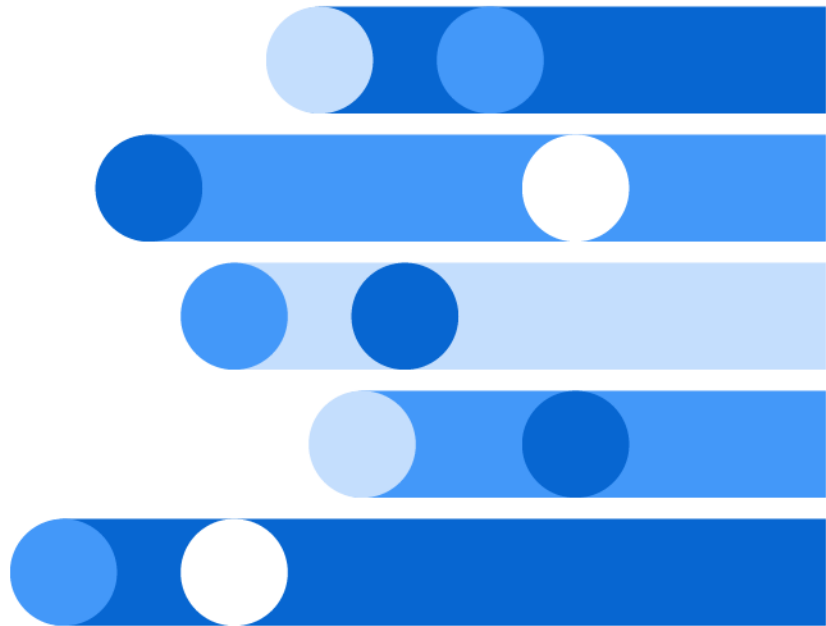




SAS[®] 9.4 Intelligence Platform: Security Administration Guide, Third Edition



The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2025. *SAS® 9.4 Intelligence Platform: Security Administration Guide, Third Edition*. Cary, NC: SAS Institute Inc.

SAS® 9.4 Intelligence Platform: Security Administration Guide, Third Edition

Copyright © 2025, SAS Institute Inc., Cary, NC, USA

All Rights Reserved. Produced in the United States of America.

For a hard copy book: No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, or otherwise, without the prior written permission of the publisher, SAS Institute Inc.

For a web download or e-book: Your use of this publication shall be governed by the terms established by the vendor at the time you acquire this publication.

The scanning, uploading, and distribution of this book via the Internet or any other means without the permission of the publisher is illegal and punishable by law. Please purchase only authorized electronic editions and do not participate in or encourage electronic piracy of copyrighted materials. Your support of others' rights is appreciated.

U.S. Government License Rights; Restricted Rights: The Software and its documentation is commercial computer software developed at private expense and is provided with RESTRICTED RIGHTS to the United States Government. Use, duplication, or disclosure of the Software by the United States Government is subject to the license terms of this Agreement pursuant to, as applicable, FAR 12.212, DFAR 227.7202-1(a), DFAR 227.7202-3(a), and DFAR 227.7202-4, and, to the extent required under U.S. federal law, the minimum restricted rights as set out in FAR 52.227-19 (DEC 2007). If FAR 52.227-19 is applicable, this provision serves as notice under clause (c) thereof and no other notice is required to be affixed to the Software or documentation. The Government's rights in Software and documentation shall be only those set forth in this Agreement.

SAS Institute Inc., SAS Campus Drive, Cary, NC 27513-2414

August 2026

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

9.4-P5:bisecag

Contents

<i>About This Document</i>	<i>vii</i>
<i>What's New in Security Administration in SAS 9.4</i>	<i>ix</i>

PART 1 Fundamentals 1

Chapter 1 / Security Overview	3
Security in the Platform	3
Auditing	4
Authorization	5
Encryption	6
Roles	6
Single Sign-On	7
Chapter 2 / User Administration	9
About User Administration	10
Metadata Users	12
Metadata Groups	13
Metadata Roles	15
Password and Secret Management	17
Identity Hierarchy	20
External Identities	22
Windows Privileges	23
Windows User ID Formats	25
Uniqueness Requirements	27
Configuring Client Machines to Use Integrated Windows Authentication	28
Chapter 3 / Access Management	33
About Access Management	33
Basics of Metadata Authorization	35
WriteMetadata and WriteMemberMetadata	38
Review: Key Points about Authorization	39
Chapter 4 / Selected Tasks	41
About Security Task Instructions	41
Create Metadata User Definitions	42
Update a Managed Password	44
Unlock an Internal Account	47
Adjust Initial Access	49

PART 2 Authorization 53

Chapter 5 / Metadata Authorization Model	55
Authorization Overview	56
Three Levels of Granularity	57
Two Relationship Networks	57
Object Inheritance	59
Permissions by Object Type	61
Permissions by Task	67
Permission Origins	72
Types of Access Controls	74
Authorization Decisions	75
Fine-Grained Controls for Data	77
Use and Enforcement of Each Permission	84
Tips for Efficiently Using Permissions	85
Chapter 6 / Access to Metadata Folders	89
Baseline ACTs	89
Example: Business Unit Separation	91
Variation 1: Regional Separation, Designated Content Creators	92
Variation 2: Functional Separation	95
Key Points about the Baseline ACT Approach	97
Further Considerations for Permissions on Folders	99
Chapter 7 / Access to Server Definitions	101
Protect Server Definitions	101
Hide Server Definitions	105
Chapter 8 / Access to SAS Data	111
About Access to SAS Data	111
Mediated Access	112
Metadata-Bound Libraries	118
Locked-Down Servers	120
PROTOLIBS System Option	128
Chapter 9 / Batch Tools for Metadata Authorization	131
About the Batch Tools for Metadata Authorization	131
Batch Tool: sas-show-metadata-access	134
Batch Tool: sas-set-metadata-access	135
Batch Tool: sas-make-act	138

PART 3 Authentication 141

Chapter 10 / Authentication Model	143
Introduction to the Authentication Model	144
Authentication to the Metadata Server	144
Authentication to Data Servers and Processing Servers	146
Mixed Providers	148

Credential Gaps	150
How Logins Are Used	152
Authentication Domains	155
Identity Passing	156
Launch Credentials	159
PUBLIC Access and Anonymous Access	164
Chapter 11 / Authentication Mechanisms	167
Introduction to Authentication Mechanisms	168
Credential Management	168
Direct LDAP Authentication	171
Host Authentication	172
Integrated Windows Authentication	174
Multifactor Authentication (MFA)	176
Pluggable Authentication Modules (PAM)	177
SAS Internal Authentication	178
SAS Token Authentication	181
Trusted Peer Connections	182
Trusted User Connections	184
Web Authentication	185
Microsoft Windows Defender Credential Guard	188
Summary of Methods for LDAP Integration	190
Summary for Single Sign-On	191
Summary by Server Type	192
Chapter 12 / Authentication Tasks	195
How to Manually Enable Oracle Wallet	196
How to Facilitate Authentication	203
How to Configure SAS Token Authentication	205
How to Configure Web Authentication	207
How to Configure Direct LDAP Authentication	209
How to Configure Integrated Windows Authentication	217
How to Configure Kerberos Constrained Delegation	229
About IBM Z Multi-Factor Authentication (MFA)	234
How to Store Passwords for the Workspace Server	235
How to Store Passwords for a Third-Party Server	237
How to Change Internal Account Policies	239
About the Workspace Server Options Tab	243
PART 4 Encryption 245	
Chapter 13 / Encryption Model	247
Introduction to the Encryption Model	247
Encryption for Data at Rest	248
Encryption for Data in Motion	250
What Are Certificates?	251
How SAS Uses Certificates	253
Obtaining Certificates	259
Implementing Certificates	262

Chapter 14 / Encryption Tasks	265
How to Maximize Encryption of Stored Passwords	266
How to Re-encrypt Stored Passwords	268
How to Configure TLS for IOM and Other SAS Servers and Change the Encryption Level	273
Configure the SAS Intelligence Platform Server Tier to Use IBM System SSL on z/OS	285
How to Change Encoding for Data in Motion for Outbound Passwords	291
How to Configure TLS between the Metadata Server and an LDAP Server	294
How to Implement Certificates	298
How to Enable IOM Communication over HTTPS Proxy	305

PART 5 Appendix 309

Appendix 1 / Checklists	311
Checklist for a More Secure Deployment	311
Members of Selected Groups and Roles	313
Permission Patterns in Predefined ACTs	314
Appendix 2 / User Import Macros	317
Overview of User Bulk Load and Synchronization	317
Canonical Tables	321
User Bulk Load	323
User Synchronization	326
Sample Code for User Synchronization	328
Sample Code for Generic Bulk Load	330
About the Sample Code for UNIX /etc/passwd	332
About the Sample Code for Active Directory	334
Location of the User Bulk Load and Synchronization Macros	336
Appendix 3 / Security Report Macros	345
Overview of the Security Report Macros	345
Authorization Data Sets	347
Additional Resources for Building Authorization Data Sets	350
Appendix 4 / Choices in Workspace Server Pooling	357
About Workspace Server Pooling	357
Benefits and Risks of Server-Side Pooling	358
Which Requests Are Eligible to Use Pooling?	360
Which Eligible Requests Actually Use Pooling?	361
Modifying the Initial Pooling Configuration	362

About This Document

Audience

This document helps SAS administrators understand the suite-wide security features of the platform.

This document is organized as follows:

- The first part, Fundamentals, contains the essential information for all security administrators:
 - a brief overview of security features
 - an orientation to managing SAS metadata identities (users, groups, and roles)
 - an orientation to the SAS metadata authorization layer (permissions)
 - instructions for selected tasks
- The rest of the document contains reference information and instructions for specialized configurations.

Here is a list of related topics that are outside the scope of this book:

- Security features that are provided by the middle tier are documented in [SAS Intelligence Platform: Middle-Tier Administration Guide](#).
- Security features that are specific to a particular product are documented in the administrative guide for that product. For example, roles and capabilities are documented as application-specific features.
- Comprehensive information about each implementation of fine-grained access to data is included in documentation for the implementation's underlying technology. For more information, see [“Fine-Grained Controls for Data” on page 77](#).
- For assistance in using a particular interface to perform security-related tasks, see the documentation for that interface.

Requirements

Prerequisites

This document assumes the following conditions:

- You are familiar with the operating systems in your environment.
- You have completed all installation, deployment, and (if applicable) migration tasks.
- You are familiar with the concepts and terminology that are introduced in [SAS Intelligence Platform: Overview](#).

What's New in Security Administration in SAS 9.4

Overview

New and enhanced features in the following areas increase security and manageability:

- access to resources
- authentication
- encryption
- user administration
- documentation enhancements

Access to Resources

- You can use the PROTOLIBS system option to specify the paths that the PROC PROTO LINK statement can use to find and register load modules. This system option is available in [SAS 9.4M6](#) and later. For more information, see [“PROTOLIBS System Option” on page 128](#).
- You can limit the reach of a SAS server process, ensuring that the process can access only designated resources in the host operating environment. This functionality is available in [SAS 9.4M1](#) and later. For more information, see [“Locked-Down Servers” on page 120](#).
- You can use either SAS Management Console or the AUTHLIB procedure to bind SAS data to metadata. All access from SAS to metadata-bound data is subject to metadata-layer permissions. For more information, see [“Metadata-Bound Libraries” on page 118](#).

- You can manage metadata-layer permissions using batch tools. The tools support viewing and setting access controls and viewing, creating, and modifying access control templates. For more information, see [Chapter 9, "Batch Tools for Metadata Authorization,"](#) on page 131.

Authentication

- In [SAS 9.4M9](#), SAS supports multifactor authentication (MFA) for SAS Logon Manager. As a result:
 - The various components of the SAS 9.4 Intelligence Platform Middle Tier can be configured to use MFA during the logon process. For example, users of SAS Studio and SAS BI Dashboard can be prompted for an additional form of authentication.
 - If you want to configure MFA using SAS Deployment Wizard, SAS supports either a third-party time-based one-time password (TOTP) authenticator application, such as Microsoft Authenticator or Google Authenticator, or the built-in SAS Authenticator component as the authentication provider. For more information, see ["Configure Multifactor Authentication \(MFA\) for SAS Logon Manager"](#) in *SAS Intelligence Platform: Installation and Configuration Guide*.
 - You can also manually configure MFA for SAS Logon Manager. For more information about MFA in [SAS 9.4M9](#), see ["Multifactor Authentication \(MFA\) in SAS 9.4M9"](#) in *SAS Intelligence Platform: Middle-Tier Administration Guide*.
 - MFA is supported for z/OS using IBM Z Multi-Factor Authentication (IBM Z MFA). To apply MFA to SAS/CONNECT and SAS/SHARE on z/OS, see ["About IBM Z Multi-Factor Authentication \(MFA\)"](#) on page 234.
- Starting with [SAS 9.4M7](#), support for Microsoft Windows Defender Credential Guard (Credential Guard) is available. Through the installation of a series of [SAS 9.4M6](#) hot fixes, Credential Guard support is also available.

Microsoft introduced Credential Guard in Microsoft Windows 10 and later and in Microsoft Windows Server 2016 and later. To prevent threat and security risks, Credential Guard isolates logon information for users from the rest of the operating system by using virtualization to store credentials in protected containers that are separated from the operating system. For more information, see ["Microsoft Windows Defender Credential Guard"](#) on page 188.
- In direct LDAP authentication, the following enhancements are available:
 - You can authenticate against multiple LDAP servers.
 - You can authenticate against multiple LDAP groups within an LDAP server.
 - You can use the `AD_STYLE` environment variable to limit authentication activity to one attempt per logon interaction. By specifying which user ID format SAS Metadata Server uses in an authentication request to LDAP, you

avoid inadvertent locking of Active Directory accounts due to repeated attempts using different formats for the submitted user ID.

- You can use the `AD_TIMEOUT` or `LDAP_TIMEOUT` environment variable to define the amount of time that elapses before an attempt to connect times out.

For more information, see [“How to Configure Direct LDAP Authentication” on page 209](#).

- Starting with [SAS 9.4M4](#), Solaris supports alternative Kerberos libraries for IWA.
- In [SAS 9.4M3](#) and previous releases, a user in multiple distinguished names (DNs) cannot be authenticated when SAS Metadata Server is configured for direct LDAP authentication. Starting with [SAS 9.4M4](#), the user can be authenticated by configuring the `SAS_ALLOW_MULTIPLE_DN_ENTRIES` environment variable. For information about setting this environment variable, see [“Configuration Files for Components of SAS Application Servers” in SAS Intelligence Platform: System Administration Guide](#).
- Starting with [SAS 9.4M3](#), additional security is provided for SAS web applications. In scenarios where applications are using the SAS middle tier as a proxy for accessing external URLs, additional security has been added through an allowlist, or security filter, and logging. You can also add certain HTTP request methods to an allowlist.

SAS web applications that require external access to URLs must also have an allowlist of URLs that can be accessed.

For more information, see these topics:

- [Allowlist of Websites and Methods Allowed to Link to SAS Web Applications](#)
- [Configure the Cross Domain Proxy Servlet through an Allowlist](#)
- [“Enable Support for Forward Proxy Authentication” in SAS Intelligence Platform: Middle-Tier Administration Guide](#)
- Starting with [SAS 9.4M2](#), Integrated Windows authentication (IWA) on Linux systems no longer requires the use of One Identity Authentication Services, which was formerly Quest Authentication Services. SAS can leverage the libraries that are shipped with the supported operating system or that are provided in most third-party authentication solutions.
- [SAS 9.4M2](#) and later support outbound and trusted authentication domains. A logon in an outbound domain can establish connections only to other servers. For outbound domain logons, the constraints on the user ID are not enforced. A logon in a trusted domain can be accessed only by a trusted system account on behalf of the user. A logon in a trusted domain also requires the authentication domain to be an outbound domain. For more information, see [“Outbound and Trusted Authentication Domains” on page 156](#).
- In SAS internal authentication, any server-level internal account policies that have been explicitly set in the metadata server's `omaconfig.xml` file are displayed in SAS Management Console. The display is Read-Only. For more information, see [“How to Change Internal Account Policies” on page 239](#).

- In the middle tier, the remote services component is no longer used. This simplifies administration and increases availability. For more information, see [“Web Authentication” on page 185](#).
- In Integrated Windows authentication, only the standard service principal name format (SAS/machine) is automatically registered. A deprecated alternate format that includes a port number is no longer automatically registered. For more information, see [“How to Configure Integrated Windows Authentication” on page 217](#).

Encryption

- The following changes have been made to encryption support in SAS 9.4M9:
 - SAS Deployment Wizard can be used to configure and enable TLS. SAS Deployment Wizard automates TLS configuration for the following components during SAS installation:
 - SAS Web Application Server

The best practice is to acquire CA-signed certificates before you install and configure SAS software.
 - SAS JMS Broker
 - Cache Locator

For more information, see [“\(SAS 9.4M9\) Use TLS for Internal Connections” in SAS Intelligence Platform: Installation and Configuration Guide](#).
 - You can continue to manually configure SAS Web Application Server, JMS Broker, and Cache Locator after you deploy SAS software. For more information, see [“Configure TLS for the SAS JMS Broker Manually” in SAS Intelligence Platform: Middle-Tier Administration Guide](#), [“Configure TLS for the SAS Cache Locator Manually” in SAS Intelligence Platform: Middle-Tier Administration Guide](#), and [“Configure TLS for SAS Web Application Server Manually” in SAS Intelligence Platform: Middle-Tier Administration Guide](#).
 - The NETENCALG= system option values of AES, DES, RC2, RC4, and TripleDES are deprecated. These values will be removed in an upcoming SAS release. As a result, a WARNING message is now generated when one of these values is specified. This WARNING message in the SAS log instructs you to change the option value to SSL to specify the use of the TLS protocol. SAS strongly recommends that you configure encryption for data in motion using TLS. For more information, see [“Configure TLS Using NETENCRYPTALGORITHM \(Starting with SAS 9.4M7\)” in Encryption in SAS](#).

The log contains important information. However, the log level can be changed from WARNING to INFO. For more information about changing the log level using the new SAS environment variable ACCEPT_RISK_ALLOW_INSECURE_HANDSHAKE, see

[“ACCEPT_RISK_ALLOW_INSECURE_HANDSHAKE Environment Variable”](#) in *Encryption in SAS*.

- PKCS#12 certificate files (file extension .p12) are now provided. The trusted certificate files are `trustedcerts.pem` and `trustedcerts.p12`. Previously, JKS-formatted certificate files (file extension .jks) were provided.

SAS recommends that you provide certificate files in PKCS#12 format instead of JKS formatted certificates for use with SAS on Windows. JKS-formatted certificates are being deprecated.

- The SAS Middle Tier is now configured with OpenSSL 3.0 and the Java TLS implementation that supports TLS 1.3 and FIPS. For more information, see [“What’s New in Middle-Tier Administration for the SAS 9.4 Intelligence Platform”](#) in *SAS Intelligence Platform: Middle-Tier Administration Guide*.

SAS Web Server supports only TLS 1.3 and TLS 1.2.

- The following changes were made to encryption support in SAS 9.4M8:

- Starting with SAS 9.4M8, SAS requires supported operating-system cryptographic libraries to be installed in order to provide encryption for SAS Foundation. For more information, see [“SAS/SECURE \(Starting with SAS 9.4M8\)”](#) in *Encryption in SAS*.
- SAS no longer provides cryptographic libraries for SAS Foundation servers. For more information, see [“Cryptographic Library Support \(Starting with SAS 9.4M8\)”](#) in *Encryption in SAS*.

For all maintenance releases of SAS 9.4 prior to SAS 9.4M8, updated versions of OpenSSL for UNIX and z/OS were provided and updated as needed through hot fixes. For more information, see [“TLS and OpenSSL Version Support \(Prior to SAS 9.4M8\)”](#) in *Encryption in SAS*.

- Starting with SAS 9.4M8, support for the IBM Z Pervasive Encryption feature is available. This support applies to SAS direct access bound libraries and to sequential access libraries. The access method that SAS uses to read and write data sets for z/OS is Execute Channel Program (EXCP). For more information, see Chapter 7 of the [Configuration Guide for SAS 9.4 Foundation for z/OS](#).
- Starting with SAS 9.4M8, encryption for data in motion is provided for z/OS using IBM System SSL. IBM System SSL supports FIPS 140-2. It provides all the capabilities of OpenSSL and is fully integrated with the z/OS security services that have been in use for many years.

Some configuration is required in order to configure a SAS deployment for System SSL. For more information, see Chapter 7 of the [Configuration Guide for SAS 9.4 Foundation for z/OS](#).

- Starting with SAS 9.4M8, SAS no longer provides OpenSSL libraries to support TLS on z/OS. For more information, see [“IBM System SSL Provides TLS Capabilities for z/OS \(Starting with SAS 9.4M8\)”](#) in *Encryption in SAS*.
- Starting with SAS 9.4M8, support for TLS 1.3 is available, and default cipher suites are updated to support TLS 1.3 and TLS 1.2. For more information, see [“TLS Versions and Cipher Suites Supported \(Starting with SAS 9.4M8\)”](#) in *Encryption in SAS*.

- OpenSSL libraries that are used by SAS Foundation ODBC drivers are installed in a different location on Linux machines. The two required OpenSSL libraries are `libcrypto.so.3` and `libssl.so.3`. When SAS 9.4M8 hot fixes are applied, the location of these libraries is changed. For more information, see [SAS Note 67666](#).
 - The `NETENCRYPTALGORITHM=` (alias `NETENCALG=`) system option values of `RC2`, `RC4`, `DES`, `TRIPLEDES`, and `AES` are deprecated when hot fixes related to [KB0041538](#) are applied. These values will be removed in an upcoming SAS release. Change the option value to `SSL` to specify the use of the TLS protocol. For more information, see “[TLS Support for IOM and Other SAS Servers](#)” in *SAS Intelligence Platform: Application Server Administration Guide*.
- Starting with SAS 9.4M7, the `NETENCRYPTALGORITHM=` (alias `NETENCALG=`) system option values of `RC2`, `RC4`, `DES`, `TRIPLEDES`, and `AES` are deprecated when hot fixes related to [KB0041538](#) are applied. These values will be removed in an upcoming SAS release. Change the option value to `SSL` to specify the use of the TLS protocol. For more information, see “[TLS Support for IOM and Other SAS Servers](#)” in *SAS Intelligence Platform: Application Server Administration Guide*.
- Starting with SAS 9.4M5, the SAS005 encoding type (AES encryption with a 256-bit fixed key, a 64-bit random salt value, and additional hash iterations) is available. The fixed key and salt value of SAS005 are the same as those of SAS004, and SAS005 has additional hashed iterations, which increase security for stored passwords. For more information, see “[Introduction to the Encryption Model](#)” on page 247.
- In SAS 9.4M3, the following changes were made to certificates, Transport Layer Security (TLS), and HTTPS security:
 - SAS provides a default truststore (the `jssecacerts` file) that takes precedence over the previous truststore (the `cacerts` file) in the SAS Private JRE.

The trusted CA bundle is a copy of the Mozilla bundle, which is the list of certificate authority (CA) certificates that are distributed with Mozilla software products. For more information, see <https://wiki.mozilla.org/CA:IncludedCAs>.
 - On UNIX, trusted certificates are now located in the trusted CA bundle in `SAS-installation-directory/SASSecurityCertificateFramework/1.1/cacerts/trustedcerts.pem`. During installation, SAS Deployment Wizard sets the `SSLCALISTLOC` system option in the `SAS-installation-directory/SASFoundation/9.4/sasv9.cfg` file to point to the `trustedcerts.pem` file by default.
 - If certificates are not used, SAS ignores `SSLCALISTLOC`.

For more information about the order in which SAS searches for certificates, see [Details](#).
 - Certificate bundle management has been added to SAS Deployment Manager.

Using SAS Deployment Manager, you can add and remove certificates to and from the trusted CA bundle. For more information, see [“Add a Certificate to the Trusted CA Bundle” in *Encryption in SAS*](#).

- In TLS between SAS Metadata Server and LDAP, the `NOSSLNAMECHECK` environment variable disables server name checking. This feature can be useful in accommodating connections to an LDAP load balancer. For more information, see [“How to Configure TLS between the Metadata Server and an LDAP Server” on page 294](#).

To learn about certificates, see [“What Are Certificates?” on page 251](#)

- You can use SAS Deployment Manager or the `METADATA` procedure to re-encrypt passwords that are stored in metadata. This feature provides an easy way to convert currently stored passwords from one encryption type to another. This feature is also used to introduce, change, or remove a passphrase key. For more information, see [“How to Re-encrypt Stored Passwords” on page 268](#).
- You can specify a passphrase to use as the encryption key for passwords that are stored in metadata. Because you supply and control the passphrase, this feature increases security, compared to using the default fixed key. For more information, see [“How to Re-encrypt Stored Passwords” on page 268](#).

See the [SAS Security Bulletin on OpenSSL](#) for the most current information about the versions of OpenSSL that are used in SAS products and about the advisories under consideration.

User Administration

- In SAS Management Console, you can no longer change the name of an existing user, group, or role. You can still change the display name.
- In the SAS metadata, you can no longer give an identity two logons in the same authentication domain by entering the same user ID in two different formats. For more information, see [“Uniqueness Requirements” on page 27](#).
- Restricted user administrators can no longer add or update logons and internal accounts for unrestricted users.

Documentation Enhancements

In [SAS 9.4M4](#), information was moved from the *SAS 9.4 Intelligence Platform: Installation and Configuration Guide, Second Edition* to this document. See the following sections:

- [“What Are Certificates?” on page 251](#)

- [“How SAS Uses Certificates” on page 253](#)
- [“Obtaining Certificates” on page 259](#)
- [“Implementing Certificates” on page 262](#)
- [“How to Implement Certificates” on page 298](#)

PART 1**Fundamentals**

Chapter 1	
Security Overview	3
Chapter 2	
User Administration	9
Chapter 3	
Access Management	33
Chapter 4	
Selected Tasks	41

Security Overview

<i>Security in the Platform</i>	3
<i>Auditing</i>	4
See Also	4
<i>Authorization</i>	5
See Also	5
<i>Encryption</i>	6
See Also	6
<i>Roles</i>	6
<i>Single Sign-On</i>	7
See Also	8

Security in the Platform

The platform provides security features including authentication, auditing, authorization, encryption, and role-based access to application features.

The platform interoperates with systems such as the host environment, web applications, and third-party databases. SAS provides security for connections among all these components.

To coordinate user identity information, at least one user ID (such as a host, Active Directory, or LDAP user ID) is stored in SAS metadata for each registered user.

SAS Metadata Server stores, retrieves, and manages metadata for a SAS environment. It provides information about resources such as users, servers, data, and security to all servers in a SAS deployment.

Auditing

Security-related events are logged as part of a system-wide logging facility. The following table describes security-related log categories.

Table 1.1 *Selected Security-Related Log Categories*

Category	Events Captured
Audit.Authentication	Authentication events, client connection information.
Audit.Meta.Security.AccCtrlAdm	Changes to explicit controls, ACTs, application of ACTs to objects, and passwords (on objects such as Tables, Connections, and ProtectedPassthru). Includes additions, deletions, modifications, and failed attempts to perform these actions.
Audit.Meta.Security.GrpAdm	Changes to memberships (for groups or roles). Includes adding members, removing members, and failed attempts to perform these actions.
Audit.Meta.Security.UserAdm	Changes to users, groups, roles, logins, internal accounts, and authentication domains. Includes additions, deletions, modifications, and failed attempts to perform these actions.
Audit.Meta.Updates.PublicObjects	Add, update, delete, and change management events on public objects.

For more information, see “Administering Logging for SAS Servers” in *SAS Intelligence Platform: System Administration Guide*.

TIP For a collection of utilities to support reporting against the generated logs, search for “Audit and Performance Measurement” at support.sas.com.

See Also

- [SAS Logging: Configuration and Programming Reference](#)

- [“Administering Logging for SAS Servers” in SAS Intelligence Platform: System Administration Guide](#)

Authorization

The platform provides a metadata-based authorization layer that supplements protections from the host environment and other systems. You can use this layer to manage access to almost any metadata object (for example, reports, data definitions, information maps, jobs, stored processes, and server definitions).

Across authorization layers, protections are cumulative. In order to perform a task, a user must have sufficient access in all applicable layers.

In the metadata layer, the following permissions are always enforced:

- the ReadMetadata permission (RM), which controls the ability to see an object
- the WriteMetadata permission (WM), which controls the ability to update or delete an object

Other permissions are specialized and affect only certain types of objects.

CAUTION

In the metadata authorization layer, not all permissions are enforced for all items. It is essential to understand which actions are controlled by each permission. For more information, see [“Use and Enforcement of Each Permission” on page 84](#).

CAUTION

Some clients enable power users to create and run SAS programs that access data directly, bypassing metadata-layer controls. It is essential to manage physical layer access in addition to metadata-layer controls. For more information, see [“Use and Enforcement of Each Permission” on page 84](#).

See Also

[“Security Report Macros” on page 345](#)

Encryption

The platform offers encryption features that help protect information about disk and in transit. Here is an overview of encryption support:

- Passwords in configuration files and the metadata are encrypted or encoded. Most other metadata is not encrypted.
- Passwords in transit to and from SAS servers are encrypted or encoded. You can choose to encrypt all such traffic, instead of encrypting only credentials.
- When you obtain and implement certificates for SAS Web Server and other middle-tier components, you can use auto-generated certificates from SAS Deployment Wizard or provide your own.

See Also

- [Encryption Model on page 247](#)
- [Encryption Tasks on page 273](#)
- [Encryption in SAS](#)

Roles

The platform supports metadata-based application roles that control which application features (such as buttons, tabs, and menu items) are visible to which users.

For example, role memberships determine who can see the Server Manager plug-in (in SAS Management Console), compare data (in SAS Enterprise Guide), or directly open an information map (in SAS Web Report Studio).

Here are some key points about metadata-based application roles:

- In general, roles do not protect data or metadata. Roles just control which features in a particular application are available to which users.
- An application feature that is under role-based management is called a capability. Each role provides multiple capabilities. A user or group can be in multiple roles.
- Not all applications have roles. Not all application features are under role management. Each application that supports roles provides a fixed set of

capabilities. You cannot convert a feature that is not a capability into a capability.

TIP If you add custom tasks (in SAS Enterprise Guide) or develop custom plug-ins (for SAS Management Console), you can register those features as capabilities. For more information, see [SAS Intelligence Platform: Desktop Application Administration Guide](#).

- Capabilities are additive. There are no negative capabilities (capabilities that limit what a user can do). It is not possible to deny a capability (capabilities are either granted or not granted).
- Capabilities can be categorized as follows:
 - explicit capabilities
 - can be incrementally added to or removed from any role (other than the unrestricted role, which always provides all explicit capabilities). Most roles have explicit capabilities.
 - implicit capabilities
 - are permanently bound to a certain role. The metadata server's roles provide implicit capabilities. For example, the user administration role provides the capability to add users, but there is no explicit **Create Users** capability.
 - contributed capabilities
 - are implicit or explicit capabilities that are assigned through role aggregation. If you designate one role as a contributing role for another role, all of the first role's capabilities become contributed capabilities for the second role.
- You cannot assign permissions to a role. You cannot assign capabilities to a group.
- A user cannot temporarily assume or relinquish a role. All of a user's roles are active at all times.

TIP You can give an administrator two user definitions. This enables the administrator to function as a regular user some of the time.

- For details about a particular application's capabilities and roles, see the administrative documentation for that application.

For more information, see ["Metadata Roles" on page 15](#).

Single Sign-On

The platform supports single sign-on (SSO) as follows:

- To bypass the initial logon prompt when launching a desktop application, use Integrated Windows authentication.
- To bypass the initial logon prompt when launching a web application, configure web authentication.
- To avoid secondary logon prompts when accessing data servers and processing servers, use a combination SAS token authentication, Integrated Windows authentication, and credential management.

TIP As a convenience for users who do not use SSO when they log on, you can permit users to save their logon credentials on the client side. For more information, see [“Client-Side Storage of Passwords” on page 18](#).

See Also

- [Authentication Model on page 155](#)
- [Authentication Mechanisms on page 168](#)

User Administration

About User Administration	10
Introduction	10
Who Can Manage Users, Groups, and Roles?	10
Where Is User Administration Performed?	11
See Also	12
Metadata Users	12
See Also	13
Metadata Groups	13
See Also	15
Metadata Roles	15
Role Definitions	15
Predefined Roles	16
Differences between Roles and Groups	17
Availability of Application Features in a New Deployment	17
Password and Secret Management	17
Password Policies	17
Client-Side Storage of Passwords	18
External Logon Passwords	18
Internal Account Passwords	19
Managed Passwords	19
Concealing Passwords and Secrets	19
See Also	20
Identity Hierarchy	20
See Also	21
External Identities	22
See Also	22
Windows Privileges	23
Access This Computer from the Network	23
Log on as a Batch Job	23
Trusted for Delegation	24
See Also	25
Windows User ID Formats	25
See Also	26
Uniqueness Requirements	27
See Also	28

Configuring Client Machines to Use Integrated Windows Authentication	28
About Configuring Client Machines to Use IWA	28
Updating Registry Settings	29
Configuring the Java Cryptography Extension	30
See Also	31

About User Administration

Introduction

In order to make access distinctions and track user activity, a security system must know who is making each request. In the platform, the primary user administration task is to store each user's external account ID in the SAS metadata. SAS uses its copy of these IDs to establish a unique SAS identity for each connecting user. All of a user's metadata-layer memberships, permissions, and capabilities are ultimately tied to the user's SAS identity.

Note: It is not necessary to store passwords in the SAS metadata for the purpose of identifying a user. SAS identity is determined by examining stored user IDs, not by examining stored passwords.

Note: For some service identities and metadata administrators, you can use a SAS internal account instead of a stored SAS copy of an external account ID.

Who Can Manage Users, Groups, and Roles?

In the initial configuration for a new deployment, the SAS Administrators group has the user administration role, so members of that group can perform almost all user management tasks. The following table outlines the distribution of user administration capabilities.

Table 2.1 *User Administration Capabilities*

Metadata Server Role	Actions Supported
Unrestricted	Perform all identity management tasks.
User administration	Add, modify, and delete most identities.

Metadata Server Role	Actions Supported
None	Update your own personal logons.

For restricted user administrators (users who have the user administration role but are not unrestricted), the following constraints apply:

- Restricted user administrators cannot update the unrestricted role.
- To update or delete an identity, restricted user administrators must have the WriteMetadata permission for that identity. For example, you can prevent a restricted user administrator from updating UserA's metadata definition by taking away his or her default grant of the WriteMetadata permission (on UserA's **Authorization** tab, explicitly deny the WriteMetadata permission to the restricted user administrator).
- To change a role's capabilities, restricted user administrators must have the WriteMetadata permission for the associated software component.
- To access user management features in SAS Management Console, restricted user administrators must have the User Manager capability.

Note: You can delegate administration of an existing identity to someone who is not a user administrator. In the target identity's metadata definition, explicitly grant the WriteMetadata permission to the delegated administrator.

Where Is User Administration Performed?

Metadata-layer user administration is performed as follows:

- To manage identity information interactively, use SAS Management Console. For more information, see [SAS Management Console: Guide to Users and Permissions](#).
- To import identity information in bulk from an external user store (such as Active Directory) to the SAS metadata, write SAS code. For more information, see ["User Import Macros" on page 317](#).
- To copy identity metadata from one SAS repository to another, use the metadata promotion tools. For more information, see ["Promotion Tools Overview" in SAS Intelligence Platform: System Administration Guide](#).
- To audit changes to metadata identity definitions, use the Audit.Meta.Security.GrpAdm and Audit.Meta.Security.UserAdm log categories. For more information, see ["Auditing" on page 4](#).

See Also

- [“How SAS Identity Is Determined” on page 144](#)
- [“PUBLIC Access and Anonymous Access” on page 164](#)

Metadata Users

In general, each SAS user has identity information in two distinct realms:

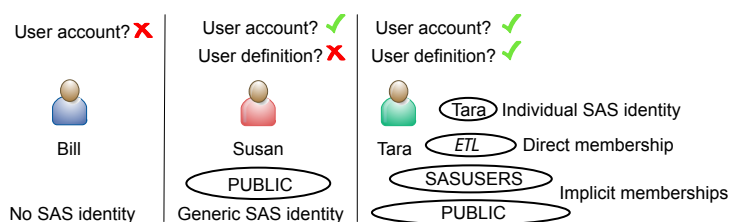
- 1 In an authentication provider, the user has an account that can access the SAS Metadata Server.
- 2 In the SAS metadata, the user has a definition that includes a copy of the account ID with which the user accesses the Metadata Server.

Coordination between these two realms establishes a unique SAS identity for each user. Each SAS identity is based on a match between the following two values:

- the account ID with which the user authenticates
- the account ID that is listed in the user's metadata definition

In the following figure, *account* refers to a user account in an authentication provider, and *definition* refers to a metadata object that represents the user. Bill cannot log on, Susan has only the generic PUBLIC identity, and Tara has an individual SAS identity.

Figure 2.1 Examples: User Accounts and User Definitions



Here are some tips for working with user definitions:

- If the SAS Metadata Server runs on Windows and uses SAS authentication, the SAS copy of each user's Windows user ID must be stored in a fully qualified format (for example, *WindowsDomain\user-ID*, *MachineName\user-ID*, or *user-ID@company.com*).
- If you find that a user has only the PUBLIC identity even though the user has a user definition, the user's stored account ID might be missing, not accurately entered, or not in the correct format. Passwords and authentication domain assignments are never the cause of this problem. The match is based only on the account ID.

- Regular users (non-administrators) can maintain their own logons, but cannot make other changes to their definitions.
- Permission settings on a user definition do not determine what that user can do. Those settings can affect the ability of other identities to update or delete the user definition itself. Special rules automatically protect user, group, and role definitions.

The following table introduces some of the predefined users:

Table 2.2 *Selected Predefined Users*

User	Description
 SAS Administrator	A predefined super user that has unrestricted access in the metadata layer.
 SAS Demo User	A predefined first user that can be useful for demonstrations.
 SAS Trusted User	A service identity that can act on behalf of other users.
 SAS Anonymous Web User	A service identity that can provide anonymous access for a few web components. ¹

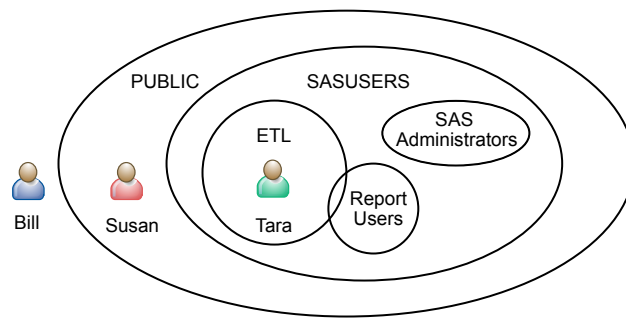
¹ For only SAS BI Web Services and the SAS Stored Processes application. For more information, see [“PUBLIC Access and Anonymous Access” on page 164](#).

See Also

- [“About User Administration” on page 10](#)
- [“Create Metadata User Definitions” on page 42](#)

Metadata Groups

Groups are primarily used in access controls, because it is more efficient to assign permissions to groups than to individual users. You can also use a group to populate a role or to make a shared credential available to multiple users. The following figure illustrates how the users in the previous topic might participate in a group structure:

Figure 2.2 Example: Users in a Group Structure

Here are some tips for working with group definitions:

- You can create a nested group structure by making one group a member of another group.
- Most groups do not have logons (stored credentials). A group logon makes a shared external account available to all members of the group. Such outbound logons typically provide access to a third-party database server and should include both a user ID and a password (as well as an authentication domain).
- Permission settings on a group definition do not determine what that group can do. Those settings can affect the ability of other identities to update or delete the group definition itself. Special rules automatically protect user, group, and role definitions.

The following table introduces some of the predefined groups.

Table 2.3 Selected Predefined Groups

Group	Description
PUBLIC	Automatically includes everyone who can access the SAS Metadata Server, either directly or through a trust relationship. A user who does not have an individual identity has only the PUBLIC group identity.
SASUSERS	Automatically includes those members of the PUBLIC group who have an individual identity. All members of the SASUSERS group are also members of the PUBLIC group.
SAS Administrators	A standard group for metadata administrators. In a standard configuration, membership in this group provides broad access and most administrative capabilities, but does not provide unrestricted status.
SAS General Servers	A service group that enables its member (the SAS Trusted User) to see server launch credentials. ¹

Group	Description
SAS System Services	A service group that enables its member (the SAS Trusted User) to see servers, cubes, and other objects.

- 1 Server launch credentials (for example, the SAS Spawnd Servers account, sassrv) are stored in logons on the **Accounts** tab of the SAS General Servers group. This facilitates launching of stored process servers and pooled workspace servers.

See Also

- [“Identity Hierarchy” on page 20](#)
- [“Differences between Roles and Groups” on page 17](#)

Metadata Roles

Role Definitions

Roles control the availability of application features such as certain menu items, plug-ins, and buttons. In the initial configuration, registered users have almost all non-administrative capabilities. In many cases, this is appropriate and sufficient. However, you can choose to alter the initial configuration by using either or both of the following techniques:

- To increase or reduce the availability of a role, change the assignment of members to the role.
- To redistribute capabilities, change the assignment of capabilities to roles, or create additional roles.

Here are some tips for working with roles:

- Before you make changes, make sure you have a complete and current backup.
- Capability assignments can be redundant across roles. To prevent someone from having a capability, you must make sure they are not in any role that provides that capability.
- Never change the name of a predefined role.
- There are no negative capabilities (capabilities that limit what someone can do). You cannot deny a capability to anyone.
- Some roles have implicit capabilities (capabilities that are bound to the role and are not displayed in the user interface).

- Contributing role relationships are monolithic. You cannot deselect a contributed capability. These relationships are also dynamic. A change to the capabilities of one role affects any roles to which the first role contributes its capabilities.
- Permission settings on a role definition do not determine what that role can do. Those settings can affect the ability of other identities to update or delete the role definition itself. Special rules automatically protect user, group, and role definitions.

Predefined Roles

The predefined roles and capabilities are described in the administrative documentation for the associated application. The following table helps you locate documentation for predefined roles.

Table 2.4 *Documentation of Predefined Roles*

Application	Documentation of Predefined Roles
Metadata Server	SAS Intelligence Platform: System Administration Guide
Desktop Applications	SAS Intelligence Platform: Desktop Application Administration Guide
Web Applications	SAS Intelligence Platform: Web Application Administration Guide
Visual Analytics: Basic	SAS Visual Analytics: Administration Guide
Visual Analytics: Report Viewing	SAS Visual Analytics: Administration Guide
Visual Analytics: Analysis	SAS Visual Analytics: Administration Guide
Visual Analytics: Data Building	SAS Visual Analytics: Administration Guide
Visual Analytics: Administration	SAS Visual Analytics: Administration Guide

Differences between Roles and Groups

Roles and groups serve distinct purposes. You cannot assign permissions to a role or capabilities to a group. Here are some additional distinctions:

- The identity hierarchy is relevant for groups, but not for roles. If you are a member of a role, you have all of that role's capabilities, regardless of whether you are a direct member of that role and what your other memberships are.
- You can deny a permission to a group, but you cannot deny a capability to a role. Each role either provides or does not provide each capability. No role takes capabilities away from its members.
- A group's permissions are not displayed as part of a group definition, but a role's capabilities are displayed as part of a role definition.
- A group can be a member of another group, but a role cannot be a member of another role. Instead, one role can contribute its capabilities to another role.

Availability of Application Features in a New Deployment

In general, the initial configuration provides appropriate access to application features. Most nonadministrative capabilities are available to either PUBLIC (everyone who can access the SAS Metadata Server) or SASUSERS (those members of PUBLIC who have a well-formed user definition). To ensure appropriate availability of features for your applications, see the administrative documentation for each application.

Password and Secret Management

Password Policies

Each authentication provider sets password policies for accounts in that provider. For example, the password expiration policy for a host account is determined by that host.

For the SAS internal authentication provider, you can set server-level policies (in the Metadata Server's omaconfig.xml file) and per-account policies (in a user's

metadata definition). For more information, see [“How to Change Internal Account Policies” on page 239](#).

Client-Side Storage of Passwords

In the initial configuration, users can choose to store their credentials in their client-side connection profiles. When credentials are stored, the logon dialog box in desktop applications is prepopulated.

For most desktop applications, the `SASSEC_LOCAL_PW_SAVE=` option controls the availability of a check box that enables users to choose whether to store credentials locally. To prevent users from creating a local copy of their credentials, set `SASSEC_LOCAL_PW_SAVE="N"` (or `"0"` or `"F"`) in the SAS Metadata Server's `omaconfig.xml` file and restart the server.

Note: A change to the `SASSEC_LOCAL_PW_SAVE=` setting takes effect after the Metadata Server is restarted. Each client uses the previous setting for its first connection, discovers the revised Metadata Server setting, and conforms to that revised setting for subsequent connections. If you change the setting to disallow saved credentials and if credentials are already present in a user's connection profile, you must remove those credentials manually.

The full-featured clients of selected SAS solutions (such as SAS Model Manager, SAS Enterprise Miner, and SAS Forecast Studio) include the ability to store credentials in client-side connection profiles. However, in these applications, credential storage is enabled by setting the `Policy.AllowClientPasswordStorage` property. This property is available on the **Plug-ins** tab in SAS Management Console. Take the following steps:

- 1 Log in to SAS Management Console, and navigate to the **Plug-ins** tab.
- 2 Select **Application Management** ⇒ **Configuration Manager** ⇒ **SAS Application Infrastructure**.
- 3 Right-click **SAS Application Infrastructure** and select **Settings** ⇒ **Policies** from the menu.
- 4 Update the value for the property **Allow client password storage**.
- 5 Click **OK** to return to the SAS Management Console main window.

External Logon Passwords

In most cases, the SAS copy of an external account includes only a user ID. For these cases, no password updates in metadata are necessary.

For any external passwords that are stored in the SAS metadata, updates are driven by changes that first occur in the external authentication provider. For example, if a copy of the password for an Oracle account or a host account is stored in a group logon, you must maintain that copy so that it always matches the actual password. Any change to the actual password (in Oracle) must be followed by a corresponding update to the SAS copy of the password (in the group logon in the SAS metadata).

You can update stored passwords in SAS Management Console. If you own logons that include passwords, you can also update those passwords in SAS Personal Login Manager. For example, to update the SAS copy of an external password in SAS Management Console, navigate to the owning user or group definition, select the **Accounts** tab, select a logon, and click **Edit**.

Internal Account Passwords

Every SAS internal account has a password. By initial policy, these passwords do not expire.

To update a SAS internal password in SAS Management Console, navigate to the owning user definition, select the **Accounts** tab, and click **Update** (at the bottom of the tab). If you have your own SAS internal account, you can also update your internal password in SAS Personal Login Manager.

TIP If repeated attempts to log on with an internal account fail, that account might be locked. For more information, see [“Unlock an Internal Account” on page 47](#).

Managed Passwords

Passwords for a few service accounts require special coordination because these passwords are included in configuration files. To update these passwords, use the SAS Deployment Manager.

Concealing Passwords and Secrets

Some passwords and encryption-key values might be included in SAS logs under specific circumstances. In order to prevent this behavior, you can redirect the SAS log to a file by using the PRINTTO procedure.

Starting with SAS 9.4M9, end users have the option to conceal passwords and encryption keys when selected macros assign passwords or otherwise handle this

sensitive content. When the [hot fix related to KB0044922](#) is applied, users can specify an option to blot these values. Users and administrators can enable the BLOTKEYWORDS macro system option to apply blotting in the output of MLOGIC, MPRINT, or SYMBOLGEN.

BLOTKEYWORDS must be enabled at start-up, but it can be enabled independently of other system options. By contrast, MLOGIC, MPRINT, and SYMBOLGEN can be enabled or disabled throughout a SAS session, and their status does not affect the setting selected for BLOTKEYWORDS. The blotting behavior has no effect if SAS programs do not include the components that might have produced cleartext passwords or other secrets in logs.

For more information, see “BLOTKEYWORDS Macro System Option” in *SAS Macro Language: Reference*.

See Also

- [“SAS Internal Authentication” on page 178](#)
- [“About Updating Managed Passwords” on page 44](#)

Identity Hierarchy

The identity hierarchy can affect authorization decisions and logon priority (in credential retrieval from the SAS metadata). The identity hierarchy is not relevant for roles.

The identity hierarchy establishes the following precedence ranking:

- 1 the user’s individual identity, based on the user’s authenticated ID.
- 2 a group that has the user as a direct member. This is a first-level group membership for the user.
- 3 a group that has another user group as a direct member. For example, assume that the user belongs to a group named ETL_Advanced, and that group is a member of another group called ETL_Basic. In that case, the ETL_Basic group is a second-level group for that user. If you have additional levels of nesting, each successive level has less precedence.
- 4 the SASUSERS implicit group, which includes everyone who has an individual identity.
- 5 the PUBLIC implicit group, which includes everyone who can access the SAS Metadata Server (regardless of whether they have an individual identity or not).

The following table provides examples of the hierarchy:

Table 2.5 Examples of Identity Hierarchies

Scenario	User's Identity Hierarchy
User has no individual identity.	Primary identity: PUBLIC
User has an identity and no explicit group memberships.	Primary identity: self First-level memberships: SASUSERS Second-level memberships: PUBLIC
User is a direct member of GroupA and GroupB. GroupA is a member of the Report Users group.	Primary identity: self First-level memberships: GroupA, GroupB Second-level memberships: Report Users Third-level memberships: SASUSERS Fourth-level memberships: PUBLIC

TIP To avoid introducing unnecessary complexity, do not make PUBLIC or SASUSERS a member of another group. For example, if you make PUBLIC a member of GroupA, then a user who is an indirect member of GroupA (through his automatic membership in PUBLIC) has GroupA as his lowest precedence membership. This contradicts the usual expectation that every user's lowest precedence membership is PUBLIC. It is not a problem for PUBLIC or SASUSERS to be a member of a role.

Note: Avoid creating circular or reciprocal group memberships (for example, GroupA is in GroupB, GroupB is in GroupC, and GroupC is in GroupA). Such relationships introduce unnecessary complexity and can negatively impact performance. When you define nested group memberships (interactively or through importing identity information), the best practice is to use as simple and non-redundant a structure as possible.

See Also

- [“Two Relationship Networks” on page 57](#)
- [“Authorization Decisions” on page 75](#)

External Identities

An external identity is a synchronization key that facilitates coordination between identity entries in the metadata and identity entries in your authentication provider. If you use batch processes to coordinate metadata identity information with your authentication provider, external identities are set up and used as follows:

- 1 In your authentication provider, you select a field to use for the mapping. This should be a field that contains a unique and unchanging value for each user, group, and role that you want to manage with batch processes. Typically, this is an identifier such as employee number.
- 2 When you perform an initial import from your authentication provider into the metadata, the keyid values in the canonical tables become external identity values in the metadata. Each imported identity has at least one external identity value.
- 3 During the synchronization process, external identity values that are extracted from the metadata are used as the keyid in the target tables. Because these values also exist in the extraction from your authentication provider, external identity values can be used to match corresponding entries in the two sets of tables.

Note: If you need to incorporate manually created identities into a batch synchronization process, manage each identity's external identity value from the **General** properties of its metadata definition.

See Also

[“User Import Macros” on page 317](#)

Windows Privileges

Access This Computer from the Network

Table 2.6 *Access This Computer from the Network*

Description	This privilege is required in order to connect to SAS servers.
To Whom	Give this privilege to all users who access SAS servers on Windows.
How	Typically, this right is already granted to the Windows group Everyone . To confirm, check the Windows local policy settings.

Log on as a Batch Job

Table 2.7 *Log on as a Batch Job*

Description	This privilege is required in order to run a stored process server or any type of workspace server.
To Whom	On the Windows computer that hosts the SAS object spawner, give this privilege to the accounts under which workspace servers and stored process servers run: ■ any service account under which one of these servers run ■ all puddle logons for any client-side pooled workspace servers ■ any user accounts under which a standard workspace server runs (users who authenticate by Integrated Windows authentication or SAS token authentication do not need this privilege)
How	Modify the local security policy. For example, on Windows 10, this right is managed from the Local Group Policy Editor (gpedit.msc) under Computer Configuration ⇒ Windows Settings ⇒ Security Settings . If you have an operating system group (such as SAS Server Users) that has this right, you just add users and service account identities to that group.

Trusted for Delegation

Table 2.8 *Trusted for Delegation*

Description	This privilege enables a process to allow each user's credentials to be sent to further machines for authentication (for example, to access a UNC path). The privilege is needed if the workspace server is accessed through Integrated Windows authentication and provides access to network resources. Note: With Integrated Windows authentication, the workspace server does not receive the requesting user's credentials, so the workspace server cannot provide credentials for downstream servers. Instead, the spawner account must be trusted to delegate each requesting user's identity as necessary.
To Whom	If the workspace server runs on Windows, give this privilege to the account under which the object spawner runs. By default, the spawner runs as a service under the local system account, so the computer account for spawner's host needs the privilege. If the workspace server runs on UNIX, give this privilege to the service principal account that is referenced in the relevant keytab. The keytab is based on service principal names that correspond to a particular service principal account. For more information, see the chapter "Configuring Integrated Windows Authentication" in <i>Configuration Guide for SAS Foundation for UNIX Environments</i> at http://support.sas.com/documentation/installcenter/).
How	As a Windows domain administrator, under Start ⇒ Control Panel ⇒ Administrative Tools ⇒ Active Directory Users and Computers, access the properties dialog box for the relevant account and grant the privilege. If your spawner runs on Windows under the local system account, select the spawner host machine in Active Directory under Computers. On the Delegation tab (or the General tab), select the Trust this computer for delegation to any service (Kerberos only) check box. If your spawner runs on Windows under a domain account, select that account in Active Directory under Users. On the Delegation tab (or the Accounts tab), select the Account is trusted for delegation check box. If your spawner runs on UNIX, select the appropriate service principal account in Active Directory under Users. On the Delegation tab (or the Accounts tab), select the Account is trusted for delegation check box.

Note: In most cases, an object spawner on Windows runs as a service under the local system account. If the spawner instead runs under some other account, that account must be a Windows administrator on the spawner's host and have the Windows user rights **Adjust memory quotas for a process**, **Replace a process level token**, and **Act as part of the Operating System**. These user rights assignments are part of the local security policy for the Windows computer that hosts the spawner.

See Also

- “Host Authentication” on page 172
- “Integrated Windows Authentication” on page 174

Windows User ID Formats

In most cases, users can launch SAS applications using the same ID and password as they use in the rest of your computing environment. However, when you create a SAS copy of a Windows user ID, you must store the user ID in the appropriate format. In most cases, you must use a particular qualified format (for example, *WindowsDomain\user*, *MachineName\user*, or *user@company.com*). With certain authentication configurations, a different format is required. Failure to appropriately qualify a stored user ID causes the user to have only the PUBLIC identity.

If your site accepts Windows IDs in disparate formats, you must coordinate the format of the copies with the format in which users submit their IDs. This table describes the common forms for an Active Directory user ID:

Table 2.9 Overview: Forms of an Active Directory User ID

Form	Basic Syntax	Example
Short	<i>user</i>	joe
UPN	<i>user@UPNsuffix</i>	joe@orionsports.com or joe@sales.orionsports.com
Down-level	<i>down-level-domain-name\user</i>	orionsports\joe or sales\joe or mymachine\joe
Kerberos	<i>user@REALM</i>	joe@ORIONSPORTS.COM

Note: User Principal Name (UPN) is an Active Directory concept. Down-level domain is a Windows NT concept. The realm in a Kerberos principal is usually a Windows domain. A Kerberos principal can include an instance (in the format *user/instance@REALM*). Additional site-specific variations might occur. The realm qualifier in a Kerberos principal is always in uppercase.

In the platform, follow these guidelines for Windows user IDs:

- If users log on interactively, they can usually use the short form. Here are some exceptions:

- ❑ The user has multiple accounts with the same user ID in different down-level domains (for example, `machine\joe`, `domain1\joe`, and `domain2\joe`).
- ❑ The site has configured direct use of LDAP and has not specified `-primpd` (the `PRIMARYPROVIDERDOMAIN` system option).
- If users log on interactively, they can also use one other site-supported form (either the UPN form or the down-level form). Use one of these approaches:
 - ❑ In the metadata, store each user ID in UPN form. Tell users not to use the down-level form when they log on.
 - ❑ In the metadata, store each user ID in down-level form. Tell users to not use the UPN form when they log on.
- If users log on to SAS desktop applications through Integrated Windows authentication, their user IDs should usually be stored in down-level form. In general, that is the form in which SAS obtains user IDs after Kerberos authentication occurs.

Note: If you prefer to store user IDs in the native Kerberos form, add the setting `SASUSEKERBNAME true` as a Windows system environment variable on the server host. Setting `SASUSEKERBNAME false` does not disable the variable. If the `SASUSEKERBNAME` variable is not needed, remove the variable. For example, on the Windows desktop, right-click **My Computer**, select **Properties**, select the **Advanced** tab, click **Environment Variables**, and add the setting under **System variables**. Then reboot the machine. This setting affects only connections that use Integrated Windows authentication. If you use this setting, you might want to make sure that the Integrated Windows authentication process always selects the Kerberos protocol.

- If users log on to SAS web applications through Integrated Windows authentication (which occurs only if you configure web authentication and have set up Integrated Windows authentication with your web provider), the form of the returned user ID might differ. For more information, see the documentation for your web application server.

Note: In the status bar of some applications, a currently connected Windows user ID is always displayed in the format `user@VALUE`, regardless of how the user logged on or how the user's ID is stored in the metadata. For example, if you log on as *Joe* and your stored user ID is `WIN\joe`, the status bar displays your authenticated ID as `joe@WIN`.

See Also

- [“Direct LDAP Authentication” on page 171](#)
- [“How SAS Identity Is Determined” on page 144](#)
- [“Integrated Windows Authentication” on page 174](#)
- [“Authentication to the Metadata Server” on page 144](#)

Uniqueness Requirements

The SAS Metadata Server enforces the following identity-related constraints:

- You cannot create a user definition that has the same name as an existing user definition. The display names do not have to be unique.

TIP SAS recommends that you avoid using spaces or special characters in the name of a user, group, or role. Not all components support spaces and special characters in identity names.

- You cannot change the name of an existing user, group, or role in SAS Management Console. You can change the display name.
- You cannot create a group or role definition that has the same name as an existing group or role definition. The display names do not have to be unique.
- You cannot assign the same external account ID to two different identities. All of the logons that include a particular ID must be owned by the same identity. This requirement enables the Metadata Server to resolve each ID to a single identity. However, a logon that is associated with an outbound authentication domain is exempt from this requirement. For more information, see [“Outbound and Trusted Authentication Domains” on page 156](#).
 - This requirement is case-insensitive. For example, you cannot assign a logon with an ID of *smith* to one user and a logon with an ID of *SMITH* to another user.
 - This requirement applies to the fully qualified form of the ID. For example, you can assign a logon with an ID of *winDEV\brown* to one user and a logon with an ID of *winPROD\brown* to another user. In this example, *winDEV* and *winPROD* are Windows domain names, which are incorporated into the fully qualified form of an external account ID.
 - This requirement cannot be mitigated by associating the logons with different SAS authentication domains. For example, assume that one user has a logon with an ID of *smith* that is associated with a SAS authentication domain named *DefaultAuth*. In that case, you cannot give any other user a logon with the ID *smith*, even if you plan to assign the logon to a different SAS authentication domain.

TIP To enable multiple users to share an account, store the credentials for that account in a logon as part of a group definition. Then add the users who share the account as members of that group definition.

- If you give a user two logons that contain the same ID, the logons must be associated with different authentication domains. Within an authentication domain, each ID must be unique. For example, if you give the person *Tara O'Toole* two logons that both have an ID of *tara*, then you cannot associate both of those logons with the *OraAuth* authentication domain.

Note: Like the previous requirement, this requirement is case-insensitive.

Note: You cannot circumvent this constraint by supplying the same user ID in different formats. For example, if you give a user a logon that contains the user ID *WIN\user* and another logon that contains the user ID *user@WIN*, those logons must be in different authentication domains.

- A user can have multiple locations, email addresses, and telephone numbers. However, each user can have only one item of a given type. For example, a user can have one *home* email address and one *work* email address, but not two *work* email addresses.

See Also

- [“Create Metadata User Definitions” on page 42](#)
- [“How SAS Identity Is Determined” on page 144](#)

Configuring Client Machines to Use Integrated Windows Authentication

About Configuring Client Machines to Use IWA

This information applies to client machines that are running Java rich clients, including those that might run clients launched through Java Web Start. In addition to enabling the **Use Integrated Windows authentication (single sign-on)** option during installation, there are manual steps that must be completed from each client machine. The following sections provide information about the manual steps that must be completed.

Updating Registry Settings

By default, Windows does not allow Java to access the session key that is associated with the ticket-granting ticket (TGT) in the cache. This might cause the Java client to prompt the user for their password, or it might refuse to authenticate with the server. To enable Java access to the session key, complete one of the following workarounds:

- Update the Windows registry. Microsoft recommends performing a system backup before editing the registry. Access the `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Lsa\Kerberos\Parameters` registry subkey. Set the following values:

Table 2.10 Windows Registry Values

Name	allowtgtsessionkey
Type	DWORD Note: Starting at Windows 10, set the type to REG_SZ.
Value	1

For more information about setting the registry values, see [this article from Microsoft](#).

- Run the `kinit -f` command to get a new TGT and session key from the Key Distribution Center and cache it on the file system.

Note: The TGT has a limited lifespan. If you select this workaround, you must repeat this process periodically.

Note: Use the `kinit` command that is provided in the `SASHOME\SASPrivateJavaRuntimeEnvironment\9.4\jre\bin` directory.

Configuring the Java Cryptography Extension

Determining Whether a Machine Is Using AES-256

In order to support IWA, the Java environment must include the Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy files if the machine is using AES-256 encrypted tickets or session keys. By default, these files are present and AES-256 is enabled. Kerberos uses AES-256 in the *aes256-cts-hmac-sha1-96* encryption type that is associated with the keytab.

Determine whether a machine is using AES-256 encrypted TGT by running the `klist` command. The command displays currently cached Kerberos tickets. Here is an example of the output that you might see:

```
C:\Users\user_name>klist
```

```
Current LogonId is LUID
```

```
Cached Tickets: (number)
```

```
#0>      Client: user_name @ domain
        Server: krbtgt/domain @ domain
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x60a00000 -> forwardable forwarded renewable pre_authent
        Start Time: 5/12/2014 11:09:03 (local)
        End Time:    5/12/2014 19:50:18 (local)
        Renew Time: 5/19/2014 9:50:18 (local)
        Session Key Type: AES-256-CTS-HMAC-SHA1-96
```

```
#1>      Client: user_name @ domain
        Server: krbtgt/domain @ domain
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x40e00000 -> forwardable renewable initial pre_authent
        Start Time: 5/12/2014 9:50:18 (local)
        End Time:    5/12/2014 19:50:18 (local)
        Renew Time: 5/19/2014 9:50:18 (local)
        Session Key Type: AES-256-CTS-HMAC-SHA1-96
```

Note: Use the `klist` command that is provided in the `SASHOME`
`\SASPrivateJavaRuntimeEnvironment\9.4\jre\bin` directory.

Enabling AES-256

SAS uses the Unlimited Strength Java Cryptography Extension (JCE) policy files that are present by default and that enable AES-256 for encryption of data at rest.

If necessary, you can download the Zulu Cryptography Extension Kit that provides JCE policies. The kit is available from [the Azul support website](#).

Install the files from the kit into the directory where the JRE is configured. For a SAS installation, the default JRE is located in `SASHome\SASPrivateJavaRuntimeEnvironment\9.4\jre`. For client machines that use Java Web Start, the JRE location depends on the Java Web Start configuration.

Verify the JRE settings in the Java Control Path by navigating to **Control Panel** ⇒ **Java**.

See Also

- [“Integrated Windows Authentication” on page 174](#)
- [“How to Configure Integrated Windows Authentication” on page 217](#)

Access Management

About Access Management	33
Introduction	33
Who Can Set Permissions?	34
Where Is Access Management Performed?	34
Basics of Metadata Authorization	35
Where Permissions Are Set	35
Permission Assignments	35
How Permissions Are Set	36
How Permissions Differ from Capabilities	37
See Also	38
WriteMetadata and WriteMemberMetadata	38
See Also	39
Review: Key Points about Authorization	39
See Also	40

About Access Management

Introduction

The platform provides a metadata-based authorization layer that supplements protections from the host environment and other systems. You can use the metadata authorization layer to manage access to almost any metadata object (for example, reports, data definitions, information maps, jobs, stored processes, and server definitions). For information specific to SAS Visual Analytics, see [SAS Visual Analytics: Administration Guide](#).

Across authorization layers, protections are cumulative. In order to perform a task, a user must have sufficient access in all applicable layers.

In the metadata layer, the following permissions are always enforced:

- the ReadMetadata permission (RM), which controls the ability to see an object
- the WriteMetadata permission (WM), which controls the ability to update or delete an object

Other permissions are specialized and affect only certain types of objects.

CAUTION

In the metadata authorization layer, not all permissions are enforced for all items. It is essential to understand which actions are controlled by each permission. For more information, see [“Use and Enforcement of Each Permission” on page 84](#).

CAUTION

Some clients enable power users to create and run SAS programs that access data directly, bypassing metadata-layer controls. It is essential to manage physical layer access in addition to metadata-layer controls. For more information, see [Access to SAS Data on page 111](#).

Who Can Set Permissions?

Table 3.1 Requirements for Setting Permissions

Task	Requirements
Set permissions on an item	WriteMetadata for the item
Change the permission pattern on an ACT	WriteMetadata for the ACT
Designate a different repository ACT	WriteMetadata for the ACT

Note: In SAS Management Console, you cannot see the **Authorization Manager** or any **Authorization** tabs unless you have the Authorization Manager capability.

Where Is Access Management Performed?

Metadata-layer access management is performed as follows:

- To manage access interactively, use SAS Management Console (a desktop application). For more information, see [SAS Management Console: Guide to Users and Permissions](#).
- To manage metadata access, use [SAS Environment Manager](#) (a web-based solution).

- To programmatically define or query authorization settings, use either batch tools (see [Batch Tools for Metadata Authorization on page 131](#)) or DATA step functions (see [SAS Language Interfaces to Metadata](#)).
- To audit changes to access controls, use the Audit.Meta.Security.AccCtrlAdm log category. For more information, see [“Auditing” on page 4](#).
- To extract authorization information from the SAS metadata into SAS data sets, use the %MDSECDS macro. For more information, see [“Overview of the Security Report Macros” on page 345](#).
- To copy metadata access controls from one SAS repository to another, use the metadata promotion tools. For more information, see [“Promotion Tools Overview” in SAS Intelligence Platform: System Administration Guide](#).

Basics of Metadata Authorization

Where Permissions Are Set

Effective access to an object is displayed in the authorization properties section of the object's metadata definition. You can set permissions on individual content objects (such as reports) and on containers (such as folders). For simplicity, try to set permissions on containers, instead of on individual content objects, whenever possible.

TIP The metadata authorization model is object-centric, not identity-centric. To examine a user's permissions, do not begin by finding their user definition. Instead, begin by navigating to the object that you want to examine.

Permission Assignments

You can assign permissions to individual users and to groups. For simplicity, try to avoid assigning permissions to individual users.

The main authorization list includes only those users and groups that participate in access controls that could affect the object. The list usually includes at least the following groups:

PUBLIC

automatically includes everyone who can access the metadata server.

SASUSERS

automatically includes everyone who is registered in the metadata (SASUSERS is a subset of PUBLIC).

SAS Administrators

should include only metadata administrators that require broad access.

SAS System Services

should include only the service identity (or, identities) that require broad access.

Anyone who is not listed in the basic authorization properties display has the access of their closest listed group, as determined by group memberships and identity precedence. Here are some examples:

- The closest (and only) listed group for an unregistered user is PUBLIC.
- The closest listed group for a registered user is often SASUSERS.
- The closest listed group for an administrator is usually SAS Administrators.

How Permissions Are Set

Use a combination of the following techniques to manage access:

direct controls

You can assign grants and denials directly on a target object. There are two types of direct controls:

explicit controls

are individual grants and denials.

access control template (ACT) controls

are grants and denials within a predefined pattern.

TIP For simplicity, try to avoid setting unnecessary (redundant) direct controls.

object inheritance

You can assign grants and denials on a parent object, and rely on inherited settings to protect each child object. For example, a report inherits effective permissions from its parent folder. In the metadata layer, access control inheritance happens automatically and cannot be turned off. All objects have inherited settings. Most objects have inherited settings only.

The following table summarizes the three types of access controls:

Table 3.2 *Direct Access Controls and Indirect Settings*

Term	Significance
Direct control: explicit	An access control is explicit for a particular identity if the control is set directly on the target object and assigned directly to that

Term	Significance
	identity. For example, if an individual access control on ReportA grants the ReadMetadata permission to UserA, then we say that UserA has a direct control on ReportA (specifically, an explicit grant of the ReadMetadata permission). Explicit controls are sometimes referred to as ACEs (access control entries).
Direct control: ACT	An access control is a direct ACT control for an identity if the control comes from an ACT that is directly applied to the target object, with a pattern that directly assigns a grant or denial to that identity. For example, if the DemoACT is directly applied to FolderA, and the DemoACT's permission pattern explicitly grants the ReadMetadata permission to GroupA, then we say that GroupA has a direct control on FolderA (specifically, an ACT-based grant of the ReadMetadata permission).
Indirect setting	An indirect setting comes from someone else (a group that has an explicit or ACT setting), from somewhere else (a parent object, the repository ACT), or from a special status (such as unrestricted). For the WriteMemberMetadata permission, indirect means that the setting mirrors the WriteMetadata setting. Note: The authorization model includes two distinct relationship networks: object inheritance (for example, a folder and its contents) and identity precedence (for example, a group and its members). A setting that is derived through either of these hierarchies is called an indirect setting.

How Permissions Differ from Capabilities

You can use metadata-based roles to control the visibility of application features (such as buttons, tabs, and menu items) to a metadata group or user. An application feature that is under role-based management is called a *capability*.

IMPORTANT Capabilities provide restricted application options to selected users and groups, but these restrictions should not be mistaken for security or authorization.

Having a certain capability is not an alternative to meeting permission requirements. Permission requirements and capability requirements are cumulative. For example, even if you are in the **Enterprise Guide: OLAP** role, you cannot access data in a cube if you lack the Read and ReadMetadata permissions for it.

Use roles and permissions in conjunction with one another. You can use permissions to constrain the scope of a role. For example, to enable a user to create reports but

add them to a single folder, give the user the **Web Report Studio: Report Creation** role, but grant them the WriteMemberMetadata permission on that folder only.

See Also

- [“Metadata Roles” on page 15](#)
- [“Identity Hierarchy” on page 20](#)
- [“Object Inheritance” on page 59](#)
- [“Authorization Decisions” on page 75](#)
- [“Use and Enforcement of Each Permission” on page 84](#)

WriteMetadata and WriteMemberMetadata

You can use the WriteMetadata (WM) and WriteMemberMetadata (WMM) permissions to enable someone to interact with a folder’s contents, but not update or delete the folder itself. On the folder, grant WMM and deny WM.

The following list explains the difference between these two permissions.

WriteMetadata (WM)

provides general control for additions, deletions, and updates. For example, to edit a report, you need WM for the report. To delete a report, you need WM for the report (and WMM for the report’s parent folder). For containers other than folders (such as repositories, libraries, and schemas), WM also affects adding and deleting child objects. For example, to add an object anywhere in a repository, you need WM at the repository level. For folders, adding and deleting child objects is controlled by WMM, not WM.

WriteMemberMetadata (WMM)

provides specialized control for adding and removing objects in a folder. You need WMM on a folder in order to add an object to the folder or delete an object from the folder. For example, to save a report to a folder, you need WMM for the folder. To remove a report from a folder, you need WMM for the folder (and WM for the report).

TIP We recommend that anyone who has a grant of WM is not denied WMM.

TIP If WMM is not directly set on a folder, the WMM setting mirrors the WM setting. WMM is never inherited from a parent object.

See Also

- [“WriteMetadata and WriteMemberMetadata” in SAS Management Console: Guide to Users and Permissions](#)

Review: Key Points about Authorization

In order to effectively use the metadata authorization layer, you must understand the following points:

- Metadata layer permissions supplement protections from the host environment and other systems. Across authorization layers, protections are cumulative. In order to perform a task, a user must have sufficient access in all applicable layers.
- The displayed effective permissions are a calculation of the net impact of all applicable permission settings in the metadata layer. However, the display does not reflect access in other layers such as the operating system.
- Setting permissions is an object-centric activity. To define permissions for someone, do not begin by finding that person's user definition. Instead, begin by navigating to an object that you want to protect or make available.
- Explicit and ACT controls on an object (such as a report) always have priority over controls on the object's parent (such as a folder). For example, if a report has an explicit denial for PUBLIC and the report's folder has an explicit grant for you, the result is a denial for you. The direct denial for PUBLIC blocks access to the report (for all restricted users), unless you offset the direct denial by adding direct grants on the report.
- In a user definition, the displayed permission settings have no effect on what the user can see or do. Those controls determine who can update or delete the user definition itself.
- Only the ReadMetadata and WriteMetadata permissions are universally relevant and enforced.
- The authorization layer that SAS provides supports very specific settings (for example, *on ReportA, grant the ReadMetadata permission to UserA*). However, for simplicity, it is preferable to set permissions at a less granular level whenever possible. The recommended approach is to create a folder structure and set permissions so that each folder offers appropriate group-level access to

its contents (for example, on *FolderA*, grant the *ReadMetadata* permission to *GroupA*).

- After you add a broad denial, review the impact that this has on everyone else. For example, if the only setting on an object is an explicit PUBLIC denial, that denial blocks access for everyone (other than unrestricted users). To offset the denial, add one or more selective explicit (or ACT) grants.
- Before you deny the ReadMetadata permission on a folder, consider the navigational consequences. Without ReadMetadata permission on a folder, you cannot navigate to objects beneath that folder. Users need a clear path of grants of ReadMetadata permission in order to navigate to the content that they use.

See Also

- [“Adjust Initial Access” on page 49](#)
- [“Use and Enforcement of Each Permission” on page 84](#)

Selected Tasks

About Security Task Instructions	41
Create Metadata User Definitions	42
Add Regular Users	42
Add Administrators	43
Add Dual Users	43
See Also	44
Update a Managed Password	44
About Updating Managed Passwords	44
Additional Steps to Update Managed Passwords on Horizontal Cluster Nodes ...	47
Unlock an Internal Account	47
See Also	49
Adjust Initial Access	49
See Also	51

About Security Task Instructions

This chapter provides information about selected security tasks.

For information about specialized configurations, see [Encryption Tasks on page 266](#) and [Authentication Tasks on page 203](#).

For information about other user-management and access-management tasks, see [SAS Management Console: Guide to Users and Permissions](#).

Create Metadata User Definitions

Add Regular Users

For accountability, SAS recommends that you create an individual SAS identity for each person who uses the SAS environment. This strategy enables you to make access distinctions and audit individual actions in the metadata layer. It also provides a personal folder for each user. To create a SAS identity for a user, store a copy of the person's user ID in the SAS metadata.

Each user should have at least the following attributes:

- a name that is unique among users within SAS Metadata Server.
- a logon that includes the user's external account ID. This might be any type of account that is known to the Metadata Server host (an LDAP, Active Directory, host, or other type of account).

Note: For a Windows account, specify the user ID in an appropriate format. For more information, see [“Windows User ID Formats” on page 25](#).

Here are some details and tips:

- As an alternative to adding users interactively, you can batch import users from a provider such as LDAP into the SAS metadata.
- SAS Metadata Server maintains its own copy of each ID, but does not keep copies of passwords for identification purposes.
- Registered users automatically belong to PUBLIC (everyone who can access the Metadata Server) and SASUSERS (those members of PUBLIC who have a well-formed user definition).
- A user who does not have a well-formed definition has only the PUBLIC identity. In the standard configuration, a PUBLIC-only user cannot access any resources. Not all applications allow a PUBLIC-only user to log on.
- If the workspace server is on Windows, give anyone who accesses that server using credential-based host authentication the Windows privilege **Log on as a batch job**.
- Authorization settings within a user definition do not determine what that user can do. Those settings affect the ability of other users to update or delete that user definition.

Add Administrators

For accountability, it is recommended that you establish individual metadata administrators rather than sharing the unrestricted SAS Administrator account.

Each administrator should have at least the following attributes:

- a name that is unique among users within the Metadata Server
- either a logon (as explained in the preceding topic) or an internal account
- membership in the SAS Administrators group

Here are some details:

- It is recommended that you use an administrative identity only to perform tasks that require a high level of privilege.
- If you log on to SAS Management Console using an internal account, you are prompted for host credentials for certain tasks (for example, validating a standard workspace server).
- If you log on with an internal account, you must include the @saspw suffix in the user ID that you submit (for example, sasadm@saspw).

Add Dual Users

To enable someone to alternately function as an administrator and as a non-administrator, create two user definitions for that person as follows:

- One definition is based on an internal account and is a member of the SAS Administrators group.
- The other definition is based on an external account and is not a member of the SAS Administrators group.

Here are some tips for working with a dual user:

- The only way to make someone a dual user is to give that person two user definitions, each based on a different account. You cannot create a dual user by adding a logon to a definition that already has an internal account or by adding two logons to one definition.
- A dual user should use a dedicated client-side connection profile for their internal account. In that profile, the user should leave the **Authentication Domain** field blank. This optimizes credential reuse.
- A dual user should log on with their internal account when they need administrative privileges and with their external account the rest of the time.

See Also

- [“Metadata Users” on page 12](#)
- [“Windows Privileges” on page 23](#)
- [“Windows Privileges” on page 23](#)
- [“Windows User ID Formats” on page 25](#)
- [“Identify or Create User Accounts” on page 203](#)
- [“Logons for Users Who Participate in Web Authentication” on page 207](#)

Update a Managed Password

About Updating Managed Passwords

Passwords for a few service accounts require special coordination because these passwords are included in configuration files. To update these passwords, use the SAS Deployment Manager. Here are some key points about using the SAS Deployment Manager to update passwords:

- The SharedServices password is required to change the password for sasadm and SharedServices.
- The utility updates both configuration files and metadata. You can update multiple passwords in a single pass.

Some restrictions apply to the passwords for SharedServices and database administrator accounts. For more information, see [“Password Policy” in SAS Intelligence Platform: Middle-Tier Administration Guide](#).
- You must run the utility on each machine that hosts affected components. If you have servers on multiple machines, run the utility on each host, beginning with the Metadata Server machine.
- In a clustered Metadata Server environment, you must run the deployment manager on all nodes within the cluster.
- In a clustered Metadata Server environment, the SAS Deployment Agent must be running on all the nodes, and you need to update the password on all the nodes.
- It might be necessary to update the same password on multiple hosts. For example, if you update the password for the SAS Trusted User on the Metadata Server host, you must do the same update on the middle-tier machine.
- Be sure to supply the same new password for an account on all machines on which you update that account.

- If you enter a plaintext password into the utility, the utility encodes that password by using SAS proprietary encoding (SAS002). Some SAS components do not support all of the available encoding types. For more information about password management, see [“Password Policy” in SAS Intelligence Platform: Middle-Tier Administration Guide](#).
- Passwords for any service accounts that you introduce in SAS Management Console are not managed by this tool. For example, if you designate a new logon as the launch credential for a server, that launch credential is not automatically added to the list of accounts that the SAS Deployment Manager can update.

Note: Server launch credentials are not added to any configuration file, so you can update any such passwords from the owning identity's **Accounts** tab in SAS Management Console.

- You can automate running SAS Deployment Manager when you need to perform the same configuration action on many machines in your deployment. The deployment manager uses the same record and playback mechanism as the SAS Deployment Wizard to perform a non-interactive, silent configuration.

CAUTION

If you choose to use the deployment manager's record and playback mechanism to update passwords, passwords are written to the response file. For increased security, delete the response file (or remove the passwords from the response file) when you have finished. A response file is present only if you use the record and playback mechanism rather than completing the task manually.

- Each run of this utility generates an UpdatePasswords.html file that documents the updates that the utility performed and provides instructions for any required post-update activities. The file is located in your equivalent of `\SAS\Config\Lev1\Documents\`.

To update a password with SAS Deployment Manager:

Note: The procedure to update the SAS Environment Manager identity password is different from the process that is detailed here. For more information, see [SAS Environment Manager: User's Guide](#).

- 1 Identify the data server where the user ID whose managed password is changing is stored. Use an administration tool, such as PG Admin, if necessary.
- 2 Verify that the target data server or multiple servers are running. Otherwise, the update password task cannot connect to the database in order to update the credentials.

If you see a connection failure during the process, it might be because the target server is not running.

- 3 (Optional) If you are updating the password for an internal account, review the server-level password policies for internal accounts. Also, check the properties for each internal account to determine whether any more (or less) stringent requirements apply.

Note: Make sure that the account is not subject to a forced password change after the password is reset. Either set the password to never expire or change the server-level policy for pre-expired passwords.

Note: By default policy, internal passwords must be at least six characters and are not required to include mixed case or numbers. The five most recent passwords for an account cannot be reused for that account.

- 4 (Optional) If you want to use stronger encoding than SAS002 (SASProprietary), use the PWENCODE procedure. For example, to prepare an AES-encrypted version of a new password, use the following code to specify SAS003, SAS004, or SAS005 (the example uses SAS003):

```
proc pwencode in='PWsassrv1' method=sas003;
run;
```

The encoded password is written to your SAS log. When you use method=sas003, the first part of the password is {SAS003}.

Note: Outbound passwords are subject to some limitations. For more information, see [“Compatibility” on page 292](#).

- 5 Stop all SAS servers and services.
- 6 If you are updating the password for an external account (for example, sassrv), change that password in your external authentication provider (for example, in the host operating system).
- 7 Restart SAS Metadata Server, the SAS Web Infrastructure database (usually the SAS Web Infrastructure Platform Data Server), and any SAS solution-specific data servers, with names like JobMonitorDataServer, SASDataManagementDataServer, or [Name-of-Solution]DataServer. Do not restart other servers or services.
- 8 On the SAS Metadata Server host, navigate to your equivalent of **SAS-installation-directory/SASDeploymentManager/9.4/** and launch sasdm.exe (Windows), sasdm.sh (UNIX), or sasdm.rexx (z/OS).

Windows Specifics: On Windows, log on as a Windows administrator of the current machine.

Operating Environment Information: On UNIX and z/OS, log on with the user ID that was used to perform the SAS installation.

- 9 In SAS Deployment Manager, select the Update Passwords task, select a configuration directory on the current machine, and log on as an unrestricted user (for example, sasadm@saspw).
-

Note: If you are changing the password for the dbmsowner account, you must use the sasadm@saspw user. The other unrestricted users will not work.

- 10 Perform the update. If you need detailed assistance with the user interface, see the Help within the utility.
- 11 If you have servers on multiple machines, repeat steps 6–8 on each server host as applicable for the accounts that you are updating. Remember that you might have to update the same account on multiple hosts.

Note: Not all accounts are used on all hosts. If the accounts that you are updating are not on a particular host, proceed to the next host.

- 12 Restart all servers and services, and complete any additional post-update tasks as specified in the generated UpdatePasswords.html file.

Note: Because of dependencies, it is important to start servers and services in a particular order. For a complete discussion, see the section titled "Operating Your Servers" in *SAS Intelligence Platform: System Administration Guide*.

Additional Steps to Update Managed Passwords on Horizontal Cluster Nodes

If you have one or more horizontal cluster nodes, and you have run the SAS Deployment Manager to update passwords on a primary middle tier, then you must complete these additional steps to update the managed passwords on each horizontal cluster node:

- 1 Stop all the servers and services on the node.
- 2 Delete the EAR and WAR files from the *SAS-configuration-directory*Lev1/Web/Staging directory.
- 3 Run the SAS Deployment Manager and select **Update Existing Configuration**.
- 4 Restart all the servers and services on the node.

Unlock an Internal Account

By initial policy, making three consecutive unsuccessful attempts to log on with a SAS internal account causes that account to be locked for one hour. The sasadm@saspw and sastrust@saspw internal accounts are configured to be exempt from this policy. This topic explains how to unlock the account immediately so that you do not have to wait until the account lockout period has passed.

The preferred approach is to locate another user who has user administration capabilities. That user can unlock the internal account by completing these steps:

- 1 Log on to SAS Management Console as someone who is unrestricted or who has user administration capabilities.
- 2 On the **Plug-ins** tab, select **User Manager**.
- 3 In the display pane, clear the **Show Groups** and **Show Roles** check boxes. Right-click the user definition of the person whose SAS internal account is locked out and select **Properties**.
- 4 Click the **Accounts** tab. A message box asks whether you want to unlock the account. Click **Yes**.
- 5 In the user's Properties window, click **OK**. The account is now unlocked.

Note: It is not necessary to reset the user's internal password as part of unlocking the user's internal account.

If no other administrator is available and you have the necessary host access, you can use this approach as a last resort:

- 1 Edit the adminUsers.txt file on the SAS Metadata Server to create an unrestricted user.
 - a Navigate to your equivalent of `SAS/Lev1/SASMeta/MetadataServer/adminUsers.txt`. Open the file with a text editor.
 - b Add a user ID for an account that is known to the SAS Metadata Server host. Include a preceding asterisk (for example, `*WIN\winID` or `*unixID`).
 - c Stop and restart the Metadata Server to make the change take effect.

CAUTION

Stopping SAS Metadata Server stops other components.

- 2 Log on to SAS Management Console using the user ID that you added to the adminUsers.txt file. On the status bar at the bottom of the application window, verify that you are unrestricted. Unlock the locked account. See the preceding instruction list for more information.
- 3 To verify that the account is unlocked, log on to SAS Management Console using the account.
- 4 Open the adminUsers.txt file, remove the entry that you added, and stop and restart the Metadata Server.

Here are some additional tips:

- If you want to change the password for the original SAS Administrator (sasadm@saspw), you must also update the deployment to set the password, and updating the deployment requires a second password change. The password for sasadm@saspw should always be changed from SAS Deployment Manager so that other services are notified of the change.

- You can customize the account lockout policy.
- SAS recommends creating individual metadata administrator accounts rather than sharing the predefined SAS Administrator account.

See Also

- [“SAS Internal Authentication” on page 178](#)
- [“How to Change Internal Account Policies” on page 239](#)

Adjust Initial Access

The initial configuration in a new deployment provides sufficient access to data and resources, with the following exceptions:

- Only unrestricted users can access data through information maps, reports that are based on information maps, the metadata LIBNAME engine, or the OLAP server. In the initial configuration, the only grants of the Read permission are in each user's personal content area (**My Folder**).
- Only unrestricted users and members of the SAS Administrators group can register cubes.

To ensure appropriate access to resources and data:

- 1 Log on to SAS Management Console as an administrator (for example, sasadm@saspw).
- 2 (Optional) Verify that all registered users have at least the minimum required repository-level access.
 - a On the **Plug-ins** tab, under **Authorization Manager**, expand the **Access Control Templates** node.
 - b Right-click the repository ACT  (**Default ACT**) and select **Properties**.
 - c On the **Permission Pattern** tab, select **SASUSERS**. Verify that the ReadMetadata and WriteMetadata permissions are granted.
- 3 (Optional) Verify that all registered users have basic access to the folder tree.
 - a On the **Folders** tab, right-click the root folder ( **SAS Folders**) and select **Properties**.
 - b On the folder's **Authorization** tab, select **SASUSERS**. Verify that the ReadMetadata permission is granted.
- 4 Provide metadata layer access to data (this is a broad approach).

- a On the **Authorization** tab for the root folder ( **SAS Folders**), select **SASUSERS**.

Note: To access this tab, select the **Folders** tab, right-click the root folder, and select **Properties**.

- b Grant the Read permission. This enables registered users to perform tasks such as querying cubes, accessing data through information maps, and viewing the contents of tables.

If you want to manage access to data more precisely, set grants of the Read permission on specific folders for specific users. Users need the Read permission as follows:

- Users need the Read permission on an information map in order to access data through that information map. For example, if Joe is denied Read permission on an information map, he cannot view reports that are based on that information map.
- Users always need the Read permission on OLAP data in order to access that data.
- Users sometimes need the Read permission on relational data in order to access that data. Read permission is required when data is accessed using the metadata LIBNAME engine.
- Users need the Select permission on metadata-bound data in order to access that data. For more information, see [“Metadata-Bound Libraries” on page 118](#).

- 5 If users who are not in the SAS Administrators group registers cubes, grant those users the WriteMetadata permission on the OLAP schema.
 - a On the **Folders** tab, expand the **Shared Data** folder and select the **SASApp - OLAP Schema** folder.
 - b In the right panel, right-click the schema  and select **Properties**.
 - c On the **Authorization** tab, select or add an identity and grant WriteMetadata permission to that identity. For example, to allow all registered users to add cubes, assign the grant of WriteMetadata permission to SASUSERS.
- 6 Verify that physical-layer access is available. Here are the general requirements:
 - A user who accesses SAS data sets from a standard workspace server needs host layer Read access to those files.
 - A user who performs tasks that involve writing to a host directory needs host layer Write access to that directory.
 - Server launch credentials need host access to any SAS data that the server retrieves. Initially, the SAS Spawned Servers account (sassrv) is the launch credential for the stored process server and the pooled workspace server.
- 7 In the initial configuration, the **Server Manager** capability is available to only the SAS Administrators group. This prevents other users from accessing server definitions under that plug-in. For increased security, use permissions to protect

server definitions. For more information, see [“Protect Server Definitions” on page 101](#).

- 8 In a new deployment, access to most resources and data is undifferentiated. All registered nonadministrators have identical metadata-layer access to content, data, and application features. Everyone who uses a stored process server or pooled workspace server has identical host-layer access to any SAS data that server retrieves. In a migrated deployment, access to most resources and data mirrors access in the original environment. To manage access to objects such as reports, stored processes, information maps, and data definitions, create custom folders that reflect the distinctions that you want to make. For more information, see [Access to Metadata Folders on page 89](#).
- 9 To fully protect SAS data sets, you must address host access. For more information, see [Chapter 8, “Access to SAS Data,” on page 111](#).

See Also

[“Permissions by Object Type” on page 61](#)

PART 2

Authorization

Chapter 5	
<i>Metadata Authorization Model</i>	55
Chapter 6	
<i>Access to Metadata Folders</i>	89
Chapter 7	
<i>Access to Server Definitions</i>	101
Chapter 8	
<i>Access to SAS Data</i>	111
Chapter 9	
<i>Batch Tools for Metadata Authorization</i>	131

Metadata Authorization Model

Authorization Overview	56
Three Levels of Granularity	57
Two Relationship Networks	57
Object Inheritance	59
See Also	61
Permissions by Object Type	61
Introduction	61
Permission Tips for Selected Content and Data Objects	63
Permission Tips for Selected System and Administrative Objects	65
See Also	66
Permissions by Task	67
Introduction	67
Working with Folders	67
Working with Reports	68
Working with Information Maps	68
Working with Stored Processes	69
Working with Publishing Channels	69
Working with Tables	70
Working with SAS OLAP Cubes	70
Working with SAS OLAP Shared Dimensions	71
See Also	71
Permission Origins	72
Introduction to Permission Origins	72
Simple Permission Origins	72
Inherited Permission Origins	73
Types of Access Controls	74
Authorization Decisions	75
Access Control Evaluation Process	75
Precedence Principles and Examples	76
See Also	77
Fine-Grained Controls for Data	77
What Are Fine-Grained Controls?	77
What Implementations are Available?	78
About Identity-Driven Properties	79
Batch Reporting Considerations	82

Permission Precedence Considerations	82
<i>Use and Enforcement of Each Permission</i>	84
General-Purpose Permissions	84
Specialized Permissions	84
<i>Tips for Efficiently Using Permissions</i>	85
Assign Permissions to Groups	85
Use Folders to Organize Content	86
Centralize Permissions with ACTs	86
Deny Broadly, Grant Selectively (To the Extent Possible)	86

Authorization Overview

The platform provides a metadata-based authorization layer that supplements protections from the host environment and other systems. You can use the metadata authorization layer to manage access to almost any metadata object (for example, reports, data definitions, information maps, jobs, stored processes, and server definitions).

Across authorization layers, protections are cumulative. In order to perform a task, a user must have sufficient access in all applicable layers.

In the metadata layer, the following permissions are always enforced:

- the ReadMetadata permission (RM), which controls the ability to see an object
- the WriteMetadata permission (WM), which controls the ability to update or delete an object

Other permissions are specialized and affect only certain types of objects.

CAUTION

In the metadata authorization layer, not all permissions are enforced for all items. It is essential to understand which actions are controlled by each permission. For more information, see [“Use and Enforcement of Each Permission” on page 84](#).

CAUTION

Some clients enable power users to create and run SAS programs that access data directly, bypassing metadata-layer controls. It is essential to manage physical layer access in addition to metadata-layer controls. For more information, see [Access to SAS Data on page 111](#).

Three Levels of Granularity

You can set permissions at the following levels of granularity:

- Repository-level controls function as a gateway and as a parent-of-last-resort. Repository-level controls are managed from the permission pattern of the repository ACT (**Default ACT**). All registered users should have ReadMetadata and WriteMetadata permissions in the foundation repository ACT's permission pattern.
- Object-level controls manage access to a specific object such as a report, an information map, a stored process, a table, a column, a cube, or a folder. You can define resource-level controls individually (as explicit settings) or in patterns (by applying access control templates).
- Fine-grained controls affect access to subsets of data within a resource. To establish fine-grained controls, you add constraints called permission conditions to explicit grants of the Read permission.

Two Relationship Networks

Permission settings are conveyed across two distinct relationship networks:

- In object inheritance, permissions that you set on one object can affect many other objects. For example, a report inherits permissions from the folder in which the report is located. This network is a simple folder tree, with exceptions such as the following:
 - The root folder is not the ultimate parent. This folder inherits from the repository (through the permission pattern of the repository ACT).
 - The root folder is not a universal parent. Some system resources (such as application servers, identities, and ACTs) are not in the folder tree. For these items, the repository ACT is the immediate and only parent.
 - Inheritance within a table or cube follows the data structure. For example, neither table columns nor cube dimensions have folders as immediate parents. Instead, a column inherits from its parent table and a dimension inherits from its parent cube.
 - Inheritance does not flow through specialty folders such as favorites folders, virtual folders, or search folders.
- In the identity relationships network, permissions that you assign to one identity can affect many other identities. For example, if you grant a group access to a report, that grant applies to everyone who is a member of the group.

This relationship network is governed by a precedence order that starts with a primary (usually individual) identity, can incorporate multiple levels of nested group memberships, and ends with implicit memberships in SASUSERS and then PUBLIC.

The following figures depict the relative priority and specificity of access controls within each of these networks. From top to bottom, the elements in each figure are ordered as follows:

- from highest precedence (hardest to override) to lowest precedence (easiest to override)
- from narrowest impact (most specific) to broadest impact (least specific)

Figure 5.1 *Priority and Specificity in Object Inheritance*

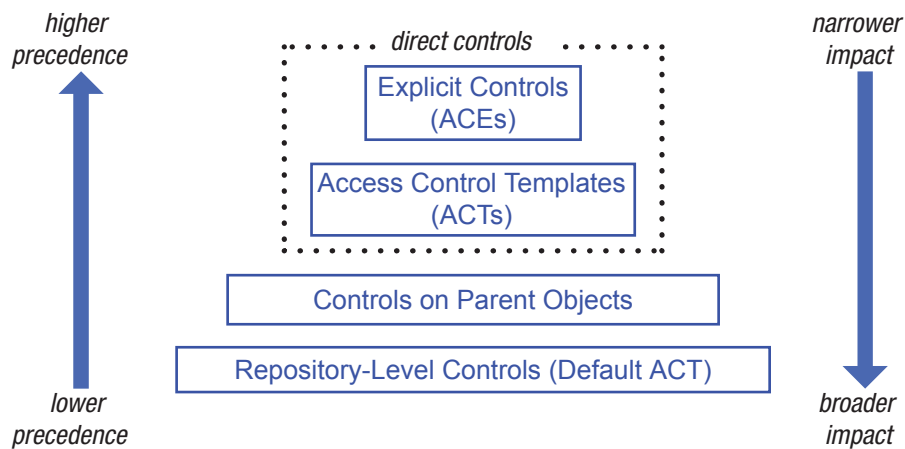
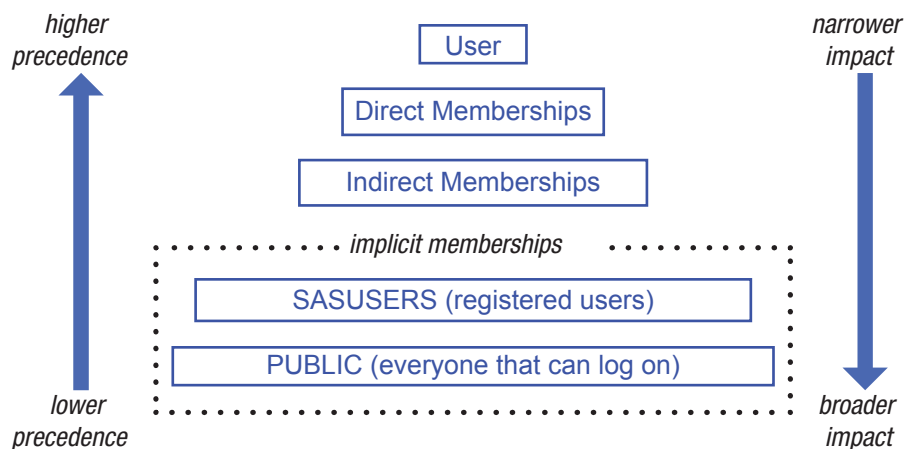


Figure 5.2 *Priority and Specificity in Identity Hierarchy*



Object Inheritance

In the metadata layer, parent objects convey their effective permissions to child objects. Children inherit the net effect of their parents' access controls, not the access controls themselves. The following figures depict inheritance paths in a foundation repository. The arrows in the first figure flow from child to parent (for example, a table inherits effective permissions from its parent folder). The arrows in the second figure flow from parent to child (for example, a folder conveys its effective permissions to the items that it contains).

Figure 5.3 *Inheritance Paths (Separated View)*

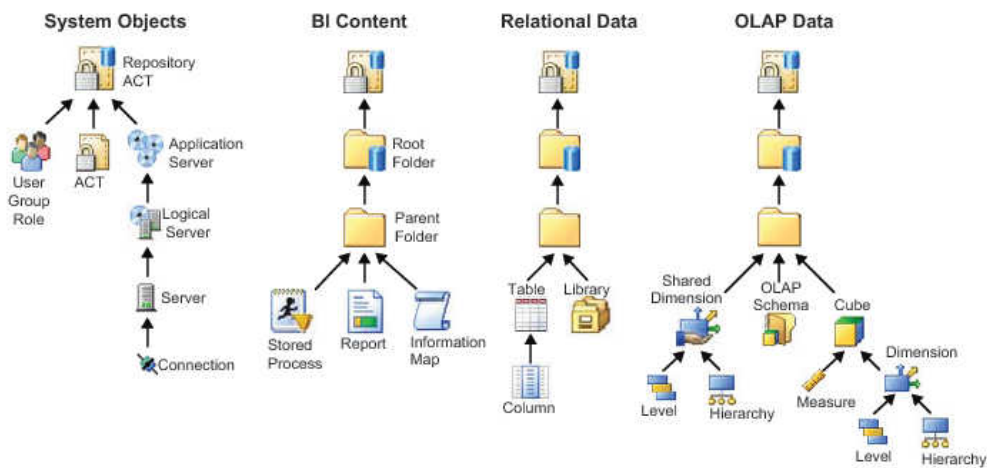
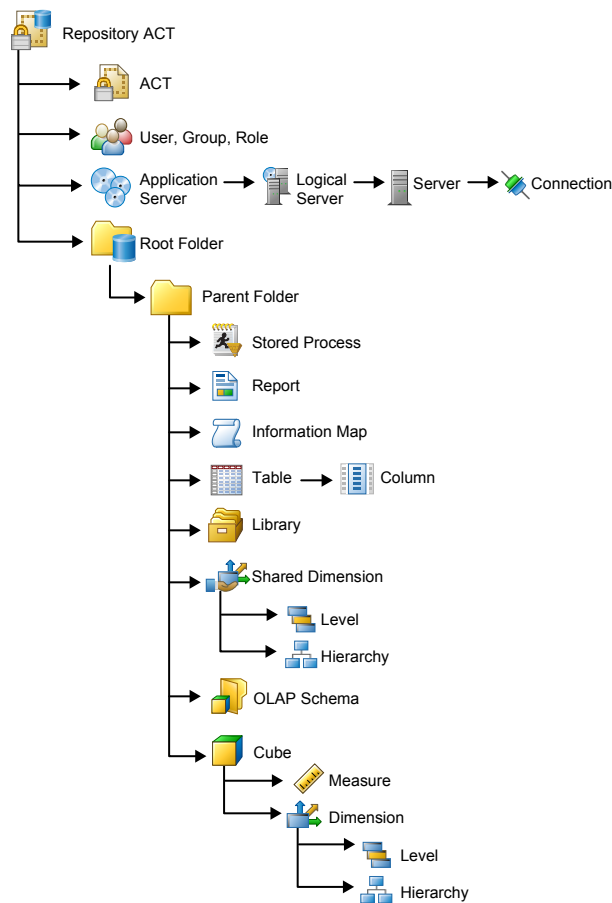


Figure 5.4 Inheritance Paths (Integrated View)

Here are some details about the preceding figures:

- The depicted folder structure is arbitrary and intended only to show the security relationships between different types of objects.
- Not all object types are depicted.

TIP In SAS Management Console, you can trace an object's inheritance by clicking **Advanced** on the object's **Authorization** tab. This feature is available to only unrestricted users.

- The root folder represents the top of the folder tree for the foundation repository (the **SAS Folders** node).
- The root folder inherits settings from the permission pattern of the repository ACT (which is usually named Default ACT).
- Any custom repositories are represented as folders (immediate children of the foundation root folder). Although these folders inherit permissions from both the foundation root folder and the repository ACT of the custom repository, access to objects within the custom repository branch should be managed from the folder side whenever possible.

- In some clients, your **My Folder**  is displayed directly below the root folder. This is just a shortcut for accessing your personal content area. This folder is not an immediate child of the root folder.
- In general, specialized folders (such as search folders, favorites folders, and virtual folders) do not convey permissions to the objects that they contain. An exception is that a favorites folder does convey permissions to any child favorites folders (favorites groups) that it contains.
- The figures show users, groups, and roles inheriting repository-level permissions. In some clients, the authorization information for a user, group, or role reflects special rules that protect identity definitions.
- The inheritance path for metadata-bound data is different. Secured table objects inherit from secured library objects, not from a parent folder. For more information, see [Object Creation, Location, and Inheritance](#).

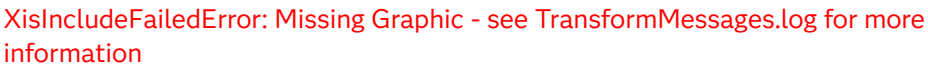
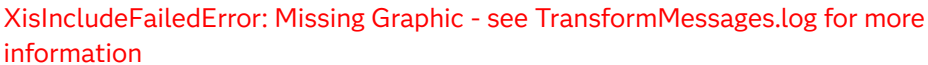
See Also

- [“Identity Hierarchy” on page 20](#)
- [“Authorization Decisions” on page 75](#)

Permissions by Object Type

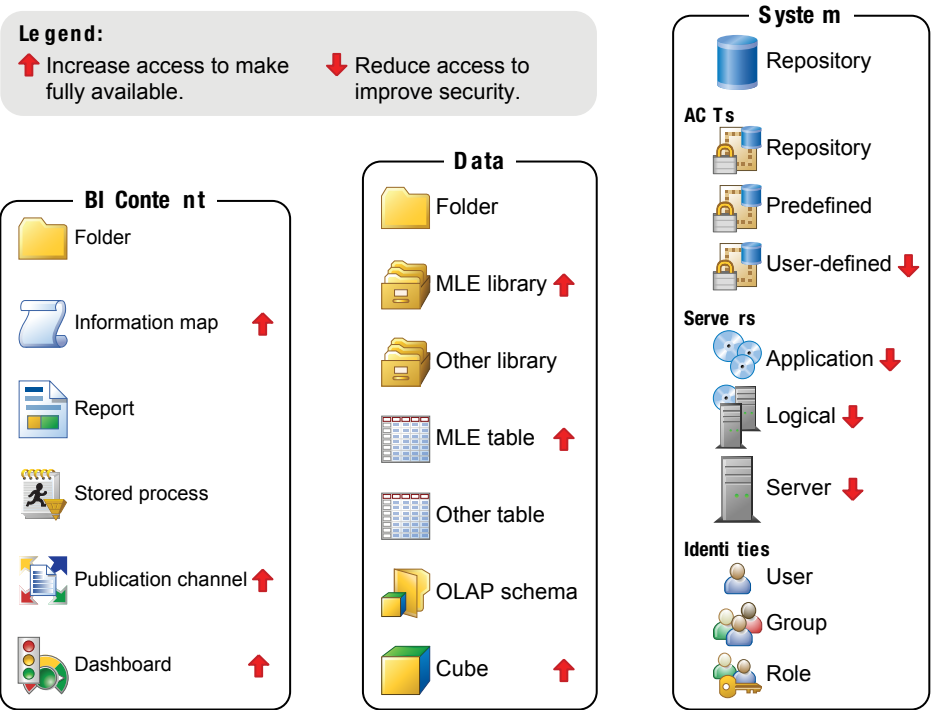
Introduction

The following figure depicts common objects and uses arrows to indicate objects whose permission settings are most likely to need an adjustment. Here are some details about the figure:

- Most of the up arrows

 indicate a need to extend the ability to read data. In the interest of a more secure initial deployment, Read access to data is not granted by default.
- Each down arrow

 indicates a need to limit the ability to modify or delete a system object. Other system items are protected by default.
- MLE refers to the metadata LIBNAME engine. The MLE items are listed to highlight the elevated permission requirements that apply when data is accessed through this engine.

- The intent of the figure is to highlight the adjustments that are most commonly needed. In general, it is preferable to set permissions on a parent (such as a folder) rather than on each individual object (such as a report).

Figure 5.5 Common Access Adjustments by Object Type



The following tables provide object-specific instructions and tips. The purpose of the tables is to highlight special requirements for common items.

TIP For any change-managed areas or resources, change-managed users should have CheckInMetadata (CM) permission (instead of WM or WMM). For more information, see [“Setting Up Change Management” in SAS Intelligence Platform: Desktop Application Administration Guide](#).

Permission Tips for Selected Content and Data Objects

Table 5.1 *Permissions Tips: Selected Content and Data Objects*

	Root folder	The root folder (the SAS Folders node) is an important point of control for narrowing WM. Unlike other folders, the root folder does not have the WriteMemberMetadata permission in its permission list. This is because the root folder is a software component object, not a true folder. The root folder can contain only other folders.
	Folder	For folders, follow these guidelines: ■ Provide users with a clear path of grants of ReadMetadata permission to all of the content that they access. This is a navigational requirement. To browse past a folder, you need ReadMetadata permission for that folder. ■ To enable users to contribute objects to a folder, grant them WriteMemberMetadata permission on that folder. ■ On a folder, grant WriteMetadata permission only to users who should be able to delete, move, or rename that folder. ■ Do not assume that every permission that is listed for a folder is relevant for every object in that folder. ■ If you deny ReadMetadata permission on a folder, make sure that you do not prevent the SAS Trusted User from having that permission on all cubes and schemas within that folder. One approach is to give the SAS System Services group a grant of ReadMetadata permission on the folder. This preserves necessary access for this privileged service identity. For more information, see Access to Metadata Folders on page 89.
	Information map	To access any data through an information map, you need Read permission on that information map. You can manage Read permission globally or selectively. For example: ■ A broad approach is to grant Read permission to SASUSERS in the repository ACT's permission pattern, or on the top folder. ■ A narrow approach is to grant Read permission to smaller groups on specific subfolders or even specific information maps.
	Report	If a user cannot run a report, check these things: ■ Does the user have Read permission for the underlying information map? ■ Does the user have Read permission for any underlying OLAP cube? ■ Does the user have Read permission for any underlying relational data that is accessed via the MLE? ■ Does the SAS Trusted User have ReadMetadata permission for any underlying cube?

- Does the account that retrieves any underlying SAS data have physical access to that data?

It is especially important to manage ReadMetadata permission to pregenerated reports, because those reports can contain embedded data. Any user who views a pregenerated report sees the same data, regardless of his or her permissions to the underlying tables or cubes.



Stored process

To run a stored process, you need ReadMetadata permission on the stored process. If you can see a stored process but cannot run it, you might lack the necessary grant of Read permission for the underlying data. To register a stored process, you need WriteMetadata permission for the target application server.

The method that you use to make a stored process available can affect data retrieval and security. For example, in the standard configuration, a stored process that is assigned to a SAS Workspace Server and embedded in an information map retrieves SAS data under the SAS Pooled Workspace Server's host identity. However, if a user opens that same stored process directly (for example, as a report in SAS Web Report Studio), the host identity of the requesting user (or group) retrieves the data.



Channel

In a new deployment, only the SAS Administrators group can add channels, subscribers, and content. To enable all registered users to publish content to a particular channel, navigate on the **Folders** tab to **System** ⇒ **Publishing** ⇒ **Channels** and grant Write and WriteMetadata permissions to SASUSERS on that channel (WM is required only if a channel has an archive persistent store).

To enable all registered users to add channels or subscribers, grant WriteMemberMetadata permission on the relevant parent folder (for example, on the **System** ⇒ **Publishing** ⇒ **Subscribers** ⇒ **Content Subscribers** folder).



Dashboard

For more information, see ["Implementing Security for SAS BI Dashboard" in SAS Intelligence Platform: Web Application Administration Guide](#).



Schema

To associate an OLAP schema with an application server, you need WriteMetadata permission for the schema and the server. To add cubes to a schema, you need WriteMetadata permission for the schema and WriteMemberMetadata permission for the target folder.

The SAS Trusted User must have ReadMetadata permission for all OLAP schemas and cubes. This access is usually granted through the SAS System Services group.



Cube

To register cubes, you need WriteMetadata permission for the OLAP schema and WriteMemberMetadata permission for the target folder. To associate a cube with a schema, you need WriteMetadata permission for the cube and the schema.

Read permission is enforced for cubes. Grant Read permission broadly (as described for information maps), or add narrower grants of Read permission on folders or individual cubes.

The SAS Trusted User must have ReadMetadata permission for all OLAP schemas and cubes. This access is granted through the SAS System Services group.



For cube components, Read permission is enforced. If you lack access to a measure that participates in a calculated measure, you can get unintended results.

Cube component There are also navigational requirements for Read permission on cube components. If a user does not have Read permission to a hierarchy, the user cannot navigate to the top levels within the hierarchy. If a user does not have Read permission to a particular level in a hierarchy, the user cannot navigate to the next level.



Shared dimension

To set permissions on a shared dimension, navigate to it directly under its parent folder. You cannot set permissions on a shared dimension that you accessed by navigating within a cube.

You cannot delete a shared dimension that is in use (that is included in one or more cubes).



Library

To associate a library with an application server, you need WriteMetadata permission for the server (but not for the library). For a library that is accessed via the metadata LIBNAME engine (MLE), you need the Create permission in order to add tables and the Delete permission in order to delete tables.



Table

To associate a table with a library, you need WriteMetadata permission for the table and the library. For a table that is accessed via the MLE, you need Read permission in order to access data. The Create, Delete, and Write permissions affect your ability to add, update, or delete data. You can grant these permissions broadly (as described in the information map row) or precisely (for example, to a small group of users on a particular table).



Column

The MLE does not support column-level access distinctions for the Read permission. Column-level access distinctions for the ReadMetadata permission are supported.

Permission Tips for Selected System and Administrative Objects

Table 5.2 *Permissions Tips: Selected System and Administrative Objects*



Repository

On a foundation repository, all participating users should have repository-level ReadMetadata and WriteMetadata permissions. Other predefined entries in the permission pattern of the repository ACT (**Default ACT**) provide necessary administrative and service access.



Application server

To monitor or operate servers other than the SAS Metadata Server, you need the Administer permission on the server. (The Metadata Server requires the **Metadata Server: Operation** role instead of the Administer permission.)

To associate a stored process, OLAP schema, or library with an application server, you need WriteMetadata permission for that application server. Certain service identities need ReadMetadata permission to all server definitions. For more information, see [Access to Server Definitions on page 101](#).



Logical server

To use a logical server, you need ReadMetadata permission for at least one of that server's connections. This is called server access security. Certain service identities need ReadMetadata permission to logical server definitions. For more information, see [“Hide Server Definitions” on page 105](#).



Identity

User administration capabilities (from the **Metadata Server: User Administration** role) enable you to create, update, and delete users, groups, and roles. You can delegate management of an identity to someone who does not have user administration capabilities by adding explicit or ACT grants of WriteMetadata permission in the identity's authorization properties. An identity's authorization properties have no effect on what that identity can do. You need ManageMemberMetadata permission to change the membership of the UserGroup and Role. ManageCredentialsMetadata enables you to manage accounts and trusted logins of User and UserGroup.



ACT

To create an ACT, you need repository-level WriteMetadata permission. Each predefined ACT is protected by direct access controls. ACTs that you create are not automatically protected. It is essential to add protections (direct controls in the ACT's authorization properties) to any ACTs that you create.



Secured data folder

A secured data folder is a specialized container for secured library objects. In each metadata repository, the first secured data folder is Secured Libraries, within the System folder. For more information, see [“Metadata-Bound Libraries” on page 118](#).



Secured library object

A secured library object is the target to which a physical SAS library is bound. The Select permission on each secured library object manages Read access to its corresponding physical library. Secured library objects are stored beneath a repository's Secured Libraries folder. For more information, see [“Metadata-Bound Libraries” on page 118](#).



Secured table object

A secured table object is the target to which a physical SAS data set (a table or view) is bound. The Select permission on each secured table object manages Read access to its corresponding physical data set. Each secured table object is stored beneath a secured library object. For more information, see [“Metadata-Bound Libraries” on page 118](#).

See Also

- [“Permissions by Task” on page 67](#)
- [“Use and Enforcement of Each Permission” on page 84](#)

Permissions by Task

Introduction

The following tables show required metadata layer permissions for selected tasks. For each task, a user must have the specified access to the specified metadata objects.

TIP For any change-managed areas or resources, change-managed users should have CheckInMetadata (CM) permission (instead of WM or WMM). For more information, see [“Setting Up Change Management” in SAS Intelligence Platform: Desktop Application Administration Guide](#).

Working with Folders

Table 5.3 Working with Folders

Task	Repository	Parent Folder	Folder	Item
Add a folder	RM, WM	RM, WMM ¹	-	-
Delete a folder	RM	RM, WMM ¹	RM, WM	-
Rename a folder	RM	RM	RM, WM	-
Set folder permissions	RM	RM	RM, WM	-
Add an item to a folder	RM, WM	RM	RM, WMM	-
Delete an item from a folder	RM	RM	RM, WMM	RM, WM
Copy/export items	RM	RM	RM	RM
Paste/import items	RM, WM	RM	RM, WMM	-

¹ If the parent folder is the root folder, you need RM, WM on the root folder.

Working with Reports

Table 5.4 Working with Reports

Task	Repository	Parent Folder	Report	Stored Process ¹	Information Map ¹	Data
Create and save a new report	RM, WM	RM, WMM	-	RM	RM, R	RM, R ²
Delete a report	RM	RM, WMM	RM, WM	-	-	-
View or refresh a report	RM	RM	RM	RM	RM, R	RM, R ²
View a batch report	RM	RM	RM	-	-	-
Edit or rename a report	RM	RM	RM, WM	-	-	-
Set report permissions	RM	RM	RM, WM	-	-	-

¹ This is not a required element for a report.

² The Read permission is required for data that is accessed through the metadata LIBNAME engine or the OLAP server.

Working with Information Maps

Table 5.5 Working with Information Maps

Task	Repository	Parent Folder	Information Map	Stored Process ¹	Data
Create and save a new information map	RM, WM	RM, WMM	-	RM	RM, R ²
Delete an information map	RM	RM, WMM	RM, WM	-	-
Set information map permissions	RM	RM	RM, WM	-	-
Edit or rename an information map	RM	RM	RM, WM	-	-

Task	Repository	Parent Folder	Information Map	Stored Process ¹	Data
Run queries in an information map	RM	RM	RM, R	RM	RM, R ²

¹ This is not a required element for an information map.

² The Read permission is required for data that is accessed through the metadata LIBNAME engine or the OLAP server.

Working with Stored Processes

Table 5.6 Working with Stored Processes

Task	Repository	Parent Folder	Application Server	Stored Process	Data
Register a stored process	RM, WM	RM, WMM	RM, WM ¹	-	-
Delete a stored process	RM	RM, WMM	RM, WM ¹	RM, WM	-
Set stored process permissions	RM	RM	RM	RM, WM	-
Run a stored process	RM	RM	RM	RM	RM, R ²

¹ The WM permission is required for stored processes with physical source code files and only on the SAS Application Server.

² The Read permission is required for data that is accessed through the metadata LIBNAME engine or the OLAP server.

Working with Publishing Channels

Table 5.7 Working with Publishing Channels

Task	Repository	Parent Folder	Channel	Subscriber
Add a channel or subscriber	RM, WM	RM, WMM	-	-
Delete a channel or subscriber	RM	RM, WMM	RM, WM	RM, WM
Edit a channel or subscriber	RM	RM	RM, WM	RM, WM
Publish content to a channel	RM, WM ¹	RM	RM, W, WM ¹	RM ²

¹ WM is required if the channel has an archive persistent store.

² Content is published to only those subscribers for whom you have RM.

Working with Tables

Table 5.8 Working with Tables

Task	Repository	Server ¹	Library	Parent Folder	Table	Column
Register a table	RM, WM	RM	RM, WM	RM, WMM	-	-
Delete a table	RM	RM	RM, WM	RM, WMM	RM, WM	-
Set table permissions	RM	-	RM	RM	RM, WM	-
Access table data	RM	RM	RM	RM	RM, R ²	RM
Register a library	RM, WM	RM, WM	-	RM, WMM	-	-

¹ SAS Application Server

² The Read permission is required for data that is accessed through the metadata LIBNAME engine.

Note: The preceding information applies to working with traditional library and table objects. If you are using metadata-bound data (secured library objects and secured table objects), see [Permissions for Metadata-Bound Data](#) instead.

Working with SAS OLAP Cubes

Table 5.9 Working with SAS OLAP Cubes

Task	Repository	Server ¹	Schema	Parent Folder	Cube	Source Data
Register a cube	RM, WM	RM	RM, WM	RM, WMM	-	RM, R ²
Delete a cube	RM	RM	RM, WM	RM, WMM	RM, WM	-
Rebuild a cube	RM	RM	RM	RM	RM, WM	RM, R ²
Refresh a cube	RM	RM	RM	RM	RM, R	RM, R ²
Access cube data	RM	RM	RM	RM	RM, R	-

Task	Repository	Server ¹	Schema	Parent Folder	Cube	Source Data
Register a schema	RM, WM	RM, WM	-	RM, WMM	-	-
Set cube permissions	RM	-	RM	RM	RM, WM	-
Use the OLAP Server Monitor	RM	A	-	-	-	-

¹ SAS Application Server

² The Read permission is required for data that is accessed through the metadata LIBNAME engine.

Working with SAS OLAP Shared Dimensions

Table 5.10 Working with SAS OLAP Shared Dimensions

Task	Repository	Server ¹	Schema	Parent Folder	Cube	Shared Dimension	Source Data
Register a shared dimension	RM, WM	RM	RM, WM	RM, WMM	-	-	RM, R ²
Delete a shared dimension ³	RM	RM	RM, WM	RM, WMM	-	RM, WM	-
Rebuild a shared dimension	RM	RM	RM	RM	-	RM, WM	RM, R ²
Use a shared dimension	RM	RM	RM	RM	RM, R	RM	-

¹ SAS Application Server

² The Read permission is required for data that is accessed through the metadata LIBNAME engine.

³ You cannot delete a shared dimension that has any associations to any cubes.

See Also

- [“Permissions by Object Type” on page 61](#)
- [“Use and Enforcement of Each Permission” on page 84](#)

Permission Origins

Introduction to Permission Origins

The permission origins feature identifies the source of each effective permission. Permission origins answers the question: Why is this identity granted (or denied) this permission?

In the origins answer, only the controlling (winning, highest precedence) access control is shown. If there are multiple tied winning controls, they are all shown. Other, lower precedence controls are not shown in the origins answer.

Simple Permission Origins

The following table provides simple examples of permission origins answers. In each example, we are interested in why UserA has an effective grant on FolderA. In each example, UserA is a direct member of both GroupA and GroupB. Each row in the table is for a different (independent) permissions scenario. In the table, the first column depicts the contents of the Origins window. The second column interprets the information.

Table 5.11 Origins: Simple Examples

Origins Information	Source of UserA's Effective Grant on FolderA
 UserA [Explicit]	On FolderA, an explicit grant for UserA
 GroupA [Explicit]	On FolderA, an explicit grant for GroupA
 GroupA [Explicit]	On FolderA, explicit grants for GroupA and GroupB
 GroupB [Explicit]	Note: Two settings are shown because they are tied and they both win (UserA is a direct member of GroupA and GroupB).
 GroupA [ACT: GroupARead]	On FolderA, an ACT pattern grant for GroupA (from a directly applied ACT)
 SASUSERS [ACT: GenRead]	On FolderA, an ACT pattern grant for SASUSERS (from a directly applied ACT)

Origins Information	Source of UserA's Effective Grant on FolderA
 GroupA [ACT: GroupARead]  GroupB [ACT: GroupBRead]	On FolderA, ACT pattern grants for GroupA and GroupB (from two different directly applied ACTs). Note: Two settings are shown because they are tied and they both win (UserA is a direct member of GroupA and GroupB).
 GroupA [ACT: GroupABRead]  GroupB [ACT: GroupABRead]	On FolderA, ACT pattern grants for GroupA and GroupB (from the same directly applied ACT). Note: Two settings are shown because they are tied and they both win (UserA is a direct member of GroupA and GroupB).
 UserA is unrestricted.	UserA's status as an unrestricted user (someone who is unrestricted is always granted all permissions)
 This setting mirrors the WriteMetadata setting.	The WriteMetadata setting on FolderA. To investigate further, examine the origins for that setting.

Inherited Permission Origins

In many cases, the controlling setting is not on the current object. Instead, the controlling setting is defined on a parent object and inherited by the current object.

The following table provides examples in which the controlling setting comes from a parent object. Because the source of the effective permission is a parent object, the answer must identify which parent object has the controlling setting. For this reason, the origins answers in the following examples identify both a particular parent object (the object that has the controlling setting) and the controlling setting itself.

In each example, we are interested in why UserA has an effective grant on FolderA. In each example, UserA is a direct member of both GroupA and GroupB. Each row in the table is for a different (independent) permissions scenario. In the table, the first column depicts the contents of the Origins window. The second column interprets the information.

Table 5.12 *Origins: Inheritance Examples*

Origins Information	Source of UserA's Effective Grant on FolderA
 ParentFolderA  UserA [Explicit]	On ParentFolderA, an explicit grant for UserA

Origins Information	Source of UserA's Effective Grant on FolderA
 ParentFolderA  GroupA [Explicit]	On ParentFolderA, an explicit grant for GroupA
 ParentFolderA  GroupA [Explicit]  GroupB [Explicit]	On ParentFolderA, explicit grants for GroupA and GroupB
 ParentFolderA  GroupA [ACT: GroupARead]	On ParentFolderA, an ACT pattern grant for GroupA (from a directly applied ACT)
 GreatGrandParentFolderA  SASUSERS [ACT: GenRead]	On GreatGrandParentFolderA, an ACT pattern grant for SASUSERS (from a directly applied ACT)
 ParentFolderA  GroupA [ACT: GroupARead]  GroupB [ACT: GroupBRead]	On ParentFolderA, ACT pattern grants for GroupA and GroupB (from two different directly applied ACTs)
 GrandParentFolderA  GroupA [ACT: GroupABRead]  GroupB [ACT: GroupABRead]	On GrandParentFolderA, ACT pattern grants for GroupA and GroupB (from the same directly applied ACT).
 SAS Folders  SASUSERS [Explicit]  Default ACT [CustomRepositoryA]  SASUSERS [Pattern]	On the SAS Folders node, an explicit grant for SASUSERS. Also, in CustomRepositoryA's default ACT, a pattern grant for UserA. Note: In this example, FolderA is within a custom repository, so it inherits from both the SAS Folders node and the custom repository's default ACT pattern. Two settings are shown because they are tied and they both win.

Types of Access Controls

explicit controls

A user has an explicit control on an object if the user is directly and individually granted or denied a permission on the object. Explicit settings have the highest precedence. However, managing a large number of explicit controls for

individual users can be cumbersome. For greater efficiency, we recommend that you set explicit controls for groups, use ACTs, and rely on inheritance.

ACT settings

A user has an ACT setting on an object if an ACT that is applied to the object has a permission pattern that explicitly grants or denies the relevant permission to the user. Each ACT adds its pattern of grants and denials to the settings for each object to which the ACT is applied.

indirect settings

One way that a user can have an indirect setting on an object is if the user belongs to a group that has an explicit or ACT setting on the object. Another way that a user can have an indirect setting on an object is through access control inheritance. Inherited settings come from a parent object (such as a folder). Inherited settings matter only if there are no relevant direct controls on the target object. The term “indirect settings” is also used to refer to a WriteMemberMetadata setting that mirrors the WriteMetadata setting, and to grants that come from a special status (such as being unrestricted).

permission conditions

Permission conditions constrain explicit grants of the Read permission, so that only a subset of the target data is available. For example, you can set permission conditions on OLAP dimensions (limiting access to members) and information maps (limiting access to rows).

Authorization Decisions

Access Control Evaluation Process

The relative precedence of each access control is based on where it is set, who it is assigned to, and how it is set. The following list summarizes how the SAS Metadata Server evaluates all relevant access controls to reach an authorization decision:

- 1 Direct controls (permissions that are set directly on the target object) are examined.
 - Any conflicts that arise from group membership are resolved by the identity hierarchy. For example, an explicit control that is assigned to a user overrides a conflicting explicit control that is assigned to a group to which the user belongs.
 - If there is a conflict between an explicit control and an ACT setting at the same level in the identity hierarchy, the explicit control takes precedence.
 - If there is a conflict between two explicit controls (or two ACT settings) at the same level in the identity precedence hierarchy, the outcome is a denial.
 - If one or more permission conditions have been defined, the condition that is assigned at the highest level of identity precedence is applied. Other

conditions that also apply to a user because of group memberships do not provide additional, cumulative access (unless there are multiple tied groups at the highest level of identity precedence).

- If there are no relevant direct controls, the evaluation process continues.
- 2 The step 1 evaluation process is applied to the immediate parent of the target object. For example, the immediate parent of a report is its folder. If no direct controls are found, each successive parent object is examined in turn.

Note: In the unusual circumstance in which an object has more than one immediate parent, a grant from any inheritance path is sufficient to provide access.

- 3 If no direct controls are found on the object or on any of its parent objects, the permission pattern of the repository ACT (Default ACT) is examined. The repository ACT serves as the inheritance parent of last resort.
 - If the repository ACT grants or denies the requested permission, that grant or denial is determinative.
 - If the repository ACT neither grants nor denies the permission, the permission is denied.
 - If there is no repository ACT, the permission is granted. You should always have a designated repository ACT.

Precedence Principles and Examples

The following table summarizes the principles of the authorization decision process for a user (Joe) who is looking for an object (LibraryA or ObjectA):

Table 5.13 *Precedence Principles for Authorization*

Principle	Example ¹	Outcome
Settings on an object have priority over settings on the object's parents.	LibraryA has an explicit denial for PUBLIC. LibraryA's parent folder has an explicit grant for Joe.	Joe cannot see LibraryA.
Conflicting settings on an object are resolved by identity precedence.	LibraryA has an explicit denial for GroupA and an explicit grant for GroupAA. Joe is a direct member of GroupA. GroupA is a direct member of GroupAA.	Joe cannot see LibraryA.
In an identity precedence tie, explicit settings have priority over ACT settings.	LibraryA has an ACT denial for GroupA and an explicit grant for GroupB. Joe is a direct member of both GroupA and GroupB.	Joe can see LibraryA.

Principle	Example ¹	Outcome
In an identity precedence tie that is not resolved by the preceding row, the outcome is a denial.	LibraryA has an explicit denial for GroupA and an explicit grant for GroupB. Joe is a direct member of both GroupA and GroupB.	Joe cannot see LibraryA.
A grant from any inheritance path can provide access.	ObjectA has no explicit or ACT settings, one immediate parent that conveys a grant, and another immediate parent that conveys a denial. ²	Joe can see ObjectA.

¹ The settings described in this column are the only explicit or ACT settings for ReadMetadata permission on LibraryA (or ObjectA).

² Having more than one immediate parent is not a common circumstance.

See Also

- [“Object Inheritance” on page 59](#)
- [“Identity Hierarchy” on page 20](#)
- [Table 5.16 on page 83](#)

Fine-Grained Controls for Data

What Are Fine-Grained Controls?

Business requirements often specify that different users should see different portions, or slices, of data. In some cases, the requirement is driven by the sensitive nature of data. For example, company policy might state that each salesperson should be able to access only his or her own salary information. In other cases, the requirement is intended to prevent information overload. For example, each regional sales team within a national organization might be interested in only the sales trend information for their region. Fine-grained access distinctions are frequently based on each user’s place in an organizational structure such as a management hierarchy or a product matrix. The visibility of data can depend on a simple, site-specific condition such as a user’s security clearance level, or on a more complex condition that consists of multiple filters.

You use fine-grained controls to specify who can access particular rows within a table or particular members within a cube dimension. These controls often subset data by a user characteristic such as employee ID or organizational unit. For example, a table that contains patient medical information might be protected by

row-level permissions that enable each doctor to see only those rows that contain data about that doctor's patients.

Unlike other access controls, fine-grained controls are based on filters and rely on target data that is modeled to work with those filters. When fine-grained controls are used, there are three possible authorization decision outcomes for a user request to view data:

Grant

The requesting user can access all data.

Deny

The requesting user cannot access any data.

Conditional Grant

The requesting user can access only the data that meets specified filtering conditions.

What Implementations are Available?

Each of the following components offers an implementation of fine-grained controls:

BI row-level permissions

provide filtering for SAS data sets and third-party relational data accessed through an information map. You define and assign the filters in SAS Information Map Studio or with the INFOMAPS procedure. For more information, see [SAS Guide to BI Row-Level Permissions](#).

CAUTION

Not all SAS clients require that users go through information maps in order to access data. Comprehensive security that incorporates BI row-level permissions requires a specialized configuration that is supported only by SAS Web Report Studio.

Note: BI row-level permissions are primarily intended for use with SAS Web Report Studio. For information about the row-level security feature that is available in SAS Visual Analytics, see [SAS Visual Analytics: Administration Guide](#).

SAS OLAP Server

provides filtering for SAS OLAP data using MDX expressions. You define and assign the filters in SAS Management Console, SAS OLAP Cube Studio, or SAS Data Integration Studio. See [SAS OLAP Server: User's Guide](#).

SAS Scalable Performance Data Server

enables you to define database views that filter rows based on the user ID of the connecting client. This functionality is provided by the @SPDSUSR system variable. The metadata-based identity-driven properties are not available in this implementation.

metadata-bound libraries

provide two methods of filtering for metadata-bound tables. The first method is to provide users with views that include a portion of a table's data without providing access to the underlying table. With this method, you can use the `_METADATA_AUTHENTICATED_USERID_` substitution parameter to filter rows based on the authenticated user ID of the requesting client. The second method, which is available with [SAS 9.4 M1](#), is to specify a filter in the secured table object's authorization properties in SAS Management Console. With this method, you can use the metadata-based identity-driven properties to filter rows. For more information, see [SAS Guide to Metadata-Bound Libraries](#).

The following table compares the implementations:

Table 5.14 Comparison of Fine-Grained Control Implementations

Implementation	Secure	Graphical Filter Creation	Authorization UI	Metadata -Aware	Identity-Driven
OLAP member-level	✓	✓	✓	✓	✓
SPD Server row-level	✓				✓
BI row-level	⚠	✓	✓	✓	✓
Metadata-bound libraries	✓		✓	✓	✓

About Identity-Driven Properties

Overview of Identity-Driven Properties

It is often necessary to make per-person access distinctions for the rows in a table or the members in a dimension. You can make a separate filter for each user (such as `where name="joe"`). However, if you have more than a few users, this approach quickly becomes cumbersome. The more efficient alternative is to create a dynamic filter (such as `where name="&name;"`) that can discover and insert the correct, user-specific value into the WHERE expression each time access is requested.

To create a dynamic filter, use an identity-driven property as the value against which values in the target data are compared. This list explains how the substitution works:

- 1 Each identity-driven property corresponds to a characteristic (such as name, user ID, or external identity).
- 2 Each user's values for these characteristics (such as `joe`, `WinXP\joe`, or `607189`) are stored in the metadata.

- 3 The identity-driven property is aware of the user ID with which a client authenticated and can locate information that is stored in the metadata for that user ID.
- 4 Each time it receives a request, the identity-driven property substitutes a user-specific value into the filter expression.

Note: This discussion is not applicable to the SPD Server, which has its own implementation of identity-based filtering.

These are the most useful identity-driven properties:

SAS.Userid

returns an authenticated user ID, normalized to the uppercase format USERID or USERID@DOMAIN.

SAS.ExternalIdentity

returns a site-specific value (for example, employee ID). This property is often useful because its values are likely to match user information in your data. An identity can have more than one external identity value. However, only the first value is returned. Unlike the values for other identity-driven properties, values for this property are not always populated in the metadata. See [“External Identities” on page 22](#).

SAS.IdentityGroups

returns a list of the groups and roles that this identity belongs to (directly, indirectly, or implicitly). The list contains the group and role names, as displayed in the **Name** field on the **General** tab for each group or role.

SAS.PersonName

returns a user name, as displayed in the **Name** field in the user’s general properties.

These identity-driven properties are also supported:

SAS.IdentityGroupName

returns a group name, as displayed in the **Name** field in the group’s general properties. If a user logs on with an ID that is stored in a login on a group definition, the name of the group that owns that login is returned. If a user logs on with a user ID that is not stored in the metadata, the PUBLIC group is returned.

This property is useful only in the unusual circumstance where a user logs on with the user ID that is defined for a group login. In almost all cases, a user logs on with a user ID that is defined for an individual user definition. Not all applications allow a group to log on. This property is not supported if client-side pooling is used.

SAS.IdentityName

returns a user name or group name, as displayed in the **Name** field in the general properties for the user or group. This property is a generalization of SAS.PersonName and SAS.IdentityGroupName.

Note: In certain circumstances, a connecting identity might not have a value for the identity-driven property that you are using. This can happen with the ExternalIdentity property (sometimes), the IdentityGroupName property (almost

always), or the `PersonName` property (rarely). When a connecting user does not have a value for the property that a query uses, an empty string is returned or the query fails.

Syntax for Identity-Driven Properties

The syntax for using identity-driven properties varies depending on which implementation you are using:

- For information maps, use the following syntax: `&property-name` (for example, `&SAS.Userid`).
- For secured table objects in metadata-bound libraries, use the following syntax: `SUB::property-name` (for example, `SUB::SAS.Userid`).

Examples of Identity-Driven Properties

For example, to enable each user to see only his or her own salary information, you could give the PUBLIC group a filter that is based on the `SAS.PersonName` property. At run time, the `SAS.PersonName` value that is associated with the connected user ID is substituted into the filter. In this way, the query is modified as appropriate for each requesting client.

This table contains examples of filters that are based on identity properties, showing representations of both the generic form and how each filter would be modified when executed by a user named Harry Highpoint. Harry is a member of the ETL and Executives groups. The example assumes that the customer has an employee information table named `EmplInfo`, which includes `Name`, `Category`, `WinID`, `Department`, and `EmplID` columns.

Table 5.15 Examples of Filters That Use Identity-Driven Properties

As Defined (Generic Form)	As Executed (Resolved Form)
<code>EmplInfo.WinID=&SAS.Userid¹</code> <code>EmplInfo.WinID="SUB::SAS.Userid"²</code>	<code>Where EmplInfo.WinID="metadata_loginid"</code>
<code>EmplInfo.EmplID=&SAS.ExternalIdentity</code> <code>EmplInfo.EmplID="SUB::SAS.ExternalIdentity"</code>	<code>Where EmplInfo.EmplID="metadata_externalid"</code>
<code>EmplInfo.Department IN &SAS.IdentityGroups</code> <code>EmplInfo.Department IN</code> <code>("SUB::SAS.IdentityGroups")</code>	<code>Where EmplInfo.Department IN ("metadata_group1"</code> <code>"metadata_group2" "metadata_groupn")</code>
<code>EmplInfo.Name=&SAS.IdentityName</code>	<code>Where EmplInfo.Name="metadata_identityname"</code>

As Defined (Generic Form)	As Executed (Resolved Form)
EmplInfo.Name="SUB::SAS.IdentityName"	
EmplInfo.Name=&SAS.PersonName EmplInfo.Name="SUB::SAS.PersonName"	Where EmplInfo.Name="metadata_personname"
EmplInfo.Category=&SAS.IdentityGroupName EmplInfo.Category="SUB::SAS.IdentityGroupName"	Where EmplInfo.Category="metadata_groupname" ³

- 1 Format to filter information maps.
- 2 Format to filter secured tables.
- 3 Because the user does not log on with a user ID that is stored as part of a group definition, the user has no value for this property. This either returns an empty string (in BI row-level permissions) or causes the query to fail (in other implementations).

Batch Reporting Considerations

When you use fine-grained controls, it is essential to understand that only dynamically generated reports display data based on the access that is defined for the requesting user. Static reports display data based on the access that is defined for the user ID that was used to generate the report. For example:

- Manually refreshed reports contain cached data (which can be updated by a user action in the report viewer).
- Pre-generated reports reflect the access of the user ID that was used to generate the report. Identity-specific access distinctions are preserved for pre-generated reports only if you define a separate report job for each user ID.

Permission Precedence Considerations

Note: This discussion applies to only OLAP member-level permissions and authorization-based prefilters. It is not applicable to the SPD Server or to general prefilters in information maps.

Fine-grained controls are assigned to users or groups in the authorization properties of the target dimension, information map, or secured table object. These filters are available only to constrain either an explicit grant of the Read permission (on dimensions and information maps) or an explicit grant of the Select permission (on secured table objects). These filters are incorporated into the access control evaluation process as permission conditions.

A permission condition is applied only if it is on the setting that is closest to the requesting user. Other conditions that are relevant because of further-removed

group memberships do not provide additional, cumulative access. If there is an identity precedence tie between multiple groups at the highest level of identity precedence, those tied conditions are combined in a Boolean OR expression. If the identity precedence tie includes an unconditional grant, access is not limited by any conditions. The following table provides examples:

Table 5.16 *Precedence for Permission Conditions*

Principle	Scenario	Outcome and Explanation
If there are multiple permission conditions that apply to a user because of the user's group memberships, the highest-precedence identity controls the outcome.	A filter on InformationMapA limits Read permission for GroupA.	The user can see only the rows that GroupA is permitted to see. GroupA has higher identity precedence than SASUSERS, so the filters that are assigned to GroupA define the user's access.
	Another filter on InformationMapA limits Read permission for the SASUSERS group. The user is a member of both GroupA and SASUSERS.	
If multiple permission conditions apply at the highest level of identity precedence, any data that is allowed by any of the tied conditions is returned.	A filter on DimensionA limits Read permission for GroupA.	The user can see any member that is permitted for either GroupA or GroupB.
	Another filter on DimensionA limits Read permission for GroupB. The user is a first-level member of both GroupA and GroupB.	

The following example describes the impact of identity precedence when a manager uses an information map that includes both of the following filters for a SALARY table:

- A permission condition that is assigned to the SASUSERS group gives each user access to his or her own salary information.
- A permission condition that is assigned to a Managers group enables each manager to see the salaries of the employees that he or she manages.

When the manager accesses the SALARY table, the filter that is assigned to the Managers group is applied and the filter that is assigned to SASUSERS is ignored. This is because the manager's direct membership in the Managers group has higher identity precedence than the manager's implicit membership in the SASUSERS group. To avoid a situation in which managers can see their employees' salaries but each manager cannot see his or her own salary, you can use either of these approaches:

- Assign the filters to two groups that have the same identity precedence. For example, if you assign the first filter to a general-purpose user-defined group (rather than to SASUSERS), and if you make each manager a direct member of that group, managers have an identity precedence tie between that group and the Managers group. This situation causes the two filters to be combined for

members of the Managers group, enabling those users to see any row that is permitted by either filter.

- Define the Managers filter in a way that encompasses all of the rows that the managers should be able to see. In other words, combine (OR together) the SASUSERS filter and the Managers filter.

Use and Enforcement of Each Permission

General-Purpose Permissions

The following table introduces the general-purpose permissions:

Table 5.17 *General-Purpose Permissions*

Permission (Abbreviation)	Actions Affected
ReadMetadata (RM)	View an object. For example, to see a report, you need the ReadMetadata permission for that report.
WriteMetadata (WM)	Edit, delete, or set permissions for an object. To delete an object, you also need the WriteMemberMetadata permission for the object's parent folder.
WriteMemberMetadata (WMM)	Add an object to a folder or delete an object from a folder. To enable someone to interact with a folder's contents but with not the folder itself, grant WMM and deny WM. ¹
CheckInMetadata (CM)	Check in and check out objects in a change-managed area. The CheckInMetadata permission is applicable only in SAS Data Integration Studio.

¹ A folder's WMM settings mirror its WM settings unless the folder has a direct setting of WMM. A grant (or denial) of WMM on a folder becomes an inherited grant (or denial) of WM on the objects and subfolders in that folder. WMM is not inherited from one folder to another. WMM is not applicable to specialized folders (such as virtual folders).

Specialized Permissions

The following table introduces some of the specialized permissions:

Table 5.18 Specialized Permissions

Permission (Abbreviation)	Actions Affected
Administer (A)	Operate (monitor, stop, pause, resume, refresh, or quiesce) certain SAS servers and spawners.
Create (C)	Add data through the metadata LIBNAME engine.
Read (R)	Read data through certain objects (for example, cubes, information maps, and tables that are accessed through the metadata LIBNAME engine).
Write (W)	Update data through certain objects (for example, data that is accessed through the metadata LIBNAME engine and publishing channels).
Delete (D)	Delete data through the metadata LIBNAME engine.
ManageMemberMetadata (MMM)	Change the membership of the Group and Role. Cannot change security or other account attributes.
ManageCredentialsMetadata (MCM)	Manage accounts and trusted logins of User and Group. Cannot change security or other account attributes.

For more information, see the following documents:

- [SAS Language Interfaces to Metadata](#) (for information about the metadata LIBNAME engine)
- [SAS Guide to Metadata-Bound Libraries](#) (for information about the Insert, Update, Select, Create Table, Drop Table, and Alter Table permissions, and an additional use of the Delete permission)

Tips for Efficiently Using Permissions

Assign Permissions to Groups

You can simplify access control management by assigning permissions to groups rather than to individual users. These examples assume that there are not other explicit or ACT settings on the object:

- To allow only unrestricted users to access an object, set denials on that object for the PUBLIC group.
- To enable only registered users to access an object, set denials for the PUBLIC group and then grant access back to the SASUSERS group.
- To enable only ETL developers and unrestricted users to access an object, create a group for the ETL developers. Then deny permissions to the PUBLIC group and grant access back to the ETL developers group.

Use Folders to Organize Content

You can simplify access control management by creating a folder structure that reflects the access distinctions that you want to make. Instead of setting permissions on each individual object, set permissions on the folders. The objects in a folder inherit the folder's effective permissions.

TIP To protect the folder structure, do not grant WriteMetadata permission on a folder to someone for whom WriteMemberMetadata permission is sufficient.

Centralize Permissions with ACTs

You can simplify access control management by using ACTs. An ACT is a reusable named pattern of settings that you can apply to multiple objects. Each ACT consists of these elements:

- a list of users and groups
- an indication of whether each permission is granted, denied, or unspecified for each user and group in the list

Deny Broadly, Grant Selectively (To the Extent Possible)

Assign denials to the broadest group (PUBLIC) and then add offsetting grants for users or groups whose access you want to preserve. Deny access at the highest point of control and then grant access back on specific containers or objects. These constraints apply:

- The highest point of control is the repository-level settings that are defined on the repository ACT's **Permission Pattern** tab. The security model requires that

participating users have ReadMetadata and WriteMetadata access at the repository level, so broadly denying access here is not a workable approach. Instead, use the next point of control (for example, the top of the folder tree on the **Folders** tab).

- Within the folder tree, users need a clear path of grants of ReadMetadata in order to navigate to the objects that they use. For this permission, setting denials on folders at a high level is not a workable approach.

Access to Metadata Folders

Baseline ACTs	89
See Also	91
Example: Business Unit Separation	91
Variation 1: Regional Separation, Designated Content Creators	92
Variation 2: Functional Separation	95
Key Points about the Baseline ACT Approach	97
Further Considerations for Permissions on Folders	99
Consolidation of ACTs	99
Separated Administration	99
End Users, Folders, and Permissions	100

Baseline ACTs

One approach to setting permissions on folders is to create a few general-use ACTs, and apply one or more of those ACTs to each folder that you need to secure. To grant access back to a particular group, supplement the ACT settings by adding explicit controls on the target folder. The examples in this chapter use three general-purpose ACTs. Each ACT reduces a particular type of access down to a minimal level, so this chapter refers to these ACTs as baseline ACTs.



Hide

gives SAS Administrators and service identities exclusive Read access to metadata (limits visibility).

Table 6.1 Example: Pattern for the Hide ACT

Group	Permission Pattern	
PUBLIC	Deny	ReadMetadata
SAS Administrators	Grant	ReadMetadata ¹

Group	Permission Pattern	
SAS System Services	Grant	ReadMetadata ²

- ¹ This grant ensures that administrators can manage all metadata (for alternatives, see [“Separated Administration” on page 99](#)).
- ² This grant ensures that the SAS Trusted User (who is a member of SAS System Services) can read certain metadata on behalf of all users.



Protect

gives SAS Administrators exclusive Write access to metadata (limits updates, deletions, and contributions).

Table 6.2 Example: Pattern for the Protect ACT

Group	Permission Pattern	
PUBLIC	Deny	WriteMetadata, WriteMemberMetadata, CheckInMetadata, Write, Administer
SAS Administrators	Grant	WriteMetadata, WriteMemberMetadata, CheckInMetadata, Write, Administer, ReadMetadata ¹

- ¹ These grants ensure that administrators can manage all metadata (for alternatives, see [“Separated Administration” on page 99](#)).



LimitData

gives unrestricted users exclusive access to data (limits the availability of data that is accessed through information maps, the OLAP server, or the metadata LIBNAME engine).

Table 6.3 Example: Pattern for the LimitData ACT

Group	Permission Pattern ¹	
PUBLIC	Deny	Read

- ¹ This pattern is unusual in that it consists of a single setting. In the future, you might use this ACT to give a restricted user access to all data.

CAUTION

Relational data that is accessed through other methods is unaffected by the Read permission. Do not rely exclusively on the metadata authorization layer to protect relational data. Use physical-layer protections (host or DBMS permissions) also.

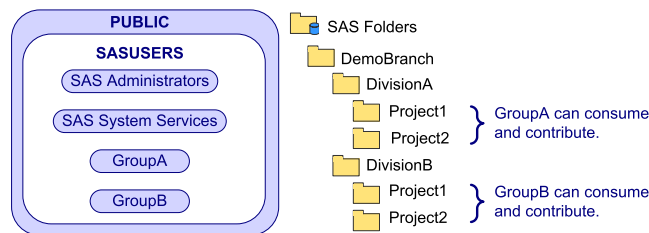
See Also

[“Use and Enforcement of Each Permission” on page 84](#)

Example: Business Unit Separation

This example creates a secured custom branch with mutually exclusive access for two divisions. In this example, the project level folders are for only organizational purposes (there are no access distinctions at the project level). The following figure depicts a partial group and folder structure.

Figure 6.1 Group and Folder Structure



The following table lists the protections for the first four folders in the branch:

Table 6.4 Mutually Exclusive Access

Folder	Direct Controls	
	ACTs	Explicit Grants
DemoBranch	Protect LimitData	
DivisionA	Hide	GroupA: +RM, +R
Project1		GroupA: +WMM
Project2		GroupA: +WMM

Here are some details about this example:

- ReadMetadata permission flows into the DemoBranch from its parent folder. In all of the examples in this chapter, the immediate parent to the DemoBranch is visible to all registered users (SASUSERS). This is a standard setting. Unless

you apply the Hide ACT to the top of your custom branch, the SASUSERS grant of ReadMetadata permission flows into your branch.

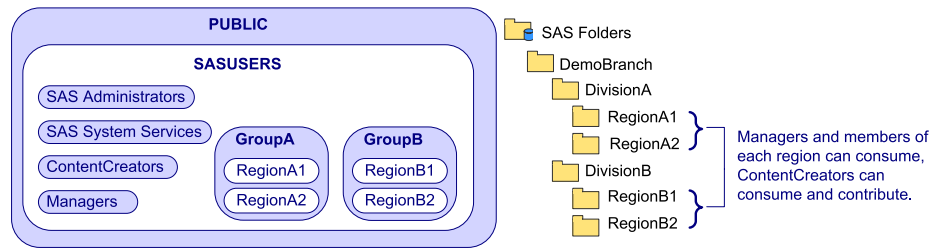
- The ability to create, manage, and delete content is shut off at the top (by the Protect ACT on the DemoBranch). This constraint flows throughout the tree (except where you add supplemental grants of WriteMemberMetadata permission to specific functional groups on specific folders). Users do not add content or access data in the DemoBranch, so no supplemental grants are needed on that folder.
- Because you want to prevent members of GroupB from seeing the DivisionA branch, you apply the Hide ACT on that branch. You then add supplemental grants for GroupA to restore their access.
- On DivisionA's project folders, you apply the Protect ACT to override GroupA's inherited grant of WriteMetadata permission (GroupA's grant of WriteMemberMetadata permission on DivisionA becomes an inherited grant of WriteMetadata permission on the immediate children of DivisionA). This prevents members of GroupA from renaming, deleting, or changing permissions on each project folder. You then add a supplemental grant of WriteMemberMetadata permission so that members of GroupA can contribute content in the project folders.
- On DivisionA's project folders, you do not apply the Hide ACT, because the ReadMetadata access that flows in from the DivisionA folder is appropriate. In this example, the requirement is that all members of GroupA should be able to access content throughout the DivisionA branch.
- Notice that it is not necessary to use separate folders for each type of object.
- Any content contributors who register cubes must have WriteMetadata permission on the OLAP schema . By default, the schema is in the **SAS Folders/Shared Data/SASApp - OLAP Schema** folder.

Variation 1: Regional Separation, Designated Content Creators

This example eliminates the project folders and introduces the following requirements:

- Regional employees see only content for their region.
- A central group of managers see all content.
- A central group of content creators creates all content.

The following figure depicts a partial group and folder structure.

Figure 6.2 Group and Folder Structure

The following table lists the protections for the first four folders in the branch:

Table 6.5 Subgroup Separation and Limited Contribution

Folder	Direct Controls	
	ACTs	Explicit Grants
DemoBranch	Protect LimitData	
DivisionA	Hide	GroupA: +RM ContentCreators: +RM Managers:+RM
RegionA1	Hide	RegionA1: +RM, +R ContentCreators: +RM, +R, +WMM Managers:+RM, +R
RegionA2	Hide	RegionA2: +RM, +R ContentCreators: +RM, +R, +WMM Managers:+RM, +R

Notice that you are repeating many of the same explicit settings on each region, and that this will be the case throughout the DemoBranch. For greater efficiency and more centralized control, create a custom ACT (called RegionLevel) that provides the supplemental grants for your content creators (RM, R, WMM) and your managers (RM, R). Remember to protect the ACT itself.

The following table lists the protections for the first four folders:

Table 6.6 Use a Supplemental ACT

Folder	Direct Controls	
	Baseline ACTs	Supplemental Grants
 DemoBranch	 Protect  LimitData	
 DivisionA	 Hide	 GroupA: +RM  ContentCreators: +RM  Managers:+RM
 RegionA1	 Hide	 RegionA1: +RM, +R  RegionLevel
 RegionA2	 Hide	 RegionA2: +RM, +R  RegionLevel

If you decide to offer content at the division level and you want that content to be available to only managers, you might make these changes:

- Create a DivisionLevel ACT with grants for Managers (RM, R) and ContentCreators (RM, R, WMM). Apply that ACT to each division folder.

.....

Note: This is the same pattern that you use for the RegionLevel ACT, so you could instead simply use that ACT. In this example, you choose to create a separate ACT because you anticipate that the requirements for division-level access and region-level access might diverge in the future.

.....
- Apply the Protect ACT on each region folder (to take away the inherited grant of WriteMetadata permission that content contributors inherit from their division-level grant of WriteMemberMetadata permission).

.....

Note: If you choose to not do this, members of the content creators group can delete, rename, or change permissions for the region folders.

.....

The following table lists the protections for the first four folders in the branch:

Table 6.7 Accommodate Division-Level Content

Folder	Direct Controls	
	Baseline ACTs	Supplemental Grants
 DemoBranch	 Protect  LimitData	
 DivisionA	 Hide	 GroupA: +RM  DivisionLevel
 RegionA1	 Hide  Protect	 RegionA1: +RM, +R  RegionLevel
 RegionA2	 Hide  Protect	 RegionA2: +RM, +R  RegionLevel

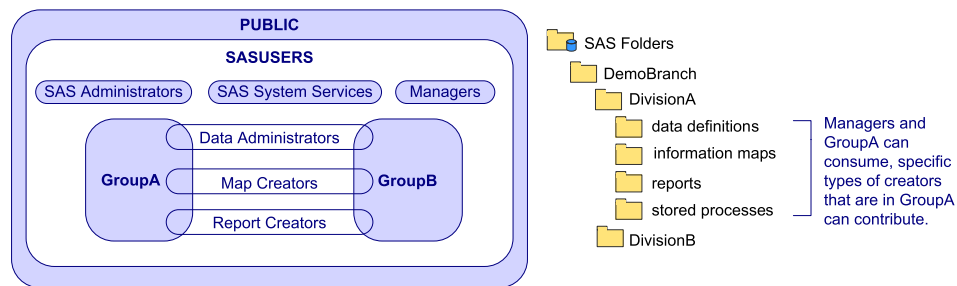
Variation 2: Functional Separation

This example includes three custom groups that represent users who have specific job responsibilities (Data Admins, Map Creators, and Report Creators). Each division folder has separate subfolders for different types of content (data definitions, information maps, report definitions, and stored processes). Supplemental Write access in each folder is limited to members of the appropriate functional groups as follows:

- Data administrators can work in each division's data definitions folder.
- Information map creators can work in each division's information maps folder.
- Report creators can work in each division's reports folder.
- Information map creators and report creators can work in each division's stored processes folder.

Note: Access for these functional groups is also limited by divisional affiliations. For example, only DivisionA's report creators (and SAS Administrators) can add, update, and delete items in DivisionA's reports folder.

The following figure depicts a partial group and folder structure.

Figure 6.3 Group and Folder Structure

The following table lists the protections for the first six folders:

Table 6.8 Functional Separation

Folder	Direct Controls	
	ACTs	Explicit Grants
DemoBranch	Protect LimitData	
DivisionA	Hide	Managers: +RM, +R GroupA: +RM, +R
data definitions	Protect ¹	Data Admins: +WMM
information maps	Protect ¹	Map Creators: +WMM
reports	Protect ¹	Report Creators: +WMM
stored processes	Protect ¹	Map Creators: +WMM Report Creators: +WMM

¹ You do not strictly need the Protect ACT here, because protections flow through from the DemoBranch folder (and are not interrupted by any supplemental grants of WM or WMM on higher level folders). However, you do need the supplemental grants of WMM on these folders, so you might choose to also apply the Protect ACT here for clarity.

Here are some details about this example:

- On DivisionA, users who are not in the Managers group, GroupA, or SAS Administrators are locked out. Such users cannot see or traverse DivisionA or any folders in the DivisionA branch. Managers and members of GroupA have supplemental grants of ReadMetadata and Read permissions on DivisionA; these grants flow through the entire DivisionA branch.

Note: In order to view a DivisionA report, you need ReadMetadata permission and (in most cases) Read permission to the report's underlying components (such as information maps, stored processes, and data).

- On each content type folder, one or more functional groups has a supplemental grant of WriteMemberMetadata permission. However, functional group access is also constrained by division-level ReadMetadata access. For example, even though all report creators have WriteMemberMetadata permission on the DivisionA reports folder, only those report creators who are members of GroupA can see the folder.
- Nothing in these settings prevents creation of a certain type of item. For example, a map creator can create a report and add that report to the map folder or the stored processes folder. Here are some techniques for increasing control:
 - You can use roles to limit the ability to create reports in SAS Web Report Studio.
 - You should limit the ability to register libraries, stored processes, and OLAP schemas by managing WriteMetadata permission on application servers.
 - You can use the map accessibility check configuration option to limit the locations from which SAS Web Report Studio will use relational maps.
- You might choose to put the supplemental grants of WriteMemberMetadata permission in additional ACTs (such as MapCreators ACT, ReportCreators ACT) instead of using explicit settings on each folder. Using ACTs is more centralized, which is beneficial in accommodating later changes to the pattern. Of course, the more divisions you have, the more work it would be to update explicit settings, so the more valuable it would be to centralize settings with ACTs.
- Consider how you would handle an exception requirement. For example, to let a particular data administrator who is not in GroupA contribute to the DivisionA data definitions folder, you might give that user a clear path of grants of ReadMetadata permission to the folder and supplemental grants of WriteMemberMetadata and Read permissions on the folder.

Key Points about the Baseline ACT Approach

- In general, it is not necessary to add protection to the predefined folders (such as the root folder, each user's My Folder, and the System folder). These folders are protected by default.
- The preceding examples do not add grants of ReadMetadata permission on the DemoBranch folder because that folder's parent (the SAS Folders node) is visible to all registered users. The grant of ReadMetadata permission to SASUSERS on the **SAS Folders** node flows into the DemoBranch.

- To hide a branch, apply the Hide ACT to the relevant folder and grant back ReadMetadata permission to any groups who should have access. Remember that a denial of ReadMetadata permission on a folder prevents navigation to content and folders that are below the hidden folder (and that this cannot be mitigated by a grant of ReadMetadata permission on a lower-level folder or other item).
- To enable a group to contribute content to a particular folder, give that group a grant of WriteMemberMetadata permission on that folder and consider applying the Protect ACT to each immediate subfolder.

Note: This protects lower-level folders that inherit the contributing group's WriteMemberMetadata permission grant (from the parent folder) as a grant of WriteMetadata permission (on the child folder). If you do not protect a child folder, the contributing group can rename, delete, or change permissions on that folder. To enable content contributions to the child folder, grant WriteMemberMetadata permission on that folder (and consider repeating the denial of WriteMetadata permission on any subfolders that are below that folder).

Note: Any content contributors who register cubes must have WriteMetadata permission on the OLAP schema. By default, the schema is in the **SAS Folders/ Shared Data/ SASApp - OLAP Schema** folder.

- If you want a group to access data through cubes, information maps, or the metadata LIBNAME engine, give that group a grant of Read permission on the folder that contains the relevant items. For any folder where you add a grant of Read permission, determine whether you want that grant to flow through to the subfolders. Apply the LimitData ACT to any immediate subfolders where you want to prevent data access. The navigational requirement for a clear path of ReadMetadata permission does not apply to the Read permission.
- The examples in this chapter ensure that content contributors have Read permission to the resources that they use. If you have a content contributor who is not also a content consumer, you can choose to not provide Read permission. For example, if you give a user ReadMetadata permission but not Read permission to an information map, that user can still design the map (but cannot perform tasks such as testing queries).
- You might be able to save some time by using this technique:
 - 1 Create one sub-branch (for example, the DivisionA branch in the preceding examples). The folders should have appropriate permission settings but contain no content.
 - 2 Copy that empty sub-branch to create the other branches (for example, to create the DivisionB folders). Change the folder name on the copy and update the settings as necessary. Often, you only have to remove and add a few supplemental grants.

Note: If this technique proves useful, consider keeping an empty template branch that is visible only to administrators. You can use the template if you need to add more branches later.

Further Considerations for Permissions on Folders

Consolidation of ACTs

In general, consolidation (using one pattern in all of the places where it is appropriate) is beneficial, because it simplifies management. However, it might be appropriate to maintain two ACTs that have the similar patterns in circumstances such as these:

- You anticipate that access requirements might diverge. For example, if you think you will eventually separate folder administration from server administration, you might create a SystemProtect ACT for items that are not in the folder tree.
- You want to use a pattern that is similar to but not exactly the same as one of the predefined ACTs. For example, the baseline Hide ACT is not very different from the predefined Private User Folder ACT. We strongly recommend that you do not modify or delete the predefined ACTs, because these ACTs are an integral part of the protections that are set up for you during installation. The usage of each predefined ACT requires certain settings. Modifying the settings on a predefined ACT can compromise the security that that ACT provides.

Note: The examples in this chapter do not demonstrate use of an ACT to protect other ACTs. Consider returning to each ACT's **Authorization** tab, removing the explicit controls, and instead applying an ACT such as the Protect ACT.

Separated Administration

If you need to separate administration privileges by department, the approach in this chapter is not granular enough. If you do not want the SAS Administrators group to have universal access, consider creating parallel sets of baseline ACTs.

For example, to separate administration for an East region and a West region, you might create ACTs such as Hide_East, Hide_West. In each baseline ACT pattern, you would replace the SAS Administrators group with a narrower administrative

group (for example, East_Admins, West_Admins). The denials to PUBLIC and grants to the SAS System Services group would not change. Any unrestricted users can still access everything.

End Users, Folders, and Permissions

Proper use of the WriteMetadata and WriteMemberMetadata permissions protects a folder structure. Keep in mind that end users can affect access to content as follows:

- A user who can update an item can add settings on that item. You cannot prevent this by limiting the availability of SAS Management Console because users can set permissions in other applications (for example, SAS Information Map Studio, SAS OLAP Cube Studio, SAS Data Integration Studio, SAS Enterprise Guide, and the SAS Add-In for Microsoft Office).
- A user who can contribute items to a folder can also add subfolders below that folder. You cannot prevent this by limiting the availability of SAS Management Console because users can add folders in other applications (for example, SAS Information Map Studio, SAS OLAP Cube Studio, SAS Data Integration Studio, SAS Enterprise Guide, and the SAS Add-In for Microsoft Office).
- If you give someone CheckInMetadata permission on a folder, that person can update or delete the folder (through change management activities), as well as check in content to that folder. Change management is an optional feature that is available only for SAS Data Integration Studio.

Access to Server Definitions

<i>Protect Server Definitions</i>	101
Introduction	101
Method	102
Instructions	102
See Also	105
<i>Hide Server Definitions</i>	105
Introduction	105
Method	105
Instructions	107
See Also	109

Protect Server Definitions

Introduction

In a new deployment, all registered users can update and delete server definitions. We recommend that you limit the ability to update or delete server metadata as follows:

- The SAS Administrators group should be able to administer, update, and delete all server definitions.
- Users who assign libraries, stored processes, or an OLAP schema to an application server must have WriteMetadata permission for that application server.
- No other users should be able to update or delete server definitions.

TIP This section explains how to manage access to metadata objects that represent server processes. If you want to limit what a particular server process can do, see [“Locked-Down Servers” on page 120](#) instead.

Method

To avoid repeatedly adding the same explicit grants and denials to multiple pieces of server metadata, create and use an ACT. If you want to enable regular users to assign libraries, stored processes, or an OLAP schema to a particular application server, supplement the ACT settings with an explicit grant of WriteMetadata permission on that application server.

The following table depicts typical protections.

Table 7.1 Example: Protecting Server Definitions

Object	Direct Controls	
	ACTs	Explicit Grants
 SASApp	 Protect	 DataAdmins: +WM
 Each logical server inside SASApp	 Protect	(none)
 SASMeta	 Protect	(none)
Other immediate children of Server Manager	 Protect	(none)

Note: The initial configuration in a new deployment limits access to the logical workspace server and the logical SAS DATA step batch server within SASMeta, so it is not necessary to add protections to those components.

Instructions

Here is one way to set the permissions:

- 1 Log on to SAS Management Console as a member of the SAS Administrators group (for example, sasadm@saspw). Select the **Plug-ins** tab.
- 2 (Optional) Examine the current settings.
 - a Expand **Server Manager**, right-click the **SASApp** application server  and select **Properties**.
 - b On the **Authorization** tab, select **SASUSERS**. Notice that this group (which includes all registered users) has both ReadMetadata and WriteMetadata

permissions. The application server inherits these grants from the standard repository-level settings. Click **OK** to close the dialog box.

- 3 If you do not already have an ACT with the appropriate pattern, create a new ACT as follows:
 - a Expand **Authorization Manager**, right-click **Access Control Templates**, and select **New Access Control Template**.
 - b On the **General** tab, enter a name such as **Protect**.
 - c On the **Permission Pattern** tab, define settings that you want to apply to all server metadata.

Table 7.2 Example: Pattern for the Protect ACT

Group	Permission Pattern	
PUBLIC	Deny	WriteMetadata, WriteMemberMetadata, CheckInMetadata, Write, Administer
SAS Administrators	Grant	WriteMetadata, WriteMemberMetadata, CheckInMetadata, Write, Administer, ReadMetadata ¹

¹ These grants ensure that administrators can manage all metadata.

TIP To increase reusability, this pattern includes permissions that are not relevant for server definitions.

- d On the **Authorization** tab, add explicit controls to protect the ACT that you are creating:
 - Select **PUBLIC** and deny WriteMetadata permission. Leave the indirect ReadMetadata setting in place.
 - Select **SAS Administrators** and grant WriteMetadata permission. Leave the indirect ReadMetadata setting in place.
- TIP** If the **Users and Groups** list box on the ACT's **Authorization** tab is empty, click **OK** to save the ACT. Then, right-click the new ACT, select **Properties**, and select the **Authorization** tab again.
- e Click **OK**.
 - 4 (Optional) If you want to enable some nonadministrators to assign libraries, stored processes, or an OLAP schema, and you do not already have a group that represents those users, create a new custom group.
 - a Right-click **User Manager** and select **New** ⇒ **Group**.
 - b On the **General** tab, enter a name such as **Data Admins**.

- c On the **Members** tab, move users (or groups) to the **Selected Identities** list box.
- d Click **OK** to save the new group. You will grant WriteMetadata permission to this group in step 5c. This group does not participate in the ACT's pattern because this group needs WriteMetadata permission on only some of the target objects.

Note: As an alternative to creating a group for only this purpose, you can skip this step and instead assign the permissions to specific users in step 5c.

5 To set the protections:

- a Expand **Server Manager**, right-click the first application server  (if this is your first pass) and select **Properties**.
- b On the **Authorization** tab, click **Access Control Templates**. In the Add and Remove Access Control Templates dialog box, move the **Protect** ACT to the **Currently Using** list box (you have to expand the **Foundation** node to get to the ACT). Click **OK** to return to the **Authorization** tab.

Note: Review the revised settings. Notice that SASUSERS is now denied WriteMetadata permission and that PUBLIC and SAS Administrators have some ACT settings. These settings come from the **Protect** ACT.

- c (Optional) If the current item is an application server  to which nonadministrators will assign libraries, OLAP schemas, or stored processes, click **Add**, add one or more identities to the **Authorization** tab, and give each of those identities an explicit grant of WriteMetadata permission. For example, you might assign the grant to a group (such as GroupA) or to individual users.

TIP For any users that are under change management, grant CheckInMetadata (CM), instead of WriteMetadata (WM). For more information, see [SAS Intelligence Platform: Desktop Application Administration Guide](#).

Note: Do not extend WriteMetadata access to the **SASMeta** application server, because that server should be used for only a few specialized administrative tasks.

- d Click **OK** to save the settings for this object.
- e Repeat steps 5a-d for every immediate child of **Server Manager**. Immediate children include objects such as other application servers, third-party servers, the share server, the content server, and spawners.

- f Apply the **Protect** ACT to every logical server  that is under an application server  where you added an explicit grant of WriteMetadata permission (for SASUSERS or a custom group such as **Data Admins**).

Note: This strategy protects lower-level server metadata that would otherwise inherit the nonadministrator's WriteMetadata grant from the application server. It also prevents nonadministrators who have WriteMetadata permission on the application server from deleting that application server.

See Also

[“Use and Enforcement of Each Permission” on page 84](#)

Hide Server Definitions

Introduction

In a new deployment, all registered users can see and use server definitions. You might choose to limit the availability of certain servers in any of the following circumstances:

- You want to create different levels of host access to data.
- You want to direct power users to a server with settings that offer advanced options.
- You want to direct high priority users to a server on hardware that offers superior performance.
- You want to enable users in SAS Information Map Studio to use a standard workspace server even when a logical pooled workspace server is present.

Method

If you choose to limit the availability of a server, preserve access as follows:

- Make sure that the SAS System Services group has ReadMetadata permission for server metadata. This enables the SAS Trusted User to see server definitions. This is necessary because the object spawner uses the SAS Trusted User to discover and read all server metadata.

Note: Users should not be members of the SAS System Services group. This group is for service identities. In the standard configuration, the only member of this group is the SAS Trusted User.

- Make sure that the SAS General Servers group has ReadMetadata permission for server metadata. This enables the metadata identity of the launched server to see the server definition. This is a requirement for stored process servers and pooled workspace servers. This is not a requirement for standard workspace servers.

Note: Users should not be members of the SAS General Servers group. This group is for service identities. In the standard configuration, the only member of this group is the SAS Trusted User.

- Metadata administrators should have ReadMetadata permission for all server metadata.
- Any user who will use a particular server needs ReadMetadata permission for that server, with the following exceptions:
 - The requirement for ReadMetadata permission does not apply to requests to use a client-side pooled workspace server. A user can use a client-side pooled workspace server even if that user cannot see that server definition.
 - The requirement for ReadMetadata permission is not enforced if the **Use Server Access Security** check box on a logical server's **Options** tab is present and not selected. This check box should always be selected.

To efficiently set the permissions, create an ACT that includes the baseline grants and denials that you would use when you hide any server. To enable selected users to use a particular server, supplement the ACT settings with an explicit grant of ReadMetadata permission on that server.

For example, the following table summarizes settings that you might add to provide mutually exclusive access to two server components beneath a standard workspace server that is configured for SAS token authentication:

Table 7.3 Example: Hiding Server Definitions

Object	Direct Controls ¹	
	ACTs	Explicit Grants
 SASApp - ServerA	 HideServer	 GroupA: +RM
 SASApp - ServerB	 HideServer	 GroupB: +RM

¹ The direct controls in this example do not determine which of the users who can see the server can also update or delete the server. See [“Protect Server Definitions” on page 101](#).

Someone who has ReadMetadata permission for both ServerA and ServerB (for example, a member of the SAS Administrators group) uses the first server in the object spawner's list of servers.

Instructions

Here is one way to set the permissions:

- 1 Log on to SAS Management Console as a member of the SAS Administrators group (for example, sasadm@saspw). Select the **Plug-ins** tab.
- 2 (Optional) To examine the current settings:
 - a Expand **Server Manager**, right-click the server or server component that you are limiting use of and select **Properties**.

Note: The **SASMeta** application server should have limited availability, because it should be used only as instructed (in a few specialized administrative tasks).

- b On the **Authorization** tab, select **SASUSERS**. Notice that this group (which includes all registered users) has ReadMetadata permission. The application server inherits the grant from the standard repository-level settings. Click **OK** to close the dialog box.
- 3 To create the ACT:
 - a Expand **Authorization Manager**, right-click **Access Control Templates**, and select **New Access Control Template**.
 - b On the **General** tab, enter a name such as **HideServer**.
 - c On the **Permission Pattern** tab, define baseline settings for hiding servers.

Table 7.4 Example: Pattern for the HideServer ACT

Group	Permission Pattern	
PUBLIC	Deny	ReadMetadata
SAS Administrators	Grant	ReadMetadata ¹
SAS General Servers	Grant	ReadMetadata
SAS System Services	Grant	ReadMetadata ²

¹ This grant ensures that administrators can manage all server metadata (for alternatives, see [“Separated Administration” on page 99](#)).

² This grant ensures that the SAS Trusted User (who is a member of SAS System Services) can read server metadata for the object spawner (on behalf of all users).

TIP This pattern, when applied to a standard workspace server, grants a little more access than is strictly necessary. For a standard workspace server, the SAS General Servers group does not need ReadMetadata permission. If you want to avoid this, consider omitting the SAS General Servers group from this ACT and remembering to add an explicit grant for this group when you are hiding a stored process server or pooled workspace server.

- d On the **Authorization** tab, protect the ACT that you are creating. Either apply an ACT or add explicit controls that deny WriteMetadata permission to PUBLIC and grant WriteMetadata permission to the SAS Administrators group.

Note: If the **Users and Groups** list box on the ACT's **Authorization** tab is empty, click **OK** to save the ACT. Then, right-click the new ACT, select **Properties**, and select the **Authorization** tab again.

- e Click **OK**.

- 4 (Optional) If you do not already have a group that represents the users who will use the server, create a new custom group.

- a Right-click **User Manager** and select **New ⇒ Group**.
- b On the **General** tab, enter a name such as *GroupA*.
- c On the **Members** tab, move users (or groups) to the **Selected Identities** list box.
- d Click **OK** to save the new group. You will grant ReadMetadata permission to this group in step 5c. This group does not participate in the general pattern because this group does not need ReadMetadata permission on all servers.

Note: As an alternative to creating a group for only this purpose, you can skip this step and instead assign the permissions directly to specific users in step 5c.

- 5 To set the permissions:

- a Under **Server Manager**, right-click the server that you are limiting use of and select **Properties**.
- b On the **Authorization** tab, click **Access Control Templates**. In the Add and Remove Access Control Templates dialog box, move the **HideServer** ACT to the **Currently Using** list box (you have to expand the **Foundation** node to get to the ACT). Click **OK** to return to the **Authorization** tab.

Note: If the **Currently Using** list already includes another ACT (such as the Protect ACT), do not remove that assignment.

Note: Review the revised settings. Notice that SASUSERS is now denied ReadMetadata permission and that PUBLIC and SAS Administrators have some ACT settings. These settings come from the **HideServer** ACT.

- c Click **Add**, add one or more identities to the **Authorization** tab, and give each of those identities an explicit grant of ReadMetadata permission. For example, you might assign the grant to a group (such as GroupA) or to individual users.
 - d Click **OK**.
- 6 If you are limiting use of a logical server or server component, ensure that the **Use Server Access Security** check box on the logical server's **Options** tab is selected. If the check box is not selected, requirements for ReadMetadata permission for that server and its components are not enforced. This option affects only enforcement of the ReadMetadata permission.

See Also

- [“Locked-Down Servers” on page 120](#)
- [“Mediated Access” on page 112](#)

Access to SAS Data

About Access to SAS Data	111
Mediated Access	112
Overview	112
Considerations	114
Example	115
See Also	117
Metadata-Bound Libraries	118
Overview	118
Model	118
Encryption Features	120
See Also	120
Locked-Down Servers	120
Overview	120
The Lockdown Point	121
The Lockdown State	121
Best Practices for Using the Lockdown Feature	123
The Audit.Lockdown Logger	124
How to Lock Down a SAS Server	124
Stop Server Processes to Pick Up Lockdown Changes	126
Using the Lockdown Feature with SAS Applications	126
Publishing Stored Processes to a Locked-Down Server	127
See Also	127
PROTOLIBS System Option	128
Overview	128
How It Works	128
See Also	129

About Access to SAS Data

A SAS table (data set) is stored as a file in a host operating system. In general, a SAS table can be accessed and read by anyone who has the necessary host

operating system permissions to that file. For this reason, it is essential to consider and address physical-layer access to SAS libraries and tables.

Here are some ways to protect SAS data:

- Provide only mediated access, so end users do not need or have physical access to the data. See [“Mediated Access” on page 112](#).
- Bind the data to metadata, so all SAS requests for the data are subject to metadata layer permissions. See [“Metadata-Bound Libraries” on page 118](#).
- Limit the reach of SAS servers in your deployment, so requests that are processed by those servers succeed only if the target host resources are included in a list of valid (permitted) resources. See [“Locked-Down Servers” on page 120](#).
- Use the `PROTOLIBS` option to disable the `PROTO LINK` statement. See [“PROTOLIBS System Option” on page 128](#).

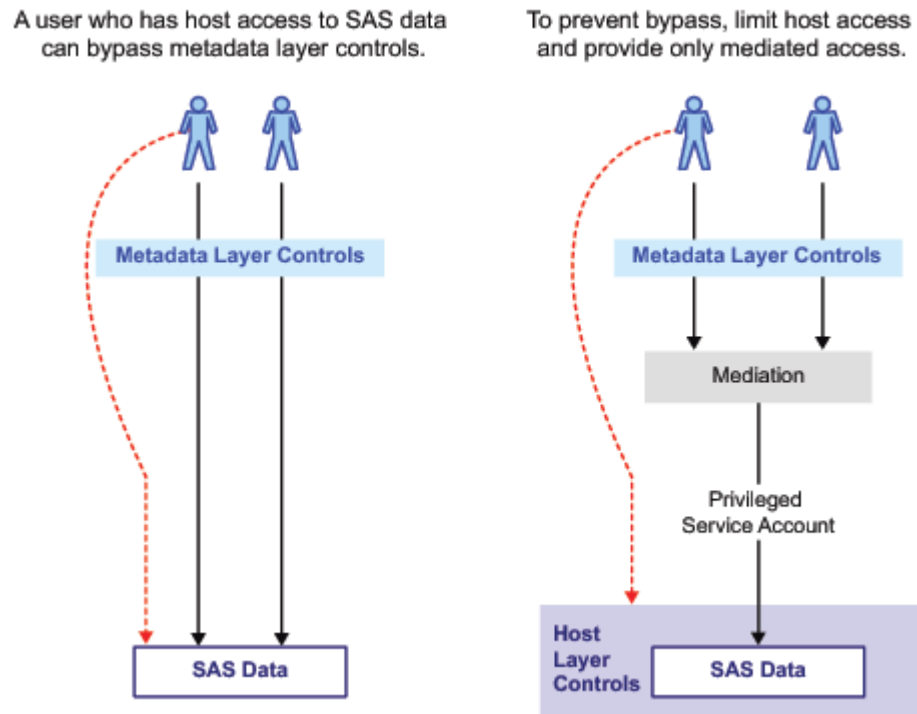
As an additional measure, you can encrypt SAS data at rest. See [“ENCRYPT= Data Set Option” in SAS Data Set Options: Reference](#).

Mediated Access

Overview

You can use the SAS metadata authorization layer to constrain access to SAS data. However, metadata-layer constraints apply only when the user requests that data in a metadata-aware context.

As a simple example, assume that you have registered a library and several tables in the metadata. In SAS Management Console, you deny userA the `ReadMetadata` permission for a table named `Salary`. This prevents userA from seeing the `Salary` table in metadata-aware applications. However, if userA has host access to the physical file that contains the table data, userA can open that file in Base SAS and examine all of the data. When the table is accessed directly, the metadata layer denial of `ReadMetadata` permission is not applied. In the following figure, the first image illustrates the issue and introduces the concept of mediated access.

Figure 8.1 Host Access to SAS Data

With mediated access, a workspace server or stored process server retrieves SAS data from the host operating system using a service account called a launch credential. The server uses one account to retrieve data on behalf of each requesting user. Mediation is an integral part of a standard configuration as follows:

- Host access from the stored process server and the pooled workspace server is always mediated.
- Host access from the standard workspace server is mediated if the server is configured to use SAS token authentication or if the server is providing client-side pooling.

Mediation offers the following advantages:

- Mediation makes pooling of workspace servers possible. For web applications, in particular SAS Web Report Studio, some form of pooling (client-side or server-side) is strongly recommended for performance reasons. Both forms of pooling always result in mediated host access to SAS data.
- Mediation facilitates partial access to a SAS table through metadata-layer controls (for example, row-level or column-level access). Host-layer controls are at the file level. Such controls can grant or deny access to only an entire table.
- Mediation helps you avoid creating end-user accounts on back-end servers that host SAS data.

CAUTION

Along with its advantages, mediation introduces some risk. If your deployment includes sensitive data in SAS tables, it is essential to review existing host-layer controls and to understand the effects of mediated access.

Considerations

Security considerations of mediated host access are as follows:

- Someone might obtain the service account ID and password and use those credentials to directly access the data.
- Someone might misuse the server in order to access data in a manner that circumvents metadata-layer controls.

The following table describes techniques for reducing risk. Keep in mind that there is no absolute security and that security goals must be balanced against other considerations.

Table 8.1 *Alternatives for Reducing Mediation Risks*

Modification	Risk Reduction	Notes
Use a different launch credential for each server. For example, configure the server-side pooled workspace server with its own dedicated launch credential.	Diversification reduces exposure for each launch credential.	Recommended if you have sensitive data in SAS tables.
Limit the availability of SAS Information Map Studio by installing that application only where it is needed.	Reduces opportunity to misuse that application.	Recommended if you have server-side pooling.
Limit the SAS folder locations from which relational information maps can be used.	Reduces opportunity to use a rogue information map, but affects only SAS Web Report Studio and SAS Web Report Viewer.	This measure is necessary as one part of making row-level permissions that are defined in information maps fully secure. See “Limit the Availability of Relational Information Maps That Implement Row-Level Security” in SAS Intelligence Platform: Web Application Administration Guide .

Modification	Risk Reduction	Notes
Set up client-side pooling using a protected, dedicated service account as the puddle logon.	Reduces opportunity to misuse the server by creating a rogue application. A client-pooled workspace server can be used from only designated web applications.	
Delete the server-side pooled workspace server (and, for performance reasons, set up client-side pooling for SAS Web Report Studio).	Eliminates opportunity to misuse that server (in SAS Information Map Studio or in a rogue application).	
Set up a restricted deployment of SAS Web Report Studio with client-side pooling.	Fully isolates the restricted client-side pool (with a dedicated pool administrator as well as a dedicated puddle logon).	See “Secure Environment for BI Row-Level Permissions” in SAS Guide to BI Row-Level Permissions .

Example

The preceding figure depicts a simple case with a single level of host access. You can establish multiple levels of host access by setting up multiple servers. Each server has its own launch credential. This strategy can require a trade off of granularity (“how many different levels of access should I establish?”) against administrative effort (“how much server metadata should I create and manage?”).

For example, the following instructions show how you can physically isolate sensitive data that is accessed through a logical workspace server that uses SAS token authentication.

- 1 In the operating system that hosts the SAS data:
 - a Establish a separate file system directory for each level of access. This enables you to avoid setting host permissions on individual files. For example, to separate HR tables, you might create an HR directory.
 - b Create or identify one service account for each level of access. For example, to provide physical protection for HR data, you might add an hrsrv account.

Note: For servers on Windows, each launch account must have the Windows privilege **Log on as a batch job**.

 - c Set operating system permissions so that each account has appropriate access to your SAS data. For example, deny the general use account (sassrv)

physical access to the HR data and grant the HR account (hrsrv) physical access to all data (including the HR data).

Note: If you want certain IT staff to be able to access the data directly using their own accounts, preserve that access.

- 2 In SAS Management Console's **User Manager**, select the SAS General Servers group, right-click, and select the **Accounts** tab. For each service account that you created in step 1, click **New** and enter the service account credentials as an additional logon for the SAS General Servers group. Include a password in each of these logons. For example, you might enter the HR logon as follows:

```
DefaultAuth | WIN\hrsrv | password
```

Note: Each of these logons are directly associated with a particular server, instead of being looked up by authentication domain. For this reason, it does not matter which authentication domain you use. You can ignore any warnings about the SAS General Servers group having multiple logons in the same authentication domain.

Note: Users should not be members of the SAS General Servers group. This group is for service identities only. Users do not retrieve these credentials, so users should not have access to these logons. In a standard configuration, these logons are available only to the privileged service identity (the SAS Trusted User) that the spawner uses to read all server metadata.

- 3 In **Server Manager**, select the logical workspace server , right-click, and select **Properties**. On the **Options** tab, verify that the **Use server access security** check box and the **SAS token authentication** radio button are selected. Click **Cancel**.
- 4 Select the logical workspace server again, right-click, and select **Add Server**. Set up one additional server  for each level of access. All of the servers can use the same port. For example, you might add a server named Human Resources.
- 5 Beneath the logical workspace server, complete these steps for each server  that you added:
 - a Select the server, right-click, select **Properties**, and select the **Options** tab. In the **Launch credentials** drop-down list, select one of the service accounts (use a different account for each server). For example, for the Human Resources server, you would first select **More logins** and then search for and select the logon that references the hrsrv account.

Note: In order to see this logon in the list, you need user administration capabilities.

- b On the **Authorization** tab, use the ReadMetadata permission to control who can use that server. Be sure to preserve necessary service access.

- 6 Under **Server Manager**, select the object spawner , right-click, and select **Properties**. On the object spawner's **Servers** tab, add the new servers to the **Selected Servers** list.
- 7 To make the changes take effect:
 - a Expand the object spawner , right-click the computer , and select **Connect**.
 - b Right-click the computer again and select **Refresh Spawner**. In the message box, click **Yes**.
 - c Right-click the computer a third time and select **Disconnect**.
- 8 To validate, expand the logical workspace server  and select a server . On each connection object  in the display area, right-click and select **Test Connection**. Test the connections for every other sibling server  that you added under the logical workspace server.

If you want to also enable power users to access data using their own accounts, set up a separate application server  for those users. In that application server, add a logical workspace server  that uses some form of host authentication. Even though regular users would not be able to authenticate (because they would not have host accounts for the workspace server machine), it is a good practice to limit ReadMetadata access to the additional application server.

You can use a similar approach to set up multiple levels of host access for a stored process server or a server-pooled workspace server. For multiple levels of host access from a client-pooled workspace server, set up multiple puddles. Always ensure that the server's host identity meets all of the requirements for a launch credential.

See Also

- [“Windows Privileges” on page 23](#)
- [“Launch Credentials” on page 159](#)
- [Choices in Workspace Server Pooling on page 357](#)

Metadata-Bound Libraries

Overview

A metadata-bound library is a physical library that is tied to a corresponding metadata object.

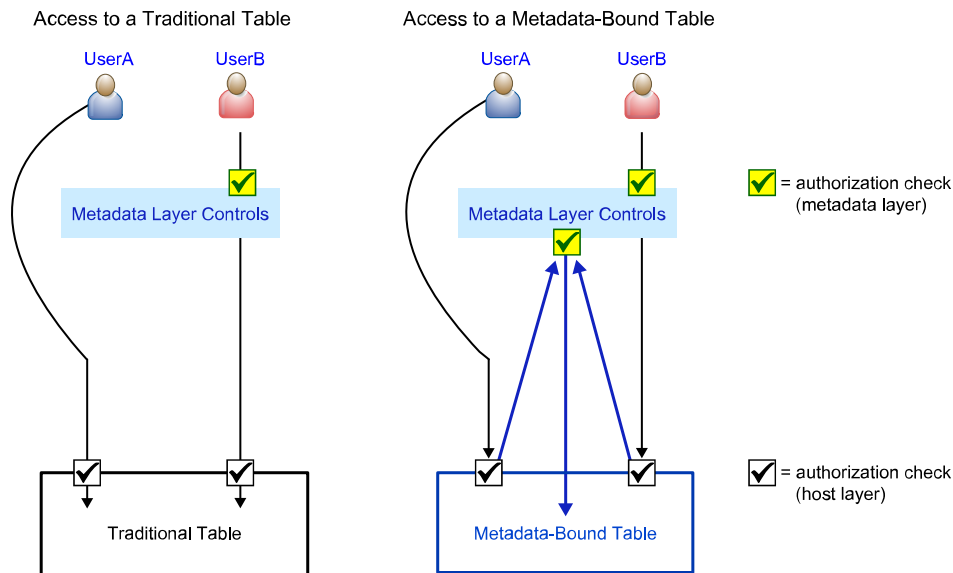
Each physical table within a metadata-bound library has information in its header that points to a specific metadata object (a secured table object). The pointer creates a security binding between the physical table and the metadata object. The binding ensures that SAS universally enforces metadata-layer permission requirements for the physical table—regardless of how a user requests access from SAS.

Access from SAS to data within a metadata-bound library is provided only if all of the following conditions are met:

- The requesting user can connect to the metadata server in which the corresponding object is stored.
- The requesting user's metadata identity has all required metadata-layer effective permissions for the requested action.
- The host identity with which the data is retrieved has all required host-layer access to the data.

Model

The following figure depicts the authorization model for a traditional table and a metadata-bound table. In both cases, UserA references the target data directly (for example, through a `LIBNAME` statement) and UserB requests the target data through a client that uses metadata to locate data (for example, SAS Web Report Studio).

Figure 8.2 Authorization Checks (by Data Type and Access Method)

The preceding figure depicts the following key difference:

- When accessing a traditional table, a user can bypass metadata-layer controls by making a direct request.
- When accessing a metadata-bound table, a user cannot completely bypass metadata-layer controls. Even on a direct request, UserA is always subject to a metadata-layer permissions check before accessing SAS data from SAS.

For the metadata-bound table, the upward-facing arrows are caused by the physical data's security binding. For each metadata-bound table, information within the table header identifies a corresponding metadata object (a secured table object). Metadata-layer permissions on each secured table object affect access from SAS to the corresponding physical table.

For the metadata-bound table, UserB is subject to two metadata-layer authorization checks against two different metadata objects.

- The first check is against a traditional table object (for example, verifying that UserB has the ReadMetadata permission).
- The second check is against a secured table object (for example, verifying that UserB has the Select permission).

TIP In the SAS metadata, traditional table objects and secured table objects are distinct and independent types of objects. For more information, see [Permissions for Metadata-Bound Data](#).

Here are some additional details about the preceding figure:

- The requesting users do not supply library or table passwords.
- The metadata-layer authorization checks are against the metadata identity of the requesting user. The host-layer authorization checks are against the identity of the SAS process that retrieves the data.

- The figure addresses access to SAS data from SAS, not interaction through host commands.
- The figure is conceptual, simplified, and abstracted. It is not intended as a detailed technical specification.

Encryption Features

Metadata-bound tables can be encrypted in the same way as traditional tables, except that the Read password is obtained from metadata instead of being supplied in SAS code. Beginning in [SAS 9.4M1](#), you can require encryption for the tables in a metadata-bound library. In addition, a metadata-bound library's AES encryption key can be stored in metadata so that the user does not have to provide the encryption key to access the data set.

See Also

[SAS Guide to Metadata-Bound Libraries](#)

Locked-Down Servers

Overview

You can limit the reach and activities of a SAS server by putting it in a locked-down state. This feature supplements the standard operating system-level and metadata-layer protections by giving SAS processes access to only specific physical resources on the server. This feature is intended for sites that need an extra level of security. For example, you might want to physically separate sensitive data from other data by placing the sensitive data on a separate, locked-down server.

The following types of servers can be locked down:

- workspace (standard)
- pooled workspace (server-side pooling)
- stored process
- batch
- grid

■ SAS/CONNECT

The lockdown feature is not supported for SAS/SHARE servers. The feature is also not supported for certain SAS products and solutions. For more information about whether a SAS product or solution supports lockdown, consult the administration guide for that product or solution.

For details about how a particular solution uses lockdown, consult the administration guide for that solution.

The Lockdown Point

During initialization of the SAS environment, all paths and files for which the operating system allows access are available. Initialization of the SAS environment includes metadata library pre-assignments and autoexec execution (which might include library pre-assignments). Having all paths and files available during initialization facilitates one-time setup of library and file assignments so that they function as usual.

When initialization is complete, SAS enters the lockdown state, which is described below.

The Lockdown State

Resources That Are Accessible to Locked-Down Servers

When a server has entered the lockdown state, the server can access only the following host resources:

- pre-assigned libraries that are defined in metadata, as well as libraries and external files that are assigned in autoexec files or through `INITSTMT` execution. Users can access these resources only by using their assigned librefs and filerefs.
- host paths and files that are included in the server's list of permitted resources. This list is referred to as the *lockdown path list*. The lockdown path list is established during server initialization and is finalized at the lockdown point. The list includes the following:
 - resources that are essential for server processing, such as system paths that are defined in SAS configuration files and paths of stored process repositories that are defined in metadata. These paths are implicitly included in the lockdown path list and do not need to be specified in the `LOCKDOWN` statement.

- ❑ for servers executing in the z/OS environment, temporary z/OS data sets that are created on behalf of clients.
- ❑ paths and files that you specify using a `LOCKDOWN` statement. You can include one or more of these statements in the autoexec file for the server. Users can access these resources (including subdirectories within the specified paths) directly by name, without relying on a libref or fileref.

TIP The lockdown path list is an *allowlist*. That is, it specifies which paths are accessible. It does not specify which paths are inaccessible.

For example, a user who connects to a locked-down workspace server from SAS Enterprise Guide can access only those host paths and files that are included in that server's list of permitted resources. Regardless of host-layer permissions, `LIBNAME` and `FILENAME` statements that users submit to a locked-down server are rejected, unless the target resource is included in the server's lockdown path list.

Note: For details about the `LOCKDOWN` statement and the contents of the lockdown path list, see [“LOCKDOWN Statement” in SAS Intelligence Platform: Application Server Administration Guide](#).

Note: When you make a change that affects the lockdown path list (for example, when you modify the `LOCKDOWN` statement or define a new stored process repository), be sure to [restart your servers to pick up the changes](#).

SAS Language Constraints for Locked-Down Servers

A server that is in the `LOCKDOWN` state is constrained as follows:

- The server cannot run DATA step Java object methods.
- The server cannot invoke certain DATA step functions.
- The server cannot run the GROOVY, HADOOP, HTTP, JAVAINFO, SOAP, or XSL procedures.
- A server running in the z/OS environment cannot perform the following:
 - ❑ run the PDS, PDSCOPY, RELEASE, SOURCE, TAPECOPY, and TAPELABEL procedures
 - ❑ execute any of the DATA step functions that are part of the SAS Data Location Assist feature. For more information, see [SAS Companion for z/OS](#).
 - ❑ use the Volume Table of Contents (VTOC) access method
- For a server running in the z/OS environment, the `DSNEXST` statement returns a value of false for a data set that is not in the lockdown path list.

For a full list of procedures, functions, and other SAS language features that are limited or disabled when SAS is in the locked-down state, see [“Language Features That Are Limited or Disabled by Default” in SAS Programmer’s Guide: Essentials](#).

File Access and the Lockdown Allowlist

When SAS is locked down, access to all local files and directories is validated through a lockdown allowlist. The lockdown allowlist is created automatically when LOCKDOWN is enabled. This allowlist contains a predefined list of resources, access methods, and files that SAS needs to execute. For more information, see [Filepaths That Are Available by Default When LOCKDOWN Is Enabled](#) and [“Default Availability of Access Methods When SAS is Locked Down” in SAS Programmer’s Guide: Essentials](#).

Some SAS functions read file paths as input and fail to execute when SAS is in the locked-down state. To ensure that these functions execute properly, an administrator must add the target paths to the lockdown allowlist.

You can add paths, access methods, and features to the lockdown allowlist by specifying the `PATH=` option or the `ENABLE_AMS=` option in the [LOCKDOWN statement](#).

For example, one or more of the access method names can be specified in the following LOCKDOWN statement:

```
LOCKDOWN ENABLE_AMS = access-method-1 <access-method-2...>;
```

Alias name pairs exist for certain access methods. If one of the access methods in the pair is enabled, the other access method is automatically enabled. For additional details about these name pairs, this statement, and some examples, see [“LOCKDOWN Statement” in SAS Intelligence Platform: Application Server Administration Guide](#).

Best Practices for Using the Lockdown Feature

When using the lockdown feature, SAS strongly recommends these best practices:

- Make sure that physical resources on the locked-down server are protected by appropriate operating system permissions. The lockdown feature is not intended to replace operating system protections.
- Make sure that the `NOXCMD` system option has been enabled on the locked-down server.
- Deny most users WriteMetadata access to the server context (for example, SASApp). Users might need WriteMetadata access to the server context for the following reasons:
 - to modify server definitions in metadata (including start-up options, ports, paths, and access controls).
 - to add or remove library definitions on the server.

- to create source code repositories on the server's file system and to publish stored processes to these repositories. For more information, see [“Publishing Stored Processes to a Locked-Down Server” on page 127](#).

The Audit.Lockdown Logger

If a user attempts to access a path that is not available, the following error is issued to the SAS log and to the Audit.Lockdown logger (if the SAS session is configured to use the SAS logging facility).

```
ERROR: The path <path> is invalid because it is not in the list of accessible
paths when SAS is in the lockdown state.
```

The following errors are issued to the SAS log, in each of the specified situations.

- If a user attempts to access a method that is not available:

```
ERROR: The access method "<name>" is disabled when SAS is in the lockdown state.
```

- If a user attempts to access a procedure that is not available:

```
ERROR: The PROCEDURE <name> is disabled when SAS is in the lockdown state.
```

How to Lock Down a SAS Server

Lock Down a Specific SAS Server

To lock down a specific SAS server (for example, a workspace server), follow these steps:

- 1 Add the `LOCKDOWN` system option to the `sasv9_usermods.cfg` file that is located in the server's configuration directory.
- 2 Add a `LOCKDOWN` global statement to the `autoexec_usermods.sas` file that is located in the server's configuration directory. Use the statement's `PATH=` argument to specify which directories and files are available to the server's users. As an alternative, you can use the `FILE=` argument to specify the name of a file that contains a list of valid directories and files.

Here are examples of typical `LOCKDOWN` statements for a workspace server. These statements provide users with access only to their home directories:

- Windows: `lockdown path="?FOLDERID_Profile";`
- UNIX: `lockdown path = "~";`
- z/OS: You can enable access to a user's personal z/OS data sets (typically cataloged under a high-level qualifier matching the user's account), to the user's UNIX home directory, or to both, as shown in the following example:

```
lockdown path=~/" ;
lockdown path="mvs:.*" ;
```

Note: When using the `FILE=` argument, if DBCS characters are included in the path list in the lockdown text file, add the `ENCODING` option to the `FILENAME` statement. This option ensures that the path list is transcoded properly to SAS session encoding.

Note: When using the `FILE=` argument, you can limit users' access to the lockdown path list by placing the file in a path that is not in the lockdown path list.

For additional details and examples, see [“LOCKDOWN Statement” in SAS Intelligence Platform: Application Server Administration Guide](#).

Lock Down a SAS Application Server Context

For simplicity, you can lock down an entire SAS Application Server context so that the constraints apply to all servers in that context. Follow these steps:

- 1 Place the `LOCKDOWN` system option in the `sasv9_usermods.cfg` file that is located in the application server's configuration directory (for example, `/Lev1/SASApp/sasv9_usermods.cfg`).
- 2 Add a `LOCKDOWN` statement to your equivalent of the following file: `/Lev1/SASApp/appserver_autoexec_usermods.sas`. In the `FILE=` or `PATH=` argument, specify which host resources are available to the servers in the context. Here is an example of a `LOCKDOWN` statement for typical application server usage:

```
lockdown path=~" "/SASData/reportdata" "/SASData/STPData" "/SASData/sales";
```

As an alternative, you can place `LOCKDOWN` statements in the `autoexec_usermods.sas` files for each of the servers in the context. This alternative is useful if you want a resource to be available only to a specific server in the context. For example, you could place the following statements in the `autoexec_usermods.sas` files for the specified servers:

- workspace server: `lockdown path="user-home-directory";` (using the appropriate syntax for your operating system)
- stored process server: `lockdown path="/SASData/STPData/";`
- pooled workspace server: `lockdown path="/SASData/ReportData/";`

Stop Server Processes to Pick Up Lockdown Changes

When you make a change that affects the lockdown path list (for example, when you modify the `LOCKDOWN` statement or define a new stored process repository), the change does not affect currently running servers. After making changes, be sure to stop or quiesce any currently running processes on the affected pooled workspace server or stored process server. For more information, see [“Use SAS Management Console to Control Processes on Servers” in SAS Intelligence Platform: System Administration Guide](#). The changes will take effect the next time a server session is started.

Using the Lockdown Feature with SAS Applications

Before SAS client applications attempt to access the file system on a locked-down server, make sure the application's default directory for server navigation is set to a location that is included in the lockdown path list. For example, take the following steps to modify the default starting path for SAS Enterprise Guide:

- 1 In SAS Management Console, open Server Manager and navigate to the appropriate application server (for example, SASApp) and logical workspace server.
- 2 Under the logical workspace server node, right-click the node for the workspace server.
- 3 Select **Properties** ⇒ **Options** ⇒ **Advanced Options** ⇒ **File Navigation**.
- 4 In **Path**, specify a path that is included in the server's lockdown path list.

Some SAS applications might require different procedures for accessing locked-down servers. For more information, consult SAS Notes or the administration documentation for the application.

TIP The lockdown feature is not supported for certain SAS products and solutions. For more information about whether a SAS product or solution supports `LOCKDOWN`, consult the administrator's guide for that product or solution.

Publishing Stored Processes to a Locked-Down Server

If users need to be able to publish stored processes to a locked-down server, it is important to consider the effect of this activity on the lockdown path list. Users can publish stored processes using two different methods:

- by keeping stored process code in source code repositories in the server's file system. To use this method, the user must have WriteMetadata permission on the server context (for example, SASApp). Users who have WriteMetadata access to the server context can also define new source code repositories. When a server process is initiated, all stored process repository paths that have been defined in metadata for that server are implicitly added to the server's lockdown path list. Therefore, WriteMetadata access should be provided to only a limited number of users. If users need to create stored processes with source files in an external directory and WriteMetadata permission on the server context grants too much permission to those users, WriteMemberMetadata permission can be granted instead. WriteMemberMetadata permission must be granted directly on the directory object (for example, navigating to SASApp in Authentication Manager to the SP Source Directory). WriteMemberMetadata permission on the directory grants WriteMetadata permission on the objects that are contained in that directory. With this permission, the user can create stored processes in the directory; the user cannot change the directory object itself.
- by keeping stored process source code in metadata instead of in the server's file system. To use this method, the user does not need WriteMetadata access to the server context. Since source code repositories do not need to be defined, the server's lockdown path list is not affected.

See Also

- [“LOCKDOWN System Option” in SAS Intelligence Platform: Application Server Administration Guide](#)
- [“LOCKDOWN Statement” in SAS Intelligence Platform: Application Server Administration Guide](#)

PROTOLIBS System Option

Overview

Starting in SAS 9.4M6, the PROTOLIBS system option is supported. This option specifies the paths that PROC PROTO can use to find and register load modules. This option is used to disable the PROC PROTO LINK statement. The site administrator can set the PROTOLIBS system option value to NONE to restrict the PROC PROTO LINK statement.

CAUTION

It can be a security issue if the [LINK Statement](#) is used with a path to a third-party library. Hence, the PROTOLIBS system option enables the site administrator to define paths that PROC PROTO can use to find load modules.

How It Works

The PROTOLIBS option is a SAS restricted option. For more information, see [Restricted Options](#). The SAS Administrator is the only person that can set a usable value for the PROTOLIBS option. By default, the PROTOLIBS option disables load modules that are specified by the PROC PROTO LINK statement.

- The PROTOLIBS option contains a list of valid paths and keywords that the load modules specified by the PROC PROTO LINK statement can access. For more information, see [“PROTOLIBS System Option” in SAS Intelligence Platform: Application Server Administration Guide](#).
- The lockdown path list that is used by the LOCKDOWN statement is also used by the PROTOLIBS option. If the server is locked down, then the paths specified in the PROTOLIBS option are replaced by the paths specified in the LOCKDOWN statement.
- The lockdown path list is set as if it were entered on the PROTOLIBS option, unless the PROTOLIBS option has been set to **NONE**.
- The PROC PROTO LINK statement is affected. It is used to load modules.

See Also

- [“PROTOLIBS System Option” in SAS Intelligence Platform: Application Server Administration Guide](#)
- [“LOCKDOWN System Option” in SAS Intelligence Platform: Application Server Administration Guide](#)
- [“LOCKDOWN Statement” in SAS Intelligence Platform: Application Server Administration Guide](#)

Batch Tools for Metadata Authorization

About the Batch Tools for Metadata Authorization	131
Introduction	131
General Options	132
Tips and Details	133
More Information	134
Batch Tool: sas-show-metadata-access	134
Syntax	134
Examples	135
Batch Tool: sas-set-metadata-access	135
Syntax	135
Examples	137
Batch Tool: sas-make-act	138
Syntax	138
Examples	139

About the Batch Tools for Metadata Authorization

Introduction

The batch tools for access control are Java scripts that enable you to connect to a SAS Metadata Server and perform the following tasks:

- View, set or remove direct access controls (explicit grants, explicit denials, permission conditions, and directly applied access control templates) for a specified object

- Create a new access control template, update an existing access control template, and display an ACT's permission pattern

Batch utilities for metadata-layer access control are located in your equivalent of the following path: `SASHome\SASPlatformObjectFramework\9.4\Tools`.

General Options

Table 9.1 Connection Options

Option	Description
-host <i>host-name</i>	Identifies the host of the Metadata Server. This option is required if the <code>-profile</code> option is not set.
-password <i>password</i>	Specifies the password of the connecting user. This option is required if the <code>-profile</code> option is not set.
-port <i>port</i>	Specifies the port where the Metadata Server listens. This option is required if the <code>-profile</code> option is not set.
-profile <i>profile-name</i>	Specifies a connection profile. You can use this option instead of using individual options (<code>-host</code> , <code>-port</code> , <code>-user</code> , and <code>-password</code>) to provide connection information. ¹
-user <i>user-ID</i>	Specifies the user ID of the connecting user. This option is required if the <code>-profile</code> option is not set.

¹ The connection profile must exist on the same physical machine where you are running the tools. For more information, see [“Administering Connection Profiles for Desktop Applications” in SAS Intelligence Platform: Desktop Application Administration Guide](#).

Table 9.2 Logging and Help Options

Option	Description
-log <i>log-file</i>	Identifies a log file or directory.
-?	Displays online Help.

Tips and Details

Specifying Objects

You can specify any public object that can be identified by its metadata location. If you specify an object other than a folder, you must specify the object type in parentheses after the object name. Object types are case-sensitive. For a list, see [“Specifying Object Types” in SAS Intelligence Platform: System Administration Guide](#).

Note: Specifying multiple objects (by using wildcards or filter criteria) is not supported.

Specifying Permissions

You can use either full permission names (such as ReadMetadata) or permission abbreviations (such as RM) to specify permissions. See [“Use and Enforcement of Each Permission” on page 84](#).

Specifying Identities

When you specify an identity name, the tool looks first for a user with that name. If a user is not found, the tool looks for a user group with that name. To avoid any possible ambiguity, you can append the object type to the name (for example, “sasdemo(User)” or “Developers(UserGroup)”.

Quotation Marks and Line Breaks

Enclose values that contain spaces or special characters in quotation marks.

Any line breaks in example code are for improved readability. Line breaks are not used in the batch utility commands.

More Information

For general information about administrative batch tools and details about other system administration tasks that you can perform in batch, see [SAS Intelligence Platform: System Administration Guide](#).

Batch Tool: sas-show-metadata-access

Syntax

The general form of the command is as follows:

sas-show-metadata-access *connection-options* *object-path* *<special-options>*

The following list provides details:

connection-options

provide connection information for the metadata server (see [Table 9.1 on page 132](#)).

object-path

specifies the metadata location (path) and name of the target object. If you specify an object other than a folder, you must specify the object type in parentheses after the object name.

<special-options>

specify the type and scope of access information to be displayed. See the following table.

Table 9.3 *Special Options for sas-show-metadata-access*

Option	Description
-effective	Displays effective access. If this option is not set, only direct access controls are displayed. ¹
-onlyGroup <i>group-name</i>	Displays access for only the specified user group.
-onlyUser <i>user-name</i>	Displays access for only the specified user.

¹ Effective access is the net effect of all applicable permission settings (both direct access controls and inherited permissions).

Examples

For the **DemoBranch/DemoFolder** object, show all direct access controls.

```
sas-show-metadata-access -profile Admin /DemoBranch/DemoFolder
```

The output resembles the following example:

```
-grant sasdemo:Read,ReadMetadata
-grant "SASAdministrators(UserGroup)":ReadMetadata
-deny "PUBLIC(UserGroup)":ReadMetadata
-grant "SAS System Services(UserGroup)":ReadMetadata
```

For the **DemoBranch/DemoFolder/DemoTable** object, show all direct access controls that are assigned to the SAS Demo User.

```
sas-show-metadata-access -profile Admin /DemoBranch/DemoFolder/
DemoTable(Table)
-onlyUser sasdemo
```

The output resembles the following example:

```
-grant sasdemo:Read,ReadMetadata
```

For the **Shared Data** folder object, show SAS Demo User's effective access.

```
sas-show-metadata-access -profile Admin "Shared Data"
-effective -onlyUser sasdemo
```

The output resembles the following example:

```
-grant sasdemo:ReadMetadata
-deny sasdemo:Administer,CheckInMetadata,Delete,
Read,Write,WriteMetadata,WriteMemberMetadata,Create
```

Batch Tool: sas-set-metadata-access

Syntax

The general form of the command is as follows:

sas-set-metadata-access *connection-options object-path special-options*

The following list provides details:

connection-options

provide connection information for the metadata server (see [Table 9.1 on page 132](#)).

object-path

specifies the metadata location (path) and name of the target object. If you specify an object other than a folder, you must specify the object type in parentheses after the object name.

special-options

provide details about the permissions that you want to set. See the following table.

Table 9.4 *Special Options for sas-set-metadata-access*

Option	Description
-addACT <i>act-name</i>	Directly applies the specified access control template. ²
-condition <i>condition-expression</i>	Adds the specified expression (as a permission condition) to the immediately preceding explicit grant. ¹
-deny <i>identity-name:permission1<,permission2...></i>	For the specified user or group, sets explicit denials of one or more permissions.
-grant <i>identity-name:permission1<,permission2...></i>	For the specified user or group, sets explicit grants of one or more permissions.
-remove <i>identity-name:ALL</i>	For the specified user or group, removes all explicit grants or denials.
-remove <i>identity-name:permission1<,permission2...></i>	For the specified user or group, removes explicit grants or denials of one or more permissions.
-removeACT <i>act-name</i>	Removes (de-applies) the specified access control template. ²
-removeAll	Removes all direct access controls (explicit grants, explicit denials, permission

Option	Description
	conditions, and directly applied ACTs).
1 For information about permission conditions, see “Fine-Grained Controls for Data” on page 77. Permission conditions for information maps are not supported by the batch tool. 2 To reference an ACT that is stored in a custom repository, use the format <i>“repository-name/template-name”</i>.	

Examples

On the **DemoBranch/DemoFolder/DemoTable** object, add explicit grants of the Read and ReadMetadata permissions for the SAS Demo User.

```
sas-set-metadata-access -profile Admin /DemoBranch/DemoFolder/
DemoTable(Table)
-grant sasdemo:Read,ReadMetadata
```

On the **DemoBranch/DemoFolder/DemoTable** object, replace explicit grants of the Read and ReadMetadata permissions for the SAS Demo User with explicit denials.

```
sas-set-metadata-access -profile Admin /DemoBranch/DemoFolder/
DemoTable(Table)
-deny sasdemo:Read,ReadMetadata
```

On the **DemoBranch/DemoFolder** object, apply the testACT.

```
sas-set-metadata-access -profile Admin /DemoBranch/DemoFolder
-addACT testACT
```

On the **DemoBranch/DemoFolder** object, remove the testACT.

```
sas-set-metadata-access -profile Admin /DemoBranch/DemoFolder
-removeACT testACT
```

On the **DemoBranch/DemoFolder** object, remove all direct access controls.

```
sas-set-metadata-access -profile Admin /DemoBranch/DemoFolder
-removeAll
```

On the **DemoBranch/DemoFolder** object, remove any existing direct access controls and add new explicit settings of the ReadMetadata permission to block access for all restricted users other than members of the SASAdministrators group.

```
sas-set-metadata-access -profile Admin /DemoBranch/DemoFolder
-removeAll -deny PUBLIC:RM -grant SASAdministrators:RM
```

On the **DemoBranch/DemoCube/Region** object, add a permission condition that enables the SAS Demo User to see only the North members within the Region dimension.

```
sas-set-metadata-access -profile Admin /DemoBranch/DemoCube/
Region(Dimension)
-grant sasdemo:Read -condition "Region=North"
```

Batch Tool: sas-make-act

Syntax

The general form of the command is as follows:

sas-make-act *connection-options* *act-name* | | *special-options*

The following list provides details:

connection-options

provide connection information for the metadata server (see [Table 9.1 on page 132](#)).

act-name

specifies the name of the target (or new) ACT.

Note: To reference a template that is stored in a custom repository, use the format "*repository-name/template-name*".

-create

creates a new access control template.

-update

updates an existing access control template.

-show

displays the permission pattern of an access control template.

-desc

specifies optional descriptive text for an access control template.

special-options

specify the permission patterns that are to be added to a new or existing ACT, or specify that all permission patterns are to be removed from an existing ACT. See the following table.

Table 9.5 *Special Options for sas-make-act*

Option	Description
-deny <i>identity-name:permission1<,permission2...></i>	For the specified user or group, adds denials of one or more permissions to the permission pattern of the new ACT.

Option	Description
-grant <i>identity-name:permission1<,permission2...></i>	For the specified user or group, adds grants of one or more permissions to the permission pattern of the new ACT.
-remove <i>identity-name:ALL</i>	For the specified user or group, removes all explicit grants or denials.
-remove <i>identity-name:permission1<,permission2...></i>	For the specified user or group, removes explicit grants or denials of one or more permissions.
-removeAll	Removes all settings from the permission pattern.

Examples

Display the permission pattern of the repository ACT (for the foundation repository).

```
sas-make-act -profile Admin "Default ACT" -show
```

Here is some example output from the preceding command:

```
-grant "SASAdministrators(UserGroup)":ReadMetadata,WriteMetadata,
CheckInMetadata,Administer
-grant "SASUSERS(UserGroup)":ReadMetadata,WriteMetadata,CheckInMetadata
-deny "PUBLIC(UserGroup)":ReadMetadata,WriteMetadata,WriteMemberMetadata,
CheckInMetadata,Read,Write,Administer,Create,Delete,Execute,
Create Table,Drop Table,Alter Table,Select,Insert,Update
-grant "SAS System Services(UserGroup)":ReadMetadata,WriteMetadata
```

In the permission pattern of the repository ACT (for the foundation repository), add grants of the Read and Write permissions for a group named Dev.

```
sas-make-act -profile Admin "Default ACT" -update -grant
Dev:Read,Write
```

Create a custom ACT called Hide that blocks the ReadMetadata permission for everyone except the SAS Administrators and SAS System Services groups.

```
sas-make-act -profile Admin Hide -create -deny PUBLIC:RM
-grant "SASAdministrators":RM -grant "SAS System Services":RM
```

TIP ACTs are not automatically protected against changes or deletion. Set permissions on each new ACT (either using the **Authorization** tab in SAS Management Console or using the `sas-set-metadata-access` batch tool).

PART 3

Authentication

Chapter 10	
Authentication Model	143
Chapter 11	
Authentication Mechanisms	167
Chapter 12	
Authentication Tasks	195

Authentication Model

<i>Introduction to the Authentication Model</i>	144
See Also	144
<i>Authentication to the Metadata Server</i>	144
How SAS Identity Is Determined	144
Authentication Process and Methods	145
<i>Authentication to Data Servers and Processing Servers</i>	146
See Also	147
<i>Mixed Providers</i>	148
About Mixed Providers	148
Solution to Mixed Providers: Align Authentication	148
Solution to Mixed Providers: Use SAS Token Authentication	149
Solution to Mixed Providers: Store User IDs and Passwords	150
See Also	150
<i>Credential Gaps</i>	150
Overview	150
Troubleshooting Credential Gaps	151
<i>How Logins Are Used</i>	152
See Also	154
<i>Authentication Domains</i>	155
What is an Authentication Domain?	155
How Many Authentication Domains Do I Need?	155
Outbound and Trusted Authentication Domains	156
<i>Identity Passing</i>	156
About Identity Passing	156
How SAS Servers Preserve Identity	157
See Also	158
<i>Launch Credentials</i>	159
About Launch Credentials	159
Criteria for a Designated Launch Credential	159
How to Create and Designate a New Launch Credential	162
Who Can Launch a Standard Workspace Server?	163
See Also	164
<i>PUBLIC Access and Anonymous Access</i>	164
About PUBLIC Access and Anonymous Access	164
How to Enable PUBLIC Access	165

Introduction to the Authentication Model

No single mechanism is applicable to all authentication events throughout a typical SAS deployment. Each deployment uses some combination of authentication processes, trust relationships, and single sign-on technologies. This model helps to balance a range of goals such as the following:

- preserve individual identity
- minimize security exposures
- provide a unified user experience
- minimize setup and maintenance efforts
- provide access to disparate systems within an environment
- integrate into a wide variety of general computing environments

See Also

[“How to Facilitate Authentication” on page 203](#)

Authentication to the Metadata Server

How SAS Identity Is Determined

When a user launches a SAS client, the following process occurs:

- 1 In the verification phase, the system ensures that the user is who he or she claims to be. For example, this credential-based host authentication method might be used:
 - a The client prompts the user for an ID and password.
 - b The user enters credentials that are known to the SAS Metadata Server host.
 - c The client sends the credentials to the Metadata Server.

- d The Metadata Server passes the credentials to its host for authentication.
 - e If the host determines that the user has a valid account, the host returns the authenticated user ID to the Metadata Server.
- 2 In the SAS identity phase, the system resolves the authenticated user ID to a particular SAS identity. In this phase, SAS examines its copies of user IDs in an attempt to find one that matches the authenticated user ID. One of the following outcomes occurs:

- A matching user ID is found, so a connection is established under the owning identity. The owning identity is the user or group whose definition includes a login with the matching user ID.

Note: Integrity constraints ensure that there will not be more than one owning identity. See [“Uniqueness Requirements” on page 27](#).

Note: Not all applications allow a group identity to log on.

- No matching user ID is found, so a connection is established under the generic PUBLIC identity. In the metadata layer, the user is a PUBLIC-only user.

Note: The matching process expects the SAS copy of the user ID to be qualified (if it is a Windows user ID).

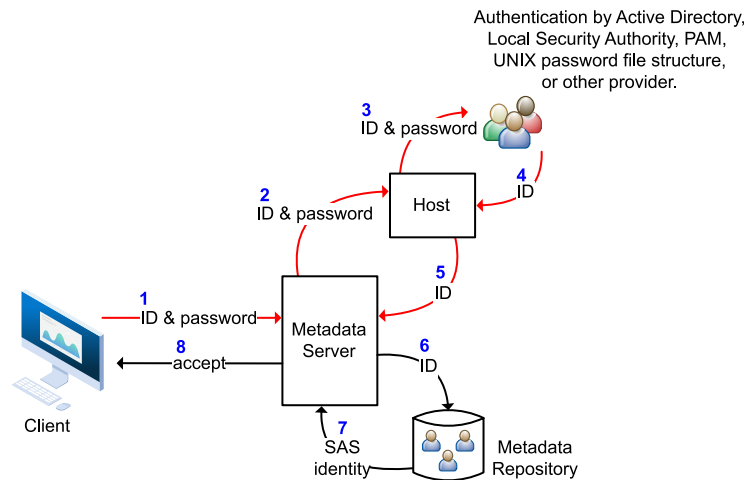
Note: Not all applications allow a PUBLIC-only user to log on. See [“PUBLIC Access and Anonymous Access” on page 164](#).

Authentication Process and Methods

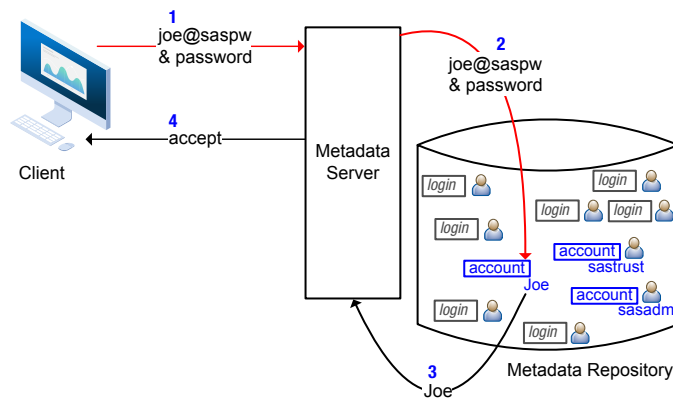
The following figures introduce the SAS Metadata Server authentication process and methods. In the following figures, notice these points:

- Only the verification phase varies; the SAS identity phase is always the same. With any approach, you need a well-formed user definition for each user account that is not a PUBLIC-only identity.
- Except where internal accounts are used, the process always involves two sets of identity information: one set that is stored in an external provider and another set that is stored in the metadata.

The following figure depicts the basic process.

Figure 10.1 Metadata Server: Host Authentication (Credential-Based)

The following figure depicts a special case where a metadata administrator named Joe uses an internal account.

Figure 10.2 Metadata Server: Internal Authentication

For alternate approaches, see “Integrated Windows Authentication” on page 174, “Web Authentication” on page 185, and “Summary of Methods for LDAP Integration” on page 190.

Authentication to Data Servers and Processing Servers

A registered user who has a connection to the SAS Metadata Server accesses the OLAP server, SAS Stored Process Server, and SAS Pooled Workspace Server seamlessly by SAS token authentication. Authentication to the following servers requires coordination:

standard SAS Workspace Server

To provide seamless access, coordinate the SAS Workspace Server with the SAS Metadata Server.

third-party data server (for example, Oracle)

To provide access, select an approach from the following table.

Table 10.1 *Authentication to a Third-Party Data Server*

Goal	Approach
Provide seamless access and preserve individual identity to the target server.	Store individual user IDs and passwords in the metadata (each on the Accounts tab of a different user definition).
Provide seamless access.	Store the user ID and password for one shared account in the metadata (on the Accounts tab of a group definition).
Provide seamless access with a few distinct access levels for resources in the target server.	Store a few shared user IDs and passwords in the metadata (each in the Accounts area of a different group definition). Or, for a hybrid approach, use a combination of personal and group logins.
Preserve individual identity.	No configuration is required. Users are prompted for credentials for the target server. Secondary prompting is supported for desktop applications and SAS Web Report Studio.

Note: In general, requesting users can access only those servers for which they have the ReadMetadata permission. An exception is client-side pooling, in which access depends on membership in a puddle group.

Note: This topic is about metadata-aware connections. Direct connections to a SAS server cannot use SAS token authentication. Direct connections use client-supplied credentials or, in some cases, Integrated Windows Authentication.

See Also

- [“Coordinate the Workspace Server” on page 204](#)
- [“How to Store Passwords for a Third-Party Server” on page 237](#)
- [“SAS Token Authentication” on page 181](#)
- [Choices in Workspace Server Pooling on page 357](#)

Mixed Providers

About Mixed Providers

If the credentials with which users initially log on are not also valid for the SAS Workspace Server, access to the standard Workspace Server is not seamless. For example, if you use an Active Directory account in an initial logon to a SAS Metadata Server on Windows, and you attempt to access a standard SAS Workspace Server on UNIX, you might be prompted for a user ID and password that are valid for the UNIX host.

Note: In general, web applications and SAS Information Map Studio use the SAS Pooled Workspace Server, not the standard SAS Workspace Server. Therefore, users who use only these applications might not be affected by a mixed provider situation.

In a mixed providers situation, achieving seamless access to the standard SAS Workspace Server requires a trade-off between configuration effort, maintenance effort, and degree of segregation for host access from that server to SAS data. The following table outlines the choices.

Table 10.2 *Choices for Seamless Access in a Mixed Provider Scenario*

Priority	Recommendation
Preserve each user's identity.	Align authentication so that both servers use the same provider.
Minimize administrative effort.	Convert the SAS Workspace Server to use SAS token authentication.

The following topics provide details about each approach.

Solution to Mixed Providers: Align Authentication

If you can enable the Metadata Server and the standard SAS Workspace Server to use the same provider, access to the Workspace Server is seamless. Here are some examples:

- If the SAS Metadata Server is on Windows and the SAS Workspace Server is on UNIX, you might extend the UNIX host authentication process to recognize Windows accounts (using a technology such as PAM). With this approach, the Windows credentials that users supply in their initial logon to the SAS Metadata Server can be reused for authentication to the SAS Workspace Server. User identity is preserved from the Workspace Server to the host, you do not have to store user passwords in the metadata, and access is seamless.

Note: If you chose to use IWA in this scenario, you should configure both servers to support that authentication method. If you configure only the SAS Metadata Server to support IWA, access to the SAS Workspace Server is not seamless because no cached credentials are available for reuse.

Note: This approach usually requires that you store two logins for anyone who accesses the standard SAS Workspace Server. One login contains the user's ID in qualified format (for example, `WIN\myID`) and the other login contains the same ID in short format (for example, `myID`). Both logins can be in `DefaultAuth`. Neither login has to include a password. An exception to this requirement is if the user names on UNIX are domain qualified (in that case, only the domain-qualified user ID is stored in metadata).

- If the SAS Metadata Server is on UNIX (or z/OS) and the SAS Workspace Server is on Windows, an additional alternative is to configure direct LDAP so that the SAS Metadata Server itself recognizes Windows accounts.

Note: Direct LDAP is not supported for the SAS Workspace Server. The SAS Workspace Server can use only some form of host authentication or SAS token authentication.

Solution to Mixed Providers: Use SAS Token Authentication

If you configure the SAS Workspace Server to use SAS token authentication, access is seamless because the SAS Workspace Server's host operating system is no longer used to validate users. Instead, users are validated by SAS Metadata Server through a single-use SAS identity token. If you need to provide a few distinct levels of host-layer access, you can define a few servers, with each server using SAS token authentication and running under a distinct launch credential.

In a mixed providers situation, using SAS token authentication is preferable to using a group login for these reasons:

- SAS token authentication enables the requesting user's SAS identity to be used for metadata layer evaluations such as library pre-assignment, permissions, and auditing.

- SAS token authentication does not give users direct access to the server launch credential.
- SAS token authentication involves less transmission of reusable credentials (the client does not retrieve credentials from the metadata and send those credentials to the SAS Workspace Server).

Solution to Mixed Providers: Store User IDs and Passwords

Note: As explained in the preceding topic, storing credentials is not a first-choice solution to a mixed-providers situation.

If you store user or group passwords for the SAS Workspace Server host, access is seamless through credential retrieval. In this solution, you treat the SAS Workspace Server as if it were a third-party server such as Oracle.

See Also

- [“How to Configure SAS Token Authentication” on page 205](#)
- [“Pluggable Authentication Modules \(PAM\)” on page 177](#)
- [“Direct LDAP Authentication” on page 171](#)
- [“How to Store Passwords for the Workspace Server” on page 235](#)

Credential Gaps

Overview

A credential gap is a situation in which a user does not seamlessly access the SAS Workspace Server for any of the following reasons:

- Server configurations are incorrect, incomplete, or incompatible.
- The user’s context does not include credentials that are known to the SAS Workspace Server host.

- The user's context does not pair credentials that are known to the SAS Workspace Server host with the SAS Workspace Server's authentication domain.
- The SAS Workspace Server is running on Windows, using credential-based authentication and the user's host account does not have the Windows privilege **Log on as a batch job**.

The usual symptom of a credential gap is a prompt for a user ID and password after a user makes a request that requires a SAS Workspace Server. A credential gap can be problematic for these reasons:

- The prompts interrupt the user experience.
- Users have to know credentials that are valid for the SAS Workspace Server host and know that those are the correct credentials to provide.
- Not all middle-tier services and web applications prompt for credentials (and, without a prompt, the user request fails).

You can use the following list to help troubleshoot a credential gap.

Troubleshooting Credential Gaps

If the user initially logs on via web authentication

The user's initial logon does not add a password to the user's context. Make sure that the web application uses some form of pooling. If the problem persists, consider configuring the SAS Workspace Server to use SAS token authentication.

If the user initially logs on via Integrated Windows authentication

The user's initial logon does not add a password to the user's context. Configure the SAS Workspace Server for Integrated Windows authentication (IWA).

If the user logs on with a user ID that ends in @saspw

Tell the user that they get the prompt because they are using an internal account. When the user gets the additional prompt, they must enter a user ID and password that are known to the SAS Workspace Server host. The host account must correspond to a metadata identity that has ReadMetadata permission for the server definition. On Windows, the host account must have the **Log on as a batch job** privilege.

If the user's connection profile contains an @saspw user ID

The user's context does not pair the credentials from the user's initial logon with the DefaultAuth authentication domain. Tell the user to create a new connection profile with external credentials (and no value in the **Authentication domain** field) and try again. To ensure optimal credential reuse, users should not use the same connection profile for both internal and external accounts.

If the user's connection profile has a value other than DefaultAuth for the authentication domain

The user's context does not pair the credentials from the user's initial logon with the DefaultAuth authentication domain. Tell the user to either clear this field or enter the value `DefaultAuth` and try again.

If the user is in SAS Enterprise Guide and accessing a SAS Workspace Server that is set to prompt

Verify the logical SAS Workspace Server's **Options** settings. If the setting is intentional, tell the user to supply host credentials.

If the SAS Workspace Server is not assigned to the correct authentication domain

Credential reuse might be impaired. In most configurations, the SAS Workspace Server should be in DefaultAuth. To verify (and, if necessary, correct) the SAS Workspace Server's authentication domain assignment, select the **Plug-ins** tab in SAS Management Console, navigate to the server , select its connection object , right-click, and select **Properties**. The authentication domain assignment is on the **Options** tab.

See Also

- [“Windows Privileges” on page 23](#)
- [“Credential Management” on page 168](#)
- [“Integrated Windows Authentication” on page 174](#)

How Logins Are Used

Table 10.3 Summary of How Logins Are Used

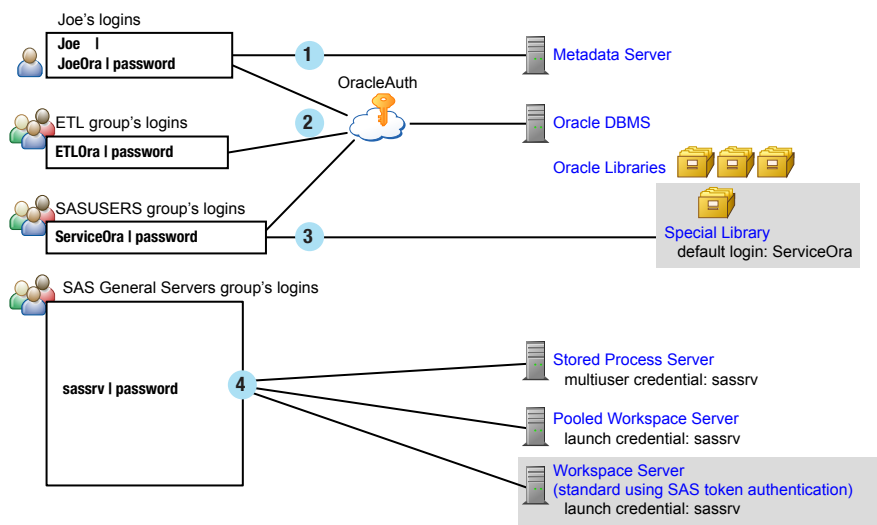
Purpose	Login Properties ¹
To enable the Metadata Server to match an incoming user ID with a particular SAS identity (inbound use).	User ID
To enable clients to seamlessly obtain user credentials for disparate systems (outbound use).	User ID, password, authentication domain
To designate one account as the preferred account for user access to a particular library and to make that account ID and password available to users. If you assign a login to a library, all users who can see that login use that login to access that library. This is a specialized form of outbound use that is sometimes used for a DBMS library.	User ID, password, authentication domain

Purpose	Login Properties ¹
To designate one host account as the account under which a particular server runs and to make that account's ID and password available to the spawner (service use).	User ID, password

1 Indicates which properties are involved. Every login should be assigned to an authentication domain.

The following figure depicts examples of login use.

Figure 10.3 Example: Use of Logins



Here are some general points about the figure:

- The SAS Workspace Server (using host authentication) is not depicted because access to that server is not usually through stored credentials.
Note: If you choose to store passwords for the SAS Workspace Server, the relationships would be comparable to the depiction of the Oracle DBMS, OracleAuth authentication domain, and Oracle logins. For example, you might place the SAS Workspace Server in WorkspaceAuth and create individual and group logins in that authentication domain.
- A group login cannot be used to manage SAS LASR Analytic Server.
- The OLAP server is not depicted because it is not accessed using logins.
- The gray shading for the depicted SAS Workspace Server indicates that this is a specialized configuration. By default the SAS Workspace Server uses some form of host authentication, not SAS token authentication.

The numbers in the figure correspond to these uses:

- 1 Joe's first login is only for inbound use to determine his metadata identity. His password is available (cached in the user context, not stored in the metadata) but is not used to determine his identity. This login should be in DefaultAuth,

but that relationship is not depicted because it is not used in determining his metadata identity.

- 2 Joe's second login provides seamless access to Oracle using an individual account. This login includes a password and must be in the Oracle server's authentication domain. The ETL group's login is a shared login for the Oracle server. Joe will not use this login because his personal Oracle login has a higher priority.
- 3 The SASUSERS login is a designated default login for the Special Library. This login is visible to Joe (through his automatic membership in SASUSERS), so it is used when Joe accesses the Special Library. Assigning a login to a library overrides the usual login priority evaluation (which is based on identity precedence).

Note: In this example, the ServiceOra login must be in the OracleAuth authentication domain. The list of available default logins for a library consists of only those logins that are in the associated server's authentication domain. The shading for the Special Library indicates that this is not a mainstream use. Most libraries do not have a designated login.

Note: In an alternate usage, the default library login is part of the user definition for a service identity that provides mediated access to the library.

- 4 The designated launch credential for each of the depicted processing servers is stored on the SAS General Servers group definition. In this example, the servers all use the same credential. Logins that contain designated launch credentials are usually in the DefaultAuth authentication domain, because these processing servers are usually in DefaultAuth. However, those logins are directly paired with each server, not looked up by authentication domain. Because the authentication domain assignment for these logins is not used, the figure does not depict that assignment.

See Also

- [“How SAS Identity Is Determined” on page 144](#)
- [“Credential Management” on page 168](#)
- [“Identity Hierarchy” on page 20](#)
- [“Criteria for a Designated Launch Credential” on page 159](#)

Authentication Domains

What is an Authentication Domain?

An authentication domain is a name that facilitates the matching of logins with the servers for which they are valid.

Note: This matching is not important when you launch a client, but it is important when you access certain secondary servers such as a third-party DBMS or, in some configurations, a standard SAS Workspace Server.

Each user ID and password is valid within a specific scope. For example, the user ID and password that you use to log on to your computer at work are probably not the same as the user ID and password that you use to log on to a personal computer at home. It is also common for database servers and web servers to have their own authentication mechanisms, which require yet another, different, user ID and password.

An enterprise application that provides access to many different resources might require that a user have several sets of credentials. Each time a user requests access to a resource, the software must determine which credentials to use to provide access. The software could challenge the user with an interactive prompt for user ID and password, but that quickly becomes an annoyance that interrupts the user experience. The software could randomly try different credentials until it finds a set that works, but authentication attempts can be expensive in terms of performance. In the SAS Intelligence Platform, the software attempts to use only the credentials that it expects to be valid for a particular resource or system.

The software's knowledge of which credentials are likely to be valid is based entirely on authentication domain assignments. For this reason, you must correctly assign an authentication domain to each set of resources that uses a particular authentication provider, and also assign that same authentication domain to any stored credentials that are valid for that provider.

How Many Authentication Domains Do I Need?

In the simplest case, all logins and SAS servers are associated with one authentication domain (DefaultAuth). This list describes the most common reasons for using more authentication domains:

- If you use web authentication, you might need a second authentication domain for the logins that contain web realm user IDs.

- If you have a third-party server (such as a DBMS server) that has its own user registry, you need a separate authentication domain for that server and its logins.
- If both of the following criteria are met, you need a separate authentication domain for the standard SAS Workspace Server and its logins:
 - The standard SAS Workspace Server does not share an authentication provider with the SAS Metadata Server (and cannot be configured to do so).
 - You want to provide seamless individualized access to the standard SAS Workspace Server.

Outbound and Trusted Authentication Domains

Beginning in [SAS 9.4M2](#) is support for outbound and trusted authentication domains. A login in an outbound domain is used only to provide SAS applications with seamless access to external resources, such as a third-party vendor database. These logins are not part of the SAS identity phase, which attempts to determine the current metadata user by matching their authenticated user ID to the user ID stored in a login. Therefore, for outbound domain logins, the uniqueness requirement on the user ID is not enforced.

Outbound authentication domains might also be set as trusted domains. A login in a trusted domain can be accessed only by a trusted user on behalf of the user. The identity to which the login is associated does not have direct access to the login (they do not see it listed on the **Accounts** tab in SAS Management Console). Instead, the login is used on their behalf by a trusted user.

Administration of logins in trusted authentication domains is performed only by unrestricted users, user administrators, and users granted the `ManageCredentialMetadata` permission.

You can use SAS Management Console to define and administer outbound and trusted authentication domains. See [SAS Management Console: Guide to Users and Permissions](#).

Identity Passing

About Identity Passing

When a request is passed from one system to another, it is often preferable that the requesting user identity is passed along with the request. This provides individualized access control evaluations and individual accountability in the receiving system. Here are a few examples:

- When a stored process server is launched, it has to use some SAS identity to contact the SAS Metadata Server in order to discover any pre-assigned libraries.
- When a user opens a SAS folder that contains cubes, the OLAP server has to use some SAS identity to contact the SAS Metadata Server to determine which cubes to show to that user.
- When a SAS Workspace Server fetches SAS data from the operating system, it has to use some host identity to authenticate to the operating system and request the SAS data set files.

In each example, the identity that is used affects the outcome (for example, which libraries are available, which cubes are visible, or which data files are returned).

How SAS Servers Preserve Identity

The following table provides details about how SAS servers preserve identity.

Table 10.4 *Preservation of User Identity*

Server	SAS Identity for Metadata Layer Evaluations	Host Identity for Host Layer Evaluations
OLAP server	The SAS Trusted User. However, user identity is preserved because this privileged service identity impersonates each requesting user. ¹	The server's host identity. This affects drilling to detail if the underlying data is in SAS tables.
Stored process server	Each requesting user's SAS identity. The exception is for library pre-assignment, which happens under the server identity. ²	The server's host identity (the multi-user credential).
Workspace server (using any form of host authentication)	The SAS identity of the server. This is the same as the SAS identity of the requesting user or group.	The server's host identity (the host identity of the requesting user or group). ³
Workspace server (using SAS token authentication)	Each requesting user's SAS identity.	The server's host identity (the launch credential).
SAS Pooled Workspace Server	Each requesting user's SAS identity.	The server's host identity (the launch credential).
Client-pooled Workspace Server	The SAS identity of the server. However, SAS Web Report Studio pre-processes requests based on	The server's host identity (the puddle login).

Server	SAS Identity for Metadata Layer Evaluations	Host Identity for Host Layer Evaluations
--------	---	--

each requesting user's SAS identity.

- 1 Except for retrieval of DBMS credentials, which happens under the SAS Trusted User's identity. This affects drilling to detail if the underlying data is in a third-party DBMS.
- 2 The requesting user is not known when library pre-assignment happens (during server initialization).
- 3 Each server process instance is launched on-demand and runs under the host identity of the requesting user (or group, if authentication is via stored group logins).

Note: You can directly specify which identity to use on a connection to the SAS Metadata Server (for example, with METAUSER and METAPASS). You can directly specify the identity to use on a connection to a host (for example, by providing explicit credentials in your code).

In the preceding table, notice that although metadata layer evaluations are almost always individualized (for example, "Does Joe have metadata layer permission to see this stored process?"), physical layer evaluations are often generalized (for example, "Does the sassrv account have host layer permissions for this data?").

In particular, requests from metadata-aware clients for SAS data often are not evaluated under the host identity of each requesting user. The host does not know the user's SAS identity, so the host cannot evaluate the user's operating system permissions. Instead, the host checks the permissions of the service account that is fetching the SAS data on behalf of the requesting user. These service accounts are referred to as *launch credentials*.

See Also

- ["Launch Credentials" on page 159](#)
- ["Mediated Access" on page 112](#)
- [Choices in Workspace Server Pooling on page 357](#)

Launch Credentials

About Launch Credentials

Workspace servers and stored process servers use different process instances (SAS sessions) for different requests. Each process instance is launched by a spawner under a host account as follows:

- For the stored process server, the SAS Pooled Workspace Server, and a standard SAS Workspace Server that uses SAS token authentication, the spawner launches process instances under a designated service account. That account is called a launch credential (or, sometimes, a multi-user credential) and is specified as part of the server's metadata definition. For example, in the initial configuration, the SAS Spawner Servers account (sassrv) is the only designated launch credential.
- For a standard SAS Workspace Server that uses host authentication, the spawner launches a new process instance on-demand for each requesting user. The server's metadata definition does not include a designated launch credential. Instead, each requesting user's personal host account is used to launch that user's process instances. Or, if the server is accessed using group logins, a group account might be used. As a result, only users who can authenticate to the SAS Workspace Server host can get a workspace server.

The following figure depicts a deployment where the sassrv account is the launch credential for all servers that run under a designated account. The last server in the figure is shaded because it depicts a specialized configuration and cannot coexist within an application server that includes another standard SAS Workspace Server.

Figure 10.4 Example: Launch Credentials for Processing Servers



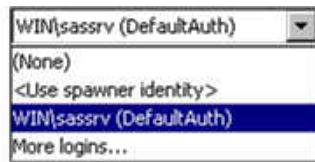
Note: If you use client-side pooling, the puddle login is comparable to the launch credential.

Criteria for a Designated Launch Credential

The **Launch credentials** (or **Multiuser credentials**) setting is available on the **Options** tab for a standard SAS Workspace Server that uses SAS token

authentication, the SAS Stored Process Server, and the SAS Pooled Workspace Server.

Figure 10.5 Choosing a Launch Credential



Note: In the preceding display, the values within parentheses indicate the authentication domain of the credential. These values are not part of the service account ID. For example, the selected account ID is `WIN\sassrv`, not `WIN\sassrv (DefaultAuth)`.

You should carefully select each server's launch credential for the following reasons:

- Not all of the choices are viable without additional configuration.
- Each server retrieves SAS data from the operating system under the host identity of its designated launch credential. Anyone who can use a particular server can potentially access all of the data that is available to that server's launch credential.

The following list explains the choices in the launch credential list:

(None)

prevents proper functioning of the server. If the **Multuser credentials** or **Launch credentials** drop-down list is present and enabled on the **Options** tab for a server , that server is not functional when **(None)** is selected.

Use spawner identity

can introduce the following complications:

- On Windows, using the spawner's host identity as a launch credential causes launched processes to run with restricted access. For example, even if the spawner runs under the local system account, the host identity of the launched process is not a member of Windows groups such as Administrators and Power Users. The downgrade is necessary in order to avoid security exposures. The downgrade can interfere with legitimate operation of the server.

Note: In the initial configuration, a SAS spawner on Windows runs as a service under the local system account. You can reconfigure the SAS spawner to run under some other account.

- In the standard configuration, the spawner's SAS identity (PUBLIC) cannot launch a SAS Pooled Workspace Server or a SAS Stored Process Server. As a PUBLIC-only identity, the SAS spawner lacks the necessary grant of the ReadMetadata permission on the server definition.

Note: One solution is to create an individual SAS identity for the spawner account. For example, create a user definition named SPAWNER and store the user ID of the spawner account as a login on the **Accounts** tab of that user definition. In the initial configuration, all registered users (SASUSERS) have ReadMetadata permission for server definitions. If you narrow access, you might have to add a grant of ReadMetadata permission on the server's **Authorization** tab for the SPAWNER user.

Note: Ensure that the server's launch credential has any necessary host access to the SAS Web Report Studio query cache library (wrstemp) and distribution library (wrsdist). See [“Configuring SAS Web Report Studio” in SAS Intelligence Platform: Web Application Administration Guide](#).

A listed login (for example, **WIN\sassrv**) or a login that you access by selecting

More logins

is the standard choice. In order to successfully function as a launch credential, a login must meet all of the following criteria:

- The login that must be visible to the spawner. All of the logins on the **Accounts** tab of the SAS General Servers group are visible to the spawner.

Note: The SAS Trusted User is a member of the SAS General Servers group. The spawner uses the SAS Trusted User to gather the metadata information needed to launch servers.

- The login must include the user ID and password for an account that is known to server's host. It does not matter which authentication domain the login is in.

Note: If the server is on Windows, the user ID in the login must be in a qualified format (for example, WIN\serviceID).

- The login must be owned by a SAS identity that has the ReadMetadata permission for the server definition. In the standard configuration, this requirement is met because the SAS General Servers group, like all other registered identities, has ReadMetadata permission for all server definitions. However, if you choose to limit ReadMetadata permission on a server definition, you must preserve access for the SAS General Servers group.
- The login must reference an account that has host layer access to any SAS data that it retrieves.
- If the server is on Windows, the login must reference an account that has the **Log on as a batch job** privilege.

Note: The login should reference a service account. This login should not correspond to a real person.

Note: Ensure that the server's launch credential has any necessary host access to the SAS Web Report Studio query cache and distribution libraries.

The preceding discussion assumes that your deployment uses the standard predefined metadata groups and users. You can choose to configure variations (for example, create a group other than the SAS General Servers group to hold logins for launch credentials). In general, such variations are not recommended because they unnecessarily increase complexity and reduce consistency.

How to Create and Designate a New Launch Credential

- 1 Identify or create the host account.
 - If the server retrieves SAS data, make sure the host account has appropriate access to those files.
 - If the server is on Windows, assign the Windows privilege **Log on as a batch job** to the account.
- 2 Log on to SAS Management Console as someone who has user administration capabilities (for example, sasadm@saspw). Other users cannot add logins to a group definition or see logins that they do not own.
- 3 In **User Manager**, create a login on the **Accounts** tab of the SAS General Servers group. This login must include both a user ID and a password.

Note: If the server is on Windows, the user ID in the login must be in a qualified format (for example, WIN\serviceID).

Note: You can ignore any warnings about having multiple logins in the same authentication domain. These logins are directly associated with servers. These logins are not looked up by authentication domain.

Note: You cannot reuse a login that already exists on the **Accounts** tab for some other user or group. The Metadata Server's integrity constraints prevent you from assigning the same user ID to two different SAS identities.

Note: You can use the same login as a launch credential for more than one server.

Note: Users should not be members of the SAS General Servers group. This group is for service identities only. Users do not retrieve these credentials, so users should not have access to these logins. In a standard configuration, these

logins are available only to the privileged service identity that the spawner uses to read all server metadata (the SAS Trusted User).

- 4 In **Server Manager**, on the server's **Options** tab, select the login as the server's launch credential (or multiuser credential).

Note: The launch credential is designated at the level of the server component , not at the level of the logical server .

Note: To locate a login that is not displayed in the drop-down list, select the **More logins** entry. Although all logins are displayed in the secondary dialog box, any login that is not visible to the SAS Trusted User is not a viable choice.

- 5 To make the changes take effect, re-initialize the spawner.
 - a Expand the object spawner , right-click the computer , and select **Connect**.
 - b Right-click the computer again and select **Refresh Spawner**. In the message box, click **Yes**.
 - c Right-click the computer a third time and select **Disconnect**.
- 6 Validate the server.

Who Can Launch a Standard Workspace Server?

In order to launch a standard SAS Workspace Server that uses any form of host authentication, a user must have access to an individual or group account that meets all of the following criteria:

- The account must be known to the SAS Workspace Server host. An internal account (such as sasadm@saspw) does not meet this requirement.
- The account must have host-layer access to any SAS data that it retrieves.
- If the server is on Windows, and the user does not connect through Integrated Windows authentication, the account must have the **Log on as a batch job** privilege.
- If the user supplies credentials interactively (for example, at a secondary logon prompt), the account must correspond to a SAS identity that has the ReadMetadata permission for the SAS Workspace Server definition. In the initial configuration, all registered users meet this requirement, but a PUBLIC-only identity (someone who does not have a well-formed user definition) does not meet this requirement.

Note: If the **Use Server Access Security** check box on the **Options** tab of the logical server is cleared, this requirement does not apply.

See Also

- [“Identity Passing” on page 156](#)
- [“Mediated Access” on page 112](#)
- [Choices in Workspace Server Pooling on page 357](#)
- [“Windows Privileges” on page 23](#)

PUBLIC Access and Anonymous Access

About PUBLIC Access and Anonymous Access

In general, only users who can authenticate and who have a well-formed user definition should use a SAS deployment. However, in order to accommodate scenarios where more general access is desired, the following specialized configurations are supported:

- PUBLIC access enables unregistered users to participate if they can authenticate to the Metadata Server (directly or through a trust mechanism). Unregistered users are referred to as PUBLIC-only users because their only SAS identity is that of the PUBLIC group. A PUBLIC-only user has the logins, permissions, and capabilities of the PUBLIC group. A PUBLIC-only user cannot belong to any other groups, or have any personal logins, or have any specialized (individual) access controls. Not all applications allow a PUBLIC-only user to log on.
- Anonymous access enables unregistered users to participate without authenticating to the SAS environment. Anonymous access is an optional configuration that is available for only a few applications. Anonymous access is supported only with SAS authentication; anonymous access is not compatible with web authentication. Anonymous access is supported as follows:
 - For SAS BI Web Services and the SAS Stored Process Web Application, a user who connects through anonymous access uses the SAS Anonymous Web User identity. This is a service identity that functions as a surrogate for users who connect without supplying credentials. See [“Use the SAS Anonymous Web User with SAS Authentication” in SAS Intelligence Platform: Middle-Tier Administration Guide](#).
 - For the SAS Information Delivery Portal, a user who connects through anonymous access uses the Unchallenged Access User identity. This is a service identity that functions as a surrogate for users who connect without supplying credentials. See [“Enabling Unchallenged Portal Access” in SAS Intelligence Platform: Web Application Administration Guide](#).

PUBLIC access and anonymous access differ in the following ways:

- In PUBLIC access, each participating user must authenticate. In anonymous access, participating connections do not require user authentication.
- In PUBLIC access, participating users share the PUBLIC group identity. In anonymous access, participating connections share a designated service identity (the surrogate identity is always a member of both the SASUSERS group and the PUBLIC group).
- You can choose to provide wide support for PUBLIC access. You cannot extend support for anonymous access beyond the specific applications that can be configured to use it.

CAUTION

If you choose to offer PUBLIC or anonymous access, you risk enabling users to see more data and content than you might expect. Carefully review and manage access control for the PUBLIC group. If you offer anonymous access, carefully review and manage access control for your surrogate service identity too.

How to Enable PUBLIC Access

You can use the following instructions to set up a specialized configuration in which unregistered users can participate from most SAS applications. However, not all SAS applications can be configured to accept PUBLIC-only users. For example, SAS Visual Analytics never accepts PUBLIC-only users.

To enable PUBLIC access, follow these steps in SAS Management Console:

- 1 Provide the necessary repository-level access.

On the **Plug-ins** tab, under **Authorization Manager** ⇨ **Access Control Templates**, right-click the repository ACT (**Default ACT**) and select **Properties**. On the **Permission Pattern** tab, grant the ReadMetadata and WriteMetadata permissions to PUBLIC.

Note: Even users who consume only content need both of these permissions at the repository level, because some applications write system information about user activity, even during what appears to be a view-only transaction.

- 2 Provide Read access as needed.

On the **Folders** tab, give the PUBLIC group the Read permission for any information maps, cubes, and MLE data that you want to make universally available. A good approach is to create a folder branch for such content, set the grant on the top folder in that branch, and allow the grant to flow through the branch.

Note: Users also need ReadMetadata permission to folders and content objects. In general, it is not necessary to set specific grants because this

permission must flow through from the repository ACT into the public areas of the folder tree (for navigational purposes).

Note: If you want to allow everyone (including unregistered users) to contribute content to a particular folder, give the PUBLIC group a grant of the WriteMemberMetadata permission on that folder's **Authorization** tab.

3 Review role assignments for the PUBLIC group.

On the **Plug-ins** tab, under **User Manager**, right-click the PUBLIC group and select **Properties**. Review the PUBLIC group's role memberships. Often, no adjustments are necessary, because the initial role assignments give the PUBLIC group basic capabilities.

4 Make sure that the PUBLIC group can use servers.

- a On the **Plug-ins** tab, under **Server Manager**, verify that the PUBLIC group has the ReadMetadata permission for any servers that the PUBLIC-only users access.
- b If necessary, add one or more logins on the PUBLIC group's **Accounts** tab (for example, to provide seamless access to a third-party DBMS).
- c If you have configured client-side pooling, verify that PUBLIC is a designated puddle group.

5 Configure middle tier properties applications to accept PUBLIC-only users.

On the **Plug-ins** tab, navigate to **Application Management** ⇒ **Configuration Manager**, and make changes as needed. Not all deployments include and use all components.

TIP If you need additional information, see the administrative documentation for your application. Not all deployments include and use all components. Not all middle-tier components have an applicable property.

- a For SAS Web Report Studio, set the **Allow Public Users** property to **Yes** (this property is in the **Application Configuration** section of the **Settings** tab of the properties dialog box for **Web Report Studio**).
- b For the SAS Stored Process Web application, set the **App.PublicIdAllowed** property to **true** (this property is on the **Advanced** tab of the properties dialog box for the **Stored Process Web App**).
- c To make the changes take effect, restart the SAS Web Infrastructure Services Application and then restart the affected web applications.

Authentication Mechanisms

<i>Introduction to Authentication Mechanisms</i>	168
<i>Credential Management</i>	168
See Also	171
<i>Direct LDAP Authentication</i>	171
See Also	172
<i>Host Authentication</i>	172
See Also	173
<i>Integrated Windows Authentication</i>	174
See Also	176
<i>Multifactor Authentication (MFA)</i>	176
<i>Pluggable Authentication Modules (PAM)</i>	177
See Also	178
<i>SAS Internal Authentication</i>	178
See Also	180
<i>SAS Token Authentication</i>	181
See Also	182
<i>Trusted Peer Connections</i>	182
<i>Trusted User Connections</i>	184
<i>Web Authentication</i>	185
See Also	188
<i>Microsoft Windows Defender Credential Guard</i>	188
Overview	188
Issues That Might Occur	189
SAS Support for Windows Defender Credential Guard	189
<i>Summary of Methods for LDAP Integration</i>	190
<i>Summary for Single Sign-On</i>	191
<i>Summary by Server Type</i>	192

Introduction to Authentication Mechanisms

This chapter describes the following features:

- Internal authentication mechanisms unify the SAS realm and provide a degree of independence from your general computing environment. The internal mechanisms are SAS internal authentication and SAS token authentication.
- External authentication mechanisms integrate SAS into your computing environment. External mechanisms include direct LDAP authentication, host authentication (credential-based), Integrated Windows authentication, and web authentication.
- Multifactor authentication mechanisms integrate with your existing authentication methods. Multifactor authentication (MFA) adds an extra authentication layer to your SAS deployment.
- Credential management provides single sign-on through reuse of cached credentials or retrieval of stored passwords.
- Pluggable authentication modules (PAM) extend UNIX host authentication.
- Trust relationships facilitate communication to the Metadata Server by permitting one privileged account to connect on behalf of other users (trusted user) or by accepting requests that use a proprietary protocol (trusted peer).

Credential Management

Table 11.1 *Credential Management*

Summary	A supporting feature in which clients reuse cached credentials or retrieve stored credentials. Clients use authentication domain assignments to determine which credentials are valid for which servers. The target server validates the client-supplied credentials against its authentication provider.
Scope	From clients that are already connected to SAS Metadata Server to third-party servers, the Scalable Performance Data Server, and, in some cases, the workspace server.
Benefits	Provides access to servers using individual or shared accounts.
Limits	■ Involves passing user IDs and passwords across the network.

- Can involve maintaining SAS copies of external passwords.

Use Always available.

Credential management techniques populate an in-memory list of credentials for each connected user. Each list is called a user context and includes these entries:

- If the user interactively provides credentials when launching a SAS client, those credentials are added to the list, with these exceptions:
 - If web authentication is configured, the password from a user's interactive logon to a web client is not available to be added to the list.
 - If the user logs on with Integrated Windows authentication (IWA), the user's password is not available to be added to the list.
- If the user interactively provides credentials at any point in the session, those credentials are added to the list.
- If the user's metadata definition has logins that include passwords, those credentials are added to the list.
- If the user belongs to any groups whose metadata definition has logins that include passwords, those credentials are added to the list.

Note: Credentials from a user or group's metadata definition are not included in the initial list that is created when a user logs on. Instead, such credentials are added to the list dynamically (when and if they are needed in the course of the user's session).

The following table depicts an example of the contents of a user context:

Table 11.2 Example: Contents of a User Context

User ID	Password	Authentication Domain
myWinID	*****	DefaultAuth
GroupDBMSid	*****	DBMSauth

Notice that each entry is assigned to an authentication domain. This enables pairing of credentials with the servers for which they are valid. The entries are created as follows:

- The client creates the first entry by caching and inserting the user ID and password that a user submits in an initial logon. This makes the user's DefaultAuth password available for reuse even though that password is not stored in the metadata. The client automatically assigns the first entry to the DefaultAuth authentication domain except in these circumstances:
 - The user's connection profile contains a user ID with an @saspw suffix.

Note: Notice that this circumstance describes the user ID in the user's connection profile, not the user ID that the user supplies in the logon dialog box.

- The **Authentication domain** field in the user's connection profile contains a value other than DefaultAuth (and is not empty).
- The user is accessing a web client at a site that has configured web authentication (or has specified a different authentication domain in the web configuration for some other reason).
- The client creates the second entry by retrieving information from the metadata (in the preceding example, from a group that the user belongs to). Such logins are for outbound use, so they must include a password and an appropriate authentication domain assignment.

When a user requests access to a server that requires credential-based authentication, the client completes these steps:

- 1 Examine the server's metadata to determine which authentication domain the server belongs to.

Note: In SAS Management Console, this information is displayed on the **Options** tab of each of the server's connection objects .

- 2 Examine the user's context to determine whether it includes any credentials that are assigned to the target server's authentication domain. The process is as follows:
 - If the context includes a cached entry for the target authentication domain, that entry is used.
 - If the user context contains a retrieved entry for that authentication domain, that entry is used. If there is more than one retrieved entry for an authentication domain, the entry that is closest to the user is used.
 - If there is an identity precedence tie among retrieved entries (for example, if a user is a direct member of two groups and both groups have logins in the relevant authentication domain), the same login is used consistently, but you cannot control which of the two logins is used.
 - If the user context contains no entries in the target authentication domain, desktop clients prompts the user for credentials. Web applications cannot prompt.

Note: SAS Web Report Studio has an interactive password management feature.

- 3 Present the credentials to the target server for authentication against its provider.

Here are some additional tips:

- Because the credentials that are added to a user context from an initial logon are not likely to be valid for a third-party DBMS, you usually have to store credentials for such servers.
- Authentication domains have no effect on an initial logon. Authentication domains affect access to secondary servers that perform credential-based authentication.
- To prevent attempts to reuse internal credentials for the workspace server (which does not accept internal credentials), users who have an internal account should use a dedicated connection profile for that account. In their internal connection profiles, users should leave the **Authentication domain** field blank (or specify an arbitrary value such as InternalAuth).

See Also

- [“How Logins Are Used” on page 152](#)
- [“Identity Hierarchy” on page 20](#)
- [“How to Store Passwords for a Third-Party Server” on page 237](#)

Direct LDAP Authentication

Table 11.3 *Direct Use of LDAP Authentication*

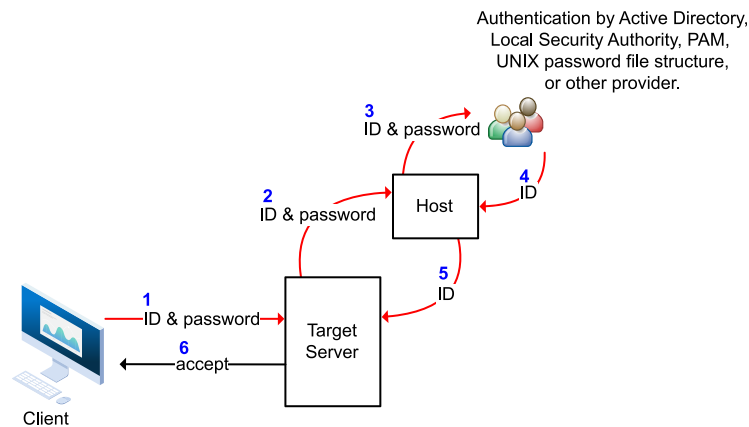
Summary	SAS Metadata Server validates some users against an LDAP provider such as Active Directory. Direct LDAP enables the Metadata Server to recognize accounts that are not known to its host; direct LDAP does not modify the host's behavior.
Scope	■ Primarily used for connections to the Metadata Server. ■ Can also be used for direct connections from a data provider to the OLAP server.
Benefits	Enables users to use their Windows accounts to authenticate to a SAS Metadata Server that runs on UNIX.
Limits	■ Not an alternative to storing user IDs in the metadata (that requirement applies to all configurations). ■ Not supported for workspace servers or stored process servers. ■ Might involve appending a special qualifier to user IDs that are stored in the metadata.
Use	Optional.

- Sometimes used for connections to the workspace server.

Benefits	No configuration is required. Can enable users to log on to SAS applications with the same credentials that they use in your general computing environment.
Limits	■ On a workspace server on Windows, requires that users have the Windows privilege Log on as a batch job. ■ Involves passing user IDs and passwords across the network.
Use	Always available.

The following figure shows one example of how this mechanism works:

Figure 11.2 Host Authentication (credential-based)



The numbers in the figure correspond to these actions:

- 1 The client obtains the user's ID and password (interactively or through credential management). The client sends those credentials to the target server.
- 2 The server passes the credentials to its host for authentication.
- 3 The host passes the credentials to its authentication provider.
- 4 After verifying that the user ID and password correspond to a valid account, the host's authentication provider returns the user's ID to the host.
- 5 The host returns the user's ID to the SAS server.
- 6 The server accepts the client connection.

See Also

- [“Windows Privileges” on page 23](#)
- [“About the Workspace Server Options Tab” on page 243](#)

- “Integrated Windows Authentication” on page 174
- “Authentication to the Metadata Server” on page 144
- “Summary of Methods for LDAP Integration” on page 190

Integrated Windows Authentication

Table 11.5 *Integrated Windows Authentication (IWA)*

Summary	Participating SAS servers accept users who have successfully authenticated to their Windows desktop.
Scope	For Windows desktop clients and SAS servers on Windows and UNIX. ■ Primarily used for connections to the Metadata Server and the standard workspace server. ■ Also supported for direct connections to an OLAP server (for example, from a data provider).
Benefits	■ Bypasses the initial logon prompt. ■ Accommodates logon mechanisms that are not password-based (such as smart cards or biometrics). ■ No user credentials are transmitted. ■ Users do not need the Log on as a batch job privilege. See “Windows Privileges” on page 23.
Limits	■ All participating clients and servers must authenticate against the same Windows domain (or against domains that trust one another). ■ If you use IWA for the Metadata Server, there are no cached credentials from an initial logon. For this reason, it is a good idea to configure IWA for the standard workspace server also, if possible (IWA is not supported on z/OS). ■ Web applications cannot participate in this implementation of IWA. However, if you configure web authentication, and your web environment offers IWA, then your web applications can use IWA. To configure IWA for the SAS Web Application Server, see “Configure SAS Web Application Server” in <i>SAS Intelligence Platform: Middle-Tier Administration Guide</i>. ■ Desktop clients that run on UNIX (for example, SAS Management Console on UNIX) cannot participate in IWA. ■ If your SAS servers use DNS aliases, you must manually register those aliases in order to support IWA connections based on Kerberos. See “Registering SPNs” on page 221. ■ Additional limits for servers on Windows: □ If you use IWA for a Windows workspace server that accesses Windows network resources (for example, remote data access), the Kerberos protocol must be used and

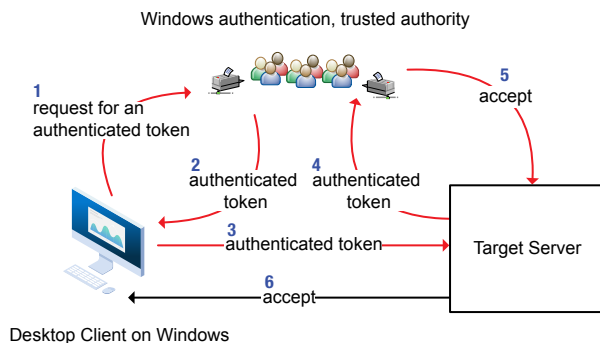
the object spawner account must have the **trusted for delegation** privilege. See [“Windows Privileges” on page 23](#).

- Additional limits for servers on UNIX:
 - In order to use IWA on UNIX platforms:
 - For SAS 9.4 and [SAS 9.4M1](#) on all platforms, you must purchase, install, and configure an additional third-party product such as One Identity Authentication Services (which was formerly Quest Authentication Services) 4.0 or later.
 - Starting with [SAS 9.4M2](#) on Linux platforms, you must ensure that a shared library that implements the GSSAPI with Kerberos 5 extensions is installed and configured to allow authentication against your Active Directory domain or Kerberos realm. One Identity Authentication Services fulfills this requirement, as do the krb5 packages that are provided in supported operating system distributions and in various third-party solutions.
 - Starting with [SAS 9.4M2](#) on all other UNIX platforms, you must purchase, install, and configure an additional third-party product (One Identity Authentication Services 4.0.1.23 or later).
 - When you use IWA on UNIX, only Kerberos connections are supported. (There is no support for NTLM on UNIX.)
 - If you use IWA for a UNIX workspace server that makes outbound Kerberos requests, the service principal account in Active Directory must have the **trusted for delegation to all services** privilege. See [“Windows Privileges” on page 23](#).

Use Optional.

The following figure is an abstraction of how this mechanism works.

Figure 11.3 *Integrated Windows Authentication*



The numbers in the figure correspond to these actions:

- 1 The client asks Windows for a token that represents the user who is currently logged on to the client computer. This step is initiated when a user launches a desktop client or makes a request for a workspace server.

Note: The token represents the client who is running the SAS application. If you use the Windows `runas` command to launch a SAS application, the token represents the host account that you chose to use.

- 2 Windows provides the token to the client.
- 3 The client sends the Windows token to the target server. Notice that only the token is sent; the user's password is not available to the target server.
- 4 The target server sends the token back to Windows for verification.
- 5 Windows tells the target server that the token is valid.
- 6 The target server accepts the connection from the client. By default, the user's authenticated ID is returned to the target server as follows:
 - If the target server is on Windows, the authenticated user ID is returned in domain qualified format (for example, WIN\joe).
 - If the target server is on UNIX, the authentication user ID is returned in short format; the user ID is not qualified (for example, joe).

Note: The format in which the authenticated user ID is returned to the target server must match the format in which that user ID is stored in the SAS metadata. See [“User Logons” on page 220](#).

See Also

- [“How SAS Identity Is Determined” on page 144](#)
- [“How to Configure Integrated Windows Authentication” on page 217](#)

Multifactor Authentication (MFA)

Starting with [SAS 9.4M9](#), multifactor authentication (MFA), which is provided by SAS Logon Manager, is supported for the SAS 9.4 Intelligence Platform middle tier. MFA provides an additional layer of security that supplements existing SAS 9.4 authentication mechanisms.

SAS Logon Manager enables users to log on to SAS web applications. The list of SAS web applications and their server assignments is available at [SAS Web Application Server Assignments](#). These components can now be protected with MFA as an additional layer of authentication. You can use SAS Deployment Wizard to configure MFA for SAS Logon Manager, or you can enable MFA manually.

SAS supports either a third-party time-based one-time password (TOTP) authenticator application, such as Microsoft Authenticator or Google Authenticator, or the built-in SAS Authenticator component as the authentication provider.

In addition to enterprise TOTP applications running on personal devices, SAS supports open-source TOTP plug-ins such as the Google Chrome plug-in. If you use a third-party TOTP authenticator, users can gain an access token after receiving a random number and specifying it at logon.

SAS Authenticator is based on Apereo Central Authentication Service (CAS) Simple. If you select SAS Authenticator, users can gain access by using email or text messages. For user messaging and token management, SAS Authenticator supports multiple third-party SMS text message providers.

MFA is not enabled for the SAS 9.4 middle tier by default. For more information about MFA in SAS 9.4M9 and how to enable it, see [“Configure Multifactor Authentication \(MFA\) for SAS Logon Manager” in SAS Intelligence Platform: Installation and Configuration Guide](#).

Pluggable Authentication Modules (PAM)

Table 11.6 PAM (Pluggable Authentication Modules)

Summary	A supporting feature that extends UNIX host authentication to recognize an additional provider such as Active Directory. When a SAS server asks its UNIX host to validate a user's credentials, the host sends the user's ID and password to the configured additional provider for verification. PAM extends the host's authentication process to recognize an additional provider; PAM does not modify SAS Metadata Server behavior.
Scope	Affects all SAS servers that run on the UNIX host and rely on the host operating system to authenticate users. Typically, SAS Metadata Server and the workspace server use host authentication.
Benefits	Can be used to enable users to use their Windows accounts to authenticate to a SAS Metadata Server or workspace server that runs on UNIX.
Limits	Not an alternative to storing user IDs in the metadata; that requirement applies to all configurations.
Use	Optional.

This mechanism is useful if both the SAS Metadata Server and the workspace server are on UNIX and you want users to use Windows accounts to access these servers.

This mechanism can also be useful if one of these servers is on Windows, the other is on UNIX, and you want to avoid credential prompts for the workspace server.

However, if you use PAM to resolve a mixed provider situation, users who access the workspace server must have two logins. One login should include the user's ID in its qualified form. The other login should include the same ID in short (unqualified) form. Both logins should be in the DefaultAuth authentication domain. Neither login should include a password. For example, a user's logins might resemble the following example:

```
DefaultAuth | WIN\joe | (no password)
DefaultAuth | joe     | (no password)
```

For configuration instructions, see *Configuration Guide for SAS Foundation for UNIX Environments* at <http://support.sas.com/documentation/installcenter>.

See Also

- “Summary of Methods for LDAP Integration” on page 190
- “Direct LDAP Authentication” on page 171
- “Mixed Providers” on page 148

SAS Internal Authentication

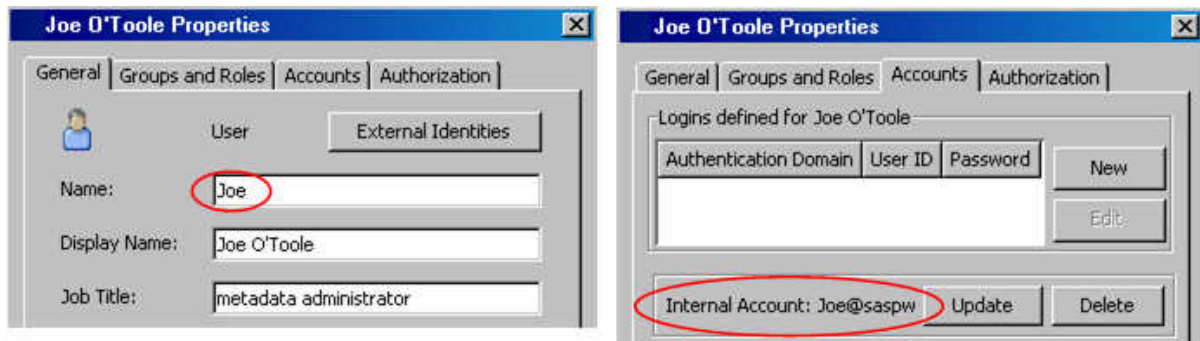
Table 11.7 Internal Authentication

Summary	Participating SAS servers validate incoming user IDs that have a special suffix (@saspw) against a list of accounts that exist only in the metadata.
Scope	■ Primarily used for connections to the Metadata Server. ■ Also supported for direct connections to the OLAP server. This is not a common use. In the SQL procedure, a connection to the OLAP server can be made using an internal account.
Benefits	■ Minimizes the need to create external accounts for service identities. ■ Facilitates intermittent use of an administrative role by enabling a user to have a second metadata identity without having a second external account. ■ Provides independence from the rest of your computing environment. For example, internal accounts do not have to follow your general password change requirements, cannot be used to access resources beyond the SAS metadata, and are not affected by changes in machine names or in your authentication configurations.
Limits	■ Internal accounts are not intended for regular users (they are intended for only metadata administrators and some service identities).

- Someone who has only an internal account cannot seamlessly access the workspace server.
- An internal account cannot participate in Integrated Windows authentication or web authentication.
- You cannot use an internal account to delete, unregister, add, or initialize a foundation repository.
- You cannot run a server under an internal account. For example, the SAS General Server User (sassrv) must be a host account.

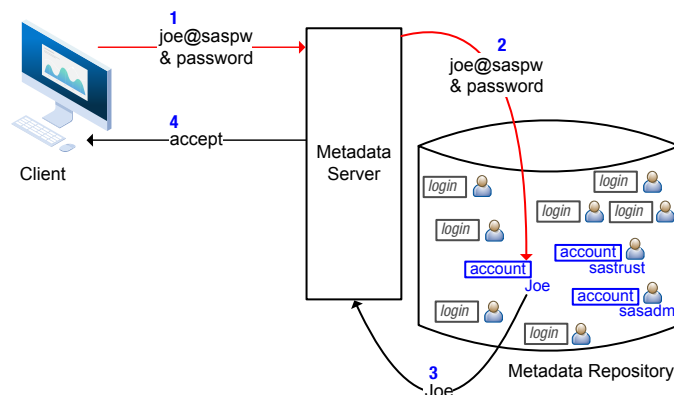
Use Always available.

Internal accounts exist only in the metadata and can be created in SAS Management Console. The following displays depict the **General** tab and **Accounts** tab for an administrator named Joe. Notice that Joe's internal user ID is Joe@saspw. The user ID for an internal account is always in the format **name@saspw**. The name comes from the **Name** field on the user's **General** tab.



The following figure depicts the internal authentication process:

Figure 11.4 Internal Authentication



The numbers in the preceding figure correspond to these actions:

- 1 At a logon prompt, Joe enters his internal credentials. The client sends those credentials to the Metadata Server for verification.

- 2 The Metadata Server recognizes that the ID is for an internal account (because the ID has the @saspw suffix), so the Metadata Server checks the credentials against its list of internal accounts.
- 3 After validating the ID and password, the Metadata Server accepts the client connection. The connection is under the identity of the SAS user who owns the account.

Note: Joe can have logins in addition to his internal account, but he does not need a login to establish his SAS identity.

Here are some tips for working with internal accounts:

- There are two distinct expiration settings. Do not confuse the account expiration date with the password expiration period.
- If repeated attempts to log on with an internal account fail, make sure you are including the @saspw suffix in the user ID. Another cause of failure is the account is locked.
- If you have both an internal account and an external account, use a dedicated connection profile for your internal account. In that profile, leave the **Authentication domain** field blank. This ensures optimal credential reuse.
- If you do not have user administration capabilities, you cannot see internal accounts, unless you are viewing your own definition and you happen to have an internal account.

Because internal accounts exist only in the metadata, they do not automatically follow the policies of other authentication providers. Here are the initial policies for internal accounts:

- Accounts do not expire and are not suspended due to inactivity.
- Passwords must be at least six characters, do not have to include numbers or mixed case, and do not expire.
- The five most recent passwords cannot be reused.
- After three failed attempts to log on, an account is locked for one hour. An administrator can unlock the account by clicking the **Accounts** tab in the user's definition in SAS Management Console.
- A forced password change occurs on first use and after a password is reset. This policy applies only to accounts with passwords that periodically expire. By initial policy, passwords do not expire, so forced password changes do not occur.

You can set all of these policies globally (at the server level). You can also selectively override many of these policies on a per-account basis.

See Also

- [“Authentication to the Metadata Server” on page 144](#)
- [“How to Change Internal Account Policies” on page 239](#)

■ “Unlock an Internal Account” on page 47

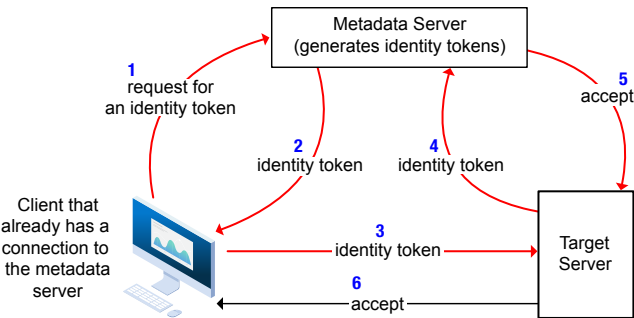
SAS Token Authentication

Table 11.8 SAS Token Authentication

Summary	SAS Metadata Server generates and validates a single-use identity token for each authentication event. As a result, participating SAS servers accept users who are connected to the Metadata Server.
Scope	- Primarily used for metadata-aware connections to the stored process server, the server-side pooled workspace server, the OLAP server, the content server, and (in a specialized configuration) the standard workspace server. - Also used by launched servers to connect back to the Metadata Server (for example, from the workspace server to the Metadata Server for library pre-assignment).
Benefits	- Preserves client identity for metadata layer access control and auditing purposes. - No individual external accounts are required, no user passwords are stored in the metadata, and no reusable credentials are transmitted.
Limits	- On the workspace server, reduces granularity of host access. - Supported only for metadata-aware connections (in which the client learns about the target server by reading the server’s metadata definition).
Use	Optional for the workspace server, otherwise mandatory within its scope.

The following figure is an abstraction of how this mechanism works.

Figure 11.5 SAS Token Authentication



The numbers in the figure correspond to these actions:

- 1 Over the user’s existing connection to the Metadata Server, the client requests an identity token for the target server. This step is initiated by a user request

that requires access to the target server (for example, by a request in SAS Enterprise Guide for a cube that is associated with the OLAP server).

- 2 SAS Metadata Server generates the token and sends it to the client.
- 3 The client provides the token to the target server.
- 4 The target server sends the token to the Metadata Server for validation.
- 5 SAS Metadata Server validates the token and returns an acceptance message and a representation of the user to the target server.
- 6 The target server accepts the connection.

See Also

- [“How to Configure SAS Token Authentication” on page 205](#)
- [“Mediated Access” on page 112](#)

Trusted Peer Connections

Table 11.9 *Trusted Peer*

Summary	The Metadata Server accepts peer SAS sessions and servers that connect using a proprietary protocol (trusting that those connecting identities have already been properly authenticated).
Scope	From any SAS session or SAS IOM server process to the Metadata Server. The scope is configurable.
Benefits	■ Enables SAS/CONNECT servers to access the Metadata Server. ■ Can enable batch processes to connect to the Metadata Server. ■ In a clustered Metadata Server environment, can support communication among nodes within the cluster. For information about clustered metadata servers, see SAS Intelligence Platform: System Administration Guide
Limits	■ On Windows, it is safer to use Integrated Windows authentication (IWA). By default, a clustered Metadata Server environment on Windows uses IWA for connections among nodes. ■ For greater security, minimize availability of this feature by modifying the <code>trustedPeers.xml</code> file, as documented in this topic.

-
- Use
- For a clustered Metadata Server on UNIX or z/OS, trusted peer connections by the installer ID and from the cluster machines must be allowed.
 - If you use your operating system scheduler to run metadata backup jobs, trusted peer connections from the host account that runs those jobs must be allowed.
-

If the Metadata Server's start command includes the `TRUSTSASPEER=` option, the referenced `trustedPeers.xml` file specifies which user IDs and machines are eligible to connect to the Metadata Server using the trusted peer protocol.

By default, all user IDs and machines are eligible. The initial contents of the `trustedPeers.xml` file are as follows:

```
<TrustedSASPeers>

    <TrustedSASPeerClients>
        <client name="SAS" />
    </TrustedSASPeerClients>

    <TrustedSASPeerUsers>
        <user name="*" />
    </TrustedSASPeerUsers>

    <TrustedSASPeerMachines>
        <machine ip="*" />
    </TrustedSASPeerMachines>

</TrustedSASPeers>
```

For greater security, we recommend that you target this mechanism so that the Metadata Server does not accept every connection that uses the proprietary protocol. You can use either or both of these constraints:

- accept only specified user IDs
- accept only connections that originate from specified machines

You can define constraints in `trustedPeers.xml` as follows:

TrustedSASPeerClients

lists eligible client types. **SAS** and **java** are the valid values. It is recommended that you reject connections from Java clients.

Typically, there is only one entry between the **TrustedSASPeerClients** tags:

```
<client name="SAS"/>
```

TrustedSASPeerUsers

lists eligible user IDs. To represent all users, use an asterisk (*). To represent all users in a Windows domain, use the format `*@domain`. For Windows accounts, provide domain-qualified (or machine-qualified) IDs. For example, you might insert these three entries between the **TrustedSASPeerUsers** tags:

```
<user name="*@winXP" />
<user name="tara" />
<user name="batchjobID" />
```

TrustedSASPeerMachines

lists eligible points of origin. Identify machines by IP address. You can use asterisks (*) as wildcards. For example, you might insert these three entries between the **TrustedSASPeerMachines** tags:

```
<machine ip="1.2.3.4"/>
<machine ip="A:B:C:D:E:F:1.2.3.4"/>
<machine ip="*.*.8.9"/>
```

If you modify the trustedPeers.xml file, keep the following points in mind:

- Only connections that meet all specified criteria are accepted. If any of the sections are empty, no trusted peer connections are allowed.
- For a clustered Metadata Server on UNIX or z/OS, you must preserve support for trusted peer connections from the cluster nodes (under the install account) as follows:
 - In the TrustedSASPeerUsers section of the trustedPeers.xml file, include the installer account (or accounts).
 - In the TrustedSASPeerMachines section of the trustedPeers.xml file, include the IP address of each cluster machine.

Note: The trustedPeers.xml file is in your equivalent of **SAS/Config/Lev1/SASMeta/MetadataServer/**.

Trusted User Connections

Table 11.10 Trusted User

Summary	SAS Metadata Server allows a privileged account to act on behalf of other users (trusting that those users have already been properly authenticated).
Scope	To the Metadata Server from the object spawner, the OLAP server, SAS web applications (if web authentication is used), and batch report processes.
Benefits	Supports the optional web authentication configuration. Enables the OLAP server and the object spawner to impersonate each requesting user on connections to the Metadata Server. Enables batch reporting processes to connect to the Metadata Server under their identities.
Limits	It is important to protect this privileged account.
Use	Required.

In a new deployment, the `trustedUsers.txt` file lists one account that serves as the trusted user for the entire deployment.

CAUTION

Do not add regular users to the `trustedUsers.txt` file. The trusted user is a privileged service identity that can act on behalf of all other users.

Note: The `trustedUsers.txt` file is in your equivalent of `SAS/Config/Lev1/SASMeta/MetadataServer/`.

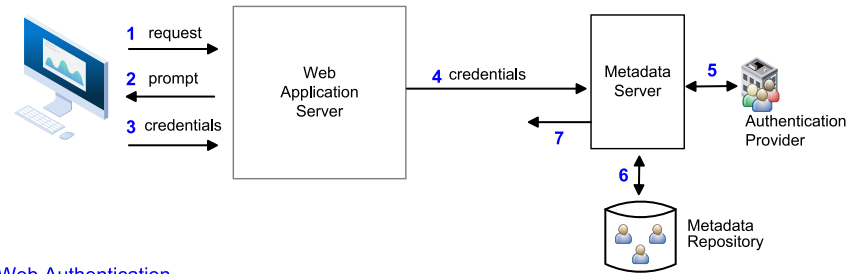
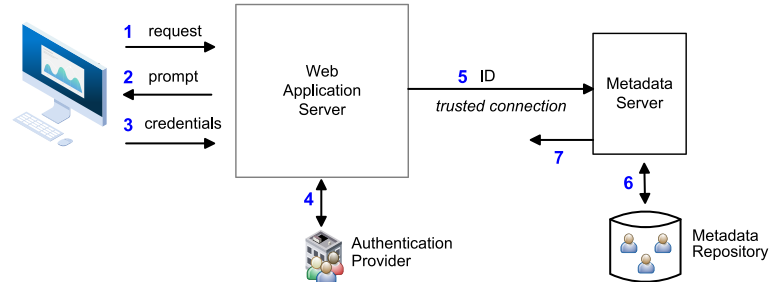
Web Authentication

Table 11.11 Web Authentication

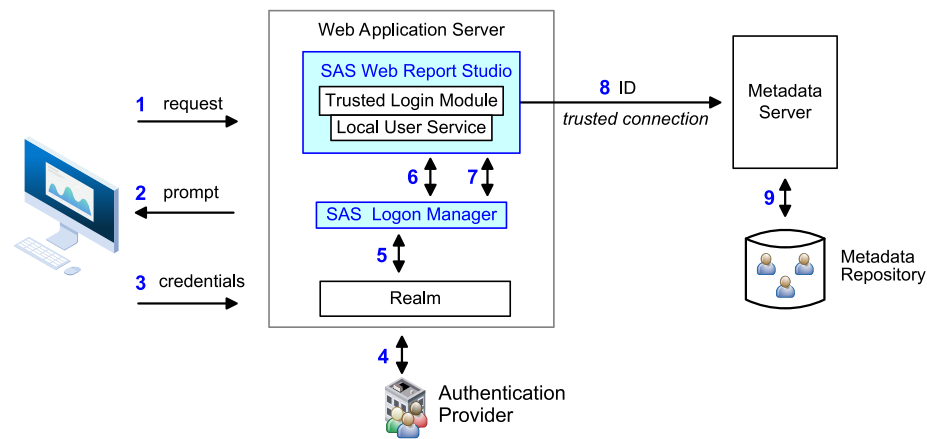
Summary	The Metadata Server accepts users who are authenticated at the web perimeter.
Scope	From SAS web applications to the Metadata Server.
Benefits	■ Facilitates single sign-on from and across the web realm. ■ Enables SAS web applications to use whatever authentication scheme you have set up in your web environment. ■ Can reduce the number of user accounts that you have to create in the Metadata Server's authentication provider, because users who use only web applications no longer need accounts with the Metadata Server's provider.
Limits	■ When you use web authentication to access the Metadata Server, no cached credentials from an initial logon are available. This prevents authentication from web applications to the standard workspace server through reuse of credentials. ■ Prevents users from logging on to a SAS web application with a SAS internal account. ■ Not compatible with anonymous access. See “About PUBLIC Access and Anonymous Access” on page 164.
Use	Optional.

The following figure depicts the high-level choice in authentication method for SAS web applications:

- SAS authentication (any form of authentication in which the Metadata Server is responsible for requesting verification)
- web authentication (any form of authentication in which verification occurs in the web realm and the Metadata Server trusts that verification)

Figure 11.6 Examples of SAS Authentication and Web Authentication**SAS Authentication****Web Authentication**

The preceding figures are simplified in order to highlight the differences between the two configurations. The following figure includes additional detail about web authentication.

Figure 11.7 Web Authentication: A Closer Look

In the preceding figure, a user who is not already authenticated at the web perimeter makes a request to access SAS Web Report Studio. The numbers in the figure correspond to the following activities:

- 1 In a web browser, the user makes a request for the target application (in this example, the target application is SAS Web Report Studio).
- 2 The web container prompts the user for credentials.

Note: If the user has already authenticated at the web perimeter, this step is omitted.

- 3 The user supplies credentials to the web container.

Note: If the user has already authenticated at the web perimeter, this step is omitted.

- 4 The web container's realm directs the container to authenticate the user against a designated third-party authentication provider.
- 5 SAS Logon Manager retrieves the authenticated user ID from the web container. The SAS Logon Manager stores the user ID for later use.

- 6 SAS Logon Manager provides a ticket for the target application and redirects the user to that application.
- 7 The target application connects back to SAS Logon Manager (over HTTP) to validate the supplied ticket. SAS Logon Manager supplies the requesting user's authenticated user ID as part of the validation response.
- 8 The target application uses the Trusted Login Module (from the SASTRUSTED JAAS context) to generate a JAAS subject from the authenticated user ID. The generated JAAS subject is given to the local user service instance (inside the target application), which creates a user context and connects to the Metadata Server.
- 9 The Metadata Server looks up the user's ID in the metadata repository in order to determine the user's SAS identity. As usual, this step does not involve password validation and is not affected by authentication domain assignments. Only the user's ID is being matched (the authentication domain assignment in a login affects only credential reuse).

Note: Some components (such as SAS Comment Manager) do not require a connection to the Metadata Server. Such components do not have a user context or a persistent connection to the Metadata Server.

See Also

[“How to Configure Web Authentication” on page 207](#)

Microsoft Windows Defender Credential Guard

Overview

SAS supports constrained delegation, which is a requirement for Microsoft Windows Defender Credential Guard (Credential Guard).

Credential Guard isolates logon information for users from the rest of the operating system. Credential Guard uses virtualization to store credentials in protected containers that are separated from the operating system. Therefore, the information that Credential Guard protects is safe even when malware or some other malicious attacks penetrate your organization's network.

Credential Guard is not enabled by default. It can be enabled using group policies, the Windows registry, or the Windows Defender Device Guard. When enabled, Credential Guard does not allow Kerberos unconstrained delegation or Data Encryption Standard (DES) encryption for credentials used to sign in or for prompted or saved credentials. For information about enabling Credential Guard, see [Manage Windows Defender Credential Guard](#).

Issues That Might Occur

After Credential Guard is enabled, some older authentication protocols will not work for credentials used to sign in to your network. These protocols include NT LAN Manager (NTLM) version 1, Microsoft Challenge-Handshake Authentication Protocol (MS-CHAP) version 2, digest authentication, and Credential Security Support Provider protocol (CredSSP). SAS does not currently support any of these protocols. In addition, single sign-on will not work with these protocols when Credential Guard is enabled.

Outbound access to data sources that depend on Kerberos protocol connections will also be broken. Data sources that use Kerberos protocol connections include, but are not limited to, the following:

- Hadoop
- Network File System (NFS)
- SQL Server

In SAS 9.4 deployments, problems might occur with selected SAS products and solutions when the following conditions exist:

- Credential Guard is enabled, and
- Integrated Windows Authentication (IWA) is implemented, and
- Windows 10 is the client-tier host platform, and
- Windows Server 2012, Windows Server 2016, Windows Server 2019, or UNIX is the host platform.

SAS Support for Windows Defender Credential Guard

If a SAS product does not currently support Windows Defender Credential Guard, SAS recommends that you perform one of these actions:

- Disable Credential Guard, or
- Disable IWA and require users to authenticate at additional points in their sessions.

For a list of SAS products that support Credential Guard, see [SAS Installation Note 63143: Windows Defender Credential Guard](#).

Note: Constrained delegation is currently supported only on Linux and Windows host systems, and resource-based constrained delegation is supported only on Windows. Traditional constrained delegation does not support a cross-realm trust. Cross-realm trust is defined as when the front-end service (SAS) is in a different domain than the back-end service (for example, Teradata, SQL Server, or a file server). For a cross-realm trust, resource-based constrained delegation is required. However, resource-based constrained delegation is available only on Windows operating systems. For more information, see [“How to Configure Kerberos Constrained Delegation” on page 229](#).

Summary of Methods for LDAP Integration

SAS supports the following methods for integration with LDAP:

host use of LDAP

The SAS server's host uses an LDAP provider as a back-end authentication provider. From the perspective of the SAS server, this is host authentication. For example:

- Active Directory is the standard back-end authentication provider on Windows.
- Some UNIX hosts recognize LDAP accounts (or can be configured to do so). See [“Pluggable Authentication Modules \(PAM\)” on page 177](#).

sasauth use of LDAP (UNIX only)

This method is used by default and provides a direct connection from sasauth (the UNIX host authentication module) to an LDAP database for authentication. This method provides an authenticated UNIX host identity for each user. For configuration instructions, see [Configuration Guide for SAS Foundation for UNIX Environments](#).

sas-services daemon use of LDAP (UNIX only)

This authentication utility, which implements UNIX authentication, is included with [SAS 9.4M6](#) and later on Linux. This service is a multi-threaded alternative for sasauth that improves the performance of SAS 9.4 environments under conditions of heavy usage. For more information, see [Configuration Guide for SAS Foundation for UNIX Environments](#).

SAS Metadata Server use of LDAP

SAS Metadata Server validates some users against an LDAP provider, such as Microsoft Active Directory. This method enables the Metadata Server to identify accounts that are not known to its host. It does not provide SAS with an

authenticated UNIX host identity for each user. See [“Direct LDAP Authentication” on page 171](#).

LDAP integration support is for authentication purposes only, not for authorization.

Summary for Single Sign-On

There is no individual mechanism that provides end-to-end single sign-on (SSO). The following authentication processes are transparent:

- Integrated Windows authentication (IWA) is based on previous authentication to your desktop and provides silent launch for SAS desktop applications (and, sometimes, silent access to the workspace server).
- Web authentication is based on previous authentication to your web realm and provides silent launch for SAS web applications.
- SAS token authentication requires a connection to the Metadata Server and provides silent access to most SAS servers.
- Credential reuse and retrieval requires a connection to the Metadata Server and can provide silent access to any server.

Some configurations can interfere with SSO to back-end servers. This table summarizes the considerations:

Table 11.12 SSO Considerations for Selected Authentication Mechanisms

Feature	Front-end SSO	Back-end SSO	Notes
Internal authentication	⚠	⚠	An internal account cannot participate in IWA or web authentication.
SAS token authentication		✓	Facilitates SSO to most SAS servers.
IWA	✓	⚠	Facilitates silent launch of desktop applications. If not fully configured, prevents SSO to a standard workspace server. ¹
Web authentication	✓	⚠	Facilitates silent launch of web applications. Prevents SSO to a standard workspace server. ¹
Direct LDAP authentication	⚠	⚠	Not compatible with silent launch. Prevents SSO to a standard workspace server. ¹
PAM		✓	Can help unify authentication.

Feature	Front-end SSO	Back-end SSO	Notes
Credential Management		✓	Facilitates SSO to third-party servers and (in some configurations) workspace servers.

¹ Unless the server is configured for SAS token authentication or accessed using stored credentials.

Summary by Server Type

This table provides a high-level review of support for different authentication mechanisms. In the table, the following symbols indicate the extent to which each server can be accessed using each feature:

- ● Supported
- ◐ Supported for only direct connections (not metadata aware connections)
- ◑ Intended for only administrators and some service identities
- ◉ Used only for certain server-to-server communications
- Not supported

Table 11.13 Summary: How Servers Can Be Accessed

Mechanism	Server Type				
	Metadata	Workspace	OLAP	Stored Process or Pooled Workspace	Client-Pooled Workspace ¹
Host authentication (credentials)	●	●	◐		◉
Integrated Windows authentication	●	●	◐		
Web authentication	●				
Direct LDAP authentication	●		◐		
Internal authentication	◑		◑	◑	
SAS token authentication	◉	●	●	●	

Mechanism	Server Type				
	Metadata	Workspace	OLAP	Stored Process or Pooled Workspace	Client-Pooled Workspace ¹
Trusted user	●				
Trusted peer	●				

¹ For the client-pooled workspace server, user access depends on membership in a puddle group.

Authentication Tasks

<i>How to Manually Enable Oracle Wallet</i>	196
Overview	196
Prerequisites	197
1. Stop SAS Servers	197
2. Verify CA Certificates in Trusted CA Bundle and Oracle Truststores	198
3. Verify That Oracle Admin Files Are Configured for TLS	198
4. Update SAS Web Application Server to Use Secure Oracle Wallet	199
5. Update or Verify Database Port in SAS Management Console	200
6. Update or Verify SAS Solution Server Files	201
7. Restart Servers	201
Prerequisite Steps Before an In-place Upgrade	202
<i>How to Facilitate Authentication</i>	203
Identify or Create User Accounts	203
Coordinate the Workspace Server	204
<i>How to Configure SAS Token Authentication</i>	205
See Also	206
<i>How to Configure Web Authentication</i>	207
Overview of Configuring Web Authentication	207
Step-By-Step Instructions for Web Authentication	207
Logons for Users Who Participate in Web Authentication	207
See Also	208
<i>How to Configure Direct LDAP Authentication</i>	209
Before You Begin	209
Active Directory	209
Other LDAP Providers	211
Advanced Configurations	213
About AUTHPD and PRIMPD	216
About Setting Environment Variables	216
<i>How to Configure Integrated Windows Authentication</i>	217
Overview	217
Instructions	217
User Logons	220
Registering SPNs	221
Forcing Kerberos	224
Supported Protocols	225
IWA to an OLAP Server	226
IWA to a Clustered Metadata Server	227

IWA to Servers That Use DNS Aliases	228
About IWA from SAS Web Applications	228
How to Configure Kerberos Constrained Delegation	229
Considerations for Windows: Identity and Ownership	229
Considerations for Windows: Enabling Paths in UNC Format	229
Configure Kerberos	230
Kerberos on UNIX	232
Kerberos on Windows	233
About IBM Z Multi-Factor Authentication (MFA)	234
How to Store Passwords for the Workspace Server	235
See Also	237
How to Store Passwords for a Third-Party Server	237
See Also	238
How to Change Internal Account Policies	239
Server-Level Policies	239
Per-Account Policies	241
See Also	242
About the Workspace Server Options Tab	243
See Also	244

How to Manually Enable Oracle Wallet

Overview

Oracle Wallet is a secure software container that stores authentication and encryption credentials. It is designed to help applications authenticate and communicate securely with databases, ensuring that sensitive information, such as passwords, certificates, and private keys, is safely managed.

Starting with a December, 2025 hot fix for [SAS 9.4M9](#), you can enable SAS to use Oracle Wallet for your secure connection to an Oracle database through SAS Deployment Wizard. For more information, see [“Enable Secure Oracle Wallet Connection” in SAS Intelligence Platform: Installation and Configuration Guide](#).

Starting with a March, 2026 hot fix for [SAS 9.4M9](#), Oracle Wallet can be used to securely store certificates required to enable TLS communication between SAS Cost and Profitability Management and Oracle databases. This can be done automatically through SAS Deployment Wizard. For more information about setting up Oracle Wallet and Oracle for SAS Cost and Profitability Management, see [Appendix J - TLS 1.x Configuration of SAS Cost and Profitability Management Installation, Migration, and Configuration Guide](#).

Note: Enabling Oracle Wallet for SAS components that use Oracle to store data does not interfere with using Oracle Wallet for your Oracle data servers that store

user data. For information about this functionality, see [SAS/ACCESS Interface to Oracle](#) and [Connect to Oracle by Using Oracle Wallet for Programmers](#).

If you did not enable Oracle Wallet during the initial configuration in SAS Deployment Wizard, you can manually enable Oracle Wallet after the deployment has completed by taking the steps described in this section.

Note: If you manually set up a secure SAS connection to Oracle Wallet in your SAS 9.4 environment and you are performing an in-place upgrade to SAS 9.4M9, some additional steps are required. For more information, see [“Prerequisite Steps Before an In-place Upgrade” on page 202](#).

Prerequisites

The following components must be installed and configured before you enable Oracle Wallet:

- You must have an existing Oracle database instance. Oracle Wallet is configured for use with this Oracle instance.

Copy the `oraclepki.jar`, `osdt_cert.jar`, and `osdt_core.jar` files from your `orainstall/jlib` directory to the `orainstall/jdbc/lib/` directory.

Note: SAS supports the use of one Oracle Wallet. One wallet can be used for multiple databases.

- Oracle Wallet must be set up with TLS. For more information, see the following websites:
 - [Setting Up Oracle Wallet](#)
 - [Create a Wallet or Certificates for a TLS Connection](#)

TIP As an example, you can use the instructions for setting up an Oracle database and Oracle Wallet that are provided in Appendix J - TLS 1.x Configuration of the [SAS Cost and Profitability Management Installation, Migration, and Configuration Guide](#).

Oracle Wallet is installed and configured, but it is not enabled for the Oracle instance that SAS products and solutions use.

1. Stop SAS Servers

It is easy to stop your SAS Servers. However, you might need to start SAS Metadata Server in a later step. Stopping your middle-tier servers is sufficient. For

more information, see [“Operating Your Servers” in SAS Intelligence Platform: System Administration Guide](#).

2. Verify CA Certificates in Trusted CA Bundle and Oracle Truststores

SAS uses secure connections to Oracle Wallet. Use SAS Deployment Manager to add your Oracle database CA certificates to the trusted certificate authority bundle. For more information, see [“Manage Certificates in the Trusted CA Bundle Using the SAS Deployment Manager” in Encryption in SAS](#).

3. Verify That Oracle Admin Files Are Configured for TLS

- 1 Verify that the following Admin files contain TLS configuration:
 - listener.ora
 - sqlnet.ora
 - tnsnames.ora
- 2 Verify that these Admin files were copied from your Oracle installation directory to your Oracle Wallet directory. (These files are also located in the `$ORACLE_HOME/network/admin` directory.)

The following code fragments are examples of TLS-configured values:

■ listener.ora admin file

The following highlighted code fragment is set in `listener.ora`:

```
SSL_CLIENT_AUTHENTICATION = FALSE
SECURE_PROTOCOL_LISTENER=(IPC)
WALLET_LOCATION =
(SOURCE =
(METHOD = FILE)
(METHOD_DATA =
(DIRECTORY = <wallet_location>)
)
)

LISTENER =
(DESCRIPTION_LIST =
(DESCRIPTION =
(ADDRESS = (PROTOCOL = IPC) (KEY = EXTPROC1))
(ADDRESS = (PROTOCOL = TCPS) (HOST = <Hostname>) (PORT = 2484))
)
)
```

Here are the values for the code fragment:

- `<wallet_location>` is the directory path of Oracle Wallet.
- `<hostname>` is the machine name where Oracle is deployed.

■ **sqlnet.ora** admin file

(The `sqlnet.ora` file is the profile configuration file. It resides on the client machines and the database server.)

The following highlighted code fragment is set in `sqlnet.ora`:

```
NAMES.DIRECTORY_PATH=(TNSNAMES,EZCONNECT)
ENCRYPTION_WALLET_LOCATION =
  (SOURCE =
    (METHOD = FILE)
    (METHOD_DATA =
      (DIRECTORY = <wallet_location>
    )
  )

SQLNET.WALLET_OVERRIDE = FALSE
SQLNET.CRYPTO_CHECKSUM_TYPES_SERVER = (MD5, SHA1, SHA256, SHA384,
SHA512)
SQLNET.AUTHENTICATION_SERVICES = (BEQ, TCPS, NTS)
SSL_VERSION = 0
SSL_CLIENT_AUTHENTICATION = FALSE
SQLNET.ENCRYPTION_TYPES_SERVER = (RC4_128, 3DES112, RC4_56, DES,
RC4_40, DES40, AES256, RC4_256, AES192, 3DES168, AES128)

WALLET_LOCATION =
  (SOURCE =
    (METHOD = FILE)
    (METHOD_DATA =
      (DIRECTORY = <wallet_location>
    )
  )

SSL_CIPHER_SUITES=(SSL_RSA_WITH_AES_256_CBC_SHA,
SSL_RSA_WITH_3DES_EDE_CBC_SHA)
```

In the preceding code fragment, `<wallet_location>` is the directory path of Oracle Wallet.

■ **tnsnames.ora**

Verify that the port is set to the secure port 2484 and that the protocol is set to TCPS.

4. Update SAS Web Application Server to Use Secure Oracle Wallet

- 1 Add the Oracle libraries to SAS Web Application Server. Copy the following JAR files to this directory:

```
/SAS_config_dir/Levn/Web/WebAppServer/SASServern.n/lib
```

- oraclepki.jar
- osdt_cert.jar
- osdt_core.jar

- 2 Modify JDBC connections in SAS Web Application Server SASServern_n instances to reference the secure Oracle database. Update the **server.xml** file.

- a Navigate to the `/SAS_config_dir/Levn/Web/WebAppServer/SASServern.n/conf` directory.
- b Open and edit `server.xml`.

Within the `<GlobalNamingResource>` section, modify the JDBC connection string for servers that use Oracle. Here is an example for the SharedServices database:

```
xaProperties.url="jdbc:postgresql://host-name:9432/SharedServices"
```

Change the code for SharedServices to the following:

```
jdbc:oracle:thin:@database_name?TNS_ADMIN=oracle_wallet_path
```

In the preceding string, *database_name* is the name of the Oracle database that is used for the SharedServices database, and *oracle_wallet_path* is the directory path to your Oracle Wallet configuration.

5. Update or Verify Database Port in SAS Management Console

Complete this step if a SAS product or solution uses Oracle Wallet only. Update the database port from 1521 to 2484.

- 1 Log in to SAS Management Console with administrator privileges. An example is `sasadm@saspw`.
- 2 On the **Plug-ins** tab, expand **Application Management** → **Configuration Manager** → **SAS Application Infrastructure**. Right-click your solution and select **Properties**. Click the **Advanced** tab.

Here is an example that uses the 8.4 release of Cost and Profitability Management.

Note: If you configured Oracle Wallet through SAS Deployment Wizard for SAS Cost and Profitability Management, the database and port are listed.

- a Click **Add**.
- b Property Name: `data.abmserver.db.host.tls.enabled`
Property Value: `true`

- c Update the port to 2484 from 1521 in the following property:
data.abmserver.db.port

6. Update or Verify SAS Solution Server Files

Update the server files on your compute tier if a SAS product or solution uses Oracle Wallet only. The following steps use an example from SAS Cost and Profitability Management.

Note: If you configured Oracle Wallet through SAS Deployment Wizard for SAS Cost and Profitability Management, verify the modifications in the settings.xml file.

- 1 Edit the settings.xml file for the solution server. It is located in the *SAS-config-dir/Levn/SASApp/solution_server/config/* directory.

- 2 Update the <TLSEnabled> tag:

```
<!-- Specify if the Database is TLS Enabled. 1 if Yes 0 if No --><TLSEnabled>1</TLSEnabled>
```

- 3 Depending on your solution, you might need to update the server start-up files to ensure a secure TLS connection. Here is an example for Model Server, which is included in SAS Cost and Profitability Management.

Modify the following file: *SAS-config-dir/Levn/SASApp/ABMServer/ABMServer.sh*.

Add -SSLCALISTLOC to the CMD_OPTIONS: variable and use the correct chain certificate:

```
CMD_OPTIONS=" -logconfigloc $CONFIGDIR/logconfig.xml -SSLCALISTLOC path/to/PEM/file"
```

You can use the following command to create the PEM file by using the chain certificate that you used for SAS Web Server SSL:

```
openssl x509 -in <.crt file with path> -out .pem file with path -outform PEM
```

7. Restart Servers

- 1 Restart Oracle and Listener.
- 2 Start SAS Servers.

Prerequisite Steps Before an In-place Upgrade

If you manually enabled Oracle Wallet earlier than SAS 9.4M9 (December 2025) and you are performing an in-place upgrade to SAS 9.4M9 (December 2025) and later, complete the following steps before you begin the in-place upgrade:

- 1 **Stop SAS Servers.** For more information, see [“Operating Your Servers” in SAS Intelligence Platform: System Administration Guide](#).

- 2 **Revert changes to the Oracle admin files.**

- a Back up the following files: listener.ora, sqlnet.ora, and tnsnames.ora.

- b Stop the Oracle Listener using the Listener Control utility:

```
lsnrctl stop
```

- c Modify the listener.ora, sqlnet.ora, and tnsnames.ora files.

- Remove any reference to Oracle Wallet.
- Revert the protocol from TCPS to TCP.
- Revert the port from 2484 to 1521.

- d Reset your local listener variable:

```
Sqlplus / as sysdba
```

```
alter system set local_listener='(ADDRESS=(PROTOCOL=TCP)
(HOST=hostname)(PORT=1521))' scope=both;
```

In the preceding example, substitute the machine name for *host-name*.

```
alter system register;
```

```
Sqlplus / as sysdba
```

- e Start the listener and confirm that it has started.

- f Test the database connection.

- 3 **Begin upgrade.**

- 4 After the in-place upgrade is complete, click **Stop SAS Servers**.

- 5 **Add Oracle Wallet configuration back to Oracle admin files.**

- a Stop the Oracle Listener:

```
lsnrctl stop
```

- b In the listener.ora, sqlnet.ora, and tnsnames.ora files, make the following changes:

- Update any protocol from TCP to TCPS.
- Update the port from 1521 to 2484.
- Restore any reference to the wallet.

Note: Refer to your backup files for your specific configuration, which might vary.

- c Start the listener and confirm that it is listening on the secure port.
 - d Test the database connection.
- 6 **Update SAS files.** The upgrade process preserves all the manual changes to the `setenv` file but reverts the `jdbc URL` to a standard Oracle URL in the `server.xml` files.

- a For each data source connection that is using Oracle Wallet, navigate to the server instance where it runs. Navigate to the `/SAS_config_dir/Levn/Web/WebAppServer/SASServern.n/conf` directory.
- b Open the `server.xml` file and find the data source definition that you are using with Oracle Wallet.

Update the `jdbc URL` to the TNS admin alias format. Here is an example:

```
jdbc:oracle:thin:@database_name?TNS_ADMIN=oracle_wallet_path
```

Here are the values in the example URL:

- `database_name` is the name of the Oracle database.
- `oracle_wallet_path` is the directory path to your Oracle Wallet configuration.

- 7 Click **Restart SAS servers**.

How to Facilitate Authentication

Identify or Create User Accounts

Each user must have an account that provides access to the SAS Metadata Server, either directly or through a trust relationship. Determine which of the following situations applies to you and complete any tasks as indicated.

- In the simplest case, users already have accounts that are known to the Metadata Server's host, so no action on your part is required. For example, SAS Metadata Server is on UNIX, and users have accounts in an LDAP provider that the UNIX host recognizes. Or the Metadata Server is on Windows, and users have Active Directory accounts.
- In some cases, users have accounts that are not currently recognized by the Metadata Server's host. Consider the examples in the following table.

Table 12.1 *Incorporating Unrelated Accounts*

Scenario	Possible Solution
You have Active Directory accounts but the Metadata Server is on UNIX.	Enable the UNIX host to recognize the accounts. For more information, see “Pluggable Authentication Modules (PAM)” on page 177 .
You have accounts in an LDAP provider that is not known to the SAS Metadata Server host.	Enable the SAS Metadata Server itself to recognize the LDAP provider. For more information, see “Direct LDAP Authentication” on page 171 .
You have accounts that are known at your web perimeter but are not known to the SAS Metadata Server host.	Enable the SAS Metadata Server to trust users who have authenticated at the web perimeter. For more information, see “Web Authentication” on page 185. Note: This is only a partial solution, because users of desktop applications still need accounts that can be validated by the SAS Metadata Server or its host.

- In other cases, you must add accounts to your environment. Although it is technically possible to instead use SAS internal accounts for this purpose, those accounts are not intended for regular users.
- Anyone who directly connects to the OLAP server (without first connecting to the Metadata Server) needs an account with the OLAP server.

Note: Regardless of the location of your user accounts, you must also create corresponding user information in the SAS metadata. Without such information, users have only the generic PUBLIC identity in the SAS realm. By default, this identity has no access to metadata and cannot log on to most applications. For more information, see [“Authentication to the Metadata Server” on page 144](#).

Coordinate the Workspace Server

Seamless access to the SAS Workspace Server depends on coordination between that server and the Metadata Server. By default, this coordination is necessary because authentication to the SAS Workspace Server is performed by the SAS Workspace Server host. The following table provides general recommendations:

Table 12.2 *Coordinate the Workspace Server with the Metadata Server*

Scenario	Recommendation
The servers run on Windows or UNIX.	Use host authentication (either credential-based or Integrated Windows authentication). For more information, see “Host Authentication” on page 172 or “Integrated Windows Authentication” on page 174 .
The servers run on z/OS.	Use credential-based host authentication for both servers.
The servers do not recognize the same accounts.	To minimize requirements for and exposure of host credentials, SAS provides several alternate configurations. For more information, see “Mixed Providers” on page 148 .

Note: Similar coordination is not necessary for OLAP servers and stored process servers, because they use SAS token authentication (instead of host authentication) for metadata-aware connections.

How to Configure SAS Token Authentication

Note: For metadata-aware connections to the stored process server, the SAS Pooled Workspace Server, and the OLAP server, SAS token authentication is always used and no configuration is involved. For the standard SAS Workspace Server, configuring SAS token authentication is one of several options for a mixed-provider situation.

The tasks in this section might have been automatically configured by the SAS Deployment Wizard during deployment. If you chose to configure SAS token authentication at that time, no further configuration steps are required. If, however, you decided to set up SAS token authentication after installation, this section documents the steps to follow.

To configure the standard SAS Workspace Server to use SAS token authentication, complete these steps:

- 1 Log on to SAS Management Console as a user with user-administration capabilities (for example, sasadm@saspw).
- 2 On the **Plug-ins** tab, expand **Server Manager** and the application server  (for example, **SASApp**). Right-click the logical server  (for example, **SASApp - Logical Workspace Server**) and select **Properties**.

- 3 On the **Options** tab, select **SAS token authentication**. Click **OK** to save this setting.
- 4 Expand the logical server , select the server , right-click, and select **Properties**.
- 5 On the **Options** tab, from the **Launch Credentials** drop-down list, select a logon and click **OK**. The basic choice is the account for the SAS General Servers group (the **sasgvr** logon).
- 6 To make the changes take effect, refresh the object spawner's information about the server context.
 - a Connect to the object spawner server. Expand the object spawner , select the server , right-click, and select **Connect**.
 - b Refresh the object spawner server. Right-click the server, select **Refresh Spawner**, and click **Yes**.
 - c Disconnect from the object spawner server. Right-click the server and select **Disconnect**.
- 7 To validate the configuration, select the logical Workspace Server , right-click, and select **Validate**.

Note: Metadata layer access from the server (for example, library pre-assignment, PROC OLAP code, and metadata LIBNAME engine use) is still based on each requesting user's identity. However, all host access from the server is under the designated launch credential.

See Also

- [“SAS Token Authentication” on page 181](#)
- [“Mixed Providers” on page 148](#)
- [“Criteria for a Designated Launch Credential” on page 159](#)
- [“Example” on page 115](#)

How to Configure Web Authentication

Overview of Configuring Web Authentication

Note: Before you configure web authentication, verify that this is an appropriate choice in your environment. For more information, see [“Web Authentication” on page 185](#).

Configuring web authentication consists of the following high-level tasks:

- 1 Make configuration changes to SAS web applications (for example, add security information to the SAS Logon Manager application).
- 2 Make configuration changes to the SAS Web Application Server (for example, configure a realm to authenticate users against an LDAP or other source).
- 3 Rebuild and redeploy the SAS web applications. Update and restart the web application server.
- 4 Verify or adjust user information in the SAS metadata so that each user who participates in web authentication has an appropriate logon in their metadata definition.

Step-By-Step Instructions for Web Authentication

For complete instructions for setting up web authentication, see [SAS Intelligence Platform: Middle-Tier Administration Guide](#).

Logons for Users Who Participate in Web Authentication

If you choose to configure web authentication, make sure that user metadata definitions include logons as explained in this topic.

Someone who uses only web applications should have a logon in the web authentication domain. For example:

web | joe | (no password)

Someone who uses both web and desktop applications might need two logons. One logon contains the user's authenticated ID after logging on to a desktop application, and the other logon contains the user's authenticated ID after logging on to a web application. For example:

DefaultAuth	WIN\joe	(no password)
web	joe	(no password)

In the preceding example, two logons are needed because the format of the authenticated user ID differs in each context as follows:

- When Joe logs on to a desktop application (as joe), SAS obtains his user ID in down-level format (WIN\joe), and that string is matched to the user ID in Joe's DefaultAuth logon.
- When Joe logs on to a web application (as joe), SAS obtains his user ID in short format (joe), and that string is matched to the user ID in Joe's web logon.

However, if the authenticated user ID is identical in both contexts, the web logon is not needed. If SAS obtains both authenticated user IDs as joe, the web logon is not needed. In the following example, the Metadata Server is not authenticating against Windows accounts and the web logon is not needed. When Joe logs on to a web application, the presence of his DefaultAuth logon (which contains the correct user ID) is sufficient for the Metadata Server to successfully determine his metadata identity.

DefaultAuth	joe	(no password)
web	joe	(no password)

Note: If your web environment uses Integrated Windows authentication, you must pay careful attention to the format in which SAS obtains user IDs from the web realm. If you find that users of web applications have only the PUBLIC identity, it is likely that the user ID in each web logon is not in the same format as the user ID that SAS obtains from the web realm.

Note: This is not a comprehensive discussion of logons; some users might have additional logons for other purposes.

See Also

- [“How Logins Are Used” on page 152](#)
- [“Authentication to the Metadata Server” on page 144](#)
- [“Windows User ID Formats” on page 25](#)

How to Configure Direct LDAP Authentication

Before You Begin

Before you use these instructions, make sure that this is an appropriate choice in your environment. For more information, see [“Direct LDAP Authentication” on page 171](#).

Active Directory

To enable a Metadata Server on UNIX to directly recognize Active Directory accounts, locate the `sasv9_usermods.cfg` file that is in your equivalent of **SAS/Config/Lev1/SASMeta/MetadataServer** and add settings as follows:

```
/* Environment variables that describe your AD server */
-set AD_HOST myhost

/* System options that make AD the primary authentication provider */
-authpd ADIR:company.com -primpd company.com
```

After you restart SAS Metadata Server, the preceding settings cause these results:

Table 12.3 Example: User ID Formats If `-authpd ADIR:company.com -primpd company.com`

How a User Logs On	Where the Metadata Server Sends the Credentials	How the User ID Must Be Stored in Metadata ¹
<code>user</code>	To Active Directory	<code>user@company.com</code>
<code>user@company.com</code>	To Active Directory	<code>user@company.com</code>
<code>user@unrecognized-qualifier</code>	To Active Directory	<code>user@unrecognized-qualifier</code>
<code>WinDomain\user</code>	To Active Directory	<code>WinDomain\user</code> or <code>user@WinDomain</code>
<code>user@saspw</code>	To its internal provider	No logon for an internal account

How a User Logs On	Where the Metadata Server Sends the Credentials	How the User ID Must Be Stored in Metadata ¹
<i>user@host</i>	To its host	<i>user</i>

¹ If the ID is not stored in the correct format, the user can log on but has only the PUBLIC identity. Put the SAS copy of each user's ID in a logon in that user's metadata definition. Assign these logons to DefaultAuth.

Note: To optimize credential reuse, do not move inbound logons (logons that provide access to the Metadata Server) out of the DefaultAuth authentication domain.

Note: If you use external accounts for the SAS Administrator (sasadm) or the SAS Trusted User (sastrust), certain configuration files that include those user IDs must conform to the format requirements in the third column of the preceding table. Only configuration files that contain the user ID for the purpose of matching an authenticated user ID must conform (for example, the adminUsers.txt and trustedUsers.txt files).

The following table provides reference information for environment variables that are used with the Active Directory provider.

Table 12.4 Environment Variables for Active Directory

AD_HOST	The host name of the machine where Active Directory is running.
AD_OPT_REFERRALS_OFF	Specifies whether to turn off Active Directory's ability to chase referrals, by setting the environment variable to any value.
AD_PORT	The port number for Active Directory. The default is 389.
AD_STYLE	Specifies one of the following user ID formats as the only format to be used in authentication. "OLD" the down-level format (<i>domain\user</i>) "UPN" the user principal name format (<i>user@UPNsuffix</i>) "USERNAME" the short format (<i>user</i>) Consider setting the AD_STYLE environment variable if user accounts are being locked due to exceeding the permitted number of unsuccessful logon attempts. If AD_STYLE is not set, as many as three authentication attempts are made (each format is tried sequentially, until an attempt succeeds or all formats fail). For more information, see "Windows User ID Formats" on page 25.

AD_TIMEOUT	Specifies the number of seconds before time-out on the first attempt to connect. Secondary attempts use the default time-out (30 seconds).
AD_TLSMODE	Enables Transport Layer Security (TLS) encrypted communication between the Metadata Server and the Active Directory server. Set this variable to 1 to activate (for example, <code>-set AD_TLSMODE 1</code>). ¹

¹ Additional configuration is required. For more information, see [“How to Configure TLS between the Metadata Server and an LDAP Server” on page 294](#).

Other LDAP Providers

To enable a Metadata Server on UNIX or Windows to directly recognize some other LDAP provider, locate the `sasv9_usermods.cfg` file that is in your equivalent of `SAS/Config/Lev1/SASMeta/MetadataServer` and add settings as follows:

```
/* Environment variables that describe your LDAP server */
-set LDAP_HOST myhost
-set LDAP_BASE "ou=emp, o=us"

/* System options that make LDAP the primary authentication provider */
-authpd LDAP:company.com -primpd company.com
```

After you restart the Metadata Server, the preceding settings cause these results:

Table 12.5 Example: User ID Formats If `-authpd LDAP:company.com -primpd company.com`

How a User Logs On	Where the Metadata Server Sends the Credentials	How the User ID Must Be Stored in Metadata ¹
<code>user</code>	To LDAP	<code>user@company.com</code>
<code>user@company.com</code>	To LDAP	<code>user@company.com</code>
<code>user@unrecognized-qualifier</code>	To LDAP	<code>user@unrecognized-qualifier@company.com</code>
<code>user@saspw</code>	To its internal provider	No logon for an internal account
<code>user@host</code>	To its host	<code>user</code>

¹ If the ID is not stored in the correct format, the user can log on but has only the PUBLIC identity. Put the SAS copy of each user's ID in a logon in that user's metadata definition. Assign these logons to DefaultAuth.

Note: To optimize credential reuse, do not move inbound logons (logons that provide access to the Metadata Server) out of the DefaultAuth authentication domain.

The following table provides reference information for environment variables that are used with other LDAP providers.

Table 12.6 *Environment Variables for Other LDAP Providers*

LDAP_HOST	The host name of the machine where LDAP is running.
LDAP_PORT	The port number for LDAP. The default is 389.
LDAP_BASE	The base DN to use. For example: o=People, dc=orion, dc=com.
LDAP_BASE_SUFFIXES	Makes a blank-delimited set of domains (LDAP groups) available as suffixes for the LDAP_BASE and LDAP_IDATTR environment variables. For more information, see “Multiple LDAP Groups” on page 213 .
LDAP_IDATTR	(Optional) an alternative LDAP attribute that the SAS server can use to find your DN. The default is uid.
LDAP_OPT_REFERRALS_OFF	Specifies whether to turn off LDAP’s ability to chase referrals, by setting the environment variable to any value.
LDAP_PRIV_DN	The privileged DN that is allowed to search for users (for example, cn=useradmin). ¹
LDAP_PRIV_PW	The password for LDAP_PRIV_DN. You can use the PWENCODE procedure to provide an encoded password.
LDAP_TIMEOUT	Specifies the number of seconds before time-out on the first attempt to connect. Secondary attempts use the default time-out (30 seconds).
LDAP_TLSMODE	Enables Transport Layer Security (TLS) encrypted communication between the Metadata Server and the LDAP server. Set this variable to 1 to activate (for example, -set LDAP_TLSMODE 1). ²

¹ Set this variable only if users connect with a user ID instead of a DN, and the LDAP server does not allow anonymous binds.

² Additional configuration is required. For more information, see [“How to Configure TLS between the Metadata Server and an LDAP Server” on page 294](#).

Advanced Configurations

Multiple LDAP Servers

SAS can recognize multiple LDAP servers as authentication providers, regardless of whether those servers have trust relationships.

For example, to cause the Metadata Server to search three Active Directory servers (AD1, AD2, and AD3) when authenticating users, set the AUTHPROVIDERDOMAIN system option as follows:

```
-authpd (ADIR:AD1, ADIR:AD2, ADIR:AD3)
```

To enable SAS to locate all three LDAP servers, supply information about each server through associated environment variables. To indicate which LDAP server each variable describes, append the server's name to the variable name. For example, with the preceding AUTHPD syntax, you might set server-specific variables as follows:

```
set AD_HOST_AD1=machine1.company1.com
set AD_HOST_AD2=machine1.company2.com
set AD_HOST_AD3=machine1.company3.com
```

TIP Notice that the suffix of each server-specific variable corresponds to a value that is already set in the AUTHPD system option.

Note: From the perspective of a SAS server, loss of contact with a particular LDAP server is not distinguishable from other possible causes of authentication failure (such as invalid credentials). For this reason, configuring multiple LDAP servers does not provide failover support across those LDAP servers.

Multiple LDAP Groups

SAS can customize identity search behavior for different LDAP groups within an LDAP server. For each LDAP group, you can set a starting point for search (the LDAP base DN) and designate an attribute on which to search (as an alternative to the default, which is to search by uid).

To enable SAS to differentiate among multiple LDAP groups, you first set an environment variable (LDAP_BASE_SUFFIXES) that registers the LDAP group names. For example, if your LDAP user store has an LDAP group for each of two organizations (EUROPE and AMERICAS), and you want SAS to use a different

search strategy for each of those organizations, you set an environment variable as follows:

```
LDAP_BASE_SUFFIXES=EUROPE AMERICAS
```

You can then set additional environment variables to establish a base DN and an identity search attribute for each group. To indicate which LDAP group each variable should affect, you append the group name to the variable name, as depicted in the following partial example:

```
LDAP_BASE_EUROPE=OU=Europe...
LDAP_IDATTR_EUROPE=CN
LDAP_BASE_AMERICAS=OU=Americas ...
LDAP_IDATTR_AMERICAS=samaccountname
```

TIP Notice that the suffix of each group-specific variable corresponds to a value that is already registered in the `LDAP_BASE_SUFFIXES` environment variable.

Like LDAP server-specific values, LDAP group-specific values are inherited if they are not explicitly set.

Inherited Values for Environment Variables

If a variable is not set at the server-specific level, SAS obtains a value for that variable from the corresponding shared variable. The preceding example assumes that all three servers are on the default port of their respective machines (389), so no `AD_PORT` variable is explicitly set. However, if all three of the servers were instead on the same (non-default) port on their respective machines, you would add a shared `AD_PORT=` variable as follows:

```
set AD_PORT=123
set AD_HOST_AD1=machine1.company1.com
set AD_HOST_AD2=machine1.company2.com
set AD_HOST_AD3=machine1.company3.com
```

If the third server was on another port, you would also set a server-specific variable as follows:

```
set AD_PORT=123
set AD_HOST_AD1=machine1.company1.com
set AD_HOST_AD2=machine1.company2.com
set AD_HOST_AD3=machine1.company3.com
set AD_PORT_AD3=456
```

With the preceding settings, the AD1 and AD2 servers use port 123 on their respective machines. Because no server-specific port value is set for AD1 and AD2, those servers inherit the value from the shared `AD_PORT` variable. The AD3 server uses an explicitly set server-specific port value (456).

This pattern of inheriting shared values in the absence of more specific settings applies to all of the `AD_` and `LDAP_` environment variables. However, any server that is listed as an LDAP value in the `AUTHPD` option must have a corresponding server-specific `LDAP_BASE_SUFFIXES` variable.

Compound Suffixes for Environment Variables

You might need to append two suffixes to an environment variable if both of the following circumstances apply:

- You want SAS to directly authenticate users against multiple LDAP servers.
- You want SAS to use different search behavior for different LDAP groups within at least one of those LDAP servers.

If you need to specify two suffixes, append the LDAP server suffix first and the LDAP group suffix last (for example, `LDAP_BASE_LDAPserver1_LDAPgroup1=value`).

For example, suppose you specify the `AUTHENTICATIONPROVIDERDOMAIN` system option and associated environment variables as follows:

```
-authpd (LDAP:SERVER1, LDAP:SERVER2)

LDAP_HOST=hostname
LDAP_BASE=CN=Users,DC=us,DC=company,DC=com
LDAP_IDATTR=CN
LDAP_PRIV_DN=CN=useradmin
LDAP_PRIV_PW=pwd
LDAP_BASE_SUFFIXES_SERVER2=GROUPA GROUPB
LDAP_BASE_SERVER2_GROUPA=CN=Groupa,DC=us,DC=company,DC=com
LDAP_IDATTR_SERVER2_GROUPA=CN
LDAP_BASE_SERVER2_GROUPB=OU=GroupB,DC=us,DC=company,DC=com
LDAP_IDATTR_SERVER2_GROUPB=CN
```

The results are as follows:

- `SERVER1` uses all of the shared values.
- `SERVER1` searches the base DN that is set by the shared `LDAP_BASE` variable.
- `SERVER2` uses shared values for the `LDAP_HOST`, `LDAP_PRIV_DN`, and `LDAP_PRIV_PW` variables.
- `SERVER2` searches within the base DN that is set in the `LDAP_BASE_SERVER2_GROUPA`. If the identity is not found within that area, `SERVER2` also searches within the baseDN that is set in the `LDAP_BASE_SERVER2_GROUPB` variable.

Configure LDAP to Not Chase Referrals

By default, LDAP servers chase referrals. You can turn off the capability in which the server searches for and attempts to connect to servers by using the following environment variables:

- For LDAP:
`LDAP_OPT_REFERRALS_OFF: any_value`

- For Active Directory:

`AD_OPT_REFERRALS_OFF: any_value`

When you set the appropriate variable to any value, referrals are not chased.

About AUTHPD and PRIMPD

Table 12.7 Reference: SAS System Options

AUTHPD	Use this option to register and name your Active Directory provider or other LDAP provider. For more information, see “AUTHPROVIDERDOMAIN System Option” in SAS System Options: Reference .
PRIMPD	Use this option to designate your Active Directory server or other LDAP provider as the primary authentication provider for the Metadata Server. The Metadata Server directly uses its primary provider when the submitted user ID has no qualifier, the -primpd qualifier, or an unrecognized qualifier. Using this option enables users to log on using their usual user IDs (no special qualifier is required at logon time). For more information, see “PRIMARYPROVIDERDOMAIN= System Option” in SAS System Options: Reference .

About Setting Environment Variables

UNIX Specifics: On UNIX, an alternate location for specifying the environment variables is in the MetadataServer.sh shell script. For example:

```
AD_HOST=myhost
export AD_HOST
```

z/OS Specifics: On z/OS, a TKMVSENV file is used to make a list of pseudo environment variables available. A TKMVSENV PDS is created at installation. To define the environment variables, create a member in the PDS that specifies the necessary variables, and then reference this PDS member in the TKMVSENV DD statement in your started task.

For more information about setting environment variables, see the SAS system option SET= in the documentation for your host:

- [SAS Companion for Windows](#)
- [SAS Companion for UNIX Environments](#)
- [SAS Companion for z/OS](#)

How to Configure Integrated Windows Authentication

Overview

Note: These instructions apply to configuring Integrated Windows authentication (IWA) from SAS desktop applications to the SAS Metadata Server and the SAS Workspace Server. Before you configure IWA, verify that this is an appropriate choice in your environment. For more information, see [“Integrated Windows Authentication” on page 174](#).

Configuration of IWA for desktop applications can involve three distinct locations:

- Client participation in IWA is determined by a setting in each client-side connection profile. If IWA is not selected by a client, IWA is not used for that client.
- Server participation in IWA is affected by invocation commands. For example, to use IWA, the SAS Metadata Server and SAS Object Spawner start commands must include the option `-sspi`.

Note: This is the default on Windows and UNIX

- For metadata-aware connections to a SAS Workspace Server, participation in IWA is also affected by settings in that server’s metadata definition.

Note: When upgrading to a newer version of SAS 9.4, important post-deployment steps are required to ensure that IWA continues to work. For more information, see [“Post-Installation Instructions” on page 233](#).

Instructions

Take the following steps to configure Integrated Windows Authentication for your SAS 9.4 deployment:

- 1 If the SAS Metadata Server or SAS Workspace Server runs on UNIX, complete the UNIX prerequisite tasks. Before you can use IWA for a SAS server that runs

on a UNIX host, you must prepare and configure the UNIX environment. For example:

- You must acquire, install, and configure the required software. For [SAS 9.4M1](#), the only supported implementation of IWA on UNIX requires One Identity Authentication Services (QAS) 4.0.1.23 or later. For [SAS 9.4M2](#) on Linux systems and [SAS 9.4M4](#) on Solaris systems, any shared library (including QAS) that implements the GSSAPI with Kerberos 5 extensions can be used. If QAS is present on the host, SAS attempts to load its libraries before attempting to load others to retain legacy behavior.

Note: In a single deployment, do not use One Identity Authentication Services with other shared libraries.

- You must create a service account and corresponding keytab file. Participating SAS processes on UNIX must be able to read the keytab file. A keytab file is functionally equivalent to a user's password and should be secured in the same way. The keytab file should be owned by the user and group that is accessing it, and it should have the least permissive access mask (preferably 0600).
- You must set certain environment variables.

These prerequisite tasks should be performed during the installation and initial configuration phases of your deployment. Some of the implementation details differ by UNIX host. For these reasons, detailed instructions for preparing your UNIX host to support IWA are provided in a separate document. See the chapter titled "Configuring Integrated Windows Authentication" in the [Configuration Guide for SAS Foundation for UNIX Environments](#) for more information.

- 2 If the Metadata Server is clustered and runs on Windows, or if your SAS servers are configured using DNS aliases, manually register SPNs. For more information, see "[Manual Registration](#)" on page 222.
- 3 Verify an IWA connection to the Metadata Server.
 - a In a client-side connection profile, select the check box that enables Integrated Windows authentication and then attempt to connect.

TIP In the Connection Profile dialog box in SAS Management Console, click **Edit**. On the **Connection Information** page of the Edit Connection Profile wizard, select the **Use Integrated Windows authentication (single sign-on)** check box.

- b If the connection fails, verify the following:
 - The Metadata Server's start-up command includes `-sspi`.
 - The advanced IWA settings in your client-side connection profile are **Negotiate** for the security package, blank (no value) for the service principal name, and **Kerberos,NTLM** for the security package list.

- Your metadata user definition includes a logon that contains your user ID in the correct format. For more information, see [“User Logons” on page 220](#).
 - You are using a Windows desktop client (the SAS implementation of IWA is not for web applications or UNIX clients).
 - If the SAS Metadata Server runs on UNIX, the prerequisite steps have been successfully completed. (See step 1 above.)
- c After the connection succeeds, examine the Metadata Server log to confirm that an IWA connection was used. If a credential-based connection occurred, make sure that your password is not cached in the client application.

TIP In SAS Management Console, you can clear the credentials cache by selecting **File** ⇒ **Clear Credentials Cache** from the main menu.

- 4 Configure the SAS Workspace Server’s metadata definition for IWA, and verify an IWA connection to that server.

TIP If you use IWA for the SAS Metadata Server, no cached credentials are available from an initial logon. For this reason, it is a good idea to configure IWA for the standard SAS Workspace Server also, if possible (IWA is not supported on z/OS).

- a On the **Plug-ins** tab in SAS Management Console, expand **Server Manager** and the application server  (for example, **SASApp**). Right-click the logical server  (for example, **SASApp - Logical Workspace Server**) and select **Properties**.
- b On the **Options** tab, select the **Host** radio button (IWA is a form of host authentication).
- Select the **Negotiate** security package.
 - Leave the **Security package list** as **Kerberos,NTLM**.
 - Leave the **Service principal name** blank. In a standard configuration, clients expect (and know how to compose) the default SPN. Entering a value here (or on the client side) overrides this default process.
- c Log on to SAS Management Console using IWA. Right-click the **Logical Workspace Server** and select **Validate**.
- d If the connection fails, verify the following:
- The object spawner’s start-up command includes `-sspi`.
 - Your metadata user definition includes a logon that contains your user ID in the correct format. For more information, see [“User Logons” on page 220](#).
 - If the SAS Workspace Server runs on UNIX, the prerequisite steps have been successfully completed. (See step 1 above.)

- e After the connection succeeds, examine the object spawner log to verify that the connection to the SAS Workspace Server was made using IWA. If the spawner log indicates that credential-based authentication occurred (instead of IWA), the user's context includes credentials for the SAS Workspace Server host. Make sure that the user does not have a cached or stored password for the SAS Workspace Server's authentication domain.

Note: Even if IWA is configured, any available cached or stored credentials are preferentially used.

- 5 If the SAS Workspace Server needs to access Kerberized network resources (such as network file systems or IWA connections to databases):
 - Edit the **Security package list** so that only **Kerberos** is specified.
 - In Active Directory, make the object spawner account trusted for delegation to all services. For more information, see [“Windows Privileges” on page 23](#).
- 6 Inform users that they can select the **IWA** option when they log on to desktop applications such as SAS Information Map Studio, SAS Data Integration Studio, SAS OLAP Cube Studio, SAS Management Console, and SAS Enterprise Guide. In general, users should not make changes to the advanced IWA settings in their client-side connection profiles.

See Also

- [“Checking the Status of Servers” in SAS Intelligence Platform: System Administration Guide](#)
- [“Default Locations for Server Logs” in SAS Intelligence Platform: System Administration Guide](#)
- [“Spawner Invocation Options” in SAS Intelligence Platform: Application Server Administration Guide](#)

User Logons

If you choose to configure IWA, make sure that user metadata definitions include logons with properly formatted user IDs. The format of the stored user IDs must match the format in which authenticated user IDs are returned to the target server. Failure to meet this requirement causes the user to have only the generic PUBLIC identity (which, by default, cannot even log on to most applications).

In the standard configuration, the appropriate format varies as follows:

- If the target server is on Windows, the authenticated user ID is returned in qualified format, so the stored user ID should be qualified (for example, WIN\joe or fred.smith@company.com).

- If the target server is on UNIX, the authenticated user ID is returned in short format (it is not qualified), so the stored user ID should not be qualified (for example, `joe` or `fred.smith`).

If you need to align formats, use the `SASUSEKERBNAME` environment variable. For example, you might use this environment variable in either of the following circumstances:

- The SAS Metadata Server is on Windows, the SAS Workspace Server is on UNIX, both are using IWA, and you do not want to store two logons for each user.
- You need to distinguish between two different users, in two different Kerberos realms, who happen to have the same `sAMAccountName` name (for example, `joe@US.COMPANY.COM` and `joe@EMEA.COMPANY.COM`).

For more information, see [“Windows User ID Formats” on page 25](#).

Registering SPNs

Account Information

When you use a service account instead of a local system account, the default registered SPNs need to be removed from the computer account. As a result, other services that rely on the SPNs need to run under the same service account. The following services use the same SPNs and therefore need to run under the same service account:

- SAS object spawner
- SAS OLAP Server
- SAS Metadata Server

Automatic Registration

In a standard configuration on Windows, you do not manually register service principal names (SPNs). Instead, each participating SAS server uses the local system account under which it runs to self-register two SPNs against the computer object of the machine where it runs. These Active Directory registrations are in the format `SAS/fully-qualified-machine-name` and `SAS/netbios-machine-name` (for example, `SAS/machineA.company.com` and `SAS/machineA`).

In a standard configuration on Windows, clients can construct a standard SPN (because they know the format and the target machine), so users are not required to supply an SPN when they initiate an IWA connection.

Manual Registration

In circumstances such as the following, manual registration of SPNs is necessary in order to support Kerberos connections:

- Your SAS Metadata Server is clustered and runs on Windows. For more information, see [“IWA to a Clustered Metadata Server” on page 227](#).
- Your SAS servers use DNS aliases, so the SPNs must reference an alias instead of a machine name. For more information, see [“IWA to Servers That Use DNS Aliases” on page 228](#).
- You are using IWA for servers on UNIX systems.

CAUTION

SPNs must be unique among the objects that are in the same Active Directory forest. A given SPN must not be assigned to more than one object. If a duplicate SPN is found in response to a TGS request, the Key Distribution Center sends an error to the client that the principal was not found.

To manually register an SPN, use the Microsoft tool [setspn](#). For example, the following command registers `customValue` as the SPN for all servers that run as services under the local system account on a machine that is named `myServer`:

```
setspn -A customValue myServer
```

Note: You must be a Windows domain administrator in order to use the `setspn` command.

Note: You must also make any necessary adjustments to client-side connection profiles (and the logical SAS Workspace Server definition, if applicable). For example, if you supplied a custom value for the SPN, specify the new `customValue` in the **Service principal name** fields.

UNIX Specifics: On UNIX, the SPNs that are used must be listed in the keytab file. In addition to running `setspn` to set a custom SPN, and making sure that client connection profiles use that custom SPN, you must generate a new keytab file that includes the new SPN. For more information, see the chapter titled “Configuring Integrated Windows Authentication” in the [Configuration Guide for SAS 9.4 Foundation for UNIX Environments](#).

Manually Registering Object Spawner SPNs

Note: The following manual registration is required only on Windows systems. It is not required on UNIX systems.

By default, the SAS Object Spawner service runs as a local system. The best practice is to set this service to run under a service level account. Although the account must be an administrator, it enables system administrators to allow delegation to a single account rather than to the whole machine if delegation must be set to allow access to third-party resources (such as SQL and network shares).

When using a service level account to run the object spawner service in a SAS Grid environment, you need to configure the default SPNs:

```
setspn -A SAS/computerNetbios -u domain\ObjectSpawnerServiceAccount
```

```
setspn -A SAS/computerFullname -u domain\ObjectSpawnerServiceAccount
```

In non-grid environments, you can configure custom SPNs, such as the following:

```
setspn -A SASWS/computerNetbios -u domain\ObjectSpawnerServiceAccount
```

```
setspn -A SASWS/computerShortname -u domain\ObjectSpawnerServiceAccount
```

Note: The preceding SPN statements differ from the one that is built if you use the local system.

Manually Registering SQL SPNs

When registering an SQL Server database to use IWA, additional configuration steps are needed. The following scenarios provide information about possible configurations.

In the first scenario, the SQL Server service runs under a service account and SQL Server service is not on the same machine as the object spawner service. For this scenario, follow these steps:

- 1 To register custom SPNs for the service, run the following commands:

```
setspn -A MSSQLSVC/computerNetboisName: sqlPort -u domain\sqlServiceAccount
```

```
setspn -A MSSQLSVC/fullyQualifiedDomainName: port -u domain\sqlServiceAccount
```

```
setspn -A MSSQLSVC/machineName: sqlInstance -u domain\sqlServiceAccount
```

Create the SPNs for each DNS alias, if any are being used.

- 2 Run the `setspn -x` command to verify that no duplicate SPNs exist.
- 3 Verify that the SQL Server is accepting Kerberos connections. Run the following command from the SQL Management Studio on the computer where the object spawner service runs:

```
SELECT auth_scheme FROM sys.dm_exec_connections WHERE session_id = @@spid;
```

If the query does not return KERBEROS, the SPNs might not be created correctly, or there might be other issues, such as port 88 (the Kerberos ticketing service) is blocked.

- 4 Create an ODBC connection on the same machine where the spawner resides, by completing the following steps:

- a From the Windows **Start** menu, select **Control Panel** ⇒ **Administrative Tools** ⇒ **Data Sources (ODBC)**. The ODBC Data Source Administrator dialog box is displayed.
- b Select the **System DSN** tab and click **Add**.
- c Use the Create New Data Source wizard to configure the connection.

In the second scenario, the SQL Server service runs under a service account and is on the same machine as the object spawner service. In this scenario, since both services reside on the same machine, no delegation is required for the object spawner to access the SQL Service.

Note: You need to configure delegation if the object spawner needs to delegate credentials across the network (for example, to access a network share).

In the third scenario, the SQL Server service runs under a local system account and is on the same machine as the object spawner service. This configuration is not a best practice and is therefore not supported by Microsoft. If a custom SPN for the SQL Server service account is required or preferred, the Microsoft SQL Server 2012 Native Client needs to be installed and configured.

Forcing Kerberos

If you choose to use the SAS implementation of Integrated Windows authentication (IWA), and you need to ensure that the Kerberos protocol is always used, follow these instructions. These instructions assume that you have already fully configured IWA.

Note: You cannot use Windows local accounts with this configuration, because those accounts cannot use Kerberos.

- 1 Specify **-secpackagelist "Kerberos"** in your equivalent of the following locations:
 - **SAS\Config\Levl\SASMeta\MetadataServer\sasv9_usermods.cfg** (for the Metadata Server)
 - **SAS\Config\Levl\SASApp\OLAPServer\sasv9_usermods.cfg** (if you need to support direct IWA connections to an OLAP server on Windows)
 - **SAS\Config\Levl\ObjectSpawner\ObjectSpawner_usermods.bat** (if the object spawner is on Windows). Complete these steps:
 - 1 From the Windows **Start** menu, stop the object spawner by selecting **All Programs** ⇒ **SAS** ⇒ **Configuration** ⇒ **Config - Levl** ⇒ **Object Spawner - Stop**.
 - 2 To uninstall the object spawner, from the spawner's configuration directory, enter the following command:


```
ObjectSpawner.bat -remove
```


- 3 Append the following command to the USERMODS variable in the ObjectSpawner_usermods.bat file:

```
-secpackagelist "Kerberos"
```

- 4 To re-install the spawner service, enter the following command:

```
ObjectSpawner.bat -install
```

- 2 Restart the SAS Metadata Server.
- 3 Make sure that the SAS Workspace Server's metadata definition includes only **Kerberos** in the **Security package list** field. This setting is located in SAS Management Console, on the **Plug-ins** tab, under **Server Manager**. The setting is on the **Options** tab of the logical SAS Workspace Server definition.

Note: On Windows systems, if the SAS Workspace Server accesses network resources (such as network file systems or IWA connections to databases), you must also mark the account under which the spawner runs as **trusted for delegation to all services**. For more information, see [“Windows Privileges” on page 23](#).

Note: On UNIX systems, the account that the SAS SPN is registered against must be trusted for delegation in Active Directory.

- 4 Restart the object spawner.

TIP In general, it is not necessary to also change the default IWA setting in client-side connection profiles. If a server accepts only Kerberos, then clients with the default setting of Negotiate (and both Kerberos and NTLM in the security package list) use Kerberos. However, in some circumstances, the client's Windows system chooses to initiate communication using NTLM and is unable to comply with the server requirement by switching to Kerberos. For example, if the client and server are on the same machine, the client chooses NTLM. In these circumstances, you must adjust the client-side settings to specify only Kerberos.

UNIX Specifics: It is not necessary to force use of Kerberos on UNIX, because IWA on UNIX supports only the Kerberos protocol.

Supported Protocols

IWA requires agreement between client and server about which security protocol to use when exchanging authentication packets. The following table provides details:

Table 12.8 Integrated Windows Authentication Settings

Server Setting	Associated Requirements
Negotiate security package	■ The client must select the Negotiate security package. ■ The server must have a security package list. By default, servers have a security package list that offers the Kerberos and NTLM protocols. ■ At least one of the protocols in the server's security package list must be offered by the client. ■ At least one of the protocols that are offered by both parties must actually be supported by both parties. The NTLM protocol is not supported on UNIX. ■ The Kerberos protocol can be used only if the client knows the server's SPN, as explained in the following row.
Kerberos security package	■ The client must select the Kerberos security package. ■ Both client and server must actually support the Kerberos protocol. ■ The client must know the server's service principal name (SPN). In a standard configuration, this is transparent. Clients expect (and know how to construct) a default SPN in the format SAS/machine (for example, SAS/machineA.na.company.com), so explicitly providing the SPN is not required. Note: For a server on UNIX, the <i>machine</i> value must be specified as a fully qualified domain name (FQDN). For a server on Windows, you can instead specify the machine's NetBIOS name, but the FQDN is preferred (because a NetBIOS name is not guaranteed to be unique).
NTLM security package	■ The client must select the NTLM security package. ■ Both client and server must actually support the NTLM protocol. The NTLM protocol is not supported on UNIX.

IWA to an OLAP Server

In the standard configuration, the OLAP server supports IWA for direct connections (for example, from an open OLAP client that uses an OLE DB Provider for OLAP). This support is similar to the Metadata Server's support of IWA for direct connections from SAS desktop clients. In both cases, client-side connection information must request that IWA is used.

Note: In the platform, most connections to the OLAP server are metadata-aware, not direct (the client first connects to the Metadata Server and then connects to the OLAP server, rather than initially connecting to the OLAP server). Metadata-aware connections to the OLAP server use SAS token authentication (they do not use IWA).

IWA to a Clustered Metadata Server

Introduction

A clustered Metadata Server on Windows runs under a domain account. No local system account is available to perform the usual self-registration of SPNs. Instead, you must manually register SPNs for all of the host machines in the cluster.

UNIX Specifics: No additional steps are needed to configure IWA in a clustered Metadata Server environment on UNIX.

Example

For a Metadata Server that runs as `sasmeta@company.com` on three dedicated machines (`machineA`, `machineB`, and `machineC`), you might register the following SPNs:

- `SAS/machineA.company.com`
- `SAS/machineA`
- `SAS/machineB.company.com`
- `SAS/machineB`
- `SAS/machineC.company.com`
- `SAS/machineC`

In this example, the SPNs are registered against the Active Directory principal that corresponds to the `sasmeta@company.com` account. For more information, see [“Manual Registration” on page 222](#).

Shared Hardware

If other SAS servers that run as services share any of the clustered Metadata Server’s hardware, Active Directory conflicts can occur, because one SPN cannot be associated with two Active Directory identities.

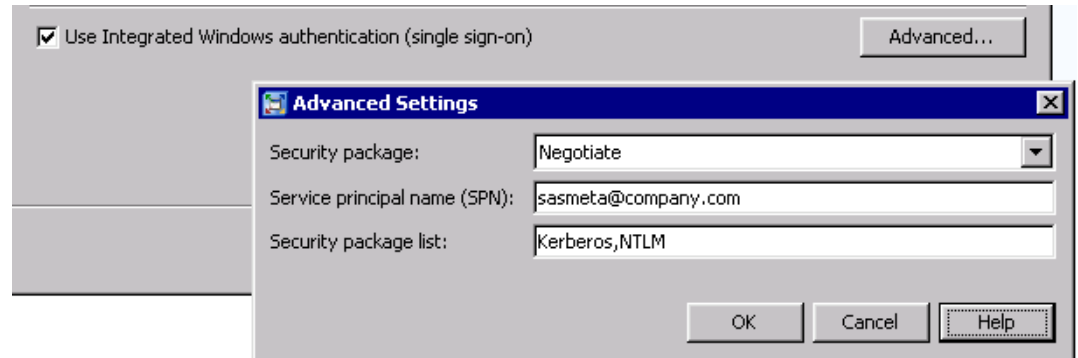
You can work around such conflicts by using either of the following methods:

- Reconfigure the other servers to run under the Metadata Server’s domain account (instead of running under the local system account).

Note: If you make this adjustment to the object spawner, the domain account must be a Windows administrator on the spawner’s host and must have the

Windows user rights **Adjust memory quotas for a process** and **Replace a process level token**. These user rights assignments are part of the local security policy for the Windows computer that hosts the spawner.

- Adjust the configuration to reference the domain account's user principal name (UPN). For example, if the Metadata Server runs as `sasmeta@company.com`, and the UPN that is registered in Active Directory for that account is `sasmeta@company.com`, then users should enter the value `sasmeta@company.com` in the **Service principal name (SPN)** field of the IWA Advanced Settings dialog box in their client-side connection profiles.



IWA to Servers That Use DNS Aliases

To support Kerberos connections to a server that uses a DNS alias, manually register SPNs that reference the alias (instead of referencing the machine name).

For example, for a Metadata Server on Windows that runs under the local system account and uses the alias *DNSAlias*, you might register the following custom SPNs.

- `SAS/DNSAlias`
- `SAS/DNSAlias.company.com`

In this example, the SPNs are registered against the Active Directory computer object that corresponds to the machine on which the Metadata Server runs. For more information, see [“Manual Registration” on page 222](#).

About IWA from SAS Web Applications

The SAS implementation of IWA is for desktop applications only. Web applications can use IWA if they are configured for web authentication and the web application server supports IWA. For more information, see [SAS Intelligence Platform: Middle-Tier Administration Guide](#).

How to Configure Kerberos Constrained Delegation

Considerations for Windows: Identity and Ownership

Without constrained delegation, Workspace Server processes run under the client's user identity. When constrained delegation is enabled, the Workspace Server process must be run under the *trusted identity*. This is the identity trusted for delegation to the specified services. The trusted identity would be the same identity under which the Metadata Server and SAS Object Spawner run. Each thread in the Workspace Server impersonates the client's user identity.

Any thread or child process of the Workspace Server that is not impersonating will have the privileges and access of the trusted identity.

Any files or any database entries created must be verified to be owned by the client user identity. Any file accessed or any database entries access must be verified such that they can only be accessed appropriately by the client.

In the Windows Task Manager, you can no longer determine which client owns a specific process. However, you can use SAS Management Console to find this information:

- 1 On the **Plug-ins** tab, expand **Server Manager** and the application server  (for example, **SASApp**). Right-click the logical server  (for example, **SASApp - Logical Workspace Server**) and select **SASApp - Workspace Server**.
- 2 Click **Connect** to the desired host.
- 3 The process ID and owner will be displayed.

Considerations for Windows: Enabling Paths in UNC Format

If you are using Kerberos constrained delegation, additional setup is required to enable SAS servers to access paths in the Universal Naming Convention (UNC) format. A domain administrator can complete the following steps in the Microsoft Windows Active Directory Domain Services:

- 1 Start the Windows Active Directory Users and Computers (ADUC) management program.
- 2 In the **Computers** folder, right-click the SAS host or hosts that require the ability to access UNC paths. Select **Properties** from the menu.
- 3 In the Properties dialog box, click the **Delegation** tab.
- 4 Click to enable the following options:
 - **Trust this computer for delegation to specific services only**
 - **Use any authentication protocol**
- 5 Click **Add**.
- 6 Add an entry to the Services table for each Common Internet File System (CIFS) service to which the SAS process needs to be able to connect.
- 7 Click **OK** to save the changes.

Configure Kerberos

Note: The following instructions apply to Linux, Solaris (deprecated with SAS 9.4 M9), and Windows systems.

However, SAS Grid Manager for Platform depends on IBM Spectrum LSF, which does not support Kerberos constrained delegation on UNIX.

Starting in SAS 9.4M6, Kerberos is support for constrained delegation. Prior to enabling Kerberos constrained delegation, SAS servers and spawners must first be configured for Kerberos, which is also referred to as Integrated Windows Authentication. For more information, see [“How to Configure Integrated Windows Authentication” on page 217](#).

To fully enforce constrained delegation, additional configuration steps are required in Active Directory. These steps must be completed by a Windows domain administrator using the Active Directory Users and Computers tool. Each Kerberos-protected resource (for example, Microsoft SQL Server) that a SAS Workspace Server can access must be defined against the account being configured for constrained delegation.

- 1 As a Windows domain administrator, from the Windows **Start** menu, select **Control Panel** ⇒ **Administrative Tools** ⇒ **Active Directory Users and Computers**. The Properties dialog box is displayed where you can access the relevant account and grant the delegation privilege.

Although the SAS Workspace Server accesses the Kerberos protected resource, the delegation privilege is granted to the SAS Object Spawner.

- If the SAS Object Spawner is running on Windows under the local system account, select the host machine in Active Directory under **Computers**. On

the **Delegation** tab, select **Trust this computer for delegation to specified services only**.

- If the SAS Object Spawner is running on Windows under a domain account, select the user account in Active Directory, typically under **Users**. On the **Delegation** tab, select **Trust this user for delegation to the specified services only**.
 - If the SAS Object Spawner is running on UNIX, select the account under which the SAS Object Spawner service principal name (SPN) is registered. The default SPN is typically **SAS**.
- 2 Select the **Use any authentication protocol** option.
 - 3 Click **Add**, and then click **Users and Computers**.
 - 4 Specify a name.
 - If the service that the SAS Workspace Server accesses is running under the local system account (which is the default for Microsoft SQL Server and CIFS), specify the host name where the service is running.
 - If the service the SAS Workspace Server accesses is running under a domain account, specify the domain account under which the service is running.
 - 5 Click **Check Names**, and then click **OK**.
 - 6 Select one or more SPNs for the services, and then click **OK**.
Repeat as necessary until all services have been added. When finished, click **OK** to exit the Properties dialog box.
 - 7 For sites using SAS Web Applications such as SAS Studio, the SAS middle-tier HTTP SPN must be trusted for delegation to the account that the SAS Metadata Server and SAS Object Spawner are running under.
 - a As a Windows domain administrator, from the **Start** menu, select **Control Panel** ⇒ **Administrative Tools** ⇒ **Active Directory Users and Computers**. From the Properties dialog box, you can access the relevant account and grant the delegation privilege.
 - On Windows systems, if the SAS middle tier is running under the local system account, select the host machine in Active Directory under **Computers**. On the **Delegation** tab, select **Trust this computer for delegation to specified services only**.
 - If the SAS middle tier is running under a domain account, select the user account in Active Directory, typically located under **Users**. On the **Delegation** tab, select **Trust this user for delegation to the specified services only**.
 - If the SAS middle tier is running on UNIX, select the account under which the HTTP SPN is registered for this host.
 - b Select the **Use any authentication protocol** option.
 - c Click **Add**, and then click **Users and Computers**.
 - d Specify a name.

- If the SAS Object Spawner runs on Windows under the local system account, specify the host name where the SAS Object Spawner is running.
 - If the SAS Object Spawner runs on Windows under a domain account, specify the domain account under which the SAS Object Spawner is running.
 - If the SAS Object Spawner runs on UNIX, specify the account under which the SAS Object Spawner service principal name (SPN) is registered.
- e Click **Check Names**, and then click **OK**.
- f Select the SPN for the SAS Object Spawner (which is typically **SAS**), and then click **OK**.
- Click **OK** to exit the Properties dialog box.
- 8 If the SAS Metadata Server is running on a separate host, repeat step [Step 7a on page 231](#) through step [Step 7f on page 232](#) for the SAS/*metadata_server_hostname*.

Kerberos on UNIX

Supported Systems and Requirements

MIT Kerberos 1.11.6 or later is required. If the operating system does not provide a supported version, the source code can be downloaded from [MIT Kerberos Distribution Page](#). Downloading requires building the source code and installing the new libraries. Instructions can be found under the **Documentation** link on the release page for the chosen version.

Note: Heimdal and Quest are currently not supported.

Note: SAS Grid Manager for Platform depends on IBM Spectrum LSF, which does not support Kerberos constrained delegation on UNIX. Further limitations are described in this [SAS Note](#).

Post-Installation Instructions

Depending on the type of keytab file being used, additional tasks might be required for specifying the SPN. After completing ["Configure Kerberos" on page 230](#), the

following steps provide guidance on additional tasks that you might need to complete:

1 Set the SPN environment variable.

- If you are using the default SPN and an SPN-based keytab, no additional tasks are required. The default SPN for SAS server-tier services is `SAS/FQDN_of_host`, where FQDN is the fully qualified domain name of the host. The default SPN for SAS Workload Orchestrator is `HTTP/FQDN_of_host`.
- If you are using a non-default SPN and an SPN-based keytab, the `SAS_SERVICE_PRINCIPAL` environment variable must be set to the desired SPN. For SAS servers, this environment variable should be set in the `SAS-configuration-directory/Levn/level_env_usermods.sh` file. If SAS Workload Orchestrator is installed, the environment variable must also be set in the `SAS-configuration-directory/Levn/Grid/sgmg_usermods.sh` file. For example:

```
SAS_SERVICE_PRINCIPAL=MYSAS/my_sas_server.com
export SAS_SERVICE_PRINCIPAL
```

- If you are using keytab files based on the User Principal Name (UPN), the `SAS_SERVICE_PRINCIPAL` environment variable must be set to the user or service account that matches the UPN. For SAS servers, this environment variable should be set in the `SAS-configuration-directory/Levn/level_env_usermods.sh` file. If SAS Workload Orchestrator is installed, the environment variable must also be set in the `SAS-configuration-directory/Levn/Grid/sgmg_usermods.sh` file on all grid worker nodes. For example:

```
SAS_SERVICE_PRINCIPAL=sas-services@REALM.COM
export SAS_SERVICE_PRINCIPAL
```

2 In the `SAS-configuration-directory/Levn/level_env_usermods.sh` file, set the `SAS_CONSTRAINED_DELEG_ENABLED` environment variable to "1". If SAS Workload Orchestrator is installed, the environment variable must also be set in the `SAS-configuration-directory/Levn/Grid/sgmg_usermods.sh` file on all grid worker nodes. For example:

```
SAS_CONSTRAINED_DELEG_ENABLED=1
export SAS_CONSTRAINED_DELEG_ENABLED
```

3 Restart the SAS spawners and servers.

```
SAS-configuration-directory/Levn/sas.servers restart
```

Kerberos on Windows

Post-Installation Instructions

Constrained delegation support in SAS is not enabled by default. After completing [“Configure Kerberos” on page 230](#), complete the following steps on all machines

that are running SAS Object Spawner, SAS/CONNECT Spawner, and all SAS Workload Orchestrator nodes:

- 1 As a local administrator, from the **Start** menu, select **Control Panel** ⇒ **System and Security** ⇒ **System** ⇒ **Advanced system settings**. The System Properties dialog box is displayed.
- 2 On the **Advanced** tab, click **Environment Variables**. The Environment Variables dialog box is displayed.
- 3 In the **System variables** section, set the **SAS_CONSTRAINED_DELEG_ENABLED** System environment variable to 1. Click **OK** to exit the Environment Variables dialog box.
- 4 Click **OK** to exit the Properties dialog box.
- 5 Restart the machine.

About IBM Z Multi-Factor Authentication (MFA)

Starting with SAS 9.4M9, to apply MFA to SAS/CONNECT and SAS/SHARE on z/OS, IBM Z Multi-Factor Authentication is supported. For SAS 9.4M5 through SAS 9.4M8, SAS provided Limited Support for administrators who followed instructions in [KB0041877](#).

Here are some key points about IBM Z MFA support in SAS 9.4M9:

- IBM Z Multi-Factor Authentication supports both In-Band and Out-of-Band authentication. SAS supports both types of authentication. In-Band authentication is supported with the SAS/CONNECT spawner on z/OS, and Out-of-Band authentication is supported with SAS/CONNECT and SAS/SHARE.
 - Out-of-Band authentication enables you to authenticate on a user-specific application with one or more factors to retrieve a cache token credential that you use to log in. Out-of-Band authentication is described in the [IBM Multi-Factor Authentication for z/OS User's Guide](#). MFA Out-of-Band authentication requires you to authenticate "out-of-band" with one or more factors to retrieve an In-Band authentication code called a "cache token credential."
 - For In-Band authentication, you generate a token and use that token to log directly in to the application.
- The RACF user account must be configured for IBM MFA Out-of-Band authentication, and a policy that defines the following factors must be supplied:
 - Token credential reuse. This factor determines whether the cache token credential can be reused and how long it can be reused.

- ❑ Authentication code criteria. If the user is configured for multiple factors, all configured authentication factors must succeed to receive an In-Band authentication code.

When you troubleshoot problems using IBM MFA on z/OS, consider the following:

- What factors are configured for the user?
- Is the SAS 9 Supervisor Call (SVC) module installed?
- Is PASSPHRASE support enabled for the user?
- What is the length of the cache token?
- Is tracing enabled? Enable MFA tracing using the Started Task Control (STC) name with the following operator command.

```
F mfa stc name,STC SET TRACE LEVEL 2 /* levels 0 -> 3 */
```

- Are logging levels adequate? Enable enhanced logging for the SAS/CONNECT spawner.

Some applications have authentication properties that can prevent certain MFA factors from working properly. Here are a few authentication properties to consider:

- No passphrase support – Some MFA authenticators can generate chars tokens that are longer than eight characters.
- No password change field – Certain MFA factors can use the **Password Change** field to change an RSA SecurID PIN during logon.
- PassTickets authenticators – These are not supported by MFA at this time.
- Replay of passwords – Some MFA credentials are different at every logon and cannot be reused.

For detailed information about how to configure IBM Z MFA, see the [IBM Z Multi-Factor Authentication User's Guide](#). After IBM z/MFA has been enabled, minor changes to your SAS programs are required. For user sign-on with MFA, where you currently provide a user ID and password (to a SAS/CONNECT session, for example), in place of the password, you provide the MFA token.

How to Store Passwords for the Workspace Server

Note: This is one of several solutions to a mixed provider situation (and it is not a preferred approach). For more information, see [“Mixed Providers” on page 148](#).

- 1 Log on to SAS Management Console as someone who has user administration capabilities (for example, sasadm@saspw).

- 2 On the **Plug-ins** tab, expand **Server Manager**, the application server  (for example, **SASApp**) and the logical server  (for example, **SASApp - Logical Workspace Server**).

(Optional) Right-click the logical server, select **Properties**, and select the **Options** tab. Make sure that the **Authentication service** is set to use **Host** with **Username/Password**. Click **OK** to close the dialog box and return to Server Manager.
- 3 Below the logical server , select the server  (for example, **SASApp - Workspace Server**).
- 4 In the display panel, right-click the server's connection object  and select **Properties**.
- 5 On the **Options** tab, notice the value in the **Authentication domain** drop-down list. If the value is something other than DefaultAuth, proceed to step 6. Otherwise, complete these steps:
 - a Next to the **Authentication domain** drop-down list, click **New**.
 - b In the New Authentication Domain dialog box, enter a name such as **UNIXAuth** or **WorkspaceAuth** (you can use any name that is meaningful to you). Click **OK**.
 - c On the **Options** tab, make sure the new authentication domain is selected in the **Authentication domain** drop-down list. Click **OK**.
- 6 Create a SAS copy of credentials that are known to the SAS Workspace Server's host operating system. In most cases, you store shared credentials in a group's metadata definition. You can also store a unique set of individual credentials in each user's metadata definition. Each logon must be assigned to the SAS Workspace Server's authentication domain. Each logon must include both a user ID and a password.

Here is an example for a SAS Workspace Server on UNIX:

```
UNIXAuth | myID | mypassword
```

Here is an example for a SAS Workspace Server on Windows:

```
WINAuth | Win\myID | myWINpassword
```

If you store credentials for a SAS Workspace Server that runs on Windows, give users the Windows privilege **Log on as a batch job**.

Note: If you do not store the passwords, users are prompted for such credentials when they make a request that requires access to the SAS Workspace Server. Only desktop applications and SAS Web Report Studio provide such secondary logon prompts.

Note: Do not instead leave the SAS Workspace Server in DefaultAuth and move inbound logons to some other authentication domain. Failure to follow this recommendation does not affect the initial logon process, but it does interfere with access to the SAS Workspace Server. By default, all clients insert the credentials that a user submits at the logon prompt into that user's context as a DefaultAuth

entry. This cached DefaultAuth entry has priority over any DefaultAuth credentials that are retrieved from logons in the metadata.

See Also

- “Credential Management” on page 168
- “How Logins Are Used” on page 152
- “Windows Privileges” on page 23

How to Store Passwords for a Third-Party Server

Note: Use these instructions to provide seamless access to a third-party server that uses a proprietary authentication provider (for example, Oracle). These instructions associate the database logons with a user or group, not directly with a database library.

- 1 Verify that the third-party server is registered in the metadata and is in its own authentication domain.
 - a Select the third-party server’s definition under **Server Manager** on the **Plugins** tab in SAS Management Console.
 - b In the display panel, right-click the server’s connection object  and select **Properties**. The server’s authentication domain assignment is on the **Options** tab.
- 2 In the server’s authentication provider, identify or create accounts. Use any of the following approaches (here, Oracle is used as an example):
 - Create an individual Oracle account for each user. This provides the greatest accountability, but can also necessitate storing many Oracle user IDs and passwords in the metadata.
 - Create one Oracle account that all users share. This greatly reduces the need to store Oracle user IDs and passwords, but also results in a loss of individual accountability.
 - Create a few Oracle accounts. Each account is shared by several users. This middle-of-the-road approach enables you to make some access distinctions in Oracle and store only a few Oracle user IDs and passwords in the metadata.

- 3 In the metadata, store the user IDs and passwords for each account as follows (here, Oracle is used as an example):

- If you created individual accounts on the Oracle server, add an Oracle logon to each user definition.
- If you created one shared account on the Oracle server, identify or create a group that contains the users who access the Oracle server. Give that group a logon that includes the user ID and password for the Oracle shared account.

.....

Note: If you want to provide access for all registered users, give the logon to the SASUSERS group.

.....

.....

Note: If you want to provide access for all users (including users who do not have an individual SAS identity), give the logon to the PUBLIC group.

.....

- If you created several shared accounts on the Oracle server, identify or create a user group in the metadata for each shared account. Give each group a logon for the Oracle server, and assign each user who connects to Oracle to one of the groups.

Assign these logons to the third-party server's authentication domain. Store both an ID and a password in each logon.

Here is an example for an Oracle server:

```
OracleAuth | myORAid | myORApasword
```

.....

Note: If you do not store the passwords, users of desktop applications are prompted for such credentials when they make a request that requires access to the server. SAS Web Report Studio has an interactive password management feature. Other web applications do not support interactive logons to secondary servers.

.....

See Also

- [“Credential Management” on page 168](#)
- [“How Logins Are Used” on page 152](#)

How to Change Internal Account Policies

Server-Level Policies

Initial Settings

Here are the initial server-level policies for internal accounts:

- Accounts do not expire and are not suspended due to inactivity.
- Passwords must be at least six characters, do not have to include numbers or mixed case, and do not expire.
- The five most recent passwords cannot be reused.
- After three failed attempts to log on, an account is locked for one hour. An administrator can unlock the account by accessing the **Accounts** tab in the user's definition in SAS Management Console.
- A forced password change occurs on first use and after a password is reset. This policy applies only to accounts with passwords that periodically expire. By initial policy, passwords do not expire, so forced password changes do not occur.

Current Settings

You can examine the current policies in SAS Management Console. On the **Plug-ins** tab, expand the **Metadata Manager**, right-click the **Active Server** node, and select **Properties**. The **Options** tab displays any server-level internal account policies that have been explicitly set.

Note: You can also examine explicitly set policies by viewing the omaconfig.xml file.

Syntax for the InternalAuthenticationPolicy Element

To change the server-level policies, edit the `InternalAuthenticationPolicy` element in the Metadata Server's omaconfig.xml file, and restart all SAS processes.

The next time the password is changed for internal accounts, the new hash function is used to store the password in the metadata.

Here is the syntax for each policy option:

Note: The following option names are case-sensitive.

Note: A value of **T** has aliases (1 or Y). A value of **F** has aliases (0 or N).

`ChangeDelayInMinutes="number"`

specifies the number of minutes that must elapse between password changes. This applies only when you are resetting your own password.

`DigitRequired="T | F"`

specifies whether passwords must include at least one digit. To enforce this requirement, specify **T**.

`ExpirationDays="number"`

specifies the number of days after a password is set that the password expires. A value of 0 prevents passwords from expiring.

`ExpirePasswordOnReset="T | F"`

specifies whether a forced password change occurs on first use and after an administrative password reset. To disable this requirement, specify **F**.

`HashPasswords="SHA256-10000 | SHA256 | MD5"`

specifies how the internal account password is stored in the metadata.

IMPORTANT Starting with SAS 9.4M8, you must have the [required cryptographic libraries](#) installed in the operating system in order to use SHA-256 and higher. In earlier releases, the prerequisite for SHA-256 and higher was SAS/SECURE. If an applicable cryptographic library is not installed, MD5 is used.

Table 12.9 Options for How the Internal Account Password Is Stored

Supported Algorithm	Description
SHA256-10000	The SHA256-10000 hash function is used. SHA (Secure Hash Algorithm) is Federal Information Processing Standard (FIPS)-compliant. SHA256-10000 uses additional iterations to enhance security and is the default.
SHA256	The SHA256 hash function is used. SHA-1, SHA-256, and SHA-512 are all FIPS-approved secure hash algorithms.
MD5	MD5 hashing is used. MD5 (Message-Digest algorithm 5) is appropriate for preventing accidental exposure of information. It is used in SAS deployments on Windows.

`MinLength="number-of-characters"`

specifies the minimum length for passwords.

`MixedCase="T | F"`

specifies whether passwords must include at least one uppercase letter and at least one lowercase letter. To enforce this requirement, specify `T`.

`NumPriorPasswords="number"`

specifies the number of passwords that are maintained in each account's password history. A user cannot reuse a password that is in the user's account history.

`InactiveDaysToSuspension="number"`

specifies the number of days after which an unused account is suspended. A value of `0` prevents suspensions due to inactivity.

Note: Beginning in SAS 9.4M4, this option is no longer supported.

`LockoutDurationInMinutes="number"`

specifies the number of minutes for which an account is locked following excessive logon failures.

`NumFailuresForLockout="number"`

specifies the number of consecutive unsuccessful logon attempts that cause an account to be locked. We recommend that you do not specify `0`, because doing so can make your system vulnerable to password guessing attacks.

Example of the InternalAuthenticationPolicy Element

```
<OMAconfig>
...
<InternalAuthenticationPolicy ChangeDelayInMinutes="0"
DigitRequired="F"
ExpirationDays="0" MinLength="6" MixedCase="F" NumPriorPasswords="5"
InactiveDaysToSuspension="0" LockoutDurationInMinutes="60"
NumFailuresForLockout="3"/>
...
</OMAconfig>
```

Per-Account Policies

To override server-level policies on a per-account basis:

- 1 Log on to SAS Management Console as someone who has user administration capabilities.
- 2 On the **Plug-ins** tab, select **User Manager**.

- 3 In the display pane, clear the **Show Groups** and **Show Roles** check boxes. Right-click the user definition of the user whose SAS internal account policies you want to change. Select **Properties**.
- 4 At the bottom of the user's **Accounts** tab, click **Update**.
- 5 Make changes in the **Custom Settings** box. Not all server-level settings can be modified on a per-account basis.

Note: Two distinct expiration settings are available. Do not confuse the account expiration date with the password expiration period.

Note: To minimize administrative maintenance effort for any predefined or service identities that have internal accounts, do not add expiration dates to these accounts or expiration periods to these passwords.

The following table maps server-level policies to corresponding account-level policies. Not all policies can be set at both levels.

Table 12.10 Internal Account Policy Mapping

Server-Level Policy	Related Account-Level Setting
ExpirationDays	Set a custom password expiration period.
LockoutDurationinMinutes	Exempt from account lockout policy.
NumFailuresForLockout	Exempt from account lockout policy.
NumPriorPasswords	Exempt from password reuse policy.

TIP If you want to force a particular user to change his or her internal password after you create (or reset) the user's internal account, but you do not otherwise want the password to expire, use the maximum custom password expiration period (32,767 days).

See Also

[“SAS Internal Authentication” on page 178](#)

About the Workspace Server Options Tab

The following tips help you interpret the **Options** tab for a logical SAS Workspace Server  in SAS Management Console:

- Changes that you make on this tab take effect after you refresh the SAS Object Spawner.
- Because the displayed settings are in the metadata, they can affect only metadata-aware connections.
- The **Use Server Access Security** check box should be selected. This selection enables standard access control enforcement for the ReadMetadata permission on the server definitions.

Note: An exception is if the SAS Workspace Server is configured for client-side pooling. In that case, the check box is disabled and cleared. Server access security is not supported with client-side pooling.

- The **Host** with the **Username/Password** setting specifies that credential-based host authentication is always used. The other security packages cause IWA to be used for clients that select the IWA option (unless cached or stored credentials are available).
- The **Host** with the **Username/Password** setting does not eliminate prompting for credentials. With this setting, all desktop applications prompt users to interactively supply credentials in any circumstance where credentials are needed and are not otherwise available.
- If you select **SAS token authentication**, you must also select a server launch credential (on the **Options** tab of the server definition). Configuring a SAS Workspace Server to use SAS token authentication can be useful in a multi-host environment. For more information, see [“Mixed Providers” on page 148](#).
- The **Host** with the **Username/Password** setting can cause SAS Enterprise Guide and SAS Add-In for Microsoft Office to silently store user passwords in metadata. The storage occurs only if the SAS Workspace Server is in its own authentication domain (for example, ServerAuth) and users interactively provide credentials when they access that server. If both of these circumstances apply to your deployment, consider selecting **Prompt** instead of **Host** with **Username/Password**.
- The **Prompt** setting is similar to **Host** with the **Username/Password** setting, but it forces SAS Enterprise Guide and SAS Add-In for Microsoft Office to prompt users.

See Also

- [“Integrated Windows Authentication” on page 174](#)
- [“SAS Token Authentication” on page 181](#)

PART 4

Encryption

Chapter 13	
Encryption Model	247
Chapter 14	
Encryption Tasks	265

Encryption Model

<i>Introduction to the Encryption Model</i>	247
Two Classes of Encryption Strength	247
Two Contexts for Encryption Coverage	248
See Also	248
<i>Encryption for Data at Rest</i>	248
<i>Encryption for Data in Motion</i>	250
<i>What Are Certificates?</i>	251
<i>How SAS Uses Certificates</i>	253
SAS Components That Use Certificates	253
The Trusted CA Bundle	255
Managing Certificates in the SAS Environment	256
Certificate Formats Supported by SAS Middle-Tier Components	258
<i>Obtaining Certificates</i>	259
Overview of Obtaining Signed Certificates	259
How Many Private Keys and Signed Certificates Do I Need?	261
<i>Implementing Certificates</i>	262

Introduction to the Encryption Model

Two Classes of Encryption Strength

Two classes of encryption strength are available:

- For compatibility with legacy systems, SASProprietary encoding is supported. This method is used by default, is available in all deployments, is appropriate for preventing accidental exposure of information, and has minimal impact on performance.

If your configuration specifies an algorithm that is not available, the SASProprietary algorithm is used instead.

- For a higher level of security, SAS recommends that you use industry-standard encryption and hashing algorithms. These methods provide stronger protection and are available in all deployments, except where prohibited by import restrictions.

Note: For SAS 9.4-SAS 9.4M7, industry-standard algorithms were provided by SAS/SECURE, which was included with Base SAS. Starting with [SAS 9.4M8](#), SAS uses the cryptographic libraries that are available from the operating system instead of those delivered by SAS/SECURE. For more information, see [“Cryptographic Library Support \(Starting with SAS 9.4M8\)” in *Encryption in SAS*](#).

SAS recommends that you use the strongest security standards available for your environment.

Two Contexts for Encryption Coverage

SAS provides encryption in two contexts:

- In encryption for data at rest, the emphasis is on protection of passwords in configuration files and in the metadata repository. You can also encrypt SAS data sets.
- In encryption for data in motion, the emphasis is on protection of passwords in transit. You can also protect all traffic in transit among SAS Integrated Object Model (IOM) servers and SAS desktop clients.

Note: To ensure that only FIPS-validated encryption algorithms are used, set the ENCRYPTFIPS system option. See [ENCRYPTFIPS System Option](#) in *Encryption in SAS 9.4*.

See Also

- [“Encryption for Data at Rest” on page 248](#)
- [“Encryption for Data in Motion” on page 250](#)

Encryption for Data at Rest

In general, encryption for data at rest is set up during installation and no further adjustments are required. The following table lists supported algorithms by storage context.

Table 13.1 Encryption for Data at Rest

Storage Context	Supported Algorithms
Login password and SASPassword objects on disk in the metadata	SASProprietary encoding, AES 256 with 16-bit salt, AES 256 with 64-bit salt, or AES 256 with 64-bit salt and additional hash iterations
Internal account password on disk in the metadata	SHA256-10000, SHA256, or MD5 hashing
Password on disk in a configuration file	SASProprietary encoding, AES 256 with 16-bit salt, AES 256 with 64-bit salt, or AES 256 with 64-bit salt and additional hash iterations

Note: If your configuration specifies an algorithm that is not available, the SASProprietary algorithm and MD5 hashing are used instead. See [“Two Classes of Encryption Strength” on page 247](#).

Note: Most other metadata is not stored in an encrypted format. Configuration files and metadata repository data sets should also be host protected. See [“First-Priority Setup Tasks” in SAS Intelligence Platform: Installation and Configuration Guide](#).

TIP To make any post-installation changes, see [“How to Maximize Encryption of Stored Passwords” on page 266](#).

TIP To encrypt SAS data sets on disk, see [“ENCRYPT= Data Set Option” in SAS Data Set Options: Reference](#).

TIP To require encryption of the tables in a metadata-bound library and store the encryption key in metadata, see the [SAS Guide to Metadata-Bound Libraries](#).

Encryption for Data in Motion

In general, encryption for data in motion settings are defined during installation and no further adjustments are required. The following table lists supported algorithms by content and network traffic context. The table also references instructions for making post-installation changes.

IMPORTANT Starting with SAS 9.4M9, SAS has deprecated the NETENCRYPTALGORITHM (NETENCALG) system option values AES, DES, RC2, RC4, and TripleDES. For SAS 9.4M7 and SAS 9.4M8, this change is made with hot fixes related to [KB0041538](#). SAS strongly recommends that you change the value to SSL to specify the use of the TLS protocol. For more information, see “[TLS Support for IOM and Other SAS Servers](#)” in *SAS Intelligence Platform: Application Server Administration Guide*.

Table 13.2 Encryption for Data in Motion

Content and Context	Supported Algorithms and Protocols	Reconfiguration
Password in transit in a logon attempt (from a SAS IOM client, data provider, or Base session to a SAS IOM server)	SASProprietary encoding, RC2, RC4, DES, TripleDES, AES, or SSL. SAS strongly recommends that you use SSL (TLS).	See “ How to Configure TLS for IOM and Other SAS Servers and Change the Encryption Level ” on page 273.
General traffic among SAS IOM servers and SAS IOM clients	None, SASProprietary encoding, RC2, RC4, DES, TLS, TripleDES, AES, or SSL ¹ SAS strongly recommends that you use SSL (TLS).	
Password in transit from a client to the Metadata Server (when the client stores a login in metadata)	SASProprietary encoding	Not configurable.
Password in transit from the Metadata Server to a client (when the client retrieves a stored login)	SASProprietary encoding or AES ²	See “ How to Change Encoding for Data in Motion for Outbound Passwords ” on page 291.

Content and Context	Supported Algorithms and Protocols	Reconfiguration
Traffic from the Metadata Server to an LDAP server	None or TLS	See “How to Configure TLS between the Metadata Server and an LDAP Server” on page 294.
Traffic between the Middle-Tier components	None or TLS	See <i>SAS Intelligence Platform: Middle-Tier Administration Guide</i> .
Traffic between a web browser and the SAS Web Server	None or TLS	
SAS Deployment Agent traffic	None or TLS	For more information, see <i>SAS Intelligence Platform: Installation and Configuration Guide</i> .
SAS Environment Manager agent-to-server traffic	None or TLS	For more information, see <i>SAS Environment Manager: User's Guide</i> .
SAS Environment Manager server-to-agent traffic	TLS	

- 1 The algorithm that you select is used if the CEL option is set to EVERYTHING.
- 2 These algorithms encode the password in the transmitted XML. In addition, the response traffic is encrypted using the algorithm that is specified in the NETENCALG option.

Note: If your configuration specifies an algorithm that is not available, the SASProprietary algorithm is used instead. For more information, see “Two Classes of Encryption Strength” on page 247.

What Are Certificates?

TLS and its predecessor, Secure Sockets Layer (SSL), are cryptographic protocols that are designed to provide communication security over the internet. TLS and SSL are protocols that provide network data privacy, data integrity, and authentication. SAS strongly recommends that you apply TLS to your deployment if you have not already done so.

Note: All discussion of TLS is applicable to the predecessor protocol, SSL.

TLS uses X.509 digital certificates and asymmetric cryptography to authenticate entities—whether that entity is a server process or a human user.

When messages are exchanged between entities, a message might be intercepted, or a process or a user might be impersonated. Certificate authorities (CAs) are trusted parties that provide confidence that a public key truly belongs to an entity.

You request that the CA issue a certificate that contains your public key and that the CA has digitally signed the certificate. The certificate is referred to as a *signed certificate*. The entity that receives your message acknowledges that your signed certificate is issued by a CA that it recognizes. If it recognizes the CA, the entity is able to verify your identity.

Figure 13.1 Types of Certificates



Establishing trust is accomplished through certificates issued by three types of CAs:

- self-signed

You serve as your own CA and obtain a certificate using the certificate management tool on your machine.

- site-signed

You go to the IT department at your site to obtain a certificate.

- third-party-signed

You go to a commercial third-party CA, such as Comodo, to purchase a certificate.

For more information about TLS, see [“Providers of Encryption” in *Encryption in SAS*](#) and for more information about how SAS uses certificates in the middle-tier, see [“Overview” in *SAS Intelligence Platform: Middle-Tier Administration Guide*](#).

How SAS Uses Certificates

SAS Components That Use Certificates

SAS Deployment Agent

SAS Deployment Agent and its remote client perform configuration management operations for clustering and backups. There is one SAS Deployment Agent and its remote client per machine in the deployment. SAS Deployment Agent uses a Java keystore to store its secure certificate.

SAS Deployment Wizard configures communication between SAS Deployment Agent and its remote client using two-way client authentication with auto-generated self-signed certificates by default. Therefore, the deployment agent and its remote client each has to have access to a keystore and truststore. When multiple machines are involved, having a single keystore and truststore is necessary so that *all* of the remote clients can communicate with *all* of the deployment agents.

Credentials are generated during the deployment of the SAS Metadata Server. These credentials are retrieved from the Metadata Server during the configuration of other SAS servers. If you have multiple SAS Metadata Servers in a deployment, SAS Deployment Manager enables you to upload a set of credentials to additional metadata servers. For more information, see “Manage SAS Deployment Agent Service” in the [SAS Deployment Wizard and SAS Deployment Manager: User’s Guide](#).

SAS recommends that users who are unfamiliar with security setup use the default self-signed certificates. If your site requires a site-signed or third-party-signed certificate, you must supply paths to both the deployment agent and remote client keystore and truststore locations and their respective passwords when prompted

by SAS Deployment Wizard. For more information, see [“Providing Your Own Certificates” on page 300](#).

TIP If you want to provide your own certificates, SAS recommends that you initially configure the deployment agent with auto-generated self-signed certificates that SAS Deployment Wizard provides. Later, you can replace the auto-generated self-signed certificates with certificates that you provide, which saves you several manual configuration steps.

SAS Environment Manager

SAS Environment Manager is an operational monitoring and management system for SAS deployments. SAS Environment Manager incorporates Hyperic technology from VMware to offer enterprise-class operational features.

During installation, SAS Deployment Wizard automatically creates self-signed certificates for you. However, you can specify your own site-signed or third-party-signed certificates instead. SAS Environment Manager uses a Java keystore to store its private key and server certificate.

SAS Web Server

SAS Web Server is an HTTP proxy server that is based on Apache Open Source HTTP Server.

SAS Web Server is built and packaged by SAS. It is compiled with OpenSSL and does not use Java keystores. SAS Web Server requires the following:

- A private key that is in RSA format, is not protected with a passphrase, and is saved in the ASCII (Base64-encoded) PEM format
- A certificate that contains the server's public key in X.509 certificate form and is saved in ASCII (Base64-encoded) PEM format.

Note: An additional requirement might apply to SAS Web Server because of the way it communicates with HTTPS clients. If you are providing your own certificates, in addition to adding the CA root certificate, you must also add all CA intermediate certificates to the trusted CA bundle.

SAS Web Application Server

SAS Web Application Server was based on Pivotal tc Server prior to the Feb 15 2022 release of SAS 9.4. It is now based on Apache Tomcat. SAS Web Application Server uses a Java keystore to store its private key and server certificate.

Your deployment might include additional instances of the web application server. If your deployment includes a SAS solution, the web applications that are required for the solution might be deployed to managed servers with names like SASServer2_1 or SASServer12_1.

SAS Deployment Wizard does not include an option to automate HTTPS configuration for SAS Web Application Server. To implement certificates for communication with SAS Web Application Server, you must perform manual steps after running the deployment wizard. For more details, see [“Configure TLS for SAS Web Application Server Manually”](#) in *SAS Intelligence Platform: Middle-Tier Administration Guide*.

The Trusted CA Bundle

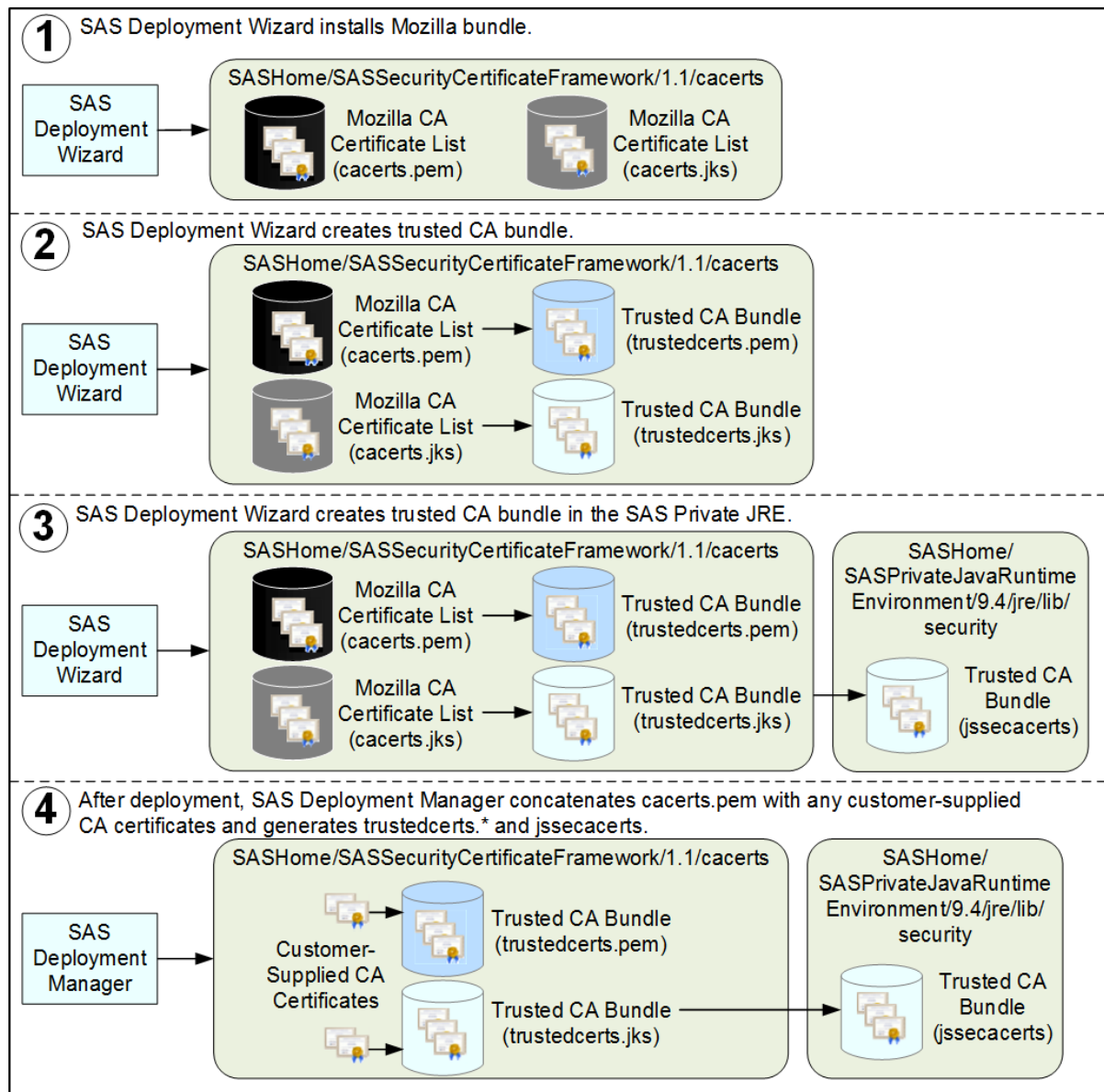
The Mozilla bundle is the list of CA certificates that are distributed with Mozilla software products. (For more information, see <https://wiki.mozilla.org/CA:IncludedCAs>.)

Starting with SAS 9.4M3, the Deployment Wizard copies the Mozilla bundle (cacerts.pem and cacerts.jks) to create the trusted CA bundle (trustedcerts.pem and trustedcerts.jks). These files reside in *SAS-installation-directory/SASSecurityCertificateFramework/1.1/cacerts*. On Windows and UNIX, SAS processes that are not Java use the trustedcerts.pem file to validate certificates.

The deployment wizard makes a copy of trustedcerts.jks called jssecacerts, and moves jssecacerts to *SAS-installation-directory/SASPrivateJavaRuntimeEnvironment/9.4/jre/lib/security*. SAS Java processes use the jssecacerts file to validate certificates. If this file resides in the same folder with cacerts, jssecacerts has precedence.

To add or remove CA certificates to or from the trusted CA bundle, you must use SAS Deployment Manager. For more information, see [“Managing Certificates in the SAS Environment”](#) on page 256.

Note: Prior to SAS 9.4M3, you must add your CA certificates to the SAS Private JRE using the `keytool -importcert` command. For more information, see [“Add Your Certificates to the SAS Private JRE”](#) in *Encryption in SAS*.

Figure 13.2 How SAS Deployment Wizard Installs the Trusted CA Bundle

Managing Certificates in the SAS Environment

Starting with SAS 9.4M3, SAS Deployment Manager enables you to add or remove CA root and intermediate certificates to or from the trusted CA bundle.

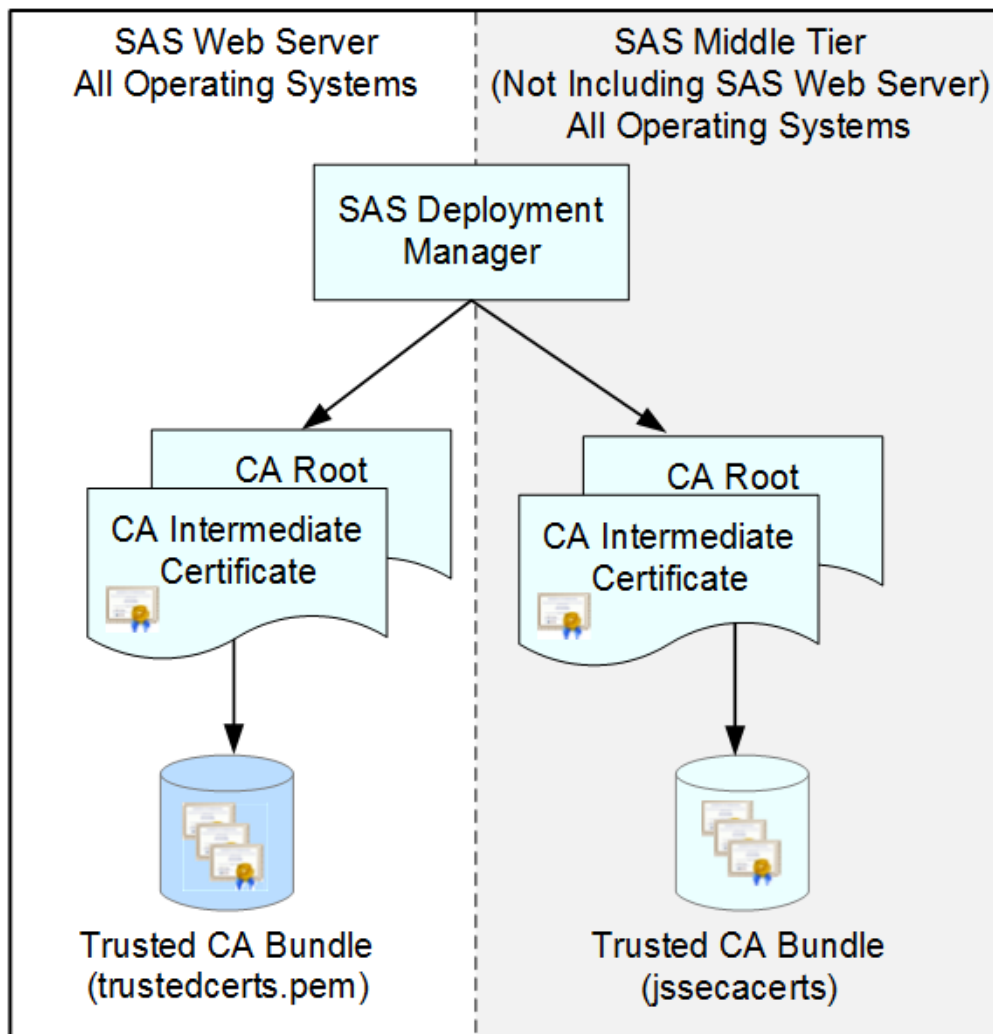
Note: Prior to SAS 9.4M3, you must add your CA certificates to the SAS Private JRE using the `keytool -importcert` command. For more information, see [“Add Your Certificates to the SAS Private JRE” in Encryption in SAS](#).

CAUTION

It is important to always perform the trusted CA bundle tasks in SAS Deployment Manager when adding or removing certificates to or from your SAS environment. Failing to perform these tasks in the deployment manager leads to security failures.

The following figure shows the process for adding certificates to the trusted CA bundle:

Figure 13.3 Adding Certificates to the Trusted CA Bundle



If you are providing your own certificates, there is an order in which you must follow. The order is important when you are first deploying your environment. First, install the SAS middle tier (if it is not already installed). Second, add your CA-signed root certificate and intermediate certificates to the trusted CA bundle. Then, configure the SAS middle tier. Your certificates must be in the trusted CA bundle when SAS Deployment Wizard loads specific products into SAS Content Server during the configuration phase of deployment. (The wizard's load content step acts as a client and sends requests through SAS Web Server. This step fails if SAS Web Server uses the trusted CA bundle and cannot establish trust of your self-signed or site-signed certificates.)

Web applications running in SAS Web Application Server act as clients of other web applications and direct requests to web applications on other instances of SAS Web Application Server. These requests are directed through SAS Web Server. Thus, each SAS Web Application Server instance must trust the certificates used by SAS Web Server. This trust is established by adding CA certificates to the trusted CA bundle.

For more information, see [“Add a Certificate to the Trusted CA Bundle” in Encryption in SAS](#).

For more information, see [“Add Your Certificates to the Windows CA Store” in Encryption in SAS](#).

Certificate Formats Supported by SAS Middle-Tier Components

The following table lists the SAS middle-tier components and the TLS certificate formats that they support:

Table 13.3 TLS Certificate Formats for SAS Middle-Tier Components

SAS Middle-Tier Component	Certificate Format Supported	CA Root and Intermediate Certificates Location
SAS Deployment Agent	Java keystore (One JKS file)	SAS Private JRE (jssecacerts)
SAS Environment Manager	Java keystore (One JKS file)	SAS Private JRE (jssecacerts)
SAS Web Server	RSA private key X.509 certificate (Two PEM files)	For SAS Intelligence Platform on Windows and UNIX: SAS Security Framework (trustedcerts.pem) For SAS Visual Analytics on Windows and Linux: SAS Security Framework (trustedcerts.pem)
SAS Web Application Server	Java keystore (One JKS/P12 file)	SAS Private JRE (jssecacerts)

Starting in [SAS9.4M9](#), you can also configure TLS for the SAS Cache Locator and the SAS JMS Broker in the middle tier. For more information, see [“Middle-Tier Servers and TLS” in SAS Intelligence Platform: Middle-Tier Administration Guide](#).

Obtaining Certificates

Overview of Obtaining Signed Certificates

The following SAS middle-tier components require one file—a Java keystore:

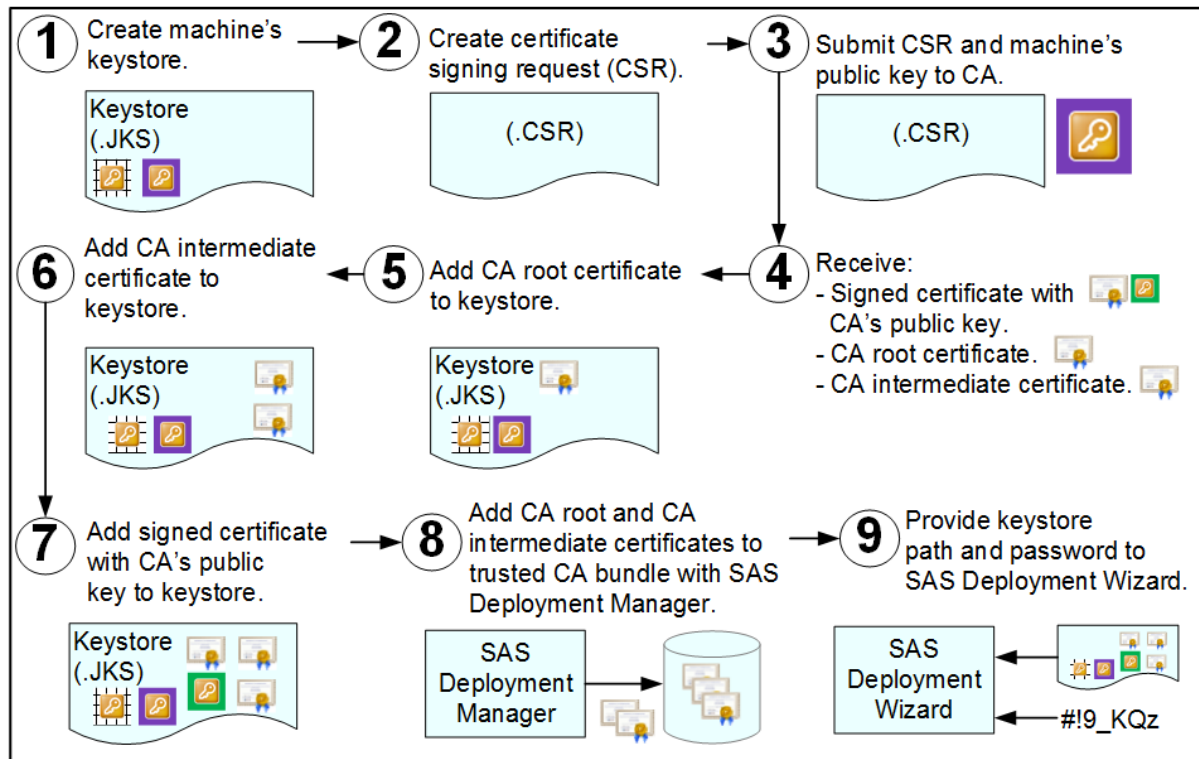
- SAS Deployment Agent
- SAS Environment Manager
- SAS Web Application Server

A Java keystore contains the following:

- private key
- CA root certificate
- CA intermediate certificates (if any)
- CA-signed certificate for the machine

The following figure summarizes the steps for obtaining a site-signed or third-party-signed certificate for a SAS middle-tier component (other than SAS Web Server):

Figure 13.4 Steps for Obtaining a Site-Signed or Third-Party-Signed Certificate for a SAS Middle-Tier Component



SAS Web Server requires two files:

- private key

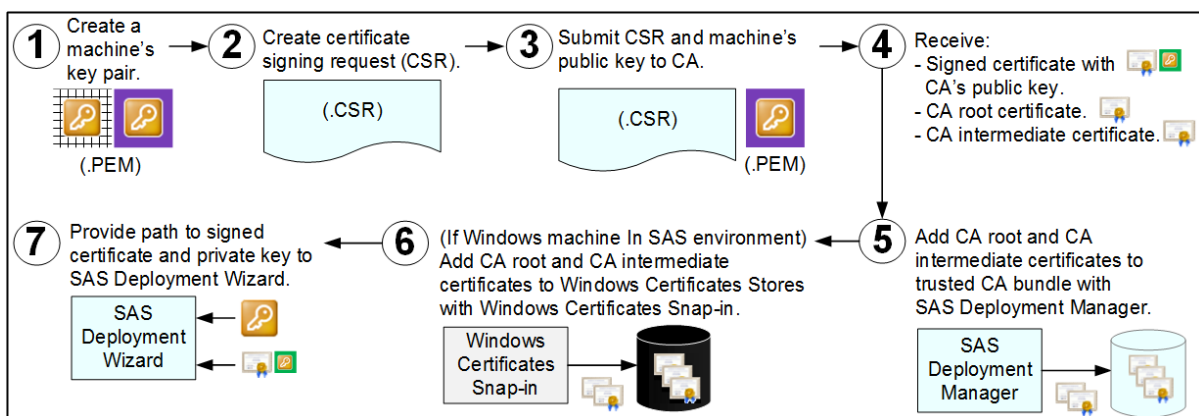
This private key is in RSA format, is not protected with a passphrase, and is saved in the ASCII (Base64-encoded) PEM format.

- CA-signed certificate

This certificate contains the server's public key in X.509 certificate form and is saved in ASCII (Base64-encoded) PEM format.

The following figure summarizes the steps for obtaining a site-signed or third-party-signed certificate for SAS Web Server:

Figure 13.5 Steps for Obtaining a Site-Signed or Third-Party-Signed Certificate for SAS Web Server



How Many Private Keys and Signed Certificates Do I Need?

If you plan to provide your own signed certificates, you need to know how many to provide.

Each middle-tier machine requires at least one private key and one signed certificate to share between the SAS middle-tier components.

Except for SAS Web Server, the SAS middle-tier components require a private key and signed certificate in a Java keystore. SAS Web Server requires a private key and signed certificate in PEM format. For this reason, if SAS Web Server is located on the same machine with one or more SAS middle-tier components, you might find it easier to create a separate private key and signed certificate for SAS Web Server.

Note: If you want to use only one private key and signed certificate per machine, you must convert the signed certificate to the proper format. For more information, see https://docs.oracle.com/cd/E35976_01/server.740/es_admin/src/tadm_ssl_convert_pem_to_jks.html.

For example, if you are implementing the middle tier on machine A and the server tier on machine B, then you would require a total of three signed certificates: 1cert_machA.jks, 2cert_machB.jks, and 3cert_machB.pem.

Table 13.4 A List of Machines and Their Certificates in a Sample SAS Middle-Tier Deployment

SAS Components	Component Location	Signed Certificate File
SAS Metadata Server	Machine A	Not applicable
SAS Deployment Agent	Machine A	1cert_machA.jks
SAS Environment Manager Agent	Machine A	1cert_machA.jks
SAS Server Tier	Machine B	Not applicable
SAS Deployment Agent	Machine B	2cert_machB.jks
SAS Environment Manager Server	Machine B	2cert_machB.jks
SAS Web Application Server	Machine B	2cert_machB.jks

SAS Components	Component Location	Signed Certificate File
SAS Web Server	Machine B	3cert_machB.pem

Implementing Certificates

The options for implementing certificates consist of the following:

- self-signed certificates automatically generated by SAS Deployment Wizard
- certificates that you provide
- combination of automatically generated certificates and provided certificates

The following table lists the certificate options for the SAS middle-tier components:

Table 13.5 *Certificate Options for the SAS Middle-Tier Components*

Middle-Tier Component	Certificate Source		
	Auto-Generated	You Provide	Combination
SAS Deployment Agent	Self-signed only	Any type ¹	Self-signed ² or Any type
SAS Environment Manager	Self-signed only	Any type	Self-signed ² or Any type
SAS Web Server	None	Any type	Any type
SAS Web Application Server	None	Any type	Any type

¹ All instances of “Any type” refers to self-signed, site-signed, and third-party-signed certificates that you provide.

² Self-signed certificate auto-generated by SAS Deployment Wizard.

Starting in [SAS9.4M9](#), you can also configure TLS for the SAS Cache Locator and the SAS JMS Broker in the middle tier. For more information, see “[Middle-Tier Servers and TLS](#)” in *SAS Intelligence Platform: Middle-Tier Administration Guide*.

For more information, see [“Implementing Certificates”](#) on page 298.

Encryption Tasks

<i>How to Maximize Encryption of Stored Passwords</i>	266
Passwords in Login and SASPassword Objects	266
Passwords for Internal Accounts	267
Passwords in Configuration Files	267
See Also	267
<i>How to Re-encrypt Stored Passwords</i>	268
Introduction	268
Set or Change a Passphrase (SAS Deployment Manager)	268
Set or Change a Passphrase (METADATA Procedure)	269
Downgrade Stored Passwords	270
How Password Re-encryption Activity Is Logged	272
See Also	272
<i>How to Configure TLS for IOM and Other SAS Servers and Change the Encryption Level</i>	273
Automatic Configuration Prior to SAS 9.4M9	273
Manual Configuration for IOM and Other SAS Servers Starting with SAS 9.4M9	274
Configure TLS Using System Options in Server Configuration Files	
for Windows and Linux	275
Update Metadata Definitions for Servers	282
About NETENCALG and Client Encryption Level (CEL)	283
See Also	285
<i>Configure the SAS Intelligence Platform Server Tier to Use IBM System SSL on z/OS</i>	285
Overview of Using IBM System SSL on z/OS	285
How To Configure TLS Using System Options in Server	
Configuration Files for z/OS	286
Use SAS Management Console to update the Metadata Server	289
Use SAS Deployment Manager to Add Certificates to the SAS Deployment	289
(Optional) Configure GSK Logging	289
Configuration Examples for Accessing System SSL Certificates	290
<i>How to Change Encoding for Data in Motion for Outbound Passwords</i>	291
Outbound Passwords	291
Instructions	292
Compatibility	292
<i>How to Configure TLS between the Metadata Server and an LDAP Server</i>	294
Before You Begin	294
Configuration Instructions	294
Connecting to an LDAP Load Balancer	296

SASSSLREQCERT	297
See Also	298
How to Implement Certificates	298
Implementing Certificates	298
Using Auto-generated Certificates	299
Providing Your Own Certificates	300
Using a Combination of Auto-generated and Your Own Certificates	305
How to Enable IOM Communication over HTTPS Proxy	305
Introduction	305
Configuring the HTTPS Proxy	306
Connecting to an IOM Server via Proxy While Using an Environment Variable ...	307
Registering Proxy Information in SAS Metadata Server for Other Servers	308

How to Maximize Encryption of Stored Passwords

Note: Starting with [SAS 9.4M8](#), SAS uses the cryptographic libraries that are available from the operating system instead of the libraries that were delivered by SAS/SECURE for SAS 9.4 Foundation servers. The [required system cryptographic libraries](#) must be present on the host operating system in order for your SAS deployment to use passwords that are stored for the following states:

- **Data at Rest:** Host cryptographic libraries are used to decrypt and verify any passwords that are stored using AES-based encoding (SAS003, SAS004, and SAS005).
 - **Data in Motion:** Host cryptographic libraries are used to establish the TLS handshake that is required to encrypt the communication channel between SAS clients and servers. The connection will fail before the password attempts to travel over the wire if the cryptographic libraries are unavailable on the host operating system.
-

Passwords in Login and SASPassword Objects

Take the following steps in order to use the strongest available encryption for passwords that are stored in metadata:

- 1 Configure SAS Metadata Server to store any new or updated passwords using AES encryption. SAS005 is the highest encoding value for AES, with a salt size of 64 bits and additional hash iterations.
 - a Locate your equivalent of `SAS/Config/Lev1/SASMeta/MetadataServer/omaconfig.xml` and open the file for edit.

TIP It is a good practice to make a backup copy of the file before you modify it.

- b Set `STOREPASSWORDS="SAS005"`.
 - c Restart the Metadata Server.
- 2 (Optional) Convert currently stored passwords to the SAS005 format. See [“How to Re-encrypt Stored Passwords” on page 268](#).
 - 3 (Optional) Supply a passphrase to use as the encryption key, rather than using the default fixed key. See [“How to Re-encrypt Stored Passwords” on page 268](#).

Passwords for Internal Accounts

To use SHA256-10000 hashing for internal account passwords, set the HashPasswords option in the Metadata Server's omaconfig.xml file. See [“How to Change Internal Account Policies” on page 239](#).

Passwords in Configuration Files

To use the strongest available encoding for passwords that are stored in configuration files:

- 1 Use the PWENCODE procedure and specify SAS005 for the highest encoding. When using SAS005, SAS protects each managed password by using AES-256 with a 64-bit salt and additional iterations of a SHA-256 hash.
- 2 Use the SAS Deployment Manager password update utility to update the deployment with the SAS005-encoded version of the managed passwords. See [“About Updating Managed Passwords” on page 44](#).

Note: Some limitations affect outbound passwords. For more information, see [“Compatibility” on page 292](#).

See Also

[“Reference Information for omaconfig.xml” in SAS Intelligence Platform: System Administration Guide](#)

How to Re-encrypt Stored Passwords

Introduction

An unrestricted user can use the SAS Deployment Manager or the METADATA procedure to change how stored passwords are encrypted. These methods provide a thorough, secure, and efficient approach for an unrestricted user to address situations such as the following:

- For increased security, you want to re-encrypt stored passwords using a passphrase as the encryption key, rather than using the default fixed key.
- You suspect that one or more stored passwords have been compromised, so you want to re-encrypt using a new passphrase.
- You have upgraded the encryption type with which new or updated passwords are stored, and you also want to upgrade the format of currently stored passwords.
- You want to downgrade the format of currently stored passwords.

CAUTION

When you use the following instructions to introduce, change, or clear a passphrase, all passwords in logon objects within the specified Metadata Server are affected. Make sure you have a current backup before you set a passphrase.

Set or Change a Passphrase (SAS Deployment Manager)

To change a passphrase using the SAS Deployment Manager:

- 1 On the Metadata Server host, navigate to your equivalent of `SAS-installation-directory/SASDeploymentManager/9.4/` and launch `sasdm.exe` (Windows), `sasdm.sh` (UNIX), or `sasdm.rexx` (z/OS).

Windows Specifics: On Windows, log on as a Windows administrator of the current machine.

Operating Environment Information: On UNIX and z/OS, log on with the user ID that was used to perform the SAS installation.

- 2 In the SAS Deployment Manager, select the change passphrase task and log on as an unrestricted user.
- 3 Change the passphrase. For assistance with the user interface, see the Help within the application.

Set or Change a Passphrase (METADATA Procedure)

Note: Starting with SAS 9.4M8, SAS uses the cryptographic libraries available from the operating system instead of those that were delivered by SAS/SECURE for SAS 9.4 Foundation servers. You must install the [required cryptographic libraries](#) on the operating system in order to use passwords that are stored with AES encryption (SAS003, SAS004, and SAS005).

To use a passphrase as the encryption key for passwords that are stored in metadata:

- 1 Verify that the Metadata Server is configured to store passwords in SAS005, SAS004, or SAS003 format. Passphrase keys are supported for only these formats.

Note: The format in which the Metadata Server stores passwords is controlled by the STOREPASSWORDS="SAS00n" setting in your equivalent of **SAS/Config/Lev1/SASMeta/MetadataServer/omacfg.xml**. Changes to this file take effect when you restart the Metadata Server.

For more information about the omacfg.xml file, see ["Reference Information for omacfg.xml" in SAS Intelligence Platform: System Administration Guide](#).

- 2 In a SAS session, connect to the Metadata Server as an unrestricted user and pause the server for administration as follows.

```
options
  metaserver='machine.company.com'
  metaport=8651
  metauser='sasadm@saspw'
  metapass='secret';
proc metaoperate
  action=PAUSE
  options='<Server STATE="ADMIN"/>';
run;
```

- 3 In the SAS session, set a passphrase as follows:

```
proc metadata
  in="<ChangePassPhrase>
    <PassPhrase>My passphrase</PassPhrase>
  </ChangePassPhrase>";
run;
```

The preceding code decrypts currently stored passwords and then re-encrypts them using the passphrase that you supply.

TIP You can use the PWENCODE procedure to prepare an encrypted version of your passphrase, and supply the encrypted version of the passphrase in the preceding code.

- 4 To verify results, inspect the Metadata Server log. See [“How Password Re-encryption Activity Is Logged”](#) on page 272.
- 5 In the SAS session, resume the Metadata Server.

```
proc metaoperate
  action=RESUME;
run;
```

To re-encrypt stored passwords using a new passphrase, follow the preceding instructions and supply the new passphrase in step 3 as follows:

```
proc metadata
  in=<ChangePassPhrase>
    <PassPhrase>My new passphrase</PassPhrase>
  </ChangePassPhrase>;
run;
```

To remove a passphrase (and use the default fixed key instead), follow the preceding instructions and submit a blank passphrase in step 3 as follows:

```
proc metadata
  in=<ChangePassPhrase>
    <PassPhrase/>
  </ChangePassPhrase>;
run;
```

If you must maintain a few stored passwords in SAS002 format, see [“Compatibility”](#) on page 292.

Downgrade Stored Passwords

To downgrade the encryption type of stored passwords:

- 1 Configure the Metadata Server to store any new or updated passwords in the downgraded format.
 - a Locate your equivalent of `SAS/Config/Lev1/SASMeta/MetadataServer/omaconfig.xml` and open the file for edit.

TIP It is a good practice to make a copy of the file before you modify it.

- b Set the following options, which specify the lowest possible downgrade option values:

- STOREPASSWORDS="SAS002"
- HashPasswords="MD5"

Note: The latter setting is within the InternalAuthenticationPolicy section of the omaconfig.xml file. It affects passwords in internal accounts. For more information, see [“How to Change Internal Account Policies” on page 239](#).

- c Restart the Metadata Server.

- 2 In a SAS session, connect to the Metadata Server as an unrestricted user and pause the server for administration as follows:

```
options
  metaserver='machine.company.com'
  metaport=8651
  metauser='sasadm@saspw'
  metapass='secret';
proc metaoperate
  action=PAUSE
  options='<Server STATE="ADMIN"/>';
run;
```

- 3 In your SAS session, set a blank pass phrase as follows:

```
proc metadata
  in="<ChangePassPhrase>
    <PassPhrase/>
  </ChangePassPhrase>";
run;
```

Notice that the PassPhrase element is empty. The preceding code decrypts stored passwords and then stores them in a downgraded format.

- 4 To verify results, inspect the Metadata Server log. See [“How Password Re-encryption Activity Is Logged” on page 272](#).

- 5 Resume the Metadata Server as follows:

```
proc metaoperate
  action=RESUME;
run;
```

- 6 Update the password in each internal account as follows:

- For a managed internal account, use SAS Deployment Manager. See [“About Updating Managed Passwords” on page 44](#).
- For other internal accounts, use SAS Management Console. For more information, see “Reset an Internal Password” in [SAS Management Console: Guide to Users and Permissions](#).

How Password Re-encryption Activity Is Logged

As the passwords within each repository are converted, repository-level messages are written to the SAS Metadata Server log. Here are some examples of the messages:

```
ChangePassPhrase: repository Id=A5KUG99E, Name=repos1, PauseState=ADMIN is being processed.
ChangePassPhrase: repository Id=A5KUG98E, Name=repos2 is not being processed because the
PauseState=OFFLINE must be ADMIN.
```

To log changes to individual objects, add the following content to the Metadata Server's logconfig.xml file:

```
<logger name="Audit.Meta.Updates.PublicObjects">
  <level value="Trace"/>
</logger>
<logger name="Audit.Meta.Updates.Projects.PublicObjects">
  <level value="Debug"/>
</logger>
```

Here are some examples of the log messages:

```
Audit Public Object Type=Login Name=user2 ObjId=A5KUG99E.AQ000001 has been updated.
Audit Public Object Type=Login Name=user3 ObjId=A5KUG99E.AQ000002 has been updated.
Audit Public Object Type=Login Name=sasadm ObjId=A5KUG99E.AQ000003 has been updated.
```

.....
Note: The log messages are written after the objects are committed to the repository. Changes to SASPassword objects are not logged.
.....

See Also

- “Connection Options” in *SAS Language Interfaces to Metadata*
- “METAOPERATE Procedure” in *SAS Language Interfaces to Metadata*
- “About Metadata Server Loggers” in *SAS Intelligence Platform: System Administration Guide*
- “Reference Information for omaconfig.xml” in *SAS Intelligence Platform: System Administration Guide*
- “PWENCODE Procedure” in *Base SAS Procedures Guide*

How to Configure TLS for IOM and Other SAS Servers and Change the Encryption Level

Automatic Configuration Prior to SAS 9.4M9

When you install the SAS Metadata Server, you select an encryption level (which traffic content is encrypted) and an encryption algorithm (how that traffic is encrypted). In order to connect to the Metadata Server, clients must conform to the Metadata Server's encryption requirements.

Note: If the FIPS encryption algorithm was set during the initial deployment and you subsequently manually change the server encryption algorithm, you must also remove the `-encryptfips` option from the following files:

- ObjectSpawner.bat (Windows) or ObjectSpawner.sh (UNIX)
- ConnectSpawner.bat (Windows) or ConnectSpawner.sh (UNIX)

For more information, see [“ENCRYPTFIPS System Option” in *Encryption in SAS*](#).

IMPORTANT Starting with SAS 9.4M9, SAS has deprecated the NETENCRYPTALGORITHM (NETENCRALG) system option values AES, DES, RC2, RC4, and TripleDES. For SAS 9.4M7 and SAS 9.4M8, this change is made with hot fixes related to [KB0041538](#). SAS highly recommends that you change the value to SSL to specify the use of the TLS protocol. For more information, see [“TLS Support for IOM and Other SAS Servers” in *SAS Intelligence Platform: Application Server Administration Guide*](#).

To change encryption settings for data in motion after the installation has completed, you must update security properties for SAS servers in two locations:

- `*_usermods` files. For more information, see [“Configure TLS Using System Options in Server Configuration Files for Windows and Linux” on page 275](#).
- SAS Metadata Server through SAS Management Console. For more information, see [“Update Metadata Definitions for Servers” on page 282](#).

Manual Configuration for IOM and Other SAS Servers Starting with SAS 9.4M9

You can select an encryption level when you install the SAS Metadata Server. This setting determines which traffic is encrypted, either **Credentials** or **Everything**. However, your only option for the encryption algorithm is **SAS Proprietary**, which determines how the selected traffic is encrypted. An example of the SAS Deployment Wizard dialog box where these options are set is provided in “[Server Encryption](#)” in *SAS Intelligence Platform: Installation and Configuration Guide*.

SAS strongly recommends that you enable TLS after the deployment has completed.

IMPORTANT Starting with SAS 9.4M9, SAS has deprecated the NETENCRYPTALGORITHM (NETENCALG) system option values AES, DES, RC2, RC4, and TripleDES. For SAS 9.4M7 and SAS 9.4M8, this change is made with hot fixes related to [KB0041538](#). SAS highly recommends that you change the value to SSL to specify the use of the TLS protocol. For more information, see “[TLS Support for IOM and Other SAS Servers](#)” in *SAS Intelligence Platform: Application Server Administration Guide*.

Note: Configuring FIPS is a post-deployment task. For more information, see “[Configure FIPS](#)” in *Configuring TLS and Enabling FIPS*. For more information about the -encryptfips option, see “[ENCRYPTFIPS System Option](#)” in *Encryption in SAS*.

To change encryption settings for data in motion after the installation has completed, you must update security properties for SAS servers in the following locations:

- *_usermods files on Linux and Windows. See “[Configure TLS Using System Options in Server Configuration Files for Windows and Linux](#)” on page 275.
- *_usermods files for z/OS. See “[Configure the SAS Intelligence Platform Server Tier to Use IBM System SSL on z/OS](#)” on page 285.
- SAS Metadata Server through SAS Management Console. See “[Update Metadata Definitions for Servers](#)” on page 282.

Configure TLS Using System Options in Server Configuration Files for Windows and Linux

IMPORTANT Starting with SAS 9.4M9, SAS has deprecated the NETENCRYPTALGORITHM (NETENCRLG) system option values AES, DES, RC2, RC4, and TripleDES. For SAS 9.4M7 and SAS 9.4M8, this change is made with hot fixes related to [KB0041538](#). SAS highly recommends that you change the value to SSL to specify the use of the TLS protocol. For more information, see “TLS Support for IOM and Other SAS Servers” in *SAS Intelligence Platform: Application Server Administration Guide*.

Each server has a *_usermods file in its configuration directory. Use this file to update or add new options. Option settings in this file override settings in the start-up configuration files, and they are not overwritten when you perform updates. Note that servers typically have more than one *_usermods file. This depends on the directory level and location of the file. To understand the order in which the options in the file are applied, see “Reference: Configuration Files for SAS Servers” in *SAS Intelligence Platform: System Administration Guide*.

The information in this section provides the location of each configuration file and describes the options to use for the TLS protocol.

Note: To change the encryption value to something other than SSL, add only the following option to the configuration files: -netencrlg "encryptionAlgorithm".

- 1 Stop your SAS servers. For information, see “Operating Your Servers” in *SAS Intelligence Platform: System Administration Guide*.
- 2 For prerequisite information about how to generate certificates and private keys, see the following information:
 - [Configure certificates and the private key on Linux](#)
 - [Configure certificates and the private key on Windows](#)
- 3 Use the tables below for the location of the configuration files and the options to add or modify to configure TLS.

Table 14.1 For Linux: Location of Configuration Files

SAS Metadata Server

`$SASConfig/Levn/SASMeta/MetadataServer/sasv9_usermods.cfg`

Add the following options:

`-netencryptalgorithm "SSL"`

`-sslcertloc $SASHOME/SASSecurityCertificateFramework/x.x/servercert.pem`

SAS Metadata Server

```
-sslpvtkeyloc Cert_Home/serverkey.pem
-objectserverparms "cel=everything other-parameters"
```

Notes:

- `sslcertloc` points to the certificate that was created for this system.
- `sslpvtkeyloc` points to the private key that was created for the certificate.
- If a password or pass phrase is used with the private key, the `-sslpvtkeypass` option must also be added to the file.
- To change the client encryption level, copy the entire `OBJECTSERVERPARMS` line from the `sasv9.cfg` file into the `sasv9_usermods.cfg` file. Then edit the `CEL` value in the `usermods` version of the file. For more information, see [“OBJECTSERVERPARMS System Option” in SAS Intelligence Platform: Application Server Administration Guide](#) and [“About NETENCALG and Client Encryption Level \(CEL\)”](#).
- If other TLS-related parameters are needed, they must also be set here. For example, if you are configuring FIPS, you must also specify the `ENCRYPTFIPS` system option in the `_usermods.cfg` file. For more information, see [“SAS System Options for Encryption” in Encryption in SAS](#).

SAS Metadata Server

SASConfig/Levn/SASMeta/MetadataServer/sasv9_nocluster.cfg

Verify that the `CEL` value of the `OBJECTSERVERPARMS` line is set to `everything` in this file. Change it if necessary.

TIP To change the encryption level, copy the entire `OBJECTSERVERPARMS` line from the `sasv9.cfg` file into the `sasv9_usermods.cfg` file. Then edit the `CEL` value in the `usermods` version of the file.

SAS Object Spawner

SASConfig/Levn/ObjectSpawner/ObjectSpawner_usermods.sh

Add the following options:

```
CERT_HOME=${SAS_HOME}/SASSecurityCertificateFramework/1.1
CERT_LOC=${CERT_HOME}/server.pem
CERT_KEY=${CERT_HOME}/serverkey.pem
CALIST_LOC=${CERT_HOME}/cacerts/trustedcerts.pem
USERMOD_OPTIONS="${JREOPTIONS} -sslpvtkeyloc ${CERT_KEY} -sslcertloc ${CERT_LOC} -sslcalistloc $
{CALIST_LOC} -netencryptalgorithm "SSL"
```

- If a password or pass phrase is used with the private key, the `-sslpvtkeypass` option must be added to the `OPTIONS` string above.
- If other TLS-related parameters are required, you must set them here also. For example, if you are configuring FIPS, you must also specify the `ENCRYPTFIPS` system option in the `_usermods.cfg` file. For more information, see [“SAS System Options for Encryption” in Encryption in SAS](#).

SAS Object Spawner

Note: The SAS Workspace Server, SAS Pooled Workspace Server, and SAS Stored Process Server automatically pick up the required TLS parameters from the Object Spawner.

SAS/CONNECT Spawner

`SASConfig/Levn/ConnectSpawner/ConnectSpawner_usermods.sh`

IMPORTANT When SAS 9.4M7, SAS 9.4M8, or SAS 9.4M9 hot fixes for [KB0044144](#) are applied, the SAS/CONNECT server inherits the system options for configuring TLS from the SAS/CONNECT spawner.

Add the following options:

```
CERT_HOME=${SAS_HOME}/SASSecurityCertificateFramework/1.1
CERT_LOC=${CERT_HOME}/server.pem
CERT_KEY=${CERT_HOME}/serverkey.pem
USERMOD_OPTIONS="{JREOPTIONS} -sslpvtkeyloc ${CERT_KEY} -sslcertloc ${CERT_LOC} -sslcalistloc ${CALIST_LOC} -netencryptalgorithm "SSL""
```

IMPORTANT To enable TLS for SAS/CONNECT, you must complete all of the following steps:

- [Configure options for TLS on the SAS/CONNECT spawner](#). When SAS 9.4M7, SAS 9.4M8, or SAS 9.4M9 hot fixes for [KB0044144](#) are not applied, in the ConnectSpawner_usermods.sh file, remove system option NETENCRYPTALGORITHM.
- [Configure options for TLS on the SAS/CONNECT server](#). Skip this step when hot fixes for [KB0044144](#) are applied.
- Use [SAS Management Console on page 282](#) to update the metadata definition for the SAS/CONNECT server by using encryption value SSL. With SAS 9.4M7, the SSL value is available by applying the hot fix for [KB0041222](#).

SAS/CONNECT Server

`SASConfig/Levn/SASApp/ConnectServer/ConnectServer_usermods.sh`

Note: You can also set the following options in `SASConfig/Levn/SASApp/ConnectServer/sas9_usermods.cfg`. Where you set these options depends on your specific SAS version and how the server is being invoked. If you use `sas9_usermods.cfg`, replace `USERMODS_OPTIONS=` with `USERMODS=` in the following code.

IMPORTANT When SAS 9.4M7, SAS 9.4M8, or SAS 9.4M9 hot fixes for [KB0044144](#) are applied, the SAS/CONNECT server inherits the system options for configuring TLS from the SAS/CONNECT spawner.

Add the following options:

```
CERT_HOME=${SAS_HOME}/SASSecurityCertificateFramework/1.1
CERT_LOC=${CERT_HOME}/server.pem
```

SAS/CONNECT Server

```
CERT_KEY=${CERT_HOME}/serverkey.pem
CALIST_LOC=${CERT_HOME}/cacerts/trustedcerts.pem
USERMODS_OPTIONS="${JREOPTIONS} -sslpvtkeyloc ${CERT_KEY} -sslcertloc ${CERT_LOC} -sslcalistloc ${CALIST_LOC} -netencryptalgorithm "ssl"
```

IMPORTANT To enable TLS for SAS/CONNECT, you must complete all of the following steps.

- [Configure options for TLS on the SAS/CONNECT spawner](#) . When [SAS 9.4M7](#), [SAS 9.4M8](#), or [SAS 9.4M9](#) hot fixes for [KB0044144](#) are not applied, in the `ConnectSpawner_usermods.sh` file, remove system option `NETENCRYPTALGORITHM`.
- [Configure options for TLS on the SAS/CONNECT server](#). Skip this step when hot fixes for [KB0044144](#) are applied.
- Use [SAS Management Console on page 282](#) to update the metadata definition for the SAS/CONNECT server by using encryption value `SSL`. With [SAS 9.4M7](#), the `SSL` value is available by applying the hot fix for [KB0041222](#).

(Optional) SAS OLAP Server

SASConfig/Levn/SASApp/OLAPServer/sasv9_usermods.cfg

Use the same security options as SAS Metadata Server.

Note: If your deployment offers direct connections from clients to the OLAP server, make the same updates to the server's configuration file.

SAS/SHARE Server

SASConfig/Levn/ShareServer/sasv9_usermods.cfg

Add the following options:

```
CERT_HOME=${SAS_HOME}/SASSecurityCertificateFramework/1.1
CERT_LOC=${CERT_HOME}/server.pem
CERT_KEY=${CERT_HOME}/serverkey.pem
USERMODS="${JREOPTIONS} -sslpvtkeyloc ${CERT_KEY} -sslcertloc ${CERT_LOC} -netencryptalgorithm "SSL"
```

- If a password or pass phrase is used with the private key, the `-sslpvtkeypass` option must be added to the `OPTIONS` string above.
- If other TLS-related parameters are required, you must set them here also. For example, if you are configuring FIPS, you must specify the `ENCRYPTFIPS` system option in the `_usermods.cfg` file.

For more information, see [“SAS System Options for Encryption” in Encryption in SAS](#).

Table 14.2 For Windows: Setting Up TLS Certificates

To ensure that your TLS certificates work correctly with servers and services, you must make the certificates available. To do so, use the Windows Certificates Snap-in to add the CA root certificate and all of its intermediate certificates to the Windows certificate store. Perform the steps in [“Add Your Certificates to the Windows CA Store”](#) in *Encryption in SAS*. After you perform the recommended steps, servers on this machine will be able to access and use the certificates to allow TLS connections.

Note: You must provide either the `SSLCERTSUBJ=` system option or system options `SSLCERTISS=` and `SSLCERTSERIAL=` to configure SAS to use Windows certificates. The `SSLCERTISS=` and `SSLCERTSERIAL=` options are used first to locate the certificate in the Windows certificate store. Using these two system options is the most reliable way to locate the certificate that is needed. If no certificate match is found, then the `SSLCERTSUBJ=` option information is used to locate the certificate.

Table 14.3 For Windows: Location of Configuration Files

SAS Metadata Server

`SASConfig\Levn\SASMeta\MetadataServer\sasv9_usermods.cfg`

Add the following options. You must provide either `-sslcertsubj` or both `-sslcertiss` and `-sslcertserial`.

```
-netencryptalgorithm "SSL"
-sslcertiss common-name
-sslcertserial serialNumber_of_certificate_for_publicKey
-sslcertsubj common-name
-objectserverparms "cel=everything other-parameters"
```

Notes:

- To change the encryption level, copy the entire `OBJECTSERVERPARMS` line from the `sasv9.cfg` file into the `sasv9_usermods.cfg` file. Next, edit the `CEL` value in the `usermods` version of the file. For more information, see [“OBJECTSERVERPARMS System Option”](#) in *SAS Intelligence Platform: Application Server Administration Guide* and [“About NETENCALG and Client Encryption Level \(CEL\)”](#).
- If other TLS-related parameters are needed, they must also be set here. For example, if you are configuring FIPS, you must also specify the `ENCRYPTFIPS` system option in the `_usermods.cfg` file. For more information, see [“SAS System Options for Encryption”](#) in *Encryption in SAS*.

SAS Metadata Server

`SASConfig\Levn\SASMeta\MetadataServer\sasv9_nocluster.cfg`

Verify that the `CEL` value of the `OBJECTSERVERPARMS` line is everything in this file. Change it if necessary. For more information, see [“OBJECTSERVERPARMS System Option”](#) in *SAS Intelligence Platform: Application Server Administration Guide*.

TIP To change the encryption level, copy the entire `OBJECTSERVERPARMS` line from the `sasv9.cfg` file into the `sasv9_usermods.cfg` file. Then edit the `CEL` value in the `usermods` version of the file.

SAS Object Spawner

`SASConfig\Levn\ObjectSpawner\ObjectSpawner_usermods.bat`

You must provide either `-sslcertsubj` or both `-sslcertiss` and `-sslcertserial`:

```
SET USERMOD_OPTIONS=%JREOPTIONS% -sslcertsubj "common-name" -netencryptalgorithm "ssl"
```

or:

```
SET USERMOD_OPTIONS=%JREOPTIONS% -sslcertiss "common-name" -sslcertserial "serial_number" -
netencryptalgorithm "ssl"
```

Here is an example where the CN=server.company.com.

```
SET USERMOD_OPTIONS=%JREOPTIONS% -sslcertiss "server.company.com" -sslcertserial "serial_number" -
netencryptalgorithm "ssl"
```

Note: If other TLS-related parameters are needed, they must also be set here. For example, if you are configuring FIPS, you must also specify the `ENCRYPTFIPS` system option in the `_usermods.cfg` file.

For more information, see [“SAS System Options for Encryption” in *Encryption in SAS*](#).

Note: The SAS Workspace Server, SAS Pooled Workspace Server, and SAS Stored Process Server automatically pick up the needed TLS parameters from the Object Spawner.

SAS Connect Spawner

`SASConfig\Levn\ConnectSpawner\ConnectSpawner_usermods.bat`

IMPORTANT When SAS 9.4M7, SAS 9.4M8, or SAS 9.4M9 hot fixes for [KB0044144](#) are applied, the SAS/CONNECT server inherits the system options for configuring TLS from the SAS/CONNECT spawner.

You must provide either `-sslcertsubj` or both `-sslcertiss` and `-sslcertserial`:

```
SET USERMODS_OPTIONS=%JREOPTIONS% -sslcertsubj "common-name" -netencryptalgorithm "ssl"
```

or:

```
SET USERMODS_OPTIONS=%JREOPTIONS% -sslcertiss "common-name" -sslcertserial "serial_number" -
netencryptalgorithm "ssl"
```

Here is an example where CN=server.company.com.

```
SET USERMODS_OPTIONS=%JREOPTIONS% -sslcertiss "server.company.com" -sslcertserial "serial_number" -
netencryptalgorithm "ssl"
```

IMPORTANT To enable TLS for SAS/CONNECT, you must complete all of the following steps:

- **Configure options for TLS on the SAS/CONNECT spawner.** When [SAS 9.4M7](#), [SAS 9.4M8](#), or [SAS 9.4M9](#) hot fixes for [KB0044144](#) are not applied, in the `ConnectSpawner_usermods.bat` file, remove system option `NETENCRYPTALGORITHM`.
- **Configure options for TLS on the SAS/CONNECT server.** Skip this step when hot fixes for [KB0044144](#) are applied.
- Use [SAS Management Console on page 282](#) to update the metadata definition for the SAS/CONNECT server by using encryption value `SSL`.

SAS Connect Spawner

With [SAS 9.4M7](#), the SSL value is available by applying the hot fix for [KB0041222](#).

SAS Connect Server

SASConfig\Levn\SASApp\ConnectServer\connectServer_usermods.bat

Note: You can also set the following options in *SASConfig\Levn\SASApp\ConnectServer\sasv9_usermods.cfg*. Where you set these options depends on your specific SAS version and how the server is being invoked. If you use *sas9_usermods.cfg*, replace `USERMODS_OPTIONS=` with `USERMODS=` in the following code.

IMPORTANT When [SAS 9.4M7](#), [SAS 9.4M8](#), or [SAS 9.4M9](#) hot fixes for [KB0044144](#) are applied, the SAS/CONNECT server inherits the system options for configuring TLS from the SAS/CONNECT spawner.

You must provide either `-sslcertsubj` or both `-sslcertiss` and `-sslcertserial`:

```
SET USERMODS_OPTIONS=%JREOPTIONS% -sslcertsubj "common-name" -netencryptalgorithm "ssl"
```

or:

```
SET USERMODS_OPTIONS=%JREOPTIONS% -sslcertiss "common-name" -sslcertserial "serial_number" -netencryptalgorithm "ssl"
```

Here is an example where CN=server.company.com:

```
SET USERMODS_OPTIONS=%JREOPTIONS% -sslcertiss "server.company.com" -sslcertserial "serialNumber_number" -netencryptalgorithm "ssl"
```

IMPORTANT To enable TLS for SAS/CONNECT, you must complete all of the following steps.

- [Configure options for TLS on the SAS/CONNECT spawner](#). When [SAS 9.4M7](#), [SAS 9.4M8](#), or [SAS 9.4M9](#) hot fixes for [KB0044144](#) are not applied, in the *ConnectSpawner_usermods.sh* file, remove system option `NETENCRYPTALGORITHM`.
- [Configure options for TLS on the SAS/CONNECT server](#). Skip this step when hot fixes for [KB0044144](#) are applied.
- Use [SAS Management Console on page 282](#) to update the metadata definition for the SAS/CONNECT server by using encryption value `SSL`. With [SAS 9.4M7](#), the `SSL` value is available by applying the hot fix for [KB0041222](#).

Note: The service for Object Spawner and SAS/CONNECT Spawner must be re-installed to pick up the changes. Run the following commands:

```
ObjectSpawner.bat -reinstall
ConnectSpawner.bat -reinstall
```

(Optional) SAS OLAP Server

`SASConfig\Levn\SASApp\OLAPServer\sasv9_usermods.cfg`

Use the same security options as SAS Metadata Server.

Note: If your deployment offers direct connections from clients to the OLAP server, make the same updates to the server's configuration file.

Note: On z/OS, the security options syntax is as follows:

```
netencralg="SSL"
objectserverparms="cel=everything other-parameters"
```

- 4 Start your SAS servers. For information about managing SAS servers, see [“Operating Your Servers” in SAS Intelligence Platform: System Administration Guide](#).

SAS/SHARE Server

`SASConfig\Levn\ShareServer\sasv9_usermods.cfg`

You must provide either `-sslcertsubj` or both `-sslcertiss` and `-sslcertserial`:

```
SET USERMODS=%JREOPTIONS% -sslcertsubj "common-name" -netencryptalgorithm "ssl"
```

or:

```
SET USERMODS=%JREOPTIONS% -sslcertiss "common-name" -sslcertserial "serial_number" -
netencryptalgorithm "ssl"
```

Here is an example where the CN=server.company.com:

```
SET USERMODS=%JREOPTIONS% -sslcertiss "server.company.com" -sslcertserial "serial_number" -
netencryptalgorithm "ssl"
```

Note: If other TLS-related parameters are needed, they must also be set here. For example, if you are configuring FIPS, you must specify the `ENCRYPTFIPS` system option in the `_usermods.cfg` file.

For more information, see [“SAS System Options for Encryption” in Encryption in SAS..](#)

Update Metadata Definitions for Servers

Use SAS Management Console to update the metadata definition for each of the following servers. For more information, see [“Administering SAS Management Console” in SAS Intelligence Platform: Desktop Application Administration Guide](#).

- SAS Metadata Server
- SAS Object Spawner
- SAS Workspace Server
- SAS Pooled Workspace Server

- SAS Stored Process Server
- SAS/CONNECT Spawner
- SAS/CONNECT Server (supports the SSL value)
- SAS OLAP Server

Starting with SAS 9.4M8 and above, to enable a TLS connection for a metadata-configured SAS/CONNECT deployment, you must configure TLS on the SAS/CONNECT server and the SAS/CONNECT spawner on your platform. For more information, see [“Configure TLS Using System Options in Server Configuration Files for Windows and Linux”](#) and [“Configure the SAS Intelligence Platform Server Tier to Use IBM System SSL on z/OS”](#). You must also use SAS Management Console to select the SSL value for the SAS/CONNECT server from the drop-down in SAS Management Console as described below. For all other SAS servers or spawners, including the SAS/CONNECT Spawner, the SSL selection for ENCRYPTION in metadata is not available.

Note: With SAS 9.4M7, the SSL option is available for the SAS/CONNECT server when applying the hot fix for [KB0041222](#).

In SAS Management Console, you can find the drop-down list of encryption values in the **Advanced Options** section of the server's Properties.

Take the following steps to select the TLS protocol (SSL value) for a SAS/CONNECT server.

- 1 Under **Server Manager**, select server definition .

Note: To access the server definition, expand the application server node  and the logical server node .

- 2 Right-click the first connection object , and select **Properties**.
- 3 In the Connection window, click the **Options** tab, and then click **Advanced Options**.
- 4 In the Advanced Options window, click the **Encryption** tab. Adjust the settings as necessary.

About NETENCALG and Client Encryption Level (CEL)

On direct connections, encryption is governed by the server's invocation command. Here are details and some examples:

NETENCRYPTALGORITHM (network encryption algorithm) is a SAS system option. The alias for NETENCRYPTALGORITHM [NETENCRYPTALGORITM=](#) is NETENCALG. The NETENCALG setting that is

defined for the SAS Metadata Server during installation is in the sasv9.cfg file on the SAS Metadata Server.

IMPORTANT Starting with SAS 9.4M9, SAS has deprecated the NETENCRYPTALGORITHM (NETENCALG) system option values AES, DES, RC2, RC4, and TripleDES. For SAS 9.4M7 and SAS 9.4M8, this change is made with hot fixes related to [KBOO41538](#). SAS highly recommends that you change the value to SSL to specify the use of the TLS protocol. For more information, see “[TLS Support for IOM and Other SAS Servers](#)” in *SAS Intelligence Platform: Application Server Administration Guide*.

- To configure TLS for the SAS Metadata Server, you must add the following code to the sasv9.cfg file:

```
-netencalg "SSL"
```

For more options needed to configure TLS, see [Table 14.3 on page 279](#) and [Table 14.1 on page 275](#).

- If a different NETENCALG setting has been added to the SAS Metadata Server sasv9_usermods.cfg file, that setting has priority.

Note: Starting with SAS 9.4M9, only SASProprietary or SSL are supported values for NETENCALG. SAS highly recommends using TLS.

CEL (client encryption level)

is a parameter in the OBJECTSERVERPARMS SAS system option. The CEL setting that is defined for the SAS Metadata Server during installation is in the SAS Metadata Server sasv9.cfg file.

- If you accept the default encryption settings during installation, the configuration file includes this line. (Credentials is the default **Server Encryption Level** in the Server Encryption window.)

```
-objectserverparms "cel=credentials {other-parameters}"
```

Note: The Credentials setting encrypts only login credentials. For more information, see “[Server Encryption](#)” in *SAS Intelligence Platform: Installation and Configuration Guide*.

- If, during installation, you selected the option to encrypt all traffic (setting Everything for the **Server Encryption Level** in the Server Encryption window), the setting in the SAS Metadata Server sasv9.cfg file is as follows:

```
-objectserverparms "cel=everything {other-parameters}"
```

Note: For more information, see “[Server Encryption](#)” in *SAS Intelligence Platform: Installation and Configuration Guide*.

- If a different CEL setting has been added to the Metadata Server sasv9_usermods.cfg file, that setting has priority.

Note: On z/OS, the preceding syntax examples are slightly different.

It is not necessary to specify encryption settings in the invocation command for every component for the following reasons:

- Encryption algorithm or protocol and encryption level are negotiated between each pair of communicating components. For example, when the OLAP server and object spawner initialize, they contact the Metadata Server and conform to its encryption settings. The same negotiation occurs when a client application contacts a server.
- For spawned servers (the stored process server, the pooled workspace server, and the workspace server), encryption is determined by metadata settings, not by a server invocation command.

See Also

- [“NETENCRYPTALGORITHM= System Option” in *Encryption in SAS*](#)
- [“Encryption for Data in Motion” on page 250](#)

Configure the SAS Intelligence Platform Server Tier to Use IBM System SSL on z/OS

Overview of Using IBM System SSL on z/OS

Starting with [SAS 9.4M8](#), encryption for data in motion is provided for z/OS using IBM System SSL. IBM System SSL supports FIPS 140-2 and FIPS 140-3. It provides all the capabilities of OpenSSL and is fully integrated with the z/OS security services (SAF) that have been in use for many years. SAS uses the Global Security Kit (GSK) as the API. The System SSL `gskkyman` shell program supports the management of certificates that are stored in a key database file or in a z/OS PKCS#12 file. RACF can also be used to generate new certificates. IBM provides documentation at [z/OS Cryptographic Services System SSL Programming Introduction](#).

Some configuration steps are required in order to configure a SAS deployment for System SSL. The configuration steps are slightly different for setting up System

SSL for z/OS 31-bit and for z/OS 64-bit. Be aware that z/OS 64-bit is supported only for a SAS Metadata Server deployment.

Prior to SAS 9.4M8, SAS included the Mozilla trusted CA bundle in the `trustedcerts.pem` file. SAS 9.4M8 and later running in z/OS can use only a key database file (`.kdb`) or a PKCS#12 file (`.p12`). PEM files are not supported. The SAS system option `SSLKEYRINGFILE=` supports `.kdb` and `.p12` files. For more information, see [“Configuration Examples for Accessing System SSL Certificates”](#).

Prior to SAS 9.4M8, you could use SAS Deployment Manager to add or remove TLS certificates from the truststore. SAS Deployment Manager cannot be used to add or remove certificates for SAS 9.4M8 (or later) clients and servers running in z/OS. However, all SAS Java applications, such as SAS Management Console, still expect to find certificates in `SASHOME (SASHOME/SASSecurityCertificateFramework/1.1/cacerts/)`. For more information about using the truststore with z/OS in SAS 9.4M8 and later, see [KB0041576](#).

More information about System SSL for SAS®9 on 31-bit z/OS is provided in [“IBM System SSL Provides TLS Capabilities for z/OS \(Starting with SAS 9.4M8\)”](#) in [Encryption in SAS](#). The following sections describe the steps to set up the Metadata Server on 64-bit z/OS.

For more information about SAS system options for System SSL, see [“SAS System Options Used for IBM z/OS System SSL Starting with SAS 9.4M8”](#) in [Encryption in SAS](#).

How To Configure TLS Using System Options in Server Configuration Files for z/OS

SAS servers require additional configuration in order to communicate with the SAS Metadata Server when System SSL is used for encryption on z/OS 64-bit. For information about the system options described in this section, see [“SAS System Options Used for IBM z/OS System SSL Starting with SAS 9.4M8”](#) in [Encryption in SAS](#).

Take the following steps to configure SAS 9.4 Intelligence Platform with System SSL:

- 1 System SSL encryption for the 64-bit z/OS is provided by hot fixes. See [Usage Note 70504: Support for IBM System SSL for z64](#) for assistance with downloading and deploying the required hot fixes.
- 2 When the SAS Intelligence platform server tier has been deployed and verified, change directories to the Unix System Services (USS) configuration directory.

Note: On z/OS, the SAS Intelligence Platform configuration directories are subdirectories of the USS configuration directory.

- 3 (Optional) Create a System SSL key database (`SystemSSL.kdb`) to store your certificates. Make sure that the key database is accessible by the keyring stash

file (SystemSSL.sth), which contains the encrypted password. Multiple types of key databases are supported, including KDB (.kdb) and SAF keyring. For more information about this step, including examples, see [“Configuration Examples for Accessing System SSL Certificates”](#).

- 4 Take the following steps to modify the configuration files in your SAS deployment. For more information about each option, see [“SAS System Options Used for IBM z/OS System SSL Starting with SAS 9.4M8”](#) in *Encryption in SAS*.

Note: The following examples use a key database (KDB file) to supply the certificates for authentication.

- a (Recommended) Make a copy of each file before modifying it.

- b Modify `./SASMeta/MetadataServer/sasv9_usermods.cfg` and add the following system options.

```
netencralg="SSL"

SSLKEYRINGFILE="/path-to-file/SystemSSL.kdb"

SSLKEYRINGSTASHFILE="/path-to-cert/SystemSSL.sth"

SSLKEYRINGLABEL="your-label-name"

SSLHWDetectMESSAGE="1"

SSLICSFERRORMESSAGE="1"
```

- c (Optional) Additional values are required if you want to capture GSK-level tracing from System SSL activities. For more information, see [“\(Optional\) Configure GSK Logging”](#).

- d Modify `./SASMeta/MetadataServer/stopMetadataServerNode.sas` and add the following system options in the file just after the `%include metaparm;` line and before the `PROC METAOPERATE NOCLUSTER` line.

```
option netencralg="SSL";

option SSLKEYRINGFILE="/path-to-file/SystemSSL.kdb";

option SSLKEYRINGSTASHFILE="/path-to-cert/SystemSSL.sth";

option SSLKEYRINGLABEL="your-label-name";

option SSLHWDetectMESSAGE="1";

option SSLICSFERRORMESSAGE="1";
```

- e Modify `./SASApp/sasv9_usermods.cfg` and add the following system options.

```
netencralg="SSL"

SSLKEYRINGFILE="/path-to-file/SystemSSL.kdb"

SSLKEYRINGSTASHFILE="/path-to-cert/SystemSSL.sth"

SSLKEYRINGLABEL="your-label-name"

SSLHWDetectMESSAGE="1"

SSLICSFERRORMESSAGE="1"
```

- f Modify `./ObjectSpawner/tkmvsenv_usermods.cfg` and add the following text to add the system options.

```
set SSLKEYRINGFILE="/path-to-file/SystemSSL.kdb"
set SSLKEYRINGSTASHFILE="/path-to-file/SystemSSL.sth"
set SSLKEYRINGLABEL="your-label-name"
set SSLHWDETECTMESSAGE="1"
set SSLICSFERRORMESSAGE="1"
```

- g (Optional) Add the following values if you want to capture GSK-level tracing from System SSL activities:

```
set SSLGSKTRACE="0xFF"
set SSLGSKTRACEFILE="/Lev1/ObjectSpawner/Logs/gskssl.objectspawner.%.trc"
```

- h Modify `./SchedulingServer/sasv9_usermods.cfg` and add the following system options.

```
netencralg="SSL"
SSLKEYRINGFILE="/path-to-file/SystemSSL.kdb"
SSLKEYRINGSTASHFILE="/path-to-file/SystemSSL.sth"
SSLKEYRINGLABEL="your-label-name"
SSLHWDETECTMESSAGE="1"
SSLICSFERRORMESSAGE="1"
```

- i Complete all of the following steps to configure SAS/CONNECT to work with a metadata-configured deployment.

IMPORTANT When SAS 9.4M8 or SAS 9.4M9 hot fixes for [KB0044144](#) are applied, the SAS/CONNECT server inherits the system options for configuring TLS from the SAS/CONNECT spawner.

- 1 Apply hot fix for [KB0044144](#) for [SAS 9.4M8](#) or [SAS 9.4M9](#).
- 2 Modify `./ConnectSpawner/tkmvsenv_usermods.cfg` and add the following system options.

```
set netencralg="SSL"
set SSLKEYRINGFILE="/path-to-file/SystemSSL.kdb"
set SSLKEYRINGSTASHFILE="/path-to-file/SystemSSL.sth"
set SSLKEYRINGLABEL="your-label-name"
set SSLHWDETECTMESSAGE="1"
set SSLICSFERRORMESSAGE="1"
```

(Optional) Add the following values if you want to capture GSK-level tracing from System SSL activities:

```
set SSLGSKTRACE="0xFF"
set SSLGSKTRACEFILE="/Lev1/ObjectSpawner/Logs/gskssl.objectspawner.%.trc"
```


- 3 Use [SAS Management Console on page 282](#) to update the metadata definition for the SAS/CONNECT server by using encryption value SSL.
- j Modify `./ShareServer/sasv9_usermods.cfg` and add the following system options.


```
netencralg="SSL"

SSLKEYRINGFILE="/path-to-file/SystemSSL.kdb"

SSLKEYRINGSTASHFILE="/path-to-file/SystemSSL.sth"

SSLKEYRINGLABEL="your-label-name"

SSLHWDTECTMESSAGE="1"

SSLICSFERRORMESSAGE="1"
```
- k (Optional) If you want to capture the GSK-level tracing from System SSL activities, some additional configuration parameters are required. For more information, see [“\(Optional\) Configure GSK Logging”](#).

Use SAS Management Console to update the Metadata Server

For information about using SAS Management Console, see [“Update Metadata Definitions for Servers”](#).

Use SAS Deployment Manager to Add Certificates to the SAS Deployment

Use SAS Deployment Manager to add your CA certificate(s) to the bundle of trusted CA certificates. For more information, see [“Certificate Management Using the Bundle of Trusted CA Certificates \(trustedcerts\)” in *Encryption in SAS*](#).

(Optional) Configure GSK Logging

Add tracing options to your configuration files if you want to capture GSK-level tracing from System SSL activities. The option `SSLGSKTRACE="0xFF"` enables GSK tracing. `SSLGSKTRACEFILE=` designates the location where the trace is stored..

Each SAS server or spawner has its own directory in the path, so you must add the `SSLGSKTRACEFILE=` option to the `sasv9_usermods.cfg` file for the applicable server, or to the spawner's `tkmvseenv_usermods.cfg` file.

Note: If you are adding the

Here are examples of server or spawner trace file locations:

Note: For all references to <config_dir>, substitute the fully-qualified path to the configuration directory. If the following options are being added to the tkmvsnv_usermods.cfg file, prepend the set command to each option.

```
set SSLGSKTRACEFILE="config_dir/Levl/SASMeta/MetadataServer/Logs/
gskssl.metaserver.%.trc"
```

```
SSLGSKTRACEFILE="config_dir/Levl/SASMeta/MetadataServer/Logs/
gskssl.metaserver.%.trc"
SSLGSKTRACEFILE="config_dir/Levl/SASMeta/MetadataServer/Logs/
gskssl.stopmetaserver.%.trc"
SSLGSKTRACEFILE="config_dir/Levl/SASApp/OLAPServer/Logs/
gskssl.olapserver.%.trc"
SSLGSKTRACEFILE="config_dir/Levl/SASApp/PooledWorkspaceServer/Logs/
gskssl.pooledworkspace.%.trc"
SSLGSKTRACEFILE="config_dir/Levl/SASApp/StoredProcessServer/Logs/
gskssl.storedprocess.%.trc"
SSLGSKTRACEFILE="config_dir/Levl/SASApp/WorkspaceServer/Logs/
gskssl.workspace.%.trc"
SSLGSKTRACEFILE="config_dir/Levl/SASApp/ConnectServer/Logs/
gskssl.connectserver.%.trc"
SSLGSKTRACEFILE="config_dir/Levl/SchedulingServer/Logs/
gskssl.schedulingserver.%.trc"
SSLGSKTRACEFILE="config_dir/Levl/ShareServer/Logs/gskssl.shareserver.%.trc"
SSLGSKTRACEFILE="config_dir/Levl/ObjectSpawner/Logs/gskssl.objectspawner.%.trc"
```

Configuration Examples for Accessing System SSL Certificates

SAS and IBM support multiple methods of communicating with System SSL where the certificates are stored. The SAS option `SSLKEYRINGFILE=` determines which method is in use. The supported methods are as follows:

- A path name to a key data base (KDB file) that was generated by (for example) the `gskkyman` utility. If a key data base is used, you must provide the password. You can configure the SAS option `SSLKEYRINGSTASHFILE=` with the path to the stash file (STH file) created by `gskkyman`, which is the preferred method. As an alternative, you can provide the password itself using the SAS option `SSLKEYRINGPW=`.
- An SAF keyring name. The syntax is `SSLKEYRINGFILE=USERID/KEYRINGNAME`. If `USERID/` is not provided, the user ID of the running process is assumed.
- A z/OS PKCS#11 token name. This parameter must be specified as `*TOKEN*/token_name`.

- A PKCS#12 file.

For more information about each option, see [“SAS System Options Used for IBM z/OS System SSL Starting with SAS 9.4M8” in *Encryption in SAS*](#).

Here is an example configuration using a key database (KDB file):

```
NETENCRYPTALG=SSL
SSLKEYRINGFILE="path-to-KDB-file.kdb"
SSLKEYRINGSTASHFILE="path-to-STH-file.sth"
```

or:

```
SSLKEYRINGPW="KDB-password"
SSLKEYRINGLABEL="label-name-in-KDB"
```

Here is an example configuration using an SAF KEYRING name:

```
NETENCRYPTALG=SSL
SSLKEYRINGFILE="SAF-keyring-name"
```

or:

```
SSLKEYRINGFILE="user-ID/SAF-keyring-name"
SSLKEYRINGLABEL="label-name-from-keyring"
```

Here is an example using a PKCS#11 token:

```
NETENCRYPTALG=SSL
SSLKEYRINGFILE="*TOKEN*/token-name"
SSLKEYRINGLABEL="label-name-from-token"
```

Here is an example using a PKCS#12 file (.P12):

```
NETENCRYPTALG=SSL
SSLKEYRINGFILE="path-to-P12-file.p12"
SSLKEYRINGPW="KDB-password"
SSLKEYRINGLABEL="label-name-in-P12"
```

How to Change Encoding for Data in Motion for Outbound Passwords

Outbound Passwords

A password is outbound when a client retrieves the password from the metadata in order to provide seamless access to a server such as Oracle. The password is outbound from the perspective of the Metadata Server. Connections to third-party servers often use outbound passwords. Most other connections do not use outbound passwords.

Instructions

Note: Starting with SAS 9.4M8, your operating system is using cryptographic libraries that are included with the operating system. These libraries are required in order to use passwords that are stored with SAS003, SAS004, or SAS0005. Prior to SAS 9.4M8, the required libraries were included with Base SAS or SAS/SECURE. For more information, see [“Cryptographic Library Support \(Starting with SAS 9.4M8\)” in Encryption in SAS](#).

In the initial configuration, outbound passwords are transmitted in SAS002 format (SASProprietary). You can choose to increase the encoding strength for outbound passwords to AES 256 with 16-bit salt (SAS003), AES 256 with 64-bit salt (SAS004), or AES 256 with 64-bit salt and additional hash iterations (SAS005).

To increase encoding strength for outbound passwords:

- 1 Edit the omaconfig.xml file on the Metadata Server to change the initial setting, RETURNPASSWORDS="SAS002", to a more secure setting (for example, RETURNPASSWORDS="SAS005"). The omaconfig.xml file is located in your equivalent of `SAS/Config/Lev1/SASMeta/MetadataServer/`.

Note: For more information, see the RETURNPASSWORDS option in [“Reference Information for omaconfig.xml” in SAS Intelligence Platform: System Administration Guide](#).

- 2 Restart the Metadata Server.
- 3 Verify that server connections continue to function as expected. If you encounter problems, either review the following topic or revert to RETURNPASSWORDS="SAS002".

Compatibility

Almost all connections involve a SAS server, and SAS servers can decode SAS005 passwords. For example, connections from SAS Information Map Studio to an Oracle server go through a workspace server. The workspace server decodes the outbound Oracle password.

However, a few specialized connections run directly from a Java client or .NET client to a third-party server. These clients cannot decode SAS003, SAS004, or SAS005 passwords. This is a deliberate limitation that reduces security exposures. Of course, a third-party server cannot decode SAS003, SAS004, or SAS005 passwords either. As a result, such connections fail if they attempt to use a password that is in either of those formats. Here are some examples of connections that cannot use a SAS003, SAS004, or SAS005 password:

- connections from a Java client or .NET client to an Esri server.

Note: The Esri server uses host authentication. If possible, avoid the use of outbound passwords by locating this server on a machine that recognizes the accounts with which users log on to SAS applications. This facilitates use of cached credentials to access the Esri server.

- certain connections within the following solutions:
 - SAS Profitability Management
 - SAS Activity-Based Management
 - SAS Merchandise Intelligence
 - SAS Model Manager (in some configurations)
 - SAS LASR Analytic Server
- internal database management system (DBMS) accounts (for example, the administrator of the SAS Web Infrastructure Platform Data Server, dbmsowner).

If your deployment requires connections that are incompatible with SAS003, SAS004, or SAS005 passwords, choose either of the following approaches:

- Preserve the initial setting of `RETURNPASSWORDS="SAS002"`.

Note: With the `STOREPASSWORDS="SAS003"` and `RETURNPASSWORDS="SAS002"` settings, outbound passwords are stored in SAS003 format and downgraded to SAS002 format before they are transmitted.

- Set `RETURNPASSWORDS="SAS003"`, but also selectively force the outbound passwords for the problematic connections to be transmitted in SAS002 format. To force a particular password to be transmitted in SAS002 format, store that password in that format. A password that is stored in SAS002 format is transmitted in that format even if `RETURNPASSWORDS="SAS003"`. The Metadata Server does not upgrade the encoding strength of a stored password when the password is transmitted.

To use this approach:

- 1 In the `omaconfig.xml` file on the Metadata Server, set `STOREPASSWORDS="SAS002"`. Restart the Metadata Server.
- 2 In SAS Management Console, under **User Manager**, locate and select the logon that contains the outbound password. The logon is on the **Accounts** tab of a user or group definition.
- 3 Click **Edit**. In the Login Properties window, enter and confirm the password. Click **OK** to close the Login Properties window. Click **OK** again to close the user or group properties window.

Note: This stores the password in the metadata in SAS002 format.

- 4 Repeat steps 2 and 3 for any other problematic outbound passwords.

- 5 In the omaconfig.xml file, upgrade the settings (for example, set STOREPASSWORDS="SAS005" and RETURNPASSWORDS="SAS005"). Restart the Metadata Server.
- 6 Verify that server connections continue to function as expected.

Note: Anytime in the future that you update the SAS002 passwords, you must repeat the preceding steps in order to store the new password in SAS002 format.

How to Configure TLS between the Metadata Server and an LDAP Server

Before You Begin

The configuration procedure that is described here is not required for every deployment. The following instructions support encryption of direct communication between SAS Metadata Server and an LDAP server using the Transport Layer Security (TLS) protocol. This information is also applicable to the TLS predecessor protocol, Secure Sockets Layer (SSL).

This functionality applies to only specialized configurations in which the SAS Metadata Server directly uses an LDAP server as an authentication provider. See [“Direct LDAP Authentication” on page 171](#).

Configuration Instructions

- 1 On the LDAP server:
 - a Make sure that a port is configured for TLS.
 - b Examine the **Client Certification** setting and note its implications for your work in step 2e below.
 - If the setting specifies that a client certificate is optional, step 2e below is optional for you. This is the default setting on most LDAP servers.
 - If the setting specifies that a client certificate is required, step 2e below is mandatory for you.

- If the setting specifies that a client certificate is not allowed, a client certificate is ignored, so there is no reason for you to perform step 2e below.

2 On the Metadata Server:

- a If support is not already in place for direct LDAP, configure that support. See [“How to Configure Direct LDAP Authentication” on page 209](#).
- b Make sure that the value that is specified in the `LDAP_PORT` (or `AD_PORT`) environment variable matches a TLS-configured port on the LDAP server.
- c In the same location where you set environment variables to configure the Metadata Server for direct LDAP, add the `LDAP_TLSMODE` (or `AD_TLSMODE`) environment variable, and set it to 1. Setting this variable causes the Metadata Server to attempt to use TLS.
- d Provide a certificate authority (CA) certificate to use TLS.

Host	Details
Windows	The CA certificate is installed using Microsoft Certificate Services.
UNIX	For SAS 9.4M2 and earlier releases, the CA certificate file is provided in an accessible directory and is referenced by the <code>SSLCALISTLOC</code> option.¹ In SAS 9.4M3 and later, certificates are located in the trusted CA bundle (<code>trustedcerts.pem</code>) by default, and the <code>SSLCALISTLOC</code> system option that points to this certificate file is automatically added at installation to the <code>sasv9.cfg</code> file. The <code>sasv9.cfg</code> file is located in <i>SAS-installation-directory/SASFoundation/9.4</i>. For more information, see “Manage Certificates in the Trusted CA Bundle Using the SAS Deployment Manager” in <i>Encryption in SAS</i>.
z/OS	The CA certificate file is provided in an accessible directory and referenced by the <code>SSLCALISTLOC</code> option. ¹

¹ Add the option to the SAS Metadata Server invocation command (in the same location where you specified `-authpd`). For syntax, see “SAS System Options for Encryption” in [Encryption in SAS](#).

- e If the LDAP server requires client certification (see step 1b above), specify the location of client certificates.

Host	Details
Windows	The client certificate is installed using Microsoft Certificate Services. The <code>SSLCERTSUBJ</code> , <code>SSLCERTSERIAL</code> , and <code>SSLCERTISS</code> options are used to locate the client certificate. ¹

Host	Details
UNIX	The client certificate files are provided in an accessible directory. The SSLCERTLOC, SSLPVTKEYLOC, and SSLPVTKEYPASS options can be used to locate the client certificate. PKCS12 certificates can also be used with the SSLPKCS12LOC and SSLPKCS12PASS options. ¹
z/OS	The client certificate files are provided in an accessible directory. The SSLCERTLOC, SSLPVTKEYLOC, and SSLPVTKEYPASS options can be used to locate the client certificate. PKCS12 certificates can also be used with the SSLPKCS12LOC and SSLPKCS12PASS options. ¹

- ¹ Add the necessary options to the Metadata Server invocation command (in the same location where you specified -authpd). For syntax, see "SAS System Options for Encryption" in [Encryption in SAS](#).

- Restart the Metadata Server.
- To test the results, use the following code. Replace the sample values with your Metadata Server connection information, and submit the code in a SAS code editor.

```
proc metaoperate
  server="a123.us.company.com"
  port=8561
  userid="myLDAPuserID@myldap"
  password="myLDAPpassword"
  action=status;
run;
```

Connecting to an LDAP Load Balancer

Potential Problem

In the default TLS configuration, each connection includes a server name check to verify that the host name in the certificate of the node to which the Metadata Server connected matches the host name of the target LDAP server.

If the connection is to an LDAP server load balancer (rather than directly to the authenticating LDAP server), the server name check might fail. The LDAP load balancer might be on a different node than the LDAP server to which the Metadata Server is redirected.

Preferred Solution: Wildcard Certificate

If you encounter the problem that is described in the preceding section, use a wildcard when specifying the subject name in the certificate.

For example, rather than specifying `myHost.myCompany.com`, specify `*.myCompany.com`.

CAUTION

If you use a wildcard, only the domain portion of the server name in the certificate is validated against the server that you connect to. Do not use this approach unless you encounter the problem that is described in the preceding section.

Note: Using a wildcard is the preferred solution because it provides domain name validation, which reduces the risk of Domain Name System (DNS) redirection or hijacking.

Alternate Solution: Environment Variable

As an alternative to using a wildcard certificate, you can disable the server name check by defining an additional environment variable (`NOSSLNAMECHECK=1`) for the Metadata Server. For example, you might add the following setting to the `sasv9.cfg` file:

```
-set NOSSLNAMECHECK 1
```

Or, you can define the environment variable in the Metadata Server start-up script as follows:

```
export NOSSLNAMECHECK=1
```

CAUTION

In this approach, neither the server name nor the domain name is validated. For this reason, using a wildcard certificate is the preferred solution to the problem that is described in the preceding section.

SASSSLREQCERT

Windows Specifics: Windows clients always request and verify server certificates. The `SASSSLREQCERT` environment variable is supported only on UNIX.

The `SASSSLREQCERT` environment variable specifies what checks to perform on server certificates in a TLS session. This environment variable supports the following settings:

DEMAND

A server certificate is requested. If no valid certificate is provided, the session terminates. `DEMAND` is the default setting.

TRY

The client requests a server certificate. If no certificate is provided, the session proceeds normally. If an invalid certificate is provided, the session terminates.

ALLOW

The client requests a server certificate, but the session proceeds normally even if no certificate is provided or an invalid certificate is provided.

NEVER

The client does not request a server certificate.

For example, add the following setting to the script:

```
export SASSSLREQCERT=ALLOW
```

See Also

- [“About Setting Environment Variables” on page 216](#)
- [Encryption in SAS](#)

How to Implement Certificates

Implementing Certificates

The following list summarizes the steps to implement the various certificate options. For more information about the use of certificates in the SAS Middle-Tier, see [“Middle-Tier Servers and TLS” in SAS Intelligence Platform: Middle-Tier Administration Guide](#).

- Using self-signed certificates automatically generated by SAS Deployment Wizard. For more information, see [“Using Auto-generated Certificates”](#).
This option consists of installing and configuring the SAS middle tier using the deployment wizard.
- Providing your own certificates. For more information, see [“Providing Your Own Certificates”](#).

This option consists of these high-level steps:

- 1 If you do not already have them, obtain your certificates and private keys. Store these certificates and private keys on any machine that has the SAS components that use them.
 - 2 Install and configure your SAS metadata tier and SAS server tier.
(Use SAS Deployment Wizard.)
-
- Note:** When prompted, use the auto-generated certificates for SAS Deployment Agent and its remote client. Later, you can replace the auto-generated certificates with certificates that you provide.
-
- 3 On both Windows and UNIX on the SAS metadata-tier and SAS server-tier machines, depending on your release of SAS 9.4, add your CA root certificate and intermediate certificates either to the trusted CA bundle or to the SAS Private JRE.

(Use SAS Deployment Manager or the Java `keytool -importcert` command.)
 - 4 Install your SAS middle tier.
(Use SAS Deployment Wizard.)
 - 5 On both Windows and UNIX middle-tier machines, add your certificates to the trusted CA bundle.

(Use SAS Deployment Manager.)
 - 6 Configure your SAS middle tier.
(Use SAS Deployment Wizard.)
 - 7 Perform manual, post-deployment steps.
- Using a combination of self-signed certificates automatically generated and providing your own certificates. For more information, see [“Using a Combination of Auto-generated and Your Own Certificates”](#).

This option has the same steps as providing your own certificates. The only difference is that you do not replace the auto-generated certificates with your own certificates.

Using Auto-generated Certificates

If you want SAS Deployment Wizard to generate self-signed certificates to secure SAS Deployment Agent and its remote client and the SAS Environment Manager server and its agents, perform these steps:

Note: Using auto-generated certificates secures your SAS Deployment Agent and SAS Environment Manager only. To secure all of the components in your SAS middle tier, you must provide your own certificates or use a combination of your

own certificates and certificates auto-generated by SAS Deployment Wizard. For more information, see [“Implementing Certificates” on page 262](#).

-
- 1 Run SAS Deployment Wizard. Make sure that both **Install SAS Software** and **Configure SAS Software** are selected.
 - 2 When you are prompted for the SAS Deployment Agent remote communication configuration, select **Generate credentials to secure the connection**.
 - 3 When you are prompted for the configuration prompting level, select **Custom**.
 - 4 When you are prompted, make sure that you select **Generate a default JKS format keystore from SAS Environment Manager**.
 - 5 When you are prompted, make sure that you select **Establish secure communication**.

Providing Your Own Certificates

For SAS Deployment Agent and its remote client and for the SAS Environment Manager server and its agents, SAS Deployment Wizard provides auto-generated, self-signed certificates by default. Because these certificates have intermachine connections, are behind your firewall, and do not involve users, the reasons to implement site-signed or third-party-signed certificates are less urgent. However, if you must use your own certificates, you can.

This topic explains how to accept the default action, which enables SAS Deployment Wizard to install auto-generated, self-signed certificates for you. After the deployment wizard has completed, you can replace the auto-generated, self-signed certificates with your own certificates, eliminating several manual configuration steps.

To provide your own certificates to secure your SAS middle tier, follow these steps:

- 1 If you do not already have them, obtain your certificates. For more information, see [“Obtaining Certificates” on page 259](#).
- 2 Select a location on each machine in your SAS environment where you will store your certificates. Make sure that your certificates are saved in these locations.
- 3 Use the following table to record keystore and truststore locations and passwords for SAS Deployment Agent and its remote client.

SAS Deployment Wizard prompts you for these locations and passwords during SAS installation and configuration. The deployment wizard encodes the passwords that you provide into SAS proprietary format (SAS002).

Note: Your keystore must be in JKS format.

TIP SAS recommends that, rather than providing your own certificates to SAS Deployment Agent during installation, you accept the default and let SAS Deployment Wizard install auto-generated, self-signed certificates for you. After the deployment wizard has completed, you can replace the auto-generated, self-signed certificates with your own certificates, eliminating several manual configuration steps.

Table 14.4 SAS Deployment Agent Keystore and Truststore Information

Keystore and Truststore Properties	Actual Value You Are Using
SAS Deployment Agent keystore path	
SAS Deployment Agent keystore password	
SAS Deployment Agent truststore path	
SAS Deployment Agent truststore password	
SAS Deployment Agent remote client keystore path	
SAS Deployment Agent remote client keystore password	
SAS Deployment Agent remote client truststore path	
SAS Deployment Agent remote client truststore password	

- 4 Use the following table to record keystore locations and passwords for the SAS Environment Manager server and its agents.
- SAS Deployment Wizard prompts you for these locations and passwords during SAS installation and configuration. The deployment wizard encodes the passwords that you provide into SAS proprietary format (SAS002).

Table 14.5 SAS Environment Manager Keystore Information

Keystore Properties	Actual Value You Are Using
SAS Environment Manager server keystore path	
SAS Environment Manager server keystore password	
SAS Environment Manager alias	
SAS Environment Manager agent keystore path	
SAS Environment Manager agent keystore password	

- 5 Use the following table to record certificate and private key locations for SAS Web Server. SAS Deployment Wizard prompts you for these locations and passwords during SAS installation and configuration.

Table 14.6 SAS Web Server Private Key and Certificate Information

Private Key and Certificate Properties	Actual Value You Are Using
SAS Web Server X.509 certificate path	
SAS Web Server RSA private key path	

- 6 Use the following table to record keystore locations and passwords for SAS Web Application Server and its agents. SAS Deployment Wizard does *not* prompt you for these locations and passwords. You must manually update the SAS Web Application Server configuration with this information after you deploy SAS.

TIP You can use the same keystore for SAS Environment Manager and SAS Web Application Server if they are saved on the same machine.

Table 14.7 SAS Web Application Server Keystore Information

Keystore Properties	Actual Value You Are Using
SAS Web Application Server keystore path	
SAS Web Application Server keystore password	

- 7 Install and configure your SAS metadata-tier and SAS server-tier machines. For more information, see [“Using Auto-generated Certificates” on page 299](#).
- 8 Add your CA root certificate and intermediate certificates to your SAS metadata-tier and SAS server-tier machines. For more information, see [“Overview of CA Certificate Management Using the SAS Deployment Manager” in *Encryption in SAS*](#).
- 9 On each of your middle-tier machines, start SAS Deployment Wizard by navigating to `SAS-installation-directory/SASDeploymentManager/9.4`, and launch `setup.exe` (Windows) or `setup.sh` (UNIX). On Windows, you can use the shortcut on the **Start** menu.
- 10 On the Select Deployment Type page, select **Perform a Planned Deployment** and **Install SAS Software**.

Note: Make sure that **Configure SAS Software** is deselected.

- 11 When prompted by the deployment wizard, select **Generate credentials to secure the connection**, and click **Next**.

TIP SAS recommends that, rather than providing your own certificates to SAS Deployment Agent during installation, you accept the default and let SAS Deployment Wizard install auto-generated self-signed certificates for you. After the deployment wizard has completed, you can replace the default SAS Environment Manager auto-generated JKS keystore with your own certificates. Selecting **Generate credentials to secure the connection** eliminates several manual configuration steps.

- 12 On the Additional Resources page, click **Finish** to close the deployment wizard.
- 13 If you have additional middle-tier machines, repeat steps 4 through 9 on each middle-tier machine.
Otherwise, proceed to the next step.
- 14 Perform one of the following steps based on your operating system:
 - Windows

Depending on your release of SAS 9.4, add your CA root certificate and intermediate certificates either to the trusted CA bundle or to the SAS Private JRE on each of your middle-tier machines. For more information, see [“Add a Certificate to the Trusted CA Bundle” in *Encryption in SAS*](#) or [“Add Your Certificates to the SAS Private JRE” in *Encryption in SAS*](#).

■ UNIX:

Depending on your release of SAS 9.4, add your CA root certificate and intermediate certificates either to the trusted CA bundle or to the SAS Private JRE on each of your middle-tier machines.

For more information, see [“Add a Certificate to the Trusted CA Bundle” in *Encryption in SAS*](#) or [“Add Your Certificates to the SAS Private JRE” in *Encryption in SAS*](#).

- 15 On each of your middle-tier machines, start SAS Deployment Wizard by navigating to `SAS-installation-directory/SASDeploymentManager/9.4`, and launch `setup.exe` (Windows) or `setup.sh` (UNIX). On Windows, you can use the shortcut on the **Start** menu.

- 16 On the Select Deployment Type page, select **Perform a Planned Deployment** and **Configure SAS Software**.

Note: Make sure that **Install SAS Software** is deselected.

- 17 When prompted for the configuration prompting level, select **Custom**.
- 18 On the SAS Web Server Configuration page, select **HTTP Protocol using Secure Sockets**, and click **Next**.

- 19 The deployment wizard prompts you for the following information. Enter the values that you recorded earlier in [Table 14.6](#), and click **Next**:

- In the X.509 certificate field, enter the path to the X.509 certificate with the DNS name of the current machine as the common name.
- In the RSA private key field, enter the path to the RSA private key that is not protected by a passphrase.

Note: SAS Web Server does not use a passphrase so that it can start without having to prompt the administrator for a passphrase.

- 20 When prompted by the deployment wizard, select **Generate a default JKS format keystore from SAS Environment Manager**, and click **Next**.

TIP SAS recommends that, rather than providing your own certificates to SAS Environment Manager during installation, you accept the default and let SAS Deployment Wizard install auto-generated self-signed certificates for you. After the deployment wizard has completed, you can replace the default SAS Environment Manager auto-generated JKS keystore with your own certificates. Selecting **Generate a default JKS format keystore from SAS Environment Manager** eliminates several manual configuration steps.

Using a Combination of Auto-generated and Your Own Certificates

For SAS Deployment Agent and its remote client and for the SAS Environment Manager server and its agents, SAS Deployment Wizard provides auto-generated, self-signed certificates by default. Because these certificates have intermachine connections, are behind your firewall, and do not involve users, there is less of a reason to implement site-signed or third-party-signed certificates. For the other middle-tier components—the SAS Web Server and the SAS Web Application Server—you provide your own certificates.

The steps for using a combination of certificates are almost identical to the steps performed when you provide your own certificates, except that you replace the auto-generated certificates with your own certificates. For more information, see [“Providing Your Own Certificates”](#).

How to Enable IOM Communication over HTTPS Proxy

Introduction

Starting with [SAS 9.4M3](#), you can configure Integrated Object Model (IOM) communication over HTTPS. You can update a SAS Web Server to act as an HTTPS proxy, allowing secured IOM communications between applications and back end IOM Servers, such as SAS Metadata Server or SAS Workspace Server.

Prior to configuring the proxy, SAS Web Server must be configured for TLS. If you did not choose to configure with secure sockets during the initial installation and configuration with SAS Deployment Wizard, you can manually configure SAS Web

Server to use HTTPS. For more information, see [“Configure TLS for SAS Web Server Manually”](#) in *SAS Intelligence Platform: Middle-Tier Administration Guide*.

Configuring the HTTPS Proxy

Note: If you are upgrading, you need to reconfigure the HTTPS proxy settings in the sas.conf file after the upgrade. During the upgrade, a backup of the sas.conf file is created in the *SAS-configuration-directory\Lev_n\Web\WebServer\conf\Backup* directory. After the upgrade, you can use the sas.conf backup file as a reference when reconfiguring the sas.conf file.

To configure an HTTPS proxy, perform the following steps to update the specific HTTP components and configuration files:

- 1 For **SAS 9.4M3** and previous releases, install the latest hot fix. The hot fix includes an updated version of mod_proxy_connect, which is required for SAS Web Server to act as an HTTPS proxy.
- 2 Edit the *SAS-configuration-directory\Levn\Web\WebServer\conf\sas.conf* file and specify which ports the proxy server should allow to connect to the back end by including the following information:

```
ProxyRequests On
ProxyVia On
# Below should include ports to be proxied, for
example, 8561 for the metadata server.
AllowCONNECT 8561 8591

<Proxy *>
    Order allow,deny
    Allow from all
</Proxy>
```

Note: The comment must be on one line. It is shown on more than one line in the preceding code sample for display purposes only.

- 3 Restart the SAS Web Server by running the following command:


```
SAS-configuration-directory\Levn\Web\WebServer\bin\httpdctl restart
```
- 4 Set the proxy list for your client. Proceed to either [“Connecting to an IOM Server via Proxy While Using an Environment Variable”](#) on page 307 or [“Registering Proxy Information in SAS Metadata Server for Other Servers”](#) on page 308.

CAUTION

Do not use both methods to set the proxy list for your client. If the environment variable is used and the proxy information is stored in metadata, some clients might interpret this as multiple proxies. It is possible to pass through multiple proxies to reach the back-end IOM server. Therefore, configuring both methods

might cause the client to attempt to go through the same proxy twice. If the proxy allows connections to itself, this results in an inefficient trip back through the same proxy. However, if the proxy does not allow connections to itself, this could result in a connection error.

Connecting to an IOM Server via Proxy While Using an Environment Variable

Now that the proxy is configured, clients must be directed to use the proxy to communicate with the IOM Server. In the following procedure, we use SAS Metadata Server and SAS Management Console. An environment variable can be used to force the client application to go through a proxy when making the outgoing connection.

Note: SAS Enterprise Guide does not read the `SAS_IOM_PROXYLIST` variable when it is set directly as a Windows environment variable. Instead, SAS Enterprise Guide sets the `SAS_IOM_PROXYLIST` environment variable only for the current SAS Enterprise Guide process, using one of the following values:

- The value that is specified in the active profile.
In the SAS Enterprise Guide connection profile, the active profile is specified in the **Set proxy list for metadata server** field.
 - The value that is temporarily specified in any additional server definitions when connecting to those servers.
-

Take the following steps:

- 1 The client application's IOM server connection information should not be changed. It should continue to connect to the same host and port where the server itself is running. If the SAS Metadata Server is running on *myMetadataServer.mydomain.com* on port 8561, verify that the connection information in SAS Management Console is the same:
 - Machine: *myMetadataServer.mydomain.com*
 - Port: 8561
- 2 Configure SAS Management Console to connect through the proxy `https://myproxy.mydomain.com` by setting the `SAS_IOM_PROXYLIST` environment variable.

Set the variable before SAS Management Console is started. The value of this variable is a semicolon-separated list of one or more proxies through which the data from the client application must pass before connecting to SAS Metadata Server. The variable should resemble the following example:

```
SAS_IOM_PROXYLIST = https://myproxy.mydomain.com
```

After this variable is set, all connections to IOM servers from this host will go through the proxy at `https://myproxy.mydomain.com`. A tunnel through this proxy is created to communicate with the back-end server. In our example, that would be SAS Metadata Server.

Registering Proxy Information in SAS Metadata Server for Other Servers

It is also possible to store proxy information in SAS Metadata Server. Applications that find server definitions in SAS Metadata Server, such as the SAS Workspace Server, uses the proxy information stored with the server definition to make a connection through the proxy to the server endpoint.

In SAS Metadata Server, individual machines are configured to use proxies using a proxy list object. Machines are registered in the proxy list object and are mapped to a list of proxies. Any client that is making a connection to the registered machines of the proxy list should find the list of proxies and make connections through these proxies to the machine registered.

To configure the proxy list in SAS Metadata Server, follow these steps.

- 1 Log on to SAS Management Console as an administrator.
- 2 On the **Plug-ins** tab, navigate to **Environment Management** ⇒ **Server Manager**.
- 3 Right-click **Server Manager**, and select **New Server**.
- 4 In the New Server Wizard window, navigate to and select the Proxy List entry, and then click **Next**.
- 5 Provide a name for the proxy list object and click **Next**. The name that you specify is for reference only.
- 6 To add a proxy to the list, click **New** under the **Proxies** section. Enter the proxy URI, and then click **OK**.
- 7 Repeat step 6 to add all of the proxies. The proxy list is ordered, and applications enforce the order of the proxies as they appear in this list.
- 8 After the proxies are added, select the machines to register with the Proxy List. In the **Proxied Machines** section, move one or more machines from the **Available** list to the **Selected** list. Click **Next** to continue.
- 9 A summary of the details about the Proxy List object that will be created is displayed. Click **Finish**.

After the proxy list is defined in metadata, any connection that is going to an IOM Server on the defined server (for example, `myServer.mydomain.com`) will go through the `https://myproxy.mydomain.com` proxy first.

PART 5

Appendix

Appendix 1	
Checklists	311
Appendix 2	
User Import Macros	317
Appendix 3	
Security Report Macros	345
Appendix 4	
Choices in Workspace Server Pooling	357

Appendix 1

Checklists

<i>Checklist for a More Secure Deployment</i>	311
Introduction	311
General	311
Web-Layer	312
Metadata-Layer	312
Password Protection	313
<i>Members of Selected Groups and Roles</i>	313
<i>Permission Patterns in Predefined ACTs</i>	314

Checklist for a More Secure Deployment

Introduction

You can use this appendix to verify that security issues are being addressed as appropriate for your environment and goals. Not all measures are relevant in all deployments. It is a good practice to review your security configuration on a periodic basis.

General

- Make sure that the configuration directories are protected by appropriate operating system permissions. See [“Recommended Operating System Protections for Windows Machines”](#) in *SAS Intelligence Platform: Installation and Configuration Guide*.

- If your deployment includes sensitive data, review host access to SAS data. See [Access to SAS Data on page 111](#).
- On Windows, minimize availability of the privilege **Log on as batch job**. See [“Windows Privileges” on page 23](#).
- Make sure that you are using an encryption model that is appropriate for your environment. See [“Introduction to the Encryption Model” on page 247](#).
- Consider limiting the scope of trusted peer connections. See [“Trusted Peer Connections” on page 182](#).
- Consider tracking changes to server logging levels. See [“Audit Server Logging Level Changes That Originate in Client Applications” in SAS Intelligence Platform: System Administration Guide](#).

Web-Layer

For a full discussion of middle-tier security, see [SAS Intelligence Platform: Middle-Tier Administration Guide](#) and [SAS Intelligence Platform: Web Application Administration Guide](#).

The following list highlights selected measures:

- Consider disabling concurrent logon sessions to the SAS middle tier. See [“Disable Concurrent Sign-In Sessions” in SAS Intelligence Platform: Middle-Tier Administration Guide](#).
- Consider reducing the HTTP Session time-out interval for the SAS web applications. See [“Configure the HTTP Session Time-Out Interval” in SAS Intelligence Platform: Middle-Tier Administration Guide](#).
- If your deployment includes sensitive data, make sure that the SAS Web Report Studio query cache library (wrstemp) and distribution library (wrsdist) are protected by appropriate operating system permissions. See [“Manage Host Access to the Query Cache Directory” in SAS Intelligence Platform: Web Application Administration Guide](#) and [“Verifying Permissions for the Distribution Library” in SAS Intelligence Platform: Web Application Administration Guide](#).
- If your deployment includes the SAS Anonymous Web user, determine whether it is necessary and appropriate to keep that identity in place. See [“PUBLIC Access and Anonymous Access” on page 164](#).

Metadata-Layer

- Limit the WriteMetadata permission on servers. See [“Protect Server Definitions” on page 101](#).
- Limit the WriteMetadata permission on ACTs. In general, only the SAS Administrators group has a grant of the WriteMetadata permission on the **Authorization** tab of an ACT.

- Limit the WriteMetadata permission on custom folders. To reduce the chance of inadvertent or deliberate changes to a custom folder, grant WriteMemberMetadata (instead of WriteMetadata) to users who should contribute only content. See [“WriteMetadata and WriteMemberMetadata” on page 38](#).
- Review the WriteMetadata permission on OLAP schemas and libraries. To prevent someone from adding cubes to an OLAP schema or tables to a library, set denials of the WriteMetadata permission on the schema or library. Remember to preserve access for administrators as appropriate.
- Review the permission pattern of the predefined ACTs. See [“Permission Patterns in Predefined ACTs” on page 314](#).
- Review who has privileged user status from metadata memberships. See [“Members of Selected Groups and Roles” on page 313](#).
- Consider tracking changes to metadata layer permissions and ACTs. See [“Auditing” on page 4](#).
- Consider limiting the locations from which SAS Web Report Studio uses information maps. See [“Limit the Availability of Relational Information Maps That Implement Row-Level Security” in SAS Intelligence Platform: Web Application Administration Guide](#).

Password Protection

- Upgrade encryption strength for passwords in configuration files. See [“How to Maximize Encryption of Stored Passwords” on page 266](#).
- Upgrade encryption strength for outbound passwords in transit. See [“How to Change Encoding for Data in Motion for Outbound Passwords” on page 291](#).
- Consider preventing users from saving their passwords in their desktop connection profiles. See [“Client-Side Storage of Passwords” on page 18](#).
- Consider strengthening password policies for internal accounts. See [“How to Change Internal Account Policies” on page 239](#).

Members of Selected Groups and Roles

The following table documents the standard members of selected metadata-layer groups and roles.

Table A1.1 *Members of Selected Groups and Roles*

Group or Role	Usual Members
 SAS General Servers	 SAS Trusted User (no others)
 SAS System Services	 SAS Trusted User (no others)
 SAS Administrators	 SAS Administrator and other administrators
 Metadata Server: Unrestricted	 SAS Administrator and possibly other administrators
 Metadata Server: Trusted Service Identities	 SAS Trusted User (no others)
 Metadata Server: User Administration	 SAS Administrators
 Visual Analytics: Analysis	 Visual Analytics Users
 Visual Analytics: Basic	 SASUSERS
 Visual Analytics: Report Viewing	 Visual Analytics Users

CAUTION

If the SAS Trusted User is unrestricted, then problems (such as the inability to launch certain servers) occur. Do not make the SAS Trusted User unrestricted.

Permission Patterns in Predefined ACTs

The following tables document the permission patterns of selected predefined ACTs within the metadata authorization layer. Each row documents the entire pattern for a particular ACT.

The predefined ACTs in the following table are used in the initial configuration and should not be modified.

Table A1.2 *Permission Patterns for Selected Predefined ACTs*

ACT Name	Denials	Grants		
	PUBLIC	SAS Administrators	SASUSERS	SSS ¹
Default ACT	All permissions	RM, WM, CM, A	RM, WM, CM	RM, WM
Private User Folder ACT	RM, WM, CM	RM, WM, CM	-	RM
SAS Administrators Settings	-	RM, WM, CM, A	-	RM

¹ SAS System Services (group)

For permission definitions and abbreviations, see [“Use and Enforcement of Each Permission” on page 84](#).

User Import Macros

<i>Overview of User Bulk Load and Synchronization</i>	317
<i>Canonical Tables</i>	321
<i>User Bulk Load</i>	323
Scope of the Import Process	323
How to Import Identities	323
<i>User Synchronization</i>	326
Scope of the Synchronization Process	326
How to Synchronize Identities	326
<i>Sample Code for User Synchronization</i>	328
<i>Sample Code for Generic Bulk Load</i>	330
<i>About the Sample Code for UNIX /etc/passwd</i>	332
<i>About the Sample Code for Active Directory</i>	334
<i>Location of the User Bulk Load and Synchronization Macros</i>	336
<i>Dictionary</i>	336
%MDUIMPC User Import Macro	336
%MDUIMPLB User Import Macro	338
%MDUEXTR User Import Macro	339
%MDUCMP User Import Macro	340
%MDUCHGV User Import Macro	342
%MDUCHGLB User Import Macro	343

Overview of User Bulk Load and Synchronization

In order to make access distinctions and track user activity, the metadata server has to know who is making each request. To enable the metadata server to make

this determination, each user's metadata definition includes that user's account ID from your authentication provider. The metadata server maintains its own copy of each ID. The metadata server does not maintain copies of external passwords for identification purposes.

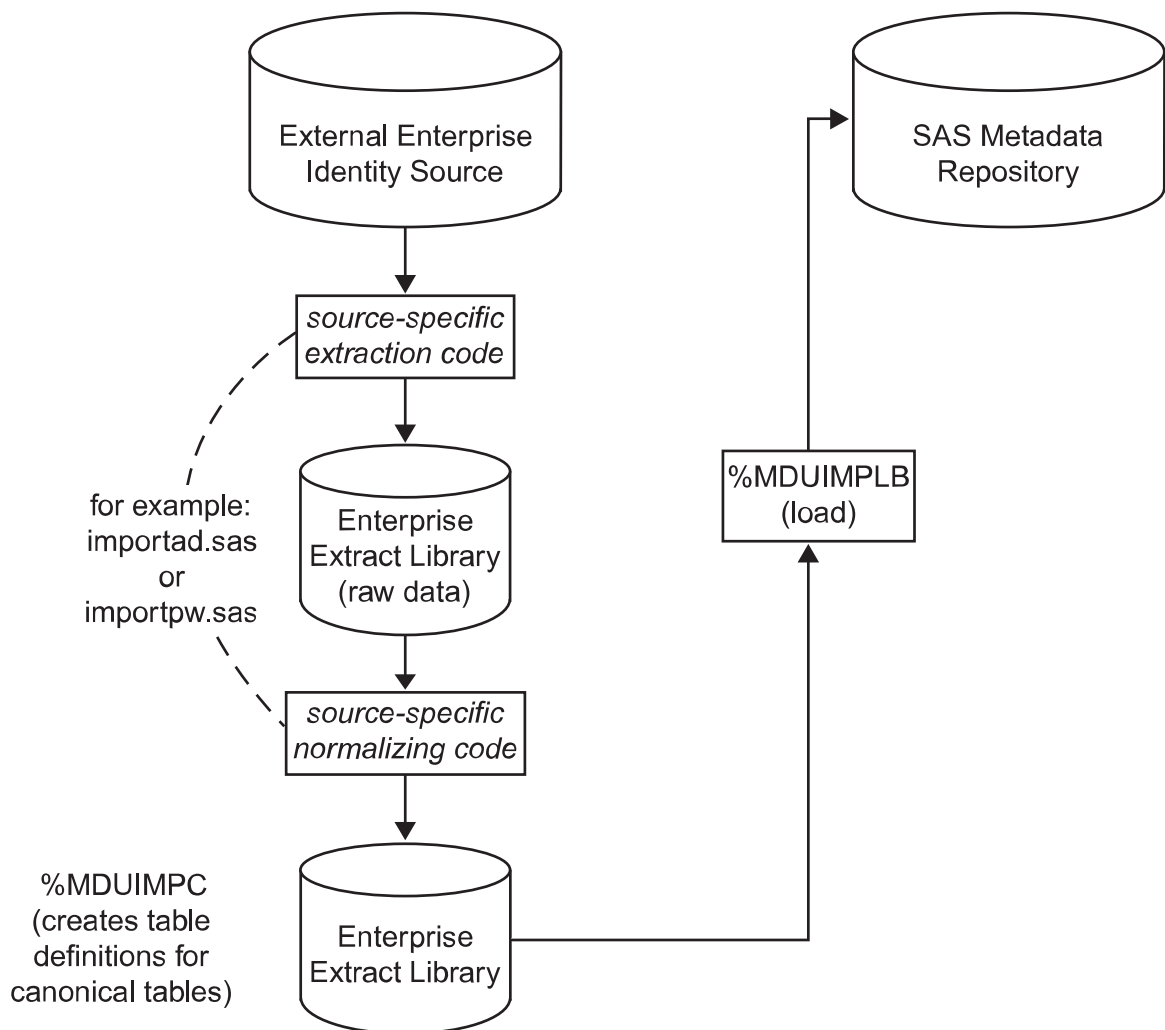
Note: For a few administrators, a SAS internal account can be used instead.

This chapter helps you use autocall macros and sample code that SAS provides to create your own programs that bulk load and manage user information. The chapter emphasizes coordination with an Active Directory server or UNIX `/etc/passwd` files but also provides information to help you extrapolate to other providers.

The following figures introduce the batch processes for identity information. In the figures, the `%MDU****` items are macros and the libraries contain SAS data sets.

The initial import extracts identity information from your authentication provider and loads that information into the metadata.

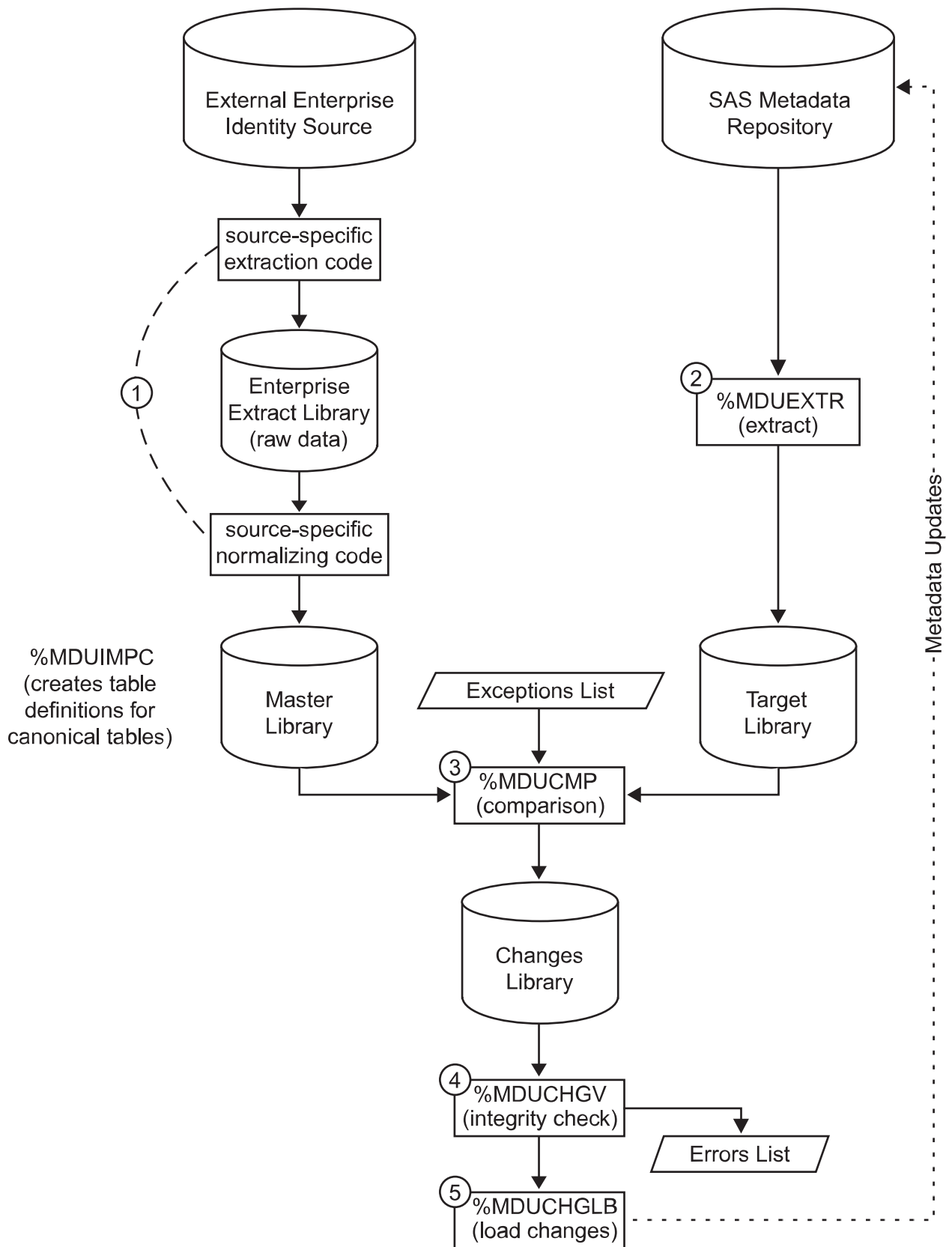
Figure A2.1 Initial Import (Bulk Load)



The synchronization performs two extractions (one from your authentication provider and another from the SAS metadata) and then loads validated updates into the metadata. The numbers in the following figure correspond to these activities:

- 1 Extract information from your authentication provider.
- 2 Extract information from the SAS metadata.
- 3 Compare the two sets of tables and identify updates that need to be made to the metadata (excluding any exceptions metadata that you want to preserve).
- 4 Validate the changes to make sure that they will not violate the metadata server's integrity constraints.
- 5 Load the updates into the metadata.

Note: Notice that the first part of the import process (the extraction from your authentication provider) is the same as the first part of the synchronization process. You will reuse your import extraction code in your synchronization program.

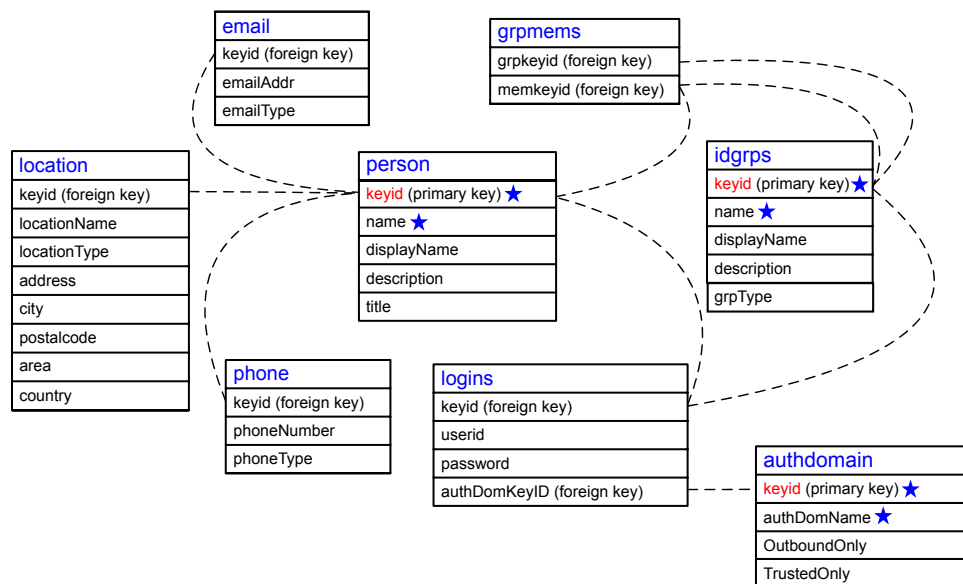
Figure A2.2 Periodic Synchronization

The following two topics document the format of the data sets and explain how corresponding identity entries are mapped between your authentication provider and the SAS metadata.

Canonical Tables

The following figure illustrates the names and structures for the tables in which identity information is stored during batch processing:

Figure A2.3 Table Structures and Relationships



Here are some key points about the tables:

- You do not have to use all of the tables. For example, if you are not going to store email addresses, you do not need an email table.
- For each table that you do use, all columns must be present. However, you do not have to include data in every column. In the figure, the required columns for the primary objects are indicated with a star.

Note: Each user should have a logon that includes a qualified user ID.

- The keyids in the person table (users), idgrps table (groups and roles), and authdomain table (authentication domains) tie each of those primary objects to its related information. In the metadata, the keyid value is stored as an external identity. For each keyid column, use a fixed, enterprise-wide identifier such as these:
 - In the person table, consider using employee identification numbers.

- In the idgrps table, consider using group names (or LDAP Distinguished Names).
- In the authdomain table, consider using authentication domain names.
- All of the relationships between a primary object and its related data are zero-or-more relationships. For example, you can store no phone numbers, one phone number, or multiple phone numbers for each user.
- We recommend that you avoid using spaces or special characters in the name of a user, group, or role. Not all components support spaces and special characters in identity names.

The following figure depicts data for a user named Tara O'Toole. The ovals indicate personal data for Tara. The check marks indicate data that is indirectly related to Tara (through her group memberships).

Figure A2.4 Example: Partial Tables Showing Selected User Data

person			grpmems		idgrps		
keyid	name	displayName	grpkeyid	memkeyid	keyid	name	grpType
0018	jhein	John Heinsler	Developers	Applications	Applications	Applications	
1390	hhigh	Harry Highpoint	Developers	ETL	Developers	Developers	
5107	totoo	Tara O'Toole	ETL	5107	ETL	ETL	
			ReportConsumers	Developers	Executives	Executives	
			ReportConsumers	Executives	ReportConsumers	ReportConsumers	

email		
keyid	emailAddress	emailType
0018	jhein@orionsports.com	work
0018	jh177@mail.net	home
1390	hhigh@orionsports.com	work
5107	totoo@orionsports.com	work
5107	otoole@hotmail.com	home

logins			authdomain	
keyid	userid	authdomKeyid	keyid	authDomain
0018	OrionSP\jhein	DefaultAuth	DB2Auth	DB2Auth
1390	OrionSP\hhigh	DefaultAuth	DefaultAuth	DefaultAuth
5107	OrionSP\totoo	DefaultAuth	OracleAuth	OracleAuth
5107	Tara	OracleAuth		
ETL	diadmin	DB2Auth		

In the figure, notice these things:

- In the person table, the employee number is used as the keyid. The name column uses user IDs (that column also requires unique values). The displayName column contains a common name for each user.
- In the grpmems table (group memberships), Tara is a direct member of the ETL group. Tara is an indirect member of the Developers group, because that group has the ETL group as one of its members. Tara also has a third-level membership in the ReportConsumers group.
- In the idgrps table, the group name serves as the keyid. This choice is appropriate because the metadata server enforces uniqueness across group and role names.
- In the idgrps table, the grpType column is empty. This indicates that the entries in this table are all groups. To create a role in the metadata, provide a grpType value of **ROLE**.

- In the email table, Tara has two email addresses and each one has a different type.
- In the logins table, Tara has personal logins in two different authentication domains. Tara can also use the ETL group's log on (for DB2Auth), because Tara is a member of the ETL group.
- In the authdomain table, the authentication domain name serves as the keyid. This is appropriate because the metadata server enforces uniqueness across authentication domain names.

User Bulk Load

Scope of the Import Process

Who Participates?

In order to participate in the initial import process, an identity must meet both of these criteria:

- The identity must be included in the import tables. If your identity information is distributed across several authentication providers or user registries, extract information from each source and then combine the resulting sets of tables into one set of canonical tables.

To limit the import tables, you can perform these tasks:

- Define a starting point. For example, when you extract identity information from Active Directory, you specify a Distinguished Name as the starting point. Only identities that exist below that Distinguished Name in the Active Directory hierarchy are extracted.
- Define filters. For example, when you extract identity information from Active Directory, you can use a filter to extract entries only for people who are members of a particular group.
- Make manual changes to the import tables.
- The identity must not already exist in the SAS environment. You cannot import an identity that has the same name as an identity that already exists in the metadata.

CAUTION

Do not delete existing SAS identities in order to include them in an initial import. When you delete a SAS identity, you lose that identity's associations (such as access controls). Creating a new identity with the same name does not restore those associations. You can incorporate a manually created identity into

the synchronization process. To do this, add an external identity on the **General** tab of that identity's metadata definition.

What Information Is Imported?

The import process can add this information to the metadata:

- user, group, and role definitions with names, display names, descriptions, and membership information
- job titles, contact information, and personal logons for users

Note: In most cases, passwords are not added to the metadata because they typically cannot be extracted from an authentication provider. If passwords are present in the extracted data, they are loaded into the metadata. It usually is not necessary to include passwords in logons.

Note: Synchronization can process logons for groups. The initial import process does not support these tasks.

- authentication domains

Note: Logons in outbound domains are excluded from all loading and synchronization updates.

These constraints apply to the initial import:

- When combined with information that already exists in the metadata, the input data must meet uniqueness requirements. For example, you cannot import an identity that has the same name as an identity that already exists in the metadata.
- In order to import a user, group, or role, only a name and one external identity value (keyid) is required. However, each user should also have at least one logon in order to establish an individual SAS identity.

Note: Avoid creating circular or reciprocal group memberships (for example, GroupA is in GroupB, GroupB is in GroupC, and GroupC is in GroupA). Such relationships introduce unnecessary complexity and can negatively impact performance. When you define nested group memberships (interactively or through importing identity information), the best practice is to use as simple and non-redundant a structure as possible.

How to Import Identities

Note: It is a good practice to run a backup before you perform an import.

To import identity information:

- 1 Locate the sample code that best fits your external identity source.
 - For import from Active Directory, see [“About the Sample Code for Active Directory” on page 334](#).
 - For import from UNIX /etc/passwd files, see [“About the Sample Code for UNIX /etc/passwd” on page 332](#).
 - For other formats, the first step is to figure out how to extract the data from your authentication provider. If you have LDAP, you might be able to modify the Active Directory sample for your purposes. Otherwise, use the %MDUIMPC macro to create empty canonical tables, and then use DATA steps to extract the information and insert it into those tables. See [“Sample Code for User Synchronization” on page 328](#).
- 2 Decide which attributes you want to add to the metadata. For each attribute, identify a corresponding field in your authentication provider.
- 3 In the SAS Program Editor, adapt the sample code. The comments in the sample code provide essential details.
- 4 Submit the code and review the log.
- 5 In the User Manager plug-in in SAS Management Console, verify that new identities exist. On the **General** tab of an imported user, group, or role, select **External Identities**. You should see an external identity value that matches the identity's keyid in the import tables.
- 6 Save a copy of your import program for inclusion in your synchronization program.

User Synchronization

Scope of the Synchronization Process

Who Participates?

By default, the synchronization process includes all identities that were originally imported into the metadata (because by default only those identities have an external identity in the metadata). You can modify participation in these ways:

- To include an identity that was manually created, add an external identity on the **General** tab of that user, group, or role definition.
- To exclude an identity that has an external identity in the metadata, define an exception in the %MDUCMP macro.

What Information Is Updated?

The synchronization process affects the metadata for participating identities as follows:

- Any identities that have been added to your authentication provider are added.
- Any new logons that have been added to your authentication provider are added. If passwords are extracted from your authentication provider, those passwords are included in the new logons.

Note: It is typically not possible to extract passwords from an authentication provider.

Note: Most logons do not need to include a password.

- Any participating identities that are not found in your authentication provider are deleted.
- By default, only logons that meet all of these criteria are deleted:
 - The logon belongs to a participating identity.
 - The logon does not exist in your authentication provider.

- ❑ The logon is in an authentication domain that exists in your authentication provider.

Note: Logons in authentication domains that do not exist in your authentication provider are preserved by default.

- If the synchronization includes groups (or roles), memberships for participating SAS identities are updated to match the memberships in your authentication provider. A change in the input grpType value (which determines whether an object is a group or a role) does not cause any update to the metadata.
- Locations, telephone numbers, and email addresses for participating identities are updated. Use exceptions to prevent updates to contact information that is added interactively.
- New authentication domains are added. By default, no authentication domains are removed.

Note: Logons in outbound domains are excluded from all loading and synchronization updates.

These constraints apply to updates:

- When combined with information that already exists in the metadata, the change data must meet all of the metadata server's uniqueness requirements. See [“Uniqueness Requirements” on page 27](#).
- In order to add a user, group, or role, only a name and one external identity value (keyid) is required. However, each user should also have at least one logon in order to establish an individual SAS identity.

How to Synchronize Identities

Note: It is a good practice to run a backup before you perform a synchronization (at least until your program is proven).

To synchronize identity information:

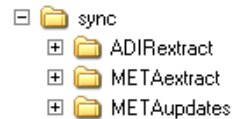
- 1 If you want to include identities that were not originally imported, use SAS Management Console to add a correct external identity value on the **General** tab of each such identity.
- 2 If you want to exclude identities or attributes from the update, create an exceptions data set.

CAUTION

If you used SAS Management Console to make updates to imported identities, those updates are not automatically preserved during batch

synchronization. To preserve such information, define exceptions in the %MDUCMP macro.

- 3 In the operating system, set up three directories: an enterprise extract directory, a metadata extract directory, and a change tables directory. For example:



- 4 In the SAS Program Editor, adapt the sample synchronization code to create your own program. See [“Sample Code for User Synchronization” on page 328](#).
- 5 Submit the code and review the log. To address any errors, make changes in the source tables, the exceptions tables, or the metadata. For details about the errors, examine the errorsds table. After making corrections, run the synchronization program again.

Note: An alternative method for dealing with errors is to re-execute the %MDUCMP macro with EXCEPTIONS=ERRORSDS. This re-creates the change tables without the offending entries. If an exceptions data set is already being used, you can append the content of the ERRORSDS data set to that data set.

- 6 In the User Manager plug-in in SAS Management Console, verify that the metadata reflects the changes that you expect to see.

Note: To ensure that current information is displayed, right-click **User Manager** and select **View** ⇒ **Refresh**.

Sample Code for User Synchronization

This example synchronization program is intended to help you complete step 4 in [“How to Synchronize Identities” on page 327](#).

Note: This example uses code that is stored in *SAS-installation-directory* \SASFoundation\9.4\core\sample\importad.sas (Windows) or *SAS-installation-directory*/SASFoundation/9.4/samples/base/importad.sas (UNIX). For more information about the importad.sas program, see [SAS Note 40628: Automating the addition of users and groups to a SAS Metadata Repository](#).

```

/*Specify connection options. Use an unrestricted user ID.*/
options metaserver=machine-name
metauser="userID"
metapass="password";
  
```



```

/*Specify the directory for the extracted AD data (master
tables).1*/
libname adir "drive:\path\ADIRextract";

/* Specify the directory for the extracted metadata (target tables).*/
libname meta "drive:\path\METAextract";

/* Specify the directory for the comparison output (change tables).*/
libname updates
"drive:\path\METAupdates";

/* Extract identity information from AD (master).2*/
%let _EXTRACTONLY = ;
%include "drive:\path\myimportad.sas";

/* Extract identity information from the metadata (target).*/
%mduextr(libref=meta);

/* Compare AD (master) to metadata (target).3*/
%mducmp(master=adir, target=meta, change=updates);

/* Validate the change tables.*/
%mduchgv(change=updates, target=meta, temp=work,
errorsds=work.mduchgverrors);

/* Load the changes into the metadata.4*/
%mduchglb(change=updates);

```

The numbers in the preceding code correspond to these points:

- 1 The AD extract location should match the importlibref location in your import program. For example, if your original importad.sas put the normalized AD data in the work library, you might revise importad.sas by replacing the line:

```
%let importlibref=work;
```

with these two lines:

```
libname adir "drive:\path\ADIRextract";
%let importlibref=adir;
```

- 2 This causes the importad.sas. program to extract only the data (and not perform the load).
- 3 If you need to exclude items from the comparison, add the EXCEPTIONS=dataset parameter here.
- 4 If you prefer to not load any changes if %MDUCHGV finds an integrity violation, replace the displayed line with this code:

```

%macro exec_mduchglb;
  %if (&MDUCHGV_ERRORS ^= 0) %then %do;
    %put ERROR: Validation errors detected by %nrstr(%mduchgv). Load not attempted.;
    %return;
  %end;
  %mduchglb(change=updates);
%mend;

```

```
%exec_mdurchlgb;
```

Sample Code for Generic Bulk Load

This code demonstrates one way to import generic identity information. This example is intended to illustrate the structure and relationships of the identity data, rather than to suggest that you enter large quantities of data using this approach. The program uses macro variables to define the tables and uses DATALINES statements to supply input data. The %MDUIMPLB macro creates XML from the tables and submits that XML (in blocks) to the metadata server.

```
/* Specify connection options. Use an unrestricted user ID. */
options metaserver=machine-name
metauser="userID"
metapass="password";

/* Initialize the macro variables that create canonical tables. */
%mduimpc();

/* Create the person table. */
data &persontbla ;
    %definepersoncols;
    infile datalines delimiter=',' missover;
    input keyid name description title;
    datalines;
P001,Michelle Harrell,Mgr of Operations,Sr. Mgr
P002,Fred Granite,NE Region Acct. Rep.,Account Rep II
P003,Brian Davis,Accounting System Developer,Senior Programmer
;

/* Create the phone table. */
data &phonetbla ;
    %definephonecols;
    infile datalines delimiter=',' missover;
    input keyid phoneNumber phoneType;
    datalines;
P001,x1532,Office
P001,(919) 555-1212,Home
P003,x2312,Office
;

/* Create the location table. */
data &locationtbla ;
    %definelocationcols;
    infile datalines delimiter=',' missover;
    input keyid locationName locationType address city postalcode
area
country;
    datalines;
P001,My Company,Office,123 Oak Ave,Clayton,20711,CA,USA
```

```

P001,Michelle Harrell,Home,105 Seth Ct.,Apex,20765,CA,USA
P002,Fred Granite,Home,2138 Pond St.,Greenlevel,20832,CA,USA
P002,My Company,Office,123 Oak Ave,Clayton,20711,CA,USA
P003,My Company,Office,123 Oak Ave,Clayton,20711,CA,USA
;

/* Create the email table. */
data &emailtbla ;
    %defineemailcols;
    infile datalines delimiter=',' missover;
    input keyid emailAddr emailtype;
    datalines;
P001,michelle@mycompany.com,business
P001,bosslady1@hotmail.com,home
P002,fred@mycompany.com,business
P003,brian@mycompany.com,business
;

/* Create the idgrp table. */
data &idgrptbla ;
    %defineidgrpcols;
    infile datalines delimiter=',' missover;
    input keyid name description grpType;
    datalines;
G001,Operations Staff,Members of the operations department,
G002,All Groups,Group containing all groups,
G003,Backup Operators, ,
;

/* Create the grpmems table. */
data &idgrpmemstbla;
    %defineidgrpmemscols;
    infile datalines delimiter=',' missover;
    input grpkeyid memkeyid;
    datalines;
G001,P001
G002,G001
G002,G003
G003,G001
G003,P003
;

/* Create the authdomain table. */
data &authdomtbla;
    %defineauthdomcols;
    infile datalines delimiter=',' missover;
    input keyid authDomName;
    datalines;
A001,DefaultAuth
A002,UnixAuth
;

/* Create the logins table. */
data &logintbla;
    %definelogincols;
    infile datalines delimiter=',' missover;

```

```

        input keyid userid password authdomkeyid;
    datalines;
P001,WinNet\Michelle, ,A001
P001,Michelle, ,A002
P002,WinNet\Fred, ,
P003,WinNet\Brian, ,
P003,Brian, ,A002
;

/* Load the information from the work library into the metadata
server. */
%mduimplb();

```

About the Sample Code for UNIX `/etc/passwd`

Note: This code is in `SAS-installation-directory\SASFoundation\9.4\core\sample\importpw.sas` (Windows) or `SAS-installation-directory\SASFoundation\9.4\samples\base\importpw.sas` (UNIX). This topic highlights key points about the code.

This program expects user information in `/etc/passwd` files in this form:

```

user name : password : uid (numeric) : primary group id : <continued>
gcoss-field: home-directory : login-shell

```

The `gcoss-field` item consists of additional comma-delimited fields in this form:

```

Person Name, Office, phone extension, misc, employeeid

```

For example:

```

user name : password : uid (numeric) : primary group id : <continued>
Person Name, Office, phone extension, misc, employeeid:<continued>
home-directory : login-shell

```

Here are details about this code:

- The colons delimit standard fields; the commas delimit items within the `gcoss-field` (you can use any delimiter other than a colon).
- If your `/etc/passwd` file has a different format, modify the sample program to either exclude information from the extraction or to extract it from the appropriate fields.
- If the `employeeid` field for an `/etc/passwd` entry is empty, that entry is dropped from the import.
- The values in the `Person Name` field should be unique. The program includes a `DUPLICATEPERSONS` macro variable that, when set to `RECODE`, changes a duplicate value in the `Person Name` field to the user ID value of the `PW` file

entry. However, if the `DUPLICATEPERSONS` macro variable is set to any other value, users with duplicate names in the `Person Name` field are deleted.

The program expects group information in a `/etc/group` file. The standard format is as follows:

```
groupname : password : gid (numeric) : comma-delimited list of users
```

Here are details about this file:

- Users are identified by user name (rather than by a numeric user ID).
- A group cannot be a member of another group.
- To exclude a user or group, enter an asterisk (*) in the password field. The import program drops entries that contain an asterisk in the password field.

The following table highlights selected variables:

Table A2.1 Selected Variables in a UNIX File Import

Variable Name	Purpose	Notes
UID	Provides an external identity value for each metadata user and group that this program creates.	Use a field that contains a unique and unchanging value for each entry in an <code>/etc/passwd</code> file and an <code>/etc/group</code> file. In the sample code, the <code>uid</code> field of the <code>gcos</code> -field is used. If this field is empty for any records, those records are excluded from the import.
MetadataAuthDomain	Enables all metadata logons that this program creates to be associated with an authentication domain.	Specify a SAS authentication domain name. The supplied value is typically <code>DefaultAuth</code> .
UNIXEMAILDOMAIN	Enables construction of an email address for each user.	Specify the UNIX domain for the extracted identities. The supplied value is appended to extracted user IDs to yield email addresses in the form <code>userid@supplied-value</code> . If your email addresses follow a different convention, modify this part of the code.
PWExtIDTAG	Provides a label for all metadata items that this program creates. The label indicates which objects were created by this program.	Specify a descriptive label that will be applied to all imported objects to indicate where they came from. The default value is <code>Password File Import</code> . Do not quote this value. If you select the External Identities button on an imported identity's General tab (in SAS Management Console), you will see this label in the Context column of the External Identities dialog box.

About the Sample Code for Active Directory

Note: This code is in `SAS-installation-directory\SASFoundation\9.4\core\sample\importad.sas` (Windows) or `SAS-installation-directory/SASFoundation/9.4/samples/base/importad.sas` (UNIX). This topic highlights key points about the code.

Here are some tips for using the program:

- The code uses the SAS interface to LDAP (the LDAP CALL Routine interface) to extract information from Active Directory.
- The code references standard Active Directory schemas to identify user and group attributes. If your site has extended the standard schema, you might need to make changes in section 3 to reference additional or alternate attributes.
- The code uses filters to segment retrieval. It might be necessary to alter the filters in sections 3 (user extraction) and 4 (group extraction) to better fit the contents of your Active Directory server. If the number of records in a request exceeds Active Directory's maximum query limit, only a subset of the requested records is returned. The Microsoft utility program LDIFDE can be useful in defining appropriate filters.
- Matching rule object identifiers (OIDs) are not supported in the filter argument of the LDAPS_SEARCH CALL routine. The use of an OID causes the following error: `ERROR: Unable to contact the LDAP server.`
- If either or both of the data sets that are extracted from Active Directory are empty, execution of the code is canceled and an error message is provided. This reduces the likelihood of inadvertent deletion of metadata identities due to a problem with the Active Directory extraction during the synchronization process.

Note: Execution is canceled if no users are extracted, and execution is canceled if no groups are extracted. If you want to extract only users from Active Directory, you must modify the 9.4 sample code. Either disable the error check or remove the entire section of the code that extracts groups.

- The code will not import membership information for a group that has more than 1500 members. (This limitation is version-specific. Check the documentation for your Active Directory server for details). To incorporate an oversize group, use an approach like this:

- 1 Rewrite the section 4 filters to exclude large groups.

- 2 Create an additional extraction that uses LDAP range retrieval specifiers to extract the large group membership information in segments. See the LDAPS_SEARCH CALL routine in the *SAS Integration Technologies: Directory Services Reference*.
 - 3 Add that membership information to the main extracted grpmems table.
- Referrals that require authentication are not supported. If a referral is encountered, the anonymous bind is used on the referred-to object. The anonymous bind must be allowed on the LDAP server. Otherwise, the bind fails.
 - If the search crosses domains in a multi-domain forest, use the Active Directory Global Catalog. Search is handled by Active Directory without the need for SAS to follow referrals.

The following table highlights selected macro variables:

Table A2.2 Selected Macro Variables in an Active Directory Import

Variable Name	Purpose	Notes
keyidvar	Provides an external identity value for each metadata user that this program creates.	Specify an LDAP attribute that contains a unique and unchanging value for each user. The sample code uses the employeeID attribute. ¹
MetadataAuthDomain	Enables all metadata logons that this program creates to be associated with an authentication domain.	Specify a SAS authentication domain name. This value is not related to a Windows domain name. In the standard configuration, the correct value is DefaultAuth (or, if you have configured web authentication and are extracting information for users who use only web clients, web).
WindowsDomain	Enables construction of a qualified user ID in each logon that this program creates.	Provide the Windows domain name for the extracted identities. The supplied value is prepended to each extracted user ID to yield qualified IDs in the form <i>supplied-value\userID</i> .
ADExtIDTag	Provides a label for all metadata items that this program creates. The label indicates which objects were created by this program.	Specify a descriptive label that will be applied to all imported objects to indicate where they came from. The default value is Active Directory Import . Do not quote this value. If you select the External Identities button on an imported identity's General tab (in SAS Management Console), you will see this label in the Context column of the External Identities dialog box.

¹ If this attribute is empty, consider using sAMAccountName or distinguished name.

Location of the User Bulk Load and Synchronization Macros

The user import and synchronization macros are in *SAS-installation-directory* \SASFoundation\9.4\core\sasmacro (Windows) or *SAS-installation-directory*/SASFoundation/9.4/sasautos (UNIX). The following topics provide basic reference information that helps you use these macros to perform standard user import and synchronization tasks.

Dictionary

%MDUIMPC User Import Macro

Defines a set of macro variables that create canonical tables and, optionally, creates empty tables for views for CSV files.

Used by: User import, user synchronization

Syntax

%MDUIMPC (<LIBREF=*libref*>, <MAKETABLE=0 | 1 | 2>, <INFILEREFS=*fileref*>, <FILEHEADER=1 | *other-value*>);

Optional Arguments

LIBREF

specifies the output location for any canonical tables or views that are created (the default is **work**). If you choose to not create tables or views, this parameter is not used.

MAKETABLE

controls whether any tables or views are created.

- 0 specifies that no tables or views are created (this is the default).
- 1 specifies that empty canonical tables are created.

- 2 specifies that DATA step views of the CSV files in the fileref directory are created.

INFILEREF

specifies the location of the input CSV source files (used only when MAKETABLE=2). The CSV input files must have names and contents that match the table names and structures of the canonical tables. On UNIX, the filenames must be lowercase.

FILEHEADERS

indicates whether the CVS input files have a header line (used only when MAKETABLE=2).

- 1 indicates that the input files have header information in the first row.

other_value indicates that the data begins in the first row.

Details

Table A2.3 Tables and Macro Variables Created by %MDUIMPC

Table	Purpose	Name/Keep List Global Variable	Column Attributes Macro
person	Create users	&persontbla;	%definepersoncols
location	Store addresses	&locationtbla;	%definelocationcols
phone	Store phone numbers	&phonetbla;	%definephonecols
email	Store e-mail addresses	&emailtbla;	%defineemailcols
idgrps	Create groups and roles	&idgrptbla;	%defineidgrpcols
grpmems	Store membership information	&idgrpmemstbla;	%defineidgrpmemscols
authdomain	Create SAS authentication domains	&authdomtbla;	%defineauthdomcols
logins	Create logins	&logintbla;	%definelogincols

%MDUIMPLB User Import Macro

Generates and submits blocks of XML to load identity information.

Used by: User import

Requirement: Connection to the metadata server

Syntax

```
%MDUIMPLB (<LIBREF=libref>, <TEMP=libref>, <FAILEDOBJs=table-name>,
<EXTIDTAG=context-value>, <BLKSIZE=number-of-records>);
```

Optional Arguments

This macro uses PROC METADATA to submit load requests.

LIBREF

specifies the location of the input data. The default is `Work`.

TEMP

specifies the location for temporary data sets during processing. The default is `Work`.

FAILEDOBJs

is a table that contains the entire contents of all blocks that contain any problematic data. The default is `mduimplb_failedobjs`.

This table is populated if integrity and completeness requirements are not met or if a metadata server request failure occurs during processing.

EXTIDTAG

indicates the origin of identity information. The default for this string is `IdentityImport`.

BLKSIZE

specifies the number of identities in a block. Each block of XML is generated and submitted to the metadata server independently. The default is 100.

Details

Table A2.4 *FAILEDOBJs*

Column	Contents
tablename	Name of an intermediate input table

Column	Contents
chgtype	A (add), U (update), or D (delete)
objtype	Internal metadata object type
Name	Name of a user, group, or role
Objid	Internal object ID
KeyId	Key ID for an item
Userid	User ID value in a login
AuthDomKeyId	Key ID of an authentication domain
MemObjType ¹	Internal metadata object type for a member
MemName ¹	Person indicates a user, IdentityGroup indicates a group
MemObjID ¹	Internal metadata object ID
MemKeyId ¹	Key ID of a member

¹ This column is populated only if there is an error in membership information.

Note: Logins in outbound domains are excluded from all loading and synchronization updates.

%MDUEXTR User Import Macro

Extracts identity information from the metadata.

Used by: User synchronization

Requirement: Connection to the metadata server

Syntax

%MDUEXTR ([LIBREF=SAS-library](#));

Required Argument

This macro uses PROC METADATA to submit extraction requests.

LIBREF

specifies the location to which the metadata is extracted.

%MDUCMP User Import Macro

Generates data sets that contain the changes that must be made to the metadata.

Used by: User synchronization

Syntax

```
%MDUCMP (MASTER=libref, TARGET=libref, CHANGE=libref,  
<EXCEPTIONS=<libref.>data-set>, <EXTERNONLY=0 | 1>,  
<AUTHDOMCOMPARE=name | keyid>);
```

Required Arguments

MASTER

specifies the location of the master tables (use the libref that you specify in %MDUIMPC).

TARGET

specifies the location of the target tables that contain information extracted from the metadata (use the libref that you specify in %MDUEXTR).

CHANGE

specifies the location for the change tables. These tables are created (xxx is the base name of each canonical table):

xxx_add	contains users, groups, and roles to be added to the target tables to make them look like the master tables.
xxx_update	contains users, groups, and roles to be modified in the target tables to make them look like the master tables.
xxx_delete	contains users, groups, and roles to be deleted from the target tables to make them look like the master tables.
person_summary	summarizes changes to users (Person objects).
idgrps_summary	summarizes changes to groups and roles (IdentityGroup objects).
authdomain_summary	summarizes changes to SAS authentication domains (AuthenticationDomain objects).

Optional Arguments

EXCEPTIONS

specifies a data set that contains exception values.

EXTERNONLY

defines the scope of the comparison. Unless the master data set has an ObjectID column, this option has no effect. A typical master data set does not include an ObjectID column. A master data set that is extracted from the SAS Metadata Repository (rather than from your authentication provider) does include an ObjectID column. Extraction of a master data set from the metadata repository happens in the identity synchronization processes for some solutions.

- 1 specifies that only identities that have an external identity value are included in the comparison. This is the default value.
- 0 specifies that all identities are included in the comparison.

If EXTERNONLY=1 but AUTHDOMCOMPARE=NAME, all authentication domains are compared. In other words, for authentication domains AUTHDOMCOMPARE=NAME overrides EXTERNONLY=1.

AUTHDOMCOMPARE

defines how authentication domains are compared.

- NAME** compares all authentication domains by name. Prevents deletion and renaming of all authentication domains. Prevents deletion of logins in authentication domains that do not exist in the master data set. This is the default.
- KEYID** compares by keyid. Can cause deletion of authentication domains that were originally imported but are not present in the master data set. Can cause renaming of authentication domains that were originally imported but have a different name in the master data set. Does not prevent deletion of logins in authentication domains that do not exist in the master data set. If you specify AUTHDOMCOMPARE=KEYID, authentication domains and logins that are interactively created might be deleted (for a standard synchronization, do not use AUTHDOMCOMPARE=KEYID).

Details

The exceptions data has these columns:

tablename

specifies the name of the canonical table to which the exception applies. Valid values are **person**, **logins**, **email**, **phone**, **location**, **idgrps**, **grpmems**, and **authdomain**.

filter

specifies a SAS WHERE clause expression (without the WHERE) to apply against the corresponding table. The WHERE clause consists of a canonical table column name and an exception value.

For example, consider this exceptions data set:

```
phone      PhoneType="manual Phone"
email      EmailType="manual Email"
logins     authDomKeyId="A002"
logins     userid="testid%"
```

Each line protects a set of objects in a particular target table, ensuring that those metadata objects are preserved.

- The first entry excludes objects in the target phone table that have a PhoneType of **manual Phone**. In SAS Management Console, the PhoneType is displayed in the **Type** field in the Phone Properties dialog box.
- The second entry excludes objects in the target email table that have the EmailType value **manual Email**. In SAS Management Console, the EmailType is displayed in the **Type** field in the Email Properties dialog box.
- The last entry excludes any objects in the target logins table that have a userid value that begins with **testid**.

Note: Logins that are in authentication domains that do not exist in the master tables are preserved by default. It is not necessary to define exceptions for such logins.

Note: Logins in outbound domains are excluded from all loading and synchronization updates.

%MDUCHGV User Import Macro

Checks the change tables against the target tables to ensure that the updates do not introduce any integrity problems.

Used by: User synchronization

Syntax

```
%MDUCHGV (TARGET=libref, <CHANGE=libref>, <TEMP=libref>,
<ERRORSDS=name>);
```

Required Argument

TARGET

specifies the location of the target canonical tables. This is typically the same libref that you specify in the %MDUEXTR macro.

Optional Arguments

CHANGE

specifies the location of the change tables (that were populated by the %MDUCMP macro).

TEMP

specifies the location for temporary tables (the default is `Work`).

ERRORSDS

identifies a data set that contains information about any integrity problems. This data set has these columns:

<i>errcode</i>	specifies a numeric code for a particular error.
<i>errmsg</i>	describes a particular error.
<i>tablename</i>	specifies the name of the canonical table from which a particular error item should be excepted if the ERRORSDS data set is fed into the %MDUCMP macro as the exception data set.
<i>filter</i>	specifies a SAS WHERE clause that is used to remove all objects related to a particular error from the change tables.
<i>Keyid</i>	specifies the keyid value of the offending object.
<i>Name</i>	specifies the Name value of the object, if the offending object is a Person or IdentityGroup.
<i>userid</i>	specifies the userid value of the object, if the offending object is a Login.
<i>authDomKeyid</i>	specifies the keyid value of an authentication domain when duplicate userid values are found in an authentication domain.

If any errors are detected, this macro sets the MDUCHGV_ERRORS column to a nonzero value and creates the ERRORSDS data set, which lists the errors that were found.

Note: Logins in outbound domains are excluded from all loading and synchronization updates.

%MDUCHGLB User Import Macro

Generates XML for loading identity updates.

Used by: User synchronization

Requirement: Connection to the metadata server

Syntax

%MDUCHGLB (*CHANGE=libref*, <*TEMP=libref*>, <*FAILEDOBJS=table-name*>, <*EXTIDTAG=context-value*>, <<*BLKSIZE=number-of-records*>>);

Required Argument

CHANGE

specifies the location of the change tables.

Optional Arguments

TEMP

specifies the location for temporary tables. The default is `Work`.

FAILEDOBJS

is a table that contains the entire contents of any blocks that were not successfully loaded into the metadata. An error within a block prevents loading of all of the data in that block. The default name for this table is `mduchglb_failedobjs`.

This table is populated if integrity or completeness requirements are not met or if a metadata server request failure occurs during processing. To address errors, see [“How to Synchronize Identities” on page 327](#).

EXTIDTAG

Indicates the origin of identity information. The default for this string is `IdentityImport`.

BLKSIZE

specifies the number of added or updated objects in a block. Each block of XML is generated and submitted to the metadata server independently. The default is 100.

For Delete operations, `BLKSIZE` has no effect.

Note: Logins in outbound domains are excluded from all loading and synchronization updates.

Security Report Macros

<i>Overview of the Security Report Macros</i>	345
<i>Authorization Data Sets</i>	347
<i>Additional Resources for Building Authorization Data Sets</i>	350
<i>Dictionary</i>	351
%MDSECDs Security Report Macro	351

Overview of the Security Report Macros

Authorization reporting creates a snapshot of metadata layer access control settings and uses that snapshot in these ways:

- as a data source for reports about current settings
- as a data point for comparing settings across time

The first task in security reporting is to extract, filter, and format authorization data for a specified set of identities, permissions, and objects. SAS provides macros that help you perform this task. For example, the following code uses the main security report macro, %MDSECDs, to generate authorization data sets for a specified folder and its contents:

```
/* connect to the metadata server */
options
  metaserver=machine-name
  metauser="sasadm@saspw"
  metapass="{SAS002}3CD4EA1E35CA49324AOC4D63";

/* run the main report macro against a target folder */
%mdsecds(folder="\demo");
run;
```

The second task in security reporting is to build reports that run against the authorization data sets. For example, the following code prints some of the data from the preceding example:

```
/* print the WriteMetadata information */
proc print data=work.mdsecds_join noobs;
var objname publictype identityname WriteMetadata;
run;
```

The output looks like this:

ObjName	PublicType	identityname	WriteMetadata
Sample: Hello World	StoredProcess	Backup Operators	Granted Indirectly
Sample: Hello World	StoredProcess	PUBLIC	Denied Indirectly
Sample: Hello World	StoredProcess	SAS System Services	Denied Indirectly
Sample: Hello World	StoredProcess	SASAdministrators	Denied Indirectly
Sample: Hello World	StoredProcess	SASUSERS	Denied Indirectly
Sample: Shoe Sales	StoredProcess	Backup Operators	Granted Indirectly
Sample: Shoe Sales	StoredProcess	PUBLIC	Denied Indirectly
Sample: Shoe Sales	StoredProcess	SAS System Services	Denied Indirectly
Sample: Shoe Sales	StoredProcess	SASAdministrators	Denied Indirectly
Sample: Shoe Sales	StoredProcess	SASUSERS	Denied Indirectly
demo	Folder	Backup Operators	Granted by ACT
demo	Folder	PUBLIC	Denied Explicitly
demo	Folder	SAS System Services	Denied Indirectly
demo	Folder	SASAdministrators	Denied Indirectly
demo	Folder	SASUSERS	Denied Indirectly

Because the output is SAS data sets, you can create sophisticated reports by using SAS reporting techniques such as these:

- create an information map that uses an authorization data set as its data source and then use SAS Web Report Studio to create reports based on that information map.
- write SAS ODS (output display system) code that builds an HTML report against the authorization data sets. For an example, see *SAS-installation-directory\SASFoundation\9.4\core\sample\secrpt.sas* (Windows) or *SAS-installation-directory\SASFoundation\9.4\samples\base\secrpt.sas* (UNIX).

For details, options, and examples, see [“%MDSECDs Security Report Macro” on page 351](#).

Authorization Data Sets

For reference, this topic documents the structure of the generated tables.

Note: The output columns for Permissions will have underscores instead of spaces.

Table A3.1 *work.mdsecds_join (the Primary Table for Reporting; Combines objs, permsw, pconds)*

Column Name	Column Contents
IdentityDispName	The display name of this user or group (for example, SAS Demo User). If there is no display name, this column contains the same value as the IdentityName column.
IdentityName	The name of this user or group (for example, sasdemo).
IdentityType	Person, IdentityGroup, or Role.
Location	A container path for this object (for example, \Shared Data\Sales). The path usually consists of folder names but can also include names of other containers.
MetadataCreated	The date on which this object was created.
MetadataType	The internal metadata type for this object.
MetadataUpdated	The most recent date that this object was updated.
ObjId	The metadata ID for this object (for example, A5HDAJSI.B900066J).
ObjName	The name of this object (for example, Shoe Sales by Region).
ObjUri	The uniform resource identifier for this object (for example, omsobj:PhysicalTable/A5XT9KUX.B80000001).
ParentObjId	The metadata ID for this object's immediate parent.
Permissions	A list of the permissions to inspect for this object. If the list is blank, then all permissions that are applicable to this type of object are inspected.
(permissions)	A separate column for each permission. Each cell in these columns contains a value that indicates whether this permission is effectively granted or denied for this

Column Name	Column Contents
	identity. The type of setting (explicit, ACT, or indirect) is also indicated (for example, Granted by ACT or Denied Explicitly or Granted Indirectly).
PublicType	The public metadata type for this object.

Table A3.2 *work.mdsecds_obj* (Contains Folder and Member Information)

Column Name	Column Contents
Desc	Description information for this object. For most objects, this information comes from the Description field on the General tab in the object's properties dialog box.
Location	A container path for this object. The path usually consists of folder names but can also include names of other containers.
MetadataCreated	The date on which this object was created.
MetadataType	The internal metadata type for this object.
MetadataUpdated	The most recent date that this object was updated.
ObjId	The metadata ID for this object.
ObjName	The name of this object.
ObjUri	The uniform resource identifier for this object.
ParentObjId	The metadata ID for this object's immediate parent.
Permissions	A list of the permissions to inspect for this object. If the list is blank, then all permissions that are applicable to this type of object are inspected.
PublicType	The public metadata type for this object.

Table A3.3 *work.mdsecds_pconds* (Contains Permission Condition Expressions)

Column Name	Column Contents
Condition	The expression that limits this grant.
IdentityName	The name of this user or group.
IdentityType	Person , IdentityGroup , or Role .

Column Name	Column Contents
ObjUri	The uniform resource identifier for this object.
Permission	The name of this permission (for example, Read).

Table A3.4 *work.mdsecds_permsw (Contains Permissions Data Transformed to Wide Format)*

Column Name	Column Contents
IdentityDispName	The display name of this user or group (or the name if there is no display name).
IdentityName	The name of this user or group.
IdentityType	Person, IdentityGroup, or Role.
ObjName	The name of this object.
ObjUri	The uniform resource identifier for this object.
(permissions)	A separate column for each permission. Each cell in these columns contains a value that indicates whether this permission is effectively granted or denied for this identity. The type of setting (explicit, ACT, or indirect) is also indicated.

Table A3.5 *work.mdsecds_permsl (Contains Permissions Data in Original Long Format)*

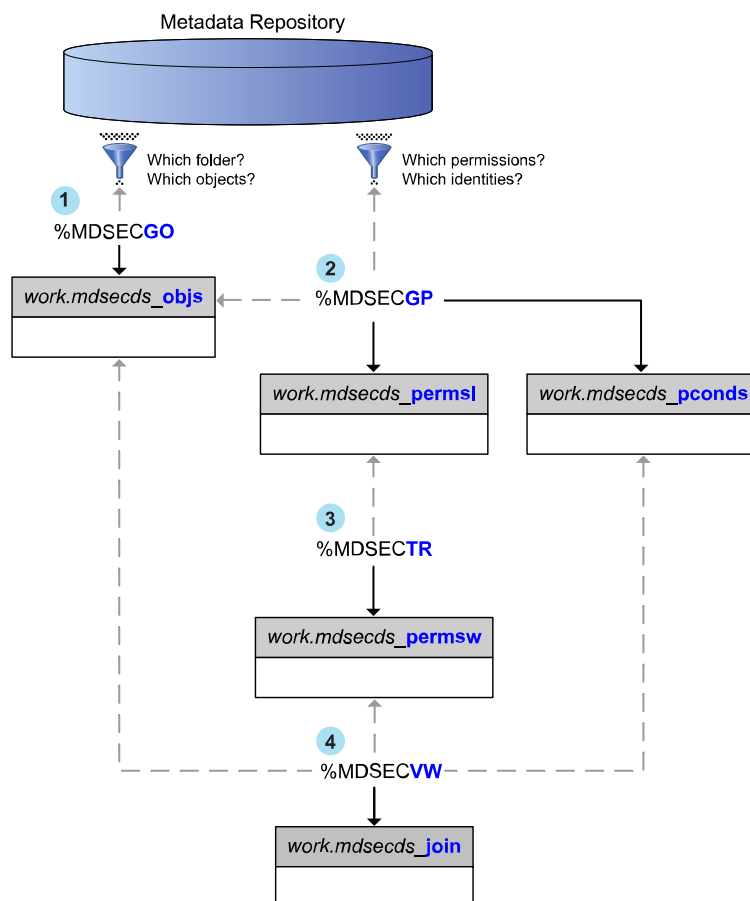
Column Name	Column Contents
Authorization	A value that indicates where this permission is granted or denied for this identity. The type of setting (explicit, ACT, or indirect) is also indicated.
Conditionflg	A value that indicates whether this permission setting is conditional. The value is either blank (no condition) or * (condition applies and is written to the pconds table).
IdentityDispName	The display name of this user or group (or the name if there is no display name).
IdentityName	The name of this user or group.
IdentityType	Person, IdentityGroup, or Role.
ObjName	The name of this object.
ObjUri	The uniform resource identifier for this object.
Permission	The name of this permission.

Additional Resources for Building Authorization Data Sets

This topic introduces the underlying security report macros.

You can choose to directly use the underlying report macros instead of using only %MDSECDs (which is the standard approach to building authorization data sets). However, the underlying macros don't offer any unique inclusion or exclusion parameters. The following figure introduces the underlying macros. In the figure, arrow direction indicates input to and output from each underlying macro.

Figure A3.1 Underlying Macros



The numbers in the preceding figure correspond to these activities:

- 1 %MDSECGO extracts information for a specified set of objects. You specify one folder and indicate whether to include subdirectories. You can also provide a list of object types to include and filter the data set by attribute value.

Note: The same level of control is provided by using %MDSECDs on its own.

- 2 For every object in a specified data set, %MDSECGP gets effective permission settings for a specified set of identities and permissions.
-

Note: This is the point at which using the underlying macros creates an opportunity for you to define a subset of the objs data set.

- 3 %MDSECTR transforms the extracted data set from a long format (a separate row for each permission) to a wide format (all permissions in the same row).
- 4 %MDSECVW creates a joined view or data set that can be used for reporting.

The following table introduces a few utility macros that can be useful in security reporting:

Table A3.6 *Utility Macros for Security Reporting*

Utility Macro	Description
%MDUTYPED	Extracts information about top-level metadata objects or locates templates for a particular object type.
%MDUGFLDR	Returns the object ID for a specified folder.
%DEFINEOBJTAB_SQL	Defines the table into which %MDSECGO inserts rows.

For more information, see the macros in *SAS-installation-directory\SASFoundation\9.4\core\sasmacro* (Windows) or *SAS-installation-directory/SASFoundation/9.4/sasautos* (UNIX).

Dictionary

%MDSECDs Security Report Macro

Generates authorization data sets. This is the top-level macro (it calls the underlying macros and should be used on its own).

Used by: Security reporting

Type: Stand-alone

Requirement: Connection to the metadata server

Syntax

%MDSECDS

```
(<OUTDATA=data-set>,
<FOLDER="path">,
<INCLUDESUBFOLDERS= | >,
<INCLUDETABLECOMPONENTS= | >,
<INCLUDECUBECOMPONENTS= | >,
<INCLUDESECUREDTABLES= | >,
<MEMBERTYPES="list">, <MEMBERFILTER="expression">,
<PERMS="list">,
<IDENTITYNAMES="list">, <IDENTITYTYPES="list">);
```

Optional Arguments

OUTDATA

provides a base name for the output. By default, the base name is `work.mdsecds`.

FOLDER

identifies a starting point folder. By default, the starting point is the server root (the **SAS Folders** node). If you provide a path (such as "`\Products\SAS Intelligence Platform\Samples`"), the starting point is the last folder in the path. To avoid having to type a long pathname, copy the path from a child item's general properties into your code.

INCLUDESUBFOLDERS

controls whether the entire subtree is included. By default, all objects in the entire subtree are included. If you specify **NO**, only the immediate contents of the starting point folder are included.

INCLUDETABLECOMPONENTS

controls whether table columns are included when a table is returned. By default, columns are included.

INCLUDECUBECOMPONENTS

controls whether cube hierarchies, levels, and dimensions are included when a cube is returned. By default, these components are included.

INCLUDESECUREDTABLES

controls whether secured table objects are included when a secured library object is returned. By default, secured table objects are included.

MEMBERTYPES

limits by object type. By default, all public types are included. If you provide a comma-delimited list of types, only those types are included.

You must provide the public type name in its TypeName format. For example, if you access the **Advanced** tab in the properties dialog box for the **Information Map (relational)** object type, you will see that its TypeName is `InformationMap.Relational`.

TIP In SAS Management Console, all public types are displayed on the **Folders** tab under **System** ⇒ **Types**.

Note: If you use this option, examine the output. The log doesn't display errors or warnings for incorrectly specified types.

MEMBERFILTER

limits by metadata attribute value. By default, no filter is applied. If you provide an expression, only objects that meet the criteria are included. The format for the expression is *@attribute-name comparison-operator 'value'* (for example, `@ID='A5HDAJSI.B90006Y5'` or `@Name=: 'Salary'`).

Comparison operators for character data include = (equals), =: (begins with), ? (contains), and ne (not equals).

Metadata attributes are associated with an object's metadata type (not public type). Here are two common attributes:

ID is the object's metadata ID, which is displayed on the object's **Inheritance** tab.

Name is the object's name, which is displayed in the **Name** field on the object's **General** tab.

To find additional attributes, determine the MetadataType of the object that you are interested in. The type is displayed on the object's **Inheritance** tab. Then, in the reference documentation for the metadata model, look up that MetadataType to find the names of its attributes.

PERMS

specifies which permissions to include. By default, all permissions that are supported for each object type are included for objects of that type. If you provide a comma-delimited list of permissions, only those permissions are included. Even if the permissions in your user interface are translated, you must specify the English-language permission names (for example, ReadMetadata).

IDENTITYNAMES

specifies which identities to include. By default, only the named participants (the identities that are listed in an object's authorization properties) are included. If you provide a comma-delimited list of identity names, only those identities are included. List identities by their names, not their display names.

If you use this option, you must also use the IDENTITYTYPES option.

IDENTITYTYPES

specifies whether names in the IDENTITYNAMES list correspond to users or to groups. For example, the first name listed in the IDENTITYNAMES parameter must match the first value in the IDENTITYTYPES list. Valid values in this list are **Person** and **IdentityGroup**.

Examples

Example 1: Permissions for All Objects within a Folder

This code extracts information about permissions on the objects in the Sales folder but doesn't include objects in subfolders:

```
%mdsecds(folder="\Shared Data\Sales", includesubfolders=no);
```

Example 2: Permissions for Two Object Types within a Folder Branch

This code extracts information about permissions on tables and schemas in the Sales folder and its subfolders:

```
%mdsecds(folder="\Shared Data\Sales", membertypes="Library,OLAPSchema");
```

Example 3: ReadMetadata Permission for Libraries for a Specified User

This code extracts information that indicates which libraries a particular user (the SAS Demo User) can see:

```
%mdsecds(identitynames="sasdemo", identitytypes="Person", membertypes="Library",  
perms="ReadMetadata");
```

Example 4: ReadMetadata Permission for Stored Processes for Two Specified Users

This code extracts information that indicates which stored processes two users (the SAS Demo User and Tara O'Toole) can see:

```
%mdsecds(identitynames="sasdemo,totoo", identitytypes="Person,Person",  
membertypes="StoredProcess", perms="ReadMetadata");
```

Example 5: WriteMetadata Permission for Reports for Specified Identities

This code extracts information that indicates which reports one user and one group (the SAS Demo User and PUBLIC) can modify:

```
%mdsecds(identitynames="sasdemo,PUBLIC", identitytypes="Person,IdentityGroup",  
membertypes="Report", perms="WriteMetadata");
```

Example 6: ReadMetadata Permission for a Subset of Reports

This code extracts information that indicates who can view reports that include the word "Salary" in their names:

```
%mdsecds(membertypes="Report", perms="ReadMetadata", memberfilter="@Name ? 'Salary'");
```

Example 7: Permissions for an Object (Referenced by Object ID)

This code extracts permission settings for an object that is referenced by its object ID:

```
%mdsecds(memberfilter="@ID='A5HDAJSI.B90006Y5'");
```

Note: Member filters are not applied to folders, so this example returns all folders (in addition to the object that has the specified ID).

Appendix 4

Choices in Workspace Server Pooling

<i>About Workspace Server Pooling</i>	357
<i>Benefits and Risks of Server-Side Pooling</i>	358
<i>Which Requests Are Eligible to Use Pooling?</i>	360
<i>Which Eligible Requests Actually Use Pooling?</i>	361
<i>Modifying the Initial Pooling Configuration</i>	362
See Also	363

About Workspace Server Pooling

The primary purpose of pooling is to enhance performance by avoiding the time penalty that is associated with launching all workspace servers on demand. In pooling, a set of workspace servers are made available to process certain types of requests. For example, pooling is used when a relational information map is queried, processed, opened, or used indirectly (through a report).

The initial configuration in a new deployment conforms to the following general recommendations:

- Use server-side pooling. The initial configuration includes a logical pooled workspace server.
- Use client-side pooling only if you have security requirements that are not met by server-side pooling. The initial configuration does not include client-side pooling.

Benefits and Risks of Server-Side Pooling

Server-side pooling offers the following benefits:

- With server side-pooling, the spawner can request user-specific metadata-layer access evaluations, such as whether a user has permission to use a particular server. This is not possible in client-side pooling because use of the pool administrator prevents the spawner from knowing who the requesting user is.
- With server-side pooling, you can track each request to a specific server instance. This level of auditing is not possible in client-side pooling because use of the pool administrator prevents the spawner from knowing who the requesting user is.
- With server-side pooling, testing of an information map in SAS Information Map Studio more closely approximates results in SAS Web Report Studio, because both requests can use the same server.
- With server-side pooling, allocation of server processes is managed across clients, using the spawner's load-balancing capabilities. Client-side pooling uses round-robin assignments on a per-application basis.
- A side effect of pooling is that the launched workspace servers run under a designated service account. This side-effect is beneficial in avoiding credential gaps that can occur when a desktop application requests a workspace server from a middle-tier service. This benefit is not provided by client-side pooling, because client-side pooling is supported only for web applications.

Server-side pooling introduces the following risks:

- Someone might write a rogue application that bypasses metadata layer access controls. This risk originates from the following difference in server control:
 - In client-side pooling, the server is controlled by a designated web application. Only that application can request a pooled workspace server, because only that application can provide the pool administrator's credentials. No other application can use (or exploit) the server.
 - In server-side pooling, there is no pool administrator or alternate form of application-based gatekeeping. Anyone who can use the server legitimately (from a SAS application) can also exploit that server by writing a rogue application that uses the server to fetch SAS tables. The security issue is that retrieval from such an application bypasses metadata-layer controls that apply when the same data is accessed in a legitimate manner. For example, a request in SAS Web Report Studio might be filtered by a metadata-layer permission condition that enables the user to see only certain rows in a data set.

Note: This exposure is also applicable to other SAS processing servers that provide mediated access (standard workspace servers that use SAS token authentication, stored process servers). This exposure occurs in client-side pooling if a user obtains the pool administrator's credentials.

Note: To avoid a similar exposure, nobody is allowed to assign a stored process to a server-side pooled workspace server.

- Someone might exploit the server-side pooled workspace server from within SAS Information Map Studio. In general, only information map creators who have host access to the target data use this application. If someone else has SAS Information Map Studio, that person could bypass metadata-layer controls when querying a relational information map from within that application.

Note: This is never a risk with client-side pooling, because client-side pooling is not available for desktop applications such as SAS Information Map Studio.

The following table summarizes the trade-offs. For completeness, the table includes a column for a standard workspace server that uses SAS token authentication and a column for a standard workspace server that uses either form of host authentication (credential-based or Integrated Windows authentication).

Table A4.1 Summary: Comparison of Workspace Server Configurations

Advantage	Pooled		Not Pooled	
	Server-Side	Client-Side	SAS Token	Host
Performance and Compatibility:				
Improves performance of web applications.	✓	✓		
Compatible with spawner-based load balancing.	✓		✓	✓
Compatible with web authentication.	✓	✓	✓	
Accommodates batch generation of scheduled reports. ¹	✓		✓	
Fully compatible with Integrated Windows authentication. ²	✓		✓	
Security:				
The SAS identity of the requesting user (or group) is used for metadata-layer evaluations.	✓		✓	✓
Server access security is supported.	✓		✓	✓
The server is controlled by designated client applications.		✓		

Advantage	Pooled		Not Pooled	
	Server-Side	Client-Side	SAS Token	Host
The user (or group) host identity is passed to the host layer.				✓
The user (or group) host identity is passed to SQL Server. ³				✓

- 1 Credentials for the workspace server are available to the batch generation process.
- 2 Accommodates requests through SAS Intelligent Query Services for a workspace server (for example, opening an information map after logging on to SAS Enterprise Guide using Integrated Windows authentication).
- 3 If the workspace server's host authentication is by IWA and any additional configuration requirements are met.

Which Requests Are Eligible to Use Pooling?

Requests that are handled by a particular query services software component are eligible to use pooling. That software component is primarily used to query, process, open, or otherwise interact with a relational information map.

Note: In SAS Enterprise Guide and the SAS Add-In for Microsoft Office, not all such requests are eligible. If the libraries that an information map references can be assigned within an existing SAS session, a request to open that information map is not eligible to use pooling. To ensure that such requests are eligible, limit physical (host operating system) access to the directories that are referenced by that information map's libraries. Deny access to users and grant access to the host identity under which a pooled server runs.

Note: Similar requests that do not involve a relational information map are not eligible, because those requests are not handled by the query services component. For example, requests to open a report that directly contains a stored process or open a report that contains OLAP data are not eligible.

Note: A few specialized low-level tasks (such as dynamically building a list of prompt values) are eligible, because the query services component is used for those tasks.

Not all requests that can use pooling actually do so. See the following topic.

Which Eligible Requests Actually Use Pooling?

Use of pooling for eligible requests is constrained as follows:

client-side pooling

can be used for eligible requests in only specially configured web applications. For example, if SAS Web Report Studio's configuration includes a pool administrator, that application uses client-side pooling to process information maps.

server-side pooling

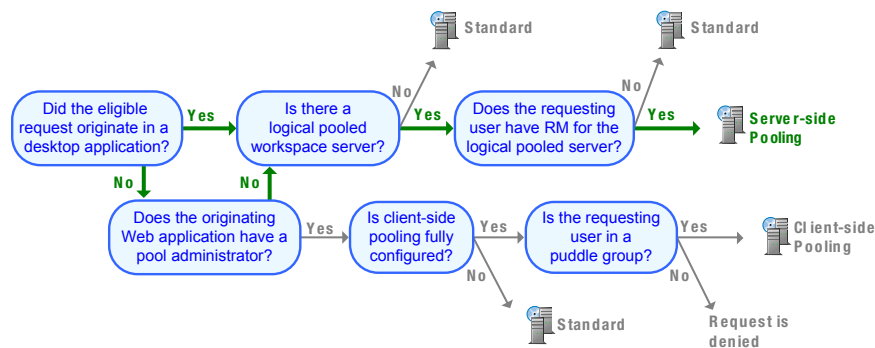
can be used for eligible requests in any application. For example, server-side pooling can be used to process information maps from SAS Web Report Studio, SAS Information Map Studio, SAS Enterprise Guide, and the SAS Add-In for Microsoft Office. However, eligible requests do not use server-side pooling in the following circumstances:

- If a web application has a configured pool administrator, requests from that application do not use server-side pooling.
- If there is no logical pooled workspace server  under a particular application server , requests for resources that are assigned to that application server cannot use server-side pooling.
- If a user does not have the ReadMetadata permission for the logical pooled workspace server , requests made by that user cannot use server-side pooling.

Note: If you chose to limit ReadMetadata permission for the server, be sure to preserve necessary access for service identities.

Note: If the **Use Server Access Security** check box on the logical server's **Options** tab is cleared, users do not need ReadMetadata permission. This check box should always be selected.

The following figure summarizes the decision sequence that determines what type of server processes an eligible request. Most of the determinative factors can be controlled by an administrator.

Figure A4.1 Use of Pooling (Applies to Only Eligible Requests)

In the preceding figure, the bold (green) text and arrows mark the decision path in a new deployment with the default configuration. In such a deployment, there are two logical workspace servers within the general-use application server (for example, **SASapp**):

- a server-side pooled workspace server (the **Logical Pooled Workspace Server**) that is visible to all registered users.
- a standard workspace server (the **Logical Workspace Server**) that is not converted to client-side pooling. There is no configured pool administrator, so client-side pooling is not attempted.

Modifying the Initial Pooling Configuration

If you are concerned about the risk that server-side pooling introduces, and that concern outweighs the advantages of server-side pooling, consider these options:

- For a high security implementation of BI row-level permissions, you must use client-side pooling in a separate deployment of SAS Web Report Studio. You can continue to use server-side pooling in the original deployment. See [SAS Guide to BI Row-Level Permissions](#).
- To prevent only web applications from using server-side pooling, configure those applications to use client-side pooling.

Note: In order to get the security benefit, you must set up client-side pooling correctly. For example, do not use the general SAS Spawned Servers credential (sassrv) as the puddle login.

- To enable information map creators to run queries under their own host identities (instead of under the launch credential of the server-side pooled workspace server), hide that server definition from those users. Someone who

cannot see the logical pooled workspace server uses the standard workspace server instead.

Note: This is appropriate only if you have an information map creator who should not be able to access all data that is available to the pooled workspace server. This does not eliminate the risk of unauthorized use of SAS Information Map Studio by someone who legitimately uses the server-side pooled workspace server from other applications.

- To completely eliminate use of server-side pooling, delete the logical pooled workspace server. For performance reasons, we recommend that you also configure client-side pooling for SAS Web Report Studio.

See Also

- [“Hide Server Definitions” on page 105](#)
- [“Mediated Access” on page 112](#)
- [“Criteria for a Designated Launch Credential” on page 159](#)
- [“Understanding Server Pooling” in *SAS Intelligence Platform: Application Server Administration Guide*](#)

