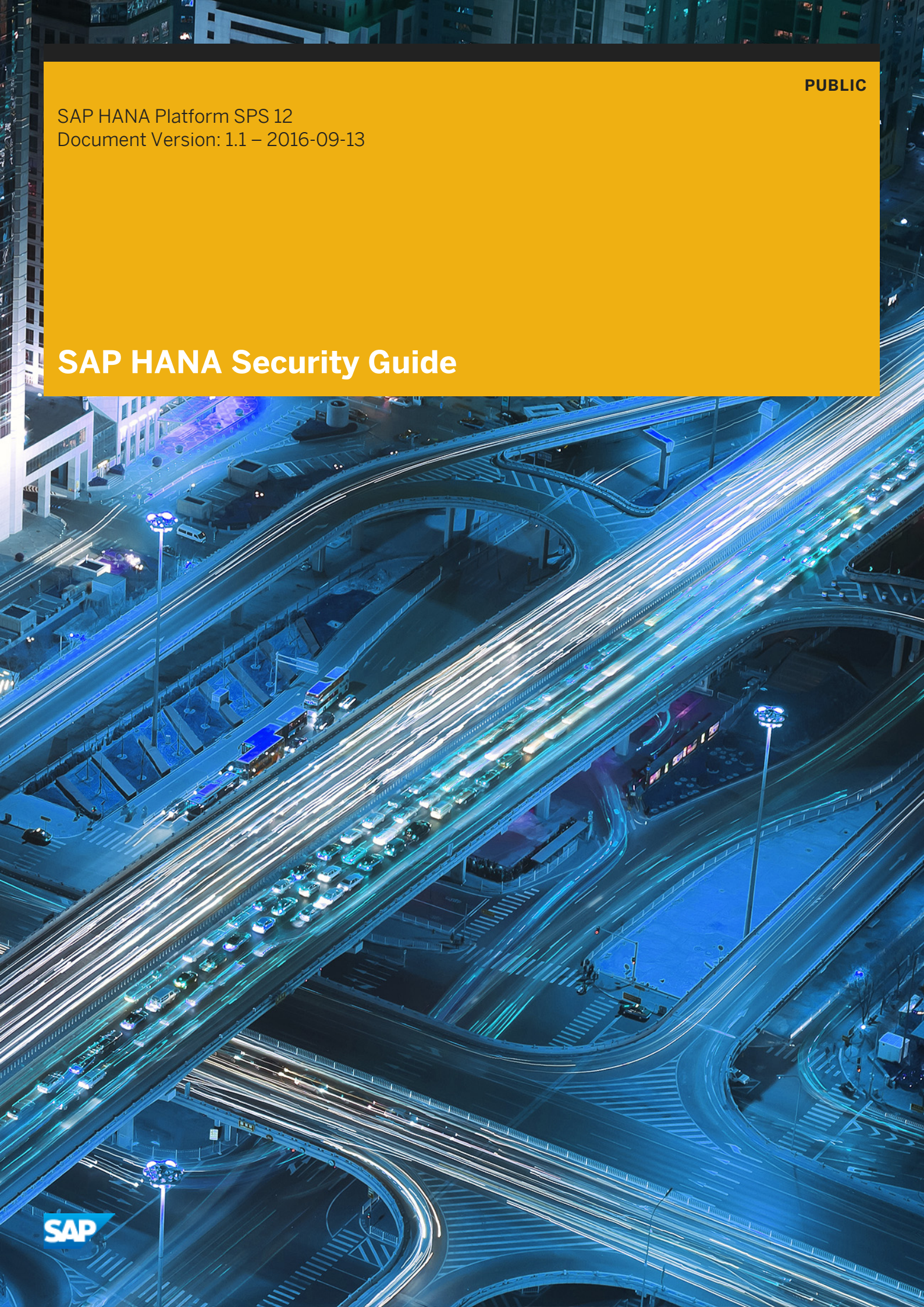
PUBLIC

SAP HANA Platform SPS 12
Document Version: 1.1 – 2016-09-13

SAP HANA Security Guide



Content

1 Important Critical Configurations. 6

2 Introduction to SAP HANA Security. 7

3 SAP HANA Security Patches. 11

4 SAP HANA Overview. 13

4.1 The SAP HANA Database. 13

4.2 SAP HANA XS and Development Infrastructure. 14

4.3 SAP HANA Implementation Scenarios. 15

SAP HANA as a Data Mart. 16

SAP HANA in a Classic 3-tier Architecture. 18

SAP HANA as Technical Infrastructure for Native Applications, Classic. 19

4.4 SAP HANA Multitenant Database Containers. 21

Security for Multitenant Database Containers. 24

Database Isolation. 26

5 SAP HANA Network and Communication Security. 29

5.1 Communication Channels. 29

5.2 Network Security. 33

5.3 Securing Data Communication. 36

Secure Communication Between SAP HANA and JDBC/ODBC Clients. 38

Secure Communication Between SAP HANA XS Classic and HTTP Clients. 50

Secure Internal Communication. 51

6 SAP HANA User Management. 62

6.1 User Types. 63

6.2 User Administration Tools. 64

6.3 Predefined Users. 66

6.4 Deactivate the SYSTEM User. 71

6.5 SYSTEM User in Multitenant Database Containers. 73

7 SAP HANA Authentication and Single Sign-On. 74

7.1 User Authentication Mechanisms. 75

User Authentication in Multitenant Database Containers. 77

7.2 SAP HANA Logon Checks. 77

7.3 Password Policy. 77

Password Policy Configuration Options. 78

Password Blacklist. 86

7.4 Single-Sign On Integration. 86
Single Sign-On Using Kerberos. 87
Single Sign-On Using SAML 2.0. 88
Single Sign-On Using SAP Logon and Assertion Tickets. 90

8 SAP HANA Authorization. 92
8.1 Privileges. 93
System Privileges. 96
Object Privileges. 102
Analytic Privileges. 107
Package Privileges. 109
Application Privileges. 111
8.2 Roles. 112
Predefined Database Roles. 114
Repository Roles. 117
8.3 Authorization in the Repository of the SAP HANA Database. 124
Developer Authorization in the Repository. 125
_SYS_REPO Authorization in the Repository. 127
Granting and Revoking Privileges on Activated Repository Objects. 127
8.4 Cross-Database Authorization in Multitenant Database Containers. 129

9 Data Storage Security in SAP HANA. 132
9.1 Server-Side Data Encryption. 134
Encryption Key Management. 138
Cryptographic Service Provider. 141
9.2 Data Volume Encryption. 141
Data Volume Encryption in Multitenant Database Containers. 144
9.3 Secure Storage of Passwords in SAP HANA. 145
Secure Internal Credential Store. 146
Secure User Store (hdbuserstore). 148
9.4 Protection of Data in SAP HANA Studio Workspaces. 152

10 Auditing Activity in SAP HANA Systems. 153
10.1 Audit Policies. 153
Actions Audited by Default Audit Policy. 156
10.2 Audit Trails. 158
Audit Trail Layout for Trail Target CSV and SYSLOG. 160
Audit Trail Layout for Trail Target Database Table. 162
10.3 Auditing Configuration and Audit Policy Management. 164
System Properties for Configuring Auditing. 165
10.4 Best Practices for Creating Audit Policies. 167

11 Certificate Management in SAP HANA. 168

11.1 Client Certificates . . . 170
11.2 Certificate Collections . . . 171
11.3 SQL Statements and Authorization for In-Database Certificate Management . . . 172

12 Security Risks of Trace and Dump Files . . . 176

13 Security for SAP HANA Extended Application Services, Advanced Model . . . 177
13.1 Technical System Landscape of SAP HANA XS Advanced . . . 178
Application Server Components . . . 181
Users and Clients . . . 183
13.2 User Administration and Authentication in SAP HANA XS Advanced . . . 184
User Management . . . 184
Predefined XSA Users . . . 186
Predefined Database Roles for XSA . . . 190
User Authentication . . . 192
User Administration Tools . . . 193
13.3 Authorization in SAP HANA XS Advanced . . . 193
Organizations and Spaces . . . 194
Scopes, Attributes, and Role Collections . . . 197
Controller Role Model . . . 200
Authorization Management Tools . . . 203
13.4 Network and Communication Security with SAP HANA XS Advanced . . . 205
Security Areas . . . 205
Public Endpoints . . . 207
Single-Host Scenario . . . 208
Multiple-Host Scenario . . . 209
Certificate Management . . . 211
13.5 Data Storage Security . . . 213
System Component Storage . . . 214
Application Storage . . . 215
13.6 Security Aspects of Data, Data Flow, and Processes . . . 216
Scenario: Login with xs CLI . . . 218
Scenario: Pushing an Application with xs CLI . . . 220
Scenario: Access Application Data via Browser . . . 222
13.7 Security-Relevant Logging and Tracing . . . 225
Audited Operations . . . 225
Audit Trails . . . 226

14 Security Aspects of SAP Web IDE for SAP HANA . . . 227
14.1 User Authorization and Authentication . . . 229
14.2 Known Security-Related Issues . . . 230

15 Security for Other SAP HANA Platform Components . . . 231

15.1 SAP HANA Platform Lifecycle Management (Security). 232
15.2 SAP HANA Content (Security). 232
15.3 SAP HANA Smart Data Access (Security). 233
15.4 SAP HANA R Integration (Security). 234
15.5 SAP HANA Information Composer (Security). 235

16 Security for SAP HANA Replication Technologies. 237

17 SAP HANA Security Reference Information. 240
17.1 Security Reference for Multitenant Database Containers. 240
Restricted Features in Multitenant Database Containers. 241
Default Blacklisted System Properties in Multitenant Database Containers. 243
17.2 Components Delivered as SAP HANA Content 245
Administration. 245
Application Lifecycle Management. 257
Runtime Libraries. 259
Configuration. 260
Supportability and Development. 261
User Interface. 264
Documentation. 267

1 Important Critical 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 specifically 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 (For SAP HANA Database)* on SAP Help Portal.

We recommend that you verify your system for critical configurations and latest security patches. Specifically, we recommend verifying that:

- The initial default 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.
- 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 (For SAP HANA Database)* on SAP Help Portal.

For more information about keeping your system up to date by installing the latest security patches, see SAP HANA Security Patches [page 11].

Related Information

SAP HANA Security Checklists and Recommendations (For SAP HANA Database) (PDF)
SAP HANA Security Checklists and Recommendations (For SAP HANA Database) (HTML)

2 Introduction to SAP HANA Security

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 for SAP HANA options and capabilities, such as SAP HANA dynamic tiering and SAP HANA smart data streaming. For more information about the security of options and capabilities, see *SAP HANA Options and Capabilities* on SAP Help Portal. Be aware that you need additional licenses for SAP HANA options and capabilities. For more information, see Important Disclaimer for Features in SAP HANA Platform, Options and Capabilities [page 270].

Why is Security Necessary?

Protecting corporate information is one of the most important topics 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.

Security Information Map

In addition to the 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. The following figure shows you where you'll find which information.

> **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://hana.sap.com/security.

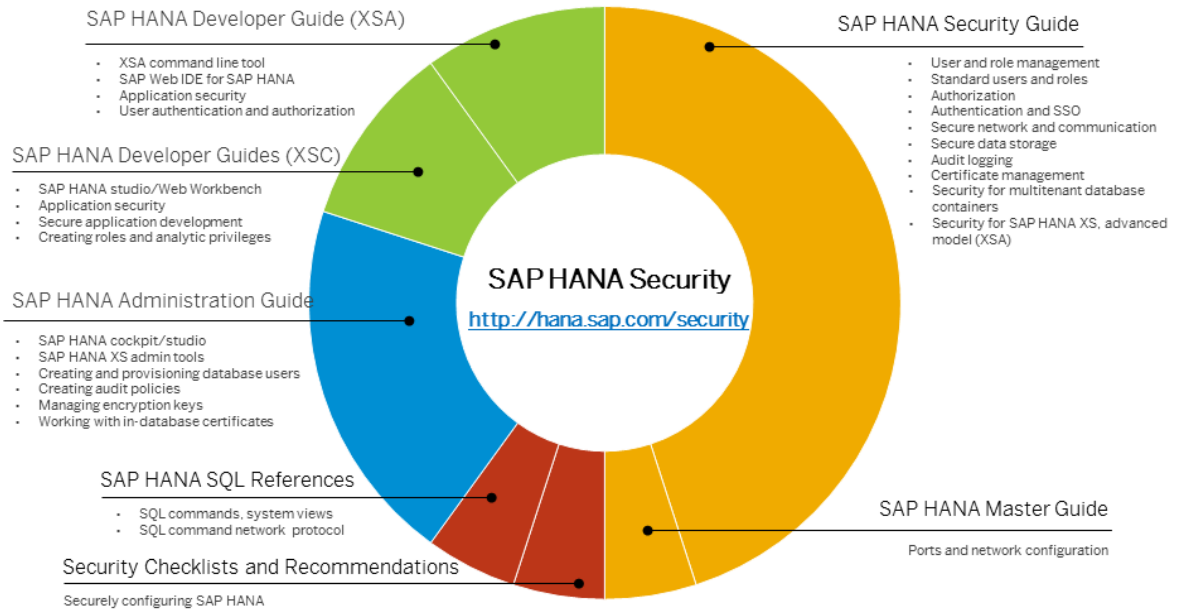


Figure 1: Security Information Map

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

Table 1: Target Audiences

Document	Target Audience	Content Type
SAP HANA Security Guide	Technology consultants, security consultants, system administrators	Concept and overview
SAP HANA Master Guide	Technology consultants, security consultants, system administrators	Concept and overview
Security Checklists and Recommendations	System administrators	Reference
SAP HANA Administration Guide	System administrators	Task- and role-oriented
SAP HANA Developer Guides (XSA)	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 Developer Guides (XSC)	Database developers, application programmers, and client UI developers working in the SAP HANA extended services (SAP HANA XS) classic model using either the SAP HANA studio or SAP HANA Web-based Developer Workbench	Task- and role-oriented
SAP HANA SQL and System Views Reference	Technology consultants, security consultants, system administrators	Reference

Document	Target Audience	Content Type
SAP HANA SQL Command Network Protocol Reference	Developers	Reference

Additional Documentation Resources

SAP HANA Documentation

For more information about the SAP HANA landscape, including installation and administration, see SAP Help Portal at http://help.sap.com/hana_platform.

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 *SAP HANA 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.

Table 2:

SAP Note	Title
1514967	SAP HANA: Central Note
1730928	Using external software in an SAP HANA appliance
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

For more information about specific topics, see the quick links in the table below.

Table 3:

Content	SAP Service Marketplace or SDN Quick Link
Other SAP Security Guides	https://service.sap.com/securityguide
Related SAP Notes	https://service.sap.com/notes https://service.sap.com/securitynotes
Released platforms	https://service.sap.com/pam
SAP Solution Manager	https://service.sap.com/solutionmanager
SAP NetWeaver	http://sdn.sap.com/irj/sdn/netweaver
In-memory computing	http://www.sdn.sap.com/irj/sdn/in-memory

Related Information

SAP HANA Options and Capabilities
SAP HANA Security Patches [page 11]

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 Master Guide

SAP HANA Server Installation and Update Guide

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.

The SAP HANA Database [page 13]
At the core of SAP HANA is the high-performance, in-memory SAP HANA database.

SAP HANA XS and Development Infrastructure [page 14]
SAP HANA includes the SAP HANA extended application services (SAP HANA XS), a layer on top of the SAP HANA database that provides the platform for running SAP HANA-based Web applications.

SAP HANA Implementation Scenarios [page 15]
How you implement SAP HANA determines what you need to consider from a security perspective.

SAP HANA Multitenant Database Containers [page 21]
SAP HANA supports multiple isolated databases in a single SAP HANA system. These are referred to as multitenant database containers.

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. SAP HANA 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.

For more information about data replication technologies, see *Security for SAP HANA Replication Technologies*.

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 237]

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 the SAP HANA database that provides the platform for running SAP HANA-based Web applications.

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.

SAP HANA XS, Advanced Model

From SPS 11, SAP HANA includes an additional run-time environment for application development: SAP HANA extended application services (XS), advanced model. 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.

For detailed information about the security architecture of SAP HANA XS, advanced model, see *Security for SAP HANA Extended Application Services, 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.

Related Information

SAP HANA as Technical Infrastructure for Native Applications, Classic [page 19]
Security for SAP HANA Extended Application Services, Advanced Model [page 177]
Security Aspects of SAP Web IDE for SAP HANA [page 227]

4.3 SAP HANA Implementation Scenarios

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

SAP HANA as a Data Mart [page 16]

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 18]

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 Applications, Classic [page 19]

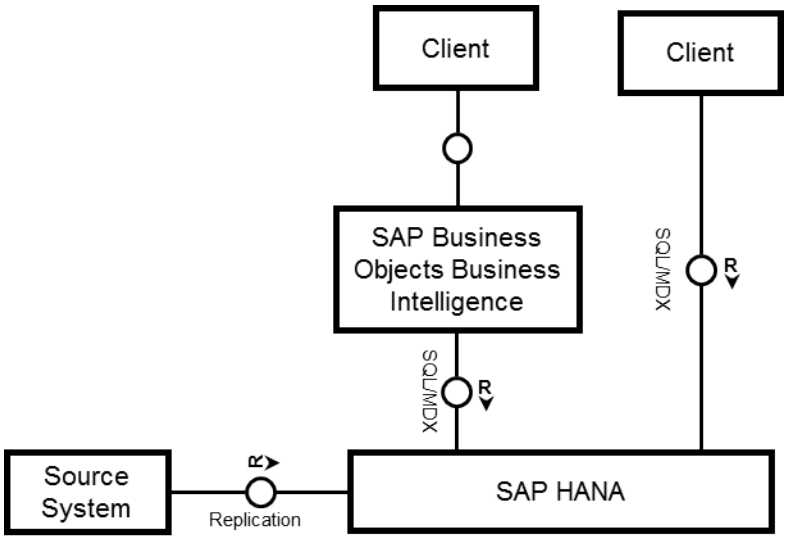
SAP HANA Extended Application Services (SAP HANA XS), classic model, embeds a full-featured application server, Web server, and development environment within SAP HANA. Applications can be developed and deployed directly on SAP HANA XS, which exposes them to end users through a web interface.

4.3.1 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:

Figure 2: 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.

Table 4:

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.
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 of data persistence layer	Data volume encryption ensures that anyone who can access the data volume on disk using operating system commands cannot see the actual data.
Auditing	Actions performed in the SAP HANA database can be audited.

Related Information

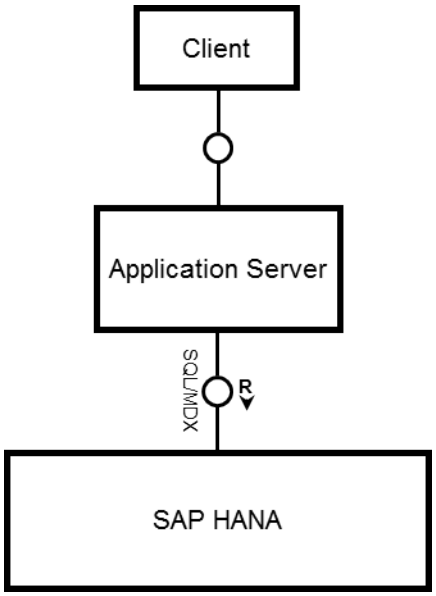
SAP HANA User Management [page 62]
SAP HANA Authentication and Single Sign-On [page 74]
SAP HANA Authorization [page 92]
Securing Data Communication [page 36]
Data Volume Encryption [page 141]
Auditing Activity in SAP HANA Systems [page 153]

4.3.2 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:

Figure 3: 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.

Table 5:

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 SAP HANA HDBSQL 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 of data persistence layer	Data volume encryption ensures that anyone who can access the data volume on disk using operating system commands cannot see the actual data.
Auditing	Actions performed in the SAP HANA database can be audited.

Related Information

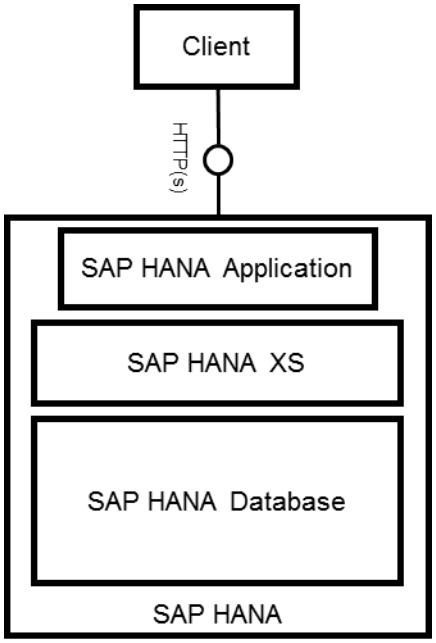
SAP HANA User Management [page 62]
SAP HANA Authentication and Single Sign-On [page 74]
SAP HANA Authorization [page 92]
Securing Data Communication [page 36]
Data Volume Encryption [page 141]
Auditing Activity in SAP HANA Systems [page 153]

4.3.3 SAP HANA as Technical Infrastructure for Native Applications, Classic

SAP HANA Extended Application Services (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.

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

Figure 4: SAP HANA as Technical Infrastructure for Classic Native Applications



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.

Table 6:

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	Secure Sockets Layer (SSL) and Transport Layer Security (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 of data persistence layer	Data volume encryption ensures that anyone who can access the data volume on disk using operating system commands cannot see the actual data.
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 *SAP HANA Developer Guide*.

- Application access and authorization
  The application access file (`.xsaccess`) specifies who or what is authorized to access the content exposed by the application package and what content they are allowed to see. For example, you use the application access file to specify if authentication is to be used to check access to package content, and whether rewrite rules are in place for the exposure of target and source URLs.
  User authentication methods and other aspects of application-security can also be configured with the XSJS application tool SAP HANA XS Administration, which is included along with other XS applications as automated content.
- Data authorization (privileges for users, roles, views, schemas, tables, packages, applications, repository, and so on)
- Server-side JavaScript (scripting best practices for XSS, XSRF, and so on; debugging roles, user authentication for debug sessions)
- ODATA services (service definition, service start, URLs, write access)
- XMLA services (service definition, service start, URLs)
- Table import (transport by key areas)

Related Information

SAP HANA User Management [page 62]
SAP HANA Authentication and Single Sign-On [page 74]
SAP HANA Authorization [page 92]
Securing Data Communication [page 36]
Data Volume Encryption [page 141]
Auditing Activity in SAP HANA Systems [page 153]
SAP HANA Developer Guide (For SAP HANA Studio)
SAP HANA Developer Guide (For SAP HANA Web Workbench)
SAP HANA Administration Guide

4.4 SAP HANA Multitenant Database Containers

SAP HANA supports multiple isolated databases in a single SAP HANA system. These are referred to as multitenant database containers.

An SAP HANA system installed in multiple-container mode is capable of containing more than one multitenant database containers. Otherwise, it is a single-container system.

A multiple-container system always has exactly one system database and any number of multitenant database containers (including zero), also called tenant databases. An SAP HANA system installed in multiple-

container mode is identified by a single system ID (SID). Database containers 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 studio, 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 in 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 system with multitenant database containers is shown below. For more information, see *Multitenant Database Containers* in the *SAP HANA Administration Guide*.

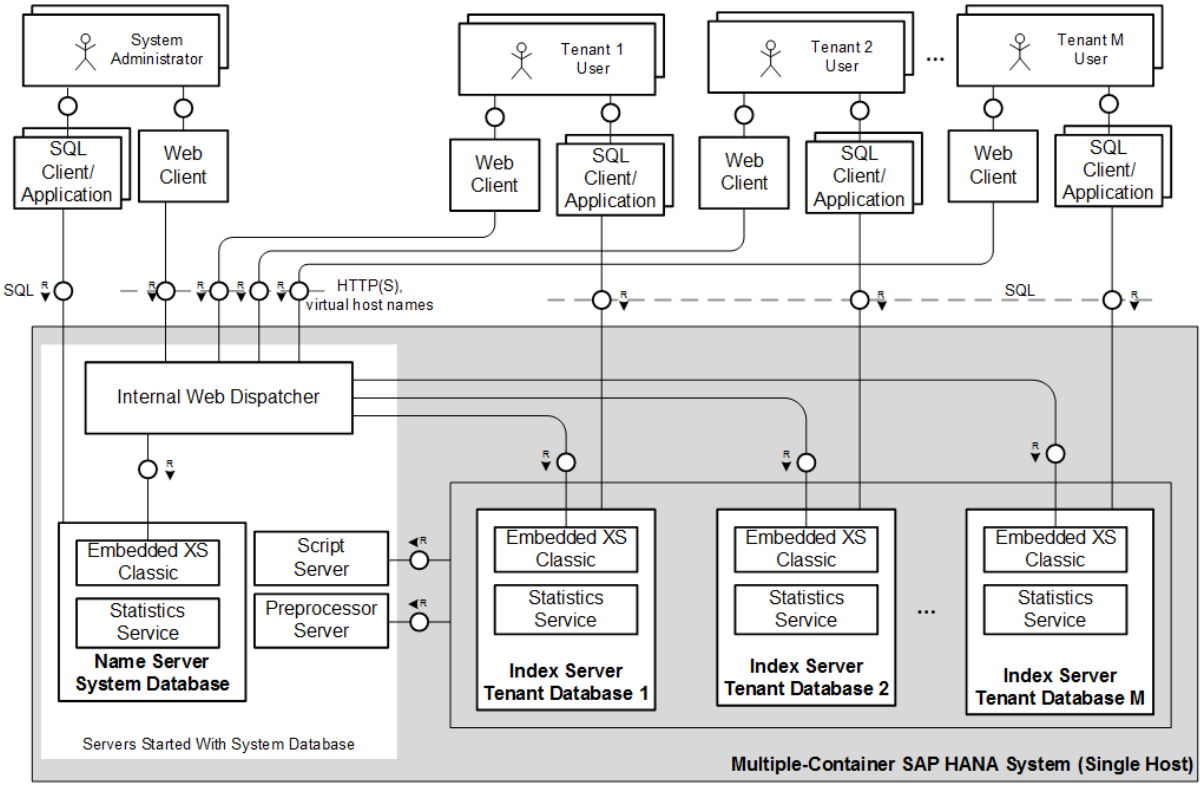


Figure 5: SAP HANA System with Multitenant Database Containers

Related Information

SAP HANA Administration Guide

4.4.1 Security for Multitenant Database Containers

In general, all security-related features of the SAP HANA database (such as authentication, authorization, encryption, and auditing) apply in systems that support multitenant database containers in the same way as in systems that do not. Some restrictions and additions apply.

The following table provides an overview of standard SAP HANA security-related features in the context of multitenant database containers:

Table 7:

Security-Related Feature	Multitenant Database Containers
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. For more information about database isolation, see the section *Database Isolation* below. For more information about the OS user `<sid>`adm, see *Predefined Users*.
Authentication and SSO	All user authentication mechanisms supported by SAP HANA are also supported in tenant databases. Whether a per-database configuration is possible depends on the authentication mechanism and the user client: • Basic authentication (with 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-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. • Database-specific trust stores cannot be configured for HTTP client access through SAP HANA Extended Services (SAP HANA XS). Therefore, user authentication based on SAML assertions and X.509 certificates cannot be database specific. For more information, see *SAP HANA Authentication and Single Sign-On*.
Authorization	SAP HANA's standard authorization mechanisms apply to users managed directly in the tenant database 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-Related Feature	Multitenant Database Containers
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. For more information, see *Certificate Management in SAP HANA* and *SSL Configuration on the SAP HANA Server*.
Encryption of data persistence layer	Data volume encryption can enabled for the system database and tenant databases individually. This ensures that anyone who can access the data volumes on disk using operating system commands cannot see the actual data. For more information, see *Data Volume Encryption in Multitenant Database Containers* in the *SAP HANA Administration Guide*.
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. 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. For more information, see *Auditing Activity in SAP HANA Systems*.

Additional Security Features for Multitenant Database Containers

Database isolation

To maximize the security of a multiple-container system 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 on the file system is protected using file and directory permissions.

Configuration change blacklist

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 blacklist (`multidb.ini`) is available for

this purpose. This blacklist contains several critical properties by default. You can customize the default configuration as well as add further properties by editing the file in the SAP HANA studio.

Restricted features

Depending on how you are implementing SAP HANA, you may want to disable certain database features that provide direct access to the file system, the network, or other resources, for example import and export operations and backup functions. For this reason, these features can be explicitly disabled in tenant databases.

Related Information

Predefined Users [page 66]
SAP HANA Authentication and Single Sign-On [page 74]
System Privileges (Reference) [page 97]
Cross-Database Authorization in Multitenant Database Containers [page 129]
Certificate Management in SAP HANA [page 168]
Data Volume Encryption in Multitenant Database Containers [page 144]
Auditing Activity in SAP HANA Systems [page 153]
Restricted Features in Multitenant Database Containers [page 241]
SAP HANA Administration Guide

4.4.2 Database Isolation

Every tenant database in a multiple-container system 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 in a multiple-container system 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 subsequently 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 is 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.

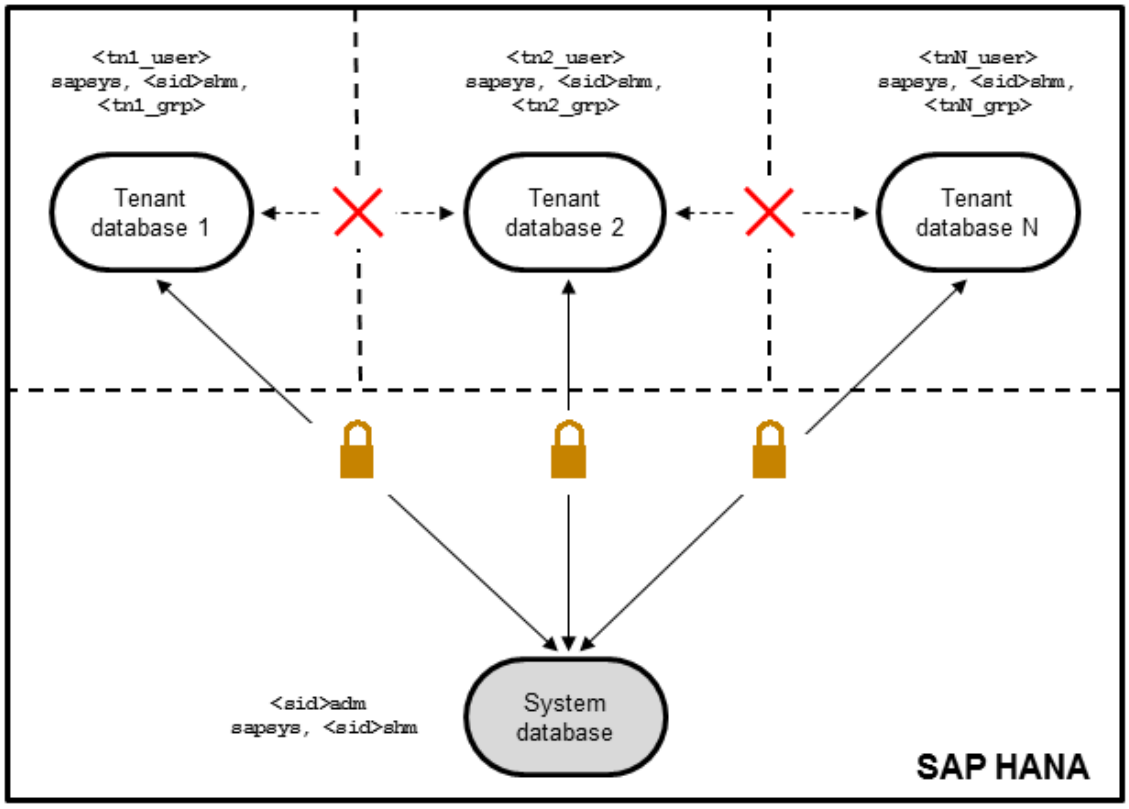


Figure 6: High Database Isolation

Configuration

You can specify the isolation level of the system during installation. The default isolation level is low. It is also possible to change the isolation level of an existing system (from low to high or from high to low) at any time. 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.

For more information, see:

- *Installing a Multitenant Database Container SAP HANA System* in the *SAP HANA Server Installation and Update Guide*
- *Increase the System Isolation Level* in the *SAP HANA Administration Guide*
- *Secure Internal Communication* in the *SAP HANA Security Guide*

Related Information

Secure Internal Communication [page 51]
SAP HANA Server Installation and Update Guide
SAP HANA Security Guide
SAP HANA Administration Guide

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.

Communication Channels [page 29]
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.

Network Security [page 33]
To integrate SAP HANA securely into your network environment, several general recommendations apply.

Securing Data Communication [page 36]
SAP HANA supports encrypted communication for client-server and internal communication.

Related Information

SAP HANA Master Guide

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

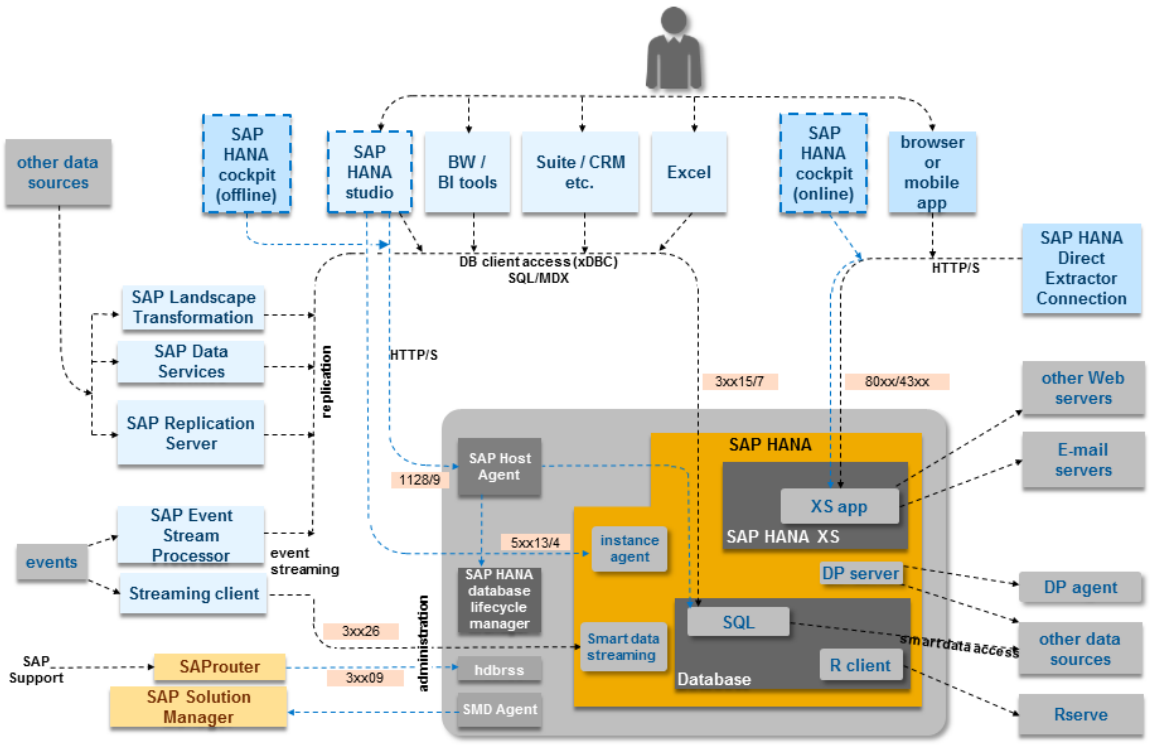
> **Note**
>
> SAP HANA internal communication has sometimes been unofficially referred to as TREXNet communication. However, the term TREXNet is not valid in the context of SAP HANA.

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 through the SAP HANA studio is depicted by the blue dotted arrows. Port numbers are shown with a pink 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 is depicted. However, the connections shown apply equally to a distributed scenario.



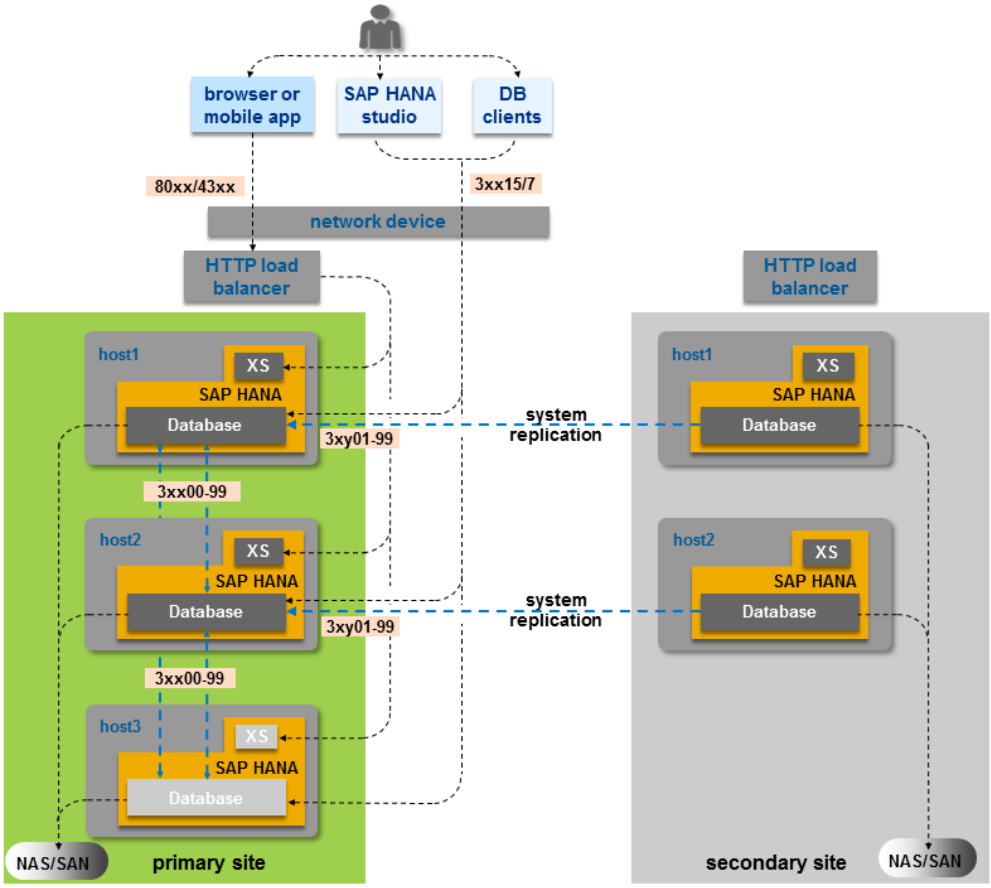
> **Note**
>
> Some components depicted in the figure are supported on Intel-based hardware platforms only (for example, SAP HANA smart data streaming). Refer to the Product Availability Matrix (PAM).

Figure 7: Connections Between SAP HANA and External Components

In addition, the different components of SAP HANA, as well as the hosts in a distributed scenario, communicate with each other over the 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 system-replicated to a secondary site to provide full disaster recovery support.

Figure 8: SAP HANA Internal Connections



Multitenant Database Containers

In systems with multitenant database containers, database clients connect to individual tenant databases through separate ports. Every database has its own internal communication port, SQL port and internal HTTP(s) port. These are assigned at the time of database creation.

The SAP Web Dispatcher, which runs as a separate database service on the system database, is used to route incoming HTTP requests from clients to the correct XS server based on virtual host names. For more information, see the *SAP HANA Administration Guide*.

The following figure shows an example of a single-host system with two tenant databases:

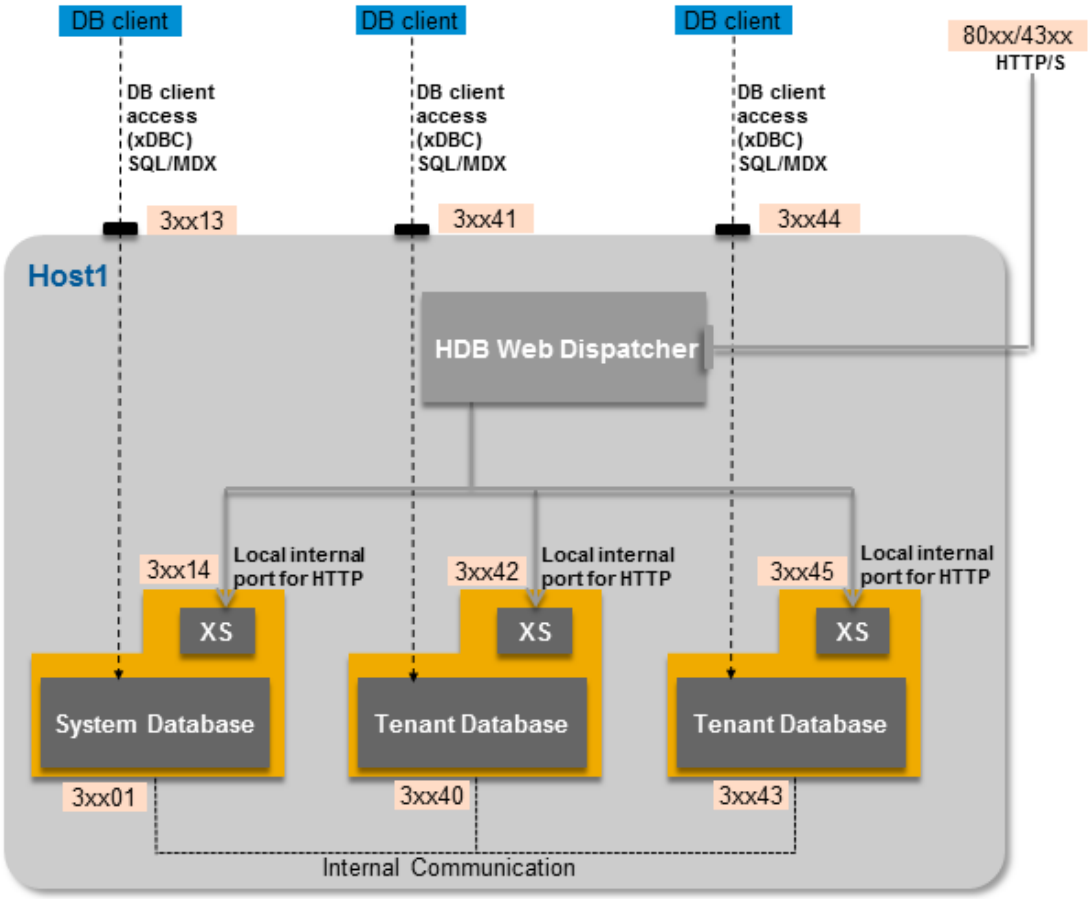


Figure 9: Connections for Multitenant Database Containers

For more information about the connections described and illustrated above, including the protocols used and the relevant ports numbers, see *The SAP HANA Network* in the *SAP HANA Master Guide*.

Related Information

Securing Data Communication [page 36]
SAP HANA Master Guide
SAP HANA Developer Guide (For SAP HANA Studio)
SAP HANA Developer Guide (For SAP HANA Web Workbench)
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 section The SAP HANA Network in the SAP HANA Master 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, see *Network Zones* in the *SAP HANA Master 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 (SAP HANA XS), 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.
  - SAP HANA XS 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

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. Prior to SPS 06, these ports were by default bound to all network interfaces.

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 ports 3`<instance>`01 to 3`<instance>`07. 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 subnetwork.

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, see *Configuring SAP HANA Inter-Service Communication* in the *SAP HANA Administration Guide*.

  > **Note**
  >
  > Prior to SPS 07, it was not possible to use this feature in the presence of a secondary site (system replication scenario). This feature can now be used 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 *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

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 and security aspects, see *Host Name Resolution for System Replication* in the *SAP HANA Master Guide* and *Configuring Hostname Resolution for SAP HANA System Replication* in the *SAP HANA Administration Guide*

Data Replication Technologies

Additional network configurations may be required depending on the implemented data replication technology. For more information, see *Security for SAP HANA Replication Technologies*.

Related Information

Secure Internal Communication [page 51]
Security for SAP HANA Replication Technologies [page 237]
SAP HANA Master Guide
SAP HANA Administration Guide

5.3 Securing Data Communication

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

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

External Communication Channels

- SAP HANA and clients via ODBC or JDBC connections, including the SAP HANA studio
- SAP HANA XS classic server and clients via HTTP
- SAP HANA lifecycle manager and the SAP HANA studio
- SAP HANA lifecycle manager and SAP Service Marketplace
- SAP HANA lifecycle manager and SAP Host Agent
- SAP HANA and the Rserve server
- SAP HANA and data providers
- SAP HANA information composer and Web browser

> **Note**
>
> SAP HANA information composer is supported on Intel-based hardware platforms only.

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

Internal Communication 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 database and server components, such as extended storage (SAP HANA dynamic tiering).

> **Caution**
>
> Be aware that you need additional licenses for SAP HANA options and capabilities such as SAP HANA dynamic tiering. For more information, see Important Disclaimer for Features in SAP HANA Platform, Options and Capabilities [page 270].

Certificate Collections (PSEs)

Separate certificate collections are supported for internal communication and external communication.

A certificate collection (also referred to as a personal security environment 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. By default, certificate collections for client-server communication over JDBC/ODBC are stored within the database. However for compatibility with previous releases, certificate collections (PSEs) can also be stored in the file system. We recommend creating the certificate collections in the database directly.

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

- Communication between database services
- Communication between hosts in a multiple-host system
- Communication between hosts and sites in a system replication scenario

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

Related Information

External Communication

Secure Communication Between SAP HANA and JDBC/ODBC Clients [page 38]
Secure Communication Between SAP HANA XS Classic and HTTP Clients [page 50]
SAP HANA Platform Lifecycle Management (Security) [page 232]
SAP HANA R Integration (Security) [page 234]
Security for SAP HANA Replication Technologies [page 237]
SAP HANA Information Composer (Security) [page 235]

Internal Communication

Secure Internal Communication [page 51]
Secure Internal Communication Between Sites in System Replication Scenarios [page 57]
Database Isolation [page 26]
SAP HANA Administration Guide

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.

Enabling TLS/SSL for client-server communication provides the following by default:

- Server certificate validation
  The server 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 server is encrypted, which provides integrity and privacy protection. An eavesdropper cannot access or manipulate the data.

It is also possible to enable client certificate validation, if the identity of the client connecting to SAP HANA should be validated.

TLS/SSL must be configured on both the server and the client.

> **Remember**
>
> Secure communication between the SAP HANA server and HTTP clients (HTTPS) must be configured separately. For more information, see *Configure HTTPS (SSL) for Client Application Access* in the *SAP HANA Administration Guide*.

Enforced TLS/SSL for Client Connections

If you want to force all clients communicating with the SAP HANA database via 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 don't use SSL.

> **Caution**
>
> Do not enforce TLS/SSL for client connections unless the monitoring and alerting functions in the system are being implemented by the **embedded** statistics service, not the statistics server. For more information, see SAP Note 2091313 and 1917938.

Related Information

SAP Note 2091313
SAP Note 1917938
SAP HANA Administration Guide

5.3.1.1 SSL Configuration on the SAP HANA Server

To use the Transport Layer Secure (TLS)/Secure Sockets Layer (SSL) protocol 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.

Before You Start

Before you can configure TLS/SSL on the SAP HANA server, the following general prerequisites must be met:

- The SAP Cryptographic Library CommonCryptoLib is available on the server.
  CommonCryptoLib (`libsapcrypto.so`) is installed by default as part of SAP HANA server 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. The OpenSSL library is installed by default as part of the operating system installation. However, it is recommended that you migrate to CommonCryptoLib after an upgrade to Support Package Stack (SPS) 09. For more information, see SAP Note 2093286.

- The SAP HANA server possesses a public and private key pair, and a public-key certificate.
  The TLS/SSL protocol uses public key technology to provide its protection. The server must possess a public and private key pair and a corresponding public-key certificate. It uses these to identify itself as the server component to a requesting client.
  In distributed SAP HANA systems, every host must have its own key pair and public key certificate. In systems that support multitenant database containers, every database can 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**
  >
  > Regardless of the tool you use, do not password protect the keystore file that contains the server's private key. For example, when using the SAP Web Dispatcher administration tool to create a personal security environment (PSE) for the server, do not specify a PIN.

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

Configuring TLS/SSL

The properties for configuring TLS/SSL on the server for external communication are available in the `communication` section of the `global.ini` configuration file, which you can edit in the Administration editor of the SAP HANA studio for example.

In general, it's not necessary to configure any of the properties explicitly. The default configuration can be used. However, you do need to create a certificate collection. You do this as follows:

1. Create a certificate collection in the database.
2. Add the server's public key certificate(s) and private key(s).
3. Add the public key certificates of trusted communication partners.
4. Assign the purpose *SSL* to the PSE.

> **Note**
>
> It is possible to use a certificate collection located on the file system and configured in the `global.ini` file. However, we recommend using a certificate collection that exists in the database.

For more information about the properties for TLS/SSL configuration, see *Server-Side TLS/SSL Configuration Properties for External Communication*. For more information about creating and configuring certificate collections in the SAP HANA database, see *Certificate Management in SAP HANA*.

TLS/SSL in Multitenant Database Containers

In systems that support multitenant database containers, every tenant database administrator must create a certificate collection with purpose SSL in their database as described above. All databases (system database and tenant databases) can have their own key pair and public key certificate.

If files located on 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 will 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, see *Default Blacklisted System Properties*.

Related Information

Server-Side TLS/SSL Configuration Properties for External Communication (JDBC/ODBC) [page 41]
Default Blacklisted System Properties in Multitenant Database Containers [page 243]
Certificate Management in SAP HANA [page 168]
Related SAP Notes
SAP Note 2093286 - Migration from OpenSSL to CommonCryptoLib (SAPCrypto)
SAP Note 1718944 - SAP HANA DB: Securing External SQL Communication (SAPCrypto)
SAP Note 1848999 - Central Note for CommonCryptoLib 8 (replacing SAPCRYPTOLIB)
SAP Note 2009483 - PSE Management in Web Administration Interface of SAP Web Dispatcher
SAP Note 2009878 - Purpose of the PSE Files in PSE Management of SAP Web Dispatcher

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's not necessary to configure any of the parameters explicitly. The default configuration can be used.

Table 8:

Parameter	Value	Default	Description
`sslMinProtocolVersion`	{SSL30,TLS10}	TLS10	The minimum available TLS/SSL protocol version
`sslMaxProtocolVersion`	{TLS10,TLS11,TLS12,MAX}	MAX	The maximum available TLS/SSL protocol version
`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.

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 **`HIGH:MEDIUM:!aNULL`** (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.

In systems that support multitenant database containers, the system administration can configure different trust and key stores for individual databases.

Remember that the trust store configured on the file system is also valid for single sign-on mechanisms that rely on trust stores (such as SAP logon and assertion tickets or SAML).

Recommendation

Create certificate collections in the database instead of using trust and key stores in the file system. That way you can create different certificate collections for different purposes.

Note

If certificate collections with a purpose (user authentication or secure client-server communication) exist in the database, the parameters below are ignored.

Table 9:

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.

Parameter	Value	Default	Description
`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.
`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.

5.3.1.3 TLS/SSL Configuration on the Client

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. TLS/SSL must be configured on both the server and the client.

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

For ODBC-based connections, the configuration properties and their names are the same as the server parameters. In addition, the `encrypt` property is available to initiate an TLS/SSL-secured connection. You set the properties according to the client operating system.

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.

For connections from the SAP HANA studio, which uses the JDBC interface, you configure the TLS/SSL properties directly in the system's properties in the SAP HANA studio (for example, while adding it in the studio). For more information, see *Configure TLS/SSL for SAP HANA Studio Connections*.

> **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.

Related Information

Configure SSL for SAP HANA Studio Connections [page 48]

5.3.1.4 Client-Side TLS/SSL Configuration 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 parameters that are used to configure SSL for ODBC client access:

Table 10:

Parameters	Value	Default	Description
`encrypt`	<bool value>	False	Enables or disables TLS/SSL encryption
`sslCryptoProvider`	{commoncrypto| sapcrypto | openssl}	1. commoncrypto or sapcrypto (if installed) 2. openssl	Cryptographic library provider used for SSL communication **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.

Parameters	Value	Default	Description
`sslKeyStore`	<file>	`$SECUDIR/sapsrv.pse` (CommonCryptoLib/SAP Cryptographic Library) or `$HOME/.ssl/key.pem` (OpenSSL)	Path to the keystore file that contains the server's private key **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.
`sslTrustStore`	<file>	$HOME/.ssl/trust.pem	Path to trust store file that contains the server's public certificate(s) (OpenSSL only) Typically, the trust store contains the root certificate or the certificate of the certification authority that signed the server's certificate(s). **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. **Note** If you are using the cryptographic library mscrypto, leave this parameter empty.
`sslValidateCertificate`	<bool value>	true	If set to true, the server's certificate is validated

Parameters	Value	Default	Description
`sslHostNameInCertificate`	<string value>	<empty>	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, the server's host name is not validated. Other wildcards are not permitted.

5.3.1.5 Client-Side TLS/SSL Configuration 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.

Table 11:

Property	Value	Default	Description
`encrypt`	<bool value>	false	Enables or disables TLS/SSL encryption
`validateCertificate`	<bool value>	true	If set to true, the server's certificate is validated.

Property	Value	Default	Description
`hostNameInCertificate`	<string value>	<empty>	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 parameter has no effect. Other wildcards are not permitted.
`keyStore`	<file | store name>	<VM default>	Location of the Java keystore
`keyStoreType`	<JKS | PKCS12>	<VM default>	Java keystore file format
`keyStorePassword`	<password>	<VM default>	Password to access the private key from the keystore file **Note** This property is not used for SAP HANA studio connections.
`trustStore`	<file | store name>	<VM default>	Path to 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).
`trustStoreType`	<JKS>	<VM default>	File format of trust store file
`trustStorePassword`	<password>	<VM default>	Password used to access the trust store file

5.3.1.6 Configure SSL for SAP HANA Studio Connections

Secure communication between the SAP HANA studio and the SAP HANA database using the Transport Security Layer (TLS)/Secure Sockets Layer (SSL) protocol.

Prerequisites

- You have configured the SAP HANA database for secure client-server communication over JDBC/ODBC. For more information, see *SSL Configuration on the SAP HANA Server* in the *SAP HANA Security Guide*.
- You have added the SAP HANA system in the SAP HANA studio.

Context

The SAP HANA studio communicates with the SAP HANA database via the JDBC client interface. The client-side configuration of the SAP HANA studio uses Java TLS/SSL properties.

Procedure

1. Using the keytool command line tool, import the truststore file that contains the server root certificate into either the Java keystore or your personal user keystore.

   By default, the SAP HANA studio client validates server certificate(s) against the root certificate stored in the Java keystore of the running VM (virtual machine). This keystore is part of the Java installation and is located in the Java home directory under `${JAVA_HOME}/lib/security/cacerts` (Linux) or `%JAVA_HOME%/lib/security/cacerts` (Windows).

   However, it is not recommended that you store the root certificate in this keystore, but in your personal user keystore instead. The user keystore is located in the home directory of the current operating system user. The file name is `.keystore`.
2. Enable and configure TLS/SSL secure communication between the SAP HANA studio and the server:

   In the SAP HANA studio, open the system's properties and choose *Connect Using SSL*.

   This corresponds to setting the Java SSL property `encrypt` to **`true`**.
3. Configure how the identity of the server is to be validated during connection (server-side authentication):
   a. In the system's properties dialog, choose the *Additional Properties* tab.
   b. If you want server certificate(s) to be validated using the default truststore, choose *Validate SSL Certificate*.

      This corresponds to setting the Java SSL property `validateCertificate` to **`true`**.

      When an TLS/SSL connection is established, the host name in the certificate being connected to and the host name in the server certificate must match. This may not always be the case. For example, in a single-host system, if a connection is established from the SAP HANA studio on the same host as the

SAP HANA server, a mismatch would arise between the host named in the certificate (fully qualified host name) and the host used to establish the connection (localhost)*.

You can override the host name specified in the server certificate by entering a host name with a defined certificate in the *Override Host Name Certificate* field. This corresponds to setting the Java SSL property `hostNameInCertificate`.

c. If you want the server certificate to be validated using the user's keystore and not the default Java keystore, choose *Use user keystore as trust store*.

This corresponds to changing the value of the Java SSL property `trustStore`.

> **Note**
>
> If you do not have a working public key infrastructure (PKI), you can also suppress server certificate validation entirely by selecting neither of these options (*Validate SSL Certificate* or *Use user keystore as trust store*). However, this is not recommended.

4. Optional: If the identity of the client is to be validated by the SAP HANA server (client certificate validation), perform the following additional steps:
   a. In the *Additional Properties* tab of the system properties, specify the path to the user keystore that contains your private key, as well as the pass phrase required to access this file.
   b. Enable validation of the client's identity on the server by changing the parameter `[communication] sslValidateCertificate` in the `global.ini` file to **`true`**.

   You can do this on the *Configuration* tab of the Administration editor.
   c. Import the client root certificate into the server truststore used for client-server communication.

   If you manage client certificates directly in the database (recommended), this means importing the certificate into the certificate store and adding it to the certificate collection with the purpose *SSL*.

Results

In the *Systems* view, a lock icon appears next to the system name (), indicating that SSL communication is active.

Related Information

SSL Configuration on the SAP HANA Server [page 39]

5.3.2 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.

For more information, see *Configure HTTPS (SSL) for Client Application Access* in the *SAP HANA Administration Guide*.

> **Note**
>
> For more information about network and communication security in the context of the SAP HANA XS advanced application server infrastructure, see *Network and Communication Security with SAP HANA XS Advanced*.

Multitenant Database Containers

For communication with HTTP clients, a per-database configuration of TLS/SSL keys and certificates is also 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 *Configure HTTP(S) Access to Multitenant Database Containers in the SAP HANA Administration Guide*.

Related Information

Network and Communication Security with SAP HANA XS Advanced [page 205]
SAP HANA Administration Guide

5.3.2.1 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 response time in milliseconds, and the size in bytes. 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 studio.

Related Information

icm/HTTP/logging_<xx>
Logging in the ICM and SAP Web Dispatcher
SAP HANA Administration Guide

5.3.3 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.

The following internal communication channels can be secured using TLS/SSL:

- Communication between the processes of individual databases in a multiple-container system
- Communication between the hosts in a multiple-host system and between processes on a host
- Communication between the sites in a system with system replication enabled
- Communication between the SAP HANA database and additional server components, such as an extended storage server (SAP HANA dynamic tiering) or a smart data streaming server (SAP HANA smart data streaming).

> ⚠ Caution
>
> Be aware that you need additional licenses for SAP HANA options and capabilities such as SAP HANA dynamic tiering. For more information, see Important Disclaimer for Features in SAP HANA Platform, Options and Capabilities [page 270].

> ℹ Note
>
> SAP HANA smart data streaming is supported on Intel-based hardware platforms only.

ℹ 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.

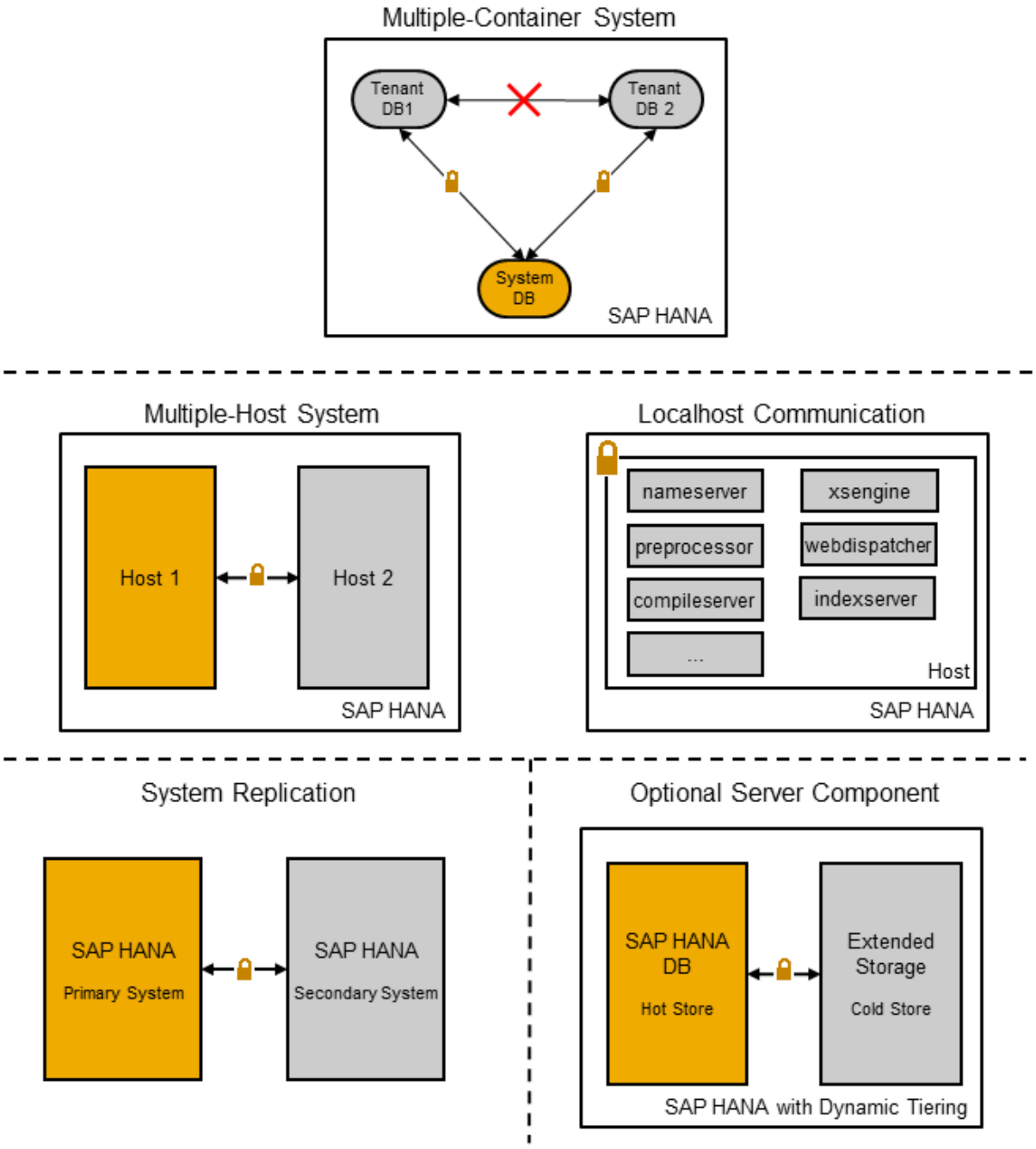


Figure 10: Secured Internal Communication Channels

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 that is automatically created during system installation. The system PKI is always used to secure communication within tenant databases in a multiple-container system 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

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.

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, see *Change the SSFS Master Keys* in the *SAP HANA Administration Guide*.

The system PKI uses CommonCryptoLib as the cryptographic library.

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:

Table 12:

Communication Channel	Configuration Required to Enable TLS/SSL
Communication between the processes of individual databases in a multiple-container system	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 **`false`** 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 *Increase the System Isolation Level* in 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 multiple-container system that is not configured for high isolation, you can still enable secure communication between hosts. Remember you 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 **`true`** to **`false`**.
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

Database Isolation [page 26]
Server-Side TLS/SSL Configuration Properties for Internal Communication [page 58]
Secure Internal Communication Between Sites in System Replication Scenarios [page 57]
Legacy Configuration of Secure Internal Communication [page 60]
SAP HANA Options and Capabilities
SAP HANA Administration Guide
SAP Note 2175672 - Migration steps from manual SSL configuration for internal communication to automatic configuration using system PKI

5.3.3.1 Secure Internal Communication Between Sites in System Replication Scenarios

The Transport Layer Security (TLS)/Secure Sockets Layer (SSL) protocol can be used to secure internal network communication between primary and secondary sites in system replication scenarios.

System replication is a mechanism for ensuring the high availability of SAP HANA systems, as well as disaster recovery. Through the continuous replication of data from a primary to a secondary system (or systems), including in-memory loading, system replication facilitates rapid failover in the event of a disaster. Production operations can be resumed with minimal downtime. The following communication channels can be secured 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

The system PKI (public key infrastructure) that is automatically created during system installation is the default and recommended mechanism for securing the above communication channels. However, you can also set up and configure your own PKI (see *Legacy Configuration of Secure Internal Communication*).

In addition to enabling TLS/SSL, you can secure communication between sites further by configuring the IP addresses of those hosts that are allowed to connect to the ports required for system replication.

TLS/SSL Configuration Using System PKI

No interaction is required to set up the system PKI, but you need to perform the following steps to enable SSL on the communication channels used during system replication.

> **Note**
>
> Perform these steps after you have configured system replication.

1. Shut down all systems.
2. On all systems, enable SSL for communication between hosts.
   In the `global.ini` configuration file, change the value of the property `[communication] ssl` from **`false`** to **`systemPKI`**.
3. Copy the system PKI SSFS data file and key file from the primary system to the same location on the secondary system(s):
   - `$DIR_INSTANCE/../global/security/rsecssfs/data/SSFS_<SID>.DAT`
   - `$DIR_INSTANCE/../global/security/rsecssfs/key/SSFS_<SID>.KEY`
4. In the primary and secondary system, enable SSL for the data channel.
   In the `global.ini` file, configure the property `[system_replication_communication] enable_ssl`. The following values are possible:

   Table 13:

Value	Description
off (default)	TLS/SSL is disabled for replication source and target systems

Value	Description
on	TLS/SSL is enabled for replication source and target systems
source	TLS/SSL is enabled for replication source system only
target	TLS/SSL is enabled for replication target system only

For a simple system replication scenario involving two systems, it is sufficient to set the property to **on** in both systems. For multitier system replication scenarios involving three systems, you can specify the values **source** and **target** in the tier 2 secondary system. This enables TLS/SSL between this system and its source system or target system – either only the communication to the primary system is secured or only the communication to the tier 3 secondary system is secured.

> **Note**
>
> To avoid communication failure between systems, TLS/SSL must be enabled on all systems at the same time. TLS/SSL won't be used unless the secondary system reconnects with the primary. This can be done by restarting the primary and secondary systems.

5. As `<sid>`adm, restart the `sapstartsrv` service on the secondary system(s):
   1. `sapcontrol -nr <instance_no> -function StopService`
   2. `/usr/sap/<sid>/HDB<instance_no>/exe/sapstartsrv pf=/usr/sap/<sid>/SYS/profile/<sid>_HDB<instance_no>_<host> -D -u <sid>adm`
6. Restart all systems.

Related Information

Secure Internal Communication [page 51]
Server-Side TLS/SSL Configuration Properties for Internal Communication [page 58]
Legacy Configuration of Secure Internal Communication [page 60]

5.3.3.2 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.

Table 14:

Property	Value	Default	Description
`ssl`	{false | systemPKI | true}	false	Enables TLS/SSL on internal communication channels The following values are possible: • **`false`** (default) With this value, TLS/SSL is disabled. In multiple-container systems configured for high isolation, but with the default value **`false`**, 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). In multiple-container systems, 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. • **`true`** 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 multiple-container system configured for high isolation, do not set the value is property **`true`**. 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>`	true	Enables TLS/SSL for communication between localhost processes This parameter is only evaluated if `ssl` has the value **`systemPKI`** or **`true`**.

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

Related Information

Secure Internal Communication [page 51]
Legacy Configuration of Secure Internal Communication [page 60]

5.3.3.3 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.

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 Certification Authority (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 **`true`**.

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, you also need to enable SSL with the 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 57]

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 *SAP HANA Authorization*.

User Types [page 63]
: 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.

User Administration Tools [page 64]
: 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.

Predefined Users [page 66]
: A number of predefined users are required for installing, upgrading, and operating SAP HANA.

Deactivate the SYSTEM User [page 71]
: 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.

SYSTEM User in Multitenant Database Containers [page 73]
: Every database in a multiple-container system has its own set of database users, including the database superuser SYSTEM. When and how the SYSTEM user password is specified depends on whether the system was installed in multiple-container mode or converted to multiple-container mode.

Related Information

SAP HANA Authentication and Single Sign-On [page 74]
SAP HANA Authorization [page 92]

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

For every person who needs to work with SAP HANA, the user administrator creates a database user.

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.

Database users are created with either the `CREATE USER` or `CREATE RESTRICTED USER` statement.

Standard Users

Standard users are 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, 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 (and cannot be 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 in the *Restricted User* editor of the SAP HANA studio.
  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.

Creating a database user as a restricted user is an irreversible action.

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

Predefined Database Roles [page 114]
SAP HANA SQL and System Views Reference

6.2 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.

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

1. Define and create roles.
2. Create users.
3. Grant roles to users.

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:

Table 15:

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 roles in the repository as they offer more flexibility (for example, they can be transported between systems). For more information, see *Catalog Roles and Repository Roles Compared*.	• *Developer Workbench* of the SAP HANA studio • *Editor* tool of the SAP HANA Web-based Development Workbench
Application developer	Create roles for new applications developed on SAP HANA Extended Services (SAP HANA XS), classic model	Design time	
User or system administrator	Create SAP HANA database users	Runtime	• *User* editor of the SAP HANA studio • *Security* tool of the SAP HANA Web-based Development Workbench • SAP HANA HDBSQL HDBSQL is useful when using scripts for automated processing. For more information about HDBSQL, see the *SAP HANA Administration Guide*.
	Grant roles to database users		
	Delete, deactivate, and reactivate database users		
	Reset user passwords		
User or system administrator	Grant roles to database users	Runtime	*Assign Roles* app of the SAP HANA cockpit

For more detailed information about roles in SAP HANA, see *Roles*.

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 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 database user SYSTEM and operating system administrator `<sid>`adm as part of system rename. For more information, see *Changing System Identifiers* in the *SAP HANA Administration Guide*.

Related Information

Roles [page 112]
Catalog Roles and Repository Roles Compared [page 118]
SAP HANA Developer Guide (For SAP HANA Studio)
SAP HANA Developer Guide (For SAP HANA Web Workbench)
SAP Note 1986645 (Allow Only Administration Users to Work on HANA Database)
SAP HANA Administration Guide
SAP NetWeaver Identity Management

6.3 Predefined Users

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

The following table lists the standard users that are available.

> **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 *Predefined XSA Users*.

Table 16:

User	Description	Password Specification
SYSTEM	The SYSTEM database user is created during the installation of the SAP HANA system. 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. In addition, to ensure that the administration tool SAP HANA cockpit can be used immediately after database creation, SYSTEM is automatically granted several roles the first time the cockpit is opened with this user. For more information, see *Repository Roles Granted to Standard Database Users*. 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. **i Note** For more information about the SYSTEM user in tenant databases in multiple-container systems, see *SYSTEM User in Multitenant Database Containers*.	The initial password of the SYSTEM user is specified by your hardware partner or certified administrator during installation. After handover, it is important that you change this password. A user administrator (that is, a user with the system privilege USER ADMIN) can do this in the SAP HANA studio. It is also possible as part of a system rename with SAP HANA lifecycle manager.

User	Description	Password Specification
<sid>adm where <sid> is the ID of the SAP HANA system	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 multiple-container system configured for high isolation, additional OS users will exist for every tenant database. Access to database-specific data is limited accordingly. For more information, see *File and Directory Permissions with High Isolation* in the *SAP HANA Administration Guide*.	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.
SYS	SYS is a technical database user. It is the owner of database objects such as system tables and monitoring views.	Not applicable This is a technical database user. It is not possible to log on with this user.

User	Description	Password Specification
XSSQLCC_AUTO_USER_<generated_ID>	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. For more information about this role, see *HANA_XS_BASE* in the reference section of the *SAP HANA Security Guide*. 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. For more information about this role, see *HANA_XS_BASE* in the reference section of the *SAP HANA Security Guide*. This user cannot be used to log on to SAP HANA.	Password-based logon is disabled by default for an automatically generated SQLCC user. Therefore, a password is not required.

User	Description	Password Specification
	Note Since the above users don't have human readable names, check the assigned roles to see which user is which. For more information about SQLCCs and the above applications, see *Maintaining the SAP HANA XS Classic Model Run Time* section of the SAP HANA Administration Guide	
_SYS_AFL	_SYS_AFL is a technical user that owns all objects for Application Function Libraries.	Not applicable This is a technical database user. It is not possible to log on with this user.
_SYS_EPM	_SYS_EPM is a technical database used by the SAP Performance Management (SAP EPM) application	Not applicable This is a technical database user. It is not possible to log on with this user.
_SYS_REPO	_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.	Not applicable This is a technical database user. It is not possible to log on with this user.
_SYS_STATISTICS	_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.	Not applicable This is a technical database user. It is not possible to log on with this user. **Note** _SYS_STATISTICS still logs on internally if you have not migrated to the new implementation of the statistics server available as of SPS 07.
_SYS_TASK	_SYS_TASK is a technical database user in SAP HANA smart data integration. This user owns all task framework objects. For more information, see SAP HANA Smart Data Integration and SAP HANA Smart Data Quality on SAP Help Portal.	Not applicable This is a technical database user. It is not possible to log on with this user.

User	Description	Password Specification
_SYS_WORKLOAD_REPLAY	_SYS_WORKLOAD_REPLAY is a technical database user used by capture and replay capability of the SAP HANA Performance Management tool. 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. For more information about SAP HANA Workload Capture and Replay, see the SAP HANA Administration Guide.	Not applicable This is a technical database user. It is not possible to log on with this user.
_SYS_XB	_SYS_XB is a technical user for internal use only.	Not applicable This is a technical database user. It is not possible to log on with this user.

Related Information

Predefined XSA Users [page 186]
Deactivate the SYSTEM User [page 71]
Repository Roles Granted to Standard Database Users [page 123]
SYSTEM User in Multitenant Database Containers [page 73]
HANA_XS_BASE [page 254]
SAP HANA Administration Guide
SAP HANA Enterprise Information Management

6.4 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

`SYSTEM` is the database superuser. It has irrevocable system privileges, such as the ability to create other database users, access system tables, and so on. In addition, to ensure that the administration tool SAP HANA cockpit can be used immediately after database creation, `SYSTEM` is automatically granted several roles the first time the cockpit is opened with this user. For more information, see *Roles Granted to Database User SYSTEM*. Note however that `SYSTEM` does not automatically have access to objects created in the SAP HANA repository.

In a system with multitenant database containers, the `SYSTEM` user of the system database has additional privileges for managing tenant databases, for example, creating and dropping databases, changing configuration (*.ini) files of databases, and performing database-specific data backups.

It is highly recommended that you do not use `SYSTEM` for day-to-day activities in production systems. 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`.

Procedure

Execute the following statement, for example, in the SQL console of the SAP HANA studio:

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

Results

The SYSTEM user is deactivated and can no longer connect to the SAP HANA database.

You can verify that this is the case 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.

Related Information

SAP HANA SQL and System Views Reference

6.5 SYSTEM User in Multitenant Database Containers

Every database in a multiple-container system has its own set of database users, including the database superuser SYSTEM. When and how the SYSTEM user password is specified depends on whether the system was installed in multiple-container mode or converted to multiple-container mode.

A database-specific SYSTEM user exists in every database of a multiple-container system. In general, this user has the same status and privileges as the SYSTEM user in a single-container system. Only 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, and performing database-specific data backups.

Password Specification

If you install your SAP HANA system in multiple-container mode, the SYSTEM user of the system database is created during installation. For every tenant database created in the system, the SYSTEM user is created and its password specified when the database is created.

If you convert an existing SAP HANA system to multiple-container mode, the system database and **one tenant database** are created during the conversion process. This tenant database contains all the data of the original system, including users, system configuration, and connection properties. The password of the SYSTEM user in this tenant database is the password of the SYSTEM user of the original system before it was converted. You must explicitly set the password of the SYSTEM user of the system database during conversion. For more information about how to do this, see *Convert an SAP HANA System to Support Multitenant Database Containers* in the *SAP HANA Administration Guide*.

For every subsequent tenant database created in the system, the SYSTEM user is created and its password specified when the database is created.

Related Information

SAP HANA Administration Guide

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, several of which can be used for the integration of SAP HANA into single sign-on environments (SSO). The mechanisms used to authenticate individual users is specified as part of the user definition.

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

User Authentication Mechanisms [page 75]
: Authentication mechanisms supported in SAP HANA. Mechanisms that are not required can be disabled.

SAP HANA Logon Checks [page 77]
: Before a user can connect to the SAP HANA database , the system performs several checks as part of the logon process.

Password Policy [page 77]
: Passwords for the basic authentication of database users are subject to certain rules. These are defined in the password policy. You can change the default password policy in line with your organization's security requirements.

Single-Sign On Integration [page 86]
: Integrate SAP HANA into single-sign environments using Kerberos, SAML 2.0, and logon and assertion tickets.

7.1 User Authentication Mechanisms

Authentication mechanisms supported in SAP HANA. Mechanisms that are not required can be disabled.

Supported Authentication Mechanisms

Table 17:

Mechanism	Description	Can Be Used for SSO
Basic authentication (user name and password)	Users accessing the SAP HANA database authenticate themselves by entering their database user name and password. For more information, see *Password Policy* and *Password Blacklist*.	No
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), 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. SAP HANA maps the external identity 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 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. SAP HANA maps the external identity to the identity of an internal database user.	Yes

Mechanism	Description	Can Be Used for SSO
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
X.509 client certificates	For HTTP/HTTPS access to SAP HANA by means of SAP HANA XS classic, users can be authenticated by client certificates signed by a trusted Certification Authority (CA), which can be stored in the SAP HANA XS trust store. **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 for HTTP/HTTPS access to SAP HANA by means of SAP HANA XS classic
Session cookies	Session cookies are not technically an authentication mechanism. However, they reconnect users who have already been authenticated by Kerberos or SAML and extend the validity period of logon and assertion tickets.	Yes

Disabling Authentication Mechanisms

By default all authentication mechanisms are enabled, but it is possible and recommended to disable those that are not used in your environment. 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 **`password,kerberos,spnego,saml,saplogon,x509xs,sessioncookie`**.

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 this parameter are audited by default if auditing is enabled.

7.1.1 User Authentication in Multitenant Database Containers

All user authentication mechanisms are supported in multitenant database containers.

Separate, database-specific authentication is possible for every certificate-based authentication mechanism (SAML assertions, X.509 certificates, and logon tickets) since it is possible to create different certificate collections for individual purposes directly in every database. However, 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.

> ⚠ 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 will be able to log on to other databases in the system.

7.2 SAP HANA Logon Checks

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

1. The system authenticates the user using the configured mechanism.
   For example, if user name/password authentication is being enforced, the provided user name and password are verified.
2. The system 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.
3. The system verifies that the user's 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 SAP HANA.

7.3 Password Policy

Passwords for the basic authentication of database users are subject to certain rules. These are defined in the password policy. You can change the default password policy in line with your organization's security requirements.

The password policy is defined by parameters in the `password policy` section of the `indexserver.ini` system properties file. Although you can configure your password policy directly in the `indexserver.ini` file,

it is recommended that you use either the *Password Policy and Blacklist* app of the SAP HANA cockpit or the *Security* editor of the SAP HANA studio.

> **Caution**
>
> Direct changes to the `indexserver.ini` file cannot be audited.

The system view M_PASSWORD_POLICY contains the parameters and their current values.

Multitenant Database Containers

It is possible to configure the password policy individually for the system database and each tenant database in a multiple-container system. Note that the password policy parameters for the system database are maintained in the `namesever.ini` file, not the `indexserver.ini` file.

Related Information

Auditing Activity in SAP HANA Systems [page 153]
SAP HANA Administration Guide
SAP HANA SQL and System Views Reference

7.3.1 Password Policy Configuration Options

The *Password Policy and Blacklist* app in the SAP HANA cockpit and the *Security* editor in the SAP HANA studio allow you to view the password policy and to change its default configuration.

The password policy is defined by parameters in the `password policy` section of the `indexserver.ini` configuration file. The following sections describe these parameters, which correspond to the configuration options available in the *Password Policy and Blacklist* app and the *Security* editor.

> **Note**
>
> The password policy parameters for the system database of a multiple-container system are maintained in the `namesever.ini` file, not the `indexserver.ini` file.

- Minimum Password Length [page 79]
- Lowercase Letter/Uppercase Letter/Numerical Digit/Special Character Required [page 79]
- Password Change Required on First Logon [page 80]
- Number of Last Used Passwords That Cannot Be Reused [page 81]
- Number of Allowed Failed Logon Attempts [page 81]
- User Lock Time [page 82]
- Minimum Password Lifetime [page 83]

- Maximum Password Lifetime [page 83]
- Lifetime of Initial Password [page 84]
- Maximum Duration of User Inactivity [page 84]
- Notification of Password Expiration [page 84]
- SYSTEM User Lock [page 85]
- Detailed Error Information on Failed Logon [page 85]

Minimum Password Length

The minimum number of characters that the password must contain

Table 18:

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 Letter/Uppercase Letter/Numerical Digit/Special Character Required

The character types that the password must contain; at least one character of each selected character type is required

Table 19:

Parameter	`password_layout`
Default Value	Aa1

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 default configuration requires passwords to contain at least one uppercase letter, at least one number, and at least one lowercase letter, with special characters being optional. **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, for example, supports passwords enclosed in double quotes ("), while the SAP HANA HDBSQL command line tool does not. **Note** 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`**. If you want to enforce the use of at least one of each character type including special characters, you specify **`A1a_`** or **`2Bg?`**.
UI Labels	*Lowercase Letter/Uppercase Letter/Numerical Digit/Special Character Required*

Password Change Required on First Logon

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

Table 20:

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]`
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

Table 21:

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 possible; the user is locked as soon as this number is reached

Table 22:

Parameter	`maximum_invalid_connect_attempts`
Default Value	6 (failed logon attempts)

Additional Information	You must enter a value of at least **1**. 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** Although this parameter is not valid for the SYSTEM user, the SYSTEM user will still be locked if the parameter `password_lock_for_system_user` is set to **`true`**. 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

Table 23:

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`. A user administrator can reset the number of invalid logon attempts and reactivate 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**. In the *Security* editor of the SAP HANA Studio or the *Authentication* app 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

Table 24:

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

Table 25:

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 only 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

Table 26:

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.
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

Table 27:

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

Table 28:

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. For more information about how to disable this check, see SAP Note 1991615.
UI Label	*Notification of Password Expiration*

SYSTEM User Not Locked

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`)

Table 29:

Parameter	`password_lock_for_system_user`
Default Value	true
UI Label	*SYSTEM User Not Locked*

Detailed Error Information on Failed Logon

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

Table 30:

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

SAP Note 1991615

7.3.2 Password Blacklist

A password blacklist is a list of words that are not allowed as passwords or parts of passwords.

The password blacklist 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 blacklist as part of password policy configuration using the *Password Policy and Blacklist* app of the SAP HANA cockpit or the *Security* editor of the SAP HANA studio.

In a system with multitenant database containers, the password blacklist can be managed for each database individually.

Related Information

SAP HANA Administration Guide

7.4 Single-Sign On Integration

Integrate SAP HANA into single-sign environments using Kerberos, SAML 2.0, and logon and assertion tickets.

Single Sign-On Using Kerberos [page 87]
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) classic, Kerberos authentication is enabled with Simple and Protected GSSAPI Negotiation Mechanism (SPNEGO).

Single Sign-On Using SAML 2.0 [page 88]
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 90]
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).

7.4.1 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) 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 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).

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

SAP Note 1837331 - How-To: HANA DB SSO Kerberos/ Active Directory
SAP HANA Administration Guide

7.4.2 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 secure sockets layer (SSL) 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, and MD5 for hash algorithms
- RSA-SHA1 and RSA-SHA256 as signature algorithms

The following SAML assertion features are supported:

- Assertion Subject with `NameID`
- Qualified `NameID` with `SPProvidedID` and `SPNameQualifier`
- Validity conditions (`NotBefore`, `NotOnOrAfter`)
- Audience restrictions

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

Table 31:

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 have been set 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.

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
  The mapping to a database user is explicitly configured within SAP HANA 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>`
  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. The corresponding assertion subject looks like this:
  `<NameIDFormat="urn:oasis:names:tc:SAML:1.1:nameid- format:unspecified" SPProvidedID="BILLG">zgc2VLavgYy4hsohfYPM21</NameID>`
  You can configure SAML identity providers and map them to database users in the SAP HANA studio.

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. It is also possible to use a trust store located on the file system. This is the same trust store configured for communication between the SAP HANA database and clients that access the SQL interface of the database. For more information, see *Server-Side SSL Configuration Properties for External Communication (JDBC/ODBC).*

Configuration for HTTP Client Access

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, see *Maintaining SAML Providers* in the *SAP HANA Administration Guide* .

Related Information

Server-Side TLS/SSL Configuration Properties for External Communication (JDBC/ODBC) [page 41]
SAP HANA Administration Guide
SAP HANA SQL and System Views Reference

7.4.3 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** with 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)

You can also 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**
>
> In systems that support multitenant database containers, this parameter is in the default configuration change blacklist (`multidb.ini`). This means that it can only be changed by the system administrator in the system database. It cannot be changed in tenant databases. For more information, see *Default Blacklisted System Properties*.

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 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

Default Blacklisted System Properties in Multitenant Database Containers [page 243]
SAP HANA Administration Guide
Using Logon Tickets

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.

Privileges [page 93]
Several privilege types are used in SAP HANA (system, object, analytic, package, and application).

Roles [page 112]
A role is a collection of privileges that can be granted to either a database user or another role in runtime.

Authorization in the Repository of the SAP HANA Database [page 124]
The authorization concept of SAP HANA applies in the repository of the SAP HANA database.

Cross-Database Authorization in Multitenant Database Containers [page 129]
Read-only queries between multitenant database containers 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.

8.1 Privileges

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

Table 32:

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, performing data backups, managing licenses, and so on. System privileges are also used to authorize basic repository operations. In a multiple-container system, system privileges granted to users in a particular multitenant database container authorize operations in that database only. The only exception is the system privilege DATABASE ADMIN. This system privilege can only be granted to users of the system database. It authorizes 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, and so on). 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. In a multiple-container system, 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 Multitenant Database Containers* 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. In a multiple-container system, 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. In a multiple-container system, 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*.
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*.

Note

In the SAP HANA studio, an additional privilege type can be granted. 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 *Debug an External Session* in the *SAP HANA Developer Guide* .

Related Information

Cross-Database Authorization in Multitenant Database Containers [page 129]
Authorization in SAP HANA XS Advanced [page 193]
SAP HANA Security Guide
SAP HANA Developer Guide (For SAP HANA Web Workbench)
SAP HANA Developer Guide (For SAP HANA Studio)
SAP HANA SQL and System Views Reference

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)

In a system with multitenant database containers, system privileges granted to users in a particular database container authorize operations in that database only. The only exception is the system privilege DATABASE ADMIN. This system privilege can only be granted to users of the system database. It authorizes 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.

For more information about the individual system privileges available, see *System Privileges (Reference)*.

Related Information

System Privileges (Reference) [page 97]

8.1.1.1 System Privileges (Reference)

System privileges control general system activities.

General System Privileges

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

Table 33:

System Privilege	Description
ADAPTER ADMIN	Controls the execution of the following adapter-related commands: CREATE ADAPTER, DROP ADAPTER and ALTER ADAPTER. It also allows access to ADAPTERS and ADAPTER_LOCATIONS system views.
AGENT ADMIN	Controls the execution of the following agent-related commands: CREATE AGENT, DROP AGENT, and ALTER AGENT. It also allows access to AGENTS and ADAPTER_LOCATIONS system views.
AUDIT ADMIN	Controls the execution of the following auditing-related commands: CREATE AUDIT POLICY, DROP AUDIT POLICY and ALTER AUDIT POLICY and the changes of the auditing configuration. It also allows access to AUDIT_LOG system view.
AUDIT OPERATOR	Authorizes the execution of the following command: ALTER SYSTEM CLEAR AUDIT LOG. It also allows access to AUDIT_LOG system view.
BACKUP ADMIN	Authorizes BACKUP and RECOVERY commands for defining and initiating backup and recovery procedures. It also authorizes changing of system configuration options with respect to backup and recovery.
BACKUP OPERATOR	Authorizes the BACKUP command to initiate a backup.
CATALOG READ	Authorizes users to have unfiltered read-only access to all system views. Normally, the content of these views is filtered based on the privileges of the accessing user.
CERTIFICATE ADMIN	Authorizes the changing of certificates and certificate collections that are stored in the database.

System Privilege	Description
CREATE R SCRIPT	Authorizes the creation of a procedure using the language R.
CREATE REMOTE SOURCE	Authorizes the creation of remote data sources using the CREATE REMOTE SOURCE command.
CREATE SCENARIO	Controls the creation of calculation scenarios and cubes (calculation database).
CREATE SCHEMA	Authorizes the creation of database schemas using the CREATE SCHEMA command. By default each user owns one schema, with this privilege the user is allowed to create additional schemas.
CREATE STRUCTURED PRIVILEGE	Authorizes the creation of Structured Privileges (Analytical Privileges). Only the owner of an Analytical Privilege can further grant or revoke that privilege to other users or roles.
CREDENTIAL ADMIN	Authorizes the credential commands: CREATE/ALTER/DROP CREDENTIAL.
DATA ADMIN	Authorizes reading all data in the system views. It also enables execution of any Data Definition Language (DDL) commands in the SAP HANA database. A user with this privilege cannot select or change data stored tables for which they do not have access privileges, but they can drop tables or modify table definitions.
DATABASE ADMIN	Authorizes all commands related to tenant databases, such as CREATE, DROP, ALTER, RENAME, BACKUP, and RECOVERY.
EXPORT	Authorizes export activity in the database via the EXPORT TABLE command. Beside this privilege, the user requires the SELECT privilege on the source tables to be exported.
EXTENDED STORAGE ADMIN	Required to manage SAP HANA dynamic tiering and create extended storage.
IMPORT	Authorizes the import activity in the database using the IMPORT commands. Beside this privilege, the user requires the INSERT privilege on the target tables to be imported.
INIFILE ADMIN	Authorizes changing of system settings.

System Privilege	Description
LICENSE ADMIN	Authorizes the SET SYSTEM LICENSE command to install a new license.
LOG ADMIN	Authorizes the ALTER SYSTEM LOGGING [ON|OFF] commands to enable or disable the log flush mechanism.
MONITOR ADMIN	Authorizes the ALTER SYSTEM commands for events.
OPTIMIZER ADMIN	Authorizes the ALTER SYSTEM commands concerning SQL PLAN CACHE and ALTER SYSTEM UPDATE STATISTICS commands, which influence the behavior of the query optimizer.
RESOURCE ADMIN	This privilege authorizes commands concerning system resources, for example ALTER SYSTEM RECLAIM DATAVOLUME and ALTER SYSTEM RESET MONITORING VIEW. It also authorizes many of the commands available in the Management Console.
ROLE ADMIN	This privilege authorizes the creation and deletion of roles using the CREATE ROLE and DROP ROLE commands. It also authorizes the granting and revocation of roles using the GRANT and REVOKE commands. 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 save point process using the ALTER SYSTEM SAVEPOINT command.
SCENARIO ADMIN	Authorizes all calculation scenario-related activities (including creation).
SERVICE ADMIN	Authorizes the ALTER SYSTEM [START|CANCEL|RECONFIGURE] commands. This privilege is 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	Controls the execution of the following commands: SET pse_store_name PURPOSE SSL. It also allows access to the PSES system view.

System Privilege	Description
STRUCTUREDPRIVILEGE ADMIN	Authorizes the creation, reactivation, and dropping of structured privileges.
TENANT ADMIN	Authorizes the tenant operations performed by the ALTER SYSTEM [RESUME|SUSPEND] TENANT commands.
TABLE ADMIN	Authorizes the LOAD/UNLOAD/MERGE of tables and its table placement.
TRACE ADMIN	Authorizes the ALTER SYSTEM [CLEAR|REMOVE] TRACES commands for operations on database trace files and authorizes changing trace system settings.
TRUST ADMIN	Authorizes commands to update the trust store.
USER ADMIN	Authorizes the creation and modification of users using the CREATE USER, ALTER USER, and DROP USER commands.
VERSION ADMIN	Authorizes the ALTER SYSTEM RECLAIM VERSION SPACE command of the multi-version concurrency control (MVCC) mechanism.
WORKLOAD ADMIN	Authorizes execution of the workload class and mapping commands: CREATE WORKLOAD CLASS, ALTER WORKLOAD CLASS, DROP WORKLOAD CLASS, CREATE WORKLOAD MAPPING, ALTER WORKLOAD MAPPING, and DROP WORKLOAD MAPPING
WORKLOAD ANALYZE ADMIN	Used by Analyze Workload, Capture Workload, and Replay Workload apps when performing workload analysis.
WORKLOAD CAPTURE ADMIN	Authorizes access to monitoring view M_WORKLOAD_CAPTURES to see the current status of capturing and captured workloads, as well of execution of actions with built-in procedure WORKLOAD_CAPTURE
WORKLOAD REPLAY ADMIN	Authorizes access to monitoring views M_WORKLOAD_REPLAY_PREPROCESSES and M_WORKLOAD_REPLAYS to see current status of preprocessing, preprocessed, replaying, and replayed workloads, as well as execution of actions with the built-in procedure WORKLOAD_REPLAY
<identifier>.<identifier>	Components of the SAP HANA database can create new system privileges. These privileges use the component-name as first identifier of the system privilege and the component-privilege-name as the second identifier.

> **Note**
>
> Additional system privileges (shown as `<identifier>.<identifier>` above) may exist and be required in conjunction with SAP HANA options and capabilities such as SAP HANA smart data integration. For more information, see *SAP HANA Options and Capabilities* on SAP Help Portal.

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.

Table 34:

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_WORK-SPACE	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_CONTRIBU-TION	

Related Information

Developer Authorization in the Repository [page 125]
SAP HANA Options and Capabilities
SAP HANA SQL and System Views Reference

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.

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.

> ℹ 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.
>
> For more information about resolving authorization errors with the authorization dependency viewer, see *Resolve Errors Using the Authorization Dependency Viewer* in the *SAP HANA Administration Guide* .

For more information about the object privileges available in SAP HANA and for which objects they are relevant, see *Object Privileges (Reference)*.

Related Information

Object Privileges (Reference) [page 103]
SAP HANA SQL and System Views Reference
SAP HANA Administration Guide

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 a HANA database.

Table 35:

Object Privilege	Command Types	Applies to	Privilege Description
ALL PRIVILEGES	DDL & DML	• 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.

Object Privilege	Command Types	Applies to	Privilege Description
ALTER	DDL	• Schemas • Tables • Views • Functions/procedures	Authorizes the ALTER command for the object.
CREATE ANY	DDL	• Schemas	Authorizes all CREATE commands for the object.
CREATE VIRTUAL FUNCTION	DDL	• Remote sources	Authorizes creation of virtual functions (REFERENCES privilege is also required)
CREATE VIRTUAL FUNCTION PACKAGE	DDL	• Schemas	Authorizes creation of virtual function packages.
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.
DELETE	DML	• Schemas • Tables • Views • Functions/procedures	Authorizes the DELETE and TRUNCATE commands 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).

Object Privilege	Command Types	Applies to	Privilege Description
DROP	DDL	• Schemas • Tables • Views • Sequences • Functions/procedures • Remote sources	Authorizes the DROP commands for the object.
EXECUTE	DML	• Schemas • Functions/procedures	Authorizes the execution of an SQLScript function or a database procedure using the CALLS or CALL command 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 command for the object. The INSERT and UPDATE privilege are both required on the object to allow the REPLACE and UPSERT commands to be used. While INSERT 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).
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) for a certain purpose. It also allows a user to reference a virtual function package.
SELECT	DML	• Schemas • Tables • Views • Sequences	Authorizes the SELECT command for this object or the usage of a sequence.

Object Privilege	Command Types	Applies to	Privilege Description
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), thus showing the existence of objects that may be located in other schemas.
TRIGGER	DDL	• Schemas • Tables	Authorizes the CREATE TRIGGER/DROP TRIGGER command for the specified table or the tables in the specified schema.
UPDATE	DML	• Schemas • Tables • Views	Authorizes the UPDATE/LOAD/UNLOAD/LOCK TABLE command for that object. While UPDATE 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).
`<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.

Note

Additional object privileges (shown as `<identifier>.<identifier>` above) may exist and be required in conjunction with SAP HANA options and capabilities such as SAP HANA smart data integration. For more information, see *SAP HANA Options and Capabilities* on SAP Help Portal.

Related Information

SAP HANA Options and Capabilities
SAP HANA SQL and System Views Reference

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.

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.

Creation of 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 SAP HANA deployment infrastructure (SAP HANA DI).

Note

HDI supports only SQL-based analytic privileges (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).

The following are the main differences between XML-based and SQL-based analytic privileges:

Table 36:

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 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

SAP HANA Modeling Guide (For SAP HANA Studio)
SAP HANA Developer Guide (For SAP HANA Studio)
SAP HANA Developer Guide (For SAP HANA Web Workbench)
SAP HANA Developer Guide for SAP HANA XS Advanced Model

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`.

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:

Table 37:

Package Privilege	Description
REPO.READ	Read access to the selected package and design-time objects (both native and imported)
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.

Table 38:

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"
     ]
}
```

8.2 Roles

A 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).

> **Note**
>
> Due to the container-based model of HDI, where each container corresponds to a database schema, roles are schema specific.

In SAP HANA XS advanced applications, database roles control access to database objects only (for example, tables, views, and procedures). Application roles and role collections are used to control and define access to applications. For more information about the authorization concept of XS advanced, see the *Authorization in SAP HANA XS Advanced* in 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

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

Authorization in SAP HANA XS Advanced [page 193]
SAP HANA Security Guide

8.2.1 Predefined Database 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.

Table 39:

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.

Role	Description
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.
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 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. For more information about how to configure this check, see SAP Note 1991615.
• 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 • *Getting Started with PAL* > *Security* in the SAP HANA Predictive Analysis Library (PAL) Reference • *Getting Started with BFL* > *Security* in the SAP HANA Business Function Library (BFL) Reference

Related Information

SAP Note 1991615
Predefined Database Roles for XSA [page 190]
SAP HANA Business Function Library (BFL) Reference
SAP HANA Predictive Analysis Library (PAL) Reference

8.2.2 Repository Roles

Roles that are created in the repository of the SAP HANA database as design-time objects and become runtime objects on activation are called repository roles.

Catalog Roles and Repository Roles Compared [page 118]
: It is possible to create roles as pure runtime objects that follow classic SQL principles or as design-time objects in the repository of the SAP HANA database. In general, repository roles are recommended as they offer more flexibility. For example, they can be transported between systems.

Roles as Repository Objects [page 119]
: Roles created in the repository differ from roles created directly as runtime objects using SQL in several ways.

Repository Roles in the Lifecycle of SAP HANA-Based Applications [page 121]
: 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.

Predefined Repository Roles [page 123]
: 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. The privileges required to use a software component delivered as SAP HANA content are contained within repository roles delivered with the component itself.

Repository Roles Granted to Standard Database Users [page 123]
: The privileges required to use software components delivered as SAP HANA content are contained within roles delivered with the component itself. The standard user _SYS_REPO automatically has all of these roles. Some may also be granted automatically to the standard user SYSTEM.

8.2.2.1 Catalog Roles and Repository Roles Compared

It is possible to create roles as pure runtime objects that follow classic SQL principles or as design-time objects in the repository of the SAP HANA database. In general, repository roles are recommended as they offer more flexibility. For example, they can be transported between systems.

The following table summarizes the differences between catalog roles and repository roles:

Table 40:

Feature	Catalog Roles	Repository 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)
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.
Relationship to creating database user	Roles are owned by the database user who creates them. To create a role, a database user requires all the privileges being granted to the role. If any of these privileges are revoked from the creating user, they are automatically revoked from the role. Similarly, if the creating user is dropped, all roles that he or she created are also dropped.	The technical user _SYS_REPO is the owner of roles, 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.
Grant and revoke process	Roles created in runtime are granted directly by the database user using the SQL GRANT and REVOKE statements. Roles can only be revoked by the grantor. If the granting user is dropped (not necessarily the role creator), all roles that he or she granted are revoked.	Roles are granted and revoked using built-in procedures. Any administrator with the EXECUTE privilege on these can grant and revoke roles. Role creation is decoupled from the grant and revoke process.

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 the technical user _SYS_REPO 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 only be revoked by the same user. 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.

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.

Related Information

Granting and Revoking Privileges on Activated Repository Objects [page 127]
Developer Authorization in the Repository [page 125]

8.2.2.2 Roles as Repository Objects

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? [page 119]
- What about the WITH ADMIN OPTION and WITH GRANT OPTION parameters? [page 120]
- How are repository roles granted and revoked? [page 120]
- How are repository roles dropped? [page 121]
- Can changes to repository roles be audited? [page 121]

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. 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:

Table 41:

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.

8.2.2.3 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)

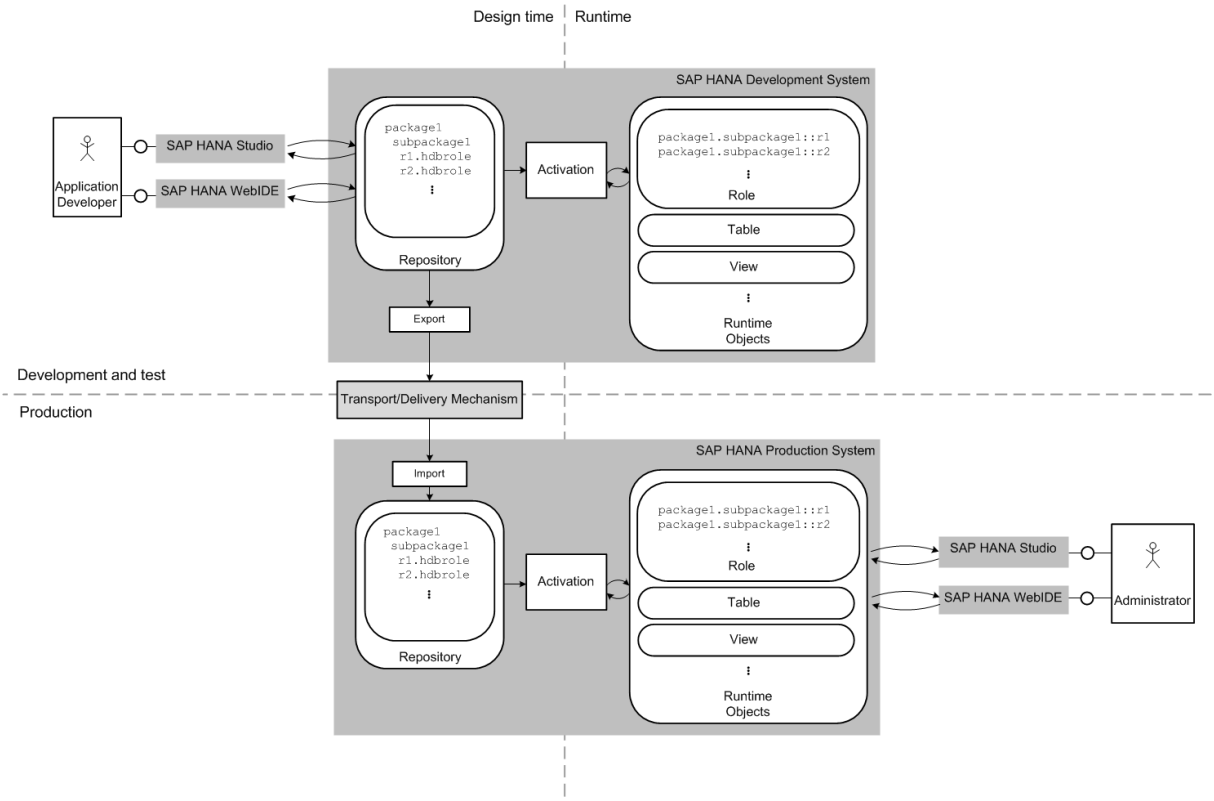
   The transport option used depends on the scenario. For example, CTS+ can be used to transport SAP HANA content in ABAP system landscapes where a CTS transport landscape is already in place.

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:

Figure 11: Lifecycle of Repository Roles



Related Information

Change and Transport System (Including CTS Plug-In)
How to transport ABAP for SAP HANA applications with HTC
SAP HANA Developer Guide (For SAP HANA Studio)

8.2.2.4 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. 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*.

> **Note**
>
> No user has any predefined repository roles initially, except the user `_SYS_REPO` (as the owner of all repository content). However, some roles may be granted automatically to the user `SYSTEM`. For more information, see *Repository Roles Granted to Standard Database Users*.

Related Information

Components Delivered as SAP HANA Content [page 245]
Repository Roles Granted to Standard Database Users [page 123]
SAP HANA Content (Security) [page 232]

8.2.2.5 Repository Roles Granted to Standard Database Users

The privileges required to use software components delivered as SAP HANA content are contained within roles delivered with the component itself. The standard user _SYS_REPO automatically has all of these roles. Some may also be granted automatically to the standard user SYSTEM.

_SYS_REPO

Like all repository roles, roles delivered with SAP HANA content are owned by the user _SYS_REPO. Therefore, _SYS_REPO has all standard repository roles.

SYSTEM

The SAP HANA cockpit is an SAP Fiori Launchpad site that provides SAP HANA administrators with a single point-of-access to a range of Web-based administration applications, which are delivered as SAP HANA content. To ensure that SAP HANA cockpit can be used "out of the box" after database creation, the database

user SYSTEM is automatically granted several roles the first time the cockpit is opened with this user. As the SYSTEM user, a user administrator can then create dedicated database users for administrative tasks and grant them these roles.

The following roles are granted:

Table 42:

Role	Description
`sap.hana.admin.roles::Administrator`	Allows users to open the SAP HANA cockpit with read-only access to monitoring data, as well as to perform database administration tasks supported by the cockpit (configure alerts, stop/start services, reset memory statistics, cancel sessions) This role also allows users to see tiles in the *SAP HANA Platform Lifecycle Management* tile catalog.
`sap.hana.xs.ide.roles::TraceViewer`	Allows users to open the *Trace* tool of the SAP HANA Web-based Development Workbench
`sap.hana.ide.roles::SecurityAdmin`	Allows users to open the *Security* tool of the SAP HANA Web-based Development Workbench
`sap.hana.admin.cockpit.sysdb.roles::SysDBAdmin`	Allows users in the system database to monitor and manage tenant databases in a multiple-container system

> **Note**
>
> `sap.hana.admin.cockpit.sysdb.roles::SysDBAdmin` is granted only if you are logging on to the system database of a multiple-container system.

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 125]
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 127]

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 127]

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:

Table 43:

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`

Privilege Type	Privileges Required
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.
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

System Privileges (Reference) [page 97]
SAP HANA Application Lifecycle Management
SAP HANA Developer Guide (For SAP HANA Web Workbench)

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.

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.
- If a user grants a privilege using the GRANT statement, the privilege 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.

When the SAP HANA studio 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 128]

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.

Table 44:

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>'/'<role>')`
	`CALL REVOKE_PRIVILEGE_ON_ACTIVATED_CONTENT ('<object_privilege>','<object>','<user>'/'<role>')`
Schema containing modeled objects	`CALL GRANT_SCHEMA_PRIVILEGE_ON_ACTIVATED_CONTENT ('<analytic_privilege>','<user>'/'<role>')`
	`CALL REVOKE_SCHEMA_PRIVILEGE_ON_ACTIVATED_CONTENT ('<analytic_privilege>','<user>'/'<role>')`
Analytic privilege	`CALL GRANT_ACTIVATED_ANALYTICAL_PRIVILEGE ('<analytic_privilege>','<user>'/'<role>')`
	`CALL REVOKE_ACTIVATED_ANALYTICAL_PRIVILEGE ('<analytic_privilege>','<user>'/'<role>')`
Application privilege	`CALL GRANT_APPLICATION_PRIVILEGE ('<application_privilege>','<user>'/'<role>')`
	`CALL REVOKE_APPLICATION_PRIVILEGE ('<application_privilege>','<user>'/'<role>')`
Role	`CALL GRANT_ACTIVATED_ROLE ('<role>','<user>'/'<role>')`

Activated Object Type	Procedure Call for Grant and Revoke
	`CALL REVOKE_ACTIVATED_ROLE ('<role>','<user>'/'<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 Multitenant Database Containers

Read-only queries between multitenant database containers 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 in a multiple-container system 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 multiple-container 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 REMOTE_USER 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

SAP HANA Administration Guide
SAP HANA SQL and System Views Reference

9 Data Storage Security in SAP HANA

Several mechanisms can be used to protect security-relevant data used by the SAP HANA database.

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, see *Recommended File System Layout* in 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 (*Configuration changes to SAP HANA system*) and SAP Note 1731000 (*Configuration changes that are not recommended*).

Server-Side Data Security

The following aspects of server-side data storage are security relevant.

- 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. This is referred to as data volume encryption.
- Passwords
  All operating system user and database user passwords are stored securely on the SAP HANA database server. In addition, credentials required by SAP HANA applications for outbound connections are stored securely in an internal credential store, which in turn is secured using the internal data encryption service.
- Data in applications developed using SAP HANA Extended Application Services (SAP HANA XS)
  Application developers can use the SAP HANA XS `$.security.Store` API to define secure stores that store application data in name-value form. These secure stores use the internal data encryption service.
- Instance secure store in the file system (SSFS)
  SAP HANA uses the instance SSFS to store all internal SAP HANA encryption keys, that is the root keys used for data volume encryption and the internal data encryption service.

  > **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. The database detects if this is not case, for example if the instance SSFS becomes corrupted, and issues an alert (check 57). It is recommended that you contact SAP Support to resolve the issue.

- System public key infrastructure (PKI) SSFS

SAP HANA uses the system PKI SSFS to protect the X.509 certificate infrastructure that is used to secure internal SSL/TLS-based communication between hosts in a multiple-host system or between processes of individual databases in a multiple-container system.

Client-Side Data Security

The following aspects of client-side data storage are also security relevant.

- Secure user store
  The SAP HANA user store (`hdbuserstore`) can be used to store user logon information to allow client applications to connect to SAP HANA without having to enter a user's password explicitly.
- 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.

Related Information

SAP Note 1730999
SAP Note 1731000
Secure Storage in the File System (AS ABAP)
SAP HANA Server Installation and Update Guide

9.1 Server-Side Data Encryption

SAP HANA features two data encryption services: data encryption in the persistence layer and an internal encryption service available to applications requiring data encryption. SAP HANA uses the secure store in the file system (SSFS) to protect the root keys for these encryption services.

Data Volume Encryption

Table 45:

What Does It Do?	What Encryption Keys Are Involved?
If enabled, this internal encryption service protects all data saved to disk from unauthorized access at operating system level. For more information, see *Data Volume Encryption* in the SAP HANA Security Guide.	Pages in the data area are encrypted using page encryption keys. Page encryption keys are encrypted with the data volume encryption root key. In a system that supports multitenant database containers, the system database and all tenant database have their own root key. The root key is generated randomly during installation. The page keys are created when data volume encryption is enabled.

Internal Data Encryption Service

Table 46:

What Does It Do?	What Encryption Keys Are Involved?
This internal 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 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 applications. For more information, see *Secure Internal Credential Store* in the SAP HANA Security Guide. • Secure stores defined using the SAP HANA XS `$.security.Store` API Application developers can create XS secure stores to store certain application data in name-value form. For more information, see *Using the Server-Side JavaScript APIs* in the *SAP HANA Developer Guide (For SAP HANA Studio)* and the *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. For more information, see *SSL Configuration on the SAP HANA Server* and *Certificate Management in SAP HANA* in the SAP HANA Security Guide.	Every consumer of the service has its own system-internal application encryption key. Application encryption keys are encrypted with the data encryption service root key. In a system that supports multitenant database containers, the system database and all tenant database have their own root key. The root key is generated randomly during installation. The application key for the internal credential store is generated randomly during the first startup. Application keys for XS secure stores are created with the XS secure store. The application key for the private key store is created when the first private key is set for an in-database PSE.

Instance Secure Store in the File System (SSFS)

Table 47:

What Does It Do?	What Encryption Keys Are Involved?
This secure store stores internal root keys in the file system.	The master key of the instance SSFS encrypts the data volume encryption root key and the data encryption service root key. **Recommendation** The initial master key that protects the instance SSFS is changed during installation or update. 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. **Note** The default path of the key file is `$DIR_GLOBAL/hdb/security/ssfs`. If you change the default path, you may need to reconfigure it in the event of a system rename.

System PKI (Public Key Infrastructure) SSFS

Table 48:

What Does It Do?	What Encryption Keys Are Involved?
This secure store stores internal root keys in the file system.	The master key of the system PKI SSFS protects the X.509 certificate infrastructure that is used to secure internal SSL/TLS-based communication between hosts in a multiple-host system or between processes of individual databases in a multiple-container system. **Recommendation** The initial master key that protects the system PKI SSFS is changed during installation or upgrade. 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.

The following figure illustrates the encryption keys protected by the instance SSFS.

Note

The system PKI SSFS is not depicted. For more information about the system PKI, see *Secure Internal Communication*.

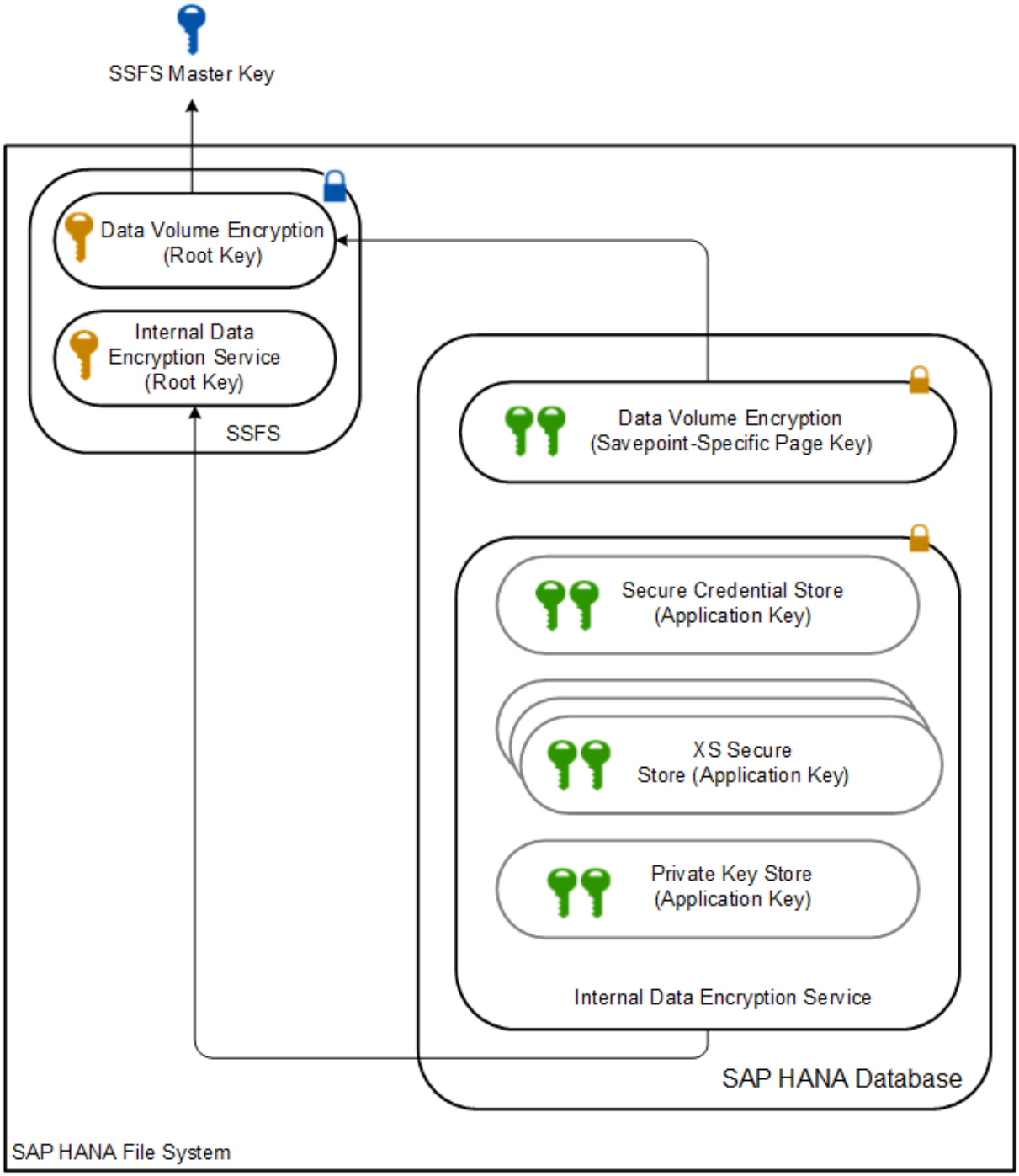


Figure 12: Encryption Keys Protected by the Instance SSFS

Related Information

Data Volume Encryption [page 141]
Secure Internal Credential Store [page 146]
SSL Configuration on the SAP HANA Server [page 39]
Certificate Management in SAP HANA [page 168]
SAP HANA Security Guide

SAP HANA Developer Guide (For SAP HANA Studio)
SAP HANA XS JavaScript API Reference
Secure Storage in the File System (AS ABAP)

9.1.1 Encryption Key Management

SAP HANA generates unique root keys on installation. However, if you received SAP HANA pre-installed from a hardware vendor, you might want to change them to ensure they are not known outside your organization. We recommend that you do this immediately after handover from your hardware partner.

The following root keys exist and can be changed:

- Instance SSFS master key
- System PKI SSFS master key
- Data volume encryption root key
- Data encryption service root key

Reinstalling your system will change all master and root keys. You can change keys manually and individually.

The following sections explain how and when you can safely change root keys. More detailed instructions are available in the *SAP HANA Administration Guide*.

SSFS Master Keys

Table 49:

How to Change	When to Change
Using the command line tool `rsecssfx` The commands are: `generatekey` and `changekey` **Remember** You'll need operating system access (`<sid>`adm user) to execute `rsecssfx` commands. For more information, see *Change the SSFS Master Keys* in the *SAP HANA Administration Guide*.	Unique master keys are generated during installation or update. However, if you received your system pre-installed from a hardware or hosting partner, we recommend that you change them immediately after handover to ensure that they are not known outside of your organization. You can also change the master keys any time later. **Note** In a system-replication configuration, you change the SSFS master keys on the primary system. To trigger replication of new keys to the secondary system, you must subsequently restart the secondary system. In multitier 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 keys have been replicated, the new keys will be overwritten with the old ones.

Data Volume Encryption Root Key

Table 50:

How to Change	When to Change
Using the *Data Volume Encryption* app of the SAP HANA cockpit or the SQL command `ALTER SYSTEM PERSISTENCE ENCRYPTION CREATE NEW ROOT KEY` For more information, see *Change the Root Encryption Key for Data Volume Encryption* in the *SAP HANA Administration Guide*.	A unique root key 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. You can also change it any time later. **Note** In a system-replication configuration, change the root key used for data volume encryption in the primary system only. The new key will be propagated to all secondary systems.

Data Encryption Service Root Key

Table 51:

How to Change	When to Change
Using the command line tool `hdbnsutil` The command is: `generateRootKeys type=DPAPI` **Remember** You'll need operating system access (`<sid>`adm user) to execute `hdbnsutil` commands. **Caution** After you change the root key with the command `generateRootKeys type=DPAPI`, you must **immediately** do the following two things: • Reset the consistency information in the SSFS using the SAP HANA tool `hdbcons`. • Change all application keys so that they are encrypted with the new root key. For more information, see *Change the Data Encryption Service Root Key* in the *SAP HANA Administration Guide*.	A unique root key 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. You must change this key at the latest before any data is encrypted using the service. This means before you create any of the following things: • A remote data source • A HTTP destination • An XS secure store • A certificate collection with a private key You can use the following system views to see whether any data has already been encrypted: • CREDENTIALS (PUBLIC) If the credential store is empty, then this view will also be empty. • P_DPAPI_KEY_ (SYS) If there are no XS secure stores, then this view will have no records with the caller `XsEngine`. If there are no certificate collections with private keys, there will be no records with the caller `PSEStore`. Only the user SYSTEM can access this view. Additionally, if the system supports multitenant database containers, you must change the root key before any tenant databases have been created. **Caution** It is important that you plan this root key change carefully as you will have to shut down the database. Not only that, but changing the root key after data has been encrypted will result in key information in the SSFS and the database becoming inconsistent and encrypted data becoming inaccessible. Rectifying the problem could result in data loss. We recommend that you contact SAP Support if errors related to inconsistent SSFS or encryption failure occur.

Related Information

SAP HANA Administration Guide

SAP Note 2097613

9.1.2 Cryptographic Service Provider

All encryption services used in SAP HANA require the availability of a cryptographic service provider on the SAP HANA server.

SAP HANA supports the following cryptographic libraries:

- CommonCryptoLib (default)
  CommonCryptoLib (`libsapcrypto.so`) is installed by default as part of SAP HANA server installation at $DIR_EXECUTABLE.
- OpenSSL
  The OpenSSL library is installed by default as part of the operating system installation.

> **Note**
>
> If you are using OpenSSL, it is recommended that you migrate to CommonCryptoLib after an upgrade to Support Package Stack (SPS) 09. For more information, see SAP Note 2093286.

Related Information

SAP Note 1848999 - Central Note for CommonCryptoLib 8 (replacing SAPCRYPTOLIB)
SAP Note 2093286 - Migration from OpenSSL to CommonCryptoLib (SAPCrypto)

9.2 Data 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.

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. Data is automatically saved from memory to disk at regular savepoints. The data belonging to a savepoint represents a consistent state of the data on disk and remains so until the next savepoint operation has completed. After a power failure, the database can be restarted like any disk-based database and returns to its last consistent state.

If data volumes are encrypted, all pages that reside in the data area on disk are encrypted using the AES-256-CBC algorithm. Pages are transparently decrypted as part of the load process into memory. When pages reside in memory they are therefore 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.

Pages are encrypted and decrypted using 256-bit page encryption keys. 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.

During start-up, administrator interaction is not required. The data volume encryption root key is stored using the secure storage in the file system (SSFS) functionality and is automatically retrieved from there. SAP HANA uses the SSFS to protect the root encryption keys that are used to protect all encryption keys used in the SAP HANA system from unauthorized access. Root keys are encrypted using the SSFS master key.

Enabling data volume encryption does not increase data size.

Implementation Considerations

SAP HANA System Replication

If you want to use data volume encryption in systems involved in system replication, be aware of how parameter replication is configured before enabling encryption.

Parameter replication determines whether changes to parameter configuration (*.ini) files made in the primary system are automatically replicated to secondary systems (default) or must be re-applied manually. Parameter replication is configured in the `global.ini` file with the parameter `[inifile_checker] enable`.

- If `[inifile_checker] enable` is set to **`true`**, enable data volume encryption in the primary system only. It will be automatically enabled on secondary systems.
- If `[inifile_checker] enable` is set to **`false`**, you must manually enable data volume encryption on secondary systems. But only do so after you have finished configuring system replication and after encryption has been enabled on the primary system.
  If you enable encryption before, the SSFS will become inconsistent and encrypted data inaccessible.

SAP HANA Options

Do **not** enable data volume encryption if you plan to use the SAP HANA dynamic tiering. It is not possible to create extended storage in encrypted SAP HANA databases. Be aware that you need additional licenses for SAP HANA options and capabilities such as SAP HANA dynamic tiering. For more information, see Important Disclaimer for Features in SAP HANA Platform, Options and Capabilities [page 270].

Data Not Encrypted

The data volume encryption feature does **not** encrypt the following data:

- Database redo log files
  If database redo log files need to be protected, we recommend using operating system facilities, such as encryption at the file system level.
- Database backups
  In general, the contents of both data and log backups are not encrypted. Only data that has been encrypted internally in the database (that is, independently of the data volume encryption feature) remains encrypted in backups. This applies to data stored in the secure internal credential store.

> **Note**
>
> To ensure that all data restored during the data and log recovery phases is encrypted, encryption must be enabled before the recovery is started.

If encryption of backups is required, we recommend using third-party solutions that integrate with the Backint for SAP HANA functionality for backups.

> **Note**
>
> Unlike data backups, data in storage snapshots **is** encrypted. This is because a storage snapshot captures the content of the data area exactly as it is at a particular point in time.

- Database traces
  For security reasons, we recommend that you do not run the system 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 beyond the respective analysis task.

Administration Tasks

The recommended process for managing data volume encryption is as follows:

1. Change the root key used for data volume encryption.
   SAP HANA generates unique root keys on installation. However, if you received SAP HANA pre-installed from a hardware or hosting partner, you might want to change the root key used for data volume encryption to ensure it is not known outside your organization.
2. Enable data volume encryption.
   Data volume encryption is not enabled by default. We recommend that you enable it immediately after installation or handover from your hardware or hosting partner.
3. Periodically change page keys.
   Depending on your security policy, we recommend periodically changing the page keys in order to limit the potential impact of a key being compromised. A new page key will be active for new data as of the next savepoint operation. The SAP HANA database provides system views that allow you to monitor encryption status (`M_PERSISTENCE_ENCRYPTION_STATUS`), as well as the page keys used for data encryption and their age (`M_PERSISTENCE_ENCRYPTION_KEYS`). An administrator can also trigger a re-encryption of the entire data area using a newly-generated page key.

The above administration tasks can be done using the *Data Volume Encryption* app of the SAP HANA cockpit, the Security editor of the SAP HANA studio, or the SQL system management statement `ALTER SYSTEM PERSISTENCE ENCRYPTION`.

For more information, see *Managing Encryption of Data Volumes in the SAP HANA Database* in the *SAP HANA Administration Guide*.

Related Information

SAP HANA SQL and System Views Reference
Secure Storage in the File System (AS ABAP)
SAP HANA Administration Guide

9.2.1 Data Volume Encryption in Multitenant Database Containers

Data volume encryption can be enabled individually for tenant databases in a multiple-container system.

Ideally, you enable encryption immediately after installation or upgrade of SAP HANA. This also applies to systems installed in multiple-container mode. Any subsequently created tenant databases will then automatically have encryption enabled. If a particular tenant database does not require encryption, the tenant database administrator can switch it off independently of the system in the Security editor of the SAP HANA studio or using the *Data Volume Encryption* app of the SAP HANA cockpit.

If encryption is not enabled after system installation, you can enable it retroactively either for all tenant databases together by making the setting in the system database, or for individual tenant databases by making the setting in the relevant tenant database.

> ⚠ Caution
>
> If you enable data volume encryption after a tenant database has been created and is already in operation, only the pages in use within the data volumes will be 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. This means that your data will only be fully protected after some delay. To attain complete protection immediately, the overall process is:
>
> 1. Perform a data backup.
> 2. Drop the tenant database.
> 3. Clean the disk space.
> 4. Create the tenant database again.
> 5. Enable encryption.
> 6. Perform a data recovery.

9.3 Secure Storage of Passwords in SAP HANA

All passwords in SAP HANA are stored securely in a hashed and salted form and never in clear text.

Server Side

On the SAP HANA database server, passwords are stored securely as follows:

- Operating system user passwords are protected by the standard operating system mechanism, `/etc/passwd` file.
- All database user passwords are hashed with the secure hash algorithm SHA-256.

In addition, a secure database-internal credential store is available that allows you to securely store in the SAP HANA database the credentials required by SAP HANA applications for outbound connections. For example, in an SAP HANA smart data access scenario, in order to retrieve data, credentials are required to access a remote source.

Client Side

On the client side, the following facilities are available for storing user passwords:

- The SAP HANA user store (`hdbuserstore`)
  The SAP HANA user store 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.
- Eclipse secure storage
  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 the *SAP HANA Administration Guide*.

> ⚠ Caution
>
> 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. 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

SAP HANA Administration Guide
SAP HANA Client Installation and Update Guide

9.3.1 Secure Internal Credential Store

A database-internal credential store is available that allows you to securely store in the SAP HANA database the credentials required by SAP HANA applications for outbound connections. For example, in an SAP HANA smart data access scenario, in order to retrieve data, credentials are required to access a remote source.

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 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:

Table 52:

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 or X509 The supported values for the this element are specific to the application.

Element	Description
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.

Example

```
CREATE CREDENTIAL FOR USER TESTUSER COMPONENT 'SAPHANAFEDERATION' PURPOSE 'ASE'
TYPE 'PASSWORD' USING 'user="remotedbuser";password="abc123"'
```

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 safe and secure place.

Note

The credential store uses the data encryption service of the SAP HANA database. The root encryption key for this data encryption service is stored in the secure store in the file (SSFS) system along with the root encryption key used for data volume encryption (if activated). During a recovery, the root encryption key for the data encryption service is restored to the target system's SSFS without interfering with the root encryption key for data volume encryption of the target system.

Related Information

Server-Side Data Encryption [page 134]
Encryption Key Management [page 138]
SAP HANA SQL and System Views Reference
SAP HANA Administration Guide
SAP HANA Developer Guide (For SAP HANA Studio)

9.3.2 Secure User Store (hdbuserstore)

The secure user store (`hdbuserstore`) is a tool installed with the SAP HANA client. You use it to store connection information to SAP HANA systems securely on the client so that client applications can connect to SAP HANA without users having to enter this information. It is typically used by scripts connecting to SAP HANA.

The secure user store allows you to store SAP HANA connection information, including user passwords, securely on clients. In this way, client applications can connect to SAP HANA without the user having to enter host name or logon credentials. You can also use the secure store to configure failover support for application servers in a 3-tier scenario (for example, SAP Business Warehouse) by storing a list of all the hosts that the application server can connect to.

> **Note**
>
> The secure user store can only be used for SQLDBC and JDBC-based connections. The SAP HANA studio does not use the SAP HANA secure user store, but the Eclipse secure storage. For more information, see the Eclipse documentation.

The secure user store is installed with the SAP HANA client package. After you install the SAP HANA client, the `hdbuserstore` program is located in one for the following directories:

- `/usr/sap/hdbclient` (Linux/UNIX)
- `%SystemDrive%\Program Files\sap\hdbclient` (Microsoft Windows)

The secure store runs on all platforms supported by SAP HANA client interfaces and SAP BASIS 7.20 EXT.

To access the secure store using JDBC, there are two connect options: `key` and `virtualHostName`. `key` is the `hdbuserstore` key that you use to connect to SAP HANA, while `virtualHostName` specifies the virtual host name. This option allows you to change where the `hdbuserstore` searches for the data and key files.

To connect, define the `hdbuserstore` key using the `key` connect option. JDBC only supports reading the key and data files for existing keys and using those keys to connect to SAP HANA.

Managing the Secure Store

Connection information stored in the secure store is saved in the secure store file `SSFS_HDB.DAT`.

On Microsoft Windows, the path of this file is defined by `<PROGRAMDATA>\.hdb\<COMPUTERNAME>\<SID>`, where:

- `<PROGRAMDATA>` is the path defined by constant `CSIDL_COMMON_APPDATA/FOLDERID_PROGRAMDATA`.
- `<COMPUTERNAME>` is the computer name.
- `<SID>` is the system ID of the user that uses the stored logon information.

For Linux/UNIX systems, the path is defined by `<HOME>/.hdb/<COMPUTERNAME>` where:

- HOME is the home directory of the user that uses the logon information.
- COMPUTERNAME is the computer name.

If the path does not already exist, then the `hdbuserstore` program creates it.

The secure store's content is stored in a platform-dependent way. You cannot copy the secure store from one platform to another.

Managing Connection Information

Use the `hdbuserstore` program to store and manage connection information in the secure store. For more information about the available commands, see *hdbuserstore Commands*.

The secure user store is user specific, so only the operating system user who owns the corresponding secure store file can access the secure store. However, it is possible, with the appropriate operating system privileges, to manage another user's secure store. This behavior is needed, for example, to manage the connection details for ABAP on Microsoft Windows since the application server is running under a different user (SAPService`<SAPSID>` instead of `<SAPSID>`adm).

Using Stored Connection Information

When the secure store is accessed in the context of the correct operating system user, you can open it with a user key.

Table 53:

Client	How to Connect Using a Stored User Key
SAP HANA HDBSQL	In SAP HANA HDBSQL, you specify the key to be used with the -U connection option: `hdbsql -U <KEY>`
ODBC	In ODBC, you specify the user store key with the @ sign in your data source: `servernode=@<KEY>`
ABAP	ABAP uses the key DEFAULT by default.

Managing the Encryption Key

The initial default encryption key of the secure user store is automatically changed when the first entry is created.

In older revisions, password information contained in the secure user store may have been encrypted using the initial default encryption key. As of revision 102, this key is automatically changed the first time the `SET` or `DELETE` commands are executed. You can also change it explicitly by using the `CHANGEKEY` command. The SET and DELETE commands implicitly execute the CHANGEKEY command. For more information, see *Change the Secure User Store Encryption Key* in the *SAP HANA Administration Guide*.

If a user forgets the stored password, then you cannot recover that password because the system does not display passwords in a human-readable form.

Connecting to a Requested Database in a Multitenant Database Container Setup

You can associate a key with tenant database information for use in a connection attempt. The server keeps track of which tenant databases are assigned to which ports for a host in the system database. You should not have multiple host name/port pairs associated with the key, but instead only supply the host name/port pair for the system database that you plan to connect to. The database name, when supplied in a connection attempt, is used to query a system database that runs on a well-defined port.

A failover may occur if one or more hosts in the connection list is down. Only one database may be supplied to a port. Therefore, whichever database fails over first is the database that is assigned to the port.

The following example sets the key for a tenant database:

```
set new-key host-name:30013@Tenant-DB-Name myusername mypassword
```

Related Information

Secure Storage in the File System (AS ABAP)
hdbuserstore Commands [page 150]
SAP HANA Administration Guide
SAP HANA Client Installation and Update Guide

9.3.2.1 hdbuserstore Commands

Several commands are available for managing connection information stored in the secure user store of the SAP HANA client (`hdbuserstore`).

You store and manage connection information in the user store with the `hdbuserstore` program. Execute commands using the following syntax:

```
hdbuserstore [OPTION]... COMMAND [PARAMETER]...}}
```

Table 54: Command Options

Option	Description
-h	Displays a help message
-H <HOST>	Assumes host name <HOST>
-i	Enables interactive mode

Option	Description
-u <USER>	Execute command on the user store of user <USER> **Note** You must have administrator privileges to work on the store of a different user.
-v	Executes command in verbose mode

Table 55: Commands

Command	Parameter	Description
HELP	-	Displays a help message
LIST	KEY	Lists entries with the key Passwords are not displayed.
DELETE	KEY	Deletes entries with the key
SET	KEY	Sets the entry key
	ENV	Sets the connection environment (host, port and optionally database name)
	USERNAME	Sets the user name for the profile
	PASSWORD	Sets the password for the profile **Note** We recommend executing the SET command in interactive mode so that you are prompted to enter the password. If you enter the password directly in the command, it is stored in your shell's command history.
CHANGEKEY	-	Randomly generates a new master encryption key and re-encrypts password of all keys with the new master key

Example

Table 56:

Action	Command	Example
Create a user key in the user store	hdbuserstore SET <KEY> <ENV> <USERNAME> <PASSWORD>	hdbuserstore SET millerj "localhost:30115" JohnMiller 2wsx$RFV

Action	Command	Example
List all available user keys (passwords are not displayed)	`hdbuserstore LIST <KEY>`	`hdbuserstore LIST millerj` The following information is displayed: • `KEY: millerj` • `ENV: localhost:30115` • `USER: JohnMiller`
Configure failover support for application server by specifying a list of host names that the server can connect to	`hdbuserstore SET DEFAULT "<hostname_node1>:3<inst>15; … ;<hostname_node(n)>:3<inst>15" <sapsid> <password>`	`hdbuserstore SET default "ld9490:33315;ld9491:33315;ld9492:33315;ld9493:33315" SAPP20 <Password>`
Configure failover support for client running on a distributed tenant database by specifying a list of host names that the client can connect to	`hdbuserstore SET <keyname> <hostname>:<port>[,<hostname2>:<port2>, …..]@<database name> <user> <password>`	`hdbuserstore SET DB1 "host1:30040;host3:30040 DB1" JohnMiller <password>`

9.4 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.

10 Auditing Activity in SAP HANA Systems

Auditing provides you with visibility on who did what in the SAP HANA database (or tried to do what) and when.

Auditing allows you to monitor and record selected actions performed in the SAP HANA database. Although auditing does not directly increase your system's security, if wisely designed, it can help you achieve greater security in the following ways:

- Uncover security holes if too many privileges were granted to some user
- 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

Constraints

Only actions that take place inside the database engine can be audited. If the database engine is not online when an action occurs, it cannot be detected and therefore cannot be audited.

This is important to bear in mind in the following cases:

- Upgrade of a SAP HANA database instance
  Upgrade is triggered when the instance is offline. When it becomes available 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
  Only changes that are made using SQL are visible to the database engine. It is also possible to change configuration files when the system is offline.

10.1 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.

Audited Actions

An action corresponds to the execution of an action in the database by SQL statement. For example, you want to track user provisioning in your system, so 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 will audit 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 other 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.

- 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.

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 narrow the number of events actually audited.

- Audited action status
  For each audit policy, it must be specified 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 for 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.
- 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 for 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 audit trail target for the audit level of the policy if configured, or the audit trail target configured for the system.

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**
>
> Policy-specific audit trails are not possible in tenant databases. The audit trail targets configured for the system or audit level apply, by default internal database table. 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*.

For more detailed information about audit trails, see *Audit Trails*.

Related Information

Audit Trails [page 158]
System Properties for Configuring Auditing [page 165]
SAP HANA SQL and System Views Reference

10.1.1 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 action 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 this audit level, entries are written to the audit trail configured for the system.

Table 57:

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. It is not possible to delete audit entries from the syslog audit trail target.

Action	Description
• `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 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

10.2 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:

Table 58:

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 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 informed. 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.
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 view AUDIT_LOG. Only SELECT operations can be performed on this view by users with the system privilege AUDIT OPERATOR or AUDIT ADMIN. To avoid the audit table growing indefinitely, it is possible to delete old audit entries by truncating the table. 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.

Additionally, the option exists to store the audit trail in a CSV text file. This should only be used for test purposes in non-production systems. A separate CSV file is created for every service that executes SQL.

⚠ Caution

You must not use a CSV text file for a production system as it has severe restrictions.

Firstly, it is not sufficiently secure. By default, the file is 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 it. In the Administration editor of the SAP HANA studio, it is listed on the *Diagnosis Files* tab, and at operating system level, any user in the SAPSYS group can access it.

Secondly, audit trails are created for each server in a distributed database system. This makes it more difficult to trace audit events that were executed across multiple servers (distributed execution).

Multiple Audit Trails

When you enable auditing, you must specify an audit trail target for the system. In addition, you can configure separate audit trail targets based on the severity of the action being audited, that is the audit level.

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 system.

Audit policy-specific targets are also possible. 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 system. Several audit trail targets are configurable for each individual policy.

Audit Entry Layout

For each occurrence of an audited action, one or more audit entries are created and written to the configured audit trail(s).

The layout of audit entries varies depending on the audit trail type.

Example

If an 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.

Audit Trails in Multitenant Database Containers

Tenant database administrators cannot change the audit trail targets at any level in their databases. The audit trail is by default always written to the internal database table.

The system administrator can configure audit trail targets for the system database. In addition, he can change the audit trail targets for tenant databases.

> **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.

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.

10.2.1 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 text file is the trail target:

Table 59:

Field	Description	Example
Event Timestamp	Local system time of event occurrence	2012-09-19 15:44:53
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 multitenant database container in a multiple-container system **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
Audit Level	Severity of audited action	CRITICAL

Field	Description	Example
Audit Action	Action that was audited and thus triggered the policy	GRANT PRIVILEGE
Active User	User who performed the action	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.
Origin database name	Name of tenant database in which the query originated; relevant for cross-database queries in multitenant systems	DB1
Origin user name	Name of database user who executed the query in the origin tenant database; relevant for cross-database queries in multitenant systems	MYADMIN

Example

```
2013-11-30 13:04:54;indexserver;myhanablade23.customer.corp;HAN;
01;30103;10.29.14.177;lu306309;6776;58060;Alter User Policy;INFO;ALTER
USER;SYSTEM;;;;;;ADAMS;SUCCESSFUL;;;;;;alter user ADAMS VAXXXXXXXXXXXXX;434597;
```

10.2.2 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.

The following table below describes the layout of the audit trail when the column store database table is the trail target:

Table 60:

Column	Data Type	Description
TIMESTAMP	TIMESTAMP	Time of event occurrence (in system local time)
HOST	VARCHAR(64)	Name of the host where the action occurred
PORT	INTEGER	Port number
SERVICE_NAME	VARCHAR(32)	Name of the service where the action occurred
CONNECTION_ID	INTEGER	ID of the session in which the statement was executed
CLIENT_HOST	VARCHAR(64)	Name of the client machine
CLIENT_IP	VARCHAR(16)	IP address of the client application
CLIENT_PID	BIGINT	Process ID of the client process

Column	Data Type	Description
CLIENT_PORT	INTEGER	Port of the client process
USER_NAME	NVARCHAR(256)	User who performed the action
APPLICATION_USER_NAME	NVARCHAR(256)	Application user who performed the action **⚠ Caution** Treat this information with caution. It comes from the application and SAP HANA has no way of verifying its authenticity.
AUDIT_POLICY_NAME	NVARCHAR(256)	Audit policy that was triggered
EVENT_STATUS	VARCHAR(32)	Execution status of the statement
EVENT_LEVEL	VARCHAR(16)	Severity of audited action
EVENT_ACTION	VARCHAR(32)	Action that was audited and thus triggered the policy
SCHEMA_NAME	NVARCHAR(256) NULL	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
OBJECT_NAME	NVARCHAR(256) NULL	Name of the object on which an action was performed, for example, a privilege was granted
PRIVILEGE_NAME	NVARCHAR(256) NULL	Name of the privilege that was granted or revoked
ROLE_NAME	NVARCHAR(256) NULL	Name of the role that was granted or revoked
GRANTEE	NVARCHAR(256), NULL	Name of the target user of the action, for example, grantee in a GRANT statement
GRANTABLE	VARCHAR(16), NULL	Indication of whether the privilege or role was granted with or without GRANT/ADMIN OPTION
FILE_NAME	VARCHAR(256), NULL	Configuration file name, for example global.ini
SECTION	VARCHAR(128), NULL	Configuration section name, for example auditing configuration
KEY	VARCHAR(128), NULL	Configuration parameter, for example `global_auditing_state`
PREV_VALUE	VARCHAR(5000), NULL	Previous value of the parameter, for example CSVTEXTFILE
VALUE	VARCHAR(5000), NULL	New parameter value, for example CSTABLE
STATEMENT_STRING	NCLOB, NULL	Statement that was executed
COMMENT	VARCHAR(5000) NULL	Additional information about the audited event **ℹ Note** Currently in case of failed logon attempts, the reason for failure appears in this field.
ORIGIN_DATABASE_NAME	NVARCHAR(256) NULL	Name of tenant database in which the query originated; relevant for cross-database queries in multitenant systems

Column	Data Type	Description
ORIGIN_USER_NAME	NVARCHAR(256) NULL	Name of database user who executed the query in the origin tenant database; relevant for cross-database queries in multitenant systems

10.3 Auditing Configuration and Audit Policy Management

To audit database activity, auditing must first be enabled in the system, 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.

You configure auditing and manage auditing policies in the *Auditing* app of the SAP HANA cockpit or the *Security* editor of the SAP HANA studio. The underlying system properties are in the `auditing configuration` section of the `global.ini` system properties file.

Multitenant Database Containers

Auditing can be enabled individually for every database in a multiple-container system. For tenant databases, the underlying system property (`[auditing configuration] global_auditing_state`) is set at the database layer of the `global.ini` file. For the system database, it is set in the `nameserver.ini` file.

Tenant database administrators **cannot** configure audit trail targets independently for their database. The default target for all audit trails in tenant databases is internal database table. The system administrator may change the default audit trail targets for tenant databases by changing the underlying property (`[auditing configuration] *_audit_trail_type`) in the `global.ini` file.

> ⚠ 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 polices are database specific and can only audit activities in that database.

Related Information

SAP HANA SQL and System Views Reference
SAP HANA Administration Guide

10.3.1 System Properties for Configuring Auditing

The system properties for configuring auditing are in the `auditing configuration` section of the `global.ini` system properties file.

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 or the *Security* editor of the SAP HANA studio instead.

Table 61:

System Property	Value	Default Value	Description
`global_auditing_state`	`<Boolean value>`	false	Activation status of auditing in the system **Note** In the system database of a multiple-container system, this property is set 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}	SYSLOGPROTOCOL if global_auditing_state is true	Default audit trail target of the database
`default_audit_trail_path`	`<file>`	`/usr/sap/<sid>/<instance>/<host>/trace` if `default_audit_trail_type` is CSVTEXTFILE	The file path of audit trail target CSVTEXTFILE
`emergency_audit_trail_type`	{SYSLOGPROTOCOL | CSTABLE | CSVTEXTFILE}	--	Audit trail target to which audit entries from audit policies with the audit level EMERGENCY are written
`alert_audit_trail_type`	{SYSLOGPROTOCOL | CSTABLE | CSVTEXTFILE}	--	Audit trail target to which audit entries from audit policies with the audit level ALERT are written
`critical_audit_trail_type`	{SYSLOGPROTOCOL | CSTABLE | CSVTEXTFILE}	--	Audit trail target to which audit entries from audit policies with the audit level CRITICAL are written

System Property	Value	Default Value	Description
`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.

Example

Configure Multiple Audit trails per Audit Level

- `ALTER SYSTEM ALTER CONFIGURATION ('global.ini','SYSTEM') set ('auditing configuration', 'emergency_audit_trail_type' ) = 'CSTABLE,SYSLOGPROTOCOL' with reconfigure;`
- `ALTER SYSTEM ALTER CONFIGURATION ('global.ini','SYSTEM') set ('auditing configuration', 'alert_audit_trail_type' ) = 'CSTABLE,SYSLOGPROTOCOL' with reconfigure;`
- `ALTER SYSTEM ALTER CONFIGURATION ('global.ini','SYSTEM') set ('auditing configuration', 'critical_audit_trail_type' ) = 'CSTABLE,SYSLOGPROTOCOL' with reconfigure;`

Example

Enable Auditing in Multitenant Database Containers

- 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';`

10.4 Best Practices for Creating Audit Policies

To reduce the performance impact of auditing, some basic guidelines for creating audit policies apply.

- Create as few audit policies as possible. It's usually better to have one complex policy than several simple ones.

  > **Note**
  >
  > Some audit actions can't 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.
- Don't 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* in the *SAP HANA Security Guide*.
- Don't 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 tables that cannot be accessed by any user, not even `SYSTEM`. Changes in these tables are carried out by internal mechanisms, and not by DML operations. Don't included these tables in an audit policy. Instead create an audit policy for changes to users (`ALTER USER` action) instead.

Related Information

Actions Audited by Default Audit Policy [page 156]
SAP HANA Security Guide

11 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 be stored and managed in files in the file system and in some cases directly in the SAP HANA database.

Certificate Management in the Database

All certificate-based user authentication mechanisms in SAP HANA, as well as secure communication between SAP HANA and clients that access the SQL interface of the database rely on X.509 client certificates for authentication and verifying digital signatures. For ease of management, it's possible to store these certificates and configure their usage directly in the SAP HANA database.

In systems that support multitenant database containers, in-database certificates are also used to secure communication during the process of copying or moving a tenant database between two systems. For more information, see *Copying and Moving Tenant Databases Between Systems* in the *SAP HANA Administration Guide*.

The following figure shows a typical certificate management workflow. A full separation of duties is possible through user authorization. For more information, see *SQL Statements and Authorization for In-Database Certificate Management*.

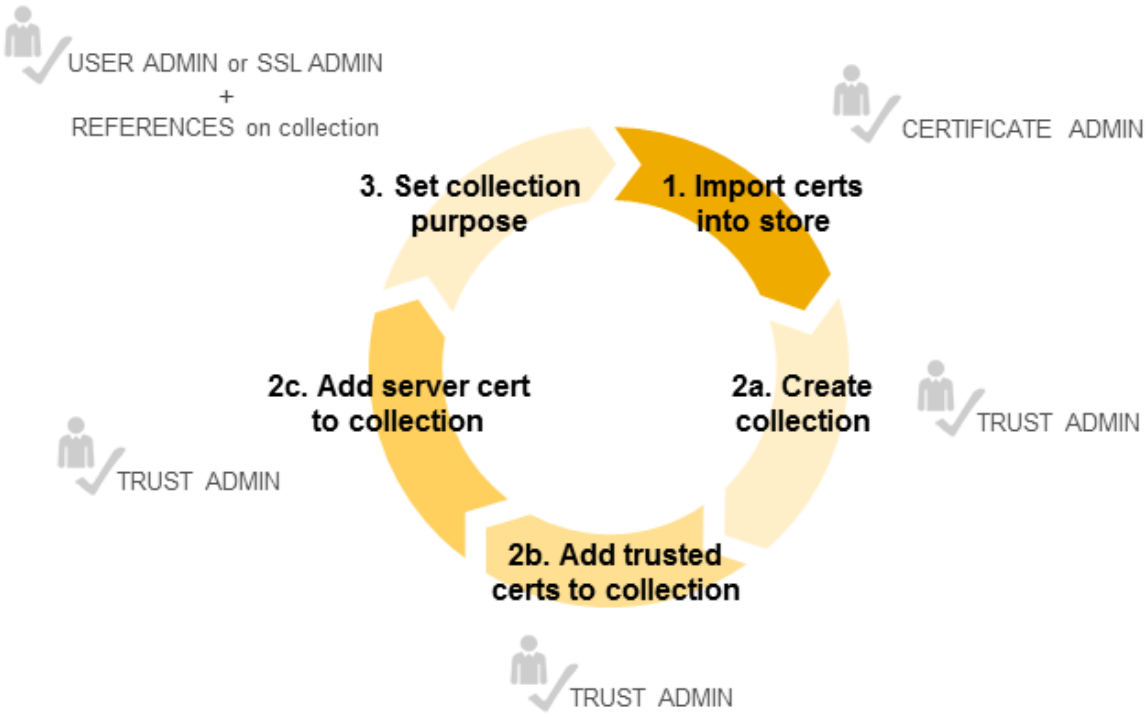


Figure 13: Certificate Management Workflow

You can manage certificates in the SAP HANA cockpit.

> **Note**
>
> Roles are required to access the certificate management apps of the SAP HANA cockpit. The privileges for certificate management indicated above are partially included in these roles.

Certificate Management in the File System

Although we recommend using in-database storage, it is possible to store and manage the certificates required for certificate-based user authentication and secure client-server communication in trust and key stores located in the file system.

> **Recommendation**
>
> If you migrate from managing certificates in the file system to managing them in the database, delete all related files from the file system to avoid any potential conflicts. For more information, see SAP Note 2175664.

The certificates required to secure all internal communication channels and HTTP client access using SAP Web Dispatcher are contained in files located in the file system. In-database storage of certificates for these communication channels is not supported. Do not delete these files from the file system.

For more information about how to configure the usage of trust and key stores in the file system, see *Server-Side SSL Configuration Properties for External Communication* in the *SAP HANA Security Guide*.

Overview of Certificate Handling

Table 62:

Certificates can be stored for...	...in the database	...in the file system
Secure client-server communication over JDBC/ODBC	Yes	Yes
Server client-server communication over HTTP	No	Yes
Secure internal communication	No	Yes
User authentication (SAML assertions, SAP logon and assertion tickets, X.509 certificates)	Yes	Yes

Related Information

SSL Configuration on the SAP HANA Server [page 39]
SQL Statements and Authorization for In-Database Certificate Management [page 172]
Server-Side TLS/SSL Configuration Properties for External Communication (JDBC/ODBC) [page 41]
SAP HANA Administration Guide
SAP Note 2175664 - Migration of file system-based X.509 certificate stores to in-database certificate stores
SAP HANA Administration Guide

11.1 Client Certificates

X.509 client certificates required for certificate-based 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.
- 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 data encryption service of the SAP HANA database. For more information, see *Server-Side Data Encryption* in the *SAP HANA Security Guide*.

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 (either secure client-server communication or a certificate-based authentication mechanism).

> **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.

Related Information

Certificate Collections [page 171]
Server-Side Data Encryption [page 134]
SAP HANA Security Guide

11.2 Certificate Collections

A certificate collection (also referred to as a personal security environment 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.

> **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.

Certificate collections uniquely serve one of the following purposes in the database in which they exist:

- User authentication based on:
  - SAML assertions
  - X.509 certificates
  - Logon and assertion tickets
- Client-server communication over JDBC/ODBC secured using the Secure Sockets Layer (SSL) protocol

- Database replication for multitenant database containers

Only one certificate collection may serve one of these purposes at any given time.

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.

Certificates used for server authentication, that is certificates that include the private key of the server, need only be assigned to the certificate collection used for secure client-server communication.

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 SSL is being enforced for all client connections and the certificate collection used for SSL is deleted, no new client connections to the database can be opened.

Related Information

SAP HANA Authentication and Single Sign-On [page 74]
Secure Communication Between SAP HANA and JDBC/ODBC Clients [page 38]
SAP HANA Security Guide

11.3 SQL Statements and Authorization for In-Database Certificate Management

All administration tasks related to in-database certificate management can be performed using SQL.

The following table lists the SQL statements for creating and managing certificates and certificate collections in the SAP HANA database, including the required authorization for each task.

> ℹ Note
>
> Certificate collections are referred to as personal security environments (PSEs) in back-end terminology.

Table 63:

To...	Execute the Statement...	With the Authorization...
See certificates in the in-database certificate store	`SELECT * FROM CERTIFICATES` **Note** You can also view certificates using the *Certificate Store* app of the SAP HANA cockpit.	System privilege CERTIFICATE ADMIN or TRUST ADMIN If you have object privilege ALTER on a certificate collection, you'll also be able to see the certificates used in this collection.
See which certificates are used in a certificate collection	`SELECT * FROM PSE_CERTIFICATES` **Note** You can also see this information in the *Certificate Store* app of the SAP HANA cockpit.	Object privilege ALTER, DROP, or REFERENCES on the certificate collection
Add a certificate to the in-database certificate store	`CREATE CERTIFICATE FROM <certificate_content> [ COMMENT <comment> ]`	System privilege CERTIFICATE ADMIN
Delete a certificate from the in-database certificate **Note** If the certificate has already been added to a certificate collection, it can't be deleted.	`DROP CERTIFICATE <certificate_id>`	System privilege CERTIFICATE ADMIN
View certificate collections in the database, including the certificates they contain	`SELECT * FROM PSE_CERTIFICATES` **Note** You can also view certificate collections using the *Certificate Collection* app of the SAP HANA cockpit.	System privilege CATALOG READ and either TRUST ADMIN, USER ADMIN, or SSL ADMIN **Note** If you own a certificate collection or you have the object privilege ALTER, DROP, or REFERENCES on a certificate collection, you'll be able to see it without the above privileges.
Create a certificate collection	`CREATE PSE <PSE_name>`	System privilege TRUST ADMIN
Add a public-key certificate to a certificate collection	`ALTER PSE <PSE_name> ADD CERTIFICATE <certificate_id>`	• Nothing if you're the owner of the certificate collection • Object privilege ALTER on the certificate collection if you're not the owner

To...	Execute the Statement...	With the Authorization...
Remove a public-key certificate from a certificate collection **Note** If the purpose of the certificate collection already been set, then system privilege USER ADMIN or SSL ADMIN is additionally required depending on whether the purpose is user authentication or secure communication.	`ALTER PSE <PSE_name> DROP CERTIFICATE <certificate_id>`	
Add a private key to a certificate collection	`ALTER PSE <PSE_name> SET OWN CERTIFICATE <certificate_content>`	• Nothing if you're the owner of the certificate collection • Object privilege ALTER on the certificate collection if you're not the owner
Set the purpose of a certificate collection **Note** If the purpose of the PSE is SSL, then it must already have a private key added.	`SET PSE <PSE_name> PURPOSE <PSE_purpose>` The following PSE purposes are possible: • SAML • SAP LOGON • X509 • SSL	• System privilege USER ADMIN or SSL ADMIN if you're the owner of the certificate collection USER ADMIN is needed if the purpose of the certificate collection is user authentication (SAML, X.509, or logon tickets), and SSL ADMIN is required if the purpose is secure client-server communication (SSL) • Object privilege REFERENCES on the certificate collection and system privilege USER ADMIN or SSL ADMIN if you're not the owner of the certificate collection
Unset the purpose of a certificate collection	`UNSET PSE <PSE_name> PURPOSE <PSE_purpose>`	• System privilege SSL ADMIN if the purpose is secure client-server communication (SSL) • System privilege USER ADMIN for all other purposes
Delete a certificate collection **Note** If the certificate collection has already been assigned a purpose, it can't be deleted.	`DROP PSE <PSE_name>`	• Nothing, if you're the owner of the certificate collection • Object privilege DROP on the certificate collection, if you're not the owner

Related Information

SAP HANA SQL and System Views Reference

12 Security Risks of Trace and Dump Files

In exceptional situations, the data output in trace and dump files may expose certain security-relevant data.

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`.

Related Information

SAP HANA Administration Guide

13 Security for SAP HANA Extended Application Services, Advanced Model

As an application platform, SAP HANA extended application services, advanced model, provides a comprehensive runtime 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.

This section of the SAP HANA Security Guide describes the security aspects of the SAP HANA XS advanced server infrastructure and covers the following main areas:

- The technical components and communication paths used by the SAP HANA XS advanced server
- User administration and authentication
- Authorization concepts such as organization and spaces, scopes and role collections, and the Controller role model
- Communication paths used by the SAP HANA XS advanced server infrastructure and the security mechanisms that apply
- Critical data that is managed by the SAP HANA XS advanced model infrastructure and the security mechanisms that apply
- Security aspects involved throughout the most widely-used processes within the SAP HANA XS, advanced model
- Audit log files that contain security-relevant information, so you can reproduce activities if a security breach does occur
- The development environment, SAP Web IDE for SAP HANA

> **Note**
>
> SAP HANA XS advanced is fully based on the SAP HANA platform. Therefore, 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 runtime 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.

Important SAP Notes

The following table lists important SAP Notes that apply to the security of SAP HANA XS advanced:

Table 64:

Title	SAP Note	Comment
Providing SSL certificates for domains defined in SAP HANA extended application services, advanced model	2243019	• How to upload certificates for domains defined in SAP HANA XS advanced • How to upload certificates for domains defined in SAP HANA XS advanced • How to upload certificates for domains defined in SAP HANA XS advanced
Domains and routing configuration for SAP HANA extended application services, advanced model	2245631	• How to enable hostname routing for SAP HANA XS advanced • How to expose only a single port for all applications and services
Enabling JDBC SSL encryption for SAP HANA extended application services, advanced model	2300943	How to enable JDBC SSL encryption for SAP HANA XS advanced applications or services

For a complete list of additional security-relevant SAP Hot News and SAP Notes, see also SAP Service Marketplace at http://service.sap.com/securitynotes.

13.1 Technical System Landscape of SAP HANA XS Advanced

SAP HANA extended application services, advanced model (XS advanced or XSA 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.

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.

Overview

As illustrated in the following diagram, the basic system architecture has a classic 3-tier approach:

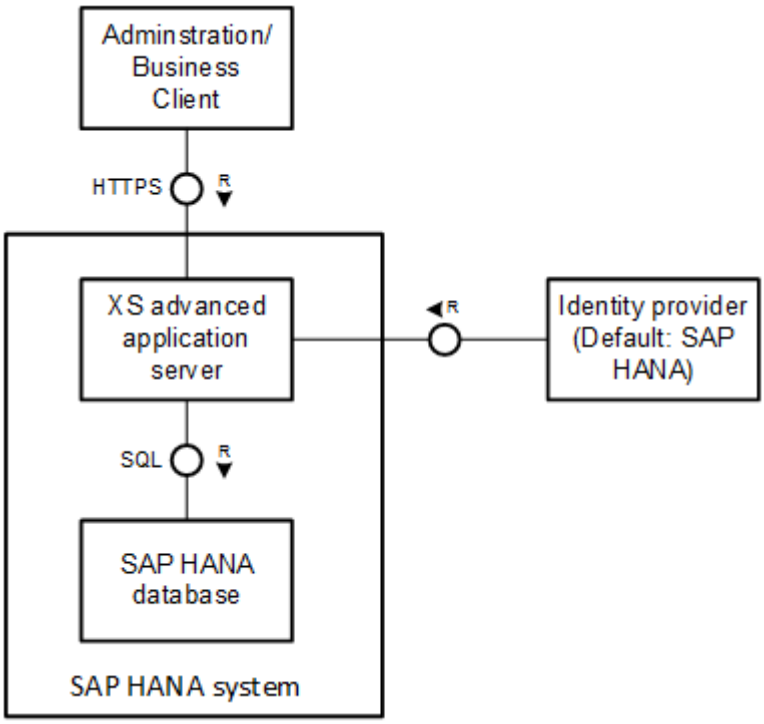


Figure 14: 3-Tier Architecture of SAP HANA with XSA

First, 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 purposes as described in SAP HANA Security Guide.

> ### ⚠ Caution
>
> As the XS advanced application server is based on the SAP HANA database, security-related configuration settings as described in the SAP HANA Security Guide 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.

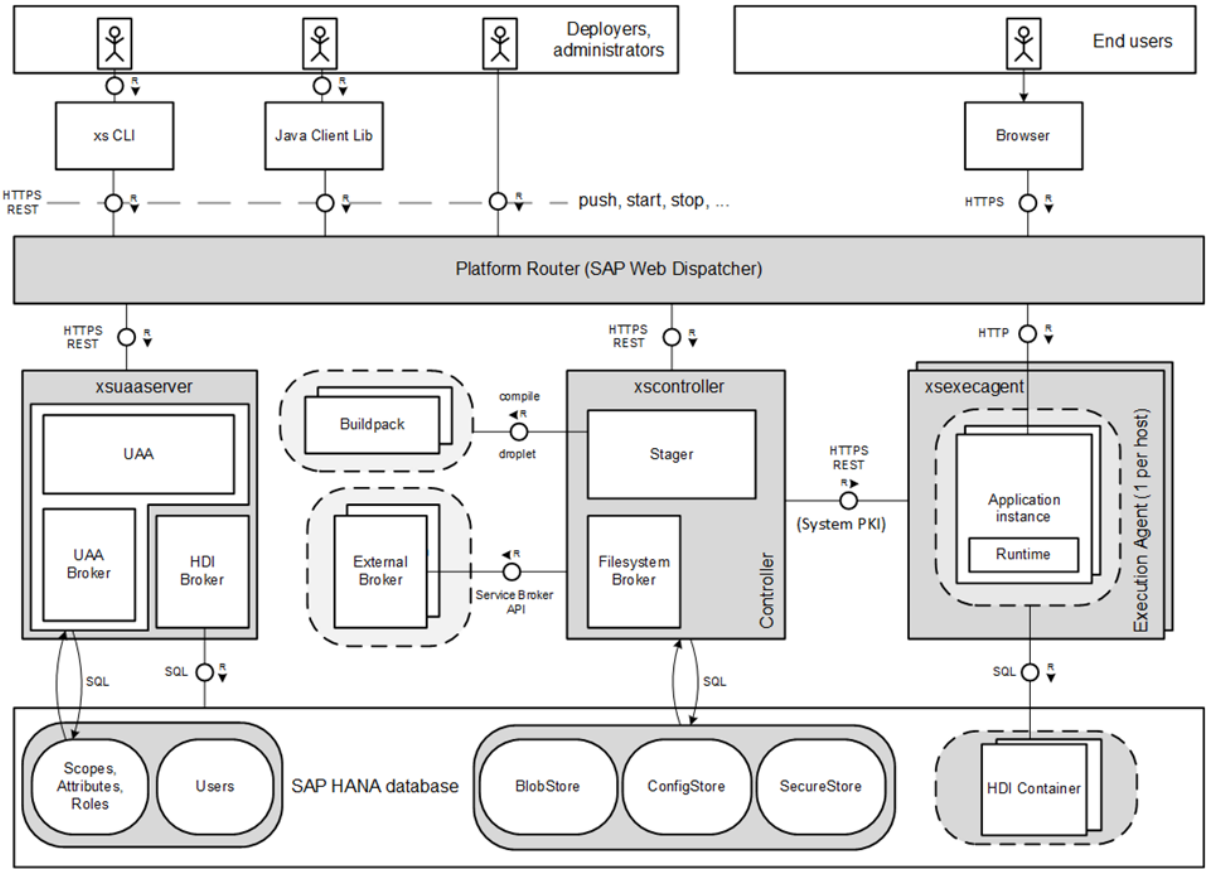


Figure 15: Technical System Landscape of XSA Application Server

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 HDI 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 keep the number of people who own its credentials as small as possible.

Related Information

Application Server Components [page 181]

13.1.1 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 XSA 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 XSA 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 (HDI Broker), user authorization (UAA Broker) and file storage (FileSystem 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 lose reachability 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 XSA 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 HDI Broker. To access HDI containers, technical SAP HANA users are created and the bound applications receive the corresponding credentials. The HDI Broker runs on a dedicated HTTP server.

Related Information

SAP HANA Network and Communication Security [page 29]
Organizations and Spaces [page 194]
User Authentication [page 192]

13.1.2 Users and Clients

All end users that access XS advanced application server components or applications are called XSA users.

We can distinguish between three different types of XSA 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 XSA 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 XSA 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 184]

13.2 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 184]
Predefined XSA Users [page 186]
Predefined Database Roles for XSA [page 190]
User Authentication [page 192]
User Administration Tools [page 193]

13.2.1 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 integration of an external identity provider (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. However, the XSA platform uses the underlying SAP HANA user store as IdP by default.

The **User Administration and Authentication service** (UAA) represents the central platform service for user management and authentication, as depicted in the following diagram:

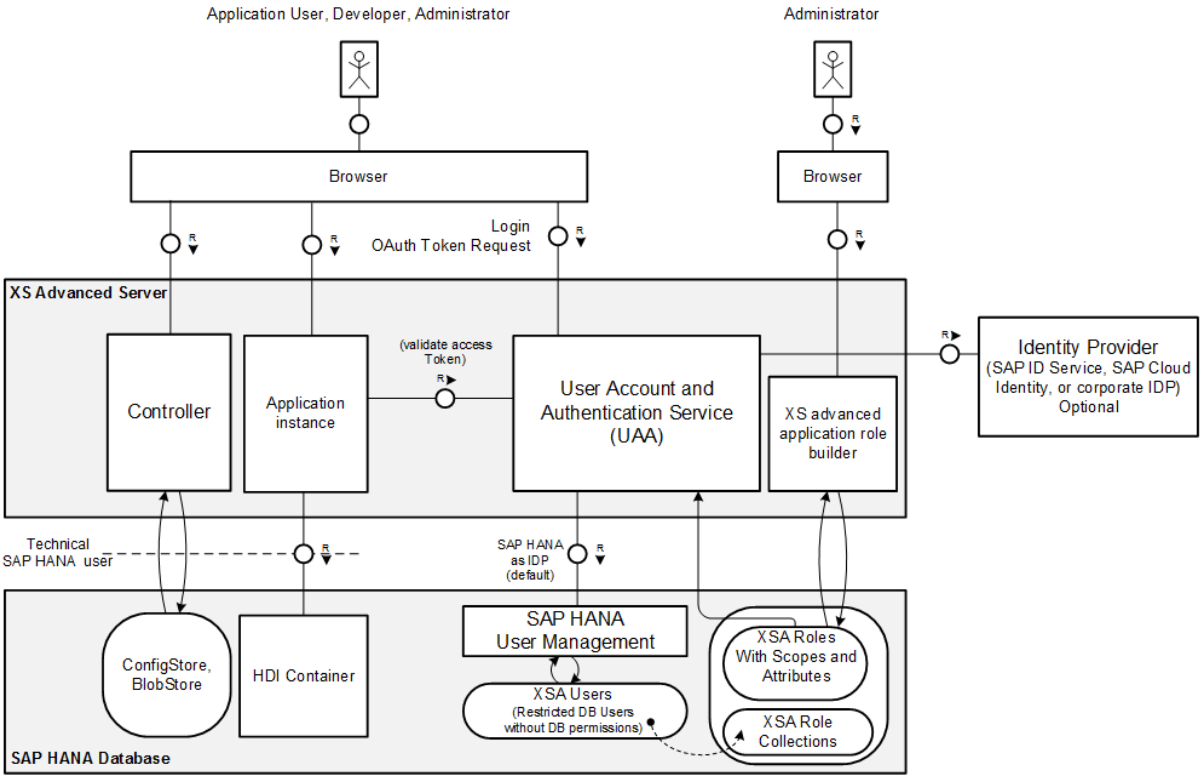


Figure 16: User Administration and Authentication Service (UAA)

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 *XSA User Authentication*.

XSA User Categories

The UAA provides the authentication endpoint for individual end users who need to interact with SAP HANA XS advanced or with applications running on top of it. Such users are referred to simply as **XSA users** and might have following responsibilities:

- **Application or business users** interact with application instances hosted on the server (for example employees, customers and so on)
- **System users who can be categorized into the following groups**:
  - Administrators who manage the configuration of application server components, in particular the Controller
  - Developers who develop, deploy, and maintain applications on the server

XSA users access the back-end instances typically through end-user interfaces such as web browsers or command-line tools. Unlike technical users, they can be additionally identified by personal data such as name, e-mail address, and so on. As the same identity provider is the basis for all of XSA users, an application user may also be granted developer privileges and the other way around.

XSA 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 XSA user has sufficient privileges. Decoupling XSA users from technical users is the precondition for leveraging external IdPs, even though XSA 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.

Operating system (OS) users also play an important role. For instance the `<sid>`adm user is created during the installation process with super user privileges. All platform services of the SAP HANA system (the application server included) run using this OS user. Therefore, `<sid>`adm is not limited in any way and needs to be handled with special care.

Related Information

User Authentication [page 192]

13.2.2 Predefined XSA Users

After the installation of the XS advanced application server, a minimum set of various users is available to operate the system.

The system's super user (`<sid>`adm) needs to be available in order to manage the lifecycle of the system. Similarly, an administrative XSA system user (`XS_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 (see *Organizations and Spaces*) and so on. Technical SAP HANA users are created during installation for all server components that need to persist data in SAP HANA schemas.

Predefined XSA System Users

The table below lists the predefined XSA system users that are necessary for operating the XS advanced application server. First, an administrative Controller user named `XS_ADMIN` is required to (re)configure the application server at a global level. None-administrative Controller users are not allowed to upload custom certificates, add custom buildpacks, or register platform service URLs. Whereas the credentials for the technical HDI Broker and UAA Broker users are generated automatically during installation, the `XS_ADMIN` user is created interactively with a user-defined password. Being a first-level administrator user with irrevocable privileges, the `XS_ADMIN` has unlimited access to the Controller and therefore needs to be handled carefully.

➡ Recommendation

- Keep the number of people with `XS_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 `XS_ADMIN`.
- Change the `XS_ADMIN` password at regular intervals.

Table 65:

User ID	Type	Description
`XS_ADMIN`	XSA user	• Administrative user for the XS advanced application server • Has unlimited access to Controller API
`HDI_BROKER_CONTROLLER`	Technical user	User for HDI Broker API
`sap_sb`	Technical user	User for UAA Broker API

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. To accomplish this, a dedicated technical SAP HANA user is generated for each such schema and the credentials are passed to the agent. As the management of these technical users is done by the infrastructure, end users typically do not interact with these users.

Table 66:

User ID	Type	Description
`SYS_XS_RUNTIME`	Technical SAP HANA user	Owns the Controller's SAP HANA schema containing BlobStore, ConfigStore and SecureStore
`SYS_XS_UAA`	Technical SAP HANA user	Owns the UAA's SAP HANA schema for user management
`SYS_XS_UAA_SEC`	Technical SAP HANA user	Owns the UAA's SAP HANA secure store for the user credentials
`SYS_XS_HANA_BROKER`	Technical SAP HANA user	Owns the HDI Broker's SAP HANA schema
`SYS_XS_SBSS`	Technical SAP HANA user	Owns SAP HANA schema containing procedures to generate user passwords in a secure manner; used by the HDI Broker
`_SYS_DI`	Technical SAP HANA user	Owns all HDI SQL-based APIs, for example all API procedures in the `_SYS_DI` schema and API procedures in containers

User ID	Type	Description
_SYS_DI_*_CATALOG	Technical SAP HANA user	Technical users used by the HDI to access database system catalog tables and views
_SYS_DI_SU	Technical SAP HANA user	Technical superuser of the HDI created at installation time
_SYS_DI_TO	Technical SAP HANA user	Owns transaction and connections of all internal HDI transactions

Technical Users for HDI Schema-Based Containers

The deployment of database objects with HDI is based on a container model where each container corresponds 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.

For example, for a container S, HDI will create the following users:

- User S: Owner of the container schema S
- User S#DI: Owner of the schema S#DI containing metadata and deployment APIs
- User S#OO: Owner of database objects in schema S
- Users _DI#S#METADATA_COM_SAP_HANA_DI_<metadata>: Owners of schemas containing build plug-in metadata

These technical users are used internally by HDI only. They are created as restricted database users who do not have any privileges by default (not even the role PUBLIC). They cannot be used to log on to the database.

For more information, see *Maintaining HDI Containers* in the *SAP HANA Developer Guide (For SAP HANA XS Advanced Model)*.

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, being installed with the XS advanced runtime 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 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)*.

Predefined OS Users

Ultimately all platform services are made up of operating system artifacts such as OS processes, network sockets, and file storages. As operating systems come with their own user management, these artifacts are necessarily owned by OS users. Consequently, the XS advanced application server can't be run without at least one OS user, although dedicated XSA users are able to perform a majority of the operational tasks.

The installation procedure creates the super OS user for the entire SAP HANA system, `<sid>adm`. Being the owner of all OS processes, this administrative 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
- Stager spawns processes running buildpacks during 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*.

The following table summarizes the OS users that are available immediately after installation:

Table 67:

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 181]
Organizations and Spaces [page 194]
SAP HANA Developer Guide (For SAP HANA XS Advanced Model)

13.2.3 Predefined Database Roles for XSA

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.

Table 68:

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 DI 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.

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.

13.2.4 User Authentication

XSA 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, XSA users managed by the UAA are authenticated on the basis of the standardized OAuth2 protocol as depicted in the following sequence diagram:

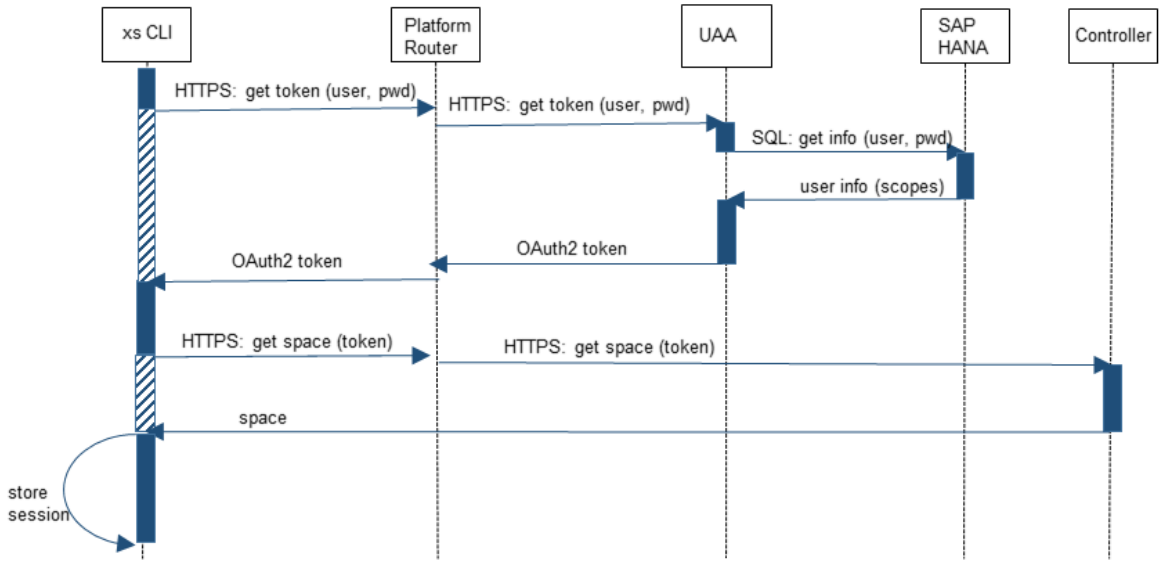


Figure 17: OAuth2 Authentication Sequence Diagram

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.

13.2.5 User Administration Tools

XSA users managed by the User Administration 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 users management 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 studio
- SAP NetWeaver Identity Management

In addition, the SAP HANA XS advanced platform comes with additional applications called the *XS Advanced Administration and Monitoring Tools*. These tools provide a comprehensive user interface for performing all tasks related to user management in XSA. For more information about these tools, see *Maintaining the SAP HANA XS Advanced Model Run Time* in the SAP HANA Administration Guide.

Related Information

SAP HANA Administration Guide

13.3 Authorization in SAP HANA XS Advanced

XSA users can access system services or interact with hosted applications. Since you don't want all XSA 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 194]
Scopes, Attributes, and Role Collections [page 197]
Controller Role Model [page 200]
Authorization Management Tools [page 203]

13.3.1 Organizations and Spaces

Resources from different applications may be isolated by leveraging the concept of organizations and spaces.

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. 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:

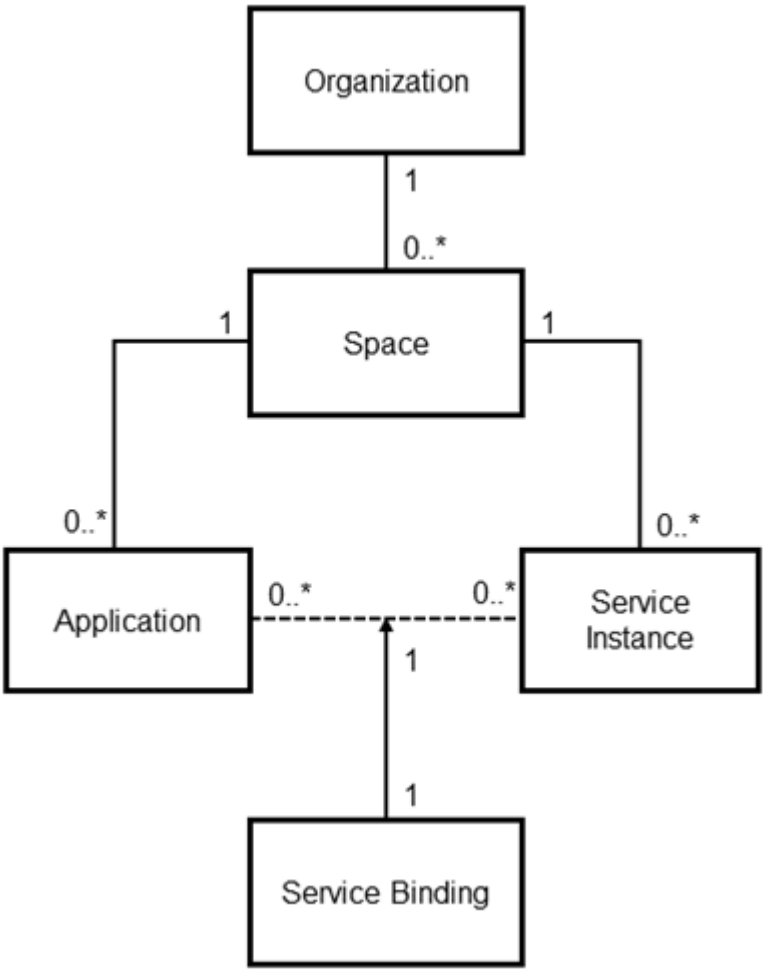


Figure 18: Organizations and Spaces

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 *Controller Role Model*.

Spaces and Operating System (OS) Users

Applications in the same space can share not only the same resources such as data storage, but their instances also run with the same OS user. Similarly, the external buildpack process that is forked by the Stager run with this user. Consequently, application instances and buildpacks in different spaces can't interfere at the OS level. This is important from a security perspective when you consider that both types of OS processes run custom code. The described behavior thus fits to the spaces' characteristic of forming a trust zone.

As a Space Manager, you may attach an available OS user to your managed space, either when the space is created or as a configuration step afterward:

```
xs create-space <name> <OS user>
xs update-space <name> <OS user>
```

The changes only have an effect on newly staged or started applications.

> **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.

By default, all spaces without an attached OS user will use the pre-installed default user `<sid>xsa`. This user is also assigned to the initial default space named `PROD`. If you don't need to separate your applications at OS level, this default user is fine to start with. But if you want to make one or more spaces using an exclusive OS user, you have to create such a user manually and it must meet the following prerequisites:

- The OS user is available on all hosts of the system that run services of the XS advanced application server, especially `xscontroller` and `xsexecagents`.
- The `<sid>adm` user needs to be set in the OS user's `sudo` file.

The following figure shows a sample Controller model with regards to organizations, spaces, and applications.

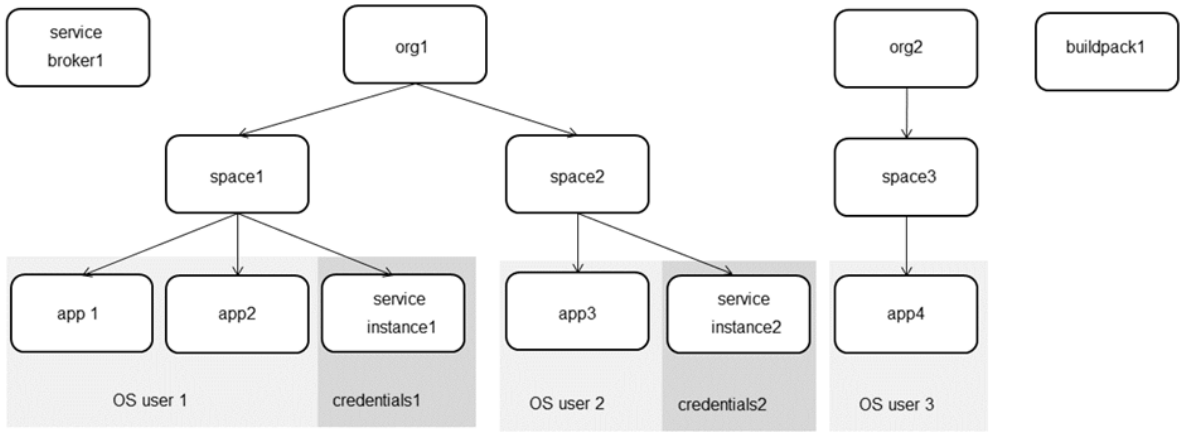


Figure 19: Example: Organizations and Spaces

In the example, organization org1 consists of two spaces that are isolated by different OS users. For each space, a service instance was created with own credentials (for example, technical SAP HANA user credentials for an SAP HANA schema in case of HDI Broker). Note that some entities like service brokers and buildpacks are created at a global model level and are shared by all spaces.

More Information

- For more information about how to create new organizations and spaces, see the SAP HANA Developer Guide for SAP HANA XS Advanced
- The section *Controller Role Model* discusses in detail how to assign Controller roles to Controller users.

Default Organizations and Spaces

The XS advanced application server comes with pre-installed applications that support the lifecycle management of MTAs:

- The `deploy-service` provides an interface to deploy MTAs in a convenient way.
- The `product-installer` and component-registry help to install and update applications.

These applications can be seen as part of the system and are therefore deployed to a separate space named `SAP` during installation. This space is an appropriate location for other system-relevant applications from SAP such as the optional Admin UI or the *Application Role Builder* tool. After system setup, there is a second space named `PROD` that you can use to deploy your custom applications. These two spaces reside in the same initial organization, but they are isolated by different OS users as shown in the following table:

Table 69:

Organization	Space	Space OS User	Content
Initial organization	`SAP`	`sap<sid>xsa`	• deployer-service • product Installer • component-registry
Initial organization	PROD	`<sid>xsa`	Empty

Related Information

Controller Role Model [page 200]
SAP HANA Developer Guide for SAP HANA XS Advanced Model

13.3.2 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 XSA 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 UAA's login page 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 define a **role collection**. Provided he or she has the proper privileges, an XSA user may 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 his or her assigned role collections. On the basis of these scopes and attributes, an application can do an authorization check after having decoded the access token.

But where do the role collections actually come from?

MTA-specific role collections are design-time artifacts. The scopes and attributes they are based on are specified in the `xs-security.json` file, which is evaluated during the deployment of the MTA. The resulting role templates can be instantiated to roles and then grouped into role collections in the *Application Role Builder* tool. Finally, XSA users are assigned to role collections in the *User Management* tool. For more information about how to handle application role collections, see the SAP HANA Developer Guide for SAP HANA XS Advanced. XSA users that need privileges to interact with system components (for instance 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*.

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 instances 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 and so on) with read or write permission. Therefore, the Controller defines three different scopes to cover these basic use cases:

- `cloud_controller.admin` (unlimited access)
- `cloud_controller.write` (write access)
- `cloud_controller.read` (read access)

In line with the authorization concept described above, these scopes are combined into three different Controller role collections:

- `XS_CONTROLLER_ADMIN` (`cloud_controller.admin` + `cloud_controller.write` + `cloud_controller.read`)
- `XS_CONTROLLER_USER` (`cloud_controller.write` + `cloud_controller.read`)
- `XS_CONTROLLER_AUDITOR` (`cloud_controller.read`)

To overcome the bootstrap problem when an XS advanced application server is installed, a single administrative Controller user (named `XSA_ADMIN` by default) is created. This user has the Controller role collection `XS_CONTROLLER_ADMIN`, which comprises all three Controller scopes. This means that the `XSA_ADMIN` can use the Controller without any restrictions and is in a position to do the initial setup of the model, that is appointing at least one Org Manager 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 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.

Table 70:

Role Collection	Application	Scope(s)	Permitted Operations
`XS_CONTROLLER_ADMIN`	Controller	• `cloud_controller.admin` • `cloud_controller.write` • `cloud_controller.read`	Unlimited access to Controller
`XS_CONTROLLER_USER`	Controller	• `cloud_controller.write` • `cloud_controller.read`	Read or write Controller resources (global resources excluded from modifications)
`XS_CONTROLLER_AUDITOR`	Controller	• `cloud_controller.read`	Read access to Controller resources

Related Information

Authorization Management Tools [page 203]
Controller Role Model [page 200]

13.3.3 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 XSA users that should be allowed to manage the space's resources collectively. Which resources can be assigned to a space?

Each Controller resource has either 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.

> **Note**
>
> "Modify" always means create or update.

Table 71:

Controller Role	Resource Scope	Permitted Operations Within Scope
`OrgManager`	Organization	• Modify spaces • Modify domains • View organization resource (including credentials) • Grant any Controller role to other user
`OrgAuditor`	Organization	• View organization resource (excluding credentials)

Controller Role	Resource Scope	Permitted Operations Within Scope
`SpaceManager`	Space	• Grant `SpaceManager`, `SpaceDeveloper`, `SpaceAuditor` to other user
`SpaceDeveloper`	Space	• Modify space resource
`SpaceAuditor`	Space	• View space resource (excluding credentials)

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.

Table 72:

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`

Controller Resource	Resource Scope	xs Commands	Minimum Authorization
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`
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

SAP HANA Developer Guide (For SAP HANA XS Advanced Model)

13.3.4 Authorization Management Tools

Various command line and UI tools can be used for user and authorization management.

XS Advanced Administration and Monitoring Tools

The administration and monitoring tools are XS Advanced applications that are deployed with the platform. Among other things, they provide a comfortable way to handle the following tasks:

Table 73:

Task	Tool
Create or delete users	*User Management*
Assign existing role-collections to users	*Application Role Builder*
Create custom role-collections based on deployed role-collection templates	*Application Role Builder*
Manage organizations and spaces	*Organization and Space Management*
Assign Controller roles (SpaceDeveloper, SpaceManager, etc.) to Controller users	*User Management*

> **Note**
>
> Users who need full access to the tools *User Management* and *Application Role Builder* 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 *Maintaining the SAP HANA XS Advanced Model Run Time* in the SAP HANA Administration Guide.

hdbsql, SAP HANA Studio

If the UAA's identity provider is SAP HANA itself (default), the established SAP HANA user-management tools can be used to create XSA 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 XSA 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 XSA 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:

Table 74:

xs Command	Description
`users`	Lists all Controller Users with a Controller role
`org-users`	Shows all Controller users with a role in a specific organization
`set-org-role`	Assigns an organization role to a Controller user
`unset-org-role`	Revokes an organization role from a Controller user
`space-users`	Shows all Controller users with a role in a specific space
`set-space-role`	Assigns a space role to a Controller user
`unset-org-role`	Revokes a space role from a Controller user
`purge-users`	Removes all Controller users that are not known to UAA anymore

> **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

SAP HANA Administration Guide

SAP HANA Developer Guide (For SAP HANA XS Advanced Model)

13.4 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 205]
Public Endpoints [page 207]
Single-Host Scenario [page 208]
Multiple-Host Scenario [page 209]
Certificate Management [page 211]

13.4.1 Security Areas

Three different 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 **XSA Public** area, the **XSA System** area, and the **SAP HANA JDBC** area.

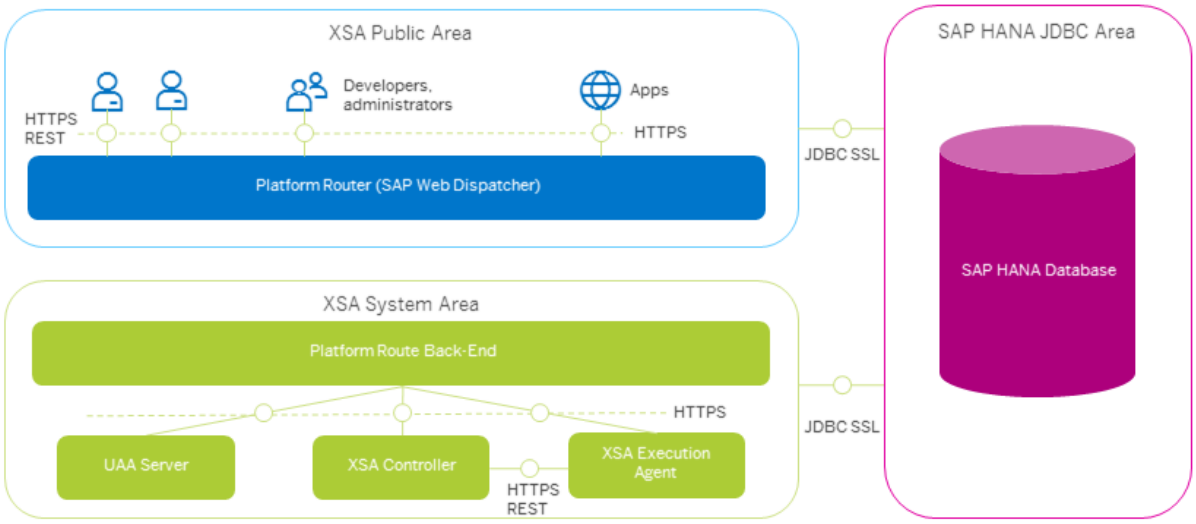


Figure 20: XSA Security Areas

The XSA Public area is managed by an XSA 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 XSA 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 *Certificate Management*.

The XSA System area covers all internal communication channels between application server components like Controller, Execution Agents, UAA, Platform Router back-end, and so on. These channels are secured with TLS/SSL based on a system public key infrastructure (PKI), which provides mutual authentication. Note that the back-end of the Platform Router is also placed in the XSA system area so it can be managed by the XSA infrastructure for internal communication. The SSL certificates are only used for internal purposes and are never exposed to other areas, neither to applications nor to any XSA clients.

The SAP HANA JDBC area includes TLS/SSL-secured connections between the SAP HANA database and XSA applications, as well as between the database and server components.

> ⚠ Caution
>
> The JDBC connection to the SAP HANA database is **not** encrypted by default. To activate JDBC TLS/SSL, custom SSL certificates need to be configured as described in section *Certificate Management*.

Related Information

Certificate Management [page 211]

13.4.2 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.

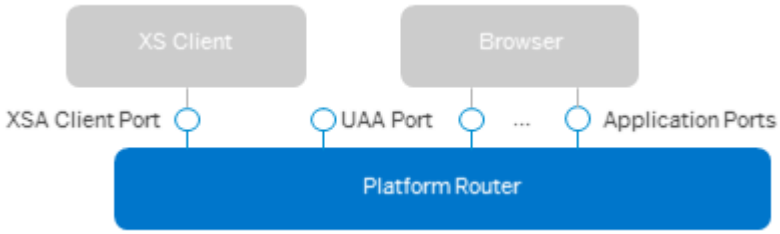


Figure 21: Port Routing

The table below shows the public ports of the Platform Router as they are exposed to clients.

Table 75:

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

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.

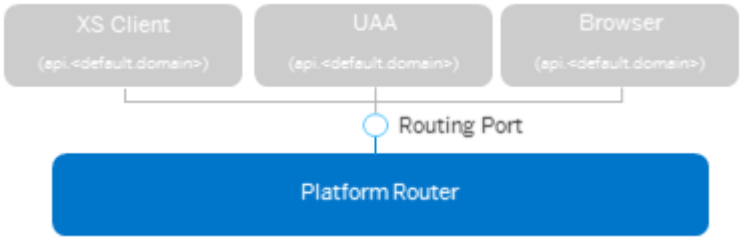


Figure 22: Routing Modes

The following table lists the URLs of system components:

Table 76:

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

13.4.3 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.

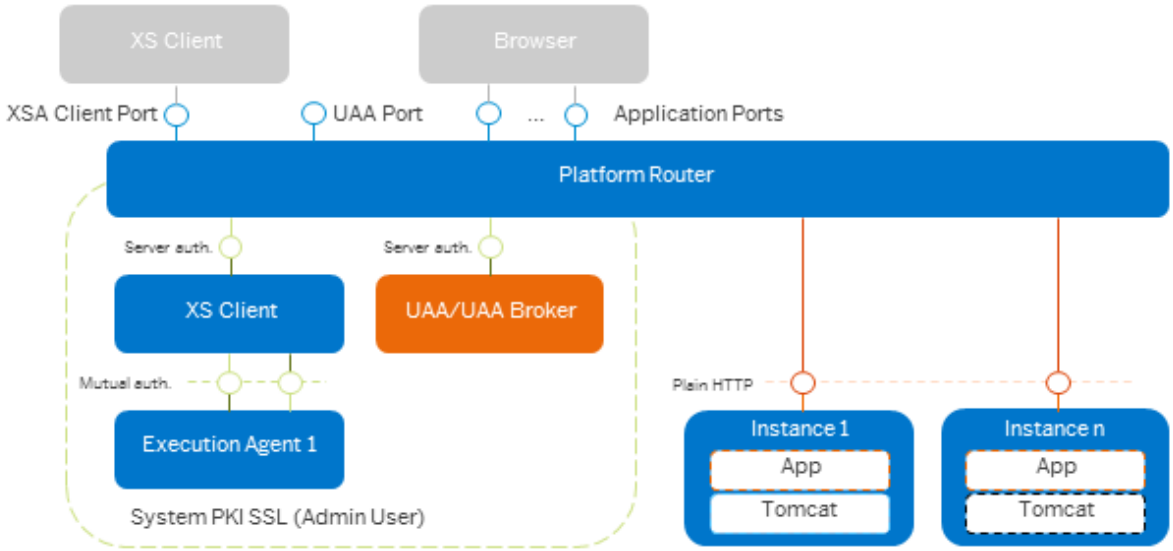


Figure 23: XSA Single-Host Scenario

Private Endpoints

Server components and application instances expose ports that are not designed for external communication. The following tables lists all private ports:

Table 77:

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)

13.4.4 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:

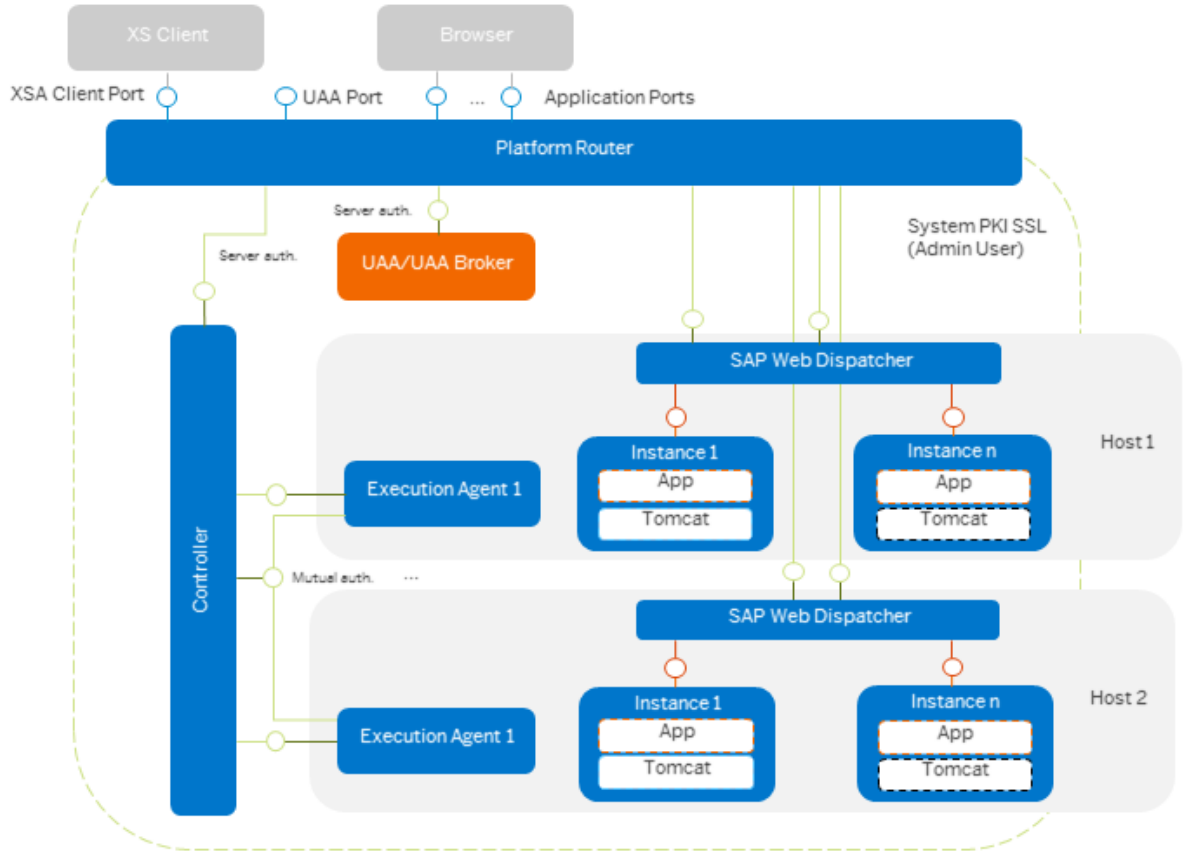


Figure 24: XSA Multiple-Host Scenario

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.

Table 78:

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)

13.4.5 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.

XSA Public Area

In the XSA Public area the XSA 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.

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 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.

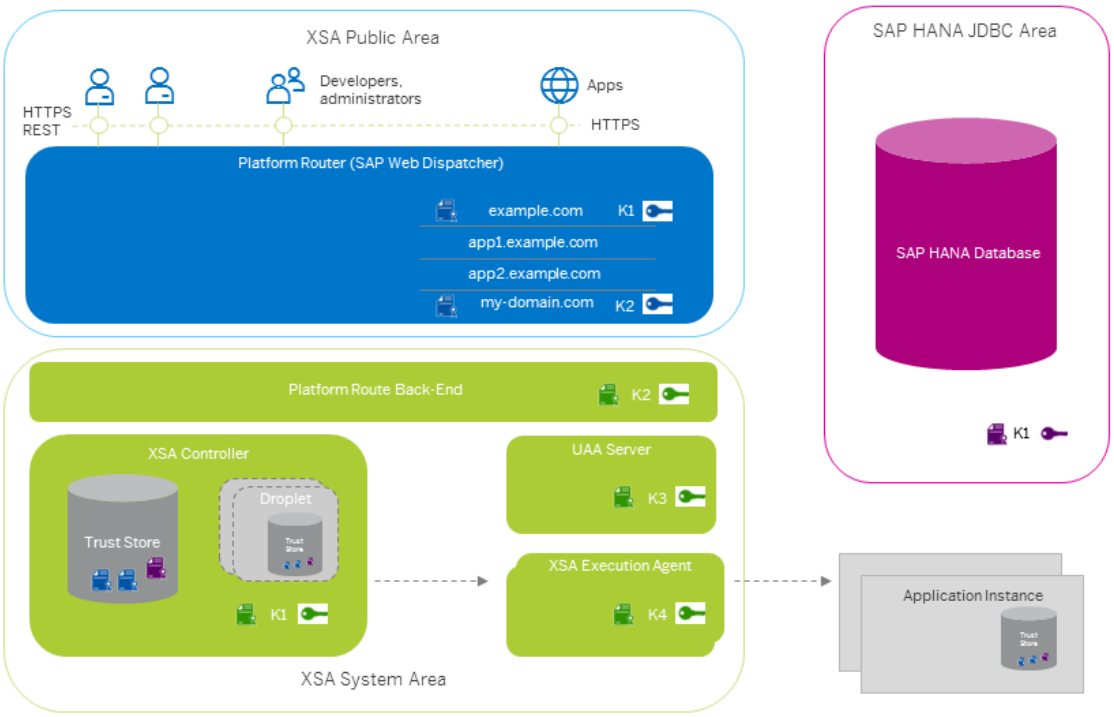


Figure 25: Certificate Management in XSA Publich Area

A security-critical scenario can arise if the certificate expires. In this case, the Controller cannot authenticate the Platform Router anymore and aborts startup. To solve this problem, the administrator has to restart the Controller with option `--reset-certificate` and a new self-signed certificate is generated.

> **Note**
>
> 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).

XSA System Area

The system administrator does not need to perform any configuration steps for the XSA 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.

XSA JDBC Area

The XSA administrator is also responsible for the certificate management in the SAP HANA JDBC area. The XS advanced platform does not encrypt the JDBC connections to SAP HANA out of the box. Therefore, custom certificates must be configured as explained in SAP Note 2300937. 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 the JDBC SSL in the platform settings and restart the system

For more information about how to set up system-wide JDBC TLS/SSL connections, see SAP Note 2300943.

Related Information

SAP HANA Developer Guide (For SAP HANA XS Advanced Model)

13.5 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 214]
Application Storage [page 215]

13.5.1 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 Controller resources that bear user credentials require special attention:

Table 79:

Resource	Field Names
Service broker	`auth_username` and `auth_password`
Service binding	`credentials`
Service key	`credentials`
User-provided service instance	`credentials`

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 HDI 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 200]

13.5.2 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 HDI Broker or the FileSystem Broker are used.

HDI Broker

The HDI 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` provides an SAP HANA schema with procedures to generate secure passwords

FileSystem Broker

When a service instance of the FileSystem 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).

13.6 Security Aspects of Data, Data Flow, and Processes

A number of representative data flows initiated on typical user interactions with the XS advanced application server are presented. The selected scenarios consider the most important security aspects of the platform.

The following table summarizes the presented scenarios including the steps that need attention from a security perspective:

Table 80:

Scenario	Description	Actions	More Information
Login with xs CLI	Start a local session with the xs CLI	• Retrieve Controller information • Retrieve JWT token from UAA • Store local session data	Scenario: Login with xs CLI [page 218]
Pushing an application with xs CLI	Deploy an application from the local file system	• Upload of application artifacts • Retrieve buildpack from Git URL • Spawn external staging process • Store compiled droplet in BlobStore • Bind application to external service broker • Transfer droplet and environment to Execution Agent • Start application instance	Scenario: Pushing an Application with xs CLI [page 220]
Application request via Browser	Process an application request with database lookup	• Access application endpoint providing a JWT token • Access the application's database schema	Scenario: Access Application Data via Browser [page 222]

The following diagram shows the sequence of passed agents, as well as the data stores for each scenario. Note that the sequence does not reflect the request flow between the agents. It rather shows the order of involved agents and stores.

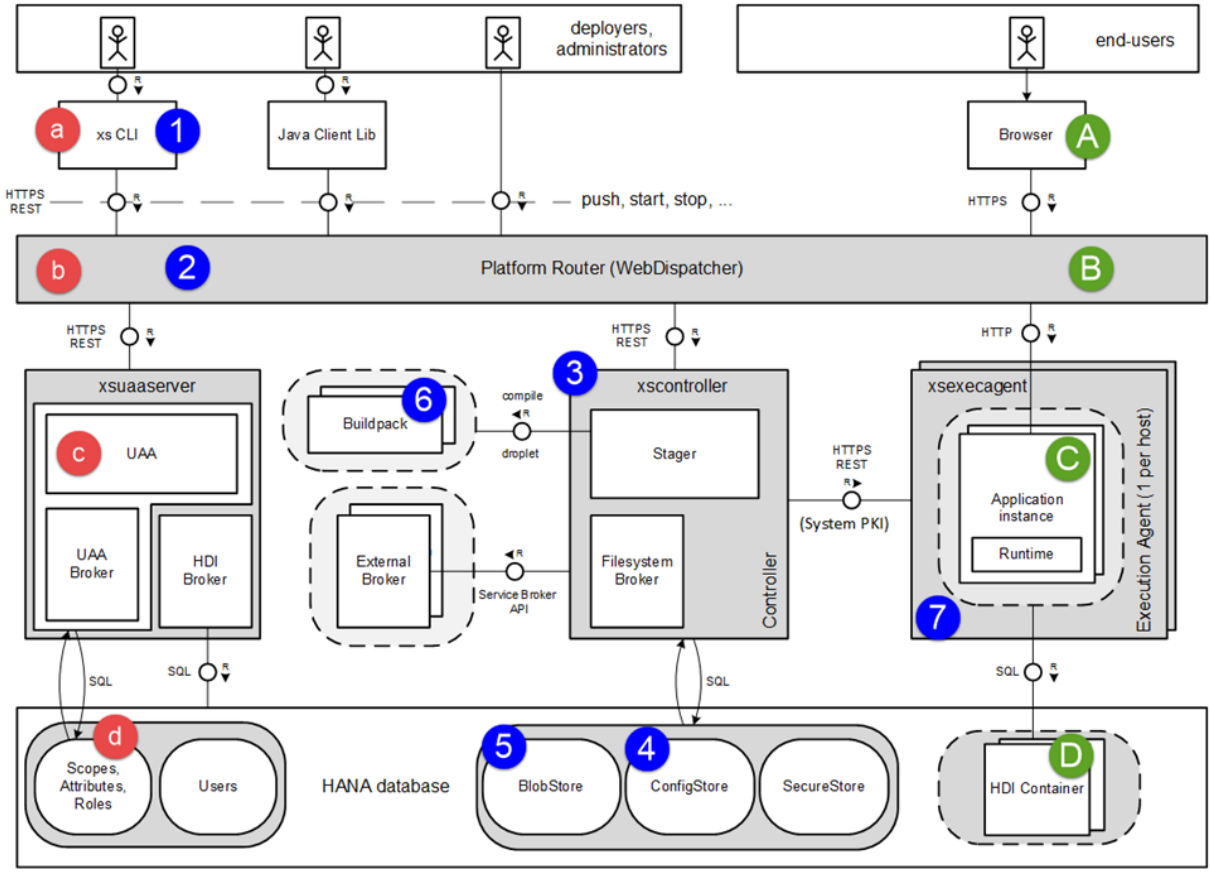


Figure 26: Scenario Flows

Table 81:

Sequence	Scenario	Agents
a-d	Login with xs CLI	xs CLI, Platform Router, UAA
1-10	Pushing an application with xs CLI	xs CLI, Platform Router, Controller, Stager, buildpack process, Service Broker, Execution Agent
A-D	Application request through browser	Browser, Platform Router, Application Instance (omitting Application Router)

All scenarios have in common that clients do not directly communicate with the back-ends. Despite this, all back-ends have an external route registered to the Platform Router. Depending on the platform's configuration, external URLs can be either port based (default), or built up in a more user-friendly with a sub-domain and domain part (name based). Generally, the endpoints that the Platform Router exposes are only reachable via the HTTPS protocol (TLS based) as described in the section *Network and Communication Security*.

> **Caution**
>
> In all scenarios, the communication between the application server and the SAP HANA database can be only be considered secure if encrypted JDBC communication to the SAP HANA database has been set up manually. For more information, see *Multiple-Host Scenario*.

Related Information

Network and Communication Security with SAP HANA XS Advanced [page 205]
Multiple-Host Scenario [page 209]

13.6.1 Scenario: Login with xs CLI

This scenario shows a basic request initiated whenever an XSA user (administrator or developer) performs an initial xs login with the command-line tool in order to start a Controller session. It is assumed that there is no previous session stored for the current OS user.

The following diagram shows the involved agents, data stores, and used transfer protocols:

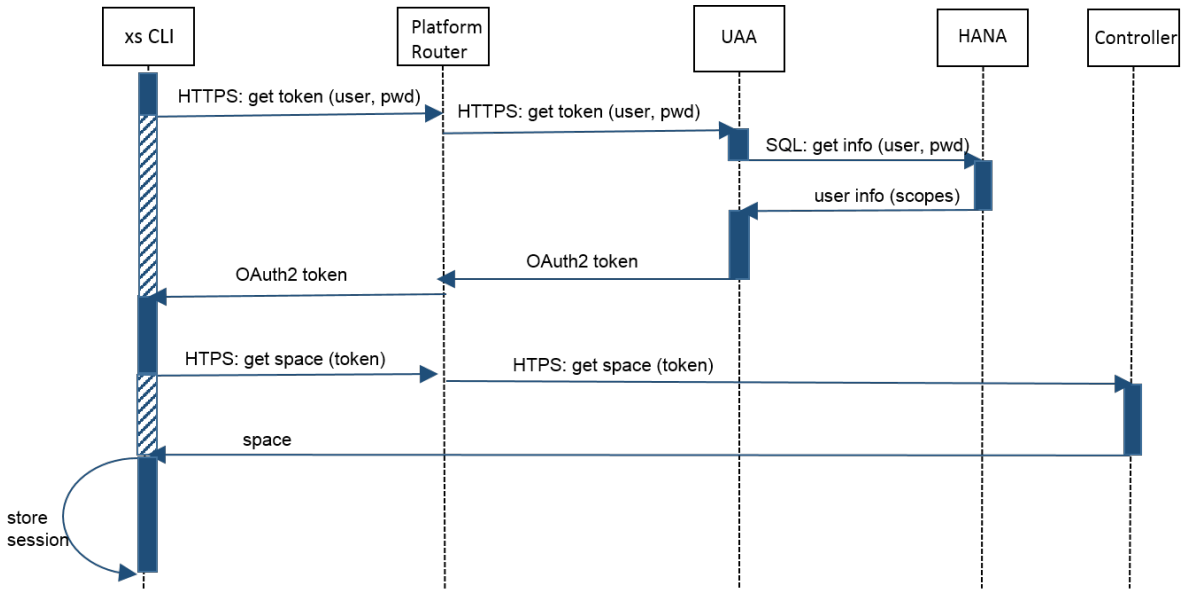


Figure 27: Scenario: Login with xs CLI

The Controller's endpoint `/info` delivers a JSON object that contains the external URL of the platform's unique UAA instance. The `/info` endpoint is the only one offered by the Controller that does not require OAuth2-based authorization. Given the external UAA URL, the xs tool sends an `/oauth/token` request to the UAA with user credentials as basic authentication. The Platform Router forwards the request to the UAA back-end.

The following table shows the security aspects to be considered for the process steps and what mechanism applies:

Table 82:

Step	Asset	Security Measure
User submits xs login with external Controller URL, target space, and user credentials	Controller user credentials (potentially from administrator user)	• User credentials are requested interactively or as data written to the xs' stdin-stream.
xs sends an `/info` request to Controller via Platform Router	not applicable (`/v2/info` is public)	Not applicable
xs sends an `/oauth/token` request to UAA via Platform Router with user credentials as basic authentication	User credentials and OAuth2 token	• Request is sent encrypted with HTTPS (TLS) • Basic authentication of XSA user
UAA reads user data from SAP HANA data store via JDBC	User information	• Encrypted JDBC SQL connection to SAP HANA
xs sends a `/spaces` request to Controller via Platform Router with Oauth2 token as authentication.	Controller model data (spaces)	• Authorization check on basis of OAuth2 token. • Request is sent encrypted with HTTPS (TLS)
xs stores session data to local file system	Session data including token	• The session data is stored in the file system of the client with exclusive access rights. By default, the session is written to the OS user's home directory.

⚠ Caution

The session data (including the OAuth2 access token) is stored to the home directory of the current OS user by default. All persons who have the credentials of this OS user will be able to reuse the session. Also, super OS (root) users will be able to take over the session, if the home directory is accessible.

➡ Recommendation

In the case of a shared OS user, we recommend that the session file is relocated to local file storage. This can be accomplished by setting the environment variable `XSCLIENT_CONTEXTFILE` to a local path. In addition, sessions should not be kept open longer than needed. Call `xs logout` to close the session as soon as possible.

13.6.2 Scenario: Pushing an Application with xs CLI

Deploying (or in Cloud Foundry terminology '"ushing") an application to the XS advanced application server is the most complex operation the Controller offers.

For simplicity, the following sequence diagram omits all requests to the Platform Router as well as read accesses to the Controller's ConfigStore. It is also assumed that the Controller session has been opened already with an appropriate `xs login` call as described in the scenario before.

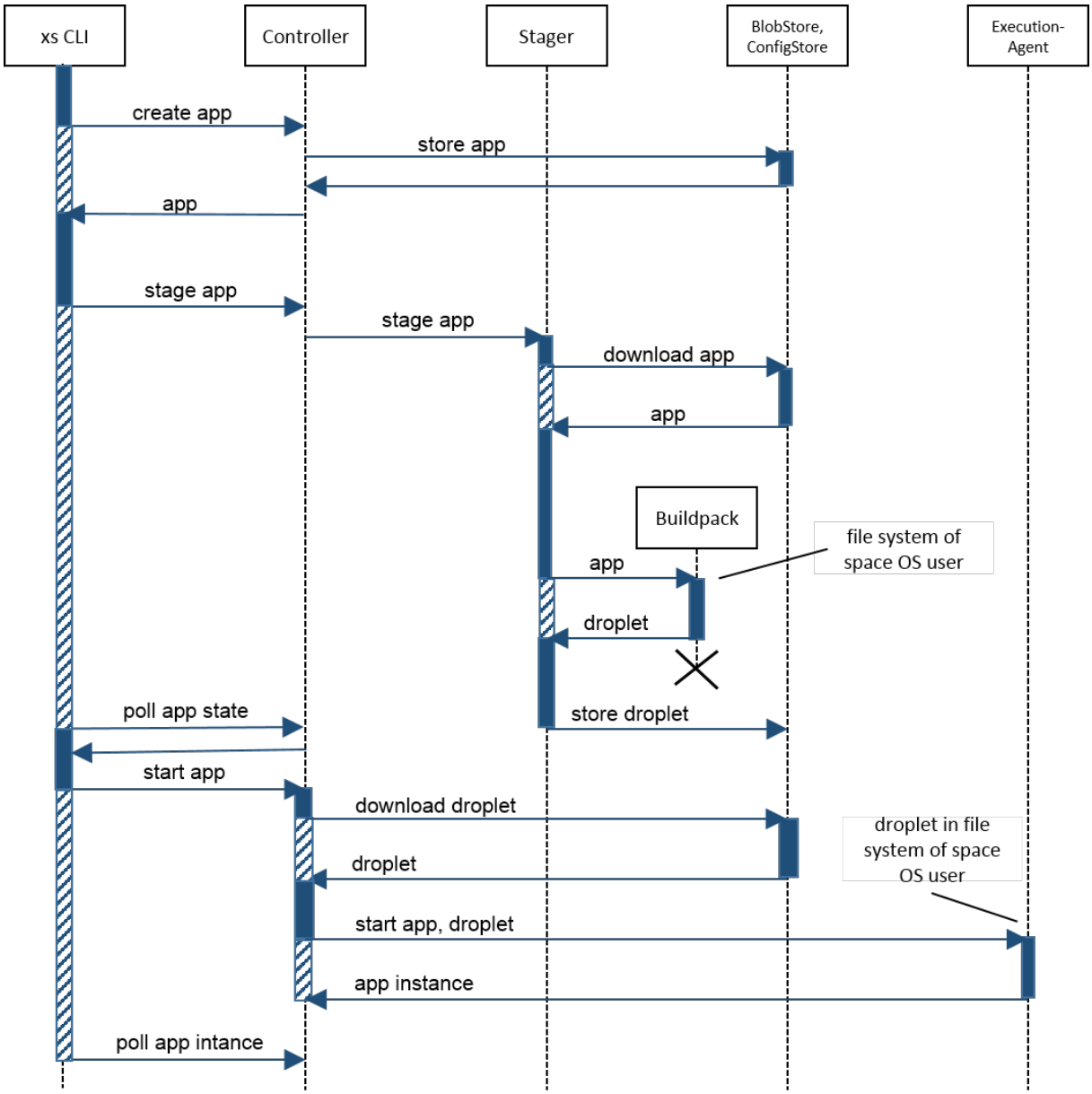


Figure 28: Scenario: Pushing an Application with xs CLI

XSA users with the privilege to perform this operation in a specific space (for example, Space Developers) submit the `xs push` command, passing the application's name together with a path to the local file system

where the application's design-time artifacts reside. From the perspective of the xs client, the push command consists of three phases:

1. Create the application resource and upload the artifacts.
2. Stage the application.
3. Start the application.

As the Controller operations for staging and starting are executed asynchronously, the client has to poll for the results before proceeding with the next step.

The Controller archives the application artifacts retrieved from the client in the BlobStore and creates a new application resource in the ConfigStore. As a result of the client's staging request, which is forwarded to the Stager, both the application files and an appropriate buildpack are restored in the file system of the OS user that is attached to the application's space.

Next, the Stager spawns a new buildpack process on behalf of the space's OS user. When the buildpack has finished with a resulting droplet in the file system, the Stager uploads the droplet to the BlobStore. As a reaction to the following start request, the Controller retrieves the droplet from the BlobStore and passes it to a proper Execution Agent, which writes the droplet to the space OS user's file system.

Finally, the Execution Agent can start a new application instance based on the droplet.

The following table shows the security aspects to be considered for the process steps and what mechanism applies:

Table 83:

Step	Asset	Security Measure
User submits `xs push` with application file directory	Application files	**⚠ Caution** Developers must ensure that the directory where the application files reside is protected from unprivileged access.
xs creates application and uploads application files	Application files, application metadata	• Encrypted SSL connection between xs client and Controller • XSA user is authorized by OAuth2 token stored in session
Controller stored application metadata in ConfigStore	Application metadata	• Encrypted JDBC connection to ConfigStore • ConfigStore in SAP HANA schema of technical SAP HANA user `SYS_XS_RUNTIME` • Credentials of bindings in SecureStore of `SYS_XS_RUNTIME`
Controller stores application files in BlobStore	Application files	• Encrypted JDBC connection to BlobStore

Step	Asset	Security Measure
Stager downloads application files from BlobStore and writes to space OS user's file system	Application files	• Encrypted JDBC connection to BlobStore • Stager directory of OS space user is created with exclusive access rights
Stager spawns buildpack process with space's OS user	XS advanced services	• Custom buildpack code runs on space's OS user and thus is separated from XS advanced services running on `<sid>adm`.
Stager stores droplet to BlobStore	Droplet	• Droplet is located in the OS user's file system not accessible for other users • Encrypted JDBC connection to BlobStore
Controller sends droplet to Execution Agent	Droplet	• Encrypted SSL connection between Controller and Execution Agent based on System PKI
Execution Agent starts the droplet	Droplet, XS advanced services	• Droplet is stored in space's OS user directory • Custom application code (droplet) runs with space's OS user and thus is separated from XS advanced services running on `<sid>adm`.

13.6.3 Scenario: Access Application Data via Browser

This scenario deals with XSA user interactions with (business) applications.

Comprehensive support is available for deploying a multi-target application (MTA) in a space using the built-in `deploy-service`. Among other things, MTAs may optionally be complemented by an Application Router

instance. This router, which is based on the node.js runtime, is preconfigured by the platform to do the routing to the different application instance back-ends, as depicted in the following diagram:

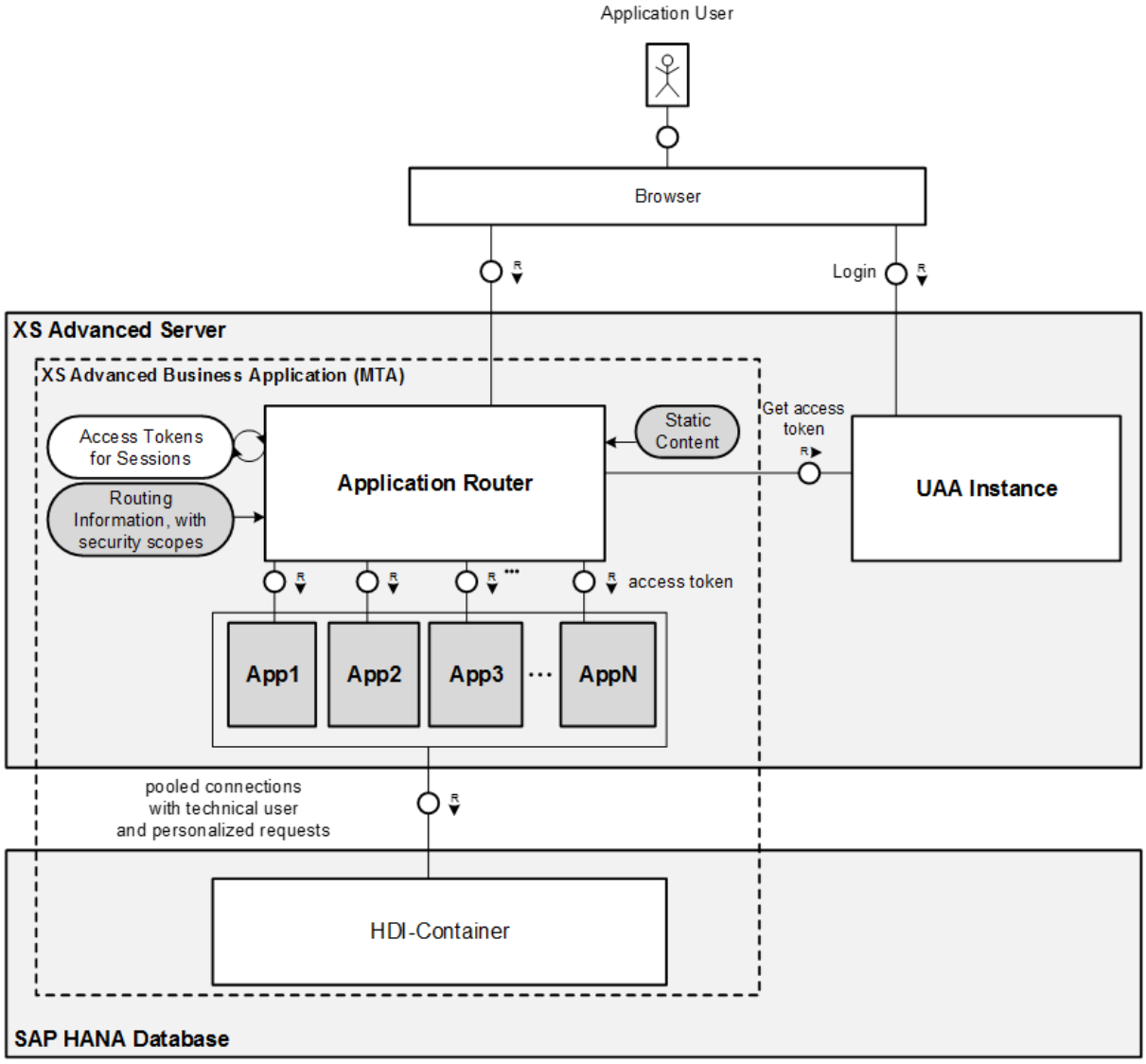


Figure 29: Application Router

The Application Router also provides a login mechanism by redirecting unauthorized web requests to the UAA's login page in order to fetch a valid OAuth2 access token. The Application Router then forwards the request along with the token to the targeted application instance in the back-end. Subsequent requests can reuse the cached token when clients pass the proper session ID in the HTTP header. The Application Router is also suitable for serving static content.

The scenario described here requires a valid session ID, in other words, the Application Router does not need to perform the redirect to the UAA in order to fetch a token but can make use of a cached token from a session before.

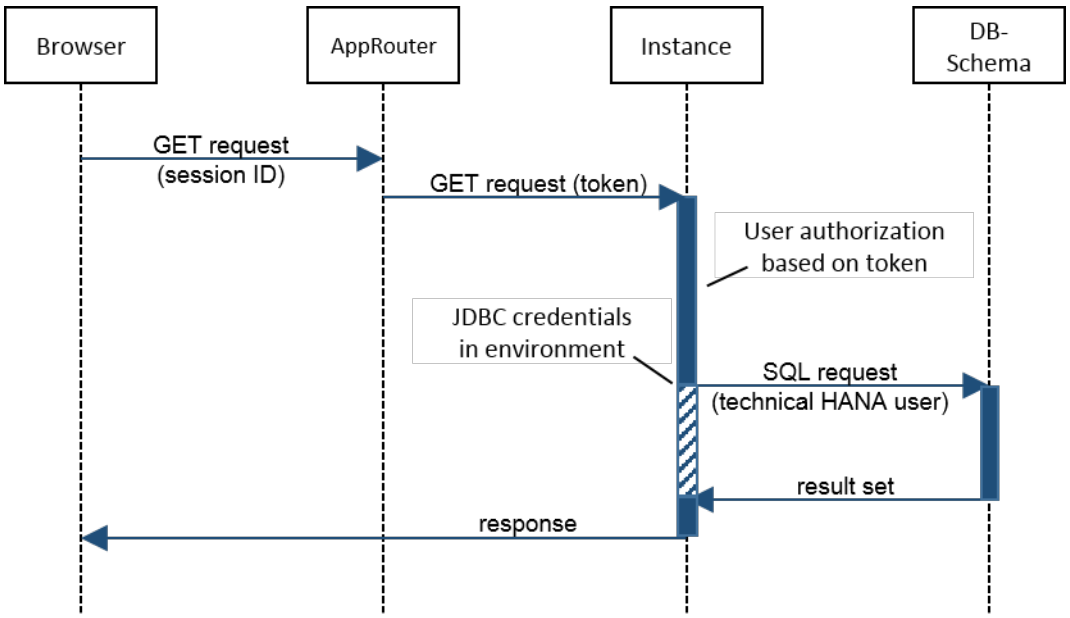


Figure 30: Scenario: Access Application Data via Browser

The sequence diagram above shows the browser request being routed to the appropriate application instance, enriched with the corresponding session token that has been resolved and stored by the Application Router before. The Application Router may perform a user authorization as a preliminary step before forwarding the request to the instance. However, the application instance itself has to decode the access token and do the authorization again, as the request could have been directly sent to its HTTP endpoint. Token validation can either be done offline or by calling the `/check/token` endpoint of the UAA (online validation). If the request is authorized, the instance issues an SQL statement to fetch data from its data storage created when the application was bound to the corresponding service (for example, an HDI container). The JDBC connection credentials (for example, for a technical SAP HANA user) are written to the process environment of the instance. The result set received is transformed into a corresponding response and sent back to the browser.

The following table shows the security aspects to be considered for the process steps and what mechanism applies:

Table 84:

Step	Asset	Security Measure
Browser submits request to Application Router with session ID.	Session ID	• Encrypted SSL connection between browser and Application Router • Session ID is assumed to be stored securely by the browser
Application Router forwards request to application instance along with the token	Access token	• Encrypted SSL connection between Application Router and SAP Web Dispatcher of host where the application instance is running • Local HTTP-communication between Web Dispatcher and application instance

Step	Asset	Security Measure
Application instance authorizes request	Not applicable	• In the case of online token validation: encrypted SSL connection between application instance and UAA
Application instance sends SQL statement to database	• JDBC credentials • Result data	• Result data is stored in the database schema of a technical database user only visible for bound application within the same space • JDBC credentials are part of the process environment of the application instance and thus are not readable for other instances that run in a different space, if the spaces are configured with different OS users • Encrypted JDBC connection to database

13.7 Security-Relevant Logging and Tracing

Auditing makes it possible to trace who has performed which kinds of operation in the XS advanced system. The written 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.

13.7.1 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

13.7.2 Audit Trails

Audit logs are written to `<sid>adm` user's tracing file system.

Table 85:

Service	Audit Log Location
Controller	`/trace/hdbxscontroller_audit.log`
Execution Agent	`/trace/hdbxsexecagent_audit.log`
UAA	`/trace/xsuaaserver.log`

Audit trails can be easily inspected in the SAP HANA studio.

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 make the Controller as well as the Execution Agent write their audit logs to syslog, add the parameter `--audit-log=SYSLOG` to the arguments of the corresponding service in the `deamon.ini` file.

For more information about how to configure syslog, refer to the documentation of your operating system.

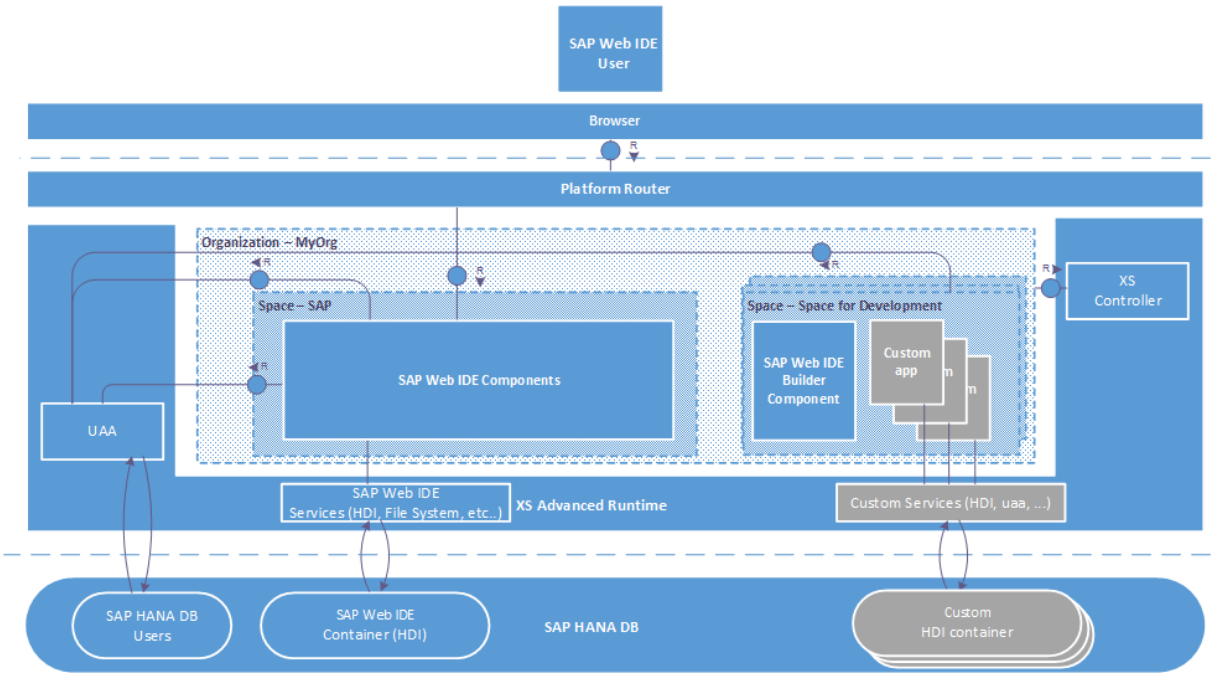
14 Security Aspects of SAP Web IDE for SAP HANA

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.



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 *SAP Web IDE for SAP HANA Installation and Upgrade Guide* in SAP Note 2304873.

> **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 *SAP Web IDE for SAP HANA Installation and Upgrade Guide*.

Related Information

Security for SAP HANA Extended Application Services, Advanced Model [page 177]
User Authorization and Authentication [page 229]
Known Security-Related Issues [page 230]

14.1 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:

- **SAP Web IDE developer** 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.
- **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 tasks that users perform, and the required roles and role collections.

Table 86:

User Type	Role/Role Collections	Description
Administrator Developer	*XS_CONTROLLER_USER* role collection	Grants read-write permissions within the assigned organization or space.
Administrator 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.
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.
Developer	A role collection containing the *WebIDE_Developer* and *xsac_hrtt_developer_template* role templates.	Enables users to develop applications using SAP Web IDE and SAP HANA Runtime Tools (HRTT).

> **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.

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.

Related Information

User Administration and Authentication in SAP HANA XS Advanced [page 184]
Security for SAP HANA Extended Application Services, Advanced Model [page 177]

14.2 Known Security-Related Issues

A list of known security-related issues of SAP Web IDE for SAP HANA.

Table 87:

Topic	Issue / Impact	Workaround
Authorization	Any SAP HANA database user can access the SAP HANA Runtime Tools application. However, the user won't be able to access the database schema information or modify any database objects without the required permissions, such as those granted by the *SpaceDeveloper* role.	No workaround is available
Backup and recovery	SAP Web IDE does not support backup and recovery for workspace content (projects, folders, files).	Use a Git repository as a backup.
File upload	No automatic virus scan and content validation of files is performed before uploading them to SAP Web IDE. Malicious content can be uploaded.	Developers should use external tools to perform a virus scan and content validation of the files before uploading them to SAP Web IDE. Alternatively, customers can use a file-based antivirus to scan the files on the disk.

15 Security for Other SAP HANA Platform Components

In addition to the SAP HANA database, the SAP HANA platform includes several other components, for example SAP HANA lifecycle manager, SAP HANA content, SAP HANA smart data access, and so on.

> **Note**
>
> This section does **not** cover security information about SAP HANA options and capabilities, which provide additional features to the SAP HANA base edition of the SAP HANA platform. For more information about the secure implementation of SAP HANA options and capabilities, see *SAP HANA Options and Capabilities* on SAP Help Portal.

SAP HANA Platform Lifecycle Management (Security) [page 232]
SAP HANA platform lifecycle management covers the tools for installing, configuring, and updating SAP HANA platform components.

SAP HANA Content (Security) [page 232]
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.

SAP HANA Smart Data Access (Security) [page 233]
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.

SAP HANA R Integration (Security) [page 234]
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.

SAP HANA Information Composer (Security) [page 235]
The SAP HANA information composer is a Web application that allows you to upload to and manipulate data on the SAP HANA database.

Related Information

SAP HANA Options and Capabilities

15.1 SAP HANA Platform Lifecycle Management (Security)

SAP HANA platform lifecycle management covers the tools for installing, configuring, and updating SAP HANA platform components.

The SAP HANA database lifecycle manager (HDBLCM) is used to install, configure, and update the components of SAP HANA. They are intended to be used by SAP HANA hardware partners within their factory process or by those holding E_HANAINS certification as part of the Tailored Datacenter Integration (TDI) approach.

During the installation process, the initial passwords for 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 database user (SYSTEM) at the same time.

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

SAP HANA Administration Guide
SAP HANA Server Installation and Update Guide

15.2 SAP HANA Content (Security)

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.

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.

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 245]

15.3 SAP HANA Smart Data Access (Security)

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.

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).

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:

Table 88:

Privilege Type	Privilege
System privilege	CREATE REMOTE SOURCE
SQL object privilege on remote source	• CREATE VIRTUAL TABLE • DROP

Related Information

Secure Internal Credential Store [page 146]

SAP HANA Administration Guide

SAP Note 2303807 (SAP HANA Smart Data Access: SSO with Kerberos and Microsoft Windows Active Directory)

15.4 SAP HANA R Integration (Security)

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 truststore 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 connection, see *Set Up SSL/TLS from SAP HANA to Rserve* in the *SAP HANA R Integration Guide*.

Related Information

Server-Side TLS/SSL Configuration Properties for External Communication (JDBC/ODBC) [page 41]
SAP HANA R Integration Guide (PDF)
SAP HANA R Integration Guide (HTML)

15.5 SAP HANA Information Composer (Security)

The SAP HANA information composer is a Web application that allows you to upload to and manipulate data on the SAP HANA database.

> **Note**
>
> SAP HANA Information Composer is supported on Intel-based hardware platforms only.

The SAP HANA information composer uses a Java server which interacts with the SAP HANA database. The Java server communicates with the SAP HANA information composer client via HTTP or HTTPS. The following ports are used by default:

- HTTP port 8080
- HTTPS port 8443

If HTTPS is used, the SSL certification must be configured by the administrator.

> **Note**
>
> The SAP HANA information composer can be configured to use anti-virus software.

The SAP HANA information composer client is accessible to users who are assigned the IC_MODELER role. This role allows users to upload new content into the SAP HANA database and to create physical tables and calculation views.

When content is marked as shared, it is accessible from users who are assigned the IC_PUBLIC role. By default, the physical tables and calculation views are marked as private. This means that they are only visible to the user who created them. Calculation views are created by the _SYS_REPO user in the _SYS_BIC schema and are visible in the *Column Views* node of the catalog in the *Systems* view of the SAP HANA studio.

The physical tables and calculation views can be shared with users who are assigned the IC_PUBLIC role. The IC_PUBLIC role is included in the IC_MODELER role.

The created calculation view inherits the analytical privileges of the source data that is being used. Objects that are based on user data (spreadsheets) have no analytical privileges.

The `SAP_IC` technical user is created during installation. After installation has completed, SAP_IC is locked.

> **Note**
>
> As long as the SAP HANA information composer is in use, the SAP_IC user must not be deleted because otherwise, the role assignments created by this user will also be deleted.

Related Information

SAP HANA Information Composer – Installation and Configuration Guide

16 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 BusinessObjects 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 BusinessObjects Enterprise Administrator's Guide* or the *SAP BusinessObjects Information Platform Services Administrator's 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 BusinessObjects Data Services database, see the *Security* section in the *SAP BusinessObjects Data Services Administrator's Guide*.

SAP HANA Direct Extractor Connection (DXC)

By default, the SAP HANA Direct Extractor Connection technology is switched off. For more information about how to switch it on, see the *SAP HANA Direct Extractor Connection Implementation Guide*.

For secure communication, the SAP HANA Direct Extractor Connection technology uses the SSL protocol (HTTPS) based on the Internet Communication Manager (ICM).

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.

> ⚠ Caution
>
> SAP LT Replication Sever is part of the SAP HANA real-time replication option. Be aware that you need additional licenses for SAP HANA options and capabilities. For more information, Important Disclaimer for Features in SAP HANA Platform, Options and Capabilities [page 270].

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 *SAP HANA Smart Data Integration and SAP HANA Smart Data Quality Administration Guide*.

> ⚠ Caution
>
> SAP HANA smart data integration is an SAP HANA option. Be aware that you need additional licenses for SAP HANA options and capabilities. For more information, Important Disclaimer for Features in SAP HANA Platform, Options and Capabilities [page 270].

Related Information

Installation Guide SAP BusinessObjects Information Platform Services 4.0 (UNIX)
Installation Guide SAP BusinessObjects Information Platform Services 4.0 (Windows)
SAP BusinessObjects Data Services Administrator's Guide
SAP HANA Direct Extractor Connection Implementation Guide
Internet Communication Manager (ICM)
SAP HANA Security Guide - Trigger-Based Replication (SLT)
SAP NetWeaver Application Server ABAP Security Guide
User and Role Administration of Application Server ABAP
SAP HANA Smart Data Integration and SAP HANA Smart Data Quality Administration Guide

17 SAP HANA Security Reference Information

Security reference information for SAP HANA

> **Note**
>
> Please also refer to the document *SAP HANA Security Checklists and Recommendations*.

Security Reference for Multitenant Database Containers [page 240]
: Reference information for secure configuration of tenant databases

Components Delivered as SAP HANA Content [page 245]
: 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.

Related Information

SAP HANA Security Checklists and Recommendations (HTML)
SAP HANA Security Checklists and Recommendations (PDF)

17.1 Security Reference for Multitenant Database Containers

Reference information for secure configuration of tenant databases

Restricted Features in Multitenant Database Containers [page 241]
: To safeguard and/or customize your system, certain features of the SAP HANA database can be disabled in tenant databases.

Default Blacklisted System Properties in Multitenant Database Containers [page 243]
: In systems that support multitenant database containers, there is configuration change blacklist `multidb.ini`, which is delivered with a default configuration.

17.1.1 Restricted Features in Multitenant Database Containers

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 *Disable Restricted Features on a Tenant Database* in the *SAP HANA Administration Guide*.

> **Note**
>
> Features are hierarchically structured. If you disable a feature with sub-features, these are also disabled.

Table 89:

Feature	Feature Description	Why Disable?
AFL	Access to Application Function Libraries (AFL) for business logic in native C ++	Feature not required in all deployment scenarios
BACKUP	Backup operations	File system access not wanted
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
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.MANAGEMENT_CONSOLE_PROC	Access to the built-in SAP HANA management console (`hdbcons`)	Safeguard against leakage of SAP HANA process information
BUILTINPROCEDURE.GEM	Procedure to use the graph engine	Feature not required in all deployment scenarios
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

Feature	Feature Description	Why Disable?
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.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 statisticsserver 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		
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
BUILTINPROCEDURE.REORG_GENERATE	Data redistribution operations	Feature not required in all deployment scenarios
BUILTINPROCEDURE.REORG_EXECUTE		
BUILTINPROCEDURE.REORG_CLEAR_LOGS		
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	

Feature	Feature Description	Why Disable?
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	
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
RINTEGRATION	R language	Feature not required in all deployment scenarios
BOE	SAP BusinessObjects Explorer API	Feature not required in all deployment scenarios

Related Information

SAP HANA Administration Guide
SAP HANA SQL and System Views Reference

17.1.2 Default Blacklisted System Properties in Multitenant Database Containers

In systems that support multitenant database containers, there is configuration change blacklist `multidb.ini`, which is delivered with a default configuration.

The table below lists the system properties that are included in the `multidb.ini` file by default. This means that tenant database administrators cannot 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, see *Prevent Changes to Specific System Properties in Tenant Databases* in the *SAP HANA Administration Guide*.

Table 90:

File/Section	Properties	Description
auditing configuration	• default_audit_trail_type • emergency_audit_trail_type • alert_audit_trail_type • critical_audit_trail_type	Prevents configuration of audit trail targets
indexserver.ini/ authentication	SapLogonTicketTrustStore	Prevents configuration of the trust store for user authentication with logon/assertion tickets
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 option)
global.ini/persistence	• basepath_datavolumes_es • basepath_logvolumes_es • basepath_databackup_es • basepath_logbackup_es	
multidb.ini/ readonly_parameters	*	Prevents configuration of the multidb.ini file itself
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

SAP HANA Administration Guide

17.2 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*.

Administration [page 245]
: SAP HANA content related to system and database administration

Application Lifecycle Management [page 257]
: SAP HANA content for application lifecycle management

Runtime Libraries [page 259]
: SAP HANA content for runtime libraries

Configuration [page 260]
: SAP HANA content for configuration

Supportability and Development [page 261]
: SAP HANA content for supportability and development

User Interface [page 264]
: SAP HANA content for user interface

Documentation [page 267]
: SAP HANA documentation delivered as SAP HANA content

Related Information

SAP HANA Content (Security) [page 232]

17.2.1 Administration

SAP HANA content related to system and database administration

- HANA_ADMIN [page 246]
- HANA_BACKUP [page 247]
- HANA_HDBLCM [page 248]
- HANA_SEC_BASE [page 249]
- HANA_SEC_CP [page 251]
- HANA_SYS_ADMIN [page 253]
- HANA_XS_BASE [page 254]

17.2.1.1 HANA_ADMIN

This component provides Web applications required for the effective administration and monitoring of the SAP HANA platform in both production and non-production environments.

Technical Details

Table 91:

Delivery unit	HANA_ADMIN
Prerequisites	HANA_UI_INTEGRATION_SVC, SAPUI5_1
Content type	Automated content
Content details	SAP HANA XS classic applications
Target users	SAP HANA database administrators
Web application URL	http(s)://<host>:<port>/sap/hana/admin/cockpit

Key Features

The SAP HANA Database Administration component includes the following applications:

- SAP HANA cockpit, an SAP Fiori Launchpad site providing administrators with a single point-of-access to applications for the administration of SAP HANA.
- Applications for the following basic database administration tasks:
  - Monitoring and managing database services
  - Monitoring alerts
  - Configuring alert checkers (for example, alerting schedule and thresholds, e-mail notifications)
  - Monitoring historical performance data of the database across a range of key performance indicators related in particular to memory, disk, and CPU usage
  - Analyze the comparative memory utilization of column tables
  - Monitoring current most critical statements
  - Monitoring system replication status

Roles

The following roles are available with the SAP HANA Database Administration component. Users must be granted one or more of these roles before they can use the component and its functions.

Table 92:

Role	Description
`sap.hana.admin.roles::Monitoring`	Allows users to open the SAP HANA cockpit with read-only access to monitoring data This role also allows users to see tiles in the *SAP HANA Platform Lifecycle Management* and *Smart Data Access Administration* tile catalogs.
`sap.hana.admin.roles::Administrator`	Allows users to open the SAP HANA cockpit with read-only access to monitoring data, as well as to perform database administration tasks supported by the cockpit (configure alerts, stop/start services, reset memory statistics, cancel sessions) This role also allows users to see tiles in the *SAP HANA Platform Lifecycle Management* tile catalog.
`sap.hana.admin.cockpit.sysrep.roles::SysRepAdmin`	Allows users read-only access to monitor system replication status

The following additional roles are available with the `HANA_ADMIN` component:

- `sap.hana.admin.roles::SolutionMangagerMonitor`
- `sap.hana.admin.roles::RestrictedUserDBSLAccess`

These roles are not required to use Web applications delivered with `HANA_ADMIN`, but may be granted to users administrating SAP HANA using SAP Solution Manager and SAP NetWeaver tools to ensure they have the required authorization in SAP HANA.

17.2.1.2 HANA_BACKUP

This component provides a Web application for backing up the SAP HANA database, monitoring the progress of a running backup and the status of existing backups.

Technical Details

Table 93:

Delivery unit	`HANA_BACKUP`
Prerequisites	`HANA_UI_INTEGRATION_SVC`, `SAPUI5_1`
Content type	Automated content
Content details	SAP HANA XS classic application
Target users	SAP HANA database administrators
Web application URL	`http(s)://<host>:<port>/sap/hana/backup`

Features

The SAP HANA Backup component provides a Web application that enables users to create a full data backup and delta backups (differential and incremental), and to monitor the progress of a running backup or the status of the last backup if currently none is running.

Roles

The following roles are available with the SAP HANA Backup component. Users must be granted one or more of these roles before they can use the component and its functions.

Table 94:

Role	Description
`sap.hana.backup.roles::Operator`	Provides access to the SAP HANA Backup application and to create a backup
`sap.hana.backup.roles::Administrator`	Includes the above role and in addition provides write access to the backup catalog

17.2.1.3 HANA_HDBLCM

This component provides access to the SAP HANA platform lifecycle management Web user interface from the SAP HANA cockpit.

Technical Details

Table 95:

Delivery unit	`HANA_HDBLCM`
Prerequisites	`HANA_ADMIN`
Content type	Automated content
Content details	SAP HANA cockpit plug-in
Target users	SAP HANA database administrators
Web application URL	`http(s)://<host>:<port>/lmsl/HDBLCM/<sid>/index.html`

Key Features

SAP HANA platform lifecycle management can be used to update the SAP HANA system, to install or update additional platform components, to add or remove hosts from the system, and to configure settings.

Roles

Users must be granted one of these roles before they can use the component and its functions:

Table 96:

Role	Description
`sap.hana.admin.roles::Monitoring`	Allows users to see the SAP HANA platform lifecycle management tiles in the SAP HANA cockpit
`sap.hana.admin.roles::Administrator`	

To open and use the SAP HANA platform lifecycle management Web user interface, the user needs to authenticate with the `<sid>`adm operating system user.

17.2.1.4 HANA_SEC_BASE

This component provides the database views, client API, and roles to be consumed by applications that provide security-related administration features.

Technical Details

Table 97:

Delivery unit	`HANA_SEC_BASE` **Note** `HANA_SEC_BASE` is delivered as `YHANA_SEC_BASE` to ensure that automated content is deployed in the correct sequence.
Prerequisites	`SAPUI5_1`
Content type	Automated content
Content details	Data service of database engine, role definitions, client side API
Target users	System administrators responsible for security-related configuration, user management and/or management of public key infrastructure (PKI)
Web application URL	None

Key Features

The SAP HANA Security Base component provides a remote API for the following functions:

- Assign roles to users
- Search for users by name and email

- List users, roles, granted privileges, effective privileges, granted roles, and user attributes
- List certificates in certificate store
- Import certificates into certificate store
- List certificate collections and the contained certificates
- Edit certificate collections
- List audit policies and global auditing settings
- Edit audit policies and global auditing settings
- List encryption information (cryptographic library, data encryption status, network encryption settings)
- Trigger the encryption and decryption of data volumes
- Change root key for data volume encryption
- Edit the password policy and password blacklist

Roles

The following roles are available with the SAP HANA Security Base component. Users must be granted one or more of these roles before they can use the component and its functions.

> **Note**
>
> These roles should only be granted to access the system using OData or Rest services. For SAP HANA cockpit access, use the roles described in HANA_SEC_CP.

Table 98:

Role	Description
`sap.hana.security.base.roles::HANACertificateAdmin`	Provides read-only access to certificates and certificate collections if they have system privileges CERTIFICATE ADMIN and TRUST ADMIN are granted. If not, only certificates in own certificate collections are displayed. If users have system privilege TRUST ADMIN, they can create new certificate collections. If they have CERTIFICATE ADMIN, they can add new certificates.
`sap.hana.security.base.roles::HANACertificateView`	Provides read-only access to view certificates and certificate collections with OData services
`sap.hana.security.base.roles::XSUserAdmin`	Provides read-only access to users, roles, granted privileges, granted roles and effective privileges of current user, and allows granting of roles to users
`sap.hana.security.base.roles::XSUserView`	Provides read-only access to users, roles, granted privileges, granted roles and effective privileges of current user, and granted roles
`sap.hana.security.base.roles::HANASecurityDashboardView`	Provides read-only access to following security-related configuration information: • Auditing, including audit policies and audit trail targets • Data volume encryption • SSFS key changes • Network communication

Role	Description
`sap.hana.security.base.roles::HANADataVolumeEncryptionAdmin`	Provides users with read-only access to security-related configuration information and allows them to trigger the encryption and decryption of data volumes and change the root key used for data volume encryption
`sap.hana.security.base.roles::HANAPasswordPolicyAdmin`	Provides users with read-only access to security-related configuration information and allows them to edit the password policy and password blacklist
`sap.hana.security.base.roles::HANAAuditPoliciesAdmin`	Provides users with read-only access to security-related configuration information and allows them to create and edit audit policies, as well as make global auditing settings such as enabling and disabling auditing and configuring audit trail targets

17.2.1.5 HANA_SEC_CP

This component provides Web applications for monitoring critical security settings and performing security-related administration tasks.

Technical Details

Table 99:

Delivery unit	`HANA_SEC_CP` **Note** `HANA_SEC_CP` is delivered as `YHANA_SEC_CP` to ensure that automated content is deployed in the correct sequence.
Prerequisites	`HANA_SEC_BASE`
Content type	Automated content
Content details	SAP HANA XS classic applications
Target users	SAP HANA user and security administrators
Web application URL	`http(s)://<host>:<port>/sap/hana/admin/cockpit`

Key Features

The SAP HANA Security Cockpit component includes the following applications for performing the following tasks:

- Monitoring the status of critical security settings (auditing, data storage, network communication, authentication configuration)

- Assigning roles to users
- Editing the password policy and password blacklist
- Encrypting and decrypting data volumes
- Changing the root key for data volume encryption
- Managing certificates and certificate collections stored directly in the database
- Configuring auditing and managing audit policies

Roles

The following roles are available with the SAP HANA Security Cockpit component. Users must be granted one or more of these roles before they can use the component and its functions.

Table 100:

Role	Description
`sap.hana.security.cockpit.roles::DisplayAssignedRoles`	Allows users to see which roles are granted to users
`sap.hana.security.cockpit.roles::EditAssignedRoles`	Allows users to grant roles to users
`sap.hana.security.cockpit.roles::DisplayCertificateStore`	Allows users read-only access to certificates and certificate collections stored in the database
`sap.hana.security.cockpit.roles::MaintainCertificates`	Allows users to import trusted certificates into the certificate store
`sap.hana.security.cockpit.roles::MaintainCertificateCollections`	Allows users to create collections, as well as add trusted certificates and server certificates to collections
`sap.hana.security.cockpit.roles::EditCertificateStore`	Allows user to set the purpose of a collection in conjunction with either system privilege USER ADMIN or SSL admin and object privilege REFERENCES on the collection
`sap.hana.security.cockpit.roles::DisplaySecurityDashboard`	Allows users to see information about critical security settings (auditing, data storage, and network communication)
`sap.hana.security.cockpit.roles::MaintainDataVolumeEncryption`	Allows users to see information about critical security settings, as well as to enable and disable data volume encryption and change the root key used for data volume encryption
`sap.hana.security.cockpit.roles::MaintainPasswordPolicy`	Allows users to see information about critical security settings, as well as to edit the password policy and password blacklist
`sap.hana.security.cockpit.roles::MaintainAuditPolicy`	Allows users to see information about critical security settings, as well as to create and edit audit policies and to make global auditing settings such as enabling and disabling auditing and configuring audit trail targets

17.2.1.6 HANA_SYS_ADMIN

This component provides Web applications for the administration and monitoring of tenant databases in SAP HANA systems that support multitenant database containers. This component is installed only on the system database of a multiple-container system.

Technical Details

Table 101:

Delivery unit	`HANA_SYS_ADMIN`
Prerequisites	`HANA_ADMIN`
Content type	Automated content
Content details	SAP HANA XS classic applications
Target users	SAP HANA system administrators
Web application URL	`http(s)://<host>:<port>/sap/hana/admin/cockpit`

Key Features

The SAP HANA System Administration component includes applications for performing the following tasks:

- Monitoring and managing (stop, start, create, delete) tenant databases
- Monitoring alerts and alert configurations in a tenant database
- Monitoring historical performance data of a tenant database across a range of key performance indicators related in particular to memory, disk, and CPU usage

Roles

The following roles are available with the SAP HANA System Administration component. Users must be granted one or more of these roles before they can use the component and its functions.

Table 102:

Role	Description
`sap.hana.admin.cockpit.sysdb.roles::SysDBAdmin`	Allows users in the system database to monitor and manage tenant databases in a multiple-container system

17.2.1.7 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

Table 103:

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 such as blacklist/whitelist email id/domain/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 be granted one or more of these roles before they can use the component and its functions.

Table 104:

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 such as blacklist/whitelist email id/domains/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.

17.2.2 Application Lifecycle Management

SAP HANA content for application lifecycle management

- HANA_XS_LM [page 257]

17.2.2.1 HANA_XS_LM

This component provides a Web application for the application lifecycle management of components developed for SAP HANA XS.

Technical Details

Table 105:

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 be granted one or more of these roles before they can use the component and its functions:

Table 106:

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 this role can view all information available but cannot make any changes or trigger any transport operations.
`sap.hana.xs.lm.roles::Execute Transport`	Users with this role can view all information as well as trigger predefined transport operations. However, users with this role cannot register or maintain systems, create transport routes, or edit details of a product, a delivery unit, or a package.
`sap.hana.xs.lm.roles::Transport`	For technical users only. This role 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 this role 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 this role 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 this role can perform all supported requests for CTS Ping SL protocol service.

For tasks that require interaction with external tools, the following additional roles are required:

Table 107:

Role	Description
`sap.hana.ide.roles::EditorDeveloper`	Inspect, create, change, delete and activate SAP HANA repository objects This role 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) This role 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 This role is required when you register a system for a transport route.

The following roles are required for SAP HANA Application Lifecycle Management Process Engine:

Table 108:

Role	Description
`sap.hana.xs.lm.pe.roles::PE_Display`	The user can monitor processes and display services
`sap.hana.xs.lm.pe.roles::PE_Execute`	In addition to the previous role, the user can start, stop, skip, and resume processes.
`sap.hana.xs.lm.pe.roles::PE_Activate`	In addition to the previous roles, the user can activate services from repository files.
`sap.hana.xs.lm.roles::Administrator`	This role includes all previous roles.

17.2.3 Runtime Libraries

SAP HANA content for runtime libraries

- HANA_XS_DBUTILS [page 260]

17.2.3.1 HANA_XS_DBUTILS

This component provides content for the simplified consumption of SAP HANA database objects for XSJS.

Technical Details

Table 109:

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.

17.2.4 Configuration

SAP HANA content for configuration

- HANA_TA_CONFIG [page 261]

17.2.4.1 HANA_TA_CONFIG

This component provides predefined configurations and dictionaries used by the SAP HANA text analysis engine and by text mining.

Technical Details

Table 110:

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.

17.2.5 Supportability and Development

SAP HANA content for supportability and development

- HANA_IDE_CORE [page 262]
- HANA_XS_IDE, HANA_XS_EDITOR [page 263]
- HANA_DT_BASE [page 263]

17.2.5.1 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

Table 111:

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 be granted one or more of these roles before they can use the component and its functions.

Table 112:

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

17.2.5.2 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.

17.2.5.3 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

Table 113:

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 be granted one or more of these roles before they can use the component and its provided services. 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.

Table 114:

Role	Description
`sap.hana.xs.dt.base::restapi`	Allows users to access the REST API

17.2.6 User Interface

SAP HANA content for user interface

- HANA_UI_INTEGRATION_SVC, HANA_UI_INTEGRATION_CONTENT [page 265]
- SAPUI5_1 [page 266]
- SAP_WATT [page 267]

17.2.6.1 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

Table 115: 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

Table 116: 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 be granted one or more of these roles before they can use the component and its provided services.

Table 117:

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

17.2.6.2 SAPUI5_1

This component provides SAP UI5, which is the library used by XS-based Web applications and tools to implement the specific user interfaces.

All Web applications delivered with SAP HANA such as the SAP HANA cockpit, SAP HANA Application Lifecycle Management, and SAP HANA Web-based Development Workbench rely on this delivery unit.

Technical Details

Table 118:

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.

17.2.6.3 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 Web applications delivered with SAP HANA such as the SAP HANA cockpit, SAP HANA Application Lifecycle Management, and SAP HANA Web-based Development Workbench rely on this delivery unit.

Technical Details

Table 119:

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.

17.2.7 Documentation

SAP HANA documentation delivered as SAP HANA content

- HDC_* [page 268]

17.2.7.1 HDC_*

These components provide product documentation for several Web applications delivered with SAP HANASAP HANA. Users can access the documentation via a tile on the application homepage and from the Help menu if available.

Technical Details

Table 120:

Delivery unit	• HDC_ADMIN • HDC_XS_BASE • HDC_IDE_CORE • HDC_SEC_CP • HDC_SYS_ADMIN • HDC_XS_LM
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 database and system administration with SAP HANA Cockpit

  > **Note**
  >
  > The HDC_SYS_ADMIN component is installed only on the system database of a multiple-container system.

- SAP HANA security administration with SAP HANA Cockpit
- 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 Platform, Options and Capabilities

SAP HANA server software and tools can be used for several SAP HANA platform and options scenarios as well as the respective capabilities used in these scenarios. The availability of these is based on the available SAP HANA licenses and the SAP HANA landscape, including the type and version of the back-end systems the SAP HANA administration and development tools are connected to. There are several types of licenses available for SAP HANA. Depending on your SAP HANA installation license type, some of the features and tools described in the SAP HANA platform documentation may only be available in the SAP HANA options and capabilities, which may be released independently of an SAP HANA Platform Support Package Stack (SPS). Although various features included in SAP HANA options and capabilities are cited in the SAP HANA platform documentation, each SAP HANA edition governs the options and capabilities available. Based on this, customers do not necessarily have the right to use features included in SAP HANA options and capabilities. For customers to whom these license restrictions apply, the use of features included in SAP HANA options and capabilities in a production system requires purchasing the corresponding software license(s) from SAP. The documentation for the SAP HANA optional components is available in SAP Help Portal at http://help.sap.com/hana_options. If you have additional questions about what your particular license provides, or wish to discuss licensing features available in SAP HANA options, please contact your SAP account team representative.

Important Disclaimers and Legal Information

Coding Samples

Any software coding and/or code lines / strings ("Code") included in this documentation are only examples and are not intended to be used in a productive system environment. The Code is only intended to better explain and visualize the syntax and phrasing rules of certain coding. SAP does not warrant the correctness and completeness of the Code given herein, and SAP shall not be liable for errors or damages caused by the usage of the Code, unless damages were caused by SAP intentionally or by SAP's gross negligence.

Accessibility

The information contained in the SAP documentation represents SAP's current view of accessibility criteria as of the date of publication; it is in no way intended to be a binding guideline on how to ensure accessibility of software products. SAP in particular disclaims any liability in relation to this document. This disclaimer, however, does not apply in cases of willful misconduct or gross negligence of SAP. Furthermore, this document does not result in any direct or indirect contractual obligations of SAP.

Gender-Neutral Language

As far as possible, SAP documentation is gender neutral. Depending on the context, the reader is addressed directly with "you", or a gender-neutral noun (such as "sales person" or "working days") is used. If when referring to members of both sexes, however, the third-person singular cannot be avoided or a gender-neutral noun does not exist, SAP reserves the right to use the masculine form of the noun and pronoun. This is to ensure that the documentation remains comprehensible.

Internet Hyperlinks

The SAP documentation may contain hyperlinks to the Internet. These hyperlinks are intended to serve as a hint about where to find related information. SAP does not warrant the availability and correctness of this related information or the ability of this information to serve a particular purpose. SAP shall not be liable for any damages caused by the use of related information unless damages have been caused by SAP's gross negligence or willful misconduct. All links are categorized for transparency (see: http://help.sap.com/disclaimer).

go.sap.com/registration/contact.html