers.

Anonymizing Data in SAP HANA

Data anonymization can be applied to SQL views or calculation views, thus enabling analytics on data while still protecting the privacy of individuals.

You define anonymization views using the SQL CREATE VIEW statement and a WITH ANONYMIZATION clause which specifies an anonymization method and includes corresponding parameters for each column in the view to meet the required privacy level. There are three types of parameter:

- Static parameters - used to configure the view as a whole, such as defining the method ('algorithm') to use
- Column parameters - used to apply configuration values to individual columns of the view
- Hierarchy parameters - used for columns which are 'quasi identifiers' to create generalization hierarchies which determine how specific values can be replaced by more generic values.

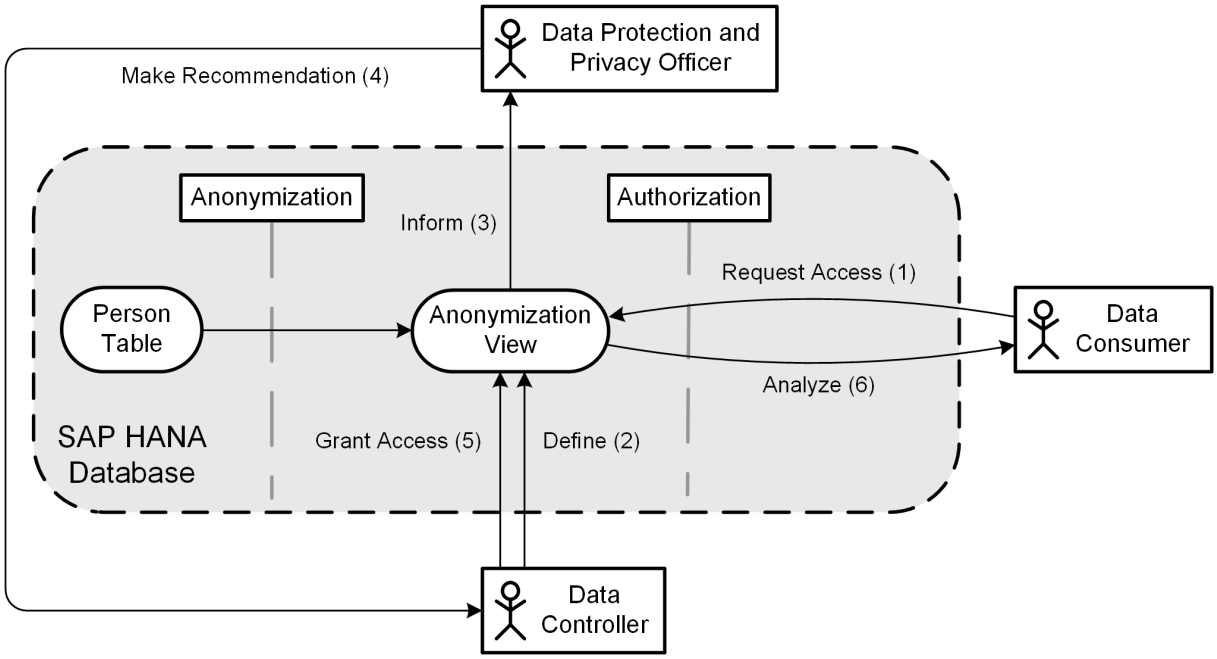
The following example shows an anonymization clause including configuration properties for three columns of a table (ID, GENDER, LOCATION):

```
WITH ANONYMIZATION ( ALGORITHM 'K-ANONYMITY'
PARAMETERS '{"k": 5}'
COLUMN ID PARAMETERS '{"is_sequence": true}'
COLUMN GENDER PARAMETERS '{"is_quasi_identifier":true, "hierarchy":{"embedded":
[["F"], ["M"]]}}'
COLUMN LOCATION PARAMETERS '{"is_quasi_identifier":true, "hierarchy":
{"embedded": [["Paris", "France"], ["Munich", "Germany"], ["Nice",
"France"]]}}');
```

Note that two of the columns are quasi identifiers and include the required hierarchy parameters to define appropriate generalization hierarchies.

Anonymization Workflow

A data controller – that is someone who determines when and how personal data is accessed and processed – defines an SQL view or a calculation view and configures the parameters of the chosen anonymization method to meet the required privacy level. Access to the anonymized view can then be granted to users using standard SAP HANA authorization mechanisms. An overview of this process is shown below:



Information about anonymized SQL views can be obtained by using the system views ANONYMIZATION_VIEWS, ANONYMIZATION_VIEW_COLUMNS, and M_ANONYMIZATION_VIEWS. For more information about anonymizing data using SQL, see the *SAP HANA SQL Reference Guide*.

A list of all anonymized views as well as all calculation views that have one or more anonymization nodes configured is available in the SAP HANA cockpit (for documentation purposes, for example). For more information about anonymizing data using calculation views, see the *SAP HANA Modeling Guide for SAP Web IDE for SAP HANA* .

Related Information

SAP HANA Data Anonymization Guide
SAP HANA Data Masking [page 215]

11 Data Storage Security in SAP HANA

Several mechanisms can be used to protect security-relevant data used by the SAP HANA database from unauthorized access.

11.1 Data Security in the File System

Data in the SAP HANA database (including configuration data) is stored in the file system of the operating system and protected by operating system permissions.

You configure the data path during installation. For more information about the recommended file system layout, see the *SAP HANA Server Installation and Update Guide*. The file permissions of the operating system are strictly configured. Therefore, do not change them after installation.

For more information see SAP Note 1730999 and SAP Note 1731000.

Hardware-Based Encryption with Persistent Memory

Persistent memory, also referred to as non-volatile RAM (NVRAM) or storage class memory, is supported in SAP HANA as a persistent storage type. If you are using persistent memory, you can use hardware-based encryption to protect your data on disk. Please contact your hardware vendor for the latest information about availability of the hardware and support for this feature.

For more information about persistent memory, see the *SAP HANA Administration Guide*.

Related Information

Recommended File System Layout
Persistent Memory
SAP Note 1730999
SAP Note 1731000

11.2 Server-Side Data Encryption Services

SAP HANA features encryption services for encrypting data at rest, as well as an internal encryption service available to applications with data encryption requirements.

Passwords

On the SAP HANA database server, all passwords are stored securely:

- Operating system user passwords are protected by the standard operating system mechanism, `/etc/shadow` file.
- All database user passwords are stored in salted hash form using PBKDF2 (Password-Based Key Derivation Function 2) and, for downward compatibility, secure hash algorithm SHA-256.
  The SAP HANA implementation of PBKDF2 uses the SHA-256 secure hash algorithm and 15,000 iterations.

  > ⓘ Note
  >
  > The hash method SHA-256 can be disabled by setting the parameter `[authentication] password_hash_methods` in the `global.ini` configuration file to **`pbkdf2`**. The default value is **`pbkdf2,sha256`**.

- Credentials required by SAP HANA applications for outbound connections are securely stored in a database-internal credential store. This internal credential store is in turn secured using the internal application encryption service.

Data-at-Rest Encryption

To protect data saved to disk from unauthorized access at operating system level, the SAP HANA database supports data encryption in the persistence layer for the following types of data:

- Data volumes
- Redo log volumes
- Data and log backups

Security-Relevant Application Data

An internal encryption service is used to encrypt sensitive application data. This includes credentials required by SAP HANA for outbound connections, private keys of the SAP HANA server stored in the database, and data in secure stores defined by developers of SAP HANA XS applications (classic or advanced) or other applications (through SQL).

Secure Stores

SAP HANA uses either the instance SSFS (secure store in the file system) or the local secure store (LSS), to protect the root keys used for all data-at-rest encryption services and the internal application encryption service. It uses the system PKI SSFS to protect the system-internal root certificates required for secure internal communication.

Related Information

Data and Log Volume Encryption [page 241]
Backup Encryption [page 243]
Internal Application Encryption Service [page 249]
Server-Side Secure Stores [page 235]
Local Secure Store (LSS) [page 253]
Encryption Key Management [page 239]

11.2.1 Server-Side Secure Stores

SAP HANA employs a secure storage system to safeguard the root keys essential for both data-at-rest encryption services and the internal application encryption service.

This protection can be configured using either the default Local Secure Store (LSS) (**recommended**) or the Instance Secure Store in the File System (Instance SSFS)(**legacy and deprecated secure store**).

SAP HANA employs the System Secure Store in the File System with Public Key Infrastructure (System PKI SSFS) to safeguard system-internal root certificates for secure internal communication.

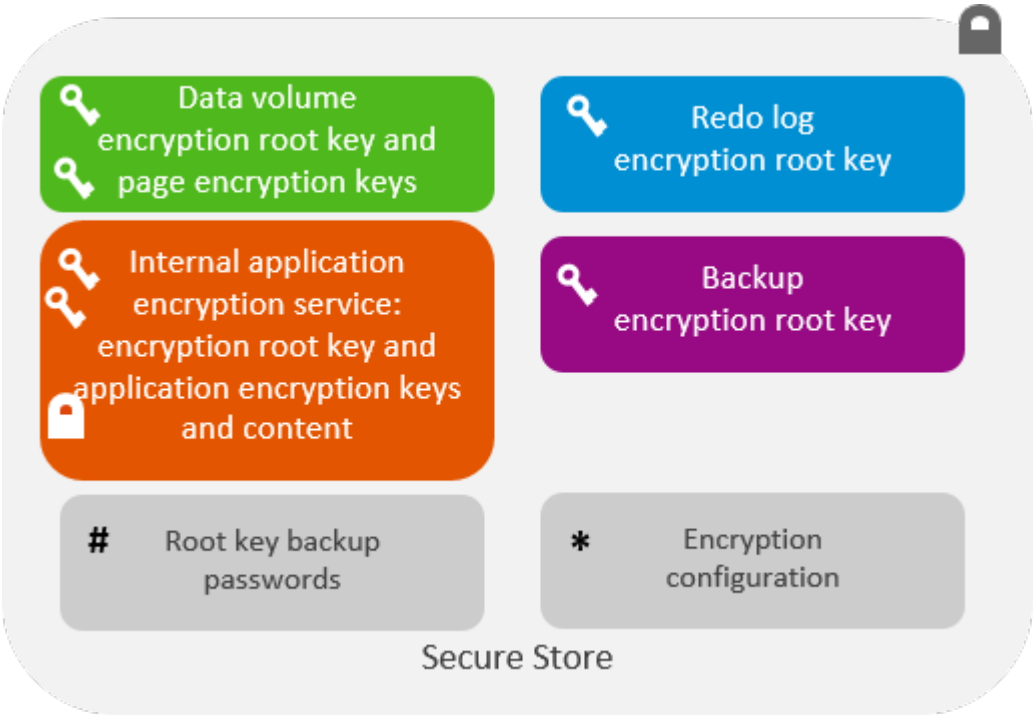
Secure Store for Encryption Root Keys and Internal Application Encryption Service (Instance SSFS and LSS)

SAP HANA uses the secure store to protect the following individually for each database:

- The root keys used for:
  - Data volume encryption (PERSISTENCE)
  - Redo log encryption (LOG)
  - Data and log backup encryption (BACKUP)
  - Internal Application Encryption Service (DPAPI)
- Content of Internal Application Encryption Service (DPAPI)
- Data page encryption keys (PERSISTENCE)
- The password of the root key backup

- The encryption configuration

The root keys protect all encryption keys used in the SAP HANA database from unauthorized access.



Secure store content of one database (system database and tenant)

Secure Store Content of a Database

There are two variations of the encryption root key secure store:

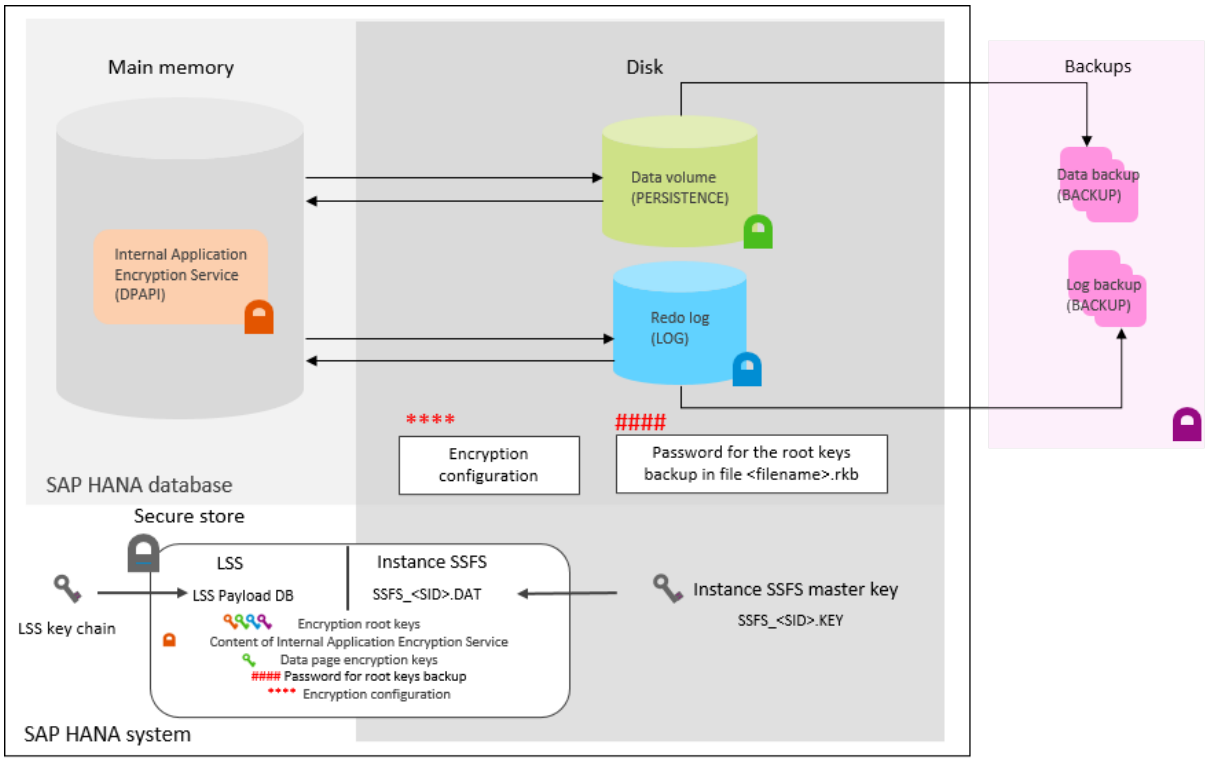
- **Local Secure Store (LSS)(Recommended)**: A separate lightweight utility that runs as a separate service on the SAP HANA server under a different operating system user. It uses tenant-specific files. It is the default secure store. For more information about the LSS, see *Local Secure Store*. For additional details, see SAP Note 3048191.
- **Instance Secure Store in the File System (Instance SSFS)(legacy and deprecated key store)**: A single file in the local file system that hosts the encryption keys for the system database and all tenants. For more information, refer to SAP Note 3544974.

> **Note**
>
> To prevent data encrypted in the SAP HANA database from becoming inaccessible, the content of the Instance SSFS and key information in the database must remain consistent. If the SSFS becomes inconsistent, the database detects this and issues alert 57. In this case, it is recommended to contact SAP Support to resolve the issue (SAP Note 2097613).

Encryption Services and Keys

The following diagram provides an overview of which data in SAP HANA can be encrypted using a dedicated encryption service, and how all associated encryption root keys are stored in the configured secure store.

> **Note**
>
> The following diagram shows only one database. However, a system always has a system database and any number of tenant databases. Every database in the system has its own encryption root keys for each of the available encryption services. The root keys of all databases are stored in the configured secure store.



Encryption Services and Keys of one database

The contents of the Instance SSFS in file SSFS_.DAT are protected by an individual master key in file SSFS_.KEY. Both files belong together.

Master Key Change

Unique master keys are generated for the instance secure store during installation or update.

If you received your system pre-installed from a hardware or hosting partner, it is recommended to change them immediately after handover to ensure they are not known outside of your organization. You can also change the master keys at any time later.

> **Note**
>
> The default path of the key file of the Instance SSFS is:
>
> /usr/sap/<sid>/SYS/global/hdb/security/ssfs/
>
> If you change the default path, you must reconfigure it after a system rename.

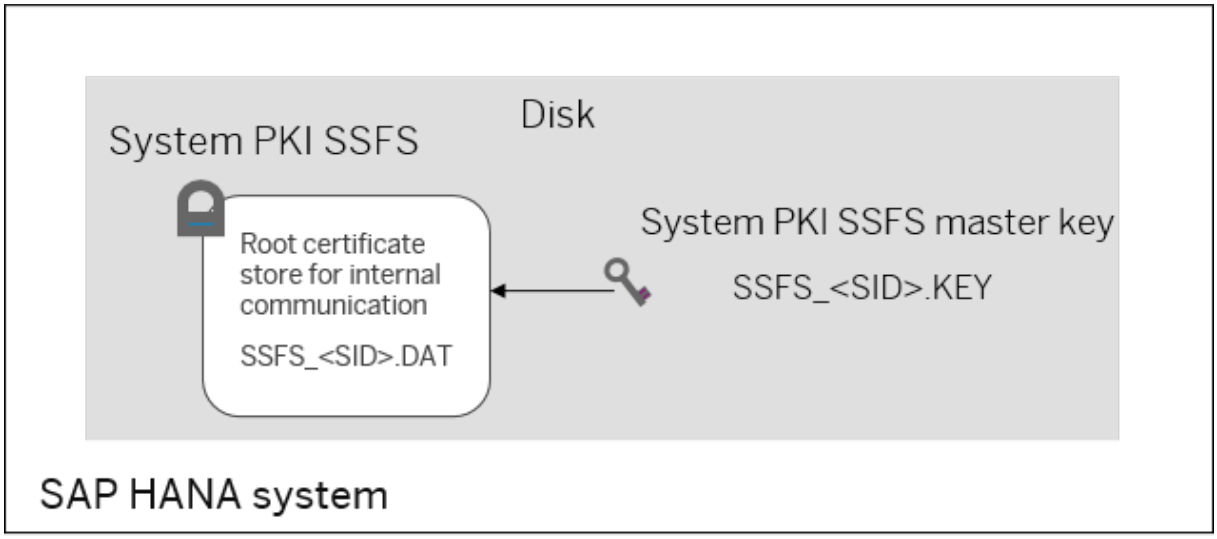
System Secure Store in the File System with Public Key Infrastructure (System PKI SSFS)

The System PKI SSFS protects the X.509 certificate infrastructure used to secure internal SSL/TLS-based communication between hosts in a multiple-host system or between processes of the individual databases in a system.

The master key of the System PKI SSFS encrypts all system-internal root certificates required for secure internal communication.

The contents of System PKI SSFS in file SSFS_.DAT are protected by an individual master key in file SSFS_.KEY. Both files belong together.

> **Note**
>
> The default path of the key file of the System PKI SSFS is:
>
> /usr/sap/<sid>/SYS/global/security/rsecssfs/data/
>
> /usr/sap/<sid>/SYS/global/security/rsecssfs/key/
>
> If you change the default path, you must reconfigure it after a system rename.



Master Key Change

Unique master keys are generated for the System PKI SSFS during installation or update. If you received your system pre-installed from a hardware or hosting partner, it is recommended to change them immediately after handover to ensure they are not known outside of your organization. You can also change the master keys at any time later.

Related Information

Local Secure Store (LSS) [page 253]
Secure Internal Communication [page 64]
Change the SSFS Master Keys
Internal Application Encryption Service [page 249]
SAP Note 3048191
SAP Note 3544974
SAP Note 2097613

11.2.2 Encryption Key Management

SAP HANA generates unique master keys and encryption root keys on installation and database creation. However, if you received SAP HANA from a hardware or hosting vendor, you might want to change them to ensure they are not known outside your organization.

We recommend that you change master keys and encryption root keys immediately after handover, but you can also change them any time later.

The following master keys and encryption root keys exist and can be changed:

- Instance SSFS master key (if the instance SSFS is used as the secure store)
- System PKI SSFS master key
- Root keys for data volume encryption, redo log encryption, backup encryption, and the application encryption service

> **Note**
>
> Unlike the root keys for the internal application encryption service and data volume encryption, the root keys for redo log encryption and backup encryption do not encrypt any other encryption keys.

Reinstalling your system will change all master and root keys. You can also change keys manually and individually. For more information about the overall configuration workflow for server-side encryption, see the *Encryption Configuration* in the *SAP HANA Administration Guide*.

Changing the SSFS Master Keys

You change the master keys of the instance SSFS, if used, and the system PKI SSFS using the command line tool `rsecssfx`. This requires operating system access (`<sid>`adm user).

> ### Note
>
> If the instance SSFS is used in a system-replication configuration, you change the instance SSFS master key on the primary system. To trigger replication of the new key to the secondary system, you must subsequently restart the secondary system. In multi-tier system replication scenarios involving three systems, restart the tier-2 secondary system first, then the tier-3 secondary system. If a secondary system takes over from its replication source before the new master key has been replicated, all systems registered will use the old key from the former secondary system instead.

Encryption Root Keys

SAP HANA uses a different set of root keys for the system database and for each tenant database. Each database always has an active version of each of the root keys. The encryption root keys are stored in the configured secure store (local secure store or instance SSFS).

Encryption root keys are symmetric keys which are generated randomly during installation. They can be rotated using SQL statements.

Changing the Encryption Root Keys

The key change process is as follows. Note that it is important to always back up all root keys:

1. Generate new root keys.
2. Back up all root keys.
3. Activate new root keys.
4. Back up all root keys.

> ### Note
>
> It is important that you always adhere to this process for all key types. The first backup ensures that the new key already has a backup when it is activated so that there is no time frame where the key is used but not backed up. The second backup is when the new key is in the active state, which is not quite so important but is more correct. Establishing this process for all root key changes in your organization will ensure that you always have an up-to-date backup of your root keys available for recovery.

> ### Note
>
> In a system-replication configuration, change root keys in the primary system only. New keys will be propagated to all secondary systems. The secondary systems must be running and replicating.

Related Information

Encryption Configuration
Change the SSFS Master Keys
Changing Encryption Root Keys
SAP Note 2097613

11.2.3 Data and Log Volume Encryption

To protect data saved to disk from unauthorized access at operating system level, the SAP HANA database supports data encryption in the persistence layer. Data volume encryption protects the data area on disk, while redo log encryption protects the log area on disk.

The SAP HANA database holds the bulk of its data in memory for maximum performance, but it still uses persistent disk storage to provide a fallback in case of failure. During normal operation, data is automatically saved from memory to disk at regular savepoints. Additionally, all data changes are recorded in redo logs. A redo log entry is written to disk with each committed database transaction. If a fault occurs, SAP HANA can be restarted in the same way as any disk-based database, and returns to its last consistent state by replaying the redo log entries since the last savepoint.

Both the data area and redo logs can be encrypted:

Encryption Type	Description
Data volume encryption	All pages that reside in the data area on disk are encrypted using a AES-256-bit algorithm. Pages are transparently decrypted as part of the load process into memory. Therefore, when pages reside in memory they are not encrypted and there is no performance overhead for in-memory page accesses. When changes to data are persisted to disk, the relevant pages are automatically encrypted as part of the write operation. Page keys are valid for a certain range of savepoints and can be changed by executing SQL statements. After data volume encryption has been enabled, an initial page key is automatically generated. Page keys are never readable in plain text, but are encrypted themselves using a dedicated data volume encryption root key.
Redo log encryption	Log entries are encrypted using a AES-256-bit algorithm before they are written to disk. Log entries are encrypted and decrypted using a 256-bit long root key.

During start-up, administrator interaction is not required. The data volume encryption and redo log root keys are stored using the functionality of the configured secure storage (LSS or instance SSFS) and are automatically retrieved from there.

> ⓘ Note
>
> SAP HANA uses the configured secure store to protect the encryption root keys that are used to protect encryption keys or persistent data in the SAP HANA system from unauthorized access. All root keys are encrypted using the instance SSFS master key or the LSS internal key chain.

→ Recommendation

Although SAP HANA provides you with the flexibility to encrypt data volumes, redo logs, and backups independently of each other, if you require full protection in the persistence layer, we recommend that you enable all services.

The use of encryption does not increase data size.

ⓘ Note

SAP HANA does not encrypt its database traces. For security reasons, we recommend that you do not run the database with extended tracing for more than short-term analysis since tracing might expose security-relevant data that would be encrypted in the persistence layer but not in the trace. Therefore, you should not keep such trace files on disk after the analysis task is completed.

Enabling Data and Log Volume Encryption in Tenant Databases

Enabling Data and Log Encryption in a New Database

Ideally, encryption is enabled in the database immediately on creation. You can ensure that this is the case for new tenant databases with the following parameters in the `database_initial_encryption` section of the `global.ini` configuration file in the system database:

- `persistence_encryption`
- `log_encryption`

To ensure that any subsequently created tenant databases automatically have encryption enabled, set the value of these parameters to **on**. If a particular tenant database does not require encryption, it can be switched off. By default these parameters are set to **`off`**.

Who can enable or disable data and log volume encryption (as well as backup encryption) in a newly created tenant database depends on how the `encryption_config_control` parameter in the `database_initial_encryption` section is configured. For more information about this parameter and changing encryption control, see *Encryption Configuration Control*.

ⓘ Note

In a system-replication configuration, enable (or disable) encryption in the primary system only. The setting will be propagated to all secondary systems. The secondary systems must be running and replicating.

Enabling Data and Log Encryption in a Running Database

It is possible to enable data volume encryption in a tenant database that already exists and is already in operation. However, only the pages in use within the data volumes are initially encrypted. Pages in the data volumes that are not in use may still contain old content and will only be overwritten and encrypted over time. Furthermore, if you enable redo log encryption after the database has been in operation, only future redo log entries are encrypted. Existing redo log entries remain unencrypted until they are overwritten. Log segment files are cyclically overwritten once they have been backed up. This means that data in data and log volumes will only be fully encrypted after some delay.

Although encryption can be switched on at any point in time, unencrypted data can remain on disk. If this is not wanted, you will need to install a new database on a fresh hard drive, activate encryption, import a backup, and low-level erase the old disks.

Related Information

Encryption Configuration
Encryption Configuration Control [page 247]
Backup Encryption [page 243]
Enable Encryption
ALTER DATABASE Statement (Tenant Database Management)
ALTER SYSTEM PERSISTENCE ENCRYPTION Statement (System Management)
ALTER SYSTEM LOG ENCRYPTION Statement (System Management)
ALTER SYSTEM ENCRYPTION CONFIGURATION Statement (System Management)

11.2.4 Backup Encryption

Data backups and log backups can be encrypted using onboard SAP HANA capabilities.

Backup encryption safeguards the privacy of the SAP HANA backup data by preventing unauthorized parties from reading the content of backups.

When backup encryption is enabled, the backup data is transferred encrypted to the backup location, both for file-based backups and for backups created using third-party backup tools.

SAP HANA backups are encrypted using AES 256-bit encryption.

How the Available Backup Types Are Encrypted

- Data and log backups
  If backup encryption is activated, SAP HANA encrypts the data that it writes to the backup location with the backup root key. To recover the database from the data and log backups, you must first import backed-up root keys to initialize the secure store. When the instance SSFS (secure store in the file system) is used as the secure store, the set of keys from the root key backup file replaces the set of keys in the instance SSFS. When the local secure store (LSS) is used, the key import ensures that only the missing keys are added to the secure store; no keys are deleted. If the LSS is being used with an external key management system, backed-up root keys can also be recovered using a second access key pair in an emergency situation. This is only necessary if the private key stored in the KMS has been lost, damaged, or becomes unusable for some reason.
- Data snapshots
  The SAP HANA data and log volumes are copied with operating-system means as is. If SAP HANA has encrypted the data and log volumes because data and log volume encryption is active, then the copies are encrypted as well, with the SAP HANA root keys for data and log encryption respectively. If SAP HANA is not configured to encrypt data and log volumes, then the copies are also not encrypted.

For more information, see *Enable Encryption* in the *SAP HANA Administration Guide (Encryption)* and *Encryption of Data Snapshots* in the *SAP HANA Administration Guide (SAP HANA Database Backup and Recovery)*.

> **Note**
>
> The backup catalog is not encrypted.

Resource Consumption With Backup Encryption

Typically, backup encryption does not impact the performance of backup and recovery, since the backup and recovery operations, such as encryption, checksum calculation, and I/O, run in parallel. However, with backup encryption, higher CPU usage is to be expected. Depending on the sizing of your SAP HANA system, you may notice a negative impact on overall system performance.

To create encrypted backups with optimal performance, SAP HANA consumes up to three times as much main memory and CPU than is needed to create unencrypted backups. Resource consumption is compared below:

Backup Type	Buffer Allocated
Unencrypted backups	The memory consumed by backup is determined by the configured I/O buffer size to read/write backups. For an unencrypted data backup, this amount of memory is allocated twice, one buffer for reading data and one buffer for writing data. This is the case when multistreaming is not used. To configure I/O buffer size, you can use the parameter `data_backup_buffer_size` in the `backup` section of the `global.ini` parameter file. For more information, see *Change the I/O Buffer Size*.
Multistreaming backups	The configured I/O buffer size is also allocated to each channel. With multistreamed backups, SAP HANA allocates one additional I/O buffer for each additional channel. For this reason, the memory consumption is calculated as `data_backup_buffer_size` * (1 + `parallel_data_backup_backint_channels`). To configure multistreaming backups, you can use the parameter `parallel_data_backup_backint_channels` in the `backup` section of the `global.ini` parameter file. For more information, see *Configure Multistreaming with Third-Party Backup Tools*.

Backup Type	Buffer Allocated
Encrypted backups	The configured I/O buffer size is also allocated to two additional I/O buffers for each channel for each persistent service. For this reason, the memory consumption is calculated as `data_backup_buffer_size` * (1 + 3 * `parallel_data_backup_backint_channels`) **Note** The size of encrypted SAP HANA backups is the same as unencrypted backups (except for the checksum).

Considerations for Backup Encryption

Depending on how much main memory and CPU are available on an SAP HANA system, consider the following options:

- Configure SAP HANA to run backup (and recovery) with sub-optimal performance.
  You can configure SAP HANA to perform backup and recovery operations with encryption enabled, but with the same resource consumption as without encryption. As a consequence, backup and recovery operations will typically take longer to complete.
  Set the parameter `enable_parallel_backup_encryption` from **`true`** (default) to **`false`**.
- Install additional CPUs in your SAP HANA system.
- Reduce the backup I/O buffer size allocated by SAP HANA.
  If the I/O buffer size is reduced, backup will typically have a lower throughput, and, as a consequence,will take longer to complete.
  If no additional physical main memory is available, the backup I/O buffer size for a database can be reduced to approximately one third of the backup I/O buffer size configured for unencrypted backups.
  For more information, see *Change the I/O Buffer Size* in the *SAP HANA Administration Guide (SAP HANA Database Backup and Recovery)*.
- If you are working with a third-party backup tool, it is not recommended to use data deduplication with encrypted backups, as this will increase backup times with no actual benefit.

Note

The block-level integrity of encrypted backups can still be checked without access to the backup encryption root key. For more information, see *Checking Whether a Recovery is Possible*.

For more information about working with encryption root keys, see *Encryption Key Management* in the *SAP HANA Security Guide* and *Changing Encryption Root Keys* in the *SAP HANA Administration Guide (Encryption)*.

Enabling Backup Encryption in Tenant Databases

You can enable backup encryption at any time. By default, backup encryption is not enabled in a new tenant database. This is controlled by the `backup_encryption` parameter in the `database_initial_encryption` section of the `global.ini` configuration file of the system database.

To ensure that backup encryption is automatically enabled when new tenant databases are created, set the value of this parameter to **on**. If a particular tenant database subsequently does not require backup encryption, it can be switched off. By default, this parameter is set to **off**.

Who can enable or disable backup encryption (as well as data and log volume encryption) in a newly created tenant database depends on how the `encryption_config_control` parameter in the `database_initial_encryption` section is configured. For more information about this parameter and changing encryption control, see *Encryption Configuration Control*.

> **Note**
>
> In a system-replication configuration, enable (or disable) encryption in the primary system only. The setting will be propagated to all secondary systems. The secondary systems must be running and replicating.

Related Information

SAP HANA Security Guide

Encryption Configuration Control [page 247]

Encryption Key Management [page 239]

Root Key Backup [page 252]

Data and Log Volume Encryption [page 241]

SAP HANA Administration Guide (Encryption)

Changing Encryption Root Keys

Enable Encryption

SAP HANA Administration Guide (SAP HANA Database Backup and Recovery)

Encryption of Data Snapshots

Creating Backups

Change the I/O Buffer Size

Checking Whether a Recovery is Possible

SQL Reference Guide

ALTER SYSTEM BACKUP ENCRYPTION Statement (System Management)

M_ENCRYPTION_OVERVIEW System View

11.2.5 Encryption Configuration Control

You can enable or disable the encryption of data and log volumes, and data and log backups in a new SAP HANA database or in an existing operational database. For a tenant database, you need to know whether encryption configuration is controlled by the tenant database or the system database.

Ownership of Encryption Control

By default, encryption configuration is controlled by the tenant database, but the control can be switched to the system database, or the system database can switch control back to the tenant database.

To see which database is controlling encryption configuration for a tenant database, you can query the system view SYS.M_ENCRYPTION_OVERVIEW. From the system database, you can query the system view SYS_DATABASES.M_ENCRYPTION_OVERVIEW.

Encryption Control in New Tenant Databases

When a new tenant database is created, the `encryption_config_control` parameter in the `database_initial_encryption` section of the `global.ini` configuration file in the system database determines whether encryption configuration is controlled by the tenant database or the system database. You can use this parameter to configure encryption control for new tenant databases:

- If the value of this parameter is `local_database` (default), then only the tenant database administrator can enable or disable encryption from the tenant database.
- If the value is `system_database`, then only the system database administrator can enable or disable encryption from the system database.

Switching Encryption Control in Existing Tenant Databases

If the tenant database controls encryption configuration, the tenant database administrator can hand over this control to the system administrator by executing the following ALTER SYSTEM statement:

```
ALTER SYSTEM ENCRYPTION CONFIGURATION CONTROLLED BY SYSTEM DATABASE
```

If the system database controls encryption configuration, the system database administrator can hand it over to the tenant database administrator by executing the following ALTER DATABASE statement:

```
ALTER DATABASE <database_name> ENCRYPTION CONFIGURATION CONTROLLED BY LOCAL
DATABASE
```

For simplicity, the system database administrator can hand over control to all tenants at once by executing the following statement:

```
ALTER SYSTEM ENCRYPTION CONFIGURATION CONTROLLED BY LOCAL DATABASES
```

Controlling Data Volume Re-encryption Performance

When enabling or disabling data volume encryption, the SAP HANA database re-encrypts or decrypts data volumes through background savepoint cycles. You can control and monitor the performance of this process using configuration parameters.

Configuration Parameters

The following parameters control the re-encryption process and can be set online without requiring a system restart:

- `data_volume_encryption_max_log_space_used_before_throttle_percent` (Default: 5%)
  - Controls when the re-encryption job throttles based on log space usage
  - When log space usage exceeds this percentage, the job slows down to prevent log overflow
- - `data_volume_encryption_max_pages_per_savepoint` (Default: varies by system)
    - Controls how many pages are processed per savepoint cycle
    - Higher values speed up re-encryption but may impact system performance

Setting Parameters

Sample Code

```
ALTER SYSTEM ALTER CONFIGURATION ('indexserver.ini', 'SYSTEM')
SET ('persistence',
'data_volume_encryption_max_log_space_used_before_throttle_percent') =
'<value>'
WITH RECONFIGURE;
ALTER SYSTEM ALTER CONFIGURATION ('indexserver.ini', 'SYSTEM')
SET ('persistence', 'data_volume_encryption_max_pages_per_savepoint') =
'<value>'
WITH RECONFIGURE;
```

Monitoring Re-encryption Progress

Monitor the progress using the following statement:

Sample Code

```
SELECT * FROM M_DATA_VOLUME_PAGE_STATISTICS_RESET;
```

Key columns to monitor:

- `ENCRYPTED_SIZE` - Size of encrypted data
- `NOT_ENCRYPTED_SIZE` - Size remaining to be encrypted
- Progress can be calculated as: (ENCRYPTED_SIZE / (ENCRYPTED_SIZE + NOT_ENCRYPTED_SIZE)) * 100

Related Information

ALTER SYSTEM ENCRYPTION CONFIGURATION Statement (System Management)
ALTER DATABASE Statement (Tenant Database Management)

M_ENCRYPTION_OVERVIEW System View

SAP Note 3659679

SAP Note 3555251

11.2.6 Internal Application Encryption Service

The internal encryption service is used internally by applications requiring data encryption.

> **Note**
>
> In the SAP HANA 1.0 documentation, the internal application encryption service was referred to as the internal data encryption service.

The internal application encryption service is used in the following contexts:

- **Secure internal credential store**
  This service stores credentials required by SAP HANA for outbound connections. It is used, for example, when data is retrieved from remote data sources using SAP HANA smart data access. It is also used during HTTP destination calls from SAP HANA XS classic applications.
  For more information, see the section on the secure internal credential store.
- **SAP HANA secure store**
  SAP HANA provides a set of database procedures to store and manage encrypted values individually per database user. A user who can execute these procedures can create and maintain encrypted values in an internal table that is only available to this user.
  SAP HANA XS advanced applications can use the SAP HANA secure store to store sensitive data such as credentials in a secure manner even if data volume encryption has not been enabled. For more information, see *Maintain Values in the SAP HANA Secure Store* in the *SAP HANA Developer Guide for SAP HANA XS Advanced Model*.
- **Secure stores for SAP HANA XS classic applications**
  Application developers can maintain values can define secure stores using the SAP HANA XS classic `$.security.Store` API.
  For more information, see *Using the Server-Side JavaScript APIs* in the *SAP HANA Developer Guide (For SAP HANA Studio)* and *Class:Store* in the *SAP HANA XS JavaScript API Reference*:
- **Private key store**
  This service stores the private keys of the SAP HANA server required for secure client-server communication, if the relevant personal security environment (PSE) is stored in the database. PSEs stored in the database are called certificate collections.
  For more information, see the section on SSL configuration on the SAP HANA server and certificate management in SAP HANA.

Every consumer of the service has its own system-internal application encryption key. These keys are generated as follows:

- The application key for the internal credential store is generated randomly during the first startup.
- A user-specific key for the SAP HANA secure store is created when a user inserts the first value into the SAP HANA secure store.
- Application keys for XS classic secure stores are created at the same time as the XS classic secure store.
- The application key for the private key store is created when the first private key is set for a certificate collection.

Application encryption keys are encrypted with the application encryption service root key.

SAP HANA generates unique root keys on installation or database creation. However, if you received SAP HANA from a hardware or hosting partner, we recommend that you change the root key of the internal application encryption service to ensure it is not known outside your organization. We recommend that you do this immediately after system installation or handover from your hardware or hosting partner.

> **Note**
>
> In a system-replication configuration, change root keys in the primary system only. New keys will be propagated to all secondary systems. The secondary systems must be running and replicating.

The system database and all tenant database have their own individual application encryption service root key.

Related Information

Secure Internal Credential Store [page 250]
Certificate Management in SAP HANA [page 303]
Maintain Values in the SAP HANA Secure Store
Using the Server-Side JavaScript APIs
Class: Store

11.2.6.1 Secure Internal Credential Store

The credentials required by SAP HANA applications for outbound connections can be securely stored in a database-internal credential store. For example, in an SAP HANA smart data access scenario, credentials required to access a remote source are protected using the internal application encryption service.

Credentials can be created and updated by users and privileged administrators using the SQL interface. However, access to credentials in unencrypted form is only available to native SAP HANA applications via an internal API.

Users can create and modify their own credentials. A user with the system privilege CREDENTIAL ADMIN can manage credentials for other users. Credentials are also created implicitly during the creation of remote data sources (SAP HANA smart data access scenario) and HTTP destinations for SAP HANA XS classic applications.

Credentials are created using the SQL statement CREATE CREDENTIAL as follows.

```
CREATE CREDENTIAL FOR USER <user_name> COMPONENT '<application>' PURPOSE
'<credential_purpose>' TYPE '<credential_type>' USING '<credential>'
```

A credential consists of the following elements:

Element	Description
User	The database user for which the credential is stored If no user name is specified, the supplied credential serves as a general entry that can be used by the application if no explicit mapping for a database user is possible. For example, in an SAP HANA smart data access scenario, the connection to a data source may always be established using the same technical user.
Component	The application for which the credential is stored The value of the 'component' element is defined by the application, for example, in an SAP HANA smart data access scenario, the component is 'SAPHANAFEDERATION'.
Purpose	The purpose for which the application is storing this credential The value of the 'purpose' element is defined by the application, for example, in an SAP HANA smart data access scenario, the purpose is the name of the remote data source.
Type	The type of credential being stored, for example PASSWORD The supported values for the this element are specific to the application.
Using	The actual credential, for example user name and password for a credential of type PASSWORD **Note** You can only set credentials using SQL. It not possible to view them. The unencrypted value of the credential is only available to the application via an internal interface.

Sample Code

```
CREATE CREDENTIAL FOR USER TESTUSER COMPONENT 'SAPHANAFEDERATION' PURPOSE
'ASE' TYPE 'PASSWORD' USING 'user="remotedbuser";password="<password>"'
```

Credentials can be changed and dropped using the ALTER CREDENTIAL and DROP CREDENTIAL statements respectively.

The system view CREDENTIALS contains information about stored credentials.

Note

Credentials stored using the credential store remain encrypted even in backups. To allow for the reconstruction of credential data in the case of database recovery, the encryption key used is also part of the backup. To avoid unauthorized access to the encrypted credentials, backups should be stored in a secure location.

Note

The credential store uses the internal application encryption service of the SAP HANA database. The encryption root key of the application encryption service is stored in the configured secure store.

Related Information

Server-Side Data Encryption Services [page 234]
Encryption Key Management [page 239]
CREDENTIALS System View
CREATE CREDENTIAL Statement (Access Control)
CREATE REMOTE SOURCE Statement (Access Control)
Maintaining HTTP Destinations

11.2.7 Root Key Backup

A backup of encryption root keys must be available at an external location to ensure recovery is possible in certain scenarios.

Encryption root keys are stored in the configured secure store and can be retrieved from there automatically during operation. However, a backup of encryption root keys must be stored in a root key backup file at an external location accessible to an SAP HANA administrator. The secure store must always be restored from this backup before a database recovery, unless:

- You have never changed any of the encryption root keys.
- You are performing a recovery into the same database from which the backup was taken, and the database's secure store is intact and contains the latest root key changes.

Root keys **must** be backed up every time they are changed, that is, every time new keys are generated or activated. If you are using the local secure store (LSS) with an external key management system (KMS) and you have deactivated automatic content backups (not recommended), it also recommended to perform a root key backup every time you create and activate a key management configuration.

> ⚠ Caution
>
> A missing backup could result in your database being unrecoverable.

The root key backup is secured using a single password, which must be set before a backup is created. The password is stored securely in the secure store and all subsequent root key backups taken are protected by this password.

For more information about setting and validating the root key backup password, see the *SAP HANA Administration Guide*.

> ⚠ Caution
>
> Store both the root key backup and the password required to read it in a secure location. Losing the backup or the password may result in the database being unrecoverable.

> ⓘ Note
>
> In a system-replication configuration, set the root key backup password in the primary system only. The password will be propagated to all secondary systems. The secondary systems must be running and replicating.

> **Note**
>
> Using the default secure store LSS, you do not need to explicitly backup the secure store, since it is automatically backed up when you do a SAP HANA database backup.

Related Information

BACKUP ENCRYPTION ROOT KEYS Statement (Backup and Recovery)
ALTER SYSTEM VALIDATE ENCRYPTION ROOT KEYS BACKUP PASSWORD Statement (System Management)
Set the Root Key Backup Password
Changing Encryption Root Keys
Using the LSS with an External Key Management System [page 270]
RECOVER ENCRYPTION ROOT KEYS Statement (Backup and Recovery)
Import Backed-Up Root Keys Before Database Recovery

11.2.8 Local Secure Store (LSS)

The local secure store (LSS) is a separate, lightweight utility for storing and securely managing encryption keys, encryption root keys, and other similarly sensitive data, such as security-relevant configuration settings. It helps protect sensitive server-side data from illegitimate or fraudulent usage.

> **Note**
>
> The LSS is the default secure store that can be installed with the SAP HANA server. For more information see, *Installing an SAP HANA System* in the *SAP HANA Server Installation and Update Guide*.

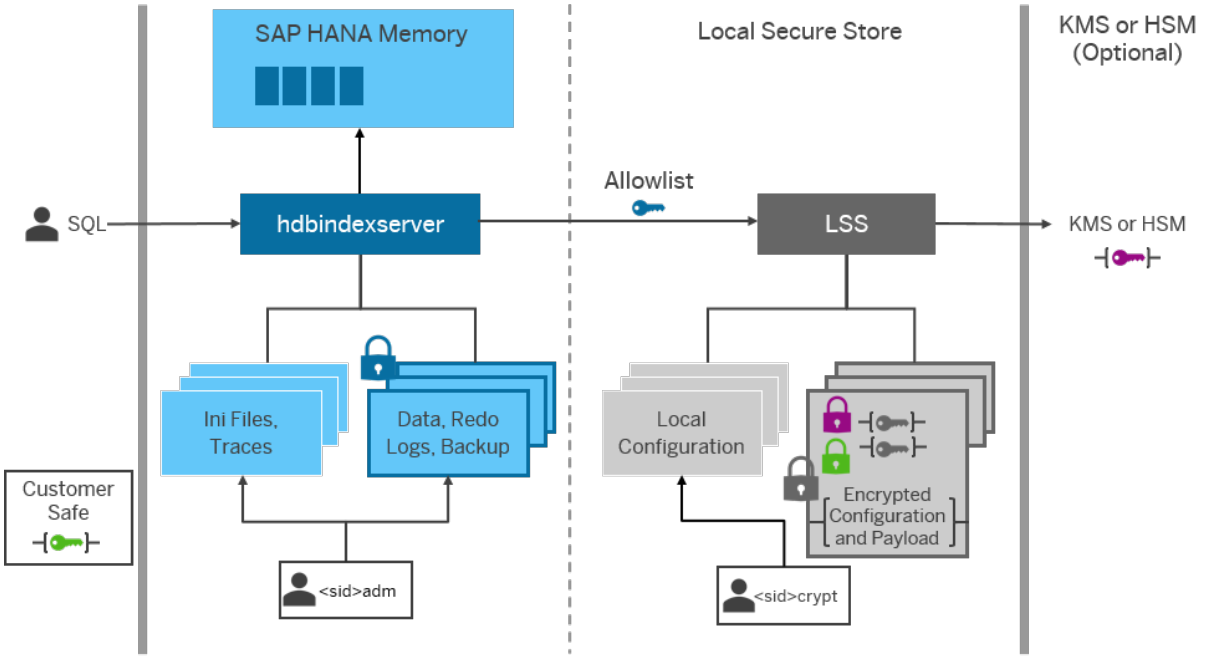
The LSS runs as a separate service on the SAP HANA server under an isolated operating system user, `<sid>crypt`, that owns the storage of the encryption keys and encryption configuration. This allows operating system-level duties to be strictly separated between system administrators and encryption key administrators and, therefore, lowers the level of trust required in a single operating system user.

At the same time, the LSS ensures that LSS clients (like SAP HANA) have unattended access to the encryption root keys that are needed for certain processes, such as automated starts and restarts. This is enabled by a secured communication channel that allows the LSS to verify each LSS client that tries to use the channel. The LSS uses an allowlist to determine the exact identity of each caller that tries to connect to it and rejects untrusted requests.

SAP HANA can use the LSS to protect the following:

- The encryption roots keys used for data volume encryption, redo log encryption, data and log backup encryption, and the application encryption service within the database
- The password of the root key backup
- Encryption configuration information

An overview is shown below:



Overview of the local Secure Store (LSS)

- The LSS payload database (DB) file (shown on the right) contains the SAP HANA encryption root keys and configuration settings of the relevant tenant database (or system database). The contents of this DB file cannot be accessed by `<sid>adm`.
- The `<sid>adm` user only has access to uncritical files and the encrypted customer data.
- The `<sid>crypt` user can read the encryption keys but not the encrypted data.
- SAP HANA processes can access and read the encrypted files. To do so, they request the encryption root keys from the LSS, which verifies the integrity of the SAP HANA processes against an allowlist.
- The payloads stored in the LSS can be secured additionally by using an external key management service (KMS) or hardware security module (HSM).

> **Note**
>
> For system replication setups, the LSS installations on the primary and secondary systems must be completely independent of each other.

> **Note**
>
> Root key actions (rotation, key management configuration changes, encryption settings), which requires the root key backup password to be set, trigger an LSS backup to be written to the log area and log backup.

Related Information

Server-Side Secure Stores [page 235]
Installing an SAP HANA System

Activate the Local Secure Store (LSS)

Using the LSS with an External Key Management System [page 270]

SAP Note 2917358 - How to Configure an SAP HANA System to Use an External Key Management System

SAP Note 2911896 - How to Configure an SAP HANA System to Use the SAP Data Custodian Key Management Service

11.2.8.1 Client Validation

Clients that try to communicate with the local secure store (LSS) through the abstract sockets that the LSS opens and listens to are always checked to ensure that they are on the allowlist before the first request is accepted. Depending on the defined system usage, the LSS reject calls from untrusted callers.

System Usage

System usage is defined during the installation of SAP HANA. It can have one of the following values:

- `production`
- `test`
- `development`
- `custom`

For non-production installations, this allows the security level to be relaxed to facilitate both installation and operation.

The SAP HANA database lifecycle manager (HDBLCM) writes the `systemUsage` setting into the encrypted common database (DB) file (`lsscommon.db`), shared by all LSS tenants.

Trusted Users and Semi-Trusted Users

The LSS has a list of trusted users and a list of semi-trusted users. In SAP HANA, the only trusted user is normally `<sid>crypt` and the only semi-trusted user `<sid>adm`.

If the calling process is started by a trusted user, it is accepted right away. If it is started by a semi-trusted user, it is analyzed further. In all other cases, it is rejected immediately.

Calls by Semi-Trusted Users

If the calling process is started by a semi-trusted user, it is analyzed as follows:

- The LSS determines its executable and its loaded libraries and calculates the hash for each binary that is writable for users other than `root`. It then checks if all hashes are contained on the allowlist. Binaries that are only writable by `root` are trusted and not checked.

- The LSS determines the trust level of each library.
  A library that appears in multiple product versions with different trust levels is assigned the highest trust level found. For example, a certain version of `libhdbbasement.so` could be part of an UNTRUSTED and a FINAL_ASSEMBLY product, so the LSS would assign the trust level FINAL_ASSEMBLY to this library.
- The LSS assigns the client the lowest trust level found. For example, a client would only be assigned the trust level FINAL_ASSEMBLY if its executable and all loaded libraries had the trust level FINAL_ASSEMBLY.
- The LSS checks the determined trust level against the required trust level, which depends on the system usage:
  - A client with the trust level FINAL_ASSEMBLY is accepted in all system usages.
  - A client with the trust level INTERNAL_BUILD is rejected in production installations of the LSS but is accepted in test and development installations.
  - A development system accepts every client (started by semi-trusted users). In development systems, `<sid>adm` is considered a trusted user and is therefore not checked.
  - In all other cases, the call is rejected.

  > **Note**
  >
  > You can list the validation profiles using the lsscfg command `listValidationProfiles`.

You can find details about why a connection was rejected in the the LSS trace (`/usr/sap/<SID>/lss/shared/data/trace`).

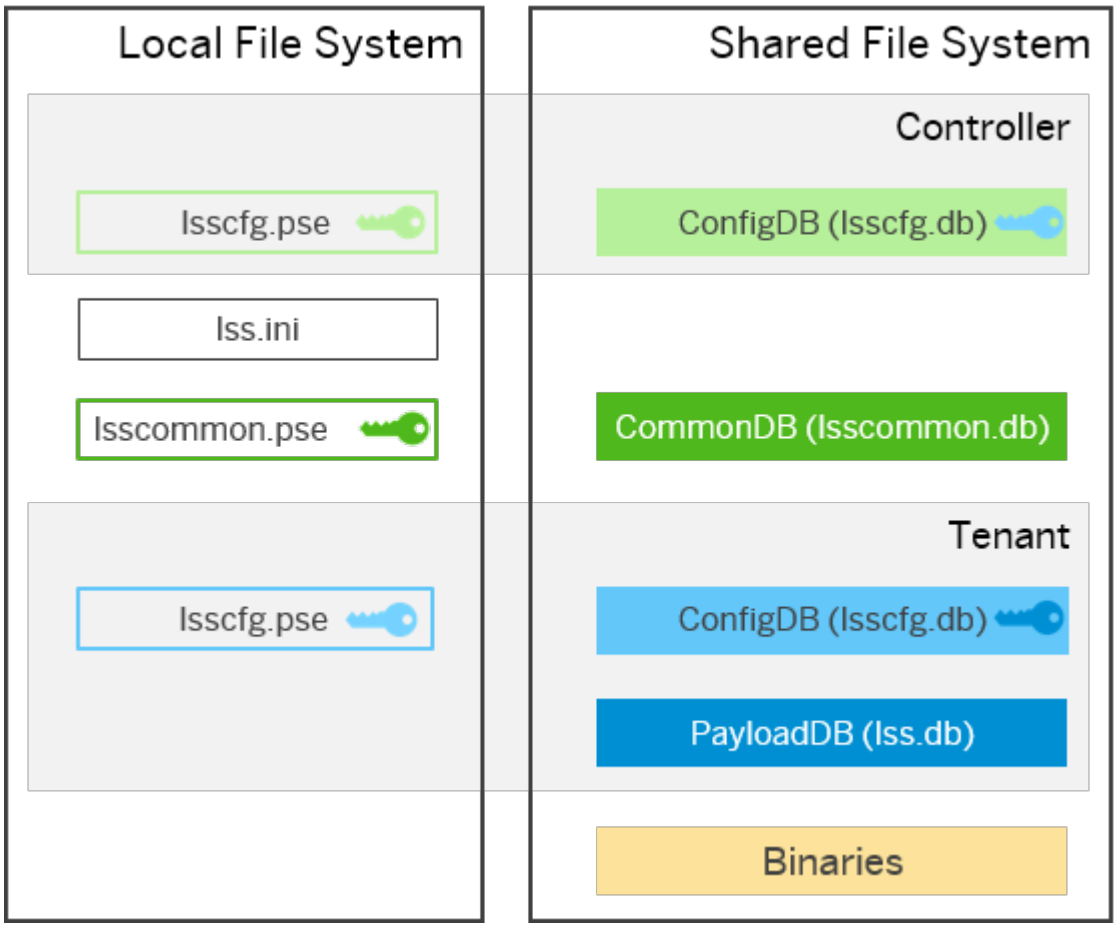
Related Information

LSS Configuration Utility (lsscfg) [page 266]
Configure System Usage Type
system_usage

11.2.8.2 Encryption Keys and Files

The local secure store (LSS) stores the encryption keys (PSE files) required to open the encryted LSS database files in the local file system. These DB files contain the "payload", that is the SAP HANA encryption root keys and configuration information of individual tenant databases. These files are stored in the shared file system.

The following figure shows the files used by the LSS and their storage locations:



Storage Locations of Encryption Keys and Files

Local File System of Host

The local file system contains the `lss.ini` file, as well as the PSE files containing LSS encryption keys:

File Name	File Description
`local/private/lss.ini`	Contains the settings needed to initialize the LSS

File Name	File Description
`local/private/config/lsscfg.pse`	Contains the key to open the Controller's configuration database (DB) file (`lsscfg.db`) **Note** The Controller process manages the topology of the LSS, starts and stops other processes in the LSS, and ensures that incoming client requests are legitimate.
`local/private/common/lsscommon.pse`	Contains the key to open the common DB file (`lsscommon.db`)
`local/private/tenants/<tenant>/config/lsscfg.pse`	Fallback copy of a tenant's PSE file

Tenant-Independent Shared File System

> **Note**
> The system database and tenant databases are both tenants in the LSS context.

The shared file system contains the following encrypted LSS DB files (`.db`):

File Name	File Description
`shared/data/config/lsscfg.db`	The Controller's configuration DB file (`lsscfg.db`) contains a full topology description of SAP HANA, including which tenants run on which host and which tenant has which key. The Controller's configuration DB file also contains the PSE files for each individual tenant. The key to open the Controller's configuration DB file is stored in the `lsscfg.pse` file in the local file system.
`shared/data/common/lsscommon.db`	The common DB file (`lsscommon.db`) contains the binaries and hashes of the installed SAP HANA databases (allowlist) with their system ID and current system usage. This information controls which client programs can access the LSS. The common DB file also contains the auto-backup password, which is required to write automatic content backups and to recover the local secure store (LSS) in the event of a disaster. The key to open the common DB file is stored in the `lsscommon.pse` file in the local file system.

Tenant-Specific Shared File System

The shared file system contains the following encrypted LSS DB files (`.db`) for every tenant.

File Name	File Description
`shared/data/tenants/<tenant>/config/lsscfg.db`	The tenant-specific configuration DB file (`lsscfg.db`) contains the key for the tenant-specific payload DB file, `lss.db`, or the credentials required to access the external key management system (KMS) or hardware security module (HSM). The key to open a tenant-specific configuration DB file is stored in the Controller's configuration DB file.
`shared/data/tenants/<tenant>/payload/lss.db`	The tenant-specific payload DB file (`lss.db`) contains the SAP HANA encryption root keys and SAP HANA encryption configuration of the tenant (tenant database or system database). The key to open the payload DB file is stored in the tenant's configuration DB file (default), or in a KMS or HSM.

> **Note**
>
> The DB files `lsscfg.db`, `lsscommon.db`, and `lss.db` are PKCS#7 files.

Related Information

Directory Structure [page 259]
Encryption Key Chain [page 260]
Using the LSS with an External Key Management System [page 270]
SAP Note 2917358 - How to Configure an SAP HANA System to Use an External Key Management System

11.2.8.2.1 Directory Structure

The local secure store (LSS) stores its data separately in a local and shared directory.

The PSE files in the local directory contain the LSS encryption keys for opening the encrypted database (DB) files stored in the shared directory. A tenant's payload DB file (`lss.db`) does not have a corresponding PSE file because the key required to open the payload DB file is stored in the tenant's configuration DB file (`lsscfg.db`).

> **Note**
>
> The system database and tenant databases are both tenants in the LSS context.

In the following example, the system ID is ABC and the tenant is tenant database `myTenant1`:

> **Sample Code**

```
/usr/sap/ABC/lss
├── [lrwxrwxrwx   19]  exe -> /lss/shared/ABC/exe
├── [drwxr-xr-x 4.0K]  home
│   └── [drwxr-xr-x 4.0K]  bin
```

```
├── [drwxr-x--- 4.0K]  local
│   └── [drwx------ 4.0K]  private
│       ├── [drwx------ 4.0K]  common
│       │   └── [-rw------- 1.7K]  lsscommon.pse
│       ├── [drwx------ 4.0K]  config
│       │   └── [-rw------- 1.7K]  lsscfg.pse
│       ├── [-rw-------  149]  lss.ini
│       └── [drwx------ 4.0K]  tenants
│           └── [drwx------ 4.0K]  myTenant1
│               └── [drwx------ 4.0K]  config
│                   └── [-rw------- 1.8K]  lsscfg.pse
└── [lrwxrwxrwx   15]  shared -> /lss/shared/ABC
    ├── [drwx------ 4.0K]  backup
    └── [lrwxrwxrwx   22]  data
        ├── [drwx------   80]  common
        │   ├── [-rw------- 6.1K]  lsscommon.db
        │   └── [-rw-------   16]  lsscommon.db.lock
        ├── [drwx------   80]  config
        │   ├── [-rw------- 4.4K]  lsscfg.db
        │   └── [-rw-------   16]  lsscfg.db.lock
        ├── [drwx------   40]  export
        ├── [drwx------   60]  tenants
        │   └── [drwx------   80]  myTenant1
        │       ├── [drwx------   80]  config
        │       │   ├── [-rw------- 5.9K]  lsscfg.db
        │       │   └── [-rw-------   16]  lsscfg.db.lock
        │       └── [drwx------   80]  payload
        │           ├── [-rw------- 1.9K]  lss.db
        │           └── [-rw-------   16]  lss.db.lock
        └── [drwx------  100]  trace
            ├── [-rw-r-----  67K]  lss_myHost1.000.trc
            ├── [drwx------  100]  myTenant1
            │   ├── [-rw-r-----  29K]  lss_myHost1.myTenant1.000.trc
            │   ├── [-rw-------    0]  stderr
            │   └── [-rw------- 3.4K]  stdout
```

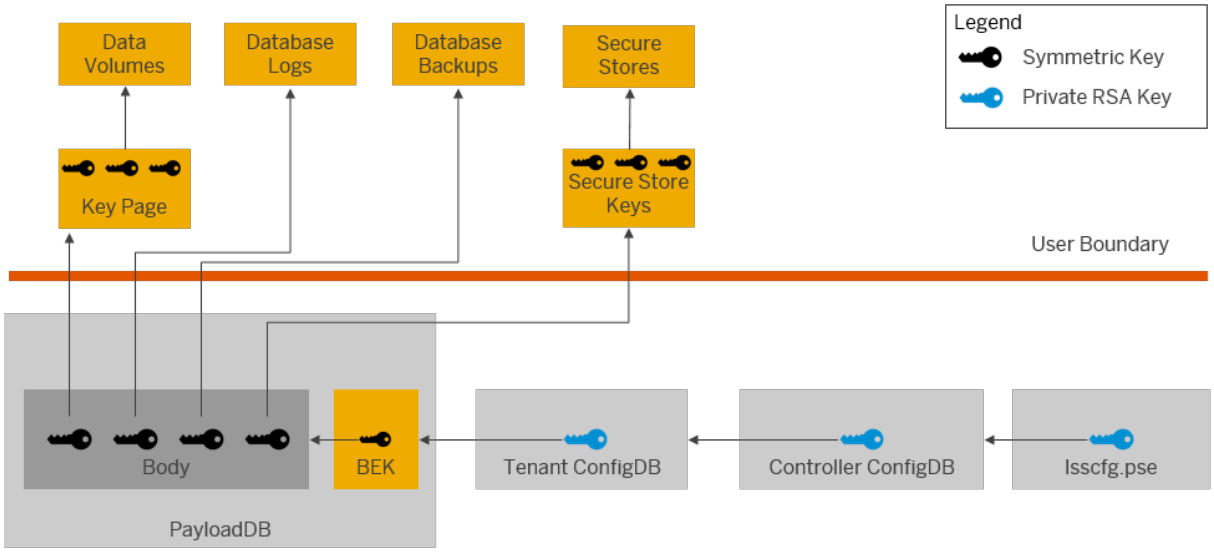
11.2.8.2.2 Encryption Key Chain

The internal key management in the local secure store (LSS) involves a chain of encryption keys.

All encryption keys are created automatically during the installation of the LSS, except the keys contained within the body of the payload database (DB) file (`lss.db`). The payload database keys are encryption root keys created and managed by SAP HANA (see the section *Encryption Key Management*). These root keys are stored in the LSS either during the installation of SAP HANA if LSS is selected as the secure store, or during a migration from the instance SSFS (secure store in the file system) to the LSS.

Local Protection (Default Setup)

The encryption key chain used in the LSS with SAP HANA is shown below:



The encryption key chain depicted above can be described as follows:

- The `lsscfg.pse` file contains the master key pair of the Controller's configuration DB file. It is stored in the local file system and therefore needs to be replicated in multiple-host scenarios.
- The Controller's configuration DB file contains the master key pairs for the tenant-specific configuration DB files of each tenant.
- A tenant-specific configuration DB file contains the master key pair to open the tenant's payload DB file (`lss.db`).

> ### Note
>
> The system database and tenant databases are both tenants in the LSS context.

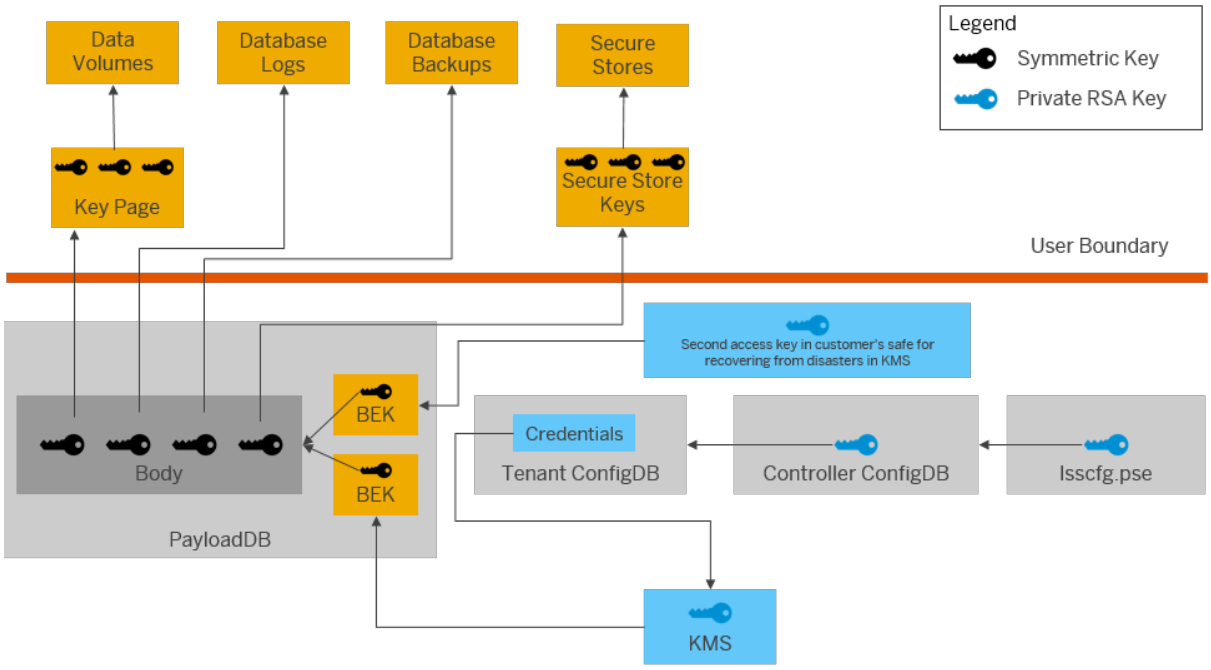
The DB files (for controller configuration, tenant configuration, and payload) are PKCS#7 files. Their body is encrypted with a body encryption key (BEK). Their header contains a copy of the body encryption key, which is encrypted with the public part of an asymmetric key (master key pair). To read the body of the file, the private part of the master key pair is needed to decrypt the body encryption key in the header so that it can be used to decrypt the corresponding body.

External Protection (Optional Setup)

If an external key management service (KMS) or hardware security module (HSM) is used, the following applies and is shown below:

- A tenant-specific configuration DB file contains the credentials required to access the external KMS or HSM.
- The external KMS or HSM contains the master key pair to open the tenant-specific payload DB file.

- A second access key pair is stored in the customer's safe so that the payload DB files can be opened if the KMS or HSM is not available due to a disaster.



Encryption Key Chain with External Key Management Service (KMS) or Hardware Security Module (HSM)

Related Information

Encryption Key Management [page 239]
Using the LSS with an External Key Management System [page 270]
SAP Note 2917358

11.2.8.2.3 LSS Trace and Dump Files

The local secure store (LSS) writes trace and dump files to a folder in the shared file system. This folder is not automatically cleaned up.

The trace folder `shared/data/trace` contains traces for database-independent processes (like the Controller), and then one sub-folder per LSS tenant (system database and every tenant database).

Each folder contains:

- Traces of the LSS service
- Traces of the command line utilities, if used (for example, `lsscfg`)
- Some overhead like `stdout`, `stderr` and dump files

Each of the above trace file types is limited to 100 files, with a limit of 30 MB per file.

> ### Note
>
> The trace folder is not automatically cleaned up. If a tenant database is dropped, its trace folder is not deleted. If you recreate the tenant database, then a new trace folder is created. Frequently dropping and

creating tenants results in a lot of trace folders with the suffix `.deletedAt-<timestamp>`. These folders must be manually cleaned up.

11.2.8.3 LSS Backup and Recovery

By automatically writing content backups, the local secure store (LSS) ensures that it can be independently recovered after a disaster.

Related Information

LSS Backup Utility (lssbackup) [page 263]

11.2.8.3.1 LSS Backup Utility (lssbackup)

`lssbackup` utility is to back up, restore and dump local secure store (LSS) database files.

The configuration and payload DB files of LSS tenants (`lsscfg.db` and `lss.db`) are backed up to and restored from a single backup file (single-file format). The Controller files (`lsscfg.db` and `lsscommon.db`) are backed up to and restored from two separate files.

> ⓘ Note
>
> Singe-file format is required if the LSS is being used in conjunction with an external key management system.

The syntax for using the `lssbackup` utility is as follows:

```
lssbackup [OPTION] COMMAND
```

Options

General Option

Option	Description
-t `<tenant>`	Operations work on the database (DB) files of the specified tenant (SYSTEMDB or tenant database name). If omitted, operations work on the Controller DB files.

Database Entry Formatting Option

Option	Description
-u	Displays user information instead of the data type

Commands

> **Note**
>
> Tenant backup files are protected with a passphrase, which you will be interactively prompted to enter.

Backup Commands

auto	Creates an automatic backup of DB files to the configured backup location. If a tenant is specified, backups of the `config` and `payload` DB files are written. Otherwise, backups of the Controller `config` and `common` DB files are written.
show `<backupfile>` [-d {payload | config}] [`<filter>`]	Prints the content of `<backupfile>` (single file) for DB file types `config` and `payload`. If a DB file type is specified, the contents for the specific DB file will be displayed (optionally filtered by `<filter>`)
export `<backupfile>` {common | config}	Exports the content of `<backupfile>` for the DB file type `common` or `config` to `<backupfile>`
export `<backupfile>`	Exports the content of a tenant's `config` and `payload` DB files to `<backupfile>` (single file)
import `<backupfile>` {common | config} [`<filter>`]	Imports the content of `<backupfile>` into the DB file of DB file type `common` or `config` (optionally filtered by `<filter>`) **Caution** Data with the same `<filter>` will be overwritten.
import `<backupfile>` [--`<filter>`]	Imports the content of `<backupfile>` (single file) into the tenant's `config` and `payload` DB files The tenant backup file is protected with a passphrase, which you will be interactively prompted to enter. **Caution** Data with the same `<filter>` will be overwritten.

restore <backupfile> {common | config}	Restores the content of <backupfile> into the DB file of database type common or config **⚠ Caution** All data in the specified DB file will be overwritten.
restore <backupfile>	Restores the content of <backupfile> (single file) into the tenant DB files of type config and payload. The tenant backup file is protected with a passphrase, which you will be interactively prompted to enter. **⚠ Caution** All data in the config and payload DB files will be overwritten.
restoreWithSecondAccessKey <backupfile> payload <configbackup> <privatekeyfile>	When the payload DB file is protected with a key in an external key management system, a second access key can be provided in addition. The second access key enables a backup to be restored even if the external key management system is not reachable. The privatekeyfile parameter specifies the file name of the PEM-encoded private key file of the second access key that was used in the LSS backup. **⚠ Caution** All data of the payload and config DB files will be overwritten. A protection change to the 'DEFAULT' configuration is performed as part of the restore operation so that a newly generated RSA key pair protects the payload DB file after a restore.
validate <backupfile>	Validates the content of the single-file format backup
Deprecated Backup Commands	
dump <backupfile> {payload <configbackup> | config | common} [<filter>]	Prints the content of the backup file for database file types common, config or payload (optionally filtered by <filter>)

export <backupfile> {payload <configbackup> | config}	Exports the content of the DB file of the specified DB file type to the specified backup file **⚠ Caution** A backup of DB file type `payload` is not self-contained; a matching backup of DB file type `config` is required in order to open the payload backup during the restore operation. Remember to export backup file of DB file type `config` as well.
import <backupfile> {payload <configbackup> | config} [<filter>]	Imports the content of the backup file into database file of database file type `payload` or `config`. The mandatory parameter `configbackup` refers to the `config` DB file's backup file, which is needed to open the payload backup file. **⚠ Caution** Data with the same <filter> will be overwritten.
restore <backupfile> {payload <configbackup> | config}	Restores the content of the backup file into database file of database file type `payload` or `config`. The mandatory parameter `configbackup` refers to the `config` DB file's backup file, which is needed to open the payload backup file. **⚠ Caution** All data of the `payload` and `config` DB files will be overwritten.

Backup Command Parameters

Parameter	Value
<filter>	A prefix of the database section (for example, 'section' 'sub section')

Related Information

Using the LSS with an External Key Management System [page 270]

11.2.8.4 LSS Configuration Utility (lsscfg)

The user <sid>`crypt` uses the `lsscfg` utility to perform advanced configuration tasks for the local secure store (LSS).

⚠ Caution

Technical expertise is required to use `lsscfg`. To avoid incorrect usage, use `lsscfg` only with the guidance of SAP HANA development support.

The syntax for using the `lsscfg` utility is as follows:

```
lsscfg [OPTION] COMMAND
```

> → Tip
>
> You can abbreviate the commands as long as they are unique. For example, `listV` for `listValidationProfiles`.

Options

General Options

Option	Description
-d	Limits the operation to the specified database (DB) file type (for example, **`common`** or **`config`**)
-f	Forces the operation to succeed
-r	Prints the result in a machine-readable format instead of a verbose human-readable format. Omits the preamble so that the output is suitable for scripting.
-t `<tenant>`	Execute operations on the DB files of the specified tenant (SYSTEMDB or tenant database name). If omitted, operations work on the Controller's DB files.

Database Entry Formatting Options

Option	Description
-m	Displays the modification timestamp instead of the data type
-u	Displays user information instead of the data type

Commands

Database Commands

Command	Description
deleteKey `<key>`	Deletes the specified key in the selected databases
deleteSection `<key>`	Deletes all keys beginning with the specified key in the selected databases
copySection `<source>`: `<target>`	Copies all keys beginning with `<source>` to the respective key starting with `<target>` in the selected databases
dump [`<key>`]	Prints the content of the selected databases (filtered by the specified key prefix)

Command	Description
getKey <key>	Gets a value using the specified key (for example, 'k1' 'k2' 'k3')
setBinary <key> <value>	Sets the specified key to the specified binary value (format: 'ff ff ff ff')
setBoolean <key> <value>	Sets the specified key to the specified boolean value (for example, 'k1' 'k2' 'k3' true)
setInteger <key> <value>	Sets the specified key to the specified integer value (for example, 'k1' 'k2' 'k3' 99)
setString <key> <value>	Sets the specified key to the specified string value (for example, 'k1' 'k2' 'k3' 'value')
setTimestamp <key> <value>	Sets the specified key to the specified UTC timestamp value (for example,'key1' '2017-01-01T14:00:30') Timestamp format: {YYYY}-{mm}-{DD}T{HH}:{MM}:{SS}

Validation Commands

Command	Description
deleteProduct <product> [<version>]	Deletes all versions of the specified product
listProducts	Lists all products that are registered for validation
trust <product> [<version>] <trust_level>	Allows connections from all versions of the specified product to the LSS
distrust <product> [<version>]	Disallows connections from all versions of the specified product to the LSS

Validation Command Parameters

Parameter	Value
<trust_level>	FINAL_ASSEMBLY | VALIDATION | INTERNAL_BUILD | UNTRUSTED

> ### Note
>
> The optional parameter <version> can be used to modify just a specific version of the specified product.

Validation Hash Commands

Command	Description
importComponentHashes <product> <version> <component_dir> [<SMF>]	Imports the hashes from the .lst files in the specified component directory as a new product
importSha256File <product> <version> <sha256_file> [<SMF>]	Imports the hashes from the specified SHA256 file generated using "sha256sum" as a new product
importTgzFile <product> <version> <tgz> [<SMF>]	Imports the hashes for all executable files in the specified archive as a new product

> **Note**
>
> Hash files can optionally be verified against a `SIGNATURE.SMF` file for automatic product trust. If an SMF file is not specified, the products are added as UNTRUSTED.

Validation Profile Commands

Command	Description
listValidationProfiles	Lists all validation profiles with their current and default settings
setValidationTrustedUser `<profile>` `<user_list>`	Sets trusted users that can connect without validation
setValidationExternalUser `<profile>` `<user_list>`	Sets external users that can connect after successful validation
setValidationMinTrust `<profile>` `<trust_level>`	Sets the minimum required validation level for external users
setValidationDebugProtection `<profile>` `<debug_protect>`	Sets debug protection enforcement (requires a Linux kernel with Yama)

Validation Profile Command Parameters

Parameter	Value
`<profile>`	`CUSTOM | DEVELOPMENT | PRODUCTION | TEST`
`<user_list>`	Comma-separated list of OS users. '`<SID>`' will be replaced by the configured SID at runtime.
`<trust_level>`	`FINAL_ASSEMBLY | VALIDATION | INTERNAL_BUILD | UNTRUSTED`
`<debug_protect>`	`DISABLED | ENABLED | STRICT`

Miscellaneous Commands

Command	Description
setBackupPassphrase	Sets the passphrase for automatic backups (the passphrase is provided interactively)
getBackupLocation `<prefix>`	Gets the path and the prefix of auto-backup files The string prefix can optionally be added to show only the prefix
setBackupLocation `<path>` [`<prefix>`]	Sets the path and optionally the prefix of auto-backup files `<path>` can be an empty string inside double quotes to define the default path (`/usr/sap/<sid>/lss/shared/backup/`).
setSystemUsage `<usage>`	Sets the usage mode to 'production', 'test', 'development', or 'custom'
getTraceLevel `<topic>`	Gets the trace level for the specified topic
listTraceLevels	Lists all available trace topics with their current level
setTraceLevel `<topic>` `<trace_level>`	Sets the specified trace level for the specified topic

Miscellaneous Command Parameters

Parameter	Value
`<trace_level>`	`DEBUG | FULL | DEBUG | INFO | WARNING | ERROR`

Change Protection Commands

Command	Description
activateKeyManagementConfiguration `<configuration_name>`	Activates a key management configuration by changing the protection of the payload database
addKeyManagementConfiguration `<configuration_name>` `<JSON>`	Adds and tests a key management configuration
dropKeyManagementConfiguration `<configuration_name>`	Drops an inactive key management configuration
getKeyManagementConfiguration `<configuration_name>`	Displays the key management configuration information for the specified configuration without credentials
listKeyManagementConfigurations	Lists all key management configurations but omits credentials
updateKeyManagementConfiguration `<configuration_name>` `<JSON>`	Updates an existing key management configuration with a JSON object

11.2.8.5 Using the LSS with an External Key Management System

The LSS payload database (DB) contains the SAP HANA encryption root keys used for encrypting SAP HANA data, log volumes, and backups.

The body of the LSS payload DB is encrypted with a AES-256-bit key. With every save-operation, a new key for this encryption is generated and stored in a header of the same LSS payload DB. This header is encrypted. By default, the header encryption key is done with a local key chain of LSS (customer-supplied encryption keys [CSEK] scenario).

If desired, the SAP HANA/LSS can be reconfigured to connect LSS to a key in Data Custodian (DC) KMS. If this is done, then an RSA key pair managed by DC KMS is used as the header encryption key. LSS can then only open its payload DB by sending the encrypted header for decryption to DC KMS. This offers the capability to disrupt the key chain by disabling or deleting the key in KMS (customer-controlled encryption keys [CCEK] scenario). However, this makes the SAP HANA/LSS strictly dependent on the key in KMS.

> **Note**
>
> Only the SAP Data Custodian Key Management Service is supported.

When SAP DC KMS is used with the LSS, the LSS payload DB is encrypted with the public key of an asymmetric key pair whose private key is stored in DC KMS. Consequently, to read and decrypt a payload DB, access to the private key of the key pair is required.

DC KMS unreachable

If DC KMS is unreachable for some reason (for example, network issues, or a downtime of KMS), SAP HANA will in most cases continue to work, thanks to the filled caches in LSS.

Only if the LSS cache is empty (usually after a full restart of SAP HANA and LSS), the SAP HANA database can hang or even crash if KMS is not reachable. The detailed error traces of LSS should help to analyze the situation.

To verify that DC KMS is available and that there is an active key, SAP HANA regularly polls into KMS. This allows LSS to respond to the following changes:

- **Key revocation**
  If the DC KMS reports that the key has been revoked (disabled or deleted), the database is shut down and cannot be restarted or recovered from a backup (until the key is available again).

  > ⚠ Caution
  >
  > Key revocation is not trivial and should never be used to temporarily prevent access to applications. While a disabled key can be re-enabled, the cost (in terms of effort and time) of resuming application operation after a temporary key disablement may be high. Deleting a key is irreversible.

- **Key rotation**
  If DC KMS reports that the key has been rotated (for example, a new primary version of the key has been created), the currently active key management configuration is updated and LSS automatically re-encrypts the payload DB with the new key. This can be confirmed by querying the `KEY_MANAGEMENT_CONFIGURATIONS` system view: the active configuration has the new key version configured.

You can configure the polling interval using the `[cryptography] key_revocation_check_interval` in the `nameserver.ini`. The value must be between 1 and 3600 seconds. The default value is 60 seconds. If you set a value outside this range, it defaults to 60 seconds. Furthermore, if you set a value under 60 seconds, a warning is displayed that the polling rate might be too high and could lead to unnecessarily high network traffic.

Second Access Key Pair

The LSS-internal caching makes HANA/LSS partly resilient against temporary KMS outages. However, a permanent loss of the key in KMS, e.g., due to a disaster in KMS, would lead to a complete loss of the HANA database.

To also eliminate this risk, you can configure a second access key pair. You generate this second access key pair yourself.

> ⓘ Note
>
> Keep the second access key pair in a safe place in case of an emergency.

Only the public key of the second access key pair is provided to LSS, as a self-signed certificate within the external key management configuration passed to SAP HANA. SAP HANA forwards this certificate to the LSS, which then uses it to add a second header to the payload DB. This allows the Payload DB to be decrypted either regularly with the help of DC KMS (using the first header), or, in an emergency case, with the private key of the second access key pair (using the second header).

To bring SAP HANA/LSS back into an operable state after a DC KMS emergency, you need to take special action in which you provide the private key of the second access key pair to LSS, by:

- Restoring the LSS component using the `lssbackup` utility (`restoreWithSecondAccessKey`)
- Recovering an `lssbackup` as the first step in recovering an encrypted SAP HANA database (`RECOVER ENCRYPTION ROOT KEYS AND SETTINGS … SECURE STORE SECOND ACCESS`)

Related Information

Create and Manage a Key Management Configuration
Create a Second Access Key
Add Second Access Certificates to the Key Management Configuration
Monitoring Key Management Configurations
Restore a Payload Database File Backup with the Second Access Key
LSS Backup Utility (lssbackup) [page 263]
Import Backed-Up Root Keys Before Database Recovery
RECOVER ENCRYPTION ROOT KEYS Statement (Backup and Recovery)
LSS Trace and Dump Files [page 262]
SAP Note 2917358
SAP Note 2911896

11.3 Client-Side Data Security

The client user store and client-side data encryption allow SAP HANA clients to protect security-relevant data from unauthorized access.

Secure User Store

The user store of the SAP HANA client (`hdbuserstore`) can be used to store user logon information for connecting to an SAP HANA system. This allows client applications to connect to the database without having to enter a user's password explicitly. It is typically used by scripts connecting to SAP HANA. For more information, see the section on the secure user store in the *SAP HANA Client Interface Programming Reference*.

Client-side Data Encryption

Client-side data encryption is a column-level data encryption capability managed by the client driver. With client-side encryption, table columns containing sensitive data (credit card numbers, for instance) are encrypted using an encryption key that is accessible only to the client. As a result, any third-party administrator can only manage the data but can't view it.

Information on Specific Client Applications

SAP HANA Cockpit

See the section on security aspects of the SAP HANA cockpit.

SAP Web IDE for SAP HANA

See the section on security aspects of the SAP Web IDE for SAP HANA.

SAP HANA Studio

- For users using the SAP HANA studio to connect to an SAP HANA system, the Eclipse secure storage can be used to store passwords. If this is not desired, the feature can be disabled for the SAP HANA studio. For more information, see *Disable Password Storage in Eclipse Secure Store* in *SAP HANA Administration with SAP HANA Studio*.The Eclipse secure storage can be used to store user passwords in the SAP HANA studio.
- Data copied to local workspaces requires additional protection. For more information, see *Protection of Data in SAP HANA Studio Workspaces*.

Microsoft Excel

Microsoft Excel is an end-user client for SAP HANA. In Microsoft Excel, you can connect to an SAP HANA system as an external data source, and then create a PivotTable to analyze that data. Connections to SAP HANA use the SAP HANA ODBO driver, which is installed with the SAP HANA client. When you are creating a connection to an SAP HANA system, you must specify a database user and password in the connection wizard.

> ⚠ Caution
>
> Although you can choose to save the password in the connection file, we recommend that you do **not** since the saved password is not encrypted.

Related Information

Client-Side Data Encryption [page 273]
SAP HANA User Store (hdbuserstore)
Security Considerations for SAP HANA Cockpit
Security Aspects of SAP Web IDE for SAP HANA [page 394]
Protection of Data in SAP HANA Studio Workspaces [page 274]
Disable Password Storage in Eclipse Secure Store

11.3.1 Client-Side Data Encryption

Client-side data encryption is a column-level data encryption capability managed by the client driver.

It provides a separation between those who own the data (and can view it) and those who manage the data (but should have no access), and delivers a built-in protection of sensitive data from other third-party database administrators and cloud administrators. With client-side encryption, table columns containing sensitive data

(credit card numbers, for instance) are encrypted using an encryption key that is accessible only to the client. Column data is encrypted and decrypted only on the client-driver, allowing the applications to read and write data in cleartext form.

Client-side encryption uses both symmetric and asymmetric encryption. Sensitive column data is encrypted with a symmetric column encryption key (CEK) which is encrypted using an asymmetric client key pair (CKP). CEKs are encrypted and stored on the SAP HANA server. The public key of the CKP is stored on the SAP HANA server and in the `hdbkeystore` (a secure key store) on the client's local machine. The private key of the CKP is stored only in the `hdbkeystore` on the client's local machine. Key generation and data encryption and decryption happens on the client driver only; SAP HANA server only stores the encrypted keys and encrypted data.

To access the encrypted data, an application must use a client driver that supports client-side encryption and the client must have access to the CEK that encrypts the column. When writing or reading encrypted data to or from the server, the application must use a prepared statement.

Client-side data encryption supports two types of encryption – non-deterministic (or randomized) and deterministic. Choose the encryption algorithm based on the intended use of the data.

Client-side data encryption also supports key rotation for column encryption keys (CEKs) and client key pairs (CKPs). Keys rotation decreases the risk of keys being breached and ensures users' data confidentiality and security. SAP Common Crypto Library (`sapcrypto.dll`) provides data encryption capabilities for client-side encryption. Client-side encryption is supported on JDBC and ODBC (SQLDBC) client drivers available with the SAP HANA client.

> **Note**
>
> Applications developed using SAP HANA Extended Applications Services, classic model (XS classic) do not use the SAP HANA client.

For more conceptual and procedural information about how to configure client-side data encryption, see the *SAP HANA Client-Side Data Encryption Guide*.

Related Information

SAP HANA Client-Side Data Encryption Guide

11.3.2 Protection of Data in SAP HANA Studio Workspaces

When users are working in the SAP HANA studio, data is copied to workspaces on their local disk for editing. This data requires additional protection.

In the SAP HANA studio, data is copied to the following workspaces on the local disks of users:

- Eclipse workspace
  When the SAP HANA studio is installed, a local workspace is created by default in the user's home directory in the `hdbstudio` sub-directory. This workspace contains for example, the connection details of SAP HANA systems that the user adds in the SAP HANA studio, as well as other configuration data.
  It is possible to change the location of this directory using the standard Eclipse *Switch Workplace* feature.

- SAP HANA repository workspaces
  In the *SAP HANA Development* perspective of the SAP HANA studio, content and application developers create repository workspaces in a local directory. This allows them to work on local copies of design-time objects from an SAP HANA repository.

To ensure that only the user can access the data in workspaces, workspaces must be created in the user's home directory. In addition, it is recommended that users encrypt the data on their hard drives using an encryption tool.

Users must delete their workspaces when they uninstall the SAP HANA studio.

11.4 Cryptographic Service Provider

All encryption services used in SAP HANA require the availability of a cryptographic service provider on the SAP HANA server and the SAP HANA client.

Server

The SAP HANA server supports the following cryptographic libraries:

- The SAP Cryptographic Library, CommonCryptoLib (default)
  CommonCryptoLib (`libsapcrypto.so`) is installed by default as part of SAP HANA server installation at `$DIR_EXECUTABLE`.
  CommonCryptoLib supports a FIPS 140-2 compliant cryptographic kernel module. If this is required, you must enable it in the system database with the parameter `[cryptography] ccl_fips_enabled` in the `global.ini` file on system database level (restart required). For more information, see SAP Note 2117112 and the *SAP HANA Administration Guide*.

  > ⓘ Note
  >
  > Encryption features available as of SAP HANA 1.0 SPS 09 require CommonCryptoLib.

- OpenSSL
  The OpenSSL library is installed by default as part of the operating system installation.

  > ⓘ Note
  >
  > **Deprecated:** OpenSSL is deprecated. You must migrate to CommonCryptoLib. For more information, see SAP Note 2093286.

Client

To use CommonCryptoLib for client communication encryption, client-side encryption, and LDAP authentication, you must download it and install it locally. For more information see the *SAP HANA Client Installation and Update Guide*.

For secure client communication, the connection property `ENCRYPT`=1 must be specified, along with any other relevant TLS/SSL options. Additionally, the `sslKeyStore` must point to a valid private key file.

Related Information

Use FIPS 140-2 Certified Cryptographic Kernel in CommonCryptoLib
Client-Side TLS/SSL Connection Properties (JDBC) [page 56]
Client-Side TLS/SSL Connection Properties (ODBC) [page 53]
Download and Install SAP Common Crypto Library
SAP Note 1848999
SAP Note 2093286
SAP Note 2117112

12 Auditing Activity in SAP HANA

Auditing allows you to monitor and record selected actions performed in the SAP HANA database. You can configure auditing independently for each database. Changes to the auditing configuration in one database have no effect on auditing in other databases in an SAP HANA system.

Although auditing does not directly increase your database's security, if wisely designed, it can help you achieve greater security in the following ways:

- Uncover security vulnerabilities if too many privileges were granted to certain users
- Show attempts to breach security
- Protect the system owner against accusations of security violations and data misuse
- Allow the system owner to meet security standards

The following actions are typically audited:

- Changes to user authorization
- Creation or deletion of database objects
- Authentication of users
- Changes to system configuration
- Access to or changing of sensitive information

Unauditable Events

Only actions that take place inside the database engine can be audited. Only changes that are made using SQL are visible to the database engine. If the database is not online when an action occurs, that action cannot be detected and therefore cannot be audited.

Events cannot be audited in the following situations:

- Upgrade of an SAP HANA system instance
  An upgrade is started when the instance is offline. When the instance is online again, it is not possible to determine which user triggered the upgrade and when.
- Direct changes to system configuration files using operating system commands
  When the system is offline, it is possible to directly edit configuration files. Such changes cannot be audited.
- Changing the password of the SYSTEM user of the system database by starting the name server in emergency mode.

> → Tip
>
> We recommend enabling audit logging in the operating system. For more information, see your operating system documentation.

SAP Host Agent

The SAP Host Agent is a tool used to perform several administration tasks in SAP systems. It is installed by default on all hosts.

The SAP HANA database lifecycle manager (HDBLCM) and the SAP HANA cockpit for offline administration rely on the SAP Host Agent to execute tasks as the system administrator user `<sid>`adm.

To audit operations performed in these SAP HANA tools, you must enable the audit logging feature of the SAP Host Agent. For more information about how to do this, see the SAP Host Agent documentation.

The SAP Host agent audit log is written to the Linux syslog.

Related Information

SAP Host Agent
SAP Note 1907566

12.1 Audit Policies

An audit policy defines the actions to be audited, as well as the conditions under which the action must be performed to be relevant for auditing. When an action occurs, the policy is triggered and an audit event is written to the audit trail. Audit policies are database specific.

Audit Actions

An audit action is an action executed in the database by an SQL statement. For example, to track user provisioning in your system, you create an audit policy that audits the execution of the SQL statements `CREATE USER` and `DROP USER`.

Although most actions correspond to the execution of a single SQL statement, some actions can cover the execution of multiple SQL statements. For example, the action `GRANT ANY` audits the granting of multiple entities on the basis of the SQL statements `GRANT PRIVILEGE`, `GRANT ROLE`, `GRANT STRUCTURED PRIVILEGE`, and `GRANT APPLICATION PRIVILEGE`.

An audit policy can specify any number of actions to be audited, but not all actions can be combined together in the same policy. Actions can be grouped in the following main ways:

- All auditable actions
  You can include all actions performed by a specific user in a single policy. This covers not only all actions that can be audited individually, but also actions that cannot otherwise be audited. Such a policy is referred to as a firefighter policy and is useful if you want to audit the actions of a particularly privileged user.

> ⚠ Caution
>
> The actions that are audited are limited to those that take place inside the database engine while it is running. Therefore, system restart and system recovery will not be audited.

> ⚠ Caution
>
> Create a firefighter policy only in exceptional circumstances, for example, to check whether a certain user is being used for everyday work or if a support user has been given access to the system. Firefighter policies may create large amounts of audit data and significantly impact performance if they are used for high-load users.

- Data manipulation actions (DML)
  You can include any actions that involve data manipulation together in a single policy, for example actions that audit `SELECT`, `INSERT`, `UPDATE`, `DELETE`, and `EXECUTE` statements on database objects. A policy that includes these actions requires at least one target object that allows the actions in question. This type of policy is useful if you want to audit a particularly critical or sensitive database object.
- Data definition actions (DDL)
  Other action types, for example actions that involve data definition, can only be combined together in a single policy if they are compatible. For example, the action `GRANT PRIVILEGE` can be combined with `REVOKE PRIVILEGE` but not with `CREATE USER`. The action `CREATE USER` can be combined with `DROP USER`.

> ⓘ Note
>
> If an audited action that involves data manipulation was executed implicitly by a procedure, the call to this procedure is audited together with the audited action. If the action does not involve data manipulation, then an implicitly executed procedure is not audited. For example, if there is an active audit policy that audits the action of creating users, the execution of `CREATE USER` statements within procedures will be audited but not the procedures themselves.

For a full list of all actions that can be audited, see the documentation for SQL access control statement `CREATE AUDIT POLICY` in the *SAP HANA SQL and Systems View Reference*.

Audit Policy Parameters

In addition to the actions to be audited, an audit policy specifies parameters that further restrict the number of events actually audited.

- Audited action status
  Each audit policy must specify when the actions in the policy are to be audited:
  - On successful execution
  - On unsuccessful execution
  - On both successful and unsuccessful execution

> ⓘ Note
>
> An unsuccessful attempt to execute an action means that the user was not authorized to execute the action. If another error occurs (for example, misspellings in user or object names and syntax errors), the action is generally not audited. In the case of actions that involve data manipulation

(that is, `INSERT`, `SELECT`, `UPDATE`, `DELETE`, and `EXECUTE` statements), additional errors (for example, invalidated views) are audited.

- Target object(s)
  Actions that involve data manipulation require at least one target object. The following target object types are possible:
  - Schemas (and all objects contained within)
  - Tables
  - Views
  - Procedures

  Target objects are specified at the level of audit policy, so if an audit policy contains several data manipulation actions, the target object must be valid for all actions in the policy. In the case of the action `EXECUTE`, the only valid target object is procedure. The reverse is also true: the only valid action for procedures is `EXECUTE`. This means that the action `EXECUTE` cannot be combined with any other actions. An object does not have to exist before it can be named as the target object of an audit policy. However, if the object does not exist, it cannot be audited by the audit policy. When an object with the specified name is subsequently created, the audit policy will apply to the object, assuming it is of a type that can be audited and the audited action applies to that object type. For example, if the audited action is `EXECUTE`, the subsequently created object must be a procedure.

  > ### Note
  > It is not possible to specify a target object for DDL actions.

- Audited user(s)
  It is possible to specify that the actions in the policy be audited only when performed by a particular user or users. Alternatively, you can specify that the actions in the policy be audited when performed by all users **except** a particular user or users. In the case of a policy that contains all auditable actions, a user must be specified.
  Users do not have to exist before they can be named in an audit policy. However, if a specified user does not exist, it cannot be audited by the audit policy. When the user is subsequently created, the audit policy will apply to the user.
- Audit level
  Each audit policy must be assigned one of the following levels:
  - EMERGENCY
  - ALERT
  - CRITICAL
  - WARNING
  - INFO

  When the audit policy is triggered, an audit entry of the corresponding level is written to the audit trail. This allows tools checking audited actions to find the most important information, for example.

Policy-Specific Audit Trail Target(s)

You can optionally configure one or more policy-specific audit trail targets. If you do not configure a policy-specific audit trail target, audit entries generated by the policy are written to the default audit trail target for the audit level of the policy if configured, or the default audit trail target configured for the system. Default audit

trail targets can be configured using the corresponding parameters in the auditing configuration section of the global.ini. see *System Properties for Configuring Auditing*.

If an action is audited by multiple audit polices and these audit policies have different audit trail targets, the audit entry is written to all trail targets.

> **Note**
>
> In tenant databases, you can only configure internal database table as the policy-specific audit trail. Otherwise, the audit trail targets configured for the database or audit level apply, which are also internal database table by default. A system administrator may change the audit trail targets for tenant databases by changing the relevant system property (`[auditing configuration] *_audit_trail_type`) in the `global.ini` file. However, this is not recommended. For more information, see *System Properties for Configuring Auditing*.

Retention Period

For audit policies that are explicitly configured to write audit entries to an internal database table, you can specify an audit retention period in days. Once the retention period has elapsed for an audit entry, the audit entry is deleted. If you add a retention period to an existing audit policy, all existing audit entries that exceed the retention period are deleted.

> **Note**
>
> A retention period can only be specified for audit policies that explicitly use the database table audit trail target. You cannot specify a retention period for audit policies that use the default audit trail targets, even if the default audit trail target is set to database table.

> **Note**
>
> The cleanup job for audit log entries runs once a day. This means that you may find audit entries up to two days outside the retention time in the audit log.
>
> For example, for a policy with a retention period of 1 day:
>
> Next cleanup job scheduled: 2021-03-01 15:00:00
>
> Most recent log entry (current time): 2021-03-01 13:00:00
>
> Last cleanup job: 2021-02-01 15:00:00
>
> Oldest log entry: 2021-01-01 15:00:00

The minimum default retention period is seven days and can be configured in the `global.ini` file with the property `[auditing configuration] minimal_retention_period`.

Related Information

Audit Trails [page 284]

System Properties for Configuring Auditing [page 295]
CREATE AUDIT POLICY Statement (Access Control)
Best Practices and Recommendations for Creating Audit Policies [page 298]

12.1.1 Audit Policies for Tenant Databases

Certain activities in tenant databases can be logged by audit policies created in the system database.

In general, audit policies for monitoring and recording activity in a tenant database are created in the tenant database with an audit that writes to a local database table. However, certain activities in tenant databases may have a security impact on the system as a whole. These activities can be monitored and recorded from the system database.

A user in the system database with the system privilege DATABASE ADMIN or DATABASE AUDIT ADMIN can create audit policies for a tenant database. The following actions may be audited:

- Actions related to session management and system configuration:
  - Creation of a user connection to the database
  - Validation of a connecting user's credentials (authentication)
  - Changes to the system configuration (*.ini) files
- Installation and deletion of licenses
- All actions related to the management of certificates and certificate collections (PSEs)
- All actions related to the management of authentication providers
- All actions related to the management of data encryption and encryption keys

Audit policies created in the system database for a tenant database are visible in the tenant database through the system view AUDIT_POLICIES. The column IS_DATABASE_LOCAL identifies where the audit policy was created, either locally in the tenant database or in the system database. In the system database, it is possible to see all audit policies created in the system database (for both the system database and all tenant databases) through the AUDIT_POLICIES view of the SYS_DATABASES schema.

Administration users in the tenant database cannot change or delete audit policies created in the system database for the tenant database, nor can they access the audit trail. Events audited by these policies are written to the audit trail defined in the system database.

Related Information

AUDIT_POLICIES System View
CREATE AUDIT POLICY Statement (Access Control)
Audit Trails [page 284]

12.1.2 Actions Audited by Default Audit Policy

If auditing is active, certain actions are always audited and are therefore not available for inclusion in user-defined audit polices. These actions are audited by the internal audit policy `MandatoryAuditPolicy`.

The actions listed below are always audited and result in audit entries with the audit level CRITICAL. Audit entries are written to the audit trail configured for this audit level. If no audit trail is configured for audit level CRITICAL, entries are written to the audit trail configured for the database.

Action	Description
• `CREATE AUDIT POLICY` • `ALTER AUDIT POLICY` • `DROP AUDIT POLICY`	Creation, modification, or deletion of audit policies
`ALTER SYSTEM CLEAR AUDIT LOG UNTIL <timestamp>`	Deletion of audit entries from the audit trail This only applies to the audit trail written to an internal database table.
• `ALTER SYSTEM ALTER CONFIGURATION ('global.ini','SYSTEM') set ('auditing configuration','global_auditing_state') = <value> with reconfigure;` • `ALTER SYSTEM ALTER CONFIGURATION ('global.ini','SYSTEM') set ('auditing configuration','default_audit_trail_type' ) = '<audit_trail_type>' with reconfigure;` • `ALTER SYSTEM ALTER CONFIGURATION ('global.ini','SYSTEM') set ('auditing configuration','default_audit_trail_path') = '<path>' with reconfigure;` • `ALTER SYSTEM ALTER CONFIGURATION ('global.ini','SYSTEM') set ('auditing configuration','audit_statement_length') = '<value in bytes>' with reconfigure;` • `ALTER SYSTEM ALTER CONFIGURATION ('global.ini','SYSTEM') set ('authentication','authentication_methods')= '<methods>' with reconfigure;` • `ALTER SYSTEM ALTER CONFIGURATION ('global.ini','SYSTEM') unset ('authentication','authentication_methods') with reconfigure;`	Changes to auditing and authentication configuration, that is: • Enabling or disabling auditing • Changing the audit trail target • Changing the location of the audit trail target if it is a CSV text file • Changing the maximum length of a statement that is audited completely • Changing enabled authentication methods

Action	Description
`ALTER DATABASE <database_name> SYSTEM USER PASSWORD <password>`	Changing the password of the SYSTEM user of a tenant database from the system database An audit entry is written to the audit trail of both the system database and the tenant database.

12.2 Audit Trails

When an audit policy is triggered, that is, when an action in the policy occurs under the conditions defined in the policy, an audit entry is created in one or more audit trails.

Audit Trail Targets

The following audit trail targets are supported for production systems:

Audit Trail Target	Description
Internal database table	Using an SAP HANA database table as the target for the audit trail makes it possible to query and analyze auditing information quickly. It also provides a secure and tamper-proof storage location. Audit entries are only accessible through the public system views AUDIT_LOG, XSA_AUDIT_LOG, and the union of these two views ALL_AUDIT_LOG. Only SELECT operations can be performed on this view by users with the system privilege AUDIT OPERATOR, AUDIT ADMIN, or AUDIT READ. **Note** By default, the audit log is stored as a Native Storage Extension (NSE) table in order to conserve main memory during operation. This behavior can be changed using the ALTER SYSTEM ALTER AUDIT LOG statement. If a database table is explicitly configured as the audit trail target of an audit policy, you can define in the audit policy a retention period after which audit entries are automatically deleted. It is also possible to delete old audit entries by truncating the table. You can do this in SAP HANA cockpit or with the SQL statement ALTER SYSTEM CLEAR AUDIT LOG. The system monitors the size of the table with respect to the overall memory allocation limit of the system and issues an alert when it reaches defined values (by default 5%, 7%, 9%, and 11% of the allocation limit). This behavior can be configured with check 64 ("Total memory usage of table-based audit log"). Only users with the system privilege AUDIT OPERATOR can truncate the audit table.

Audit Trail Target	Description
Logging system of the Linux operating system (syslog)	The syslog is a secure storage location for the audit trail because not even the database administrator can access or change it. There are also numerous storage possibilities for the syslog, including storing it on other systems. In addition, the syslog is the default log daemon in UNIX systems. The syslog therefore provides a high degree of flexibility and security, as well as integration into a larger system landscape. For data protection and privacy the SAP HANA audit log uses the `authpriv` facilities `HDB_TENANTDB` and `HDB_SYSTEMDB`. Check your operating system configuration files for the location of the syslog files. For more information about how to configure syslog, refer to the documentation of your operating system. **⚠ Caution** If the syslog daemon cannot write the audit trail to its destination, you will not be notified. To avoid a situation in which audited actions are occurring, but audit entries are not being written to the audit trail, ensure that the syslog is properly configured and that the audit trail target is accessible and has sufficient space available.
CSV text file	A separate CSV file is created for every service that executes SQL. By default, CSV files are written to the same directory as trace files (`/usr/sap/<sid>/<instance>/<host>/trace`). This means that database users with the system privilege DATA ADMIN, CATALOG READ, TRACE ADMIN, or INIFILE ADMIN can access them. In the SAP HANA database explorer, they are listed under *Database Diagnostics Files*, and at operating system level, any user in the SAPSYS group can access them. CSV audit trail files are created for each server in a distributed database system. For a holistic view, you need to merge the individual files in a distributed setup.

ⓘ Note

Additionally, the option exists to write the audit trail to a kernel trace file (`*.ltc`) in the trace directory (`/usr/sap/<sid>/<instance>/<host>/trace`).

The kernel trace output is not human-readable. It must be converted into a CSV files using the command-line tool `hdbltracediag` and then loaded into relational tables for SQL analysis.

`hdbltracediag` is available on the SAP HANA server at `/usr/sap/<sid>/HDB<instance>/exe`.

Multiple Audit Trails

Separate audit trail targets may be configured for the audit level, that is the severity of the action being audited.

Audit entries from audit policies with the audit level EMERGENCY, CRITICAL, or ALERT are written to the audit trail target(s) specified for the audit level in question. If no audit trail target is configured for an audit level, entries are written to the audit trail target configured for the database.

Audit policy-specific targets are also possible in the system database. In this case, audit entries from a particular policy are written to the specified audit trail target(s). If no audit trail target is configured for an

audit policy, entries are written to the audit trail target for the audit level if configured, or the audit trail target configured for the database. Several audit trail targets can be configured for each individual policy.

Audit Trails for Tenant Databases

By default, tenant database administrators **cannot** configure audit trail targets independently for their database since the underlying system properties are in the default configuration change blocklist (`multidb.ini`). For maximum isolation of audit trails between individual tenants, the default target for all audit trails in tenant databases is internal database table. It is possible to change the audit trail target of a tenant database in the following ways:

- The system administrator changes the audit trail targets for individual tenant databases directly by configuring the relevant system property (`default_audit_trail_type`, `emergency_audit_trail_type`, `alert_audit_trail_type`, `critical_audit_trail_type`) in the `global.ini` file.
- The system administrator removes the relevant system property (`default_audit_trail_type`, `emergency_audit_trail_type`, `alert_audit_trail_type`, `critical_audit_trail_type`) from the configuration change blocklist, thus enabling the tenant database administrator to change the audit trail target.

> ⚠ Caution
>
> To ensure the privacy of tenant database audit trails, it is recommended that you do **not** change the default audit trail target (internal database table) of tenant databases.

Audit Trail in Active/Active (Read Enabled) Scenario

Active/Active (read enabled) is a feature that enables SAP HANA system replication to support read access on the secondary system. In this scenario, audit log entries on the secondary system cannot be written to the internal audit log table. Therefore, on the secondary system all audit log entries that are written to the internal database table will automatically be forwarded to the syslog audit trail instead.

> ⓘ Note
>
> It is possible to configure CSV text file as the alternative audit trail target in the secondary system using the global.ini parameter `[auditing configuration] sr_audit_trail_type_cstable_override`. However, this is not recommended for the reasons mentioned above.

Related Information

Delete Audit Entries
ALTER SYSTEM CLEAR AUDIT LOG Statement (System Management)
Audit Trail Layout for Trail Target CSV and SYSLOG [page 287]

Audit Trail Layout for Trail Target Database Table [page 290]
ALTER SYSTEM ALTER AUDIT LOG (System Management)

12.2.1 Audit Trail Layout for Trail Target CSV and SYSLOG

For each occurrence of an audited action, one or more audit entries are created and written to the audit trail. The layout of audit entries varies depending on the audit trail type.

The following table describes the layout of the audit trail when either the syslog or a CSV file is the trail target:

Field	Description	Example
Event Timestamp	Local system time of event occurrence	2013-06-18 09:17:35 (ansi_sec) 2021-02-22T14:44:27.192575Z (iso) **Note** The timestamp format for entries in the syslog is always iso (in line with syslog protocol requirements). For entries in a CSV file, the timestamp configured with the parameter `[auditing_csvtextfile] timestamp_format` applies (default ansi_sec).
Service Name	Name of the service where the action occurred	Indexserver
Hostname	Name of the host where the action occurred	myhanablade23.customer.corp
SID	System ID	HAN
Instance Number	Instance number	23
Port Number	Port number	32303
Database Name	The name of the tenant database **Note** This field is available only in the syslog audit trail.	SYSTEMDB or the name of the tenant database
Client IP Address	IP address of the client application	127.0.0.2
Client Name	Name of the client machine	lu241511
Client Process ID	Process ID of the client process	19504
Client Port Number	Port of the client process	47273
Policy Name	Audit policy that was triggered	AUDIT_GRANT, MandatoryAuditPolicy

Field	Description	Example
Audit Level	Severity of audited action	CRITICAL
Audit Action	Action that was audited and thus triggered the policy	GRANT PRIVILEGE
Session User	User who is connected to the session	MYADMIN
Target Schema	Name of the schema where the action occurred, for example, a privilege was granted on a schema, or a statement was executed on object in a schema	PRIVATE
Target Object	Name of the object on which an action was performed, for example, a privilege was granted	HAXXOR
Privilege Name	Name of the privilege that was granted or revoked	SELECT
Grantable	Indication of whether the privilege or role was granted with or without GRANT/ADMIN OPTION	NON GRANTABLE
Role Name	Name of the role that was granted or revoked	MONITORING
Target Principal	Name of the target user of the action, for example, grantee in a GRANT statement	HAXXOR
Action Status	Execution status of the statement	SUCCESSFUL
Component	Name of the configuration file in which a parameter value was changed	indexserver.ini
Section	Name of the configuration file section in which a parameter value was changed	auditing_configuration
Parameter	Name of the configuration parameter whose value was changed	global_auditing status
Old Value	Previous value of the parameter	CSVTEXTFILE
New Value	New parameter value	CSTABLE
Comment	Additional information about failed user connection attempts	user is locked Currently in case of failed logon attempts, the reason for failure appears in this field.
Executed Statement	Statement that was executed	GRANT SELECT ON SCHEMA PRIVATE TO HAXXOR
Session ID	ID of the session in which the statement was executed	400006

Field	Description	Example
Application User Name	Application user name	A099999 **⚠ Caution** Treat this information with caution. It comes from the application and SAP HANA has no way of verifying its authenticity.
Role Schema Name	Name of the schema in which a role was created/dropped, or the schema of a granted/revoked role	MYSCHEMA
Grantee Schema Name	Name of the schema of a granted or revoked role	MYSCHEMA
Origin Database Name	Name of the tenant database in which the query originated; relevant for cross-database queries between tenant databases	DB1
Origin User Name	Name of the database user who executed the query in the origin tenant database; relevant for cross-database queries between tenant databases	MYADMIN
XS Application User Name	XS application user name	XSA_ADMIN
Application Name	Name of the application	sap.hana.cons
Statement User Name	Name of the user who executed the statement	DEMO
Create Time	Currently only filled for XSA events: Time of event occurrence at client side	01.05.2018 13:13:21.273
XSA_MESSAGE_IP	Currently only filled for XSA events: IP address of event occurrence	127.0.0.1
XSA_TENANT	Currently only filled for XSA events: XSA tenant GUID	testAuditlogServiceTenant
XSA_UUID	Currently only filled for XSA events: Unique audit log message ID generated by the audit service	A48E7693FEFE0089CA0A776BDAB7F745
XSA_CHANNEL	Currently only filled for XSA events: Communication protocol (for example, http, JCO, websockets) that was used when the audit event was triggered	UI
XSA_ATTACHMENT_ID	Currently only filled for XSA events: ID of the attachment that triggered the event	id:456

Field	Description	Example
XSA_ATTACHMENT_NAME	Currently only filled for XSA events: Name of the attachment that triggered the event	File.text
XSA_ORGANIZATION_ID	Currently only filled for XSA events: Application organization GUID	48a341df-5869-4b80-ba5e-dc9102d06e70
XSA_SPACE_ID	Currently only filled for XSA events: Application space GUID	87c9f8ff-511b-4a81-a6b8-d81b701839d7
XSA_INSTANCE_ID	Currently only filled for XSA events: GUID of the used auditlog service instance	74f3d921-55b9-434a-bbf1-13487e424828
XSA_BINDING_ID	Currently only filled for XSA events: Application binding GUID in regards to the specific auditlog service instance that is being used	358ca146-490e-4fb6-bc18-693720f6f089
XSA_OBJECT	Currently only filled for XSA events: Object containing the accessed personal data	{"type":"type","id": {"key1":"value1","key2":"value2","key3":"value3"}}
XSA_DATA_SUBJECT	Currently only filled for XSA events: Owner of the accessed personal data	{"type":"type","role":"role","id": {"key1":"value1","key2":"value2","key3":"value3"}}

Example

```
2013-11-30
13:04:54;indexserver;myhanablade23.customer.corp;HAN;01;30103;10.29.14.177;lu3
06309;6776;58060;Alter User Policy;INFO;ALTER
USER;SYSTEM;;;;;;ADAMS;SUCCESSFUL;;;;;;alter user ADAMS
VAXXXXXXXXXXXXX;434597;
```

Related Information

System Properties for Configuring Auditing [page 295]

12.2.2 Audit Trail Layout for Trail Target Database Table

For each occurrence of an audited action, one or more audit entries are created and written to the audit trail. The layout of audit entries varies depending on the audit trail type.

When database table is the trail target, audit entries are accessible through the public system views AUDIT_LOG and XSA_AUDIT_LOG, as well as the union of these views ALL_AUDIT_LOG. The table below describes the layout of the full audit trail, that is ALL_AUDIT_LOG.

> **Note**
>
> Only SELECT operations can be performed on these views by users with the system privilege AUDIT OPERATOR, AUDIT READ, or AUDIT ADMIN. AUDIT READ also allows access to the AUDIT LOG table.

Column name	Data type	Description
TIMESTAMP	TIMESTAMP	Displays the time that the event occurred.
HOST	VARCHAR(64)	Displays the name of the host where the event occurred.
PORT	INTEGER	Displays the port number.
SERVICE_NAME	VARCHAR(32)	Displays the name of the service.
CONNECTION_ID	INTEGER	Displays the connection ID.
CLIENT_HOST	NVARCHAR(256)	Displays the client host.
CLIENT_IP	VARCHAR(45)	Displays the IP address of the client application.
CLIENT_PID	BIGINT	Displays the PID of the client process.
CLIENT_PORT	INTEGER	Displays the port of the client process.
USER_NAME	NVARCHAR(256)	Displays the name of the user that is connected to the database.
STATEMENT_USER_NAME	NVARCHAR(256)	Displays the name of the user who executed the statement.
APPLICATION_NAME	NVARCHAR(256)	Displays the name of the application.
APPLICATION_USER_NAME	NVARCHAR(256)	Displays the name of the application user. **Caution** Treat this information with caution. It comes from the application and SAP HANA has no way of verifying its authenticity.
XS_APPLICATION_USER_NAME	NVARCHAR(256)	Displays the name of the XS application user.
AUDIT_POLICY_NAME	NVARCHAR(256)	Displays the name of the Audit Policy hit.

Column name	Data type	Description
EVENT_STATUS	VARCHAR(32)	Displays whether the event was successful or not.
EVENT_LEVEL	VARCHAR(16)	Displays the severity level of the event.
EVENT_ACTION	VARCHAR(64)	Displays the action performed by the audit event.
SCHEMA_NAME	NVARCHAR(256)	Displays the name of the schema.
OBJECT_NAME	NVARCHAR(256)	Displays the name of the object.
PRIVILEGE_NAME	NVARCHAR(256)	Displays the name of the granted privilege.
ROLE_SCHEMA_NAME	NVARCHAR(256)	Displays the schema name of the granted role.
ROLE_NAME	NVARCHAR(256)	Displays the name of the granted role.
GRANTEE_SCHEMA_NAME	NVARCHAR(256)	Displays the schema name of the grantee in the GRANT/REVOKE statements.
GRANTEE	NVARCHAR(256)	Displays the grantee in the GRANT/REVOKE statements.
GRANTABLE	VARCHAR(16)	Displays whether or not the privilege or role is grantable.
FILE_NAME	VARCHAR(256)	Displays the name of the configuration file that has been changed.
SECTION	VARCHAR(128)	Displays the configuration that has been changed.
KEY	NVARCHAR(2000)	Displays the attribute that was changed.
PREV_VALUE	NVARCHAR(5000)	Displays the old value of the attribute.
VALUE	NVARCHAR(5000)	Displays the new value of the attribute.
STATEMENT_STRING	NCLOB	Displays the SQL statement that caused the event.
COMMENT	VARCHAR(5000)	Displays extra information about the event.

Column name	Data type	Description
ORIGIN_DATABASE_NAME	NVARCHAR(256)	Displays the original database name on cross-database queries.
ORIGIN_USER_NAME	NVARCHAR(256)	Displays the original user name on cross-database queries.
CREATE_TIME	TIMESTAMP	(Applies to XSA-events) Displays the time of the event occurrence on the client side.
XSA_MESSAGE_IP	VARCHAR(45)	(Applies to XSA-events) Displays the PI address of the event occurrence.
XSA_TENANT	VARCHAR(36)	(Applies to XSA-events) Displays the XSA tenant GUID.
XSA_UUID	VARCHAR(256)	(Applies to XSA-events) Displays the unique audit log message ID generated by the audit service.
XSA_CHANNEL	VARCHAR(16)	(Applies to XSA-events) Displays the communication protocol that was used when the audit event was triggered.
XSA_ATTACHMENT_ID	NVARCHAR(256)	(Applies to XSA-events) Displays the ID of the attachment that triggered the event.
XSA_ATTACHMENT_NAME	NVARCHAR(256)	(Applies to XSA-events) Displays the name of the attachment that triggered the event.
XSA_ORGANIZATION_ID	VARCHAR(36)	(Applies to XSA-events) Displays the application organization GUID.
XSA_SPACE_ID	VARCHAR(36)	(Applies to XSA-events) Displays the application space GUID.
XSA_INSTANCE_ID	VARCHAR(36)	(Applies to XSA-events) Displays the GUID of the used auditlog service instance.
XSA_BINDING_ID	VARCHAR(36)	(Applies to XSA-events) Displays the application binding GUID in regards to the specific auditlog service instance that is being used.

Column name	Data type	Description
XSA_OBJECT	NVARCHAR(5000)	(Applies to XSA-events) Displays the object containing the accessed personal data.
XSA_DATA_SUBJECT	NVARCHAR(5000)	(Applies to XSA-events) Displays the owner of the accessed personal data.
PASSPORT_ROOTCONTEXT_ID	NVARCHAR(256)	Displays the passport root context ID.
PASSPORT_TRANSACTION_ID	NVARCHAR(256)	Displays the passport transaction ID.
PASSPORT_CONNECTION_ID	NVARCHAR(256)	Displays the passport connection ID.

Related Information

AUDIT_LOG System View
XSA_AUDIT_LOG System View
ALL_AUDIT_LOG System View

12.3 Auditing Configuration and Audit Policy Management

To audit database activity, auditing must first be enabled in the database, and if necessary audit trails configured. It is then possible to create and activate the required audit policies. Audit policies can also be deactivated and reactivated later, or deleted altogether.

System properties in the `global.ini` file (tenant databases) and the `nameserver.ini` file (system database) control the configuration of auditing and audit trail targets. However, we recommend that you configure auditing and manage audit policies using the *Auditing* app of the SAP HANA cockpit. Additional properties are available for configuring the audit trail target CSV text file.

Related Information

System Properties for Configuring Auditing [page 295]
Configuring Database Auditing

12.3.1 System Properties for Configuring Auditing

The system properties for configuring auditing are in the `auditing configuration` section of the `global.ini` system properties file (tenant databases) or `nameserver.ini` file (system database).

The following system properties are used to configure auditing. It is recommended that you not edit these properties directly, but use the *Auditing* app of the SAP HANA cockpit.

> **Note**
>
> All `*_audit_trail_type` properties (such as `default_audit_trail_type`, `emergency_audit_trail_type`, `alert_audit_trail_type`, and `critical_audit_trail_type`) are in the default configuration change blocklist (`multidb.ini`). This means that they cannot initially be changed in tenant databases. They must be changed from the system database. If appropriate for your scenario, you can remove these properties from the change blocklist. SAP HANA deployment options are described in the *SAP HANA Master Guide*. For more information about how to edit the change blocklist, see the *SAP HANA Administration Guide*.

System Property	Value	Default Value	Description
`global_auditing_state`	`<Boolean value>`	false	Status of auditing in the database **Note** In the system database, this property can only be in the `nameserver.ini` file, not `global.ini`. This makes it possible to enable auditing for the system database independently of tenant databases.
`default_audit_trail_type`	{SYSLOGPROTOCOL | CSTABLE | CSVTEXTFILE | KERNELTRACE }	• SYSLOGPROTOCOL if `global_auditing_state` is true (system database) • CSTABLE if `global_auditing_state` is true (tenant database)	Overall audit trail target of the database
`default_audit_trail_path`	`<file>`	`/usr/sap/<sid>/<instance>/<host>/trace` if `default_audit_trail_type` is CSVTEXTFILE (system database)	The file path of audit trail target CSVTEXTFILE

System Property	Value	Default Value	Description
`emergency_audit_trail_type`	{SYSLOGPROTOCOL | CSTABLE | CSVTEXTFILE | KERNELTRACE}	--	Default audit trail target to which audit entries from audit policies with the audit level EMERGENCY are written unless an audit trail target is specified in the audit policy definition
`alert_audit_trail_type`	{SYSLOGPROTOCOL | CSTABLE | CSVTEXTFILE | KERNELTRACE}	--	Default audit trail target to which audit entries from audit policies with the audit level ALERT are written unless an audit trail target is specified in the audit policy definition
`critical_audit_trail_type`	{SYSLOGPROTOCOL | CSTABLE | CSVTEXTFILE | KERNELTRACE}	--	Default audit trail target to which audit entries from audit policies with the audit level CRITICAL are written unless an audit trail target is specified in the audit policy definition
`audit_statement_length`	`<Value in bytes>`	-1	The maximum length of a statement that is audited completely Statements that exceed the maximum length are truncated once the limit is reached. The default value sets no limit. The complete statement is written to the audit trail. **⚠ Caution** Limiting the length of the audit statement string output might compromise your audit log. For example, an attacker who knows about this limitation can simply prefix sensitive statements with the corresponding number of whitespace characters to prevent the actual statement string being written to the audit trail.

System Property	Value	Default Value	Description
`sr_audit_trail_type_cstable_override`	{SYSLOGPROTOCOL | CSVTEXTFILE}	SYSLOGPROTOCOL	In an active/active (read enabled) scenario, the audit trail target to which audit entries are written in the secondary system if CSTABLE is configured as a trail type in the primary system. This is necessary because it is not possible to write to an internal database table in the secondary system.
`minimal_retention_period`	`<Integer greater than or equal to 2>`	7	Minimum number of days that can be configured as the retention period in an audit policy The retention period is the number of days after which audit entries written by an audit policy to the audit trail target CSTABLE are deleted. **Note** Only audit policies that have the audit trail target CSTABLE explicitly configured can have a retention period.

Example

- Enable auditing in system database: `ALTER SYSTEM ALTER CONFIGURATION ( 'nameserver.ini', 'system' ) set ( 'auditing configuration' , 'global_auditing_state' ) = 'true' ;`
- Enable auditing in tenant database from that database: `ALTER SYSTEM ALTER CONFIGURATION ('global.ini', 'system') set ('auditing configuration', 'global_auditing_state') = 'true';`

Additional Properties for Configuring Audit Trail Target CSVTEXTFILE

The following properties are in the `auditing_csvtextfile` section of the `global.ini` system properties file (tenant databases) or `nameserver.ini` file (system database). They can be changed in the SAP HANA cockpit on the *Database Configuration* page.

> **Note**
>
> The parameters `max_file_size` and `max_files` are in the default configuration change blocklist (`multidb.ini`). See note above.

System Property	Value	Default Value	Description
`timestamp_format`	{iso | ansi_sec | ansi_msec | ansi_usec}	ansi_sec	Specifies the required time-stamp format in the csv files
`max_file_size`	`<Value in MB>`	100	Maximum file size of each csv formatted audit log file in MB. This parameter is ignored if `[auditing_csvtextfile] max_files` is set to 0.
`max_files`	`<Integer greater than or equal to 0>`	0	Maximum number of audit log files maintained by the service The value 0 disables file splitting.

Related Information

SAP HANA Master Guide
Prevent Changes to System Properties in Tenant Databases
Configuring Database Auditing
Modify a System Property in SAP HANA Cockpit

12.4 Best Practices and Recommendations for Creating Audit Policies

Best practices and recommendations to help you create audit policies

Principles and Best Practices

Create audit policies for the following purposes in the database:

- Intrusion detection
- Security change log and traceability

- Data protection and privacy
- System change log
- Monitoring

When creating audit policies, consider the following general principles:

- Define audit policies that avoid unnecessary events in the audit log as much as possible.
- Log all changes related to security configuration.
- Log all changes related to users and user authorization.
- Unexpected events often give the most information. The principle is to define auditing for everything and exclude the expected.

To reduce the performance impact of auditing, consider the following basic guidelines:

- Create as few audit policies as possible. It is usually better to have one complex policy than several simple ones.

  > → Remember
  >
  > Some audit actions cannot be combined in the same policy.

- Use audit actions that combine other actions where possible.

  > ✤ Example
  >
  > Audit the `GRANT ANY` action instead of the `GRANT PRIVILEGE` and the `GRANT STRUCTURED PRIVILEGE` actions.

- Create audit policies for DML actions only if required. Auditing DML actions impacts performance more than auditing DDL actions.
- Do not create audit policies for actions that are automatically audited, for example `CREATE AUDIT POLICY`. For a list of actions that are always audited, see *Actions Audited by Default Audit Policy*.
- Do not create audit policies for database-internal tables that are involved in administration actions. Create policies for the administration actions themselves.

  > ✤ Example
  >
  > `P_USER_PASSWORD` is an internal database table that cannot be accessed by any user, not even the `SYSTEM`. Changes in these tables are carried out by internal mechanisms, and not by DML operations. Don't include these tables in an audit policy. Instead create an audit policy for changes to users (`ALTER USER` action) instead.

- Create a firefighter policy (that is, a policy that audits all actions for a user) only in exceptional circumstances, for example, to check whether a certain user is being used for everyday work or if a support user has been given access to the system. Firefighter policies may create large amounts of audit data and significantly impact performance if they are used for high-load users.

Recommended Audit Policies

We recommend creating the basic set of audit policies detailed below in the SAP HANA database. All of these policies can be created using the basic setup wizard for auditing in the SAP HANA cockpit.

> **Note**
>
> As the audit log written by these audit policies will take up space in your database, all audit policies are defined with a suggested retention period. This retention period has been calculated based on SAP's assessment of how critical each individual audit policy is and the volume of audit entries each policy is expected to generate. We recommend that you closely monitor the size of the AUDIT_LOG table to ensure that its size does not affect your daily operations. Delete or export unnecessary entries in the AUDIT_LOG table as appropriate. For more information about the audit trail "database table", see *Audit Trails*.

All of the following policies can be created using the auditing configuration wizard of the SAP HANA cockpit.

Intrusion detection

- Unsuccessful connection attempts:

```
CREATE AUDIT POLICY "_SAP_session connect" AUDITING UNSUCCESSFUL CONNECT
LEVEL ALERT TRAIL TYPE TABLE RETENTION 20;
ALTER AUDIT POLICY "_SAP_session connect" ENABLE;
```

- Attempts to validate user credentials

```
CREATE AUDIT POLICY "_SAP_session validate" AUDITING ALL VALIDATE USER LEVEL
ALERT TRAIL TYPE TABLE RETENTION 20;
ALTER AUDIT POLICY "_SAP_session validate" ENABLE;
```

Security configuration

- Changes to authorization

```
CREATE AUDIT POLICY "_SAP_authorizations" AUDITING ALL GRANT ANY, REVOKE ANY
LEVEL INFO TRAIL TYPE TABLE RETENTION 180;
ALTER AUDIT POLICY "_SAP_authorizations" ENABLE;
```

- Changes to users

```
CREATE AUDIT POLICY "_SAP_user administration" AUDITING SUCCESSFUL ALTER
ROLE, ALTER USER, ALTER USERGROUP, CREATE ROLE, CREATE USER, CREATE
USERGROUP, DROP ROLE, DROP USER, DROP USERGROUP LEVEL INFO TRAIL TYPE TABLE
RETENTION 180;
ALTER AUDIT POLICY "_SAP_user administration" ENABLE;
```

- Changes to structured privileges in development systems

```
CREATE AUDIT POLICY "_SAP_structured privileges" AUDITING SUCCESSFUL ALTER
STRUCTURED PRIVILEGE, CREATE STRUCTURED PRIVILEGE, DROP STRUCTURED PRIVILEGE
LEVEL INFO TRAIL TYPE TABLE RETENTION 180;
ALTER AUDIT POLICY "_SAP_structured privileges" ENABLE;
```

System configuration

- Changes to certificates and certificate collections

```
CREATE AUDIT POLICY "_SAP_certificates" AUDITING ALL ALTER PSE, CREATE
CERTIFICATE, CREATE PSE, DROP CERTIFICATE, DROP PSE LEVEL INFO TRAIL TYPE
TABLE RETENTION 180;
ALTER AUDIT POLICY "_SAP_certificates" ENABLE;
```

- Changes to authentication providers

```
CREATE AUDIT POLICY "_SAP_authentication provider" AUDITING ALL ALTER JWT
PROVIDER, ALTER LDAP PROVIDER, ALTER SAML PROVIDER, CREATE JWT PROVIDER,
```

```
CREATE LDAP PROVIDER, CREATE SAML PROVIDER, DROP JWT PROVIDER, DROP LDAP
PROVIDER, DROP SAML PROVIDER, VALIDATE LDAP PROVIDER LEVEL CRITICAL TRAIL
TYPE TABLE RETENTION 180;
ALTER AUDIT POLICY "_SAP_authentication provider" ENABLE;
```

- Changes to client-side encryption

```
CREATE AUDIT POLICY "_SAP_clientside encryption" AUDITING ALL ALTER
CLIENTSIDE ENCRYPTION COLUMN KEY, ALTER CLIENTSIDE ENCRYPTION KEYPAIR, CREATE
CLIENTSIDE ENCRYPTION COLUMN KEY, CREATE CLIENTSIDE ENCRYPTION KEYPAIR, DROP
CLIENTSIDE ENCRYPTION COLUMN KEY, DROP CLIENTSIDE ENCRYPTION KEYPAIR LEVEL
CRITICAL TRAIL TYPE TABLE RETENTION 180;
ALTER AUDIT POLICY "_SAP_clientside encryption" ENABLE;
```

- Changes to user `_SYS_REPO` authorization in development systems:

```
CREATE AUDIT POLICY "_SAP_designtime privileges"
  AUDITING SUCCESSFUL
    EXECUTE ON
      "_SYS_REPO"."GRANT_ACTIVATED_ANALYTICAL_PRIVILEGE",
      "_SYS_REPO"."GRANT_ACTIVATED_ROLE",
      "_SYS_REPO"."GRANT_APPLICATION_PRIVILEGE",
      "_SYS_REPO"."GRANT_PRIVILEGE_ON_ACTIVATED_CONTENT",
      "_SYS_REPO"."GRANT_SCHEMA_PRIVILEGE_ON_ACTIVATED_CONTENT",
      "_SYS_REPO"."REVOKE_ACTIVATED_ANALYTICAL_PRIVILEGE",
      "_SYS_REPO"."REVOKE_ACTIVATED_ROLE",
      "_SYS_REPO"."REVOKE_APPLICATION_PRIVILEGE",
      "_SYS_REPO"."REVOKE_PRIVILEGE_ON_ACTIVATED_CONTENT",
      "_SYS_REPO"."REVOKE_SCHEMA_PRIVILEGE_ON_ACTIVATED_CONTENT"
  LEVEL INFO TRAIL TYPE TABLE RETENTION 180;
ALTER AUDIT POLICY "_SAP_designtime privileges" ENABLE;
```

- Changes to SAP HANA configuration files (*.ini files)

```
CREATE AUDIT POLICY "_SAP_configuration changes" AUDITING ALL STOP SERVICE,
SYSTEM CONFIGURATION CHANGE LEVEL INFO TRAIL TYPE TABLE RETENTION 180;
ALTER AUDIT POLICY "_SAP_configuration changes" ENABLE;
```

- Changes to the SAP HANA license key

```
CREATE AUDIT POLICY "_SAP_license addition" AUDITING ALL SET SYSTEM LICENSE
LEVEL INFO TRAIL TYPE TABLE RETENTION 180;
ALTER AUDIT POLICY "_SAP_license addition" ENABLE;
CREATE AUDIT POLICY "_SAP_license deletion" AUDITING ALL UNSET SYSTEM LICENSE
LEVEL INFO TRAIL TYPE TABLE RETENTION 180;
ALTER AUDIT POLICY "_SAP_license deletion" ENABLE;
```

Monitoring

- Backup and recovery of database

```
CREATE AUDIT POLICY "_SAP_recover database" AUDITING ALL BACKUP CATALOG
DELETE, BACKUP DATA, RECOVER DATA LEVEL INFO TRAIL TYPE TABLE RETENTION 180;
ALTER AUDIT POLICY "_SAP_recover database" ENABLE;
```

> **Note**
>
> This policy does not audit scheduled backups, only manually executed backups. You must audit the CREATE SCHEDULER JOB, ALTER SCHEDULER JOB, and DROP SCHEDULER JOB events to log backup schedule modifications.

Additional policies in production systems

In production systems, additional audit policies are usually required to log further activities as defined by IT policy and to meet governance and legal requirements such as SOX compliance.

We also recommend auditing not only successful events but unsuccessful events by defining the audit action status `ALL`. Knowing about unsuccessful events might be a prerequisite to discovering an attack on your system.

> ⚠ Caution
>
> SAP HANA audit policies are defined at the database level and cannot cover all requirements for data protection and privacy. The business semantics of data are part of the application definition and implementation. It is therefore the application that "knows", for example, which tables in the database contain sensitive personal data, or how business level objects, such as sales orders, are mapped to technical objects in the database.

Related Information

CREATE AUDIT POLICY Statement (Access Control)
Actions Audited by Default Audit Policy [page 283]
Audit Trails [page 284]
Configure the Basic Setup for Audit Policies

13 Certificate Management in SAP HANA

SAP HANA uses X.509 client certificates as the basis for securing internal and external communication channels, as well as for several user authentication mechanisms. Certificates can – or in some case must – be stored and managed in the SAP HANA database. For some purposes, files stored in the file system are possible.

13.1 Certificate Management in the Database

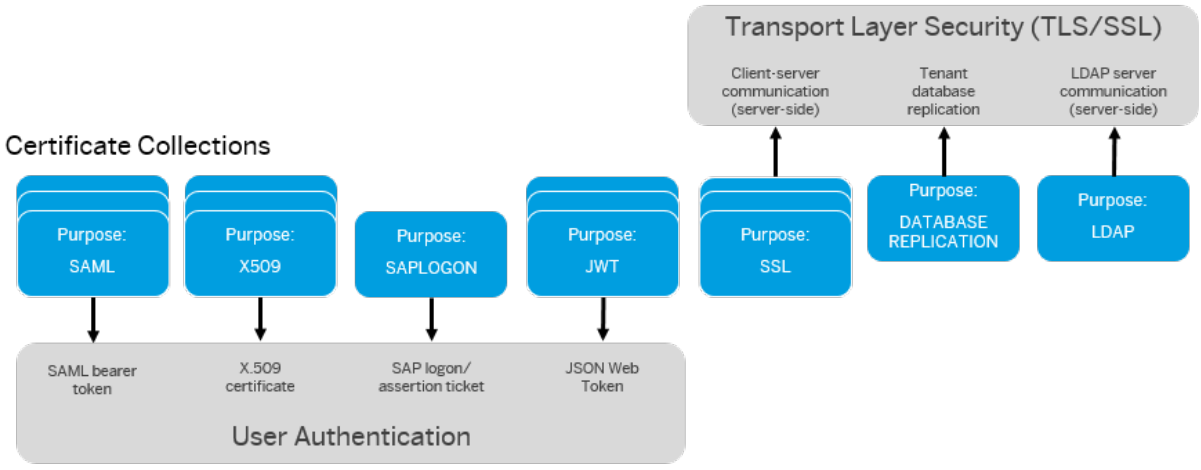
The X.509 client certificates used for securing external communication channels and several user authentication mechanisms can – or in some case must – be stored and managed in the SAP HANA database.

Certificates stored in the SAP HANA database are used for trust validation. They are the public-key certificates of trusted communication partners or root certificates from trusted Certification Authorities. In other words, they contain the public part of a user's or component's public and private key pair.

Once they have been imported into the database, certificates can be assigned to certificate collections. Certificate collections are also created and managed directly in the database, where they serve a unique purpose (for example, SAML- or JWT-based user authentication).

> **Note**
>
> It is also possible to store public keys (without a surrounding certificate) in the SAP HANA database. Public keys can be used to validate JSON Web Tokens (JWT) only.

The following figure shows for which purposes certificates stored in the database are possible:



In-Database Certificates

Optional In-database Storage

You **may** store and manage the certificates for the following purposes directly in the SAP HANA database:

- User authentication based on:
  - SAML assertions
  - Logon and assertion tickets
  - X.509 certificates for HTTP access via SAP HANA XS classic model

    > ### Note
    >
    > SAP HANA XS advanced also supports X.509 certificate-based user authentication, however, does not require a certificate collection in the SAP HANA database. For more information, see the *SAP HANA Administration Guide*.
- Client-server communication over JDBC/ODBC secured using TLS/SSL

> ### Note
>
> Although we recommend storing and managing these certificates in the database, files containing certificates may also be stored in the file system for the above purposes. See *Certificate Management in the File System*.

Mandatory In-database Storage

You **must** store and manage the certificates for the following purposes directly in the SAP HANA database:

- User authentication based:
  - JSON Web Tokens (JWT)
  - X.509 certificates for JDBC/ODBC client access
- Secure communication between the following components:
  - Two SAP HANA systems during the process of copying or moving a tenant database
  - SAP HANA and an LDAP server being used for user authentication and/or authorization
  - SAP HANA and remote data sources in an SAP HANA smart data access scenario

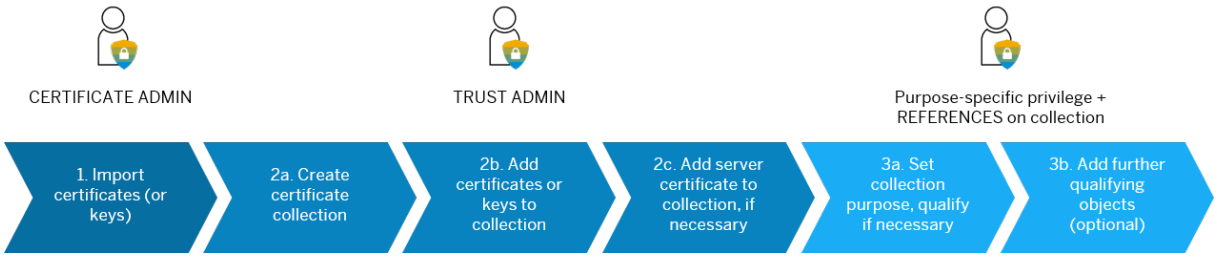
You can manage the certificates (and keys) required for trust validation in the SAP HANA cockpit.

Related Information

Certificate Management in the File System [page 318]
Maintaining Single Sign-On for XS Advanced Applications
Managing Client Certificates and Public Keys

13.1.1 In-Database Certificate Management Workflow

Managing certificates in the SAP HANA database follows a typical workflow. A full separation of duties is possible through user authorization. The full workflow is supported by the SAP HANA cockpit.



In-Database Certificate Management Workflow

1. Import certificates and keys

A user with `CERTIFICATE ADMIN` system privilege imports the certificates and public keys (JWT based authentication only) of trusted communication partners, as well as the root certificates of trusted Certification Authorities into the certificate store (or public key store).

2. Create certificate collection

A user with `TRUST ADMIN` system privilege:

1. Creates the required certificate collections.
2. Adds trusted certificates (or keys) from the certificate (or key) store to certificate collections.
3. Adds the SAP HANA server certificate(s) to those collections that will be used for server authentication (for example, secure client-server communication over JDBC/ODBC).

3. Set collection purpose and qualify, if necessary

Collections for user authentication purposes

A user with the `USER ADMIN` system privilege and object privilege `REFERENCES` on the collection:

1. Sets the purpose of individual collections with an optional qualification for one or more specific identity providers.
2. Optional, at a later point in time: Adds one or more identity providers.

> **Note**
>
> For collections with purpose `X509`, do not add an identity provider if the collection is intended for authenticating users accessing via the SAP HANA XS classic server.

Collection for secure client-server communication over JDBC/ODBC

A user with `SSL ADMIN` system privilege and `REFERENCES` object privilege on the collection:

1. Sets the purpose of the individual collection to SSL with an optional qualification for one or more hosts.
2. Optional, at a later point in time: Adds one or more host names.

Collections for secure communication to remote sources (SAP HANA smart data access)

A user with `CREATE REMOTE SOURCE` system privilege and object privilege `REFERENCES` on the collection:

1. Sets the purpose of the individual collection to REMOTE SOURCE with an optional qualification for one or more remote sources.
2. Optional, at a later point in time: Adds one or more remote sources.

Collection for secure communication between two SAP HANA systems during the process of copying or moving a tenant database

A user with `DATABASE ADMIN` system privilege and `REFERENCES` object privilege on the collection sets the purpose of the individual collection to DATABASE REPLICATION.

> ⓘ Note
>
> A certificate collection is required in both the source system and the target system.

Collection for secure communication between SAP HANA and an LDAP server being used for user authentication and/or authorization

A user with `LDAP ADMIN` system privilege and `REFERENCES` object privilege on the collection sets the purpose of the individual collection to LDAP.

> ⓘ Note
>
> If you have qualified the purpose of a collection by adding an identity provider, a remote source, or a host name, this collection takes precedence over a collection with the same purpose that has not been qualified.
>
> For example, if you have assigned your SAML provider SAML_PROVIDER to a certificate collection with the purpose SAML, only this collection is used to validate requests signed by SAML_PROVIDER. If you have other SAML providers defined in your system, requests from those providers are validated using the certificate collection with the purpose SAML that has not been qualified. If such a collection does not exist, then the corresponding PSE file in the file system is used for validation (by default, `sapsrv.pse`). For more information about changing the PSE file, see *Server-Side TLS/SSL Configuration Properties for External Communication (JDBC/ODBC)*.

13.1.2 Client Certificates

X.509 client certificates required for certificate-based user authentication and secure communication between SAP HANA and clients that access the SQL interface of the database can be stored and managed directly in the SAP HANA database.

Certificates stored in the SAP HANA database can be used for:

- Trust validation

Certificates used for trust validation are the public-key certificates of trusted communication partners or root certificates from trusted Certification Authorities. These certificates contain the public part of a user's or component's public and private key pair.

> **Note**
>
> It is also possible to store public keys (without a surrounding certificate) in SAP HANA. Public keys can be used to validate JSON Web Tokens (JWT) only.

- Server authentication
  Certificates used for server authentication are the public-key certificates of the SAP HANA server used to identify the server to connecting clients. In addition to the public-key information of the server, these certificates contain the server's private keys, as well as the intermediate certificates that complete the trust chain from the server certificate to the root certificate that the communication partner (client) trusts.

  > **Note**
  >
  > Private keys are stored securely using the internal application encryption service of the SAP HANA database. For more information, see the section on the internal application encryption service.

Once they have been imported into the database, certificates can be assigned to certificate collections. Certificate collections are also created and managed directly in the database, where they serve a unique purpose.

> **Note**
>
> Although we recommend creating and managing both certificates and certificate collections in the database, files containing certificates may also be stored in the file system in some cases.

Related Information

Internal Application Encryption Service [page 249]
CERTIFICATES System View
CREATE CERTIFICATE Statement (System Management)
CREATE PUBLIC KEY Statement (System Management)

13.1.3 Certificate Collections

A certificate collection is a secure location where the public information (public-key certificates) and private information (private keys) of the SAP HANA server are stored. A certificate collection may also contain the public information (public-key certificates or public keys) of trusted communication partners or root certificates from trusted Certification Authorities..

Certificate Collection Purposes

Certificate collections are created and managed as database objects directly in the SAP HANA database, where they uniquely serve one of the following purposes:

- User authentication based on:
  - SAML assertions (`SAML`)
  - JSON Web Tokens (`JWT`)
  - X.509 client certificates (`X509`)
  - Logon and assertion tickets (`SAP LOGON`)
- Secure communication between:
  - SAP HANA and external clients (`SSL`)

    > **Note**
    >
    > If the client public key infrastructure (PKI) is enabled, the purpose SSL is set for the automatically generated certificate collection `_SYS_CLIENTPKI`. As long as the client PKI is enabled, it is not possible to set the purpose SSL for any other collection. For more information about the client PKI, see *TLS/SSL Configuration on the SAP HANA Server*.

  - SAP HANA and an LDAP server being used for user authentication and/or authorization (`LDAP`)
  - SAP HANA and remote sources using SAP HANA smart data access (`REMOTE SOURCE`)
  - Two SAP HANA systems for the purposes of database replication when copying or moving a tenant database to another system (`DATABASE REPLICATION`)

The client certificates required for each purpose are assigned to the corresponding certificate collection from the in-database certificate store. A certificate can be assigned to more than one certificate collection.

> **Note**
>
> A certificate collection with the purpose `JWT` may also have public keys assigned.

> **Note**
>
> You only need to set your own key or a private key in collections used for client-server communication, that is with purpose `SSL`. Although it is possible to do so, there is no need to set a private key for certificate collections used for trust validation.

Certificate Collections with Qualified Purpose

Collections with the following purposes may be qualified further:

Purpose	Optional Qualification
SAML	Identity providers
JWT	Identity providers

Purpose	Optional Qualification
X509	Identity providers **Note** For authenticating user access via the SAP HANA XS classic server, do not add an identity provider to the collection.
REMOTE SOURCE	SAP HANA remote sources
SSL	Hosts

Qualifying the purpose of a collections allows you to assign the same purpose to several collections. In this way, the assertions, tokens and certificates issued by different providers can be validated separately, and the connections to different remote sources can be secured separately.

> **Note**
>
> Identity providers and remote sources must already exist in the database before they can be added to a collection, and each provider or remote source can only be assigned to one certificate collection.

Similarly, for client-server communication, you can assign the purpose `SSL` to several collections by adding host names to each collection. This allows you to configure separate sets of certificates for different hosts. Host names must only be entered in the correct format (for example, example.acme.com), and each host name can be assigned to several certificate collections.

If multiple collections with the same purpose exist, note the following behavior:

- Collections with qualified purposes take precedence over a collection with the same purpose that has not been qualified, that is a collection with the same purpose that has not been assigned any providers, remote sources or hosts.

  > **Note**
  >
  > Only one collection with an unqualified purpose may exist.

- If there is only one collection with a particular purpose and it is qualified, any unassigned providers, remote sources or host names that may be configured cannot be validated.

  > **Note**
  >
  > In the case of collections with purpose SAML and SSL, the trust store located on the file system and configured with the `global.ini` parameter `[communication] sslTrustStore` is used instead (`sapsrv.pse` by default).

This behavior can be illustrated with the following example.

> **Example**
>
> Multiple SAML identity providers are configured in your database (providerA, providerB, and providerC). Initially, two certificate collections are used for SAML-based user authentication. The first collection (pse1) is assigned the purpose `SAML` but no specific identity providers have been added. The second collection

(pse2) is also assigned the purpose SAML but is qualified for a specific identity provider (providerA). This is configured as follows:

```
SET PSE pse1 PURPOSE SAML;
SET PSE pse2 PURPOSE SAML FOR PROVIDER providerA;
```

With this configuration, collection pse2 has a qualified purpose and is used to validate requests signed by providerA only. Collection pse1 is not qualified with any specific providers and so is used to validate requests from all other providers configured in the database.

If you now drop collection pse1, requests from providerB and providerC can no longer be validated. This is because there is no collection with the unqualified purpose SAML, and these providers have not been added to any other collection with purpose SAML. This means that it is no longer possible to log on or connect to the database using providerB and providerC (assuming that `sapsrv.pse` does not contain the necessary certificates).

In addition to creating a new collection with the unqualified purpose SAML, you could resolve the situation in a number of ways:

- Create one or more additional collections with the purpose SAML and assign the other providers to these:

  Sample Code

  ```
  CREATE PSE pse3;
  SET PSE pse3 PURPOSE SAML FOR PROVIDER providerB;
  CREATE PSE pse4;
  SET PSE pse4 PURPOSE SAML FOR PROVIDER providerC;
  ```

  Collections pse3 and pse4 are now used to validate requests signed by providerB and providerC respectively.
- Assign the other providers to the existing collection with qualified SAML purpose (pse2):

  Sample Code

  ```
  ALTER PSE pse2 ADD PROVIDER providerB, providerC;
  ```

  pse2 is now used to validate requests signed by providerA, providerB, and providerC.
- Remove the provider assigned to the existing collection, thus making it applicable to all providers:

  Sample Code

  ```
  SET PSE pse2 PURPOSE SAML;
  ```

  pse2 is now used to validate requests signed by all providers.

Ownership of Certificate Collections

A certificate collection is a database object created in runtime. It is therefore owned by the database user who creates it. If a certificate collection is in use, in other words it has been assigned one of the above purposes, it

is not possible to change it (for example, add or remove certificates) or to delete it. However, if the owner of the certificate collection is deleted, the certificate collection will be deleted **even if it currently in use**.

> ⚠ Caution
>
> The deletion of a certificate collection that is assigned a purpose could render the database unusable. For example, if TLS/SSL is being enforced for all client connections and the certificate collection used for TLS/SSL is deleted, no new client connections to the database can be opened.

Related Information

SAP HANA Authentication and Single Sign-On [page 100]
Secure Communication Between SAP HANA and JDBC/ODBC Clients [page 42]
Secure Communication Between SAP HANA and an LDAP Directory Server [page 59]
Copying and Moving Tenant Databases
Create an SAP HANA On-Premise Remote Source
CREATE JWT PROVIDER Statement (Access Control)
CREATE SAML PROVIDER Statement (Access Control)
CREATE X509 PROVIDER (Access Control)
CREATE PSE Statement (System Management)
SET PSE Statement (System Management)

13.1.4 SQL Statements and Authorization for In-Database Certificate Management (Reference)

All administration tasks related to the management of public-key certificates (and public keys) can be performed using SQL.

This section lists the SQL statements for creating and managing certificates, public keys, and certificate collections in the SAP HANA database, including the required authorization for each task. For more detailed information about the syntax of the statements mentioned, see the *SAP HANA Database SQL Reference*.

> ⓘ Note
>
> Certificate collections are referred to as personal security environments (PSEs) in back-end terminology.

It is recommended that you use the SAP HANA cockpit to perform these tasks where possible.

To...	Execute the Statement...	With the Authorization...
See certificates or keys in the certificate or key store	• `SELECT * FROM CERTIFICATES` • `SELECT * FROM PUBLIC_KEYS`	System privilege CERTIFICATE ADMIN or TRUST ADMIN

To...	Execute the Statement...	With the Authorization...
See certificate collections	`SELECT * FROM PSES`	System privilege TRUST ADMIN If you have object privilege ALTER, DROP, or REFERENCES on a certificate collection or are the owner of the collection, you'll also be able to see this collection.
See which certificates or keys are used in a specific certificate collection	• `SELECT * FROM PSE_CERTIFICATES` • `SELECT * FROM PSE_PUBLIC_KEYS`	Object privilege ALTER, DROP, or REFERENCES on the certificate collection
See which certificate collections have qualified purposes	`SELECT * FROM PSE_PURPOSE_OBJECTS`	System privilege TRUST ADMIN If you have object privilege ALTER or REFERENCES on a certificate collection or are the owner of the collection, you'll also be able to see information about that collection.
View certificate collections in the database, including the certificates and keys they contain	• `SELECT * FROM PSE_CERTIFICATES` • `SELECT * FROM PSE_PUBLIC_KEYS`	System privilege CATALOG READ and TRUST ADMIN If you have object privilege ALTER or REFERENCES on a certificate collection or are the owner of the collection, you'll be able to see it without the above privileges.
Store a new public-key certificate or public key in the database	• `CREATE CERTIFICATE FROM <certificate_content> [ COMMENT <comment> ]` • `CREATE PUBLIC KEY <name> FROM '<pem>' [ KEY ID HINT '<hint>'] [ COMMENT '<comment>' ]`	System privilege CERTIFICATE ADMIN
Delete a public-key certificate or public key from the database **Note** If the certificate or key has already been added to a certificate collection, it can't be deleted.	• `DROP CERTIFICATE <certificate_id>` • `DROP PUBLIC KEY <public_key_name>`	System privilege CERTIFICATE ADMIN
Create a certificate collection	`CREATE PSE <PSE_name>`	System privilege TRUST ADMIN

To...	Execute the Statement...	With the Authorization...
Add (or remove) a public-key certificate or public key to (or from) a certificate collection	• `ALTER PSE <PSE_name> ADD|DROP CERTIFICATE <certificate_id>` • `ALTER PSE <pse_name> ADD|DROP PUBLIC KEY <public_key_name>`	If you are not the owner of the certificate collection, then you must have the object privilege ALTER on the certificate collection. **Note** If the purpose of the certificate collection has already been set, then the purpose-specific system privilege is additionally required (for example, USER ADMIN for user authentication purposes)
Add a private key to a certificate collection	`ALTER PSE <PSE_name> SET OWN CERTIFICATE <certificate_content>`	If you are not the owner of the certificate collection, then you must have the object privilege ALTER on the certificate collection.
Set the purpose of a certificate collection and qualify if necessary	`SET PSE <PSE_name> PURPOSE <purpose>` The following purposes are possible:	Purpose-specific privilege and if you are not the owner of the certificate collection, object privilege REFERENCES on the collection
	SAML You can qualify the purpose SAML for specific identity providers as follows: `SET PSE <PSE_name> PURPOSE SAML FOR PROVIDER <provider, ...>` **Note** A provider can only be used to qualify the purpose of one collection. If you use SET PSE to assign a provider that is already assigned to another collection, the provider is removed from the other collection. If this was the only provider assigned to the other collection, the purpose is also removed.	System privilege USER ADMIN

To...	Execute the Statement...	With the Authorization...
	JWT You can qualify the purpose JWT for specific identity providers as follows: `SET PSE <PSE_name> PURPOSE JWT FOR PROVIDER <provider, ...>` **Note** A provider can only be used to qualify the purpose of one collection. If you use SET PSE to assign a provider that is already assigned to another collection, the provider is removed from the other collection. If this was the only provider assigned to the other collection, the purpose is also removed.	System privilege USER ADMIN
	X509 If the collection will be used for authenticating users accessing SAP HANA via JDBC/ODBC clients, you can qualify the purpose X509 for specific identity providers as follows: `SET PSE <PSE_name> PURPOSE X509 FOR PROVIDER <provider, ...>` **Note** A provider can only be used to qualify the purpose of one collection. If you use SET PSE to assign a provider that is already assigned to another collection, the provider is removed from the other collection. If this was the only provider assigned to the other collection, the purpose is also removed. If the collection will be used for authenticating users accessing via the SAP HANA XS classic server, do not add an identity provider.	System privilege USER ADMIN
	SAP LOGON	System privilege USER ADMIN

To...	Execute the Statement...	With the Authorization...
	SSL You can qualify the purpose SSL for specific host names as follows: `SET PSE <PSE_name> PURPOSE SSL FOR HOST <host_name, ...>`. **Note** The same host name can be used to qualify the purpose of multiple collections. **Note** You can only assign the purpose SSL if a private key has already been added. **Note** If the client public key infrastructure (PKI) is enabled, the purpose SSL is set for the automatically generated certificate collection `_SYS_CLIENTPKI`. As long as the client PKI is enabled, it is not possible to set the purpose SSL for any other collection. For more information about the client PKI, see *TLS/SSL Configuration on the SAP HANA Server*.	System privilege SSL ADMIN
	DATABASE REPLICATION	System privilege DATABASE ADMIN
	LDAP	System privilege LDAP ADMIN

To...	Execute the Statement...	With the Authorization...
	REMOTE SOURCE You can qualify the purpose REMOTE SOURCE for specific remote sources as follows: `SET PSE <PSE_name> PURPOSE REMOTE SOURCE FOR REMOTE SOURCE <remote_source, ...>` **Note** A remote source can only be used to qualify the purpose of one collection. If you use SET PSE to assign a remote source that is already assigned to another collection, the remote source is removed from the other collection. If this was the only remote source assigned to the other collection, the purpose is also removed.	System privilege CREATE REMOTE SOURCE
Unset the purpose of a certificate collection	`UNSET PSE <PSE_name> PURPOSE <PSE_purpose>` **Note** If the collection has identity providers, remote sources or host names assigned, these are also removed from the collection.	Purpose-specific privilege and if you are not the owner of the certificate collection, object privilege REFERENCES on the collection

To...	Execute the Statement...	With the Authorization...
Add an identity provider, remote source or host name to a certificate collection	• `ALTER PSE <PSE_name> ADD PROVIDER <identity_provider>` • `ALTER PSE <PSE_name> ADD REMOTE SOURCE <remote_source>` • `ALTER PSE <PSE_name> ADD HOST <host_name>` **Note** You can only use this statement to add further providers, remote sources or host names to a collection. In other words, the collection must have at least one provider, remote source or host name already added. To add the first provider, remote source or host name, use the SET PSE statement.	Purpose-specific privilege and if you are not the owner of the certificate collection, object privilege REFERENCES on the collection
Remove an identity provider, remote source or host name from a certificate collection	• `ALTER PSE <PSE_name> DROP PROVIDER <identity_provider>` • `ALTER PSE <PSE_name> DROP REMOTE SOURCE <remote_source>` • `ALTER PSE <PSE_name> DROP HOST <host_name>` **Note** You cannot use this statement to remove the last provider, remote source or host name assigned to a collection. To remove the last provider, remote source or host name, use the UNSET PSE statement.	Purpose-specific privilege and if you are not the owner of the certificate collection, object privilege REFERENCES on the collection
See which certificate collections have qualified purposes	`SELECT * FROM PSE_PURPOSE_OBJECTS`	System privilege TRUST ADMIN If you have object privilege ALTER or REFERNCES on a certificate collection or are the owner of the collection, you'll also be able to see information about that collection.

To...	Execute the Statement...	With the Authorization...
Delete a certificate collection	`DROP PSE <PSE_name>` **Note** If the certificate collection has already been assigned a purpose, it can't be deleted.	If you are not the owner of the certificate collection, then you must have the object privilege DROP on the collection.

Related Information

CREATE CERTIFICATE Statement (System Management)
DROP CERTIFICATE Statement (System Management)
CREATE PUBLIC KEY Statement (System Management)
DROP PUBLIC KEY Statement (System Management)
CREATE PSE Statement (System Management)
SET PSE Statement (System Management)
ALTER PSE Statement (System Management)
UNSET PSE Statement (System Management)
DROP PSE Statement (System Management)
CERTIFICATES System View
PSES System View
PSE_CERTIFICATES System View
PSE_PURPOSE_OBJECTS System View
PSE_PUBLIC_KEYS System View
PUBLIC_KEYS System View
Managing Client Certificates and Public Keys

13.2 Certificate Management in the File System

The X.509 client certificates used for securing internal and external communication channels, as well as for some user authentication mechanisms may be stored and managed in trust and key stores located in the file system.

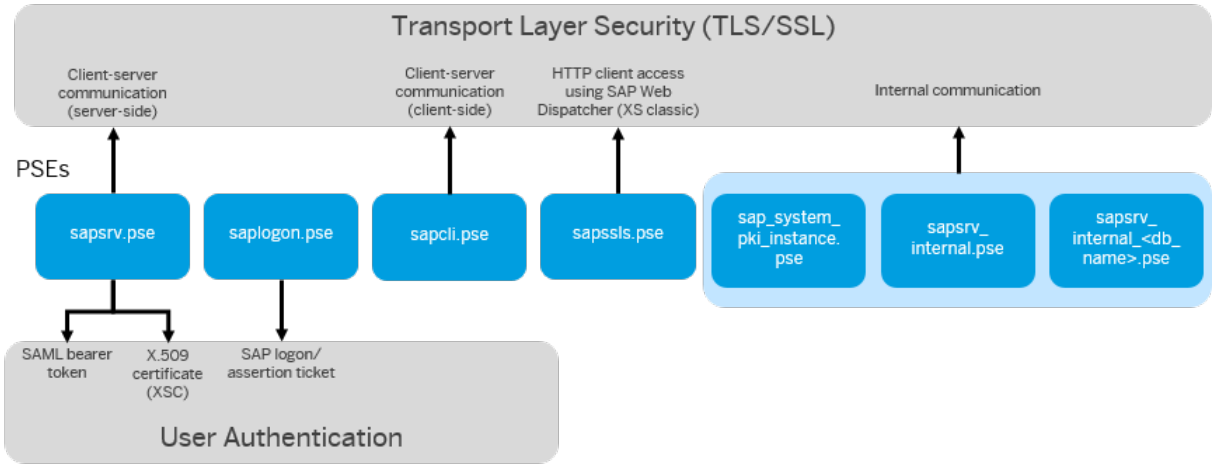
Note

We usually recommend storing certificates directly in the database. However, there may be situations where using file system-based certificate stores, also known as personal security environments (PSEs), can be better. For instance, if you are recovering backups of a database instance onto different hosts, using file system-based PSEs can make it easier to manage TLS/SSL certificate settings.

The following figure shows for which purposes certificates stored in PSEs in the file system are possible. These PSEs can be managed using for example the SAP Web Dispatcher administration tool or the SAPGENPSE tool,

both of which are delivered with SAP HANA. If you are using OpenSSL, you can also use the tools provided with OpenSSL.

> **Note**
>
> OpenSSL is deprecated. If you are using OpenSSL, migrate to CommonCryptoLib. For more information, see SAP Note 2093286.



Default File-Based PSEs

> **Caution**
>
> By default, the same PSE in the file system is shared by all databases for all external communication channels (including HTTP) and certificate-based authentication. Different PSEs must be explicitly configured for tenant databases.

> **→ Recommendation**
>
> You can migrate certificates from file-system based storage to in-database storage. If you do migrate certificates in the file system to the database, delete all related files from the file system to avoid any potential conflicts. For more information, see SAP Note 2175664.

Mandatory File System Storage

Not all certificates can be stored in the database, in particular the certificates required to secure internal communication channels using the system public key infrastructure (system PKI), and HTTP client access (SAP HANA XS, classic model) using SAP Web Dispatcher. These certificates are contained in PSE files located in the file system.

> **Caution**
>
> Do not delete these PSE files from the file system.

Related Information

Secure Internal Communication [page 64]
Maintaining HTTP Access to SAP HANA
SAP Note 2175664
SAP Note 2093286

14 Data Protection and Privacy in SAP HANA

SAP HANA provides the technical enablement and infrastructure to allow you run applications on SAP HANA to conform to the legal requirements of data protection in the different scenarios in which SAP HANA is used.

Introduction to Data Protection

Data protection is associated with numerous legal requirements and privacy concerns. In addition to compliance with general data privacy regulations, it is necessary to consider compliance with industry-specific legislation in different countries. SAP HANA provides specific features and functions to support compliance with regard to relevant legal requirements, including data protection.

This section and any other sections in this Security Guide do not give any advice on whether these features and functions are the best method to support company, industry, regional, or country-specific requirements. Furthermore, this guide does not give any advice or recommendations regarding additional features that would be required in specific IT environments; decisions related to data protection must be made on a case-by-case basis, taking into consideration the given system landscape and the applicable legal requirements.

> ⓘ Note
>
> In the majority of cases, compliance with data privacy laws is not a product feature. SAP software supports data privacy by providing security features and specific functions relevant to data protection, such as functions for the simplified blocking and deletion of personal data. SAP does not provide legal advice in any form. The definitions and other terms used in this guide are not taken from any given legal source.

Glossary

Term	Definition
Blocking	A method of restricting access to data for which the primary business purpose has ended.
Business purpose	A legal, contractual, or in other form justified reason for the processing of personal data. The assumption is that any purpose has an end that is usually already defined when the purpose starts.
Consent	The action of the data subject confirming that the usage of his or her personal data shall be allowed for a given purpose. A consent functionality allows the storage of a consent record in relation to a specific purpose and shows if a data subject has granted, withdrawn, or denied consent.
Deletion	Deletion of **personal data** so that the data is no longer available.
End of purpose (EoP)	End of purpose and start of blocking period. The point in time, when the primary processing purpose ends (e.g. contract is fulfilled).

Term	Definition
End of purpose (EoP) check	A method of identifying the point in time for a data set when the processing of **personal data** is no longer required for the primary **business purpose**. After the **EoP** has been reached, the data is **blocked** and can only be accessed by users with special authorization (for example, tax auditors).
Personal data	Any information relating to an identified or identifiable natural person ("data subject"). An identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural, or social identity of that natural person
Residence period	The period of time between the end of business and the end of purpose (EoP) for a data set during which the data remains in the database and can be used in case of subsequent processes related to the original purpose. At the end of the longest configured residence period, the data is blocked or deleted. The residence period is part of the overall retention period.
Retention period	The period of time between the end of the last business activity involving a specific object (for example, a business partner) and the deletion of the corresponding data, subject to applicable laws. The retention period is a combination of the residence period and the blocking period.
Sensitive personal data	A category of personal data that usually includes the following type of information: • Special categories of personal data, such as data revealing racial or ethnic origin, political opinions, religious or philosophical beliefs, trade union membership, genetic data, biometric data, data concerning health or sex life or sexual orientation, or personal data concerning bank and credit accounts. • Personal data subject to professional secrecy • Personal data relating to criminal or administrative offenses • Personal data concerning insurances and bank or credit card accounts
Where-used check (WUC)	A process designed to ensure data integrity in the case of potential blocking of business partner data. An application's where-used check (WUC) determines if there is any dependent data for a certain business partner in the database. If dependent data exists, this means the data is still required for business activities. Therefore, the blocking of business partners referenced in the data is prevented.

SAP HANA Approach to Data Protection

Many data protection requirements depend on how the business semantics or context of the data stored and processed in SAP HANA are understood.

> ⓘ Note
>
> Using capabilities to communicate with other data sources, SAP HANA may also be used to process data that is stored in other systems and accessed through virtual tables.

In SAP HANA installations, the business semantics of data are part of the application definition and implementation. SAP HANA provides the features for working with technical database objects, such as tables.

It is therefore the application that "knows", for example, which tables in the database contain sensitive personal data, or how business level objects, such as sales orders, are mapped to technical objects in the database. Applications built on top of SAP HANA need to make use of features provided by SAP HANA to implement compliance requirements for their specific use case.

> **Note**
>
> This is also true for SAP HANA installations that include SAP HANA dynamic tiering.

SAP HANA provides a variety of security-related features to implement general security requirements that are also required for data protection and privacy:

Aspect of Data Protection and Privacy	SAP HANA Feature	More Information

Access control	Several features in SAP HANA provide access control:
	• Authentication
	• Authorization
	• Data masking
	• Data anonymization
	• Data encryption:
	  • Data volume encryption
	  • Redo log encryption
	  • Backup encryption
	  • Client-side column encryption
		• Section SAP HANA Authentication and Single Sign-On [page 100]
		• Section SAP HANA Authorization [page 135]
		• Section SAP HANA Data Masking [page 215]
		• Section Data Storage Security in SAP HANA [page 233]

Access logging	Audit logging	Section Auditing Activity in SAP HANA [page 277]
Transmission control/communication security	Support for encrypted communication on all internal and external channels	Section SAP HANA Network and Communication Security [page 34]
Availability control	Several features in SAP HANA provide availability control: • Backup and recovery • Storage replication • System replication • Service auto-restart • Host auto-failover	Section on availability and scalability in the *SAP HANA Administration Guide*
Separation by purpose	Separation by purpose is subject to the organizational model implemented and must be applied as part of the authorization concept. Isolated data storage can be achieved in SAP HANA using: • Database schemas protected using authorization • Tenant databases	Section Technical System Landscape [page 18]

> ⚠ Caution
>
> Database trace and dump files may potentially expose personal data, for example, a trace set to a very high trace level such as DEBUG. In addition, if you are using the capture and replay feature, captured and preprocessed workload files may also contain personal data since they contain input parameters for SQL statement execution. For more information, see the section on the security risks of trace, dump, and captured workload files.

> ⚠ Caution
>
> The extent to which data protection is ensured depends on secure system operation. Network security, security note implementation, adequate logging of system changes, and appropriate usage of the system are the basic technical requirements for compliance with data privacy legislation and other legislation.

Related Information

SAP HANA Implementation Scenarios [page 26]
Security Risks of Trace, Dump, and Captured Workload Files [page 326]
High Availability for SAP HANA
SAP HANA Dynamic Tiering
Data Protection and Privacy in SAP HANA XS Advanced [page 387]
Data Protection and Privacy in SAP HANA Cockpit
Data Protection in SAP HANA Database Explorer

14.1 Deletion of Personal Data

SAP HANA supports the deletion of data in tables using SQL deletion commands. Applications running on SAP HANA must make use of such commands to implement deletion requirements of personal data.

End of purpose checks, including implementation of legally required retention periods and data blocking, are managed by the application. Applications can implement data blocking using SAP HANA mechanisms such as authorization and table creation. For example, an application could transfer blocked data to separate database tables that are protected by special authorizations.

Once data has been deleted, the delete operation cannot be undone using SQL statements.

For more information about permanent deletion of data in system-versioned tables, please refer to 'Deleting Data' in the System-Versioned Tables section of the *SAP HANA Administration Guide*.

> ⓘ Note
>
> Following standard practice, deletion of personal data is not enforced in backups. Common practice is that deleted data disappears from backups following typical backup-rotation mechanisms. SAP HANA supports backup lifecycle management by providing functions for deleting backups according to time stamps.

Related Information

System-Versioned Tables

15 Security Risks of Trace, Dump, and Captured Workload Files

In exceptional situations, the data output in trace and dump files may expose certain security-relevant data. If you are using the capture and replay feature, captured and preprocessed workload files may also contain personal data.

Trace and Dump Files

Trace files are used to troubleshoot problems in the SAP HANA database. Dump files containing useful information for error analysis may also be created. Under normal circumstances, security-relevant data is not written to the files. However, if the default configuration is changed, for example when a trace is activated with a high trace level in a support situation, query strings including WHERE clause restrictions are written to trace files, for example, the database trace file of the index server. Query result sets and information about users may be output.

> **Note**
>
> Passwords are never output.

The following files may contain security-relevant data:

- Trace files generated through the activation of the following trace types:
  - SQL trace
  - Database trace, including user-specific and end-to-end traces
  - Expensive statement trace
  - Performance trace
- Dump files
  - Core dump files (for example, crash dump files)
    The system generates these files automatically.
  - Runtime dump files
    The generation of these files can be triggered using the command line tool `hdbcons`.

Captured and Preprocessed Workload Files

SAP HANA capture and replay allows you to capture the workload of a production system and to replay the captured workload on a target system. Workload is any change executed on the database using SQL.

During both the capturing process and the preprocessing step required to make a workload replayable, files are written to the file system (by default to the trace directory `$DIR_INSTANCE/<host name>/trace`). These files contain input parameters for SQL statement execution, which may contain personal data:

- Captured workload files (*.cpt)
- Preprocessed workload files (files in the sub-directory `<SID>_<capture_id>`)

> **Note**
>
> The system privileges WORKLOAD CAPTURE ADMIN and WORKLOAD REPLAY ADMIN are required to capture workloads and preprocess captured workloads respectively.

Related Information

Diagnostic Files and Logs
Capturing and Replaying Workloads

16 Security of Further SAP HANA Components and Capabilities

Security information for some SAP HANA components, as well additional capabilities that may be installed in the SAP HANA system, is available separately.

For security information about...	See...
SAP HANA platform lifecycle management	Security Aspects of SAP HANA Platform Lifecycle Management [page 329]
SAP HANA auto content	Security of SAP HANA Content [page 330]
SAP HANA cockpit	Security Considerations for SAP HANA Cockpit
SAP HANA database explorer	Security in the SAP HANA Database Explorer
SAP HANA extended application services, advanced model	Security for SAP HANA Extended Application Services, Advanced Model [page 336]
SAP Web IDE for SAP HANA	Security Aspects of SAP Web IDE for SAP HANA [page 394]
SAP HANA smart data access	Security Aspects of SAP HANA Smart Data Access [page 331]
File Loader	File Loader Guide for SAP HANA
SAP File Processing	SAP File Processing for SAP HANA
SAP HANA R integration	Security Aspects of SAP HANA R Integration [page 332]
SAP HANA Hadoop integration	SAP HANA Spark Controller Installation Guide
SAP Enterprise Architecture Designer	SAP Enterprise Architecture Designer, Edition for SAP HANA Security Guide
SAP HANA accelerator for SAP ASE	SAP HANA Accelerator for SAP ASE: Administration Guide
SAP HANA data warehousing foundation	SAP HANA data warehousing foundation administration guides available on SAP Help Portal at SAP HANA Data Warehousing Foundation
SAP HANA smart data integration and SAP HANA smart data quality	Security for SAP HANA Replication Technologies [page 333] and the Installation and Configuration Guide for SAP HANA Smart Data Integration and SAP HANA Smart Data Quality
SAP HANA streaming analytics	SAP HANA Streaming Analytics: Security Guide
SAP HANA real-time replication with SAP Landscape Transformation Replication Server	Security for SAP HANA Replication Technologies [page 333] and SAP Landscape Transformation Replication Server Security Guide

Related Information

Important Disclaimer for Features in SAP HANA [page 440]

16.1 Security Aspects of SAP HANA Platform Lifecycle Management

Security information for SAP HANA platform lifecycle management

The SAP HANA database lifecycle manager (HDBLCM) is used to install, configure, and update the components of SAP HANA. The components of SAP HANA can only be installed by certified hardware partners, or any person holding the required certification on validated hardware running an approved operating system.

Before the installation and update of SAP HANA software components, the authenticity and integrity of the software should be verified. For more information about how to do this, see the *SAP HANA Server Installation and Update Guide*.

During the installation process, the initial passwords of a number of standard users are specified. Once you receive SAP HANA, we recommend that you change these initial passwords. If you are changing system identifiers (host name, SID, or instance number), it is possible to change the system administrator (`<sid>`adm) password and the password of the SYSTEM database user of the system database at the same time.

> ⚠ Caution
>
> Do not use the `SYSTEM` user for day-to-day activities. Instead, use this user to create dedicated database users for administrative tasks and to assign privileges to these users. It is recommended that you then deactivate the `SYSTEM` user.

> ⓘ Note
>
> The `SYSTEM` user is not required to update the SAP HANA database system; a lesser-privileged user can be created for this purpose. However, to upgrade SAP support package stacks, SAP enhancement packages and SAP systems using the Software Update Manager (SUM) and to install, migrate, and provision SAP systems using the Software Provisioning Manager (SWPM), the `SYSTEM` user **is required** and needs to be temporarily reactivated for the duration of the upgrade, installation, migration or provisioning.

SAP HANA platform lifecycle management tasks can be performed on multiple-host SAP HANA systems centrally, by running the SAP HANA database lifecycle manager (HDBLCM) from any worker host and using remote execution to replicate the call on all remaining SAP HANA system hosts. Otherwise, the platform LCM tasks can be executed first on a worker host, and then re-executed manually on each remaining host. This method is considered decentralized execution.

Related Information

Software Authenticity Verification
Predefined Database Users [page 89]
Renaming a System
Executing Platform LCM Tasks
Deactivate the SYSTEM User [page 97]
Recent changes in the SAP HANA Technology certification program 2016

16.2 Security of SAP HANA Content

SAP HANA is delivered with a set of preinstalled software components implemented as SAP HANA Web applications, libraries, and configuration data. These components are developed on SAP HANA Extended Services (SAP HANA XS), classic model, and together with other configuration components are referred to as SAP HANA content.

> ⓘ Note
>
> SAP HANA XS, classic and the SAP HANA repository are deprecated as of SAP HANA 2.0 SPS 02. For more information, see SAP Note 2465027.

Software components delivered as SAP HANA content are an integral part of the SAP HANA platform. They provide essential features for Web-based configuration, administration and monitoring, application lifecycle management, and supportability.

Installation and Update

SAP HANA content is contained in delivery units (DUs). DUs containing automated content are deployed after the core SAP HANA database engine is started up during platform installation or upgrade and every time a new logical SAP HANA database is created. During an upgrade of an SAP HANA platform instance, the software components are updated to the version residing on the installation medium. DUs containing non-automated content need to be manually imported into the SAP HANA repository by a system administrator. DUs containing non-automated content are also automatically updated during an upgrade of an SAP HANA platform instance. For more information importing DUs, see *Deploy a Delivery Unit Archive (*.tgz)* in the *SAP HANA Master Guide*.

Content Security

Several software components available as SAP HANA content are Web applications and are therefore intended to be accessed by users through a Web browser. Only authenticated SAP HANA database users who have been explicitly authorized to use these software components by a user administrator can access them from their Web browser. The privileges required to use a software component are contained within roles delivered with the component itself. No user has these roles initially, except the user `_SYS_REPO` (as the owner of all repository content).

> → Recommendation
>
> As repository roles delivered with SAP HANA can change when a new version of the package is deployed, either do not use them directly but instead as a template for creating your own roles, or have a regular review process in place to verify that they still contain only privileges that are in line with your organization's security policy. Furthermore, if repository package privileges are granted by a role, we recommend that these privileges be restricted to your organization's packages rather than the complete repository. To do this, for each package privilege (`REPO.*`) that occurs in a role template and is granted

on `.REPO_PACKAGE_ROOT`, check whether the privilege can and should be granted to a single package or a small number of specific packages rather than the full repository.

Users are authenticated and authorization checks are performed by the standard authentication and authorization mechanisms implemented by SAP HANA XS classic.

It is therefore guaranteed that no functionality is provided to or exposed to any user after a plain installation or upgrade.

More Information

For a list of all software components installed as SAP HANA content, including a detailed description of their purpose and functional scope, see the section *Components Delivered as SAP HANA Content*. The roles required to use each component are also listed with information about which functionality is made available by which role.

Related Information

Components Delivered as SAP HANA Content [page 420]
Deploy a Delivery Unit Archive (*.tgz)
SAP Note 2465027

16.3 Security Aspects of SAP HANA Smart Data Access

Security information for SAP HANA smart data access

SAP HANA smart data access makes it possible to connect remote data sources and to present the data contained in these data sources as if from local SAP HANA tables. This can be used, for example, in SAP Business Warehouse installations running on SAP HANA to integrate data from remote data sources.

In SAP HANA, virtual tables are created to represent the tables in the remote data source. Using these virtual tables, joins can be executed between tables in SAP HANA and tables in the remote data source. The linked database feature allows DML queries on remote data sources without the need to first create virtual tables.

Connections to the remote data source can be authenticated as follows:

- By one technical user credential
  In this case, all connections to the remote data source share one and the same credential for the data source.
- By multiple secondary SAP HANA user-specific credentials
  In this case, there is one credential per user per data source.
- By a Kerberos SSO credential
  In this case, connections to the remote source (SAP HANA remote sources only) are authenticated through Kerberos single sign-on (SSO).

- By a JSON Web Token (JWT) SSO credential for smart data access (SDA)
  In this case, connections to the remote source (SAP HANA remote sources only) are authenticated through JWT single sign-on (SSO).

All credentials are stored securely in SAP HANA's internal credential store.

Authorization to access data in the remote data source is determined by the privileges of the database user as standard. In SAP Business Warehouse (BW) scenarios, authorization is applied in the BW layer.

The following privileges are required to manage remote sources and virtual tables:

To...	You Need...
Manage remote sources	System privileges CREATE REMOTE SOURCE, CREDENTIAL ADMIN
To create and manage virtual tables on a remote source	• Object privilege CREATE VIRTUAL TABLE • Additional object level privileges (for example, INSERT, UPDATE, DELETE) for virtual tables owned by others
Use the linked database feature	System privilege LINKED DATABASE

Related Information

Secure Internal Credential Store [page 250]
SAP HANA Smart Data Access
SAP Note 2303807

16.4 Security Aspects of SAP HANA R Integration

Security information for the integration of the SAP HANA database with R

R is an open source programming language and software environment for statistical computing and graphics. The integration of the SAP HANA database with R makes it is possible to embed R code in the SAP HANA database context.

R and SAP HANA

SAP does not ship the R environment with the SAP HANA database, as R is open source and is available under the General Public License. SAP does not provide support for R. In order to use the SAP HANA integration with R, you need to download R from the open-source community and configure it. You also need Rserve, a TCP/IP server that allows other programs to use facilities of R without the need to initialize R or link against R library. For more information, see the *SAP HANA R Integration Guide*.

Security Considerations

Users require additional privileges to execute R procedures. To ensure that only authorized users and programs can connect to Rserve, SAP also recommends implementing user authentication for calls from SAP HANA to Rserve. For more information, see *Security for R* in the *SAP HANA R Integration Guide*.

Secure Communication with SAP HANA

SAP recommends securing the communication channel between SAP HANA and the Rserve server using the Transport Layer Security (TLS)/Secure Sockets Layer (SSL) protocol. In this scenario, the SAP HANA server is the SSL client and the Rserve server is the SSL server. Configuration is required on both the SAP HANA server and the Rserve server.

On the SAP HANA side, the parameters for secure external client communication in the `global.ini` file apply, and it is not necessary to configure any of these parameters explicitly.

This communication channel only supports the use of a truststore stored in the file system. Therefore, you import the certificate of the Rserve server into the trust store specified by the parameter `[communication] sslTrustStore`, that is `sapsrv.pse`.

For more information about the configuration on the Rserve side and establishing communication via an SSL/TLS connection, see *Set Up SSL/TLS from SAP HANA to Rserve* in the *SAP HANA R Integration Guide*.

Related Information

Set Up SSL/TLS from SAP HANA to Rserve
Security for R
Server-Side TLS/SSL Configuration Properties for External Communication (JDBC/ODBC) [page 48]
SAP HANA R Integration Guide

16.5 Security for SAP HANA Replication Technologies

SAP HANA supports several replication technologies. Security features and considerations depend on the implemented technology.

SAP HANA Extraction-Transformation-Load (ETL) Data Services

The SAP HANA Extraction-Transformation-Load (ETL) data replication technology uses SAP Data Services (hereafter referred to as Data Services) to load the relevant business data from the source system (for example, SAP ERP) and replicate it to the target SAP HANA database. This method allows you to read the

required business data at the application layer level. You deploy this method by defining data flows in Data Services and scheduling the replication jobs.

Since this method uses batch processing, it also enables data checks, transformations, synchronization with additional data providers, and the merging of data streams. The main components are the Data Services Designer, where you model the data flow, and the Data Services Job Server for the execution of the replication jobs. An additional repository is used to store the metadata and the job definitions.

Data Services relies on the Central Management Server (CMS) for authentication and security features. For information about the security features provided by the CMS, see the *SAP Data Services Administrator Guide* or the *Information platform services Administrator Guide*.

To ensure security for your Data Services environment, use a firewall to prevent unintended remote access to administrative functions. In a distributed installation, you need to configure your firewall so that the Data Services components are able to communicate with each other as needed. For information about configuring ports on your firewall, see your firewall documentation.

For more information about ETL data replication technology using the SAP Data Services database, see the security section in the *SAP Data Services Administrator Guide*.

Trigger-Based Data Replication using SAP LT (Landscape Transformation) Replication Server (SLT)

SAP Landscape Transformation replication server is a replication technology that provisions data from SAP systems to an SAP HANA environment.

When using a distributed system, you need to ensure that your data and processes support your business needs without allowing unauthorized access to critical information. User errors, negligence, or attempted manipulation of your system should not result in loss of information or processing time. These demands on security apply likewise to the trigger-based data replication using the SAP LT replication server.

The SAP LT replication server and the SAP source system use the user management and authentication mechanisms provided by the SAP NetWeaver platform, in particular the SAP NetWeaver Application Server. Therefore, the security recommendations and guidelines for user administration and authentication as described in the *SAP NetWeaver Application Server ABAP Security Guide* also apply to the SAP LT Replication Server and an SAP source system.

The SAP LT replication server and the SAP source system use the authorization concept provided by the SAP NetWeaver AS ABAP. Therefore, the recommendations and guidelines for authorizations as described in the*SAP NetWeaver Application Server ABAP Security Guide* also apply to the SAP LT replication server. In SAP NetWeaver, authorizations are assigned to users based on roles. For role maintenance, use the profile generator (transaction PFCG) on the AS ABAP.

SAP HANA Smart Data Integration

SAP HANA smart data integration is data provisioning technology that allows real-time change data capture and batch loading from any source into SAP HANA. Because smart data integration uses a Data Provisioning Agent that is installed on a separate system than the SAP HANA system to manage adapters that link a

source to SAP HANA, care must be taken to ensure secure connections. Security recommendations, as well as guidelines for user administration and authentication, are described in the *Installation and Configuration Guide for SAP HANA Smart Data Integration and SAP HANA Smart Data Quality*.

Related Information

SAP Data Services on SAP Help Portal
SAP Real-Time Replication on SAP Help Portal
SAP NetWeaver on SAP Help Portal
SAP HANA Smart Data Integration and SAP HANA Smart Data Quality on SAP Help Portal

17 Security for SAP HANA Extended Application Services, Advanced Model

An overview of the tools used to configure and ensure security in the XS advanced model platform.

This section of the *SAP HANA Security Guide* describes the security aspects of the SAP HANA XS advanced model's server infrastructure. As an application platform, SAP HANA extended application services, advanced model, provides a comprehensive run-time environment, in which deployed applications may be run in a secure manner. Application developers are encouraged to make use of the available platform services such as the identity provider to protect critical data from unauthorized access.

> ⓘ Note
>
> SAP HANA XS advanced is based on the SAP HANA platform, so information in other sections of the *SAP HANA Security Guide* also applies.

> → Recommendation
>
> SAP recommends that customers and partners who want to develop new applications use SAP HANA XS, advanced model. If you want to migrate existing XS classic applications to run in the new XS advanced run-time environment, SAP recommends that you first check the features available with the installed version of XS advanced; if the XS advanced features match the requirements of the XS classic application that you want to migrate, then you can start the migration process. For more information, see the *SAP HANA XS Advanced Migration Guide*.

Additional Documentation Resources

Further SAP HANA Guides

SAP HANA Guide	Comment
SAP HANA Security Checklists and Recommendations	Overview of recommendations to help you operate and configure the SAP HANA XS advanced model run time securely
SAP HANA Administration Guide	How to manage the various components of the SAP HANA XS advanced model run time
SAP HANA Developer Guide for XS Advanced Model	How to build and deploy applications to run in the SAP HANA XS advanced model run time
SAP HANA XS Advanced Migration Guide	How to migrate applications built for SAP HANA XS, classic model, to run in the SAP HANA XS, advanced model run-time environment

Important SAP Notes

The following table lists important SAP Notes that apply to the security of SAP HANA XS advanced:

SAP Note	Title	Comment
2243019	Providing SSL certificates for domains defined in SAP HANA extended application services, advanced model	How to upload certificates for domains defined in SAP HANA XS advanced
2245631	Domains and routing configuration for SAP HANA extended application services, advanced model	• How to enable hostname routing for SAP HANA XS advanced • How to expose only a single port for all applications and services
2300943	Enabling JDBC SSL encryption for SAP HANA extended application services, advanced model	How to enable JDBC SSL encryption for SAP HANA XS advanced applications or services
2243156	Secure user setup for SAP HANA extended application services, advanced model	How to set up SAP HANA XS advanced to run application processes and staging processes as different operating-system (OS) users on Unix

For a complete list of additional security-relevant SAP Hot News and SAP Notes, see also SAP Support Portal at https://launchpad.support.sap.com/#/securitynotes.

Downloading XS Advanced from SAP Marketplace

SAP HANA Extended Application Services, advanced model, is available not only on the SAP HANA media but also as a separate component on SAP Marketplace. Users with the required S-User ID can download the latest version of XS advanced component in the package `SAP EXTENDED APP SERVICES 1` from the following location:

Service Marketplace > *Software Downloads [Downloads]* > *SUPPORT PACKAGES & PATCHES* > *By Alphabetical Index (A-Z)* > *H* > *SAP HANA PLATFORM EDITION* >:

- *SAP HANA PLATFORM EDITION 2.0* > *SAP EXTENDED APP SERVICES 1* >

> → Tip
>
> SAP HANA Extended Application Services, advanced model, is backwards compatible; you can provide access to new features by installing the latest version of the XS advanced component even on older versions of SAP HANA. To download the package `SAP EXTENDED APP SERVICES 1`, see *SAP Software Download Center* in *Related Information* below.

Related Information

SAP Software Download Center (Logon required)

17.1 Technical System Landscape of SAP HANA XS Advanced

SAP HANA extended application services, advanced model (XS advanced for short) provides a comprehensive platform for the development and execution of micro-service oriented applications, taking advantage of SAP HANA's in-memory architecture and parallel execution capabilities.

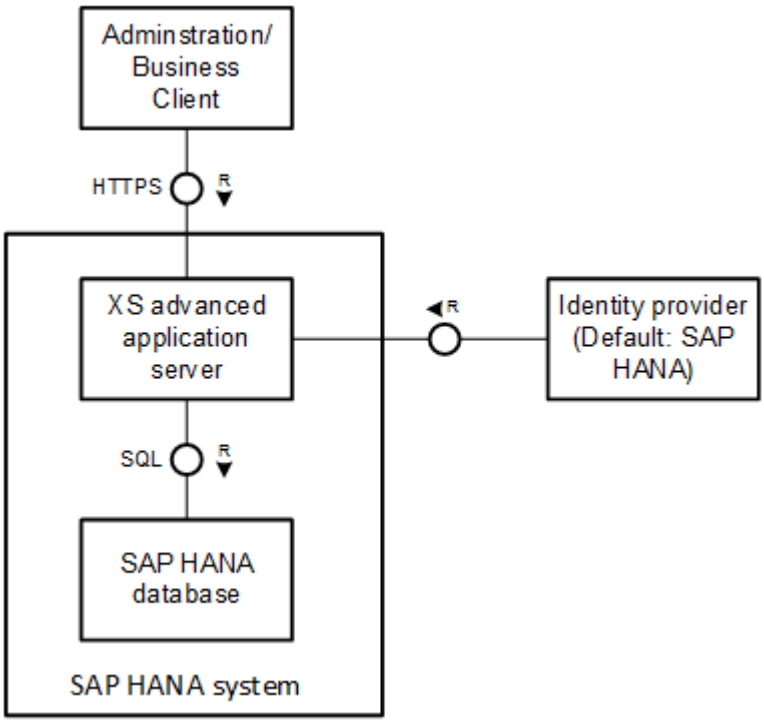
About SAP HANA XS Advanced

SAP HANA XS advanced offers a rich set of embedded services that enable an end-to-end support for web-based applications including lightweight web servers, persistency services, and a configurable identity provider. Furthermore, the platform supports polyglot application development with a core set of pre-deployed runtimes that are accepted as industry standard, for example, node.js or JavaEE.

Although the built-in runtimes come with first-class development and monitoring support, the platform has an open architecture that allows you to add custom runtimes. This high flexibility makes it essential that you put a strong focus on security concepts, not only when configuring and setting up the infrastructure, but also throughout operating the system.

Architecture Overview

As illustrated in the following diagram, the basic system architecture has a classic 3-tier approach:



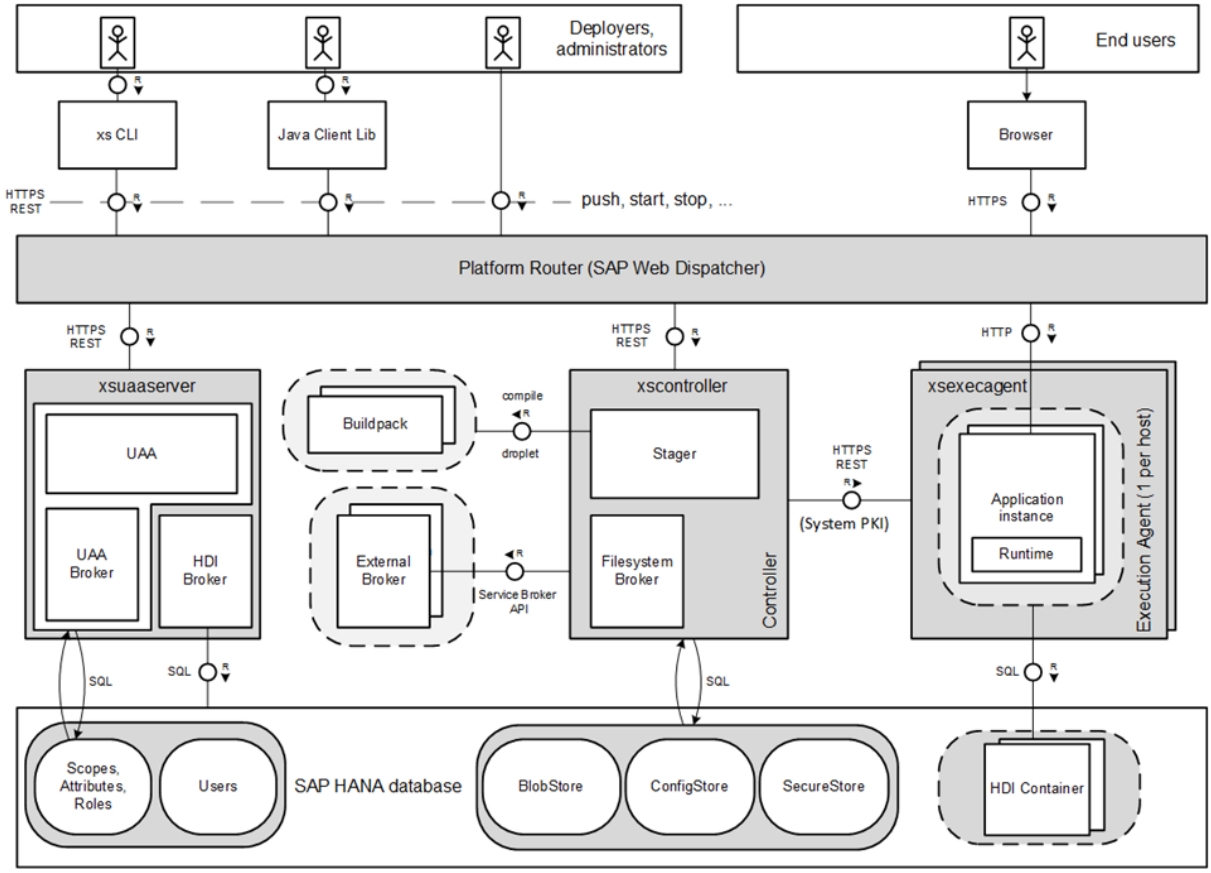
3-Tier Architecture of SAP HANA with XS Advanced

There is a distinction between the overall SAP HANA system and the SAP HANA XS advanced application server. The SAP HANA system refers to the entire SAP HANA platform part of the integrated solution. The SAP HANA XS advanced application server describes only the runtime platform as an integral part of the solution. All services of the SAP HANA system share the same system identifiers (that is, instance number and SID) and are controlled by the `hdbdaemon` service.

The third tier, represented by an SAP HANA database, provides persistency services, that is, data storage. In contrast, the application server components in the middle tier are responsible for deploying, running, and monitoring the applications. Most security-related features such as authentication, authorization, and auditing are primarily enforced in this layer. End users interact on the client layer with system or business users that are authenticated by an identity provider (IdP), which is SAP HANA user management by default. However, both the server components and the applications themselves access the SAP HANA database only through technical database users that the platform generates implicitly. Direct access to the database is only intended for database administration and monitoring purposes.

> ⚠ Caution
>
> As the XS advanced application server is based on the SAP HANA database, security-related configuration settings in the database also have direct effects on the SAP XS advanced application server. For example, if you configure JDBC connections not to support TLS/SSL (which is not the default), application artifact and runtime data could be compromised when transferred between applications and database.

The following diagram provides a more detailed overview of the technical system landscape of the XS advanced application server. All relevant components and storages used by the application server layer are highlighted with a gray background.



Technical System Landscape of XS Advanced Application Server

The XS advanced application server relies on the following SAP HANA services contributing to the integrated platform solution:

1. `xscontroller` (Controller, FileSystem Broker, Platform Router)
2. `xsexecagent` (Execution Agent)
3. `xsuaaserver` (UAA, UAA Broker and SAP HANA Service Broker)

The exact functions of these services are explained in the section *Application Server Components*. These services are configured and administrated with the same tools that are already available for other SAP HANA services, for example, the `hdbsql` or `sapcontrol` command line tools, or the SAP HANA studio. Be aware that all SAP HANA services share the same administrative `<sid>`adm user at operating system (OS) level.

> ### → Recommendation
>
> Due to the fact that the `<sid>`adm has the role of the system super user at the OS level and thus is enabled to access all critical data, it is strongly recommended to limit the number of people who know these credentials.

Related Information

Application Server Components [page 341]

17.1.1 Application Server Components

The XS advanced application server comprises the SAP HANA services `xscontroller`, `xsexecagent`, and `xsuaaserver` services, which are complemented by the Platform Router.

The services `xscontroller` and `xsuaaserver` run on a dedicated host of the system referred to as the XS advanced master host, which is not necessarily the master host for the database. The Platform Router, which is responsible for processing external requests, is managed by the `xscontroller` service and thus always runs on the XS advanced master host.

The execution agent is capable of running application instances on a host where the underlying `xsexecagent` service is started. To deploy an application, at least one execution agent is necessary. But in general, application instances may be scattered on different hosts of a distributed system.

xscontroller Service

The `xscontroller` service provides the central HTTP/REST interface to deploy, run, and monitor web applications. Deploying an application (or more generally a multi-target application or MTA) to the platform consists of several consecutive steps starting with the upload of the application files to the controller. These design-time artifacts typically include various types of content such as code binaries, source files, configuration files, or static HTML content.

Staging (according to Cloud Foundry terminology) denotes the process of transforming the application files into an executable representation by adding an appropriate runtime environment with an integrated web server. This step is performed by the Stager, which has to choose a suitable buildpack to apply to the application files in an external process. For instance, a Java web application archive is placed in a Tomcat server environment. The result of this compilation is a droplet, which represents the executable application on the file system. Due to the fact that the platform can be enriched with third-party buildpacks that support arbitrary runtime containers (they may even be downloaded during staging from a GIT repository), staging is highly security relevant.

The Controller stores the compiled droplets in the BlobStore, which resides in the SAP HANA database. The BlobStore is optimized to store file contents in a very efficient manner. For application start-up, the controller needs to download droplets from BlobStore very quickly to serve the HTTP/REST endpoint of the chosen execution agent. Controller resources such as application or buildpack metadata are stored by the ConfigStore, which is also located in the controller's database schema.

As part of the deployment, applications may be bound to services offered by (external) service brokers. Administrators are free to register arbitrary service brokers that implement the standardized Service Broker API, but most use cases are covered by the system platform brokers for SAP HANA persistency (SAP HANA Service Broker), user authorization (UAA Broker) and file storage (File-system Broker). To consume an offered service, an application has to be bound to the service and typically receives credentials to access the service. These credentials passed by the brokers are generally stored in an SAP HANA secure store.

Platform Router

The Platform Router, which is realized by an SAP Web Dispatcher instance, exposes the public endpoint for the entire system. The router is configured in a way that all application and public server endpoints are represented

by an external URL. External requests are routed to the appropriate back-end instance according to the internal routing table.

> → Recommendation
>
> It is strongly recommended that you limit network access to your system in a way that only the Platform Router's endpoints are accessible from outside the system. This can be accomplished by means of network zones and firewalls. For more information, see the section *Network and Communication Security* in the SAP HANA Security Guide.

The Platform Router instance is managed by the `xscontroller` service.

xsexecagent Service

Execution Agents, established through the `xsexecagent` service, are primarily responsible for starting and stopping application instances in a well-defined environment. To be reachable for end users, launched application instances typically provide a public HTTP port. As a basic monitoring service, the availability of this endpoint is checked periodically by the Execution Agent. Instances that are not reachable are restarted automatically. Different instances of the same application do not necessarily run on the same host in a distributed system (only the concept of host pinning could enforce this). Execution Agents also ensure that application instances of different spaces are not visible to each other at the operating system (OS) layer, if spaces have different OS users attached. For more information, see *Organizations and Spaces*.

> ⓘ Note
>
> Even if application instances run on different OS users, they compete for common system resources like CPU, memory, and disk space by default. To isolate a set of applications, consider pinning the applications (or alternatively their space) to a dedicated host.

As part of the deployment process, applications may be bound to services. The resulting credentials, for example, for accessing a database schema, are added to the process environment of the application instance.

xsuaaserver Service

The `xsuaaserver` service bundles a set of additional server components to complete the platform offering:

- The **User Account and Authentication** service (UAA) is the central user management for all end users interacting either with applications or server components. These are referred to as XS advanced users. The UAA uses the OAuth2 protocol based on the exchange of access tokens (see *User Authentication*). XSA users are named SAP HANA database users by default, but they could also originate from an external identity provider (IdP).
- The **UAA Broker** helps applications to protect their services from unauthorized access. Its API is fully Service Broker API compliant. Its services are consumed at deployment time when application-specific authorizations are requested. The UAA Broker runs on the same HTTP server as the UAA.
- **HDI containers** provide application-specific data storage and the deployment infrastructure in SAP HANA. They can be created during binding of applications against services offered by the SAP HANA Service

Broker. To access HDI containers, technical SAP HANA users are created and the bound applications receive the corresponding credentials. The SAP HANA Service Broker runs on a dedicated HTTP server.

Related Information

SAP HANA Network and Communication Security [page 34]
Organizations and Spaces [page 358]
User Authentication [page 356]

17.1.2 Users and Clients

All end users that access XS advanced application server components or applications are called XS advanced users.

We can distinguish between three different types of XS advanced user:

- **Application users** are all end users who interact with applications hosted on the application server (employees, customers and so on)
- **Developers** are users who develop, deploy, or maintain applications on the platform server
- **Administrators** are users who are allowed to set up and change the configuration of the application server; for instance, they may add new buildpacks or upload custom SSL certificates

> ⓘ Note
>
> Administrators of the XS advanced application server cannot manage the lifecycle of the SAP system, that is they cannot install, configure, start or stop SAP HANA services. This is handled at the OS level.

An XS advanced user's identity generally has its source in the UAA instance. To access a back-end instance, they first have to be authorized at the UAA endpoint and fetch an OAuth2 access token. For instance, if a developer using the xs command-line tool wants to push an application, she first has to enter her credentials as the basis for UAA authentication. As a result, the client receives the signed access token. This contains not only the user's identity but also the set of granted privileges. Based on the user information in the token, the Controller performs an authorization check and rejects invalid requests. The same procedure applies to business application requests. Application users represent the vast majority of all users, interacting with deployed web applications from local browsers.

For more information about user management, see *User Administration and Authentication*.

> ⓘ Note
>
> Although XS advanced users are named SAP HANA users, both applications and server components access SAP HANA artifacts by means of technical users that are generated during the deployment process.

In addition to application requests initiated by users via a web browser, developers and administrators interact with the Controller's HTTP/REST interface. The xs command-line tool, which is installed in the `bin` directory of `/hana/shared/<SID>/xs`, provides a user-friendly way to accomplish typical tasks like listing deployed applications or uploading a custom SSL certificate. Similarly, the Controller API can be consumed

programmatically using the delivered Java client library within Java processes. In general, the server endpoints are intended to be consumed with remote clients not necessarily running on the same host.

Related Information

User Administration and Authentication in SAP HANA XS Advanced [page 344]

17.2 User Administration and Authentication in SAP HANA XS Advanced

Both applications and platform services require user information to perform operations on behalf of an end user. User information in this context covers both authentication and authorization. A user management service lets you control precisely the group of users that are allowed to use specific system services or applications, modify sensitive data, or even do global system configuration.

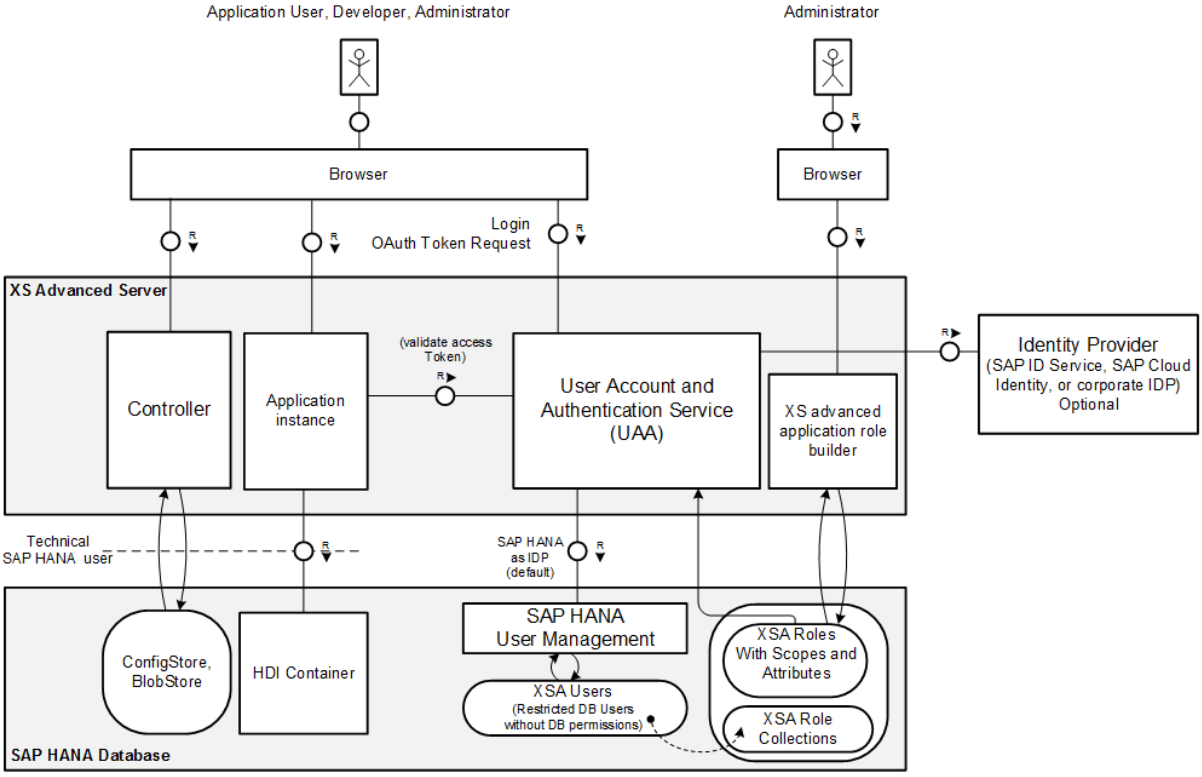
Related Information

User Management [page 344]
Predefined XS Advanced Users [page 347]
Predefined Database Roles for XS Advanced [page 353]
User Authentication [page 356]
User Administration Tools [page 357]

17.2.1 User Management

In traditional application servers, user information is kept in a local user store. In contrast, the SAP HANA XS advanced platform allows the use of SAP HANA as an identity provider (IdP) but enables the integration of an external IdP such as SAP ID Service or SAP Cloud Identity. Custom IdPs can also be configured, as long as they implement the SAML 2.0 standard. The XS advanced platform uses the underlying SAP HANA user store as IdP by default.

The **User Account and Authentication service** (UAA) represents the central platform service for user management and authentication, as depicted in the following diagram:



User Account and Authentication Service (UAA)

User information, such as first name, last name, user ID and user privileges, is provided in the form of signed OAuth2 access tokens the central UAA issues when a client logs in successfully. For more information about the authentication procedure, see the section on XS advanced user authentication.

XS Advanced User Categories

XS advanced users access the back-end instances typically through end-user interfaces such as a Web browser or command-line tools. Unlike technical users, application users and some types of system user can be also identified by personal data such as name, e-mail address, and so on. As the same identity provider is the basis for all of XS advanced users, an application user may also be granted developer privileges and the other way around.

XS advanced users who have their source in the SAP HANA user store (default) are typically restricted users with no access to SAP HANA database schemas. In contrast, applications and server components use **technical SAP HANA users** with certain access privileges. The platform passes these credentials to applications, enabling them to execute SQL statements, if the XS advanced user has sufficient privileges. Decoupling XS advanced users from technical users is the precondition for leveraging external IdPs, even though XS advanced users are also SAP HANA users by default. As technical SAP HANA users are generated by the platform in the background, you typically won't use them to interact with the system.

The XS Advanced User Account and Authentication (UAA) service provides the authentication end point for individual users who need to interact either with SAP HANA XS, advanced model, or with applications deployed

and running on the XS advanced model platform. Although such users are often referred to simply as **XS advanced users**, they can have the following roles and areas of responsibility:

- Platform users [page 346]
- Application or business users [page 346]
- Operating-system users [page 346]
- Technical database users [page 347]

XS Advanced Platform Users

Platform users are administrators or developers who are assigned to one or more specific organizations or spaces in the XS advanced platform. An XS advanced administrator user (for example, `XSA_ADMIN`) is allowed to perform any platform operation in any organization or space. However, it is also possible to maintain additional platform user **roles** and use them to restrict the type of access granted to certain users for particular organizations or spaces. XS advanced "administration users" are system users who manage the configuration of the XS advanced application server components, and in particular the XS Controller.

XS advanced platform users are SAP HANA users who have been assigned to a specific XS advanced role collection. Non-administrator platform users can also be managed by means of an external Identity Provider (IdP).

> → Tip
>
> You can use the `xs` command-line interface to maintain XS advanced platform users. For more information, see *Maintaining Platform Users in XS Advanced with the XS CLI* in *Related Information* below.

XS Advanced Application (Business) User

Often referred to as "business users", application users interact with application instances deployed to and running on the XS advanced run-time platform. Application users are also referred to as business users, for example, employees, customers, and so on.

Application users can be identified by personal data such as name or e-mail address, and this data along with other credentials are stored in a user store, for example, an Identity Provider (IdP); any request to log on to an XS advanced application is managed by the XS advanced User Account and Authentication service (UAA). Authorization scopes (defined in user roles) are granted to application users to restrict or enable access to particular data.

XS Advanced Operating-System Users

In the context of XS advanced, the following predefined operating system users are available by default:

- `<sid>adm`
  Operating-system and administrative SAP HANA system user who owns all platform services as well as the system's file storage.
- `sap<sid>xsa`
  Operating-system user required for staging and running applications in the pre-configured `SAP` space.
- `<sid>xsa`
  Operating-system user required for staging and running applications in the pre-configured `PROD` space.

The `<SID>adm` operating system user exists to provide an operating system context. From the operating system perspective, the operating system administrator is the user that owns all SAP HANA files and all related operating system processes. Certain administration operations require the operating system user's credentials, for example, starting or stopping the system.

> **Note**
>
> XS advanced application files are also owned by the `*xsa` operating-system users `sap<sid>xsa` and `<sid>xsa`.

XS Advanced Technical Database Users

A technical database user does not correspond to a real person and should be used for administrative tasks such as creating objects and granting privileges for a particular application. For example, an application server may log on to the SAP HANA database using a dedicated technical database user. In the context of XS advanced, technical SAP HANA users are generated by the platform in the background.

For XS advanced, the technical user `SYS_XS_RUNTIME` owns the XS Advanced Controller's SAP HANA schema, which contains the Blob Store, Config Store, and Secure Store. Similarly, the technical user `SYS_XS_UAA` owns the SAP HANA schema provided for the User Account and Authentication (UAA) for user management

Additional technical database users are created on demand and as required for application-specific purposes. For example, the `SBSS_*` users are created as a result of an application-service binding. XS advanced also makes use of a number of `USR_*` users, too; `USR_*` users are created by the SAP HANA Service Broker for the service plans `schema`, `securestore`, and `sbss`. Similar to the predefined users created when binding an application to an HDI container, `USR_*` users are used by applications to access their schema. For more information, see the section on predefined users.

> **Note**
>
> As of SAP HANA 2.0 SPS 03, the SAP HANA Service Broker no longer uses the `SBSS_` prefix for HDI container users. Instead, these HDI container users have the name of the corresponding HDI container as the prefix. For example, for users created during service binding, the following format is used: `<HDI_Container_Name>_<GUID>_DT` (design-time access) or `<HDI_Container_Name>_<GUID>_RT` (run-time access). Binding users are assigned the role `PUBLIC` by default.

Related Information

User Authentication [page 356]
Maintaining Platform Users in XS Advanced
Predefined XS Advanced Users [page 347]

17.2.2 Predefined XS Advanced Users

The installation of the XS advanced application server creates a small set of predefined users that enable the operation of the underlying system.

The system's super user (`<sid>adm`) needs to be available in order to manage the life cycle of the system. Similarly, an administrative XS advanced system user (`XSA_ADMIN` by default) is necessary to perform the initial setup of the application server, for example, granting other users the privilege to create spaces in a dedicated organization and so on. Technical database users are created during installation for all server components that need to persist data in SAP HANA schemas.

The following predefined users exist:

- Predefined XS Advanced System Users [page 348]
- Predefined Technical SAP HANA Users [page 349]
- Predefined OS Users [page 352]

Predefined XS Advanced System Users

The table below lists the predefined XS advanced system users that are necessary for operating the XS advanced application server. First, an administrative user named `XSA_ADMIN` is required for the XS advanced Controller; this administrative user configures the application server at a global level. Non-administrative users of the XS advanced Controller are not allowed to perform administration tasks, for example, uploading custom certificates, adding custom buildpacks, or registering platform service URLs. Bear in mind that, although the credentials for the technical users for the SAP HANA Service Broker and UAA Broker are generated automatically during installation, the `XSA_ADMIN` user is created interactively with a user-defined password. As a first-level administrator user with irrevocable privileges, the `XSA_ADMIN` has unlimited access to the XS advanced Controller and therefore needs to be handled carefully.

→ Recommendation

- Keep the number of people with `XSA_ADMIN` credentials as small as possible. Delegate specific tasks like space management to lower-privileged users instead.
- Avoid creating other powerful users with privileges similar to `XSA_ADMIN`.
- Change the `XSA_ADMIN` password at regular intervals.

User ID	User Type	Description
`XSA_ADMIN`	XS advanced user	Administrative user for the XS advanced application server with unlimited access to XS advanced Controller API
`SYS_XS_SUPPORT`	Database user	Owns the stored procedures used to grant access privileges on schemas of XS-advanced-related technical database users. Permission to call the stored procedures is also granted to the `SYSTEM` user. See also SAP note 2656132 in *Related Information* below. **Note** The `SYS_XS_SUPPORT` user is deactivated by default.

Although the technical users in this table are created in the SAP HANA database, and database authentication checks are used to confirm the technical users' credentials, the technical users are not used to connect to the SAP HANA database.

Predefined Technical SAP HANA Users

Most of the server agents require a data store in the SAP HANA database and therefore need secure access to schemas. For this reason, a dedicated technical SAP HANA user is generated for each such schema, and the credentials of the technical SAP HANA user are passed to the server agent. As the management of technical users is performed at the infrastructure level, end users do not interact with these users.

> ⚠ Caution
>
> It is not recommended to change the properties of a predefined technical SAP HANA user, for example, by changing a user's password or deactivating a user. Modifying a technical user manually will cause XS advanced services to stop working properly. However, for security reasons, it is recommended to rotate the passwords of these technical users from time to time, for example, with the `XSA` command-line interface and the command `XSA renew-passwords-of-technical-users`, as described in the *SAP HANA Administration Guide*.

Starting with SPS04, the technical users listed in the following table are grouped into a single user group named `SYS_XS_UG_RUNTIME_<db-id>`, where `<db-id>` is the unique identifier for a database. The only exception to this user-group rule is `SYS_XSA`, which is created in the user group `SYS_XS_UG_RUNTIME`. These technical users are used to connect to the SAP HANA database with a specific set of conditions.

User ID	Service	Description
`HDI_ADMIN_USER`	SAP HANA service broker	Owns SAP HANA schema of SAP HANA Service Broker
`HDI_BROKER_CONTROLLER`	SAP HANA service broker	Has authorization to access the service broker API of SAP HANA service broker
`SYS_XS_HANA_BROKER`	SAP HANA service broker	Owns the SAP HANA Service Broker's SAP HANA schema
`SYS_XS_HANA_BROKER_INTERNAL`	SAP HANA service broker	Has authorization to execute stored procedures for creating users, and so on.
`SYS_XS_INSTANCE_MANAGER_ADMIN_USER`	Instance Manager	Owns SAP HANA schema of the Instance Manager
`SYS_XS_INSTANCE_MANAGER_BROKER_USER`	Instance Manager	Has authorization to access service broker API of Instance Manager
`SYS_XS_OID_USER`	OIDC	Owns the SAP HANA schema for the OpenID Connect provider
`SYS_XS_OID_USER_SEC`	OIDC	Owns the SAP HANA secure store for the OpenID Connect provider

User ID	Service	Description
SYS_XS_RUNTIME	Controller	Owns the Controller's SAP HANA schema containing BlobStore, ConfigStore and SecureStore
SYS_XS_SBSS	SAP HANA service broker	Owns SAP HANA schema containing procedures to generate user passwords in a secure manner; used by the SAP HANA service broker
SYS_XS_SYSTEMDB_INFO	Controller	Has authorization to access database system catalog and configuration
SYS_XS_UAA	UAA	Owns the UAA's SAP HANA schema for user management
SYS_XS_UAA_SEC	UAA	Owns the UAA's SAP HANA secure store for user credentials
SYS_XS_UAA_USER_ADMIN	UAA	Owns the UAA stored procedures that perform user-management operations and assign role collections
SYS_XSA	Installer	Owns SAP HANA schema containing a unique tenant ID
_SYS_DI	HDI	Owns all HDI SQL-based APIs, for example all API procedures in the _SYS_DI schema and API procedures in containers
_SYS_DI_*_CATALOG	HDI	Technical users used by the HDI to access database system catalog tables and views
_SYS_DI_SU	HDI	Technical superuser of the HDI created at installation time
_SYS_DI_TO	HDI	Owns transaction and connections of all internal HDI transactions

Technical Users for HDI Schema-Based Containers

The deployment of database objects with SAP HANA Deployment Infrastructure (HDI) is based on a container model where each container corresponds roughly to a database schema. Each schema, and the database objects deployed into the schema, are owned by a dedicated technical database user.

For every container deployed, a new technical database user and schema with the same name as the container are created. Additional schemas and technical users required for metadata and deployment APIs are also created.

These technical database users are used internally by HDI only. As a general rule, these users are created as restricted database users who do not have any privileges by default (not even the role PUBLIC), and cannot be

used to log on to the database. The only exception to this rule concerns user S#00 who, from SAP HANA 2.0 SPS 02 revision 20, is granted the role PUBLIC by default.

> → Tip
>
> For more information about HDI security, see *SAP HANA Deployment Infrastructure (HDI) Reference for SAP HANA Platform* in *Related Information* below.

Technical Users for Default Application Services

XS advanced applications can make use of a number of services managed by a service broker. To make use of a service, an instance of the service must be created and the application must be bound to the specified service instance. Several services are available by default; they are installed with the XS advanced run-time platform.

The installation of the following default application services results in the creation of a number of internal technical users:

- `Product-Installer`
  Used for the installation and installation management of applications
- `Deploy-Service`
  Used in the technical deployment of applications packaged in multi-target application (MTA) archives

The operation of binding these services to an application generates a technical user and random password according to the following naming convention USR_<generated_ID>. These technical users are required to make database schemas available for applications. For every combination of application and schema, such a technical user is created.

In addition, the `Job-Scheduler` service, used to create and schedule long-running operations in the XS advanced environment, uses an HDI container with the SBSS_ prefix and a randomly generated name. The above-mentioned HDI schemas and users will be created for this container.

For more information, see *The SAP HANA XS Advanced Services: Deployment Infrastructure* in the *SAP HANA Developer Guide (For SAP HANA XS Advanced Model)*.

User Groups for XS Advanced Technical Users

Starting with SAP HANA 2.0 SPS04, technical SAP HANA users are managed in user groups. This is also true for the technical users owned by the XS advanced run-time and the users created when the SAP HANA Service Broker creates a service instance or a service binding for an XS advanced application. The following table provides an overview of the user groups that contain users who are relevant for XS advanced:

User Group Name	Description	Example Group Members
`SYS_XS_UG_RUNTIME_<db-id>`	Includes all technical users created by the core XS advanced run-time services	`SYS_XS_RUNTIME` `SYS_XS_UAA`

`_SYS_DI #SYS_XS_HANA_BROKER`	Includes all technical users created when the SAP HANA service broker creates a service instance with the service plan "`hdi-shared`"	If "S" is the container name, then:
		`S`
		`S#DI`
		`S#00`

User Group Name	Description	Example Group Members
SYS_XS_UG_BROKER_HDI_SHARED #SYS_XS_HANA_BROKER	Includes all service binding users created when the SAP HANA service broker creates a service instance with the service plan "hdi-shared"	*_RT *_DT
SYS_XS_UG_BROKER_SCHEMA #SYS_XS_HANA_BROKER	Includes all technical users created when the SAP HANA service broker creates a service instance with the service plan "schema"	USR_*
SYS_XS_UG_BROKER_SECSTORE #SYS_XS_HANA_BROKER	Includes all technical users created when the SAP HANA service broker creates a service instance with the service plan "securestore"	USR_*
SYS_XS_UG_BROKER_SBSS #SYS_XS_HANA_BROKER	Includes all technical users created when the SAP HANA service broker creates a service instance with the service plan "sbss"	USR_*

Predefined OS Users

Ultimately all platform services are made up of operating-system (OS) artifacts such as OS processes, network sockets, and file storage. Since operating systems come with their own user management features, these artifacts are out of necessity owned by OS users. Consequently, the XS advanced application server cannot be run without at least one OS user, although dedicated XS advanced users are able to perform the majority of the operational tasks.

The installation procedure creates the "super" OS user <sid>adm for the entire SAP HANA system. As the owner of all operating-system processes, the <sid>adm user is very powerful from a security perspective. For this reason, we strongly recommend that you limit the number of people with <sid>adm credentials as far as possible.

As described in the section *Application Server Components*, some platform services launch new processes at runtime:

- Execution Agents start application instances.
- The application "Stager" spawns processes running build packs during application staging.

In both cases, custom code comes to execution. If these processes ran as the system's <sid>adm user, the whole system could be compromised. To prevent this, the platform generally spawns external processes with OS users that are attached to the application's space. To support this approach, the initial setup includes OS user <sid>xsa user for the PROD space and OS user sap<sid>xsa for the SAP space. For more information about this isolation concept, see *Organizations and Spaces* and SAP Note 2243156 *Secure user setup for SAP HANA extended application services, advanced model* .

The following table summarizes the OS users that are available immediately after installation:

User ID	Type	Description
`<sid>adm`	OS user	Administrative SAP HANA system user who owns all platform services as well as the system's file storage
`<sid>xsa`	OS user	OS user for staging and running applications in the pre-configured `PROD` space
`sap<sid>xsa`	OS user	OS user for staging and running applications in the pre-configured `SAP` space

Related Information

Application Server Components [page 341]
Organizations and Spaces [page 358]
Maintaining SAP HDI Containers
SAP HANA and HDI Services in XS Advanced
SAP HANA Developer Guide for XS Advanced Model (SAP Web IDE)
SAP Note 2243156

17.2.3 Predefined Database Roles for XS Advanced

Several predefined database roles are necessary for operating the XS advanced application server.

> **Note**
>
> The following roles are SQL-based roles available in the catalog of the SAP HANA database.

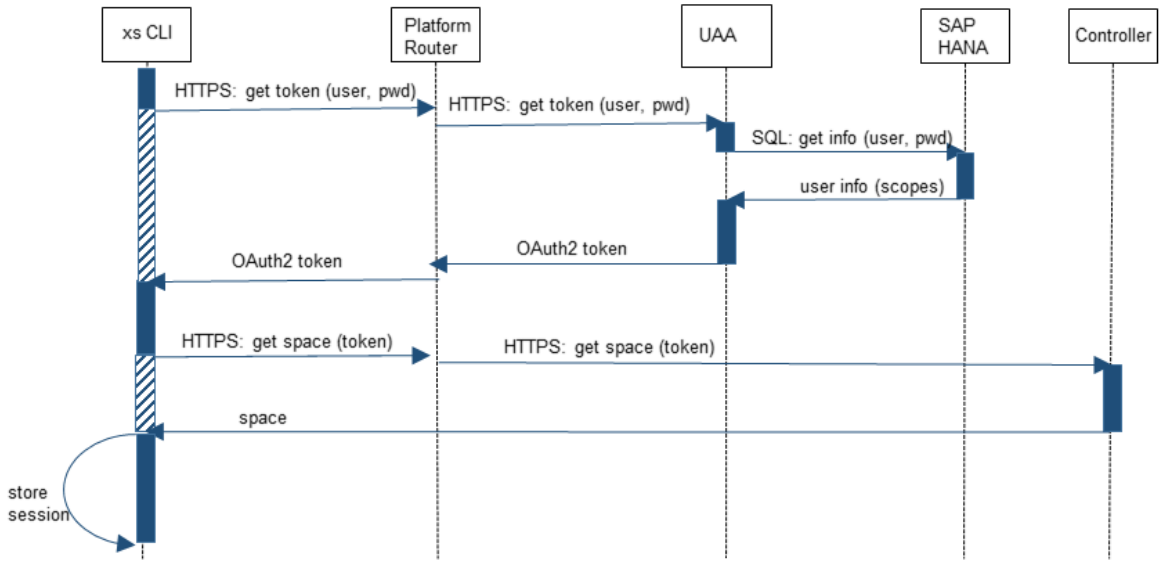
Role	Description
`_SYS_DI_OO_DEFAULTS`	This role contains the set of default privileges that are granted to all HDI container object owner users (`<container>`#OO users). SAP HANA Deployment Infrastructure (HDI) uses this role internally to grant default privileges instead of using the `PUBLIC` role. It contains only privileges to SYS views where additional security checks apply. The role contains SELECT privileges on the views: `SYS.DUMMY`, `SYS.PROCEDURES`, `SYS.PROCEDURE_PARAMETERS`, `SYS.TABLES`, `SYS.TABLE_COLUMNS`. This role is not intended to be granted to database users. **Note** Do **not** extend this role in a production system.
`_SYS_DI#<container_group>._SYS_DI_OO_DEFAULTS`	This role contains the set of default privileges that are granted to all HDI container object-owner users (`<container>`#OO users) who are part of the container group `<container_group>`. The role does not contain any privileges by default. Privileges granted to a container group-specific role will automatically be available to all HDI container object-owner users of the specified container group. **Note** This role is not intended to be granted to database users.

Role	Description
SYS_XB_SBSS_VIEWER	This role contains selected privileges for monitoring the status of the Service Broker Security Support (SBSS) component. The SBSS component provides service brokers with functions for creating, validating, and deleting the credentials they need for service bindings. Credential handling is achieved by creating restricted database users with secure random passwords. Specifically, this role contains read access to the SBSS component version table, in addition to read access to the SBSS bindings table that lists the credential names that have already been created with the SBSS API as well as some meta data for the bound credentials. This role is intended only for support users so they can query information such as SBSS version, number of credentials, names of services brokers that called the SBSS API. **Note** This role does not grant access to any SBSS credentials.

17.2.4 User Authentication

XS advanced user management is supported by a state-of-the-art user authentication strategy.

For technical SAP HANA users, the basic authentication mechanism applies as described in the SAP HANA Security Guide. In contrast, XS advanced users managed by the UAA are authenticated on the basis of the standardized OAuth2 protocol as depicted in the following sequence diagram:



OAuth2 Authentication Sequence Diagram

Using OAuth2 terminology, a client needs to fetch a protected resource from a resource server. Before being able to receive the resource from the server, the client sends a token request to the authorization server along with the user credentials. The authorization server checks the user credentials and composes the maximum set of privileges the user is granted. The user's identity, together with the authorization information, is encoded into a signed OAuth2 token, which is then sent back to the client. Now, the client can submit the resource server request with the attached access token. The resource server decodes the token (done offline without the authorization server), validates the user, and checks the privileges. If the privileges shown in the token allow access the resource, the server responds to the client request by sending the relevant resource. In the XS advanced application server infrastructure, the central UAA instance fulfils the role of authorization server. Application instances and the Controller are resource servers.

> ### Note
>
> Since providing a valid token at a server endpoint allows clients to access resources, tokens are never transferred unencrypted.

17.2.5 User Administration Tools

XS advanced users managed by the User Account and Authentication service (UAA) need to be administrated, in other words, users need to be created, updated, and also possibly deleted.

As the UAA service uses SAP HANA's user management mechanisms by default, all tools described in the SAP HANA documentation for managing SAP HANA users can be used, including:

- The `hdbsql` command-line tool
- SAP HANA cockpit
- SAP NetWeaver Identity Management

In addition, the SAP HANA XS advanced platform comes with the application *SAP HANA XS Advanced Cockpit*, which provides tools for performing all tasks related to user management in XS advanced. The `xs` command line interface (`xs` CLI) can also be used for some user management tasks. For more information, see the section on maintaining the SAP HANA XS advanced model runtime in the *SAP HANA Administration Guide*.

Related Information

Maintaining the XS Advanced Run-time Environment with a Graphical User Interface
Maintaining the XS Advanced Run-time Environment with a Command-Line Interface

17.3 Authorization in SAP HANA XS Advanced

XS advanced users can access system services or interact with hosted applications. Since you don't want all XS advanced users to be able to view or even modify all resources, an appropriate authorization concept is required to allow you to control precisely which resource entities may be read or edited by a specific user.

In the XS advanced model, user permissions are derived from assigned roles. In addition, resources from different applications may be isolated by leveraging the concept of organizations and spaces. As the central entry point for system users, the Controller comes with a complementary role model. Various tools help you to define roles and assign them to users.

Related Information

Organizations and Spaces [page 358]
Scopes, Attributes, and Role Collections [page 364]
Controller Role Model [page 367]
Authorization Management Tools [page 371]

17.3.1 Organizations and Spaces

Resources from different applications may be isolated by leveraging the concept of organizations and spaces in combination with separated operating system users.

Introduction to Multi-Target Applications (MTAs)

A microservice-driven architecture of a solution is typically characterized by the cooperation of several service instances fulfilling dedicated task. Only by combining microservices can the solution meet its overall requirements. In the XS advanced context, several applications work closely together, forming what is referred to as a **multi-target application**, or MTA.

To illustrate, let's assume a simple MTA consists of two applications. A UI-based application needs to read database tables, which in turn are written by the other application. Consequently, both applications need exclusive access to the same database schema. The applications should also have a common authorization concept that applies to the same pool of end users. However, all other applications outside this MTA should be strongly isolated from the MTA's resources, in other words they should not be allowed to access the stored data nor the MTA's HTTP endpoints.

For more information about MTAs, see the *SAP HANA Developer Guide for SAP HANA XS Advanced*.

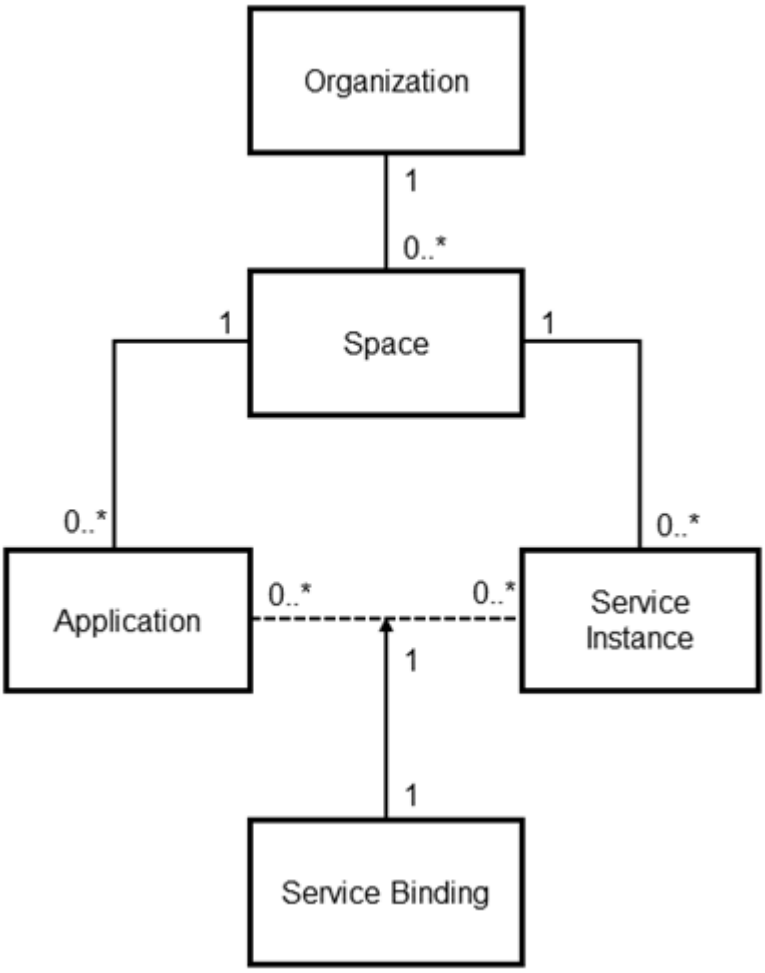
Controller Model

In general, you'll have applications that were deployed by the same Controller user, share the same set of resources, and are used by the same group of end users. This tight coupling of applications can be modeled by leveraging the central concept of **spaces** in the Controller model.

The main idea of spaces is that they form a kind of trust zone, which basically means that all applications deployed to the same space may share common resources like data storage and user authorizations and passwords. A space is intended to be shared by several developers, but developers may also have their own private space as well. Each application must be deployed to an existing space that has already been set up. Also service instances, provided by a service broker and typically representing a resource, can only be created within a space. An application in the space of this service instance may gain access to its resource by explicitly binding it to the service instance. The service binding entity then bears the credentials the service broker has issued during binding. The Execution Agent passes these credentials to the instances of bound applications by writing them to their process environment during start-up.

An **organization** may comprise several spaces. This helps to manage and administrate the spaces in a collective manner. For instance, an organization may group all spaces of a specific functional area of a company. In contrast to spaces, the organization of an application does not have an essential impact on the runtime behavior. There is only one exception: you can specify organization-specific domains that are the basis of the applications' external URLs in case of name-based routing.

The relationship between the involved model entities are represented in the following diagram:



Organizations and Spaces

All applications of an MTA are deployed to the same space, but other applications might be deployed there as well. Here, the Controller role model comes into play demanding deployment privileges for each single space. At the level of organizations, privileged Controller users with the role `OrgManager` are allowed to create new spaces within their organization and appoint other Controller users to be the manager of this specific space (for example, `SpaceManager`). Space Managers in turn may grant Controller users deployment privileges. For more information, see the section on controller role model.

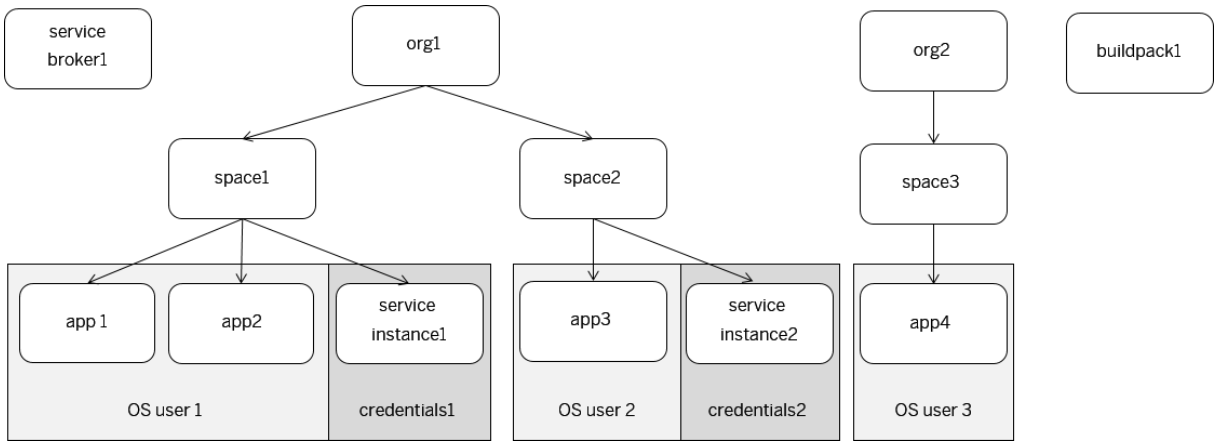
Spaces and Operating System (OS) Users

The isolation of spaces depends on different OS users. Spaces can be mapped to dedicated OS users, and only spaces running with different OS users are isolated from each other.

Applications running in the same space share all resources such as data storage, user authorizations, and passwords. Furthermore, the external buildpack process that is forked by the Stager run with this OS user.

Application instances and buildpacks in different spaces are only isolated at OS level if their space is running with a dedicated OS user. This is important from a security perspective when you consider that both types of OS processes run custom code.

The following figure shows a sample Controller model with regards to organizations, spaces, and applications and isolation on OS level.



Example: Organizations and Spaces

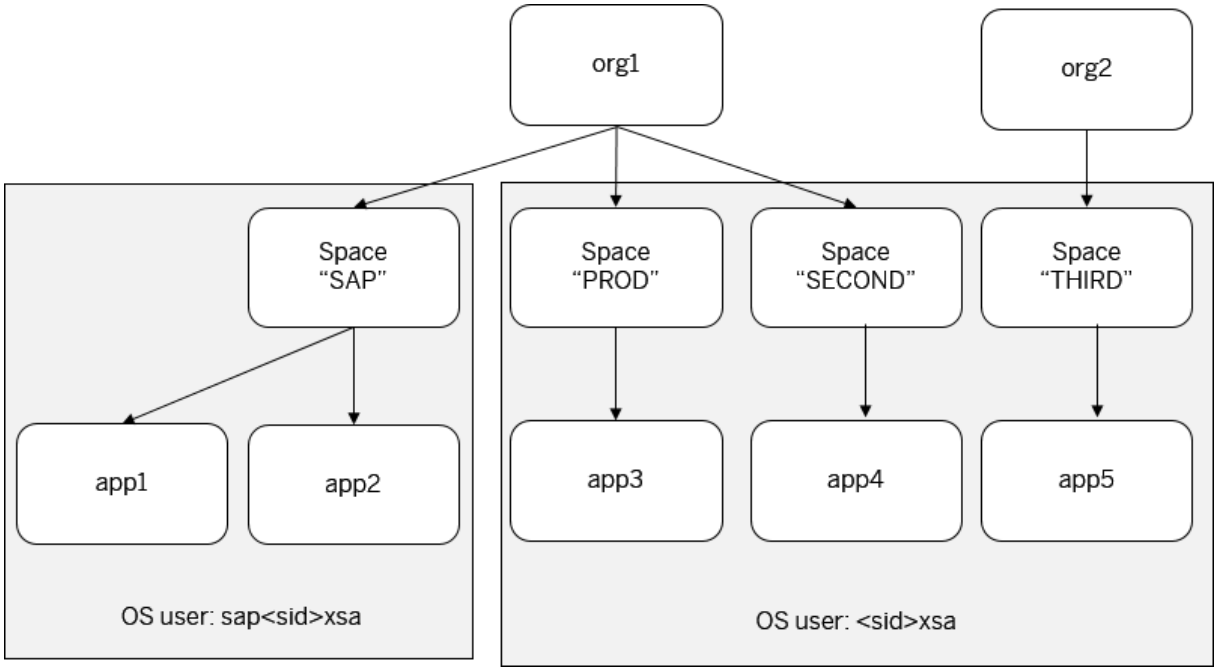
Default Organizations and Spaces

The XS advanced application server comes by default with two spaces that are generated and mapped to dedicated OS users, and therefore isolated on OS level.

Organization	Space	Space OS User	Content
Initial organization	SAP	`sap<sid>xsa`	Contains pre-installed SAP applications, for example: • Deploy-service for MTAs • Product-Installer and component-registry for application installation and update This space is an appropriate location for other system-relevant applications from SAP such as the optional administration UI or the Application Role Builder tool.
Initial organization	PROD	`<sid>xsa`	Empty after installation The space PROD can be used to deploy your custom applications.

> **Note**
>
> Since the SAP space contains administrative applications, it is highly recommended to keep the number of users with access to the SAP space as small as possible.

The following diagram provides a graphical representation of the isolation at the user and operation-system levels between the default spaces in XS advanced.



Space Isolation of Default Spaces

Custom Organizations and Spaces

By default, all spaces created by customers use the default pre-defined user `<sid>xsa`. This user is also assigned the initial default space named `PROD`.

> **Note**
>
> In the default set-up, all applications in the initial space `PROD` and all additionally created spaces share the same OS user `<sid>xsa` and as a result, share resources such as data storage, user authorizations, and passwords on the OS level.

As `SpaceManager` you can configure space isolation by attaching a dedicated OS user to a newly created space. You can do this with the XS command line tool, either when you create the space or as a configuration step afterwards:

- `xs create-space <OS user name>`
- `xs update-space <OS user name>`

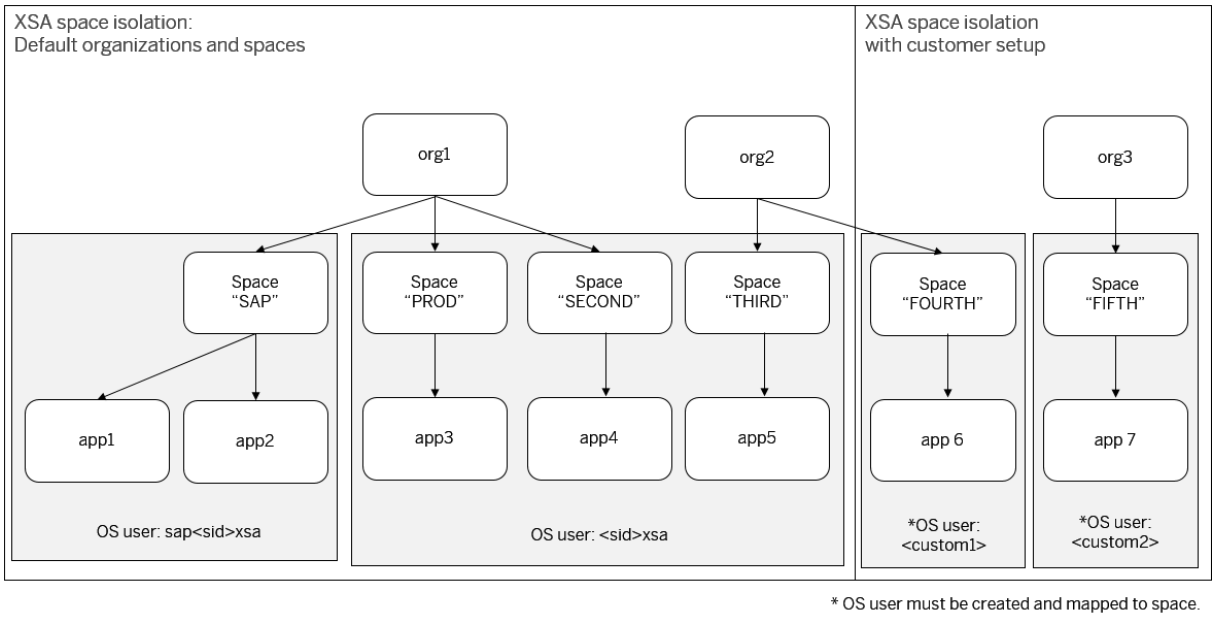
> **Note**
>
> The changes only take effect on newly staged or restarted applications.

You can review the assign of spaces to OS users with the XS command `xs spaces`. All spaces mapped to the same OS user share resources.

The creation of OS users for space isolation is described in SAP Note 2243156. The OS user must be available on all hosts of the system that run services of the XS advanced application server, especially `xscontroller` and `xsexecagent` services.

> **Note**
>
> For security reasons, you are not allowed to set the `<sid>adm` as a space OS user. Also be aware that the space OS user runs custom code on the executing host. So, restrict its privileges as much as possible.

The following figure shows a sample Controller model with regards to organizations, spaces, and applications and OS users, and the resulting space isolation.

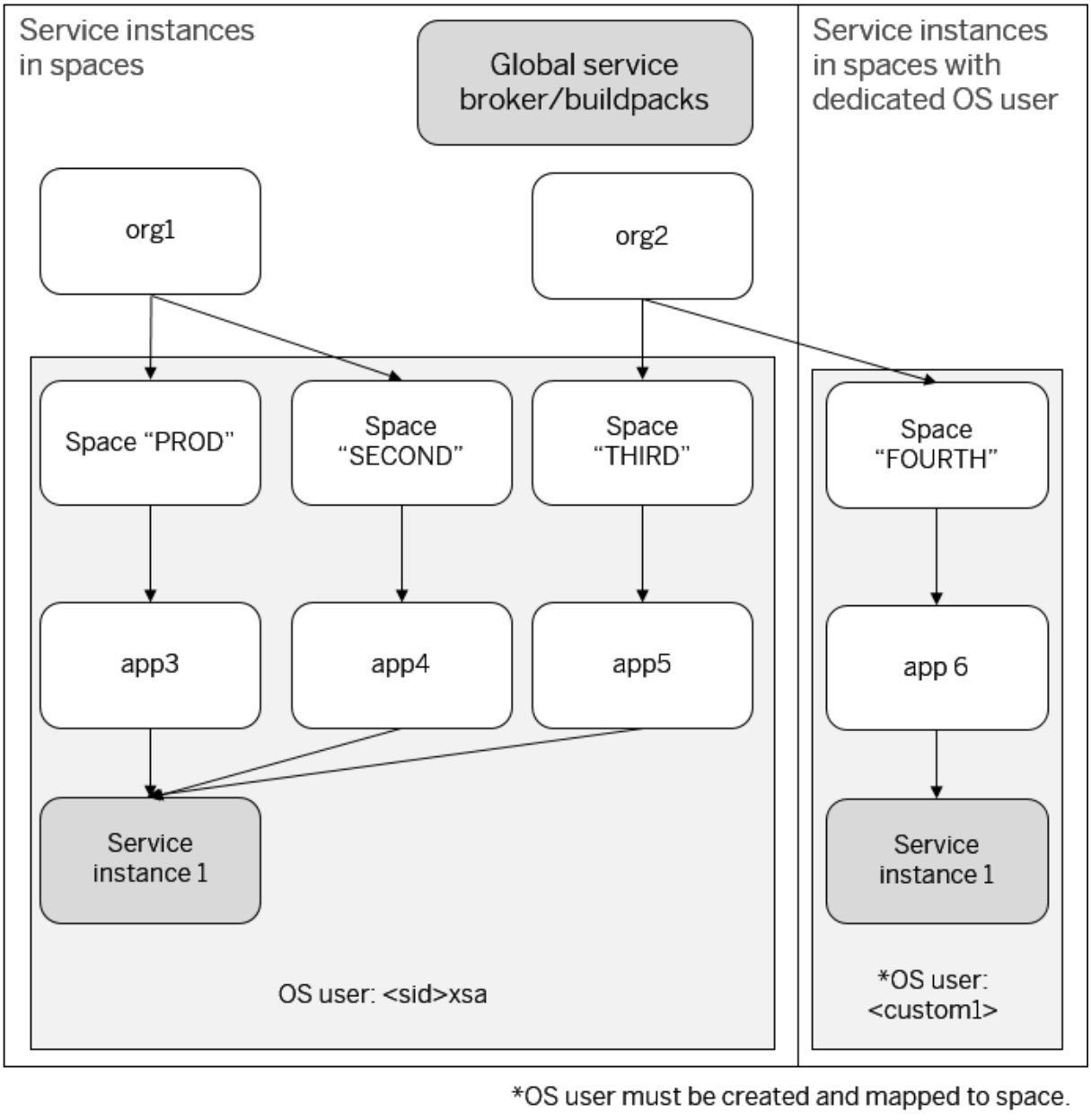


Space Isolation with Custom OS Users

- Space SAP in organization org1 is running with the OS user `sap<sid>xsa` and therefore all applications running in the space SAP are isolated from other spaces not running with`sap<sid>xsa`. Applications app1 and app2 share resources such as data storage and passwords on the OS level.
- Spaces PROD, SECOND, and THIRD are all running with the same OS user `<sid>xsa` and applications app3, app4 and app5 share resources such as data storage and passwords on the OS level. Spaces are not isolated as they are mapped to the same OS user.
- Space FOURTH is mapped to a dedicated OS user <custom1> and therefore the space and application app6 is isolated from applications in other spaces. The OS user <custom1> must be created manually before it can be mapped to the space.
- Space FIFTH is also isolated as it too is running with a dedicated OS user <custom2>. Similarly, the OS user <custom2> must be created manually before it can be mapped to the space.

Service Instances in Spaces

Service instances are created per space, but for isolation they also depend on the OS user mapped to the space. Service instances can be reached by all applications running in spaces that share the same OS user. Service bindings in applications might contain credentials to service instances. Note that some entities like service brokers and buildpacks are created at a global model level and are shared by all spaces.



Isolation of Service Instances

More Information

- For more information about how to assign Controller roles to Controller users, see the section *Controller Role Model*.

- For more information about how to create new organizations and spaces, see the *SAP HANA Administration Guide*.
- To isolate application instances with respect to resource consumption, make use of host pinning as described in the *SAP HANA Administration Guide*.

Related Information

Controller Role Model [page 367]
Maintaining Organizations and Spaces in XS Advanced
Maintaining Host Pinning
SAP Note 2243156

17.3.2 Scopes, Attributes, and Role Collections

Scopes define the actions that can be performed within a service. Attributes define the application's entities a user may access. A role collection is a list of scopes combined with a list of attributes.

The XS advanced authorization concept is based on the OAuth2 protocol, which requires users to pass an access token with each server request. This not only applies to business users who want to access the endpoints of deployed applications (to be more precise, they are redirected to logon page of the User Account and Authentication (UAA) service to fetch the token), but also Controller users. Privileges to perform specific operations are associated with so-called **scopes**, which are simply represented by static strings defined by the resource servers (applications or server components like Controller). In other words, scopes define the actions that can be performed within a service.

Attributes, on the other hand, define the application's entities a user may access. A list of scopes combined with a list of attributes define a role, and a list of several roles defines a **role collection**. An XS advanced user with the appropriate privileges can be assigned several role collections. The key point is: An OAuth2 token issued by the UAA on a user request contains all scopes and attributes that are granted to the user based on the assigned role collections. On the basis of these scopes and attributes, an application can perform an authorization check after having decoded the access token.

MTA-specific role collections are design-time artifacts. The scopes and attributes a role collection is based on are defined in the MTA's security descriptor, `xs-security.json`, which is read and evaluated during the deployment of the MTA. The resulting role templates can be instantiated to roles and then grouped into role collections using the XS CLI role-collection commands or the SAP XS Advanced cockpit. Finally, XS advanced users are assigned role collections, using either the XS CLI role-collection commands or the SAP XS Advanced cockpit. For more information about how to handle application role collections, see the *SAP HANA Developer Guide for SAP HANA XS Advanced*. XS advanced users who need the privileges required to interact with system components (for example, the Controller) need to be assigned predefined role collections. These standard role-collections are created automatically during installation.

For more information about how to assign role collections to users, see *Authorization Management Tools* in *Related Information* below.

Standard Controller Role Collections

Users who interact with the Controller may trigger different types of operations, for example:

- Create, update, or view a space
- Create, update, or view an application in a space
- Create a service instance and bind an application to it
- Start, stop, or scale an application
- Add a custom buildpack
- Add an external service broker
- Upload custom SSL certificate
- Grant Controller roles to other Controller users

Some of the operations with a system-wide effect require administrative privileges (for example, uploading certificates). Others need to operate on the Controller's resources (applications, spaces, etc.) with read or write permission. To cover these basic use cases, the Controller defines the following scopes:

- `cloud_controller.admin` (unlimited access)
- `cloud_controller.admin_read_only` (unlimited read access)
- `cloud_controller.global_auditor` (read access to all resources except system resources and sensitive data)
- `cloud_controller.write` (write access)
- `cloud_controller.read` (read access)
- `cloud_controller.credentials.view` (read access to sensitive data in restricted mode)

In line with the authorization concept described above, these scopes for the XS advanced controller are combined into the following role collections:

- `XS_CONTROLLER_ADMIN`
- `XS_CONTROLLER_ADMIN_READ_ONLY`
- `XS_CONTROLLER_AUDITOR`
- `XS_CONTROLLER_CREDENTIALS_VIEWER`
- `XS_CONTROLLER_GLOBAL_AUDITOR`
- `XS_CONTROLLER_USER`

To overcome the bootstrap problem when an XS advanced application server is installed, a single administrative XS advanced Controller user (named `XSA_ADMIN` by default) is created. This user has the XS advanced Controller role collection `XS_CONTROLLER_ADMIN`, which comprises all three XS advanced Controller scopes. This means that `XSA_ADMIN` can use the XS advanced Controller without any restrictions and is in a position to do the initial setup of the model, that is, appoint at least one `OrgManager` who is able to set up the spaces. Global resources like buildpacks or external brokers can also only be managed by an administrative Controller user.

> → Recommendation
>
> After you have finished the initial setup of the system, deactivate the bootstrap administrative user `XSA_ADMIN` with the following SQL statement:
>
> ```
> ALTER USER XSA_ADMIN DEACTIVATE USER NOW
> ```

In an emergency, a user with system privilege `USER ADMIN` can reactivate this user with the following SQL statement:

```
ALTER USER XSA_ADMIN ACTIVATE USER NOW
```

The role collection `XS_CONTROLLER_USER` is designed for typical Controller users such as developers who work in one or more spaces (or even at organization level), reading and modifying their resources. Note that such users additionally need a so-called **Controller role** to gain access to a specific organization or space. If you want a user to have only read privileges, for example to audit some parts of the system, assign the role collection `XS_CONTROLLER_AUDITOR`.

> ### Note
>
> Having the role collections `XS_CONTROLLER_USER` or `XS_CONTROLLER_AUDITOR` assigned is just the prerequisite for making a user a Controller user. As these role collections do not scope the Controller resources that the user may access, an additional Controller role is required to fill the gap (see *Controller Role Model*).

The following table gives an overview of all available standard role collections for the Controller, supplemented by the corresponding scopes and the permitted operations.

Role Collection	Scope(s)	Permitted Operations

`XS_CONTROLLER_ADMIN`	`cloud_controller.admin`
	`cloud_controller.write`
	`cloud_controller.read`
		Unlimited access to the XS advanced Controller API

`XS_CONTROLLER_ADMIN_READ_ONLY`	`cloud_controller.admin_read_only` `cloud_controller.read`	Read-only access to the XS advanced Controller API
`XS_CONTROLLER_USER`	`cloud_controller.write` `cloud_controller.read`	Read or write XS advanced Controller resources (global resources excluded from modifications)
`XS_CONTROLLER_GLOBAL_AUDITOR`	`cloud_controller.global_auditor` `cloud_controller.read`	Read-only access to all XS advanced Controller resources except system resources and sensitive data
`XS_CONTROLLER_AUDITOR`	`cloud_controller.read`	Read access to Controller resources
`XS_CONTROLLER_CREDENTIALS_VIEWER`	`cloud_controller.credentials.view` `cloud_controller.read`	With the necessary resource privileges, read access to sensitive data even if restricted access mode is activated.

Related Information

Authorization Management Tools [page 371]
Controller Role Model [page 367]

17.3.3 Controller Role Model

Controller role collections are associated with essential authorizations like read and write permissions, but they do not control the permission to access a specific resource.

Resources in the Controller are entities such as:

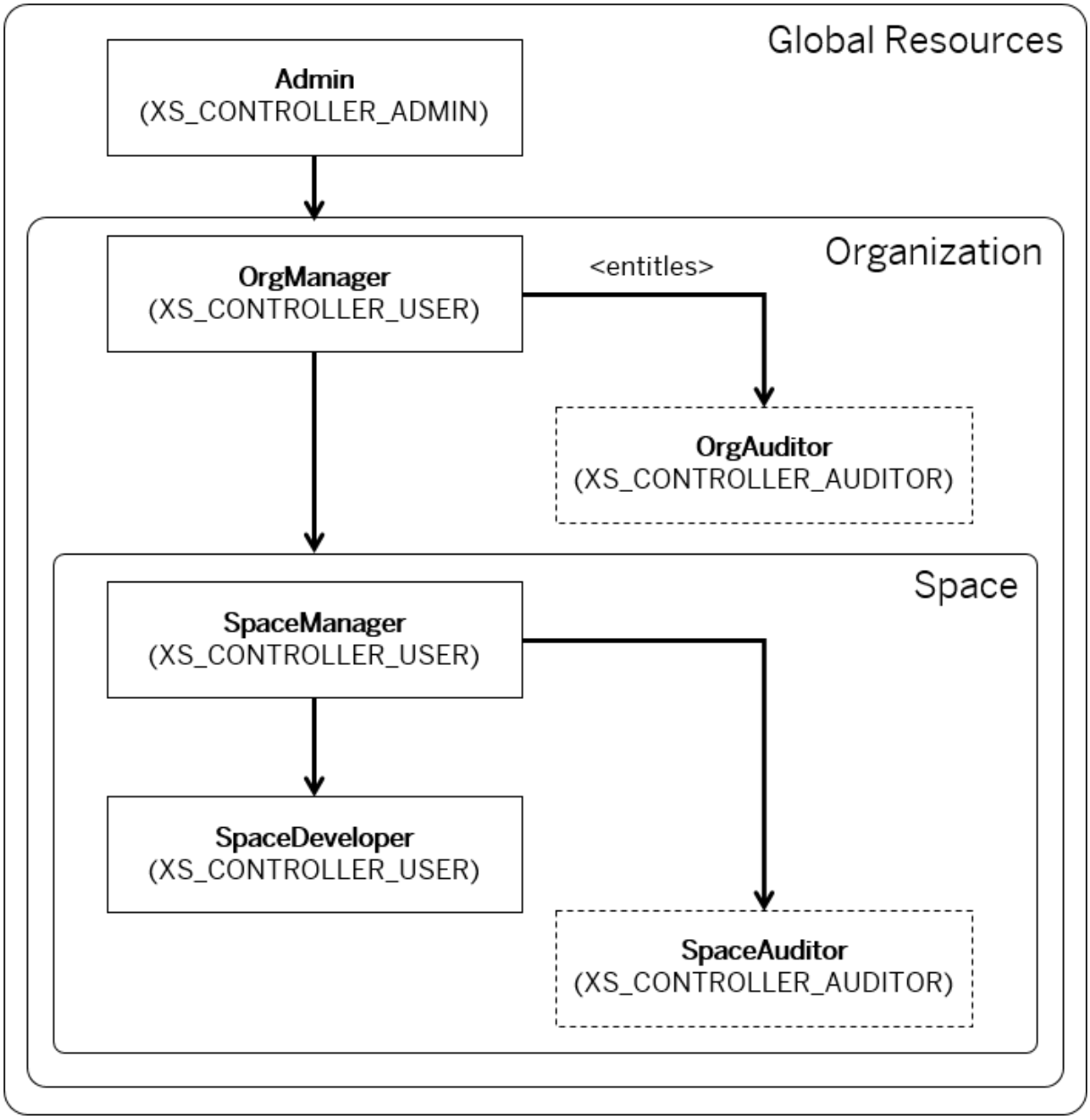
- Applications
- Service instances and bindings
- Domains and routes
- Services and plans making up the marketplace of a service broker
- Spaces and organizations
- Service brokers
- Buildpacks

Spaces are a central concept for grouping applications that are tightly coupled and can run in a shared trust zone. Organizations simply embrace spaces at a higher level. Spaces not only determine the trust zones during runtime, but also provide a way to define the XS advanced users that should be allowed to manage the space's resources collectively. Which resources can be assigned to a space?

Each Controller resource has a reference to a space (applications, service instances, bindings, and so on) and therefore is scoped to this space, or it is designated as a global resource (service broker, buildpacks and so on). This is where Controller roles come into play. A Controller role is granted to a Controller user for a specific space or organization. Information about which roles are granted to a Controller user is not stored in the UAA, but attached to the space or organization entities in the Controller model. Five different categories of Controller roles are defined.

Controller Role	Resource Scope	Permitted Operations Within Scope
`OrgManager`	Organization	• Create a new space or update an existing space within the organization • Create or change a domain for the organization (not a shared domain) • View all Controller resources within the organization like spaces, apps and so on (including credentials) • Grant `OrgManager`, `OrgAuditor`, `SpaceManager`, `SpaceDeveloper` or `SpaceAuditor` to other platform users
`OrgAuditor`	Organization	• View all Controller resources within the organization like spaces, apps etc. (excluding credentials)
`SpaceManager`	Space	• Grant `SpaceManager`, `SpaceDeveloper`, `SpaceAuditor` to other platform users
`SpaceDeveloper`	Space	• Create or change resources within the space like apps, service instances or service bindings • Authorized to push an application to the space
`SpaceAuditor`	Space	• View the resources of a space like apps, application logs, application environments (excluding credentials)

Controller roles are hierarchical as depicted in the following diagram (dotted lines denote read access only):



Controller Role Model

When a user submits a request to the Controller, his or her user ID and scopes are extracted from the OAuth2 access token as a first step. If a non-global resource is requested, the Controller then checks the user list for the resource's space (or organization).

Having extracted the Controller scopes and space or organization role, the Controller finally allows authorization according to following rules:

- Global resources without reference to a space or organization can only be modified by users with scope `cloud_controller.admin`. `cloud_controller.read` is sufficient for viewing global resources.
- A Controller user (that is a user with some Controller scope) who has been granted a Controller role to a specific space (or organization) may access all resources in this space (or organization) according to the assigned Controller role (for example, Space Developers may start an application in the space,

Space Auditors may only view this application). If the user has only `cloud_controller.read` scope, no resource may be modified.

The following list shows the scope of some resource types together with xs CLI commands.

Controller Resource	Resource Scope	xs Commands	Minimum Authorization
Organization	Organization	`orgs`, `create-org`, `delete-org`	`admin`
Domain	Organization	`domains`, `create-domain`, `update-domain`, ...	`OrgManager`
User	Organization	`set-org-role`, `unset-org-role`	`OrgManager`
Space	Space	`create-space`, `update-space`, ...	`OrgManager`
Application	Space	`apps`, `push`, `scale`, `delete`, `start`, `stop`, ...	`SpaceDeveloper`
Route	Space	`create-route`, `delete-route`, ...	`SpaceDeveloper`
Service instance	Space	`create-service`, `delete-service`, ...	`SpaceDeveloper`
User-provided service instance	Space	`create-user-provided-service`, `delete-user-provided-service`, ...	`SpaceDeveloper`
Service key	Space	`create-service-key`, ...	`SpaceDeveloper`
Service binding	Space	`bind-service`, `unbind-service`, ...	`SpaceDeveloper`
User	Space	`set-space-role`, `unset-space-role`	`SpaceManager`
Buildpack	<global>	`create-buildpack`, ...	`admin`
Runtime	<global>	`create-runtime`, ...	`admin`
Service broker	<global>	`create-service-broker`, ...	`admin`

Controller Resource	Resource Scope	xs Commands	Minimum Authorization
Service URL	<global>	`register-service-url`, `unregister-service-url`	`admin`

For more information about xs commands, see *The XS Command-Line Interface Reference* in the SAP HANA Developer Guide (For SAP HANA XS Advanced Model).

Related Information

The XS Command-Line Interface Reference

17.3.4 Authorization Management Tools

Various command line and UI tools can be used for user and authorization management.

SAP HANA XS Advanced Cockpit

The SAP HANA XS Advanced cockpit is an XS Advanced application that is provided as additional content on the SAP HANA installation medium. Among other things, it provides a comfortable way to handle the following tasks:

Task	Navigation
Create or delete users	*User Management*
Assign existing role-collections to users	*User Management*
Create custom role-collections based on deployed role-collection templates	*Security* > *Role Collections*
Manage organizations and spaces	*Organizations* > `<organization>/<space>` > *Members*
Assign Controller roles (SpaceDeveloper, SpaceManager, etc.) to Controller users	*User Management*

> **Note**
>
> Users who need full access to the *User Management* tool need the role collection `XS_USER_ADMIN`. To view user settings only, `XS_USER_DISPLAY` is sufficient. Please note that the initial administrative user `XSA_ADMIN` has both role collections after installation.

For more information about the *XS Advanced Administration and Monitoring Tools*, see the section on maintaining the SAP HANA XS advanced model runtime in the *SAP HANA Administration Guide*.

SAP HANA SQL Interface

If the UAA's identity provider is SAP HANA itself (default), the established SAP HANA user-management tools can be used to create XS advanced users for both business users and system users. The SQL statement `CREATE RESTRICTED USER <HANA_USER>` creates a user without initial privileges in SAP HANA. Restricted SAP HANA users are sufficient as these users won't access SAP HANA artifacts directly: applications access SAP HANA with technical users on XS advanced users' behalf.

You can make a standard SAP HANA user into a Controller user with the following SQL statement (assuming you have the corresponding privileges):

```
ALTER USER <HANA_USER> SET PARAMETER XS_RC_XS_CONTROLLER_USER='XS_CONTROLLER_USER'
```

You can revoke Controller role collections with:

```
ALTER USER <HANA_USER> CLEAR PARAMETER XS_RC_XS_CONTROLLER_USER
```

xs Command Line Client

This tool is available in each standard XS advanced installation and is located in the `/xs/bin` directory of the installation (`/hana/shared/<sid>/xs/bin/xs` by default). You cannot use this tool to create new XS advanced users, but you can use it to view and manage Controller roles of users that have already been granted Controller role collections. Controller users must first be created using another tool, preferably the *User Management* application.

The following table provides a list of all relevant xs client commands for user management:

Command	Description
`users`	List all users
`purge-users`	Delete all users in Controller who are not known to the User Account and Authentication (UAA) service [-f]
`space-users`	Show space users by role
`set-space-role`	Assign a space role to a user

Command	Description
`unset-space-role`	Revoke a space role from a user
`org-users`	Show organization users by role
`set-org-role`	Assign a organization role to a user
`unset-org-role`	Revoke a organization role from a user
`roles`	Display a list all existing application roles
`role-collections`	Display a list of all existing application role collections
`role-collection`	Display details of a specific application role collection
`create-role-collection`	Create a new application role collection
`update-role-collection`	Modify an existing application role collection
`assigned-role-collections`	Display a list of the application role collections currently assigned to a specific user
`assign-role-collection`	Assign an application role collections to a specific user
`unassign-role-collection`	Remove an assigned application role collection from a specific user

> **Note**
>
> The xs command line client cannot be used to change the initial password of SAP HANA users. So, the first time you log on with a newly created SAP HANA user, you get a warning message demanding a password change. The password can be changed on the UAA login page. The URL of UAA's login page can be extracted with the command `xs version`.

For more information about xs commands, see the XS command-line interface reference in the *SAP HANA Developer Guide (For SAP HANA XS Advanced Model)*.

Related Information

Maintaining the SAP HANA XS Advanced Model Run Time
The XS Command-Line Interface Reference

17.4 Network and Communication Security with SAP HANA XS Advanced

Security mechanisms are applied to protect the communication paths used by the SAP HANA XS advanced server infrastructure. SAP provides network topology recommendations to restrict access at the network level.

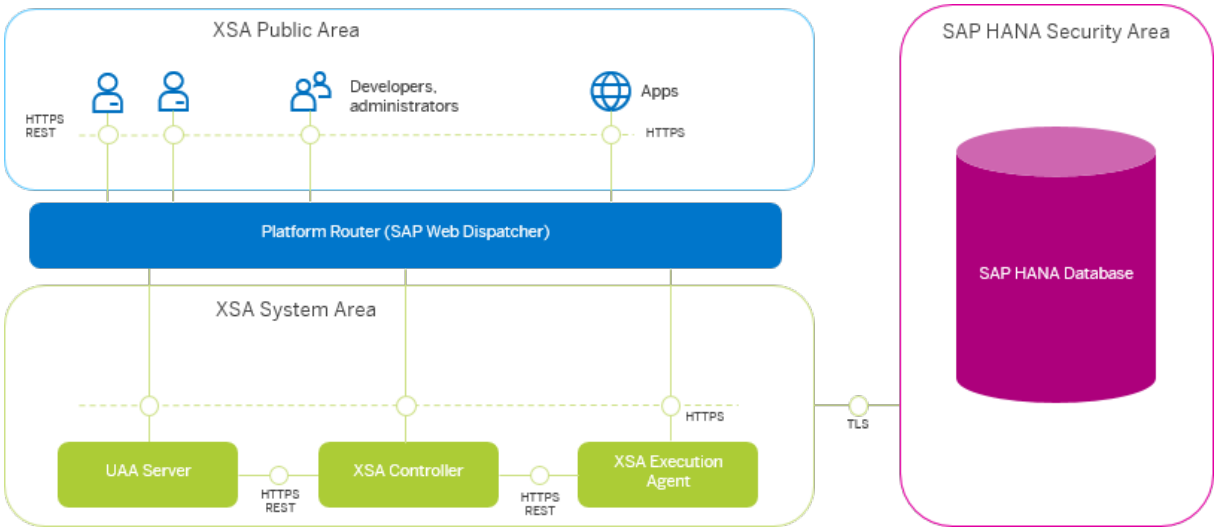
Related Information

Security Areas [page 374]
Public Endpoints [page 375]
Single-Host Scenario [page 377]
Multiple-Host Scenario [page 378]
XS Advanced Certificate Management [page 380]

17.4.1 Security Areas

Three different logical security areas can be identified in the XS advanced application server infrastructure. These cover communication channels that are all secured by TLS/SSL.

The different areas are characterized in the way how certificate management is done. As you can see in the figure below, the areas are referred to as the **XS Advanced Public** area, the **XS Advanced System** area, and the **SAP HANA Security** area.



XS Advanced Security Areas

The XS Advanced Public area is managed by an XS advanced administrator who configures the connections between clients and the public endpoints of applications and server components like the Controller or UAA. Inter-application communication is also related to this area because when another application is called, the

request has to be sent to the public endpoint of the application exposed by the Platform Router. The XS advanced administrator is able to deploy custom SSL certificates for application domains or the system's administration domain. By default, the platform provides self-signed certificates for these endpoints. For more information about how this is accomplished, see *XS Advanced Certificate Management* in *Related Information* below.

The XS Advanced System area covers all internal communication channels between application server components like Controller, Execution Agents, UAA, Platform Router, and so on. These channels are secured automatically with TLS/SSL using the SAP HANA system public key infrastructure (system PKI), which provides mutual authentication. The SSL certificates are only used for internal purposes and are never exposed to other areas, neither to applications nor to any XS advanced clients. The required certificates are configured in the SAP HANA database

The SAP HANA Security area includes TLS/SSL-secured connections between the SAP HANA database and XS advanced applications, as well as between the database and server components.

> ⚠ Caution
>
> The connection to the SAP HANA database is **not** encrypted by default. To activate TLS/SSL, custom SSL certificates need to be configured as described in section *Certificate Management*.

Related Information

XS Advanced Certificate Management [page 380]

17.4.2 Public Endpoints

The number of public endpoints directly depends on the configured routing mode.

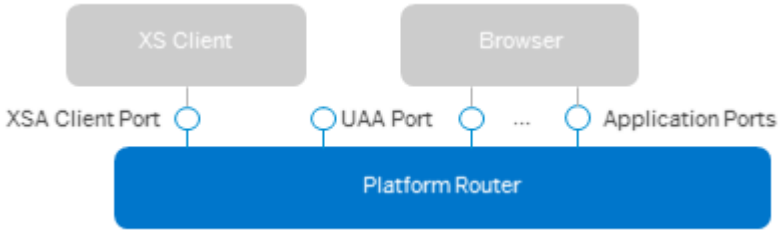
The XS advanced application server supports two routing modes: port routing and hostname routing. With port routing, the endpoints need an own port. In contrast, hostname routing requires only a single port but additional domain name server (DNS) entries. As URLs in the hostname routing scenario are user friendly and there is only a single public port, this mode is recommended for production usage.

> → Recommendation
>
> To provide a high level of protection to your system, a firewall should reject all client requests against non-public ports.

Public Endpoints with Port Routing

With port routing, the endpoint's URLs are composed of the shared domain along with a dedicated port: `https://<shared-domain>:<port>`. Routing to the back-end component is solely based on this port. To avoid clashes with other systems, the port numbers are derived from the instance number of the system. Port

routing is suitable if you don't want to or can't edit the DNS configuration. It works without additional manual effort and is therefore the installation default.



Port Routing

The table below shows the public ports of the Platform Router as they are exposed to clients.

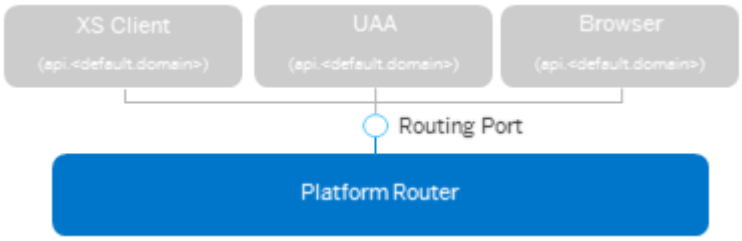
Endpoint	Protocol	Authentication	Port(s)
Controller (public)	HTTPS REST	Server	3<instance_no>30
UAA (public)	HTTPS REST/WEB	Server	3<instance_no>32
Applications (public)	HTTPS	Server	51000 – 51500

> ⚠ Caution
>
> Some browser versions do not distinguish between different server ports in cookie handling. Port-based routing is therefore not recommended for use in production systems. Moreover, this mode occupies many more ports compared to hostname routing mode.

> ⚠ Caution
>
> Some browsers are not able to distinguish between the cookies for different applications. Depending on the scenario, this might be a security issue.

Public Endpoints with Hostname Routing

With hostname routing, only the single public port 3<instance_no>33 of the Platform Router is required. In contrast to port routing, the routing to the internal back-end components is entirely based on the URL's host specification provided with the request. The host names are derived from the routes that are created during application deployment. A route contains the application's name by default, which is complemented by the default domain as suffix. Administrators may add custom shared domains, Org Managers are allowed to create domains for their specific organization.



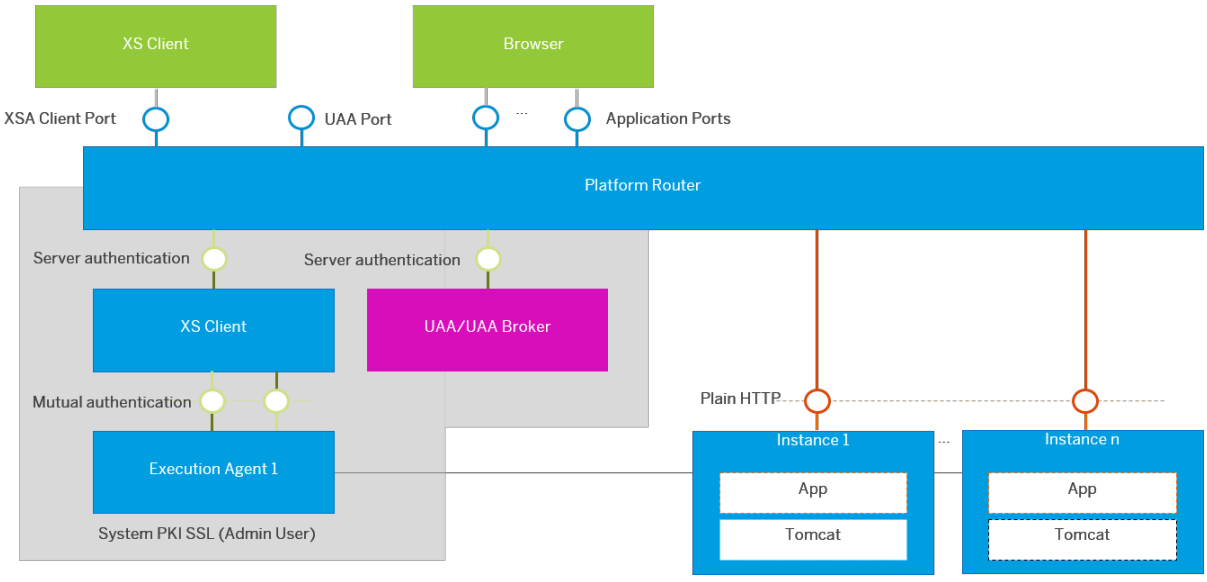
Routing Modes

The following table lists the URLs of system components:

Endpoint	Authentication	URL
Controller (public)	Server	`https://api.<default-domain>:3<instance_no>33`
UAA (public)	Server	https://uaa-server.`<default-domain>`:3`<instance_no>`33"
Applications (public)	Server	https:// `<hostname>.<domain>`:3`<instance_no>`33

17.4.3 Single-Host Scenario

In a single-host system, internal communication between the Platform Router back-end and the application instances is not secured. Since application instances are not part of the internal system public key infrastructure (PKI), the Platform Router cannot authenticate them. However, given that the communication is bound to the local host only, the data transfer can be considered secure.



XS Advanced Single-Host Scenario

Private Endpoints

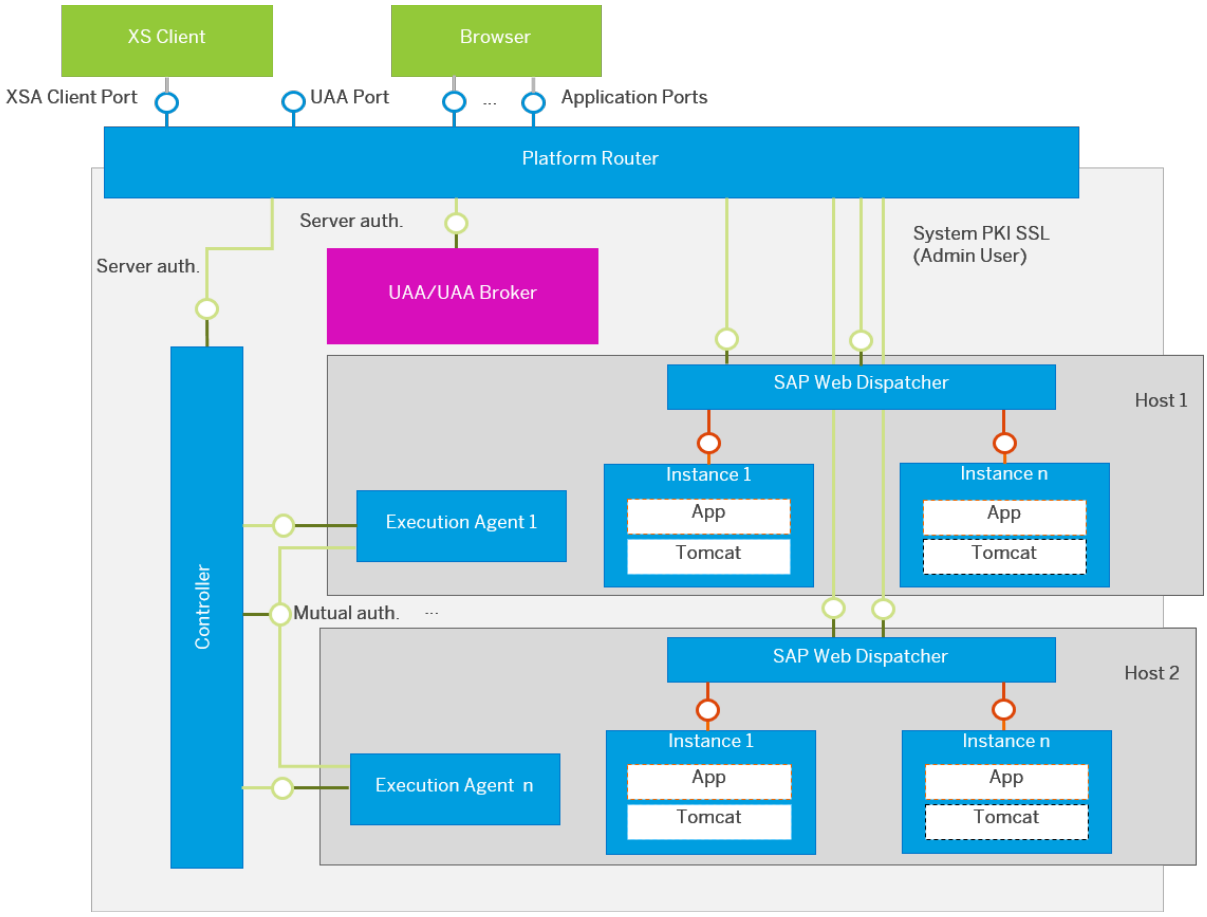
Server components and application instances expose ports that are not designed for external communication. The following tables lists all private ports:

Endpoint	Protocol	Authentication	Port(s)
Controller	HTTPS REST	Server (system PKI)	Within 51000 – 51500 (only local)
Controller (for Execution Agent)	HTTPS REST	Mutual (system PKI)	3<instance_no>29
Execution Agent(s)	HTTPS REST	Mutual (system PKI)	free
Application instances	HTTP	Client (OAuth2)	50000 – 50999
UAA	HTTPS REST/WEB	Server (system PKI)	3<instance-no>31 (only local)

17.4.4 Multiple-Host Scenario

In a multiple-host scenario, the Platform Router and the application instances typically run on different hosts. Due to the fact that the connection is based on HTTP, the data transferred to the application instances could be compromised. To solve this problem, each Execution Agent manages an additional SAP Web Dispatcher instance, which bridges the gap between the Platform Router host and the host of the application instance. Since both the Platform Router and the SAP Web Dispatcher instances are integrated into the system public key infrastructure (PKI), the channel between them is protected.

The following diagram gives an overview of this setup:



XS Advanced Multiple-Host Scenario

As you can see above, there are slightly more private ports in use than in the single-host scenario. Especially the number of application ports needs to be doubled because each application port has to be exposed firstly, by the instance itself and secondly, by the additional SAP Web Dispatcher on each host with an Execution Agent.

Private Endpoints

In contrast to the single-host scenario, the internal port range for application is shared by the host router and application instances. Similarly to the single-port scenario, protecting private ports with an adequate firewall is strongly recommended.

Endpoint	Protocol	Authentication	Port(s)
Controller	HTTPS REST	Server	Within 51000 – 51500 (only local)

Endpoint	Protocol	Authentication	Port(s)
Controller (for Execution Agent)	HTTPS REST	Mutual	3<instance_no>29
Execution Agent(s)	HTTPS REST	Mutual	free
Application instances	HTTP	Client (OAuth2)	50000 – 50499
Host Web Dispatcher	HTTPS	Server	50500 – 50999
UAA	HTTPS REST/WEB	Server	3<instance_no>31 (only local)

17.4.5 XS Advanced Certificate Management

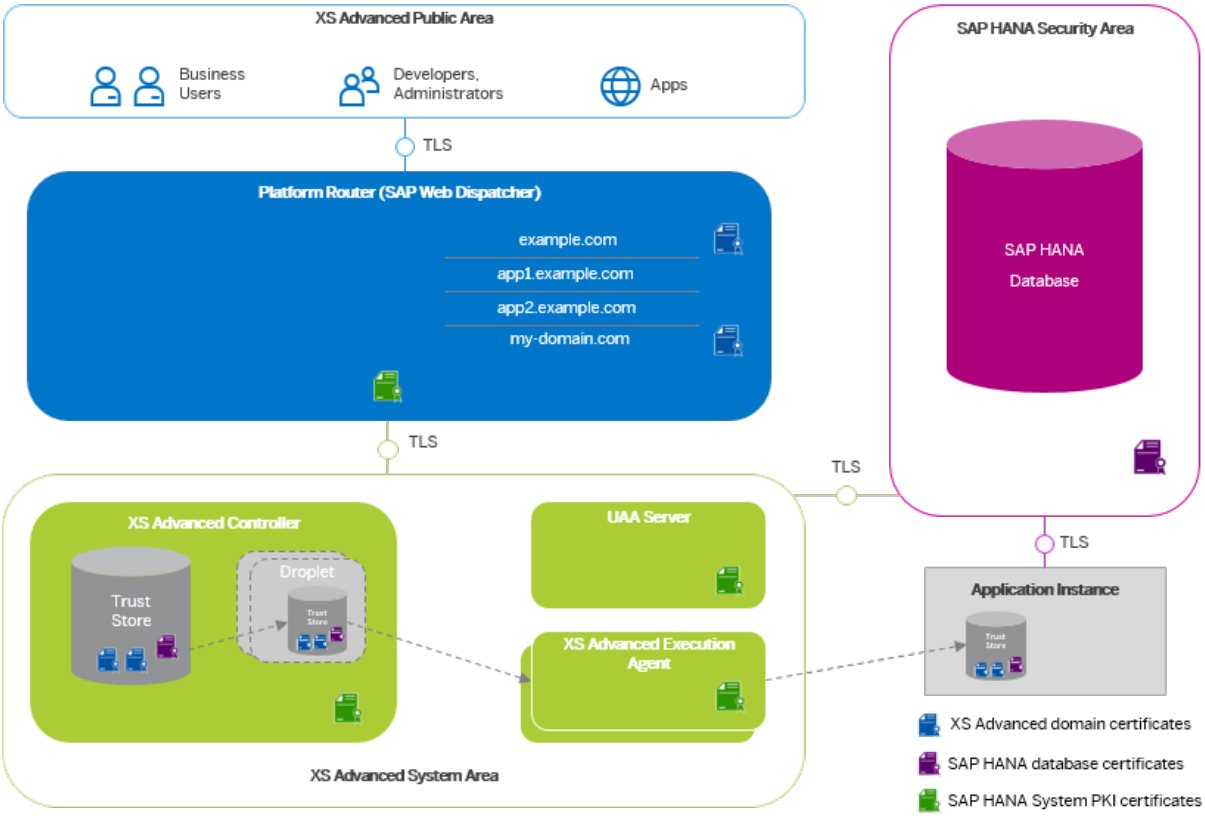
The three security areas in the XS advanced server infrastructure have a slightly different certificate management. The Controller is the central instance that performs global certificate management, providing the necessary trust certificates for the corresponding components.

XS Advanced Public Area

In the XS advanced Public area the XS advanced administrator is responsible for deploying the domain-specific certificates. These can be either self-signed or issued by the global certificate authority (CA). The certificates can be deployed in the xs client using the `set-certificate` command. This is explained in detail in SAP Note 2243019. However, by default, the system generates self-signed certificates that the administrator can manually and securely distribute among the clients.

> **Note**
>
> In production XS advanced installations, replace the self-signed certificate with one trusted in your organization.

The distribution of private keys and their certificates in the XS advanced server environment is illustrated in the figure below. The Platform Router is totally managed by the Controller, which means that each time the administrator deploys a certificate for the specified domain (for example, by submitting `xs set-certificate DOMAIN -k KEY_FILE -c CERT_FILE`), the Controller stores the certificates in its trust store and adapts the Platform Router configuration accordingly. Due to this approach, the Controller is aware of all custom certificates and is therefore able to authenticate all external endpoints exposed by the Platform Router. On the other hand, the Controller uses its trust store for passing it to the application instances in order to allow them to authenticate the Platform router endpoints as well.



Certificate Management in XS Advanced Public Area

A security-critical scenario can arise if the certificate expires. In this case, the Controller cannot authenticate the Platform Router anymore and aborts start-up. To solve this problem, the administrator has to apply new certificates by executing the XSA `set-certificate` or `reset-certificate` commands as a `<sid>`adm user.

For more information about `xs` commands, see *The XS Command-Line Interface Reference* in the *SAP HANA Developer Guide (For SAP HANA XS Advanced Model)*.

XS Advanced System Area

The system administrator does not need to perform any configuration steps for the XS advanced system area. The internal system PKI is responsible for certificate management in this area. Each component within this infrastructure gets its own private certificate, which is signed by the root CA of the system. In this case, mutual authentication of these components is assured. This kind of certificate is never exposed to external clients.

SAP HANA Security Area

The XS advanced administrator is also responsible for the certificate management in the SAP HANA security area. The XS advanced platform does not encrypt database connections to SAP HANA out of the box.

Therefore, trust certificates for database connections must be configured in XS advanced as explained in SAP Note 2300943. The following steps are required:

- Deployment of the custom certificate in the SAP HANA database in order to provide the certificate to all index server instances
- Publishing of the certificate to the XS advanced application server components (`xs trust-certificate` ...)
- Enabling of SSL for database connections in xscontroller.ini and restarting the system

More Information

For more information about certificate management in the context of XS advanced, see the following documentation:

- For enabling SSL encryption for database connections for SAP HANA extended application services, advanced mode, see SAP Note 2300943
- For setting up certificates in an XS advanced landscape with reverse proxy, see the section describing the installation of XS advanced in the *SAP HANA Server Installation and Update Guide*
- For configuring allowed TLS versions and cipher suites for internal and external communication, see the section describing the configuration of the XS Advanced Platform router in the *SAP HANA Administration Guide*

Related Information

Configuring the XS Advanced Platform Router
Installing an SAP HANA System Including the XS Advanced Runtime
SAP Note 2243019
SAP Note 2300943

17.5 Data Storage Security

Security mechanisms are applied to protect critical data managed by the SAP HANA XS advanced model infrastructure.

Most system components need to persist data provided by end users. The Controller stores application-related data (for example, its design time artifacts), as well as staged droplets ready for execution. Moreover, the Controller model, which is structured by organizations and spaces, may be modified by privileged users. Bindings to service instances typically bear credentials that also need special attention as they are not expected to be stored as plain text in SAP HANA tables. Similarly, the central UAA instance makes usage of a storage to which user information and credentials are written in a secure manner.

Finally, some system components, or to be more precise the standard service brokers, provide database and file system storage for applications at deployment time.

Related Information

System Component Storage [page 383]
Application Storage [page 384]

17.5.1 System Component Storage

Controller Storage

As part of deployment, application files are uploaded to the Controller and subsequently written to the **BlobStore**. The BlobStore is optimized to store file contents (Blobs) in an efficient manner, avoiding redundancy in cases where the very same file content is used in different contexts (for example, the same JAR file is shared by different Java-based applications). This file store is located in the SAP HANA schema, which is owned by the Controller's technical SAP HANA user `SYS_XS_RUNTIME`. When a resource is requested, only the Blobs that are referenced by this resource are available for download. As different Controller resources may share the same Blob in this store, a modification will result in a new Blob.

> **Note**
>
> Only administrative Controller users (for example, `XSA_ADMIN` or users with role collection `XS_CONTROLLER_ADMIN`) have full BlobStore access. Some commands in the xs command line client therefore need administrative privileges.

Controller resources, such as application or buildpack metadata, are written to the ConfigStore, which also resides in the Controller's database schema. User requests that need to fetch data from ConfigStore are authorized by the mechanisms described in the section *Controller Role Model*.

The following sensitive Controller data is stored encrypted in SAP HANA secure store:

Data	Content
Service broker	`broker credentials`
Service binding	`service credentials`
Service key	`service credentials`
User-provided service instance	`instance data`
Application environment	`environment variables`

To access the API of an (external) service broker, for example when a new service instance is requested, basic authentication is required (`auth_username` and `auth_password`). Service binding entities keep the credentials to access the offered services created by a service broker. A prominent example is the credentials of the technical SAP HANA user for a HDI container that has been created by the SAP HANA Service

Broker. In the case of user-provided service instances, you may want to make explicit credentials available for applications.

UAA and UAA Broker Storage

Similar to Controller storage, information stored by the UAA or the UAA Broker is separated according to security relevance. User information, user secrets, and tokens are written encrypted to an SAP HANA secure store with the technical user SYS_XS_UAA_SEC. However, common metadata like the scope, role, and attribute definitions are kept in standard SAP HANA tables in the schema of SYS_XS_UAA.

Related Information

Controller Role Model [page 367]

17.5.2 Application Storage

By default, applications may consume two different kinds of storage provided by the XS advanced application server: SAP HANA storage and file-system storage. Both are requested during application deployment when the SAP HANA Service Broker or the File-system Broker are used.

SAP HANA Service Broker

The SAP HANA Service Broker offers different service plans to match various customer needs:

- `hdi-shared` provides a full HDI container with a technical user
- `schema` provides a plain SAP HANA schema with a technical user
- `securestore` provides an SAP HANA secure store to write encrypted data
- `sbss` (Service Broker Security Support) provides an SAP HANA schema with procedures to generate secure passwords

File-system Broker

When a service instance of the File-system Broker is created (for example, by calling `xs create-service`) within a specific space, a new directory on a dedicated file system is created. This directory is configured with exclusive access rights for the OS user attached to the space of the service instance. In this way, it is guaranteed that the directory is only visible to applications within the same space (or to applications within a space having the same OS user).

17.6 Security-Relevant Logging and Tracing

Auditing makes it possible to trace who has performed which kinds of operations in the XS advanced system and applications.

System audit logs may help you to detect undesired modifications that could be the result of a misconfiguration in the user authorization setup. It could also uncover attempts to breach the system security.

For application auditing, the audit-log service writes audit log events received from applications to SAP HANA auditing.

17.6.1 Audited Operations

Several operations are automatically audited.

By default, the system logs all operations submitted to the Controller endpoint:

- Read operations for all Controller resources like applications, spaces, and buildpacks
- Update operations for all Controller resources
- Create and delete operations for all Controller resources
- Starting and stopping of application instances
- Settings like tracing, backup requests, SSL certificate uploads and so on

The UAA service additionally logs:

- Login activities
- Issuing of access tokens
- All UAA Broker operations, for example, creating a new service instance or binding an application
- Configurations like changing the identity service provider

For each requested operation, a new log line will appear in the audit log containing:

- The name of the user who triggered the operation
- The time stamp the operation was requested
- A short description of the operation containing the affected resources

17.6.2 Audit Trails

Audit logs are written to `<sid>adm` user's tracing file system.

Service	Audit Log Location
Controller	`/trace/hdbxscontroller_audit.log`

Service	Audit Log Location
Execution Agent	`/trace/hdbxsexecagent_audit.log`
UAA	`/trace/uaa-audit.log` `/trace/uaa-oidc-audit.log`

Audit trails can be easily inspected in the SAP HANA cockpit.

If you enable the SAP HANA audit log, XS advanced stores audit logs from the XS Controller and the Execution Agents service in the SAP HANA audit log. Audit logs created by the UAA service are not stored in the SAP HANA audit log. For more information about enabling the SAP HANA audit log and configuring the correct audit policy, see *Application Auditing* in *Related Information*.

Syslog Support

The Linux operating system provides a comprehensive logging system called `syslog`. It is highly flexible, provides integration possibilities, and is therefore also provided for audit logs in the XS advanced system. In order to ensure that system components write their audit logs to `syslog`, set the parameter `syslog` in the section `audit` of the Controller's `xscontroller.ini` file to `true`.

For more information about how to configure `syslog`, refer to the documentation for your operating system.

Related Information

Application Auditing [page 386]

17.6.3 Application Auditing

Deployed applications can be configured to use the audit-log service of the SAP HANA XS advanced platform. The audit-log service writes audit log events received from applications to SAP HANA auditing. The XS Controller and the Execution Agent service use SAP HANA auditing as well to store audit logs.

To activate SAP HANA auditing for all events from an XS advanced application bound to the audit-log service and all events from the XS Controller and Execution Agents, create an audit policy with the following audit actions on all objects:

- CONFIGURATION CHANGE
- PERSONAL DATA ACCESS
- PERSONAL DATA MODIFICATION
- SECURITY EVENT

You can create an audit policy using SQL, as shown in the following example:

> **Note**
>
> The SQL statements must be run by a user with the system privilege `AUDIT ADMIN`.

> **Sample Code**

```
CREATE AUDIT POLICY xsa_policy AUDITING ALL SECURITY EVENT, PERSONAL DATA
ACCESS, PERSONAL DATA MODIFICATION, CONFIGURATION CHANGE LEVEL INFO;
ALTER AUDIT POLICY xsa_policy ENABLE;
```

Alternatively, you can use the *Auditing* app in the SAP HANA cockpit.

The audit log entries written by XS advanced applications and services can be viewed by querying the system view XSA_AUDIT_LOG. System privilege `AUDIT OPERATOR` or `AUDIT ADMIN` is required.

Related Information

Configure an Application to Use the Audit Log Service
Create an Audit Policy
System Properties for Configuring Auditing [page 295]
Audit Trail Layout for Trail Target Database Table [page 290]
CREATE AUDIT POLICY Statement (Access Control)

17.7 Data Protection and Privacy in SAP HANA XS Advanced

As an application platform server, the SAP HANA extended application server, advanced model (XS advanced), provides operators with a selection of tools and functions to conform to the legal requirements of data protection concerning data processed by deployed applications, as well as the server infrastructure itself.

Introduction to Data Protection

Data protection is associated with numerous legal requirements and privacy concerns. In addition to compliance with general data privacy regulations, it is necessary to consider compliance with industry-specific legislation in different countries. SAP HANA XS advanced provides tools and functions to support compliance with regard to relevant legal requirements, including data protection.

This section and any other sections in this Security Guide do not give any advice on whether these features and functions are the best method to support company, industry, regional, or country-specific requirements. Furthermore, this guide does not give any advice or recommendations regarding additional features that would be required in specific IT environments; decisions related to data protection must be made on a case-by-case basis, taking into consideration the given system landscape and the applicable legal requirements.

> **Note**
>
> In the majority of cases, compliance with data privacy laws is not a product feature. SAP software supports data privacy by providing security features and specific functions relevant to data protection, such as functions for the simplified blocking and deletion of personal data. SAP does not provide legal advice in any form. The definitions and other terms used in this guide are not taken from any given legal source.

Glossary

Term	Definition
Blocking	A method of restricting access to data for which the primary business purpose has ended.
Business purpose	A legal, contractual, or in other form justified reason for the processing of personal data. The assumption is that any purpose has an end that is usually already defined when the purpose starts.
Consent	The action of the data subject confirming that the usage of his or her personal data shall be allowed for a given purpose. A consent functionality allows the storage of a consent record in relation to a specific purpose and shows if a data subject has granted, withdrawn, or denied consent.
Deletion	Deletion of **personal data** so that the data is no longer available.
End of purpose (EoP)	End of purpose and start of blocking period. The point in time, when the primary processing purpose ends (e.g. contract is fulfilled).
End of purpose (EoP) check	A method of identifying the point in time for a data set when the processing of **personal data** is no longer required for the primary **business purpose**. After the **EoP** has been reached, the data is **blocked** and can only be accessed by users with special authorization (for example, tax auditors).
Personal data	Any information relating to an identified or identifiable natural person ("data subject"). An identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural, or social identity of that natural person
Residence period	The period of time between the end of business and the end of purpose (EoP) for a data set during which the data remains in the database and can be used in case of subsequent processes related to the original purpose. At the end of the longest configured residence period, the data is blocked or deleted. The residence period is part of the overall retention period.
Retention period	The period of time between the end of the last business activity involving a specific object (for example, a business partner) and the deletion of the corresponding data, subject to applicable laws. The retention period is a combination of the residence period and the blocking period.

Term	Definition
Sensitive personal data	A category of personal data that usually includes the following type of information: • Special categories of personal data, such as data revealing racial or ethnic origin, political opinions, religious or philosophical beliefs, trade union membership, genetic data, biometric data, data concerning health or sex life or sexual orientation, or personal data concerning bank and credit accounts. • Personal data subject to professional secrecy • Personal data relating to criminal or administrative offenses • Personal data concerning insurances and bank or credit card accounts
Where-used check (WUC)	A process designed to ensure data integrity in the case of potential blocking of business partner data. An application's where-used check (WUC) determines if there is any dependent data for a certain business partner in the database. If dependent data exists, this means the data is still required for business activities. Therefore, the blocking of business partners referenced in the data is prevented.

Scope of This Documentation

The XS advanced platform functions presented in this section only deal with aspects of protecting data that is managed by the platform itself, for example residing in server trace files or application logs. The XS advanced platform may be used to process data stored in other systems by leveraging communication capabilities with other data sources.

For aspects of data protection and privacy related to the SAP HANA database as basis of the platform stack, see the section *Data Protection in SAP HANA*.

General security requirements that are mandatory for data protection and privacy are covered elsewhere in this section on security for XS advanced. These include for example data-access control and logging, as well as communication security.

Personal Data in SAP HANA XS Advanced Applications

Applications deployed to the SAP HANA XS advanced server may process and store information related to an identified or identifiable natural person. Depending on the way a person interacts with the system, different server artifacts might be affected. For example, personal data of an internal platform user (for example, a Controller user) might be traced to platform trace logs or audit logs that are necessary to generally meet specific quality standards of the platform; whereas personal data of business users might appear in application logs controlled by the platform or even be stored to application-controlled services like an SAP HANA schema.

There are three categories of artifact to which personal data may be related:

- **Category 1: Platform-controlled artifacts like audit logs and server trace files**
  Personal data originates from platform users. The data are required to guarantee operation of the platform.
- **Category 2: Platform-controlled artifacts like application log files or audits**

Personal data originates from business users and is transparent for the platform. These services are provided by the platform to support the operation of applications. With respect to personal data, they may be used by applications only for this purpose.

- **Category 3: Application-controlled artifacts like a persistence service or file system service**
  Personal data originates from business users and is transparent for the platform.

Compliance with any existing data privacy law is not a product feature of the XS advanced application server. However, the XS advanced application server provides a variety of security-related features to implement general security requirements that can also be used by administrators for data protection and privacy.

By default, potential personal data stored in any of the three artifact types mentioned above is protected from unauthorized access. Additionally, for platform-controlled artifacts, there are retrieval functions in place for authorized users to query the stored data. A general deletion function is also available (for platform administrators only).

> ⚠ Caution
>
> From the perspective of applications, the platform acts as a container that is not aware of the type or structure of the potentially processed or stored personal data even if the services are provided by the platform. Therefore, applications are fully responsible for providing the features to fulfill personal data requirements in application-controlled artifacts.

Related Information

Processing of Personal Data in Platform-Controlled Artifacts [page 390]
Processing of Personal Data in Standard XS Advanced Applications and Services [page 394]
Data Protection and Privacy in SAP HANA [page 321]

17.7.1 Processing of Personal Data in Platform-Controlled Artifacts

Personal data originating both from platform users and business users may be present in server artifacts.

As a general rule, platform users are typically internal users who operate the application services and server infrastructure (for example, Controller users like SpaceDeveloper or administrators). Business users, on the other hand, interact with deployed applications.

Server artifacts that may contain personal data originating from the server infrastructure (category 1)

Various interaction sequences with the XS advanced server initiated by **platform users** may leave personal data in server artifacts. These are listed in the table below.

All server logs are written to a rotating file, which means that older log lines are removed from the log history when a specified file size threshold is met. The size and number of all rotated application log files may be adjusted to suit customer needs. Data is generally discarded sooner if low file-size limits and a small number of history files are configured.

Platform Artifact	Location	Type	User Type	Content
`xscontroller_<i>.log`	Trace directory of the SAP HANA instance (alias 'cdtrace') and `<xs-base>/controller_data/controller/tracing/log/`	Log file (rotated)	Platform user	User name
`xsexeagent_<i>.log`	Trace directory of the SAP HANA instance (alias 'cdtrace') and `<xs-base>/controller_data/controller/tracing/log/`	Log file (rotated)	Platform user	User name
`hdbxscontroller_audit_<i>.log`	Trace directory of the SAP HANA instance (alias 'cdtrace')	Log file (rotated)	Platform user	User name, client IP address
`hdbxsexecutionagent_audit_<i>.log`	Trace directory of the SAP HANA instance (alias 'cdtrace')	Log file (rotated)	Platform user	User name, client IP address
`uaa.log`	Trace directory of the SAP HANA instance (alias 'cdtrace')	Truncated and rotated log file	Platform and business users	User name, client IP address
`uaa-audit.log`	Trace directory of the SAP HANA instance (alias 'cdtrace')	Truncated and rotated log file	Platform and business users	User name, client IP address
`xsoidc_<i>.log`	Trace directory of the SAP HANA instance (alias 'cdtrace')	Truncated and rotated log file	Platform and business users	User name, client IP address
`access_log-controller-route-api.log`	`<xs-base>/controller_data/controller/router/webdispatcher/logs/`	Log file (rotated)	Platform user	Client IP address
`access_log-external-uaa-route-uaa-server.log`	`<xs-base>/controller_data/controller/router/webdispatcher/logs/` (in `xs base` directory alias 'cdxs')	Log file (rotated)	Platform and business users	client IP address
STOREDEVENT table	Schema `SYS_XS_RUNTIME`	(Truncated) persistence	Platform users	User name

Platform Artifact	Location	Type	User Type	Content
UAA shadow user	Schema SYS_XS_UAA (only if an external identity providers is configured)	persistence	Platform and business users	User name, email and so on
`syslog`	`/var/log/messages`	According to UNIX settings	Platform and business users	User name, client IP address
XS advanced run time	SAP HANA audit log	Database table	Platform users	User name, client IP address

> **Note**
>
> `<xs-base>` denotes the XS advanced base directory (alias 'cdxs').

The following platform-provided functions can be used on the artifacts listed above:

- To retrieve content of the log files, either access the file from the UNIX command line as `<sid>`adm, or view them with the SAP HANA database explorer (*Database Diagnostics Files*)
- To retrieve the content of stored events (table STOREDEVENT), use the `xs events` command.
- To delete all entries in the listed log and audit files older than a specified date, use the command XSA `delete-personal-data <date> --exclude APP, USERS` executed as `<sid>`adm.

  > **Caution**
  >
  > Deleted data cannot be recovered.

  The `exclude` option allows you to prevent the deletion of specific artifacts:
  - PLATFORM
    Platform logs including the event log in the database and the trace files
  - AUDIT
    Audit log files of XS advanced
  - APP
    Application log files
  - ACCESS
    WebDispatcher access logs
  - USERS
    Shadow users in the UAA

> **Note**
>
> The size and number of all rotated log files may be adjusted to suit customer needs. Note that the lower the file-size limit and the shorter the length of time for which files are retained, the sooner logged data will be discarded.

> **Note**
>
> If audit events are written to the UNIX syslog, the system operator must configure the retention policy, and so on.

For more information about the `xs` and `XSA` commands mentioned here, see the section on maintaining the XS advanced runtime environment in the *SAP HANA Administration Guide*.

Server artifacts that may contain personal data originating from applications (category 2)

Various interaction sequences with applications deployed to the XS advanced server may also leave personal data related to **business users** in artifacts that are managed by the server.

> ⚠ Caution
>
> There may be additional relevant data managed by the application itself, for example, data that is retained in an application-specific persistence. For this kind of data, it is the responsibility of the application to provide ways to fulfill data-protection and privacy requirements.

Artifact	Location	Type	User Type	Content
Application logs (stdout, stderr)	`<app_working>/host/executionroot/<app-instance>/app-logs`	Log file (rotated)	All types of users	Application specific
HTTP access logs of application instances	`<xs base>/controller_data/controller/router/webdispatcher/logs`	Log file (rotated)	All types of users	Client IP address
Auditlog service	SAP HANA audit log	Database table	All types of users	User name, client IP address

The following platform-provided functions can be used on the artifacts listed above:

- To retrieve the full content of application logs, execute the command `xs logs <app> --all` to retrieve all log lines of the application you are interested in.
- To delete all application log entries older than a specified date, execute the command `XSA delete-personal-data <date> --exclude PLATFORM, AUDIT, USERS` as `<sid>`adm.

  > ⚠ Caution
  >
  > Deleted data cannot be recovered.

- To view the entries written to the auditlog-service, use the available SAP HANA tools to view and manage the data.

Related Information

The XSA Command Reference
Data Protection and Privacy Tools in XS Advanced

Audit Trails [page 284]

17.7.2 Processing of Personal Data in Standard XS Advanced Applications and Services

Personal data originating from business users and written by system applications may be present in server artifacts (category 1 – 3).

Application Name	Personal Data	Location	Comment
AppRouter	User ID, name, and client IP address in the event of unsuccessful login	Audit log service and/or application log (category 2)	Necessary for application operation
deploy-service	User ID, name, client IP of platform user	Application log (category 2), internal persistency scoped to deployment process (Category 3)	Necessary for application deployment
product-installer	User ID, name, client IP address of platform user	Application log (category 2), internal log persistency scoped to installed product (category 3)	Necessary for application installation
job-scheduler	User ID, name, client IP address of platform user	Application log (category 2), internal log persistency scoped to created service (category 3)	Necessary for service deployment
Admin tools	User ID, name, client IP address of platform user	Application log (category 2)	Necessary for platform operation
XSA cockpit	User ID, name, client IP address of platform user	Application log (category 2)	Necessary for platform operation

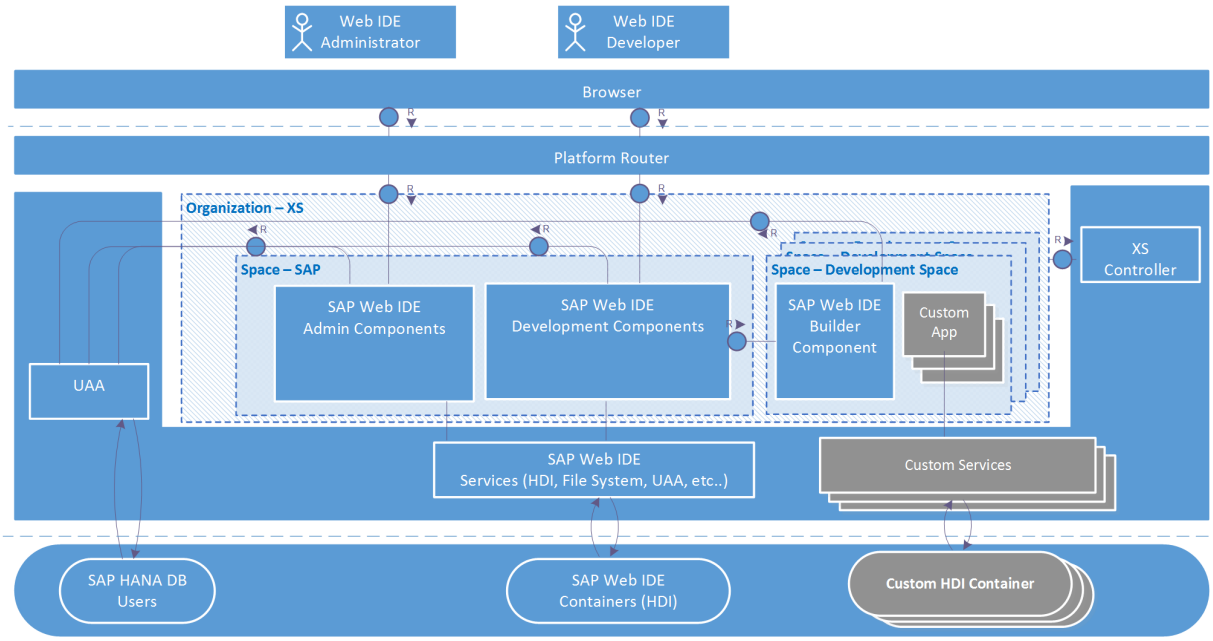
17.8 Security Aspects of SAP Web IDE for SAP HANA

SAP Web IDE for SAP HANA (SAP Web IDE) is a browser-based integrated development environment for the development of SAP HANA-based applications. These applications are comprised of web-based or mobile UIs, business logic, and extensive SAP HANA data models.

SAP Web IDE supports developers who use SAP HANA Extended Application Services, advanced model (XS Advanced), by providing a variety of tools. These tools include syntax-aware editors for code and SAP HANA artifacts, graphical editors for Core Data Services (CDS) data models and calculation views, as well as inspection, testing, and debugging tools.

Architecture

SAP Web IDE is comprised of several application components deployed in XS Advanced application server. The following diagram provides an architectural overview of SAP Web IDE in XS Advanced.



> ### Note
>
> Since SAP Web IDE is not tenant-aware, customers who use it in an environment with multiple tenant databases should install an instance of SAP Web IDE in each XS Advanced organization that represents a tenant.

Multiple Spaces for Development

SAP Web IDE supports multiple spaces in XS Advanced, in which developers build and run their applications. To ensure the isolation of the development environment, different development teams should use separate spaces.

> ### Note
>
> We strongly recommend not to use the predefined *SAP* space for development. Doing so compromises the SAP Web IDE security.

SAP Web IDE provides an administration tool to enable XS Advanced spaces for development. For more information about this tool, see *Managing Spaces for Development* in *SAP Web IDE for SAP HANA - Installation and Upgrade Guide*.

> ### Note
>
> Developers who use the same space can access each other's application artifacts. In fact, any authenticated SAP HANA database user assigned to the *SpaceDeveloper* role for a specific space has

full access to all applications in this space, and can potentially cause disruption or misuse of these applications.

All SAP Web IDE components, except the Builder component, are installed in the predefined *SAP* space, whereas the Builder component is installed in each space used for development.

Communication with Remote Systems

SAP Web IDE supports access to remote systems, such as a Git source control system. For remote Git repositories, which issue SSL certificates that are not trusted publicly, SAP Web IDE provides a command-line tool that enables administrators to manage SSL certificates. For more information about this tool, see *Managing SSL Certificates* in *SAP Web IDE for SAP HANA - Installation and Upgrade Guide*.

Related Information

Security for SAP HANA Extended Application Services, Advanced Model [page 336]
User Authorization and Authentication [page 396]
Known Security-Related Issues [page 398]

17.8.1 User Authorization and Authentication

User administration, authorization, and authentication concepts of SAP Web IDE for SAP HANA.

User Administration and Authentication

SAP Web IDE is installed on top of XS Advanced, and uses its User Account and Authorization service (UAA) and the application router to manage user logon and logout requests. The UAA service centrally manages the issuing of tokens for propagating the user identity to application containers and to the SAP HANA database.

Authorization

Authorization grants access to application resources and services based on the defined user permissions. Authorization checks are performed on the levels of the XS Advanced controller and each of the SAP Web IDE components.

SAP Web IDE supports the following user types:

- A developer user in SAP Web IDE for SAP HANA is an SAP HANA database user who has additional permissions to create, modify, build, and run applications in SAP Web IDE. These developers are assigned to personal SAP Web IDE workspaces, where they manage their own application artifacts, such as projects, modules, and files. Workspace access is granted only to its owner.
  The following table lists the roles and role collections that are required for a developer user using Web IDE:

User Type	Role/Role Collections	Description
Developer	*XS_CONTROLLER_USER* role collection	Grants read-write permissions within the assigned organization or space.
Developer	*SpaceDeveloper* role	Assigned per space in XS Advanced. Enables users to access the shared resources of the space, and to deploy, build, and run applications.
Developer	A role collection containing the *WebIDE_Developer*role	Enables users to develop applications using SAP Web IDE.

> ### Note
> To facilitate the work of developers, the SAP Web IDE npm registry cache component provides Node.js modules, which are otherwise available to developers via the SAP public npm registry. Any user has read-only access to this registry without any authentication or authorization.

- **SAP Web IDE administrator** is an SAP HANA database user who has additional permissions to perform administration tasks for SAP Web IDE.
  The following table lists the required roles and role collections for an administrator of Web IDE.

User Type	Role/Role Collections	Description
Administrator	*XS_CONTROLLER_USER* role collection	Grants read-write permissions within the assigned organization or space.
Administrator	*SpaceDeveloper* role	Assigned per space in XS Advanced. Enables users to access the shared resources of the space, and to deploy, build, and run applications.
Administrator	A role collection containing the *WebIDE_Administrator* role template	Enables users to access the SAP Web IDE administration tools, such as SSL management and space enablement.

Authentication and Authorization for Custom Applications

Custom applications developed using SAP Web IDE are standard XS Advanced applications, which are deployed into spaces in XS Advanced. SAP Web IDE does not perform any UAA functions on behalf of the applications. Therefore, application developers should implement their own authentication and authorization support using the platform security functions, which are provided by XS Advanced.

Connections to databases in the SAP HANA database explorer

When you specify the credentials to connect to an SAP HANA database, those credentials are not saved to your browser. To have the credentials persist between sessions, save the credentials to the SAP HANA secure store

Related Information

User Administration and Authentication in SAP HANA XS Advanced [page 344]
Security for SAP HANA Extended Application Services, Advanced Model [page 336]

17.8.2 Known Security-Related Issues

A list of known security-related issues of the SAP Web IDE for SAP HANA.

Topic	Issue / Impact	Workaround
Backup and recovery	SAP Web IDE does not support backup and recovery for workspace content (projects, folders, or files).	Use a Git repository as a backup.
File upload	No automatic virus scan or content validation of files is performed before uploading them to SAP Web IDE. Malicious content can be uploaded.	Use external tools to validate the content of the files and scan them for viruses before uploading them to SAP Web IDE. Alternatively, use a file-based antivirus tool to scan the files on the disk.
local-npm-cache allows unauthenticated enumeration of npm packages	The local-npm-cache used by SAP Web IDE for SAP HANA allows unauthenticated users to enumerate the npm modules contained in it. This might provide attackers with detailed information about the used npm modules and potential vulnerabilities.	1. Find the port used by the application by running the following command: `-Xs a` This will display a list of the apps running on the installation. 2. Search for the "`di-local-npm-registry`" application. The URL of the application including the port will be displayed next to it. 3. Send the port or the URL to the team in charge of blocking access. Blocking access to the application or port is dependent on your network's architecture. For a simple solution, the port can be blocked by using an entry in the iptables. Other solutions may exist if a firewall is installed on the machine.

Related Information

Securing the SAP HANA Database Explorer

18 Security for SAP HANA Deployment Infrastructure (HDI)

An overview of the components and features used to configure and ensure security in the SAP HANA Deployment Infrastructure (HDI).

The information in this section of the *SAP HANA Security Guide for SAP HANA Platform* describes the aspects of security that need to be considered when setting up and using the SAP HDI infrastructure. The information covers the following security-related areas:

HDI Security Component	Summary
Technical System Landscape of SAP HDI [page 400]	The architecture of SAP HDI and the protocols used for communication between the various high-level components
SAP HDI Users [page 401]	The predefined users which SAP HDI relies on and a description of each user's scope
SAP HDI Database Roles [page 402]	The predefined database roles that are required to operate the SAP HANA Deployment Infrastructure (HDI) and an explanation of each role
Data Storage Security in SAP HDI [page 403]	The mechanisms used to secure and protect critical data managed by the SAP HDI
Network and Communication Security with SAP HDI [page 404]	The security mechanisms applied to networking and communication in the context of SAP HDI
Logging and Tracing in SAP HDI [page 404]	The auditing process that enables you to trace the different kinds of operations performed by HDI users in the context of SAP HDI
Data Protection and Privacy in SAP HDI [page 405]	The scope of SAP HDI's adherence to (and compliance with) data protection and privacy regulations
SAP HDI Security in the Context of XS Advanced [page 405]	The security considerations to bear in mind when enabling SAP HDI for use in the context of SAP HANA extended application services, advanced model

Technical System Landscape of SAP HDI

HDI clients such as the client libraries for Java and JavaScript use the HDI SQL API to manage containers and their content. So, too, does the HDI deployer tool as well as the SAP HANA extended application services, advanced model (XS advanced), which also uses HDI containers as a way of implementing database service instances.

The HDI API is based on SQL and database procedures. For example, the application programming interface (API) for system-wide, HDI-container management includes procedures for creating and dropping HDI

containers. Each container has its own container API consisting of container-specific procedures for uploading (`WRITE`), listing (`READ`), and retrieving (`LIST`) design-time objects, for deploying uploaded design time objects (`MAKE`), and for granting/revoking access to/from the container's schemas and API (grant, revoke).

It is also possible to assign HDI containers to container groups where a corresponding group-level management API restricts the management operations (for example, create/drop, configure, etc.) to the set of containers included in the specific group.

Users and Clients

Users and applications interact with SAP HDI by means of one of the available clients, which use the SAP HDI's SQL application-programming interface to manage HDI containers and the corresponding content.

- HDI client
  HDI provides client libraries for Node.js (`@sap/hdi`) and Java (`sap-java-hdi`) which include a selection of methods that can be used to connect to the HDI and access HDI functionality.
- The development environment
  The HDI's API is based on a library of SQL procedures, which can be used (among other things) to create and drop containers; upload design-time objects to the virtual file system of the HDI container; deploy uploaded objects (make) to the container run time; and grant (or revoke) the permissions required to access HDI containers.
- Deployment to the SAP HANA XS advanced platform
  The Node.js packages `@sap/hdi-deploy` and `@sap/hdi-dynamic-deploy` are based on the HDI's SQL API and enable the deployment of database content to HDI containers.

> → Tip
>
> For more information about HDI security, see *SAP HANA Deployment Infrastructure (HDI) Reference for SAP HANA Platform* in *Related Information* below.

SAP HDI Users

HDI uses several container-specific technical users to separate the different container-related tasks such as creating the schemas, triggering deployment, and creating database objects for XS advanced applications. The technical users listed in the following table are used to connect to the SAP HANA database with a specific set of conditions.

User ID	Service	Type	Description
`HDI_ADMIN_USER`	SAP HANA Broker	Technical database user	Owns the SAP HANA schema of the SAP HANA Service Broker; only used in the context of XS advanced

User ID	Service	Type	Description
`HDI_BROKER_CONTROLLER`	SAP HANA Broker	Technical database user	Has the authorization required to access the service-broker API of the SAP HANA Service Broker; only used in the context of XS advanced
`_SYS_DI`	HDI	Technical database user	Owns all HDI SQL-based APIs, for example all API procedures in the `_SYS_DI` schema and API procedures in containers
`_SYS_DI_*_CATALOG`	HDI	Technical database user	Technical users used by the HDI to access database system catalog tables and views
`_SYS_DI_SU`	HDI	Technical database user	Technical superuser of the HDI created at installation time
`_SYS_DI_TO`	HDI	Technical database user	Owns transaction and connections of all internal HDI transactions

Technical Users for HDI Schema-Based Containers

The deployment of database objects with SAP HANA Deployment Infrastructure (HDI) is based on a container model where each container corresponds roughly to a database schema. Each schema, and the database objects deployed into the schema, are owned by a dedicated technical database user.

For every container deployed, a new technical database user and schema with the same name as the container are created. Additional schemas and technical users required for metadata and deployment APIs are also created.

These technical database users are used internally by HDI only. As a general rule, these users are created as restricted database users who do not have any privileges by default (not even the role `PUBLIC`), and cannot be used to log on to the database. The only exception to this rule concerns user `S#00` who, from SAP HANA 2.0 SPS 02 revision 20, is granted the role `PUBLIC` by default.

> → Tip
>
> For more information about HDI security, see *SAP HANA Deployment Infrastructure (HDI) Reference for SAP HANA Platform* in *Related Information* below.

SAP HDI Database Roles

Several predefined database roles are necessary for operating the SAP HANA Deployment Infrastructure (HDI). The following roles are SQL-based roles that are available in the catalog of the SAP HANA database.

Role	Description
`_SYS_DI_OO_DEFAULTS`	This role contains the set of default privileges that are granted to all HDI container object-owner users (`<container>`#OO users). SAP HANA Deployment Infrastructure (HDI) uses this role internally to grant default privileges instead of using the `PUBLIC` role. It contains only privileges to `SYS` views where additional security checks apply. The role contains `SELECT` privileges on the views: `SYS.DUMMY`, `SYS.PROCEDURES`, `SYS.PROCEDURE_PARAMETERS`, `SYS.TABLES`, `SYS.TABLE_COLUMNS`. This role is not intended to be granted to database users. **Note** Do **not** extend this role in a production system.
`_SYS_DI#<container_group>._SYS_DI_OO_DEFAULTS`	This role contains the set of default privileges that are granted to all HDI container object-owner users (`<container>`#OO users) who are part of the container group `<container_group>`. The role does not contain any privileges by default. Privileges granted to a container group-specific role will automatically be available to all HDI container object-owner users of the specified container group. **Note** This role is not intended to be granted to database users.

Data Storage Security in SAP HDI

Since SAP HDI is a service layer on top of the SAP HANA database, it adheres to all the rules and regulations that apply to the SAP HANA database itself and described in detail in the *Data Storage Security in SAP HANA* section of the *SAP HANA Security Guide*.

Database objects (tables, views, procedures, and so on) have an owner: the user who created the object. When the owner of a database object is deleted, all objects owned by the deleted user are removed from the database, too. In addition, if application objects are created by end-users, the objects are deleted when the end-user is deleted, for example when the employee leaves the organization. HDI ensures that during deployment all database objects are created by a container-specific **technical user**, which is never deleted as long as the container exists.

→ Tip

For more information about HDI security, see *SAP HANA Deployment Infrastructure (HDI) Reference for SAP HANA Platform* in *Related Information* below.

Network and Communication Security with SAP HDI

Security mechanisms are applied to protect the communication paths used by the SAP HANA Deployment Infrastructure (HDI). SAP provides network topology recommendations that enable administrators to restrict access at the network level. The JDBC connection to the SAP HANA database is not encrypted by default. To activate JDBC TLS/SSL, it is necessary to configure custom SSL certificates as described in *SAP HDI Certificate Management* below.

Since the SAP HDI does not encrypt the JDBC connections to SAP HANA by default, the HDI administrator is responsible for the configuration and management of custom certificates in the SAP HANA JDBC area.

> → Tip
>
> For more information about HDI security, see *SAP HANA Deployment Infrastructure (HDI) Reference for SAP HANA Platform* in *Related Information* below.

Security-relevant Logging and Tracing in SAP HDI

The auditing process enables you to trace who has performed which kinds of operations in the context of SAP HDI.

System audit logs can help detect undesired modifications that have security implications, for example, changes to user authorization, creation and deletion of database objects, or changes to the system configuration. Audit logs can also help uncover attempts to breach system security or reveal security holes, for example, where a specific user has privileges that are not necessary. Auditing can also help protect the system owner against the threat of security violations and data misuse or may serve to meet security standards of a company. For SAP HDI auditing, audit-log events can be written to SAP HANA's central auditing service.

Auditing in SAP HDI

The standard capabilities provided by the SAP HANA audit policies are sufficient for auditing actions in the context of SAP HDI, too. The functionality of SAP HDI is provided by means of an SQL interface and implemented either as built-in procedures or stored procedures. Using the existing auditing features, it is possible to audit all calls to SAP HDI.

Auditing SAP HDI Procedures

In SAP HDI, general procedures enable the execution of high-level operations, for example, creating and dropping containers, granting and revoking privileges and roles, or configuring SAP HDI itself. For more information about how to enable the auditing of procedures, see *Security-relevant Logging and Tracing in SAP HDI* in *Related Information* below.

Auditing Container-Specific SAP HDI Procedures

Developers using SAP HDI typically work within a single container, calling container-specific SAP HDI procedures, for example: writing, reading, and deleting files and folders, or deploying and undeploying these artifacts. For this scenario, it is recommended to create and enable an audit policy that audits container-specific SAP HDI procedures for a specific container. For a container named “C”, the procedure to be audited might be `C#DI.CANCEL` or `C#DI.CONFIGURE_CONTAINER`.

Auditing All SAP HDI Database Operations

The standard capabilities provided by the SAP HANA auditing mechanism are sufficient for most scenarios. In rare cases, however, it can be helpful to audit **all** SAP HDI database operations, for example, creating and dropping tables and users or inserting data into tables. As all these database operations are executed internally by a dedicated SAP HDI transaction owner named "`_SYS_DI_TO`", it is sufficient to limit auditing to this user. For example, you can create and enable an audit policy that audits all SAP HDI database operations for the SAP HDI transaction owner `_SYS_DI_TO`.

> → Tip
>
> For more information about HDI security, see *SAP HANA Deployment Infrastructure (HDI) Reference for SAP HANA Platform* in *Related Information* below.

Data Protection and Privacy in SAP HDI

The SAP HDI server (`diserver`) provides clients with a selection of tools and functions that conform to the legal requirements for the protection and privacy of data included in deployed artifacts, as well as the server infrastructure itself. The information in this section or any other sections in this Security Guide is not intended as advice on whether these features and functions are the best method to support company-, industry-, regional-, or country-specific requirements. Furthermore, this guide does not give any advice or recommendations regarding additional features that might be required in specific IT environments; decisions related to data protection must be made on a case-by-case basis, taking into consideration the given system landscape and the applicable legal requirements.

> → Tip
>
> For more information about HDI security, see *SAP HANA Deployment Infrastructure (HDI) Reference for SAP HANA Platform* in *Related Information* below.

SAP HDI Security in the Context of XS Advanced

In the context of SAP HANA extended application services, advanced model (XS advanced), SAP HDI provides a service that enables you to deploy database development artifacts such as tables, views, and procedures to HDI containers. The deployment process populates the database run-time with the corresponding catalog objects. In addition to database artifacts, HDI also enables you to import and export table content such as business configuration data and translatable texts. The security-related information for XS advanced covers the following components:

- The SAP HDI Deployer
- The SAP HDI SQL API
- SAP HDI containers
- SAP HDI users in XS advanced
- SAP HDI roles in XS advanced
- SAP HDI Data Storage in XS advanced

→ Tip

For more information about HDI security in the context of XS advanced, see *SAP HANA Deployment Infrastructure (HDI) Reference for SAP HANA Platform* in *Related Information* below.

Related Information

The SAP HANA Deployment Infrastructure Reference
Predefined Database Users [page 89]
SAP HANA DI Roles [page 191]
Security-relevant Logging and Tracing in SAP HDI

19 SAP HANA Security Reference Information

Security reference information for SAP HANA

> **Note**
>
> Please also refer to the document *SAP HANA Security Checklists and Recommendations*.

Related Information

SAP HANA Security Checklists and Recommendations

19.1 SQLScript Security Procedures

SQLScript procedures for security functions

For more information about calling SQLScript procedures from clients, see *SAP HANA SQLScript Reference*.

Related Information

SAP HANA SQLScript Reference

19.1.1 GENERATE_STRUCTURED_PRIVILEGE_PFCG_CONDITION (SYS)

SQLScript procedure to generate a filter condition based on ABAP authorization objects that can be used in the CONDITION PROVIDER clause of an analytic privilege

Procedure Call

```
CALL SYS.GENERATE_STRUCTURED_PRIVILEGE_PFCG_CONDITION(<parameter_list>)
```

Input Parameters

- Schema in which the SAP authorization tables UST12 and USRBF2 reside
- An expression that contains at least one CHECKID

  > **Note**
  >
  > Multiple CHECKIDs may be concatenated. Expression can contain AND, OR and NOT Boolean operators.

- A JSON-formatted string that contains the information required to translate each ABAP authorization object referenced. For each CHECKID, the following must be specified:
  - `authobj`
    This specifies the name of the ABAP authorization object to which the CHECKID is mapped. This may contain an empty value if the CHECKID itself is the name of the ABAP authorization object.
  - `filter`
    This allows you to specify fixed conditions that the user's ABAP authorizations need to match. It is a list of required fields (`key`) and their values (`values`).
  - `mappings`
    This specifies the mapping of FIELD value (`fieldName`) in the ABAP authorization object to column name (`mappedName`) of the target view to be protected.

Output Parameters

An NCLOB that contains the SQL filter condition

> **Note**
>
> The procedure returns 1>1 if the user has no permissions and 1=1 if the user has full access.

Examples

> **Example**
>
> Input:

```
CALL SYS.GENERATE_STRUCTURED_PRIVILEGE_PFCG_CONDITION(
'A_TEST_SCHEMA',
'CHECKID1',
'{"data":
    {
        "CHECKID1":
        {
            "authobj":"OBJ1",
            "filter":[{"key":"ACTVT","valueList":["03"]}],
            "mappings":[{"fieldName":"SACMTSOID", "mappedName":"SO_ID"},
{"fieldName":"SACMTSOLCS", "mappedName":"LIFECYCLE_STATUS"}]
        }
```

```
    }
}',
?)
```

Related Information

Shared Business Authorizations in SAP HANA [page 210]

19.1.2 GET_INSUFFICIENT_PRIVILEGE_ERROR_DETAILS

SQLScript procedure to determine which privilege a user is missing when he or she gets an "insufficient privilege" error.

> **Note**
>
> Not all "insufficient privilege" errors can be resolved by granting the user a missing privilege, for example, some operations can only be performed by the SYS user, or by a user in the system database. In these cases, the error message provides the explanation.

Procedure Call

```
CALL SYS.GET_INSUFFICIENT_PRIVILEGE_ERROR_DETAILS ('<GUID in "insufficient
privilege" error>', ?)
```

Input Parameters

GUID output in the error message received by the user experiencing the authorization issue

Output Parameters

- GUID
- CREATE_TIME
- CONNECTION_ID
- SESSION_USER_NAME
- CHECKED_USER_NAME
- PRIVILEGE

- IS_MISSING_ANALYTIC_PRIVILEGE
- IS_MISSING_GRANT_OPTION
- DATABASE_NAME
- SCHEMA_NAME
- OBJECT_NAME
- OBJECT_TYPE
- PASSPORT_ACTION
- PASSPORT_CONNECTION_COUNTER
- PASSPORT_CONNECTION_ID
- PASSPORT_COMPONENT_NAME
- PASSPORT_TRANSACTION_ID
- PASSPORT_ROOTCONTEXT_ID

> **Note**
>
> CHECKED_USER_NAME is the user whose authorization is checked. SESSION_USER_NAME is the user who is connected to the database.

> **Note**
>
> If the user is missing several privileges, only the first missing privilege determined by the authorization check is returned. You can find out what other privileges are missing by successively running `GET_INSUFFICIENT_PRIVILEGE_ERROR_DETAILS` with the GUID returned with each subsequent error message.

Example

> **Example**
>
> An unauthorized user (JOHN) tries to create a new user. The following error message is returned:
>
> ```
> insufficient privilege: Detailed info for this error can be found with guid
> '4043AC8661D7D64B82449FCD18D23D1B'
> ```
>
> Input:
>
> ```
> CALL SYS.GET_INSUFFICIENT_PRIVILEGE_ERROR_DETAILS
> ('4043AC8661D7D64B82449FCD18D23D1B', ?)
> ```
>
> Output:
>
> ```
> GUID;4043AC8661D7D64B82449FCD18D23D1B
> CREATE_TIME;30-Jul-2018 14:16:01.05
> CONNECTION_ID;103,788
> SESSION_USER_NAME;JOHN
> CHECKED_USER_NAME;JOHN
> PRIVILEGE;USER ADMIN
> IS_MISSING_ANALYTIC_PRIVILEGE;FALSE
> IS_MISSING_GRANT_OPTION;FALSE
> DATABASE_NAME;
> SCHEMA_NAME;
> ```

```
OBJECT_NAME;
OBJECT_TYPE;
```

The user JOHN is missing the system privilege USER ADMIN.

Related Information

Resolve Insufficient Privilege Errors

19.1.3 IS_VALID_PASSWORD (SYS)

SQLScript procedure to validate whether a given password will be accepted by the SAP HANA database as part of a CREATE USER or ALTER USER statement.

Procedure Call

```
CALL SYS.IS_VALID_PASSWORD (<parameter_list>)
```

Input Parameters

- Password

> **Note**
>
> If the password contains double quotes ("), use the security function ESCAPE_DOUBLE_QUOTES to return the password with escaped double quotes, and then enter the result as the input parameter. Alternatively, you can use the function direclty in the procedure call: `CALL SYS.IS_VALID_PASSWORD( ESCAPE_DOUBLE_QUOTES(?), ?, ? )`.
>
> For more information about the ESCAPE_DOUBLE_QUOTES function, see the *SAP HANA SQL and System Views Reference*.

Output Parameters

- ERROR_CODE
- ERROR_MESSAGE

Examples

Example

Input:

```
CALL SYS.IS_VALID_PASSWORD('Abcd1234', ? ,?)
```

Output:

```
Out(1);Out(2)
0     ;
```

The password is valid.

Example

```
'CALL SYS.IS_VALID_PASSWORD(ESCAPE_DOUBLE_QUOTES(?), ?, ? )'
```

Example

Input:

```
CALL SYS.IS_VALID_PASSWORD('asdf', ?, ?)
```

Output

```
Out(1);Out(2)
412   ;invalid password layout: minimal password length is [8]
```

The password is not valid.

Related Information

ESCAPE_DOUBLE_QUOTES Function (Security)

19.1.4 IS_VALID_USER_NAME (SYS)

SQLScript procedure to validate whether a given user name will be accepted by the SAP HANA database as part of a CREATE USER or ALTER USER statement.

Procedure Call

```
CALL SYS.IS_VALID_USER_NAME (<parameter_list>)
```

Input Parameters

- User name

Output Parameters

- ERROR_CODE
- ERROR_MESSAGE

Examples

Example

Input:

```
CALL SYS.IS_VALID_USER_NAME('bob', ? ,?)
```

Output:

```
Out(1);Out(2)
0     ;
```

The user name is valid.

Example

Input:

```
CALL SYS.IS_VALID_USER_NAME('billy bob', ?, ?)
```

Output:

```
Out(1);Out(2)

257   ;sql syntax error: incorrect syntax near \"bob\": line 1 col 19 (at pos
19)
```

The user name is not valid.

19.2 Security Reference for Tenant Databases

Reference information for secure configuration of tenant databases

19.2.1 Restricted Features in Tenant Databases

To safeguard and/or customize your system, certain features of the SAP HANA database can be disabled in tenant databases.

Some features of the SAP HANA database are not required or desirable in certain environments, in particular features that provide direct access to the file system, the network, or other resources. The table below lists those features that you can explicitly disable in tenant databases.

The system view M_CUSTOMIZABLE_FUNCTIONALITIES lists all features that can be disabled and their status. This view exists in both the SYS schema of every database, where it contains database-specific information, and in the SYS_DATABASES schema of the system database, where it contains information about the enablement of features in all databases. For more information, see M_CUSTOMIZABLE_FUNCTIONALITIES in the *SAP HANA SQL and Systems View Reference*.

For more information about how to disable features, see in the *SAP HANA Administration Guide*.

> **Note**
>
> Features are hierarchically structured. If you disable a feature with sub-features, these are also disabled.

Feature	Feature Description	Why Disable?
RINTEGRATION	R language	Feature not required in all deployment scenarios
XB_MESSAGING_SUBSCRIPTIONS	Subscriptions For External Messaging Providers	
AFL	Access to Application Function Libraries (AFL) for business logic in native C++	Feature not required in all deployment scenarios
XB_EXTERNAL_CONNECTIVITY	External Connectivity With XB Message Bus	
BACKUP	Backup operations	File system access not wanted
BACKUP.IGNOREPATH_RESTRICT	Ignore path restrictions for backup operations	File system access not wanted, safeguard against directory traversal attacks
IMPORTEXPORT	Import and export operations	File system access not wanted
IMPORTEXPORT.IMPORT	Import operations	File system read access not wanted
IMPORTEXPORT.EXPORT	Export operations	File system write access not wanted

Feature	Feature Description	Why Disable?
IMPORTEXPORT.IGNORE_PATH_RESTRICT	Ignoring of path restrictions for import and export	File system access not wanted, safeguard against directory traversal attacks
BUILTINPROCEDURE	Execution of procedures associated with critical and/or optional functions	--
BUILTINPROCEDURE.REORG_GENERATE	Data redistribution operations	Feature not required in all deployment scenarios
BUILTINPROCEDURE.REORG_EXECUTE		
BUILTINPROCEDURE.REORG_CLEAR_LOGS		
BUILTINPROCEDURE.GEM	Procedure to use the graph engine	Feature not required in all deployment scenarios
BUILTINPROCEDURE.MANAGEMENT_CONSOLE_PROC	Access to the built-in SAP HANA management console (`hdbcons`)	Safeguard against leakage of SAP HANA process information
BUILTINPROCEDURE.KERNELCALL	Access to rowstore internal maintenance features	Safeguard against leakage of SAP HANA process information
BUILTINPROCEDURE.TREXVIADBSL	Operation of an SAP Business ByDesign system	Feature not required in all deployment scenarios
BUILTINPROCEDURE.COMPRESS_FILE	Compression of trace files before they are transferred	Feature not required in all deployment scenarios
BUILTINPROCEDURE.GET_FULL_SYSTEM_INFO_DUMP	Triggering of complete information dump of the entire system	Feature not required in all deployment scenarios
BUILTINPROCEDURE.FULL_SYSTEM_INFO_DUMP_CREATE	Triggering creation of complete information dump of the entire system	Feature not required in all deployment scenarios
BUILTINPROCEDURE.FULL_SYSTEM_INFO_DUMP_DELETE	Triggering deletion of complete information dump of the entire system	Feature not required in all deployment scenarios
BUILTINPROCEDURE.FULL_SYSTEM_INFO_DUMP_RETRIEVE	FULL_SYSTEM_INFO_DUMP_RETRIEVE execution	Feature not required in all deployment scenarios
BUILTINPROCEDURE.DSO	Creation of and access to DataStore Objects (DSOs) for SAP Business Warehouse (BW) powered by SAP HANA	Feature not required in all deployment scenarios
BUILTINPROCEDURE.STATISTICSSERVER_CONFIGCHECKPROC	Validation of the statistics server configuration and its e-mail notification capability	Feature not required in all deployment scenarios
BUILTINPROCEDURE.BW_PRECHECK_RELEASE_LOCK	Operation of an SAP BW powered by SAP HANA system	Feature not required in all deployment scenarios
BUILTINPROCEDURE.BW_PRECHECK_ACQUIRE_LOCK_WITH_TYPE		
BUILTINPROCEDURE.BW_PRECHECK_ACQUIRE_LOCK		
BUILTINPROCEDURE.BW_CONVERT_CLASSIC_TO_IMO_CUBE		

Feature	Feature Description	Why Disable?
BUILTINPROCEDURE.BW_F_FACT_TABLE_COMPRESSION		
BUILTINPROCEDURE.UPDATE_LANDSCAPE_CONFIGURATION	Changes to system landscape and the available services in a system	Feature not required in all deployment scenarios
ALTERSYSTEM	Execution of the statement ALTER SYSTEM RECONFIGURE SERVICE, which re-reads the service configuration	Feature not required in all deployment scenarios
ALTERSYSTEM.RECONFIGURE_SERVICE		
SMARTDATAACCESS	Federated access to other database systems through virtual tables	Feature not required in all deployment scenarios
DXC	Data acquisition and consumption of data models from the SAP Business Suite	Feature not required in all deployment scenarios
DYNAMIC_TIERING	SAP HANA Dynamic Tiering operations	Feature not required in all deployment scenarios
DYNAMIC_TIERING.CREATE_EXTENDED_STORAGE	Creation of extended storage	
DYNAMIC_TIERING.DROP_EXTENDED_STORAGE	Deletion of extended storage	
DYNAMIC_TIERING.ALTER_EXTENDED_STORAGE	Changes to extended storage	
DYNAMIC_TIERING.ALTER_TABLE_TYPE	Conversion of a regular database table to an extended table or the reverse	
DYNAMIC_TIERING.BULK_INSERT_OPTIMIZATION	Bulk insert optimization that executes large inserts into extended tables using a load statement	
DYNAMIC_TIERING.QUERY_PLAN_RELOCATION	Query relocation operation that moves data from SAP HANA and SAP HANA Dynamic Tiering for optimal query performance	
ACCELERATOR_FOR_ASE	SAP HANA Accelerator for SAP ASE operations	
BOE	SAP BusinessObjects Explorer API	Feature not required in all deployment scenarios
SMART_DATA_STREAMING	SAP HANA Streaming Analytics operations	Feature not required in all deployment scenarios
GRAPH_ENGINE	SAP HANA Graph Engine	Feature not required in all deployment scenarios

Feature	Feature Description	Why Disable?
LDAP	All operations related to LDAP group authorization including creating/modifying LDAP providers, changing user authorization mode to `LDAP`, mapping LDAP groups to SAP HANA roles, and so on. **⚠ Caution** If the LDAP feature is disabled, existing users with authorization mode `LDAP` cannot log on. The authorization mode of these users can be changed to `LOCAL`.	Feature not required in all deployment scenarios
EPMPLANNING	SAP HANA Enterprise Performance Management planning feature	Feature not required in all deployment scenarios
PLANNINGENGINE	Planning engine features	Feature not required in all deployment scenarios

Related Information

Disable Features on a Tenant Database
M_CUSTOMIZABLE_FUNCTIONALITIES System View

19.2.2 Default Blacklisted System Properties in Tenant Databases

In systems that support tenant databases, there's configuration change blacklist `multidb.ini`, which is delivered with a default configuration.

The following table lists the system properties that are included in the `multidb.ini` file by default. So, tenant database administrators can't change these properties. System administrators can still change these properties in the system database in all layers.

You can customize the default configuration change blacklist by changing existing entries in the `multidb.ini` file and adding new ones. For more information about how to prevent changes to specific system properties in tenant databases, see *Prevent Changes to System Properties in Tenant Databases* in the *SAP HANA Administration Guide*.

File/Section	Properties	Description
auditing configuration	• default_audit_trail_type • emergency_audit_trail_type • alert_audit_trail_type • critical_audit_trail_type • audit_statement_length	Prevents configuration of audit trail targets and the maximum audit statement length
auditing_csvtextfile	• max_file_size • max_files	Prevents configuration of the CSV text file audit trail files
communication	*	Prevents configuration of default key and trust stores, as well as other critical communication settings
global.ini/customizable_functionalities	*	Prevents disabling of restricted features
global.ini/extended_storage	*	Prevents configuration of extended storage (SAP HANA dynamic tiering)
global.ini/persistence	• basepath_datavolumes_es • basepath_logvolumes_es • basepath_databackup_es • basepath_logbackup_es	
global.ini/system_replication	• keep_old_style_alert • enable_full_sync • operation_mode	Prevents configuration of certain system replication settings
global.ini/system_replication_communication	*	
global.ini/system_replication_hostname_resolution	*	
global.ini/xb_messaging	*	Prevents configuration of messaging
multidb.ini/readonly_parameters	*	Prevents configuration of the multidb.ini file itself
indexserver.ini/authentication	SapLogonTicketTrustStore	Prevents configuration of the trust store for user authentication with logon/assertion tickets

File/Section	Properties	Description
`memorymanager`	• `allocationlimit` • `minallocationlimit` • `global_allocation_limit` • `async_free_threshold` • `async_free_target`	Prevents configuration of memory allocation parameters
`execution`	`max_concurrency`	Prevents configuration of threading and parallelization parameters
`session`	• `maximum_connections` • `maximum_external_connections`	
`sql`	`sql_executors`	

Related Information

Prevent Changes to System Properties in Tenant Databases
Unlock Blocklisted Parameters
Copy Blocklisted Parameters

19.3 Unpermitted Characters in User Names

User names can contain any CESU-8 characters except for a small subset.

The following characters are not allowed as user names:

Unicode Character	Character	Name
U+0021	!	Exclamation mark
U+0022	"	Quotation mark
U+0024	$	Dollar sign
U+0025	%	Percent sign
U+0027	'	Apostrophe
U+0028	(	Left parenthesis
U+0029	)	Right parenthesis
U+002A	*	Asterisk
U+002B	+	Plus sign
U+002C	,	Comma
U+002E	.	Full stop

Unicode Character	Character	Name
U+002F	/	Solidus
U+003A	:	Colon
U+003B	;	Semicolon
U+003C	<	Less-than sign
U+003D	=	Equals sign
U+003E	>	Greater-than sign
U+003F	?	Question mark
U+0040	@	Commercial at
U+005B	[	Left square bracket
U+005C	\	Reverse solidus
U+005D	]	Right square bracket
U+005E	^	Circumflex accent
U+0060	`	Grave accent
U+007B	{	Left curly bracket
U+007C	|	Vertical line
U+007D	}	Right curly bracket
U+007E	~	Tilde

19.4 Components Delivered as SAP HANA Content

The following sections provide the technical details, key features, and roles of all software components delivered with the SAP HANA platform as SAP HANA XS classic content.

For more information about what SAP HANA content is, see *SAP HANA Content*.

Related Information

Security of SAP HANA Content [page 330]

19.4.1 Administration

SAP HANA content related to system and database administration

- HANA_ADMIN [page 421]

- HANA_BACKUP [page 421]
- HANA_HDBLCM [page 422]
- HANA_SEC_BASE [page 422]
- HANA_SEC_CP [page 422]
- HANA_SYS_ADMIN [page 422]
- HANA_XS_BASE [page 422]

19.4.1.1 HANA_ADMIN

This component is available for downward compatibility reasons. It does not contain any content.

For more information about Web-based administration using the SAP HANA cockpit in SAP HANA 2.0, see the *SAP HANA Administration Guide*.

Only the following roles are available with the HANA_ADMIN component:

- `sap.hana.admin.roles::SolutionManagerMonitor`
- `sap.hana.admin.roles::RestrictedUserDBSLAccess`

The privileges contained in these roles provide users administrating SAP HANA using SAP Solution Manager and SAP NetWeaver tools with the required authorization in SAP HANA.

→ Recommendation

As repository roles delivered with SAP HANA can change when a new version of the package is deployed, either do not use them directly but instead as a template for creating your own roles, or have a regular review process in place to verify that they still contain only privileges that are in line with your organization's security policy. Furthermore, if repository package privileges are granted by a role, we recommend that these privileges be restricted to your organization's packages rather than the complete repository. To do this, for each package privilege (`REPO.*`) that occurs in a role template and is granted on `.REPO_PACKAGE_ROOT`, check whether the privilege can and should be granted to a single package or a small number of specific packages rather than the full repository.

19.4.1.2 HANA_BACKUP

This component is available for downward compatibility reasons. It does not contain any content.

For more information about Web-based backup using the SAP HANA cockpit in SAP HANA 2.0, see the *SAP HANA Administration Guide*.

19.4.1.3 HANA_HDBLCM

This component is available for downward compatibility reasons. It does not contain any content.

For more information about access to the SAP HANA platform lifecycle management Web user interface from the SAP HANA cockpit in SAP HANA 2.0, see the *SAP HANA Administration Guide*.

19.4.1.4 HANA_SEC_BASE

This component is available for downward compatibility reasons. It does not contain any content.

For more information about Web-based security administration using the SAP HANA cockpit in SAP HANA 2.0, see the *SAP HANA Administration Guide*.

19.4.1.5 HANA_SEC_CP

This component is available for downward compatibility reasons. It does not contain any content.

For more information about Web-based security administration using the SAP HANA cockpit in SAP HANA 2.0, see the *SAP HANA Administration Guide*.

19.4.1.6 HANA_SYS_ADMIN

This component is available for downward compatibility reasons. It does not contain any content.

For more information about Web-based administration of tenant databases using the SAP HANA cockpit in SAP HANA 2.0, see the *SAP HANA Administration Guide*.

19.4.1.7 HANA_XS_BASE

This component provides a Web application for configuring and managing SAP HANA XS classic applications and system-level settings such as SMTP and security-related details (SAML, trust store, and so on).

Technical Details

Delivery unit	HANA_XS_BASE
Prerequisites	None

Content type	Automated content
Content details	SAP HANA XS classic applications (including data model, tables, roles, and user interfaces)
Target users	SAP HANA XS classic application administrators
Web application URL	`http(s)://<host>:<port>/sap/hana/xs/admin`

Key Features

The SAP HANA XS Administration Tool enables users to configure and manage SAP HANA XS classic applications and system-level settings. It provides the following features:

- SAP HANA XS application configuration
  Supports the configuration of application security (public/private) and user authentication methods (basic, form-based, logon tickets, X509, and SAML). It also supports the management of SQL connection configurations, HTTP destinations, and job schedules.
- SAML configuration
  Enables the configuration and management of SAML service providers (URLs, metadata) and identity providers (IDP metadata, certificates, destinations)
- Trust management
  Enables trust store configuration and management, and certificate management
- SMTP configurations
  Enables the configuration and management of e-mail server settings for outbound e-mail connections. It also supports the management of authentication type with credentials, transport security settings, and socket proxy settings.
- User self-service administration
  Enables the administration of user self-service requests (acceptance/rejection of user requests). It also support the activation of users, the granting of roles and the management of access lists for areas such as email address, network domain, and IP range, adding constraints to the user self-service process
- Online Translation Tool
  Enables the user to provide manual translation for text strings used in the application's user interface, error messages, and documentation. Also the user can export and import XLIFF-formatted files into the tool. The tool is integrated with SAP Translation Hub for recommendations of the translated texts.

  > **Note**
  >
  > Access to external translation services is not granted in the SAP HANA license. To use external translation services such as the SAP Translation Hub, an additional license is required.

- SAP Web Dispatcher HTTP Tracing application
  HTTP tracing for individual SAP HANA XS applications can be enabled in the SAP HANA Web Dispatcher. The SAP HANA XS Administration Tools include the SAP Web Dispatcher HTTP Tracing application, which you can use to enable and disable HTTP tracing in the SAP Web Dispatcher for SAP HANA XS applications.

Roles

The following roles are available with this component. Users must have the privileges contained in one or more of these roles before they can use the component and its functions.

> → Recommendation
>
> As repository roles delivered with SAP HANA can change when a new version of the package is deployed, either do not use them directly but instead as a template for creating your own roles, or have a regular review process in place to verify that they still contain only privileges that are in line with your organization's security policy. Furthermore, if repository package privileges are granted by a role, we recommend that these privileges be restricted to your organization's packages rather than the complete repository. To do this, for each package privilege (`REPO.*`) that occurs in a role template and is granted on `.REPO_PACKAGE_ROOT`, check whether the privilege can and should be granted to a single package or a small number of specific packages rather than the full repository.

Role	Description
`sap.hana.xs.debugger::Debugger`	Use of the debugging features of the SAP HANA Web-based Development Workbench
`sap.hana.xs.admin.roles::HTTPDestAdministrator`	Full access to HTTP destination configurations (display and edit)
`sap.hana.xs.admin.roles::HTTPDestViewer`	Read-only access to HTTP destination configurations, which are used to specify connection details for outbound connections, for example using the server-side JavaScript Connectivity API that is included with SAP HANA XS
`sap.hana.xs.admin.roles::JobAdministrator`	Full access to the configuration settings for SAP HANA XS job schedules (defined in .xsjob files); user can specify start/stop times, the user account to run the job, and the locale
`sap.hana.xs.admin.roles::JobSchedulerAdministrator`	Full access to the configuration settings for SAP HANA XS job schedules (defined in .xsjob files); user can specify start/stop times, the user account to run the job, and the locale User can also enable or disable scheduling of jobs.
`sap.hana.xs.admin.roles::JobViewer`	Read-only access to the configuration settings for SAP HANA XS job schedules (defined in .xsjob files).
`sap.hana.xs.formLogin.profile::ProfileOwner`	Management of user profile settings such as date/time format and locale It also allows the changing of user password.
`sap.hana.xs.admin.roles::RuntimeConfAdministrator`	Full access to the configuration settings for SAP HANA XS application security and the related user-authentication providers
`sap.hana.xs.admin.roles::RuntimeConfViewer`	Read-only access to the configuration settings for SAP HANA XS application security and the related user-authentication providers, for example, SAML or X509

Role	Description
`sap.hana.xs.admin.roles::SAMLAdministrator`	Full access to SAML configurations, including both the service provider and the identity providers User can add new entries and make changes to existing service or identity providers, as well as parse the resulting metadata
`sap.hana.xs.admin.roles::SAMLViewer`	Read-only access to SAML configurations that are used to provide details of SAML service providers and identity providers
`sap.hana.xs.admin.roles::SMTPDestAdministrator`	Read-only access to SMTP configurations that are used to specify configuration settings for outbound mail connections to external mail servers
`sap.hana.xs.admin.roles::SMTPDestViewer`	Full access to SMTP configurations (display and edit) User can maintain mail server details, authentication type with credentials, transport security settings and socket proxy settings.
`sap.hana.xs.admin.roles::SQLCCAdministrator`	Full access to SQL connection configurations (SQLCC)
`sap.hana.xs.admin.roles::SQLCCViewer`	Read-only access to SQL connection configurations (SQLCC), which are used to enable the execution of SQL statements from inside your server-side JavaScript application with credentials that are different to the credentials of the requesting user
`sap.hana.xs.admin.roles::TrustStoreAdministrator`	Full access to the SAP HANA XS application trust store that manages the certificates required to start SAP HANA XS applications
`sap.hana.xs.admin.roles::TrustStoreViewer`	Read-only access to the trust store that contains the server's root certificate or the certificate of the certification authority that signed the server's certificate
`sap.hana.xs.admin.roles::USSAdministrator`	Administration of user requests submitted by end users through the User Self-Services application It is also possible to manage access lists for areas such as e-mail address, network domain, and IP range, adding constraints to the user self-service process
`sap.hana.xs.admin.roles::USSExecutor`	Role assigned to technical user to enable user self-service application in the system
`sap.hana.xs.wdisp.admin::WebDispatcherAdmin`	Full access to the SAP HANA Web Dispatcher Administration tool used by administrators to maintain secure inbound communication, for example, to enable SSL/TLS connections between browser front-ends or an ABAP system and an SAP HANA XS application
`sap.hana.xs.wdisp.admin::WebDispatcherMonitor`	Read-only access to the information displayed in the SAP HANA Web Dispatcher Administration tool
`Translator`	Enables an SAP HANA user to maintain translation text strings with the SAP HANA Online Translation Tool

Role	Description
`WebDispatcherHTTPTracingViewer`	Read-only access to the HTTP setting of SAP HANA XS applications running on the selected SAP HANA instance. This role extends the `JobViewer` role to enable the user to view details of the xsjob configuration (`httptracing.xsjob`) that starts and stops the HTTP tracing tasks.
`WebDispatcherHTTPTracingAdministrator`	Full access required to maintain HTTP tracing in the SAP Web Dispatcher for SAP HANA XS applications. This role extends the `JobAdministrator` role to enable the user to maintain the XS job file (`httptracing.xsjob`) used to configure and enable HTTP tracing for XS applications in the SAP Web Dispatcher.

19.4.2 Application Lifecycle Management

SAP HANA content for application lifecycle management

- HANA_XS_LM [page 426]

19.4.2.1 HANA_XS_LM

This component provides a Web application for the application lifecycle management of components developed for SAP HANA XS.

Technical Details

Delivery unit	`HANA_XS_LM`
Prerequisites	SAPUI5_1
Content type	Automated content
Content details	SAP HANA XS classic application
Target users	Application developers, content administrators
Web application URL	`http(s)://<host>:<port>/sap/hana/xs/lm`

Key Features

The SAP HANA Application Lifecycle Management application enables application developers to create products, delivery units, packages, and basic application components. Administrators can use the application

to set up the transport of delivery units, start and monitor transports, and upload or download delivery unit archives.

Roles

The following roles are available with the SAP HANA Application Lifecycle Management component. Users must have the privileges contained in one or more of these roles before they can use the component and its functions.

> → Recommendation
>
> As repository roles delivered with SAP HANA can change when a new version of the package is deployed, either do not use them directly but instead as a template for creating your own roles, or have a regular review process in place to verify that they still contain only privileges that are in line with your organization's security policy. Furthermore, if repository package privileges are granted by a role, we recommend that these privileges be restricted to your organization's packages rather than the complete repository. To do this, for each package privilege (`REPO.*`) that occurs in a role template and is granted on `.REPO_PACKAGE_ROOT`, check whether the privilege can and should be granted to a single package or a small number of specific packages rather than the full repository.

Role	Description
`sap.hana.xs.lm.roles::Administrator`	Full read/write access to all the features in the SAP HANA application lifecycle management tool, including the access privileges granted to all other user roles available in the SAP HANA application lifecycle management, for example, Display, Execute Transport, and Transport.
`sap.hana.xs.lm.roles::Developer`	Enables the user to work on a change to which he is assigned and to approve own contributions to the change. This role includes the privileges of the Display role.
`sap.hana.xs.lm.roles::DevelopmentExpert`	Enables the user to perform all actions involved in change recording (for example, create, assign objects to, release, delete, assign other users to a change, approve own or foreign contributions). This role includes the privileges of the Display and the Developer roles.
`sap.hana.xs.lm.roles::Display`	View-only access; some features and options are hidden. A user with a role based on this role template can view all information available but cannot make any changes or trigger any transport operations.
`sap.hana.xs.lm.roles::Execute Transport`	Users with a role based on this role template can view all information as well as trigger predefined transport operations. However, they cannot register or maintain systems, create transport routes, or edit details of a product, a delivery unit, or a package.

Role	Description
`sap.hana.xs.lm.roles::Transport`	For technical users only. A role based on this role template cannot be assigned to normal users; it is granted as part of the Execute Transport role. The Transport role grants the privileges required for export or import actions during a transport operation. The credentials and privileges of a technical user with the Transport role cannot be used for interactive logons, for example, to start SAP HANA application lifecycle management.
`sap.hana.xs.lm.roles::SLP_display`	For technical users used for HTTP-based deployment when using CTS Transport. Users with a role based on this role template can perform all supported read requests for SL protocol services.
`sap.hana.xs.lm.roles::SLP_CTS_deploy_admin`	For technical users used for HTTP-based deployment when using CTS Transport. Users with a role based on this role template can perform all supported requests for CTS Deploy SL protocol service.
`sap.hana.xs.lm.roles::SLP_CTS_ping_admin`	For technical users used for HTTP-based deployment when using CTS Transport. Users with a role based on this role template can perform all supported requests for CTS Ping SL protocol service.

For tasks that require interaction with external tools, the privileges in the following additional roles are required:

Role	Description
`sap.hana.ide.roles::EditorDeveloper`	Inspect, create, change, delete and activate SAP HANA repository objects A role based on this role template is required when you select the *Packages* tile in order to maintain SAP HANA repository packages in Web-based Development Workbench.
`sap.hana.xs.admin.roles::HTTPDestAdministrator`	Full access to HTTP destination configurations (display and edit) A role based on this role template is required when you register a system for a transport route.
`sap.hana.xs.admin.roles::RuntimeConfAdministrator`	Full access to the configuration settings for SAP HANA XS application security and the related user-authentication providers A role based on this role template is required when you register a system for a transport route.

The the privileges in the following roles are required for SAP HANA Application Lifecycle Management Process Engine:

Role	Description
`sap.hana.xs.lm.pe.roles::PE_Display`	The user can monitor processes and display services

Role	Description
`sap.hana.xs.lm.pe.roles::PE_Execution`	In addition to the previous role, the user can start, stop, skip, and resume processes.
`sap.hana.xs.lm.pe.roles::PE_Activation`	In addition to the previous roles, the user can activate services from repository files.
`sap.hana.xs.lm.pe.roles::PE_Administrator`	This role includes all previous roles.

19.4.3 Runtime Libraries

SAP HANA content for runtime libraries

- HANA_XS_DBUTILS [page 429]

19.4.3.1 HANA_XS_DBUTILS

This component provides content for the simplified consumption of SAP HANA database objects for XSJS.

Technical Details

Delivery unit	`HANA_XS_DBUTILS`
Prerequisites	None
Content type	Automated content
Content details	XSJS libraries
Target users	Developers using the libraries for more convenient access to SAP HANA database objects
Web application URL	None

Key Features

The SAP HANA XS DB UTILITY LIBS component comes as set of XS JavaScript libraries that wrap the database interface of XS with JavaScript-native access methods and object representations:

- Invocation of SQL procedures as if they were JavaScript functions
- JavaScript CDS client and query builder

These libraries can be consumed only by applications deployed on XS. In other words, they cannot be accessed directly via HTTP from outside the XS container.

Roles

This component does not come with any roles. As the component simply wraps the standard XS database interface, the role definitions and authorizations of that interface directly apply.

19.4.4 Configuration

SAP HANA content for configuration

- HANA_TA_CONFIG [page 430]

19.4.4.1 HANA_TA_CONFIG

This component provides predefined configurations and dictionaries used by the SAP HANA text analysis engine and by text mining.

Technical Details

Delivery unit	`HANA_TA_CONFIG`
Prerequisites	None
Content type	Automated content
Content details	Configuration files
Target users	SAP HANA developers
Web application URL	None

Key Features

The Text Analysis Configuration component includes the following:

- Predefined configuration files containing text analysis options to be used when creating a full text index
- Predefined configuration files containing text mining options

Roles

This component does not come with any roles.

The Text Analysis Configuration component contains configuration files. It does not contain any executable software. Any user with permission to execute the SQL statement CREATE FULLTEXT INDEX can use the text analysis engine and text mining, which use the HANA_TA_CONFIG data.

19.4.5 Supportability and Development

SAP HANA content for supportability and development

- HANA_IDE_CORE [page 431]
- HANA_XS_IDE, HANA_XS_EDITOR [page 432]
- HANA_DT_BASE [page 433]

19.4.5.1 HANA_IDE_CORE

This component provides a Web-based integrated development environment (IDE) that can be used to build and test development artifacts in SAP HANA. The SAP HANA Web-based Development Workbench is a quick and easy alternative to the SAP HANA studio for developing native SAP HANA applications in SAP HANA XS classic.

Technical Details

Delivery unit	`HANA_IDE_CORE`
Prerequisites	`SAPUI5_1`, `SAP_WATT`, `HANA_XS_BASE`
Content type	Automated content
Content details	SAP HANA applications
Target users	SAP HANA developers and support staff
Web application URL	http(s)://`<host>`:`<port>`/sap/hana/ide

Key Features

The SAP HANA Web-based Development Workbench includes the following tools:

- Editor (IDE)
  Inspect, create, change, delete , and activate SAP HANA repository objects or development artifacts such as database entities, XS JavaScript code, Web content (HTML, CSS, etc.), OData service definitions
- Catalog
  Create, edit, execute, and manage SQL catalog artifacts
- Security

Manage users and roles, assign objects and manage security

- Traces
  View and download traces for SAP HANA XS applications and set trace levels

Roles

The following roles are available with the SAP HANA IDE component. Users must have the privileges contained in one or more of these roles before they can use the component and its functions.

> → Recommendation
>
> As repository roles delivered with SAP HANA can change when a new version of the package is deployed, either do not use them directly but instead as a template for creating your own roles, or have a regular review process in place to verify that they still contain only privileges that are in line with your organization's security policy. Furthermore, if repository package privileges are granted by a role, we recommend that these privileges be restricted to your organization's packages rather than the complete repository. To do this, for each package privilege (`REPO.*`) that occurs in a role template and is granted on `.REPO_PACKAGE_ROOT`, check whether the privilege can and should be granted to a single package or a small number of specific packages rather than the full repository.

Role	Description
`sap.hana.ide.roles::Developer`	A combined user role which incorporates all the following roles and provides access to all tools
`sap.hana.ide.roles::EditorDeveloper`	Provides access to the IDE/Editor tool
`sap.hana.ide.roles::CatalogDeveloper`	Provides access to the Catalog tool
`sap.hana.ide.roles::TraceViewer`	Provides access to the Trace tool
`sap.hana.ide.roles::SecurityAdmin`	Provides access to the Security tool

19.4.5.2 HANA_XS_IDE, HANA_XS_EDITOR

These components provide browser redirection to the SAP HANA Web-based Development Workbench.

> ⓘ Note
>
> The components HANA_XS_IDE and HANA_XS_EDITOR are available for downward compatibility reasons. They do not contain any functionality except redirection to the SAP HANA Web-based Development Workbench.

19.4.5.3 HANA_DT_BASE

This component provides the SAP HANA REST application programming interface (API).

The SAP HANA REST API allows development tools to access the SAP HANA repository and database catalog via HTTP(S) in a standard-compliant way. It builds upon the Eclipse Orion server API protocol version 1 on SAP HANA. For SAP- specific tools, the Orion server protocol has been extended with SAP HANA-specific features such as activation, change tracking, and database catalog search

Technical Details

Delivery unit	HANA_DT_BASE
Prerequisites	None
Content type	Automated content
Content details	SAP HANA REST API
Target users	SAP HANA developers and support staff
Web application URL	None

Key Features

The SAP HANA REST API includes the following features:

- File and folder operations such as reading, writing, moving and deleting files and folders (packages)
  It is possible to read and write file and folder metadata. Examples of SAP-specific metadata are the version, the activation time, and the activating user. In addition to the Orion standard, mass operations are available to get and set the metadata of many files with one request.
- Activation of repository objects
- Change tracking
- Handling of user preference data (for example, the SAP HANA Web-based Development Workbench and other development and support tools)
- Existence checks and search suggestions for metadata
  These functions can be used to implement searching in the repository and in the database catalog, with auto-completion. The metadata suggestion request returns all resources that match a specified pattern

Roles

The following roles are available with the REST API component. Users must have the privileges contained in one or more of these roles before they can use the component and its functions. Additionally, users need the appropriate authorization on SAP HANA repository entities and catalog entities to be able to view or change repository or database content.

→ Recommendation

As repository roles delivered with SAP HANA can change when a new version of the package is deployed, either do not use them directly but instead as a template for creating your own roles, or have a regular review process in place to verify that they still contain only privileges that are in line with your organization's security policy. Furthermore, if repository package privileges are granted by a role, we recommend that these privileges be restricted to your organization's packages rather than the complete repository. To do this, for each package privilege (`REPO.*`) that occurs in a role template and is granted on `.REPO_PACKAGE_ROOT`, check whether the privilege can and should be granted to a single package or a small number of specific packages rather than the full repository.

Role	Description
`sap.hana.xs.dt.base::restapi`	Allows users to access the REST API

19.4.6 User Interface

SAP HANA content for user interface

- HANA_UI_INTEGRATION_SVC, HANA_UI_INTEGRATION_CONTENT [page 434]
- SAPUI5_1 [page 436]
- SAP_WATT [page 436]

19.4.6.1 HANA_UI_INTEGRATION_SVC, HANA_UI_INTEGRATION_CONTENT

These components provide SAP HANA UI Integration Services (UIS), which is a set of Eclipse-based tools and client-side APIs that enable you to integrate standalone SAP HANA client applications into Web-based application sites.

Technical Details

HANA_UI_INTEGRATION_SVC

Delivery unit	`HANA_UI_INTEGRATION_SVC`
Prerequisites	None
Content type	Automated content
Content details	Database tables, views, stored procedures, UIs, HTML, JavaScript
Target users	Developers (design time) and end users (runtime)

Web application URL	None

HANA_UI_INTEGRATION_CONTENT

Delivery unit	`HANA_UI_INTEGRATION_CONTENT`
Prerequisites	`HANA_UI_INTEGRATION_SVC`
Content type	Automated content
Content details	Application sites and catalogs (.xsappsite and .xswidget files)
Target users	XS developers
Web application URL	None

Key Features

- For developers and designers: tools for creating content and designing application sites
- For end users: personalization capabilities and role-based access to application sites and their content

Roles

The following roles are available with the SAP HANA UIS components. Users must have the privileges contained in one or more of these roles before they can use the component and its provided services.

> → Recommendation
>
> As repository roles delivered with SAP HANA can change when a new version of the package is deployed, either do not use them directly but instead as a template for creating your own roles, or have a regular review process in place to verify that they still contain only privileges that are in line with your organization's security policy. Furthermore, if repository package privileges are granted by a role, we recommend that these privileges be restricted to your organization's packages rather than the complete repository. To do this, for each package privilege (`REPO.*`) that occurs in a role template and is granted on `.REPO_PACKAGE_ROOT`, check whether the privilege can and should be granted to a single package or a small number of specific packages rather than the full repository.

Role	Description
`sap.hana.uis.db::SITE_DESIGNER`	Create and edit standard and Fiori Launchpad applications sites and catalogs Assign permissions to standard and Fiori Launchpad application sites and their content
`sap.hana.uis.db::SITE_USER`	Access standard and Fiori Launchpad application sites and catalogs

19.4.6.2 SAPUI5_1

This component provides SAP UI5, which is the library used by XSC-based Web applications and tools to implement the specific user interfaces.

All XSC-based Web applications delivered with SAP HANA such as SAP HANA Application Lifecycle Management and the SAP HANA Web-based Development Workbench rely on this delivery unit.

Technical Details

Delivery unit	`SAPUI5_1`
Prerequisites	None
Content type	Automated content
Content details	Web content such as HTML, CSS, JavaScript
Target users	Used by XS-based Web applications
Web application URL	None

Roles

Since this component provides purely Web content consumed by arbitrary Web applications, it is not protected by any specific mechanisms. Any browser can download the artifacts in this library.

19.4.6.3 SAP_WATT

This component provides the SAP WATT Web library, which is an additional Web library used by the SAP HANA Web-based Development Workbench. It contains additional Web content such as HTML, CSS, and JavaScript libraries to build Web development environments.

All XSC-based Web applications delivered with SAP HANA such as SAP HANA Application Lifecycle Management and SAP HANA Web-based Development Workbench rely on this delivery unit.

Technical Details

Delivery unit	`SAP_WATT`
Prerequisites	None
Content type	Automated content

Content details	Web content such as HTML, CSS, JavaScript
Target users	Used by the SAP HANA Web-based Development Workbench
Web application URL	None

Roles

Since this component provides purely Web content consumed by arbitrary Web applications, it is not protected by any specific mechanisms. Any browser can download the artifacts in this library.

19.4.7 Documentation

SAP HANA documentation delivered as SAP HANA content

- HDC_* [page 438]

19.4.7.1 HDC_*

These components provide product documentation for several Web applications delivered with SAP HANA. Users can access the documentation via a tile on the application homepage and from the Help menu if available.

Technical Details

Delivery unit

- `HDC_XS_BASE`
- `HDC_IDE_CORE`
- `HDC_XS_LM`

> **Note**
>
> The following documentation DUs are available for downward compatibility reasons. They do not contain any content.
>
> - `HDC_ADMIN`
> - `HDC_SYS_ADMIN`
> - `HDC_SEC_CP`
> - `HDC_BACKUP`
>
> For Web-based administration using the SAP HANA cockpit in SAP HANA 2.0, see the *SAP HANA Administration Guide*.

Prerequisites	None
Content type	Automated content
Content details	HTML files, image files
Target users	Application users
Web application URL	`http(s)://<host>:<port>/public/sap/docs/hana`

Features

Product documentation is available for the following Web applications delivered with SAP HANA:

- SAP HANA XS Admin Tools
- SAP HANA Application Lifecycle Management
- SAP HANA Web-based Development Workbench

Roles

These components do not come with any roles. Access to the content is controlled by the standard XS-application security mechanism, the .xsaccess file.

Important Disclaimer for Features in SAP HANA

For information about the capabilities available for your license and installation scenario, refer to the Feature Scope Description for SAP HANA.

Important Disclaimers and Legal Information

Hyperlinks

Some links are classified by an icon and/or a mouseover text. These links provide additional information.
About the icons:

- Links with the icon : You are entering a Web site that is not hosted by SAP. By using such links, you agree (unless expressly stated otherwise in your agreements with SAP) to this:
  - The content of the linked-to site is not SAP documentation. You may not infer any product claims against SAP based on this information.
  - SAP does not agree or disagree with the content on the linked-to site, nor does SAP warrant the availability and correctness. SAP shall not be liable for any damages caused by the use of such content unless damages have been caused by SAP's gross negligence or willful misconduct.
- Links with the icon : You are leaving the documentation for that particular SAP product or service and are entering an SAP-hosted Web site. By using such links, you agree that (unless expressly stated otherwise in your agreements with SAP) you may not infer any product claims against SAP based on this information.

Videos Hosted on External Platforms

Some videos may point to third-party video hosting platforms. SAP cannot guarantee the future availability of videos stored on these platforms. Furthermore, any advertisements or other content hosted on these platforms (for example, suggested videos or by navigating to other videos hosted on the same site), are not within the control or responsibility of SAP.

Beta and Other Experimental Features

Experimental features are not part of the officially delivered scope that SAP guarantees for future releases. This means that experimental features may be changed by SAP at any time for any reason without notice. Experimental features are not for productive use. You may not demonstrate, test, examine, evaluate or otherwise use the experimental features in a live operating environment or with data that has not been sufficiently backed up.
The purpose of experimental features is to get feedback early on, allowing customers and partners to influence the future product accordingly. By providing your feedback (e.g. in the SAP Community), you accept that intellectual property rights of the contributions or derivative works shall remain the exclusive property of SAP.

Example Code

Any software coding and/or code snippets are examples. They are not for productive use. The example code is only intended to better explain and visualize the syntax and phrasing rules. SAP does not warrant the correctness and completeness of the example code. SAP shall not be liable for errors or damages caused by the use of example code unless damages have been caused by SAP's gross negligence or willful misconduct.

Bias-Free Language

SAP supports a culture of diversity and inclusion. Whenever possible, we use unbiased language in our documentation to refer to people of all cultures, ethnicities, genders, and abilities.

www.sap.com/contactsap

© 2026 SAP SE or an SAP affiliate company. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP SE or an SAP affiliate company. The information contained herein may be changed without prior notice.

Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP SE (or an SAP affiliate company) in Germany and other countries. All other product and service names mentioned are the trademarks of their respective companies.

Please see https://www.sap.com/about/legal/trademark.html for additional trademark information and notices.