SAS® Viya® 3.5 Administration: Authentication

Authentication: Overview ................................................................. 2
  Authentication Options ............................................................ 2

Authentication: How To ................................................................. 3
  Authentication Mechanisms ........................................................ 3
  Session Management Using SAS Environment Manager .................. 37
  Additional Authentication Topics .................................................. 39

Authentication: Concepts ............................................................... 45
  Authentication Architecture ......................................................... 45
  Authentication and SAS Viya Services ........................................... 45
  In-bound and Out-bound Authentication ....................................... 46
  Authentication Options ............................................................. 47
  Authentication Mechanisms ......................................................... 52
  Additional Authentication Topics .................................................. 69

Authentication: Guest Access (Linux) .......................................... 72
  About Guest Access ................................................................... 72
  Enable Guest Access ................................................................. 72
  Connect as Guest Users ............................................................. 74
  Generate Custom Links to Reports .............................................. 74
  Disable Guest Access .................................................................. 75

Authentication: OpenID Connect with ISAM Scenario (Linux Full Deployment) .................................................. 77
  Configure OpenID Connect Provider Properties for IBM Security Access Manager ........................................... 77
  Configure OpenID Connect Provider in IBM Security Access Manager ......................................................... 79
  OpenID Connect and IBM Security Access Manager ......................... 80

Authentication: OpenID Connect with Okta Scenario (Linux Full Deployment) .................................................. 82
  Create the Web Application Using the Okta Admin Console .......... 82
  Configure OpenID Connect Provider Properties for Okta ............... 83

Authentication: Passwords ............................................................ 85
  Update Account Passwords on Windows ....................................... 85

Authentication: Reference ............................................................ 86
  CAS Environment Variables for Clients ........................................... 86
Authentication: Overview

Authentication is the process of verifying the identity of a user that is attempting to log on to or access software.

In SAS Viya, authentication options vary, based on which interface and operating system are being used in your environment:

<table>
<thead>
<tr>
<th>Type of Deployment</th>
<th>Operating System</th>
<th>Authentication Mechanism</th>
</tr>
</thead>
<tbody>
<tr>
<td>full deployment</td>
<td>Linux</td>
<td>The pluggable authentication module (PAM) validates the user’s credentials when accessing SAS Studio 5.2 (Basic) and CAS Server Monitor. Batch jobs submit credentials that require validation. Users can be authenticated through SAS Logon Manager, using an LDAP provider, Kerberos, Security Assertion Markup Language (SAML), PAM, or OAuth and OpenID Connect.</td>
</tr>
<tr>
<td>Windows</td>
<td></td>
<td>Host authentication validates the user’s credentials when accessing SAS Studio 5.2 (Basic) and CAS Server Monitor. Batch jobs submit credentials that require validation. Kerberos is the only supported authentication mechanism for SAS Viya visual interfaces and configuration of the middle tier environment.</td>
</tr>
<tr>
<td>programming-only deployment</td>
<td>Linux</td>
<td>The only supported authentication mechanism is PAM.</td>
</tr>
<tr>
<td></td>
<td>Windows</td>
<td>The only supported mechanism is Windows host authentication.</td>
</tr>
</tbody>
</table>
Authentication: How To

Authentication Mechanisms

Overview

Authentication mechanisms integrate SAS into your computing environment. External mechanisms include direct LDAP authentication (which is referred to as LDAP in this documentation), host authentication, Kerberos, Security Assertion Markup Language (SAML), and OAuth 2.0 with OpenID Connect. Pluggable authentication modules (PAM) extend UNIX host authentication.

The following sections are listed alphabetically. Configure the authentication mechanism that is appropriate for your environment. For more information, see “Authentication Mechanisms” on page 52.

Note: On Windows deployments, Windows host authentication validates the user's credentials when accessing SAS Studio 5.2 (Basic) and CAS Server Monitor, and for batch jobs. For SAS Viya visual interfaces and configuration of the middle-tier environment, Kerberos is the only supported authentication mechanism.

Configure Kerberos (Linux Full Deployment)

To configure Kerberos on Linux, you must do the following:

- Perform prerequisites to verify that certain conditions are met.
- Set up SAS Logon Manager, SAS Cloud Analytic Services, SAS Launcher Server, and SAS Object Spawner.
- (Optional) Configure Kerberos constrained delegation.
- Configure your web browser for Kerberos.

Verify Kerberos Prerequisites

Before configuring Kerberos on Linux, make sure that the following items exist:

Note: These prerequisite components are usually configured by the Active Directory administrator.

1. Four service accounts exist in Active Directory.
2. A service principal name (SPN) for each of the service classes listed in Table 2 is mapped to the service accounts from Step 1.
Table 2  Servers for Which Kerberos Can Be Configured on Linux

<table>
<thead>
<tr>
<th>Server</th>
<th>Service Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAS Logon Manager</td>
<td>HTTP</td>
</tr>
<tr>
<td>SAS Cloud Analytic Services</td>
<td>sascas</td>
</tr>
<tr>
<td>SAS Launcher Server</td>
<td>sas-launcher</td>
</tr>
<tr>
<td>SAS Object Spawner</td>
<td>SAS</td>
</tr>
</tbody>
</table>

Verify that a mapping is already configured by running the `setspn -F -Q service_class/fully.qualified.hostname` command for each of the servers listed in Table 2.

You should see output similar to the following:

```
CN=user-logon-name,OU=Service Accounts,OU=Domain Controllers,OU=Servers,DC=EXAMPLE,DC=com
  service_class/fully.qualified.hostname
  service_class/hostname
Existing SPN found!
```

**Note:** The host name specifies the fully qualified domain name of the machine on which the server is running. If using unconstrained delegation, then this service account must be trusted for unconstrained delegation, allowing delegation to all services. If using constrained delegation, then see “Configure Kerberos Constrained Delegation in Active Directory” on page 10.

If an SPN is not found, then contact your information technology support group for assistance with registering the SPN.

Verify that the service is linked to the service account by running the `setspn -L user-logon-name` command.

The value for `user-logon-name` is the same one identified in the common name (CN) from the previous command output, or as the sAMAccountName on the service account in Active Directory.

You should see output similar to the following:

```
Registered ServicePrincipalNames for CN=user-logon-name,OU=Service Accounts,OU=Servers, DC=EXAMPLE,DC=com:
  service_class/fully.qualified.hostname
  service_class/hostname
```

For the sascas service class, a user principal name (UPN) is required.

The matching account is used by the CAS Server to initialize Kerberos credentials for outbound authentication. UPNs are not required for the other service classes, but it is good practice to set them.

Verify that a keytab file has been generated by issuing the `ktutil rkt path-to-keytab-file.keytab list -e` command.
The following is sample output. Your keytab file is different.

**Output 1**  Sample Linux Output

<table>
<thead>
<tr>
<th>slot</th>
<th>KVNO</th>
<th>Principal</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>HTTP/&lt;hostname&gt;@&lt;example&gt;.com</td>
</tr>
</tbody>
</table>

For more information about the *ktutil* command, see the vendor documentation.

5 If the servers are accessed under aliases, an SPN must be added for each possible name used to reach the server. This applies to the HTTP service class, but could also apply to the sascas service class, if it is accessed directly by a client, such as SAS 9.4 or Python.

**Note:** It is possible to use a single service account for all four SPNs. In that case, all SPNs and the UPN for sascas must be assigned to the single service account.

---

**Configure Kerberos for SAS Logon Manager**

1 If you have not already done so, from SAS Environment Manager, add your user ID or an Active Directory group that contains the environment administrators, as a member of the SAS Administrators group. Then, log off from SAS Environment Manager. For more information, see “Add or Remove Custom Group Members” in *SAS Viya Administration: Identity Management*.

**CAUTION**

You must specify your personal user ID. Your user ID must be in your specified LDAP provider. It must match the user ID that you use to log on to your system. Also, your user ID must be added to the SAS Administrators group because once Kerberos is configured, you can no longer sign in as the sasboot user.

2 Make sure that the keytab file is saved to a directory that is accessible to the user account that runs the SAS services.

3 Verify that the service principal name (SPN) is mapped to the user principal name (UPN).

```bash
setspn -F -Q HTTP/hostname.example.com
```

4 Configure the Kerberos authentication properties.

a From SAS Environment Manager, navigate to the SAS Logon Manager configuration definitions. For more information, see “Edit Authentication Configuration Instances” on page 37.

b In the **Definitions** list, select **sas.logon.kerberos**.

c In the top right corner of the window, click **New Configuration**.

d In the New sas.logon.kerberos Configuration dialog box, enter the values for the following fields, based on your environment.
### Table 3  Configuration Fields and Values

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Services</td>
<td>Global or SAS Logon Manager</td>
<td>By default, <strong>Global</strong> is selected and is the required value if you plan to configure Kerberos constrained delegation. For unconstrained delegation, choose <strong>SAS Logon Manager</strong>.</td>
</tr>
<tr>
<td>debug</td>
<td>On</td>
<td>Specifies whether to write debug messages in the log.</td>
</tr>
<tr>
<td>disableDelegationWarning</td>
<td>On</td>
<td>Specifies whether to display a warning message to users when Kerberos credential delegation is not properly configured.</td>
</tr>
<tr>
<td>holdOnToGSSContext:</td>
<td>On</td>
<td>This option is required to enable Kerberos delegation from SAS Logon Manager.</td>
</tr>
<tr>
<td>keyTabLocation</td>
<td>file://path-to-http-keytab-file</td>
<td>Specifies the Uniform Resource Identifier (URI).</td>
</tr>
<tr>
<td>servicePrincipal</td>
<td>principal-name-from-keytab</td>
<td><strong>Note:</strong> If the environment includes multiple realms, this field should include the realm (for example, HTTP/fully.qualified.hostname@REALM). Issue the <code>ktab -l -k FILE:path-to-http-keytab-file.keytab</code> command.</td>
</tr>
<tr>
<td>spn</td>
<td>service-principal-name</td>
<td>Specifies the SPN, if it differs from the principal name in the keytab.</td>
</tr>
<tr>
<td>stripRealmForGss</td>
<td>On</td>
<td>Specifies whether to remove the realm from the UPN.</td>
</tr>
<tr>
<td>impersonate</td>
<td>Off</td>
<td>Specifies whether to impersonate the user credentials using the Microsoft S4USelf extension to Kerberos for outgoing connections. <strong>Note:</strong> If using constrained delegation, set <strong>impersonate</strong> to <strong>On</strong>.</td>
</tr>
</tbody>
</table>

**Note:** Contact your administrator for the keytab location and the host name of the service principal.
5 Add Kerberos to the active profile.
   a In the navigation pane, switch to the All services list and select SAS Logon Manager.
   b In the spring instance, click .
   c In the Edit spring Configuration dialog box, add kerberos to the profiles.active field.
      The following value should be specified for the profiles.active field:
      ldap,postgresql,kerberos
   d Click Save.

6 Restart the SAS Logon Manager service.
   a For Red Hat Enterprise Linux 6.7:
      sudo service sas-viya-saslogon-default restart
   b For Red Hat Enterprise Linux 7.x or later and SUSE Linux:
      sudo systemctl restart sas-viya-saslogon-default

   Note: It might take several minutes to restart SAS Logon Manager.

---

Configure Kerberos for SAS Cloud Analytic Services

1 Create a keytab file for CAS to use.
   The file is used to validate incoming user Kerberos tickets and generate server identity Kerberos tickets for access to Kerberized resources, such as Hadoop. By default, the keytab file should reside in the /etc/sascas.keytab file and be readable only by CAS. If you save the file in a different directory or use a different file name, set the KRB5_KTNAME environment variable (for example, env.KRB5_KTNAME = 'fully-qualified-filename') to the fully qualified file name. For more information, see “CAS Environment Variables” in SAS Viya Administration: SAS Cloud Analytic Services.

2 Verify that the service principal name (SPN) is mapped to the principal name.
   setspn -F -Q sascas/fully.qualified.hostname

3 If you changed the default principal name, set the CAS_SERVER_PRINCIPAL environment variable (for example, env.CAS_SERVER_PRINCIPAL = 'principal-name').
   By default, CAS uses the following Kerberos principal name: sascas/fully-qualified-DNSname. CAS searches for this principal in the keytab file.

4 Add the 'kerb' option to the cas.provlist configuration file option (for example, cas.provlist = 'oauth.ext.kerb').
   For more information about the configuration file option, see “Configuration File Options” in SAS Viya Administration: SAS Cloud Analytic Services.

5 Enable the Kerberos option for authentication to CAS and SAS Compute Server.
   a From SAS Environment Manager, navigate to the Launcher service configuration definitions.
      For more information, see “Edit Authentication Configuration Instances” on page 37.
   b In the Definitions list, select sas.compute.
Configure Kerberos for SAS Launcher Server

1. Create a keytab file for SAS Launcher Server to use.
2. Save the keytab file on the file system of any host where the SAS Launcher Server is running.
   There is no default location where the keytab file should be saved, so it can be placed anywhere on the file system.
3. Make sure that the keytab file is accessible to the “sas” account, the Linux operating system account that runs the process for SAS Launcher Server.
4. Verify that the service principal name (SPN) is mapped to the principal name.
   `setspn -F -Q sas-launcher/fully.qualified.hostname`
5. Complete the following steps as a user with root or sudo privileges:
   a. Source the consul.conf file to add configuration values that use the SAS Security framework certificate truststore.
      `source SAS-Viya-configuration-directory/consul.conf`
   b. Run the `sas-bootstrap-config` script for the SAS Launcher Server keytab.
      ```
      Note: Enter the command on a single line. Multiple lines are used for the command to improve readability.
      
      ```
6. Set the SPN.
   - If you are using an SPN-based keytab file, then no additional steps are required. The default SPN for SAS Launcher Server is `sas-launcher/fully.qualified.hostname`.
   - If you are using a User Principal Name (UPN) based keytab file, the `SAS_SERVICE_PRINCIPAL` environment variable must be set to the user account that matches the UPN, `principal_name@REALM.COM`, where `principal_name` is the user account where the sas-launcher SPN is registered, and REALM.COM is the realm or domain.
     1. Source the consul.conf file to add configuration values that use the SAS Security framework certificate truststore.
        `source SAS-Viya-configuration-directory/consul.conf`
2 Run the `sas-bootstrap-config` script for the SAS Launcher Server for `SAS_SERVICE_PRINCIPAL`.

   **Note:** Enter the command on a single line. Multiple lines are used for the command to improve readability.

   ```bash
   ```

7 Restart SAS Launcher Server.
   - For Red Hat Enterprise Linux 6.7:
     ```bash
     sudo service sas-viya-runlauncher-default restart
     ```
   - For Red Hat Enterprise Linux 7.x or later and SUSE Linux:
     ```bash
     sudo systemctl restart sas-viya-runlauncher-default
     ```

8 Enable the Kerberos option for authentication to SAS Compute Server.
   a From SAS Environment Manager, navigate to the Launcher service configuration definitions. For more information, see "Edit Authentication Configuration Instances" on page 37.
   b In the **Definitions** list, select **sas.compute**.
   c Click 📨.
   d In the **Edit sas.compute Configuration** dialog box, select the **kerberos.enabled** option.
   e Click **Save**.

9 To enable Kerberos constrained delegation, see "Configure Kerberos Constrained Delegation in Active Directory" on page 10.

### Configure Kerberos for SAS Object Spawner

You can enable Kerberos for direct connections from SAS Enterprise Guide 8.2 to SAS Object Spawner on SAS Viya. Complete the following steps:

1 Disable TLS for SAS Object Spawner in a Linux full deployment.
   a Edit the `SAS-Viya-configuration-directory/etc/spawner/default/spawner_usermods.sh` file.
   b Add the following line to the bottom of the file:
      ```bash
      spawner_options="${spawner_options//-ssl*/}"
      ```
   c Restart SAS Object Spawner.
      - For Red Hat Enterprise Linux 6.7:
        ```bash
        sudo service sas-viya-spawner-default restart
        ```
      - For Red Hat Enterprise Linux 7.x or later and SUSE Linux:
        ```bash
        sudo systemctl restart sas-viya-spawner-default
        ```
   d Verify your changes by looking at the output from the following command:
The command should no longer include the following options:

- -sslpvcketloc
- -sslcertloc
- -sslpvkeypassfile
- -sslicalistloc


3. Add the following lines before the existing USERMODS line:

   ```
   CMD_OPTIONS=-sspi
   export KRBS_KTNAME=/opt/sas/sas.keytab
   ```

4. Update the existing USERMODS line to the following:

   ```
   USERMODS="$JREOPTIONS $CMD_OPTIONS"
   ```

5. Restart SAS Object Spawner.

   - For Red Hat Enterprise Linux 6.7:
     ```
     sudo service sas-viya-spawner-default restart
     ```
   - For Red Hat Enterprise Linux 7.x or later and SUSE Linux:
     ```
     sudo systemctl restart sas-viya-spawner-default
     ```

Configure Kerberos Constrained Delegation in Active Directory

**Note:** If you are configuring Kerberos unconstrained delegation, you do not have to complete the following steps.

1. As the Windows domain administrator, navigate to **System and Security ↪ Administrative Tools ↪ Active Directory Users and Computers** to access the properties dialog box for the relevant account and grant the delegation privilege.

2. For SAS Logon Manager, add all Kerberos protected services for the servers listed in Table 4 on page 10:

   **Table 4** Kerberos Protected Servers

<table>
<thead>
<tr>
<th>Server</th>
<th>Service Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAS Cloud Analytic Services</td>
<td>sascas</td>
</tr>
<tr>
<td>SAS Launcher Server</td>
<td>sas-launcher</td>
</tr>
<tr>
<td>SAS Object Spawner</td>
<td>SAS</td>
</tr>
</tbody>
</table>

   a. Select the account to which the HTTP service class is defined.
   
b. Right-click the name and select **Properties**.
c In the Properties dialog box, select the **Delegation** tab.

d On the **Delegation** tab, select the **Trust this user for delegation to specified services only** and the **Use any authentication protocol** check boxes. Then, click **Add**....

e In the Add Services dialog box, click **Users or Computers**....

f In the Select Users or Computers dialog box, complete the following for each of the Kerberos protected service classes (sascas and sas-launcher):

i In the **Enter the object names to select** text box, enter the domain account to which the service class is defined from Table 4 on page 10. Then, click **Check Names**.

ii If the name is found, click **OK**. Otherwise, check the spelling of the name and enter it again.

iii Repeat the previous two steps to select additional SPNs for the service class.

iv When you are done, click **OK**.

g In the Add Services dialog box, click **OK**.

h In the Properties dialog box, click **OK**.

3 For SAS Cloud Analytic Services:

a Select the account to which the service class is defined.

b Right-click the name and click **Properties**.

c In the Properties dialog box, select the **Delegation** tab.

d On the **Delegation** tab, select the **Trust this user for delegation to specified services only** and **Use any authentication protocol** check boxes. Then, click **Add**....

e In the Add Services dialog box, click **Users or Computers**....

f In the Select Users or Computers dialog box, complete the following for the Kerberos protected services that the server accesses:

i In the **Enter the object names to select** text box, enter the account for the Kerberos protected service SAS Cloud Analytic Services accesses, such as Microsoft SQL Server. Then, click **Check Names**.

ii If the name is found, click **OK**. Otherwise, check the spelling of the name and enter it again.

iii Repeat the previous two steps to select additional SPNs for the sascas service.

iv When you are done, click **OK**.

g In the Add Services dialog box, click **OK**.

h In the Properties dialog box, click **OK**.

4 For SAS Launcher Server:

a Select the account to which the service class is defined.

b Right-click the name and click **Properties**.

c In the Properties dialog box, select the **Delegation** tab.

d On the Delegation tab, select the **Trust this user for delegation to specified services only** and **Use any authentication protocol** check boxes. Then, click **Add**....
In the Add Services dialog box, click **Users or Computers**.

In the Select Users or Computers dialog box, complete the following for the Kerberos protected services that the server accesses:

1. In the **Enter the object names to select** text box, enter the account for the Kerberos protected service the SAS Compute Server accesses. This should include sascas, as well as any other services, such as Microsoft SQL Server. Then, click **Check Names**.
2. If the name is found, click **OK**. Otherwise, check the spelling of the name and enter it again.
3. Repeat the previous two steps to select additional SPNs for the sas-launcher service.
4. When you are done, click **OK**.

In the Add Services dialog box, click **OK**.

In the Properties dialog box, click **OK**.

---

**Note:** No additional configuration of SAS Object Spawner is required to support constrained delegation.

---

**Note:** It is not necessary to restart the SAS services. However, Windows client users need to log off and log back on to their desktop. Or, from the command prompt, they can use the `klist purge` command to clear their Kerberos credentials cache. Windows regenerates the Kerberos credentials as necessary.

---

**Validate Kerberos Configuration**

All users are authenticated using OAuth 2.0 and OpenID Connect. Complete the following steps to verify that Kerberos is configured correctly:

1. Check the CAS log to see how the non-delegated user authenticated to CAS by running the following command:
   ```
   cat /var/log/sas/viya/cas/default/* | grep non_delegated_user | grep authenticated | tail -1
   ```
   
   Look for output similar to the following:
   
   ```
   2018-06-12T11:03:35,376 INFO  [00002846] <non_delegated_user> local MAIN NoUser [tkidentgss.c:741] - User <non_delegated_user>@<domain_name> successfully authenticated using the OAuth authentication provider.
   ```

   On Linux systems, delegation occurs only for users who are in the CASHostAccountRequired custom group. Users with delegated Kerberos credentials are also authenticated with the Kerberos authentication provider to delegate their identity to CAS. To validate Kerberos for the delegated user, complete the following steps:

   1. Check the CAS log to see how the delegated user authenticated to CAS.

   On Linux, run the following command:
   ```
   cat /var/log/sas/viya/cas/default/* | grep delegated_user | grep kerberos | tail -1
   ```
   
   Look for output similar to the following:
Configure Microsoft Internet Explorer, Microsoft Edge, and Google Chrome to Use Kerberos

Configure Security Settings
1. In the Windows Control Panel, open Internet Options.
2. In the Internet Properties dialog box, select the **Security** tab.
3. Select **Local intranet**, and then click **Sites**.
4. In the Local intranet dialog box, configure the intranet domain settings.
   a. Verify that the check boxes for the following items are selected:
      - Include all local (Intranet) sites not listed in other zones
      - Include all sites that bypass the proxy server
   b. Click **Advanced** and add your domain name to the **Websites** list to ensure that Internet Explorer recognizes any site with your domain name as the intranet.
   c. Click **Close**, and then click **OK**.
5. Configure intranet authentication.
   a. In the **Security level for this zone** area, click **Custom level**.
   b. In the Security Settings - Local Intranet Zone dialog box, scroll to the **User Authentication** section, select **Automatic Logon only in Intranet Zone**, and click **OK**.

Configure Connection Settings
If your site uses a proxy server, follow these steps:
1. In the Internet Properties dialog box, select the **Connections** tab.
2. Click **LAN settings**.
3. In the Local Area Network (LAN) Settings dialog box, verify that the proxy server address and port number are correct.
4. Click **Advanced**.
5. In the Proxy Settings dialog box, verify that the correct domain names are entered in the **Exceptions** field. Then, click **OK**.
6. Click **OK**.

Configure Integrated Windows Authentication
1. In the Internet Properties dialog box, select the **Advanced** tab.
2. Scroll to the **Security** section, and verify that **Enable Integrated Windows Authentication** is selected.
3. Click **OK** and restart your computer to activate the changes.
Configure User Delegation for Microsoft Internet Explorer

Complete the following steps after configuring Integrated Windows Authentication:

1. In the Windows Control Panel, open Internet Options.
2. In the Internet Properties dialog box, select the Security tab.
3. Select Trusted Sites, and then click Sites.
4. In the Trusted sites dialog box, enter the middle-tier host name in the Add this website to the zone field and click Add.
5. Click Close, and then click OK.

Note: For Internet Explorer to pass a forwardable ticket to the SAS Viya machine, the service account in Active Directory holding the SPNs must be trusted for delegation.

Configure User Delegation for Microsoft Edge

Complete the following steps after configuring Integrated Windows Authentication:

1. Open the Windows registry editor.
2. Add the following REG_SZ keys:
   \HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Edge\AuthServerAllowlist
   Specifies which servers to enable for integrated authentication. Set the value to the SAS Web Server host name: hostname.example.com.

   \HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Edge\AuthNegotiateDelegateAllowlist
   Specifies which servers Microsoft Edge can delegate to. Set the value to the SAS Web Server host name: hostname.example.com.

Configure User Delegation for Google Chrome

By default, Chrome disables the delegation of Kerberos credentials. The Windows registry must be updated. Microsoft recommends performing a system backup before editing the registry. Complete the following steps to enable Kerberos delegation after configuring Integrated Windows Authentication:

1. Open the Windows registry editor.
2. Add the following REG_SZ keys:
   \HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Google\Chrome\AuthServerWhitelist
   Specifies which servers should be whitelisted for integrated authentication. Set the value to the SAS Web Server host name: hostname.example.com.

   \HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Google\Chrome\AuthNegotiateDelegateWhitelist
   Specifies which servers Chrome can delegate to. Set the value to the SAS Web Server host name: hostname.example.com.

Note: You might also need to add Google and Chrome under Policies.

Configure Mozilla Firefox to Use Kerberos

Configure Kerberos

1. From a browser window, navigate to about:config.
To configure Kerberos on Windows, you must do the following:

- Perform prerequisites to verify that certain conditions are met.
- Set up SAS Logon Manager and SAS Object Spawner.
- (Optional) Configure Kerberos constrained delegation.
- Configure your web browser for Kerberos.

Verify Kerberos Prerequisites

Before configuring Kerberos, make sure that the following items exist:

- Three service accounts exist in Active Directory.
- The cas account requires the following properties:
  - Membership in the local Administrators groups on the machine where the CAS Server is installed.
  - Has the following privileges:
    - Log on as a service
    - Replace a Process Level Token
  - The recommended account name is cas. However, the name must be unique for the equivalent user on the domain. The maximum length of the name is 20 characters.
  - This account requires a password. If the password expires, the CAS service no longer starts.
- A service principal name (SPN) for each of the service classes listed in Table 5 is mapped to the service accounts from Step 2.
Table 5  Servers for Which Kerberos Can Be Configured on Windows

<table>
<thead>
<tr>
<th>Server</th>
<th>Service Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAS Logon Manager</td>
<td>HTTP</td>
</tr>
<tr>
<td>SAS Cloud Analytic Services</td>
<td>sascas</td>
</tr>
<tr>
<td>SAS Object Spawner</td>
<td>SAS</td>
</tr>
</tbody>
</table>

Verify that a mapping is already configured by running the `setspn -F -Q service_class/fully.qualified.hostname` command for each of the servers listed in Table 5.

You should see output similar to the following:

```
CN=user-logon-name,OU=Service Accounts,OU=Domain Controllers,OU=Services,DC=EXAMPLE,DC=com
  service_class/fully.qualified.hostname
  service_class/hostname
```

Existing SPN found!

Note: The host name specifies the fully qualified domain name of the machine on which the server is running.

Note: For CAS, the SPN must be registered on the service account that is running the server. If using unconstrained delegation, then this service account must be trusted for unconstrained delegation, allowing delegation to all services. If using constrained delegation, then see “Configure Kerberos Constrained Delegation in Active Directory” on page 20.

If an SPN is not found, then contact your information technology support group for assistance with registering the SPN.

Verify that the service is linked to the service account by running the `setspn -L user-logon-name` command.

The value for `user-logon-name` is the same one identified in the common name (CN) from the previous command output, or as the sAMAccountName on the service account in Active Directory.

You should see output similar to the following:

```
Registered ServicePrincipalNames for CN=user-logon-name,OU=Service Accounts,OU=Servers,DC=EXAMPLE,DC=com:
  service_class/fully.qualified.hostname
  service_class/hostname
```

For the sascas service class, a user principal name (UPN) is required.

The matching account is used by the CAS Server to initialize Kerberos credentials for outbound authentication. UPNs are not required for the other service classes, but it is good practice to set them.
Verify that a keytab file has been generated by issuing the `ktab.exe -l -k FILE:path-to-keytab-file.keytab` command.

The following is sample output. Your keytab file is different.

**Output 2  Sample Windows Output**

```
Keytab name: <filename>.keytab
KVNO       Principal
-         -
1          HTTP/<hostname>@<example>.com
```

For more information about the `ktab` command, see the vendor documentation.

If the servers are accessed under aliases, an SPN must be added for each possible name used to reach the server. This applies to the HTTP service class, but could also apply to the sascas service class, if it is accessed directly by a client, such as SAS 9.4 or Python.

Verify that the computer machine object must be trusted for delegation. SAS Launcher Server runs under the local system account on the machine it is deployed on and registers its own SPN. Therefore, a sas-launcher service class is not required.

- For **unconstrained** delegation, the server on which SAS Launcher Server is running must be marked as trusted in Active Directory as trusted for delegation to any service.
- For **constrained** delegation, the server on which SAS Launcher Server is running must be marked in Active Directory as trusted for delegation to specified services.

**Note:** If the computer is not marked as trusted, it cannot use the user's Kerberos ticket to access remote file systems, nor can it launch CAS sessions under user identity.

**Note:** It is possible to use a single service account for all three SPNs. In that case, all SPNs and the UPN for sascas must be assigned to the single service account.

### Configure Kerberos for SAS Logon Manager

1. If you have not already done so, from SAS Environment Manager, add your user ID or an Active Directory group that contains the environment administrators, as a member of the SAS Administrators group. Then, log off from SAS Environment Manager. For more information, see “Add or Remove Custom Group Members” in *SAS Viya Administration: Identity Management*.

**CAUTION**

You must specify your personal user ID. Your user ID must be in your specified LDAP provider. It must match the user ID that you use to log on to your system. Also, your user ID must be added to the SAS Administrators group because once Kerberos is configured, you can no longer sign in as the sasboot user.

2. Make sure that the keytab file is saved to a directory that is accessible to the user account that runs the SAS services.

3. Verify that the service principal name (SPN) is mapped to the user principal name (UPN).
   
   ```
   setspn -F -Q HTTP/hostname.example.com
   ```

4. Configure the Kerberos authentication properties.
a. From SAS Environment Manager, navigate to the SAS Logon Manager configuration definitions. For more information, see “Edit Authentication Configuration Instances” on page 37.

b. In the Definitions list, select sas.logon.kerberos.

c. In the top right corner of the window, click New Configuration.

d. In the New sas.logon.kerberos Configuration dialog box, enter the values for the following fields, based on your environment.

<p>| Table 6  Configuration Fields and Values |
|---------|----------------------------------------|
| Field   | Value                                  | Description                                                                 |
| Services| Global or SAS Logon Manager            | By default, Global is selected and is the required value if you plan to configure Kerberos constrained delegation. For unconstrained delegation, choose SAS Logon Manager. |
| debug   | On                                     | Specifies whether to write debug messages in the log.                        |
| disableDelegationWarning | On                                       | Specifies whether to display a warning message to users when Kerberos credential delegation is not properly configured. |
| holdOnToGSSContext: | On                                       | This option is required to enable Kerberos delegation from SAS Logon Manager. |
| keyTabLocation | file:///path-to-http-keytab-file         | Specifies the Uniform Resource Identifier (URI).                              |
| servicePrincipal | principal-name-from-keytab             | On Windows, issue the ktab.exe -1 -k FILE:&lt;userSuppliedValue&gt;path-to-http-keytab-file&lt;/userSuppliedValue&gt;.keytab command from the directory where Java is installed on your machine. |
| spn     | service-principal-name                  | Specifies the SPN, if it differs from the principal name in the keytab.       |
| stripRealmForGss     | On                                       | Specifies whether to remove the realm from the UPN.                           |</p>
<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>impersonate</td>
<td>Off</td>
<td>Specifies whether to impersonate the user credentials using the Microsoft S4USelf extension to Kerberos for outgoing connections. Note: If using constrained delegation, set impersonate to On.</td>
</tr>
</tbody>
</table>

Note: Contact your administrator for the keytab location and the host name of the service principal.

e  Click Save.

5 Add Kerberos to the active profile.
   a In the navigation pane, switch to the All services list and select SAS Logon Manager.
   b In the spring instance, click .
   c In the Edit spring Configuration dialog box, add kerberos to the profiles.active field.
      The following value should be specified for the profiles.active field:
      ldap,postgresql,kerberos
d  Click Save.

6 Restart the SAS Logon Manager service. In Windows Services Manager, right-click SAS Logon Manager service and select Restart.

   Note: It might take several minutes to restart SAS Logon Manager.

Note: Once Kerberos is enabled on Windows, a browser running on the same machine where the services are deployed cannot connect to SAS Viya visual interfaces.

Configure Kerberos for SAS Object Spawner

You can enable Kerberos for direct connections from SAS Enterprise Guide 8.2 to SAS Object Spawner on SAS Viya. Complete the following steps:

1 Edit the SAS-Viya-configuration-directory/etc/spawner/default/spawner_usermods.sh file.

2 Add the following lines before the existing USERMODS line:
   
   CMD_OPTIONS=-sspi
   export KRB5_KTNAME=/opt/sas/sas.keytab

3 Update the existing USERMODS line to the following:
   
   USERMODS="$JREOPTIONS $CMD_OPTIONS"
4 Restart SAS Object Spawner. In Windows Services Manager, click SAS Object Spawner service and select Restart.

Configure Kerberos Constrained Delegation for SAS Launcher Server

Note: If you are configuring Kerberos unconstrained delegation, you do not have to complete the following steps.

1 Set the SAS_CONSTRAINED_DELEG_ENABLED environment variable.
   a As the local administrator, in the Windows Control Panel, open System.
   b Click the Advanced system settings link.
   c In the System Properties window, click Environment Variables.
   d In the Environment Variables dialog box, in the System variables section, click New.
   e In the New System Variable dialog box, in the Variable name: field, enter SAS_CONSTRAINED_DELEG_ENABLED. In the Variable value: field, enter 1. Then, click OK.
      Note: The variable value can be set to any value.
   f In the Environment Variables dialog box, click OK.
   g In the System Properties dialog box, click OK.

2 In Windows Services Manager, restart SAS Launcher Server and SAS Cloud Analytic Services.
   a Right-click the SAS Runlauncher Service and select Restart.
   b Right-click the SAS Cloud Analytic Services and select Restart.

3 To enable Kerberos constrained delegation, see “Configure Kerberos Constrained Delegation in Active Directory” on page 20.

Configure Kerberos Constrained Delegation in Active Directory

Note: If you are configuring Kerberos unconstrained delegation, you do not have to complete the following steps.

1 As the Windows domain administrator, navigate to System and Security ➔ Administrative Tools ➔ Active Directory Users and Computers to access the properties dialog box for the relevant account and grant the delegation privilege.
For SAS Logon Manager, add all Kerberos protected services for the servers listed in Table 7 on page 21:

Table 7  Kerberos Protected Servers

<table>
<thead>
<tr>
<th>Server</th>
<th>Service Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAS Cloud Analytic Services</td>
<td>sascas</td>
</tr>
<tr>
<td>SAS Launcher Server</td>
<td>sas-launcher</td>
</tr>
<tr>
<td>SAS Object Spawner</td>
<td>SAS</td>
</tr>
</tbody>
</table>

a. Select the account to which the HTTP service class is defined.
b. Right-click the name and select Properties.
c. In the Properties dialog box, select the Delegation tab.
d. On the Delegation tab, select the Trust this computer for delegation to specified services only and the Use any authentication protocol check boxes. Then, click Add....
e. In the Add Services dialog box, click Users or Computers....
f. In the Select Users or Computers dialog box, complete the following for each of the Kerberos protected service classes (sascas and sas-launcher):
   i. In the Enter the object names to select text box, enter the domain account to which the service class is defined from Table 7 on page 21. Then, click Check Names.
   ii. If the name is found, click OK. Otherwise, check the spelling of the name and enter it again.
   iii. Repeat the previous two steps to select additional SPNs for the service class.
   iv. When you are done, click OK.
g. In the Add Services dialog box, click OK.
h. In the Properties dialog box, click OK.

3 For SAS Cloud Analytic Services:
   a. Select the account to which the service class is defined.
   b. Right-click the name and click Properties.
   c. In the Properties dialog box, select the Delegation tab.
   d. On the Delegation tab, select the Trust this user for delegation to specified services only and Use any authentication protocol check boxes. Then, click Add....
   e. In the Add Services dialog box, click Users or Computers....
   f. In the Select Users or Computers dialog box, complete the following for the Kerberos protected services that the server accesses:
      i. In the Enter the object names to select text box, enter the account for the Kerberos protected service SAS Cloud Analytic Services accesses, such as Microsoft SQL Server. Then, click Check Names.
ii If the name is found, click OK. Otherwise, check the spelling of the name and enter it again.

iii Repeat the previous two steps to select additional SPNs for the sascas service.

iv When you are done, click OK.

g In the Add Services dialog box, click OK.

h In the Properties dialog box, click OK.

4 For SAS Launcher Server:

a Select the account to which the service class is defined.

b Right-click the name and select Properties.

c In the Properties dialog box, select the Delegation tab.

d On the Delegation tab, select the Trust this computer for delegation to specified services only and Use any authentication protocol check boxes. Then, click Add....

e In the Add Services dialog box, click Users or Computers....

f In the Select Users or Computers dialog box, complete the following for the Kerberos protected services that the server accesses:

i In the Enter the object names to select text box, enter the account for the Kerberos protected service the SAS Compute Server accesses. This should include sascas, as well as any other services, such as Microsoft SQL Server. Then, click Check Names.

ii If the name is found, click OK. Otherwise, check the spelling of the name and enter it again.

iii Repeat the previous two steps to select additional SPNs for the sas-launcher service.

iv When you are done, click OK.

g In the Add Services dialog box, click OK.

h In the Properties dialog box, click OK.

5 For SAS Object Spawner:

a As the local administrator, in the Windows Control Panel, open System.

b Click the Advanced system settings link.

c In the System Properties window, click Environment Variables.

d In the Environment Variables dialog box, in the System variables section, click New.

e In the New System Variable dialog box, in the Variable name: field, enter SAS_CONSTRAINED_DELEG_ENABLED. In the Variable value: field, enter 1. Then, click OK.

Note: The variable value can be set to any value.

f In the Environment Variables dialog box, click OK.

g In the System Properties dialog box, click OK.

h In Windows Services Manager, restart SAS Object Spawner and SAS Cloud Analytic Services.

   i Right-click the SAS Object Spawner service and select Restart.
Right-click the SAS Cloud Analytic Services and select Restart.

Note: If a SAS Viya application is going to access a universal naming convention (UNC) path, then the computer object on which the SAS service is running is trusted for delegation to the desired common internet file system (CIFS) service class. This is due to Windows using server message block (SMB) to access UNC paths. SMB runs under the local system computer account.

Validate Kerberos Configuration

All users are authenticated using OAuth 2.0 and OpenID Connect. Complete the following steps to verify that Kerberos is configured correctly:

1. Check the CAS log to see how the non-delegated user authenticated to CAS. Navigate to the C:\ProgramData\SAS\Viya\var\log\cas\default directory and view the contents of the cas_date_hostname file.

2. Look for output similar to the following:

   2018-06-12T11:03:35,376 INFO [00002846] <non_delegated_user> local MAIN NoUser [tkidentgss.c:741] - User <non_delegated_user>@<domain_name> successfully authenticated using the OAuth authentication provider.

On Windows systems, users are automatically delegated. Users with delegated Kerberos credentials are also authenticated with the Kerberos authentication provider to delegate their identity to CAS. To validate Kerberos for the delegated user, complete the following steps:

1. Check the CAS log to see how the delegated user authenticated to CAS.

   On Windows, navigate to the C:\ProgramData\SAS\Viya\var\log\cas\default directory and view the contents of the cas_date_hostname file.

2. Look for output similar to the following:


Configure Microsoft Internet Explorer, Microsoft Edge, and Google Chrome to Use Kerberos

Configure Security Settings

1. In the Windows Control Panel, open Internet Options.

2. In the Internet Properties dialog box, select the Security tab.

3. Select Local intranet, and then click Sites.

4. In the Local intranet dialog box, configure the intranet domain settings.
   a. Verify that the check boxes for the following items are selected:
      - Include all local (Intranet) sites not listed in other zones
      - Include all sites that bypass the proxy server
b Click Advanced and add your domain name to the Websites list to ensure that Internet Explorer recognizes any site with your domain name as the intranet.

c Click Close, and then click OK.

5 Configure intranet authentication.
   a In the Security level for this zone area, click Custom level.
   b In the Security Settings - Local Intranet Zone dialog box, scroll to the User Authentication section, select Automatic Logon only in Intranet Zone, and click OK.

Configure Connection Settings
If your site uses a proxy server, follow these steps:
1 In the Internet Properties dialog box, select the Connections tab.
2 Click LAN settings.
3 In the Local Area Network (LAN) Settings dialog box, verify that the proxy server address and port number are correct.
4 Click Advanced.
5 In the Proxy Settings dialog box, verify that the correct domain names are entered in the Exceptions field. Then, click OK.
6 Click OK.

Configure Integrated Windows Authentication
1 In the Internet Properties dialog box, select the Advanced tab.
2 Scroll to the Security section, and verify that Enable Integrated Windows Authentication is selected.
3 Click OK and restart your computer to activate the changes.

Configure User Delegation for Microsoft Internet Explorer
Complete the following steps after configuring Integrated Windows Authentication:
1 In the Windows Control Panel, open Internet Options.
2 In the Internet Properties dialog box, select the Security tab.
3 Select Trusted Sites, and then click Sites.
4 In the Trusted sites dialog box, enter the middle-tier host name in the Add this website to the zone field and click Add.
5 Click Close, and then click OK.

Note: For Internet Explorer to pass a forwardable ticket to the SAS Viya machine, the service account in Active Directory holding the SPNs must be trusted for delegation.

Configure User Delegation for Microsoft Edge
Complete the following steps after configuring Integrated Windows Authentication:
1 Open the Windows registry editor.
Add the following REG_SZ keys:

\HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Edge\AuthServerAllowlist
  Specifies which servers to enable for integrated authentication. Set the value to the SAS Web Server host name: hostname.example.com.

\HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Edge\AuthNegotiateDelegateAllowlist
  Specifies which servers Microsoft Edge can delegate to. Set the value to the SAS Web Server host name: hostname.example.com.

Configure User Delegation for Google Chrome

By default, Chrome disables the delegation of Kerberos credentials. The Windows registry must be updated. Microsoft recommends performing a system backup before editing the registry. Complete the following steps to enable Kerberos delegation after configuring Integrated Windows Authentication:

1. Open the Windows registry editor.
2. Add the following REG_SZ keys:

\HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Google\Chrome\AuthServerWhitelist
  Specifies which servers should be whitelisted for integrated authentication. Set the value to the SAS Web Server host name: hostname.example.com.

\HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Google\Chrome\AuthNegotiateDelegateWhitelist
  Specifies which servers Chrome can delegate to. Set the value to the SAS Web Server host name: hostname.example.com.

Note: You might also need to add Google and Chrome under Policies.

Configure Mozilla Firefox to Use Kerberos

Configure Kerberos

1. From a browser window, navigate to about:config.
2. Click I accept the risk! to accept the security warning.
3. In the Search field, enter network.negotiate.
4. Double-click the network.negotiate-auth.trusted-uris Preference Name, enter http://hostname.example.com, in the Enter string value field, and then click OK.

Note: The values in the Enter string value field are comma-separated.

Configure User Delegation

1. From a browser window, navigate to about:config.
2. Click I accept the risk! to accept the security warning.
3. In the Search field, enter network.negotiate.
4. Double-click the network.negotiate-auth.delegation-uris Preference Name, enter http://hostname.example.com in the Enter string value field, and then click OK.
Configure OAuth and OIDC (Linux Full Deployment)

Overview

The following sections provide details for configuring OpenID Connect (OIDC) in a single tenant environment. For information about configuring OAuth and OIDC in a multi-tenant environment, see “About Authentication in Multi-tenancy” in SAS Viya Administration: Multi-tenancy.

Configuration of OIDC typically follows the following pattern:

- Configure the OIDC identity provider (IdP).
- Create a configuration for the sas.logon.oauth.providers.external_oauth definition in SAS Environment Manager with information about the IdP. For more information, see “Configure OIDC” on page 26.

Instructions are provided for configuring OIDC with the following IdPs:

- ISAM Scenario on page 77
- Okta Scenario on page 82

Configure OIDC

To configure the OIDC, complete the following:

1. From SAS Environment Manager, navigate to the SAS Logon Manager configuration definitions. For more information, see “Edit Authentication Configuration Instances” on page 37.

2. In the Definitions list, select sas.logon.oauth.providers.external_oauth.

3. In the top right corner of the window, click New Configuration.

4. In the New sas.logon.oauth.providers.external_oauth Configuration dialog box, enter values for the required fields, based on your environment. The following table provides guidance about the information needed for the listed fields:

   Table 8 OAuth Configuration Fields, Values, and Descriptions

<table>
<thead>
<tr>
<th>Configuration Field</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>addShadowUserOnLogin</td>
<td>On</td>
<td>Specifies that a local shadow user should be added once authentication is successful.</td>
</tr>
<tr>
<td>Note:</td>
<td>This option should always be on.</td>
<td></td>
</tr>
<tr>
<td>attributeMapping.user_name</td>
<td>By default, the value is user_name.</td>
<td>Specifies the attribute from the provider, which contains the user name. The value specified is used by the scopes option.</td>
</tr>
<tr>
<td>authUrl</td>
<td>No default value</td>
<td>Specifies the URL to the authorization endpoint of the third-party.</td>
</tr>
<tr>
<td>Configuration Field</td>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>emailDomain</td>
<td>domain_name1, domain_name2, domain_name3</td>
<td>Specifies a comma-separated list of email domains of users that can sign on with this provider. It is used with identity provider (IdP) discovery and is optional.</td>
</tr>
<tr>
<td>issuer</td>
<td><a href="https://hostname/auth/realms/realm_name">https://hostname/auth/realms/realm_name</a></td>
<td>Specifies the principal that issued the token, specified as a case-sensitive string or URI. This value must match the issue claim in the token.</td>
</tr>
<tr>
<td>linkText</td>
<td>The default value is “Use your corporate credentials”.</td>
<td>Specifies the text that should be displayed on the sign-in page.</td>
</tr>
<tr>
<td>relyingPartyId</td>
<td>account-name-OAuth</td>
<td>Specifies the client ID that is registered with the provider.</td>
</tr>
<tr>
<td>relyingPartySecret</td>
<td>No default value</td>
<td>Specifies the secret that is registered with the provider for the client ID.</td>
</tr>
<tr>
<td>scopes</td>
<td>The list should contain openid.</td>
<td>This option depends on what is defined for the attributeMapping.userName. The scope tells the provider what fields to get back from the provider. Depending on the provider, they might need to include a scope to get back the user name field.</td>
</tr>
<tr>
<td>showLinkText</td>
<td>On</td>
<td>Specifies that the link text should be shown on the sign-in page.</td>
</tr>
<tr>
<td>tokenUrl</td>
<td>No default value</td>
<td>The URL to obtain the token key. endpoint. Specify either this value or the tokenKey, but not both.</td>
</tr>
<tr>
<td>type</td>
<td>By default, the value is oidc1.0.</td>
<td>Specifies the protocol type. Note: SAS Viya requires an id_token in the authorization response from the provider. However, some providers return an id_token when the scope in the authorization request is openid and response_type=token. For those providers, use type oauth2.0.</td>
</tr>
<tr>
<td>tokenKey</td>
<td>No default value</td>
<td>Specifies the HMAC key or RSA public key that is used to sign tokens.</td>
</tr>
</tbody>
</table>
## Configure Identity Provider Discovery for OIDC

You can complete the following steps to enable identity provider discovery:

1. From SAS Environment Manager, navigate to the SAS Logon Manager configuration definitions. For more information, see “Edit Authentication Configuration Instances” on page 37.

2. In the Definitions list, select sas.logon.zone.

3. In the top right corner of the window, click New Configuration.

4. In the New sas.logon.zone Configuration dialog box, enable the idpDiscovery.enabled option.

   Note: Specify the emailDomain option that is defined in the sas.logon.oauth.providers.external_oauth definition.

5. Click Save.

## Configure PAM (Linux)

Default PAM configuration files are installed for both the CAS server and SAS Studio.

1. As a user with root authority, edit the $Viya-configuration-directory/etc/pam.d/service file. For the CAS server, service is cas. For SAS Studio, service is sasauth.

   The following information is displayed for the CAS server:

   ```bash
   $ vi /etc/pam.d/cas
   #PAM-1.0
   auth include password-auth
   ```
account include password-auth
password include password-auth
session include password-auth

The following information is displayed for SAS Studio:

$ vi /etc/pam.d/sasauth
#%PAM-1.0
auth include password-auth
account include password-auth

2 Make any modifications to the file that are necessary for your environment.

3 Save the file and exit.

Configure SAML (Linux Full Deployment)

Overview

Before configuring the Security Assertion Markup Language (SAML) in a single tenant environment, you must generate an RSA private key in PKCS#1 format and a certificate. You can generate this yourself or use an existing one (for example, the private key and certificate used by the httpd server). For more information, see “Generate a JWT Signing Key” in Encryption in SAS Viya: Data in Motion.

Configuration for SAML typically follows this pattern:

1 “Configure SAS Viya as a SAML Service Provider” on page 29
2 “Configure the SAML Identity Provider – Relying Party Configuration” on page 31
3 “Configure SAS Viya with Information about the SAML Identity Provider” on page 31

For information about configuring SAML in a multi-tenant environment, see “About Authentication in Multi-tenancy” in SAS Viya Administration: Multi-tenancy.

Note: By default, SAS Viya allows only same-origin requests. Authentication requests from the SAML identity provider might be seen as cross-origin. Therefore, the origin of the SAML provider might need to be added. For details, see “Configure Cross-Origin Resource Sharing” on page 39.

Configure SAS Viya as a SAML Service Provider

1 From SAS Environment Manager, navigate to the SAS Logon Manager configuration definitions. For more information, see “Edit Authentication Configuration Instances” on page 37.

2 In the Definitions list, select sas.logon.saml.

Note: If you change any of the sas.logon.saml properties, the new metadata must be provided to the Relying Party in the federated service. If it is not, the SAML connections might fail.

3 In the top right corner of the window, click New Configuration.
4 In the New sas.logon.saml Configuration dialog box, enter values for the required fields, based on your environment. The following table provides guidance on what information needs to be provided for the listed fields:

Table 9  SAML Configuration Fields and Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>entityBaseURL</td>
<td>The external URL for the SAS Logon web application in SAS Viya (for example, <a href="https://hostname.example.com/SASLogon">https://hostname.example.com/SASLogon</a>).</td>
</tr>
<tr>
<td>entityID</td>
<td>The unique ID that represents the service provider that is included in protocol messages between relying parties. Change from the default value that is pre-populated.</td>
</tr>
<tr>
<td>serviceProviderCertificate</td>
<td>Paste a copy of the PEM-encoded (base64) certificate, which is used by the service provider.</td>
</tr>
<tr>
<td>serviceProviderKey</td>
<td>Paste a copy of the PEM-encoded (base64) key, which is used by the service provider.</td>
</tr>
<tr>
<td>serviceProviderKeyPassword</td>
<td>Provide the password for the service provider, or leave blank if there is no password.</td>
</tr>
<tr>
<td>setProxyParams</td>
<td>IMPORTANT This field should not be modified. The value should remain false.</td>
</tr>
<tr>
<td>signMetaData</td>
<td>Specifies whether the local service provider should sign the metadata.</td>
</tr>
<tr>
<td>signRequest</td>
<td>Specifies whether the local service provider should sign the SAML requests.</td>
</tr>
<tr>
<td>wantAssertionSigned</td>
<td>Specifies whether the assertions should be signed.</td>
</tr>
<tr>
<td>signatureAlgorithm</td>
<td>Specifies the algorithm for SAML signatures. Acceptable values are SHA1, SHA256, and SHA512. The default value is SHA256.</td>
</tr>
<tr>
<td>socket.connectionManagerTimeout</td>
<td>Specifies the amount of time (in milliseconds) before the connection pooling times out for HTTP requests for SAML metadata. The default value is 10000.</td>
</tr>
<tr>
<td>socket.soTimeout</td>
<td>Specifies the amount of time (in milliseconds) before the read times out for HTTP requests for SAML metadata. The default value is 10000.</td>
</tr>
<tr>
<td>maxAuthenticationAge</td>
<td>Specifies the maximum time (in seconds) between users initial authentication with the identity provider (IdP) and processing of an authentication statement. The default value is 864000.</td>
</tr>
</tbody>
</table>

5 Click **Save**.

6 Restart the SAS Logon Manager Service.
For Red Hat Enterprise Linux 6.7:

```bash
sudo service sas-viya-saslogon-default restart
```

For Red Hat Enterprise Linux 7.x or later and SUSE Linux:

```bash
sudo systemctl restart sas-viya-saslogon-default
```

On Windows, in Windows Services Manager, right-click the **SAS Logon Manager Service** and select **Restart**.

Note: It might take several minutes to restart SAS Logon Manager.

---

**Configure the SAML Identity Provider – Relying Party Configuration**

You can either configure the relying party trust or supply the required information to your information technology support group, in order for them to add the relying party trust. Here is an overview of the steps to perform, if you configure the relying party trust. The steps might vary, depending on which tool you use for configuration.

1. If the identity provider requires it, configure Transport Layer Security (TLS), if it has not already been configured. For more information, see "Update Apache HTTP Server TLS Certificates and Cryptography" in *Encryption in SAS Viya: Data in Motion*.

2. Download the application metadata.xml file, which contains information about the service provider, or provide the https://hostname/SASLogon/saml/metadata link to your information technology support group.

3. Request that your information technology support group configure a relying party in the identity provider.

---

**Configure SAS Viya with Information about the SAML Identity Provider**

1. Complete the following steps in SAS Environment Manager:

   a. In the **Definitions** list, select `sas.logon.saml.providers.external_saml`.

   b. In the top right corner of the window, click **New Configuration**.

   c. In the New `sas.logon.saml.providers.external_saml` Configuration dialog box, enter values for the required fields, based on your environment. The following table provides guidance on what information needs to be provided for the listed fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>addShadowUserOnLogin</td>
<td>Add a local shadow user upon successful authentication. If set to false, users must already exist in the database to log on.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>assertionConsumerIndex</td>
<td>The index of the assertion consumer service to use from identity provider metadata. The value must be a positive integer. The default value is 0.</td>
</tr>
<tr>
<td>idpMetadata</td>
<td>The metadata XML content. This can be useful if manual changes need to be made to the identity provider metadata.</td>
</tr>
<tr>
<td>linkText</td>
<td>The hyperlink to display on the sign-in page.</td>
</tr>
<tr>
<td>metadataTrustCheck</td>
<td>Specify whether to trust the identity provider certificate.</td>
</tr>
<tr>
<td>nameID</td>
<td>The field is populated with a default value. Verify with your information technology support group that the value is correct.</td>
</tr>
<tr>
<td>showSamlLoginLink</td>
<td>Determines whether a link should be displayed on the sign-in page for this identity provider.</td>
</tr>
<tr>
<td>skipSslValidation</td>
<td>Specifies whether to skip the TLS validation of the certificate.</td>
</tr>
<tr>
<td>emailDomain</td>
<td>Specifies a comma-separated list of email domains for users that can sign on with the SAML provider. It is used with IdP discovery and is optional.</td>
</tr>
<tr>
<td>authnContext</td>
<td>The comma-separated list of authentication contexts that are included in SAML requests to the IdP.</td>
</tr>
</tbody>
</table>

32

Click **Save**.

2 Edit the `SAS-Viya-configuration-directory/etc/sysconfig/sas-javaesntl/sas-java-services` file, and uncomment the highlighted line in the following block:

```bash
if [ -f $truststore ]; then
    export java_global_option_truststore="-Djavax.net.ssl.trustStore=$truststore"
    export java_global_option_truststore_password="-Djavax.net.ssl.trustStorePassword=changeit"
fi
```

3 Restart the SAS Logon Manager Service.

- For Red Hat Enterprise Linux 6.7:
  ```bash
sudo service sas-viya-saslogon-default restart
  ```
- For Red Hat Enterprise Linux 7.x or later and SUSE Linux:
  ```bash
  sudo systemctl restart sas-viya-saslogon-default
  ```

On Windows, in Windows Services Manager, right-click the **SAS Logon Manager Service** and select **Restart**.

**Note:** It might take several minutes to restart SAS Logon Manager.
Configure Identity Provider Discovery for SAML

You can complete the following steps to enable identity provider discovery:

1. From SAS Environment Manager, navigate to the SAS Logon Manager configuration definitions. For more information, see “Edit Authentication Configuration Instances” on page 37.

2. In the Definitions list, select sas.logon.zone.

3. In the top right corner of the window, click New Configuration.

4. In the New sas.logon.zone Configuration dialog box, enable the idpDiscovery.enabled option.

   **Note:** Specify the emailDomain option that is defined in the sas.logon.oauth.providers.external_oauth definition.

5. Click Save.

Configure the SameSite Attribute

1. From SAS Environment Manager, navigate to the configuration definitions. For more information, see “Edit Authentication Configuration Instances” on page 37.

2. In the Definitions list, select sas.commons.web.security.cookies.

3. In the top right corner of the window, click New Configuration.

4. In the New sas.commons.web.security.cookies Configuration dialog box:
   a. Specify Global in the Services field.
   b. Specify a value for the SameSite property. Valid values are listed in Table 11:

   **Table 11** Possible SameSite Property Values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unset</td>
<td>The same-site cookie attribute is not set. This is the default value.</td>
</tr>
<tr>
<td>Strict</td>
<td>The browser does not send the cookie in any cross-site requests.</td>
</tr>
<tr>
<td>Lax</td>
<td>The browser sends the cookie in same-site requests and cross-site top-level GET requests.</td>
</tr>
<tr>
<td>None</td>
<td>The same-site cookie attribute is set and the cookie is sent in cross-site requests.</td>
</tr>
</tbody>
</table>

   **Note:** In order for SAS Visual Analytics SDK to work properly, the SameSite property must be set to None.

   c. Click Save.

5. Restart the SAS Logon Manager Service.
   - For Red Hat Enterprise Linux 6.7:
sudo service sas-viya-saslogon-default restart

- For Red Hat Enterprise Linux 7.x or later and SUSE Linux:
  
  sudo systemctl restart sas-viya-saslogon-default

On Windows, in Windows Services Manager, right-click the SAS Logon Manager Service and select Restart.

Note: It might take several minutes to restart SAS Logon Manager.

---

### Configure Single Sign-On with Automatic Redirect

The following section provides instructions on how to configure single sign-on (SSO) so that it automatically redirects to an already configured third-party Security Assertion Markup Language (SAML) or OpenID Connect (OIDC) provider. SSO with either SAML or OIDC enables end users to access the SAS Viya web applications without having to interact with SAS Logon Manager.

Note: This configuration does not impact the command line interfaces. It also does not affect SAS Studio 5.2 (Basic).

1. Add an entry in the emailDomain option for the SAML or OIDC provider. For SAML this is under the `sas.logon.saml.providers.external_saml` definition. For OIDC, this is under the `sas.logon.oauth.providers.external_oauth` definition.

   Note: The string entered in the emailDomain option does not have to be the actual email domain for the end users. You can specify any string (for example `matchme.com`) since we configure the email domain in the Apache HTTP Server. Therefore, this configuration works well when the end users have many different email domains and when the SAS administrator does not know the email domains of the end-users.

2. Configure the login_hint option.

   a. Add `login_hint` to the authorize request.

      This requires a change to the Apache HTTP Server. Add a rewrite rule to the configuration to set the `login_hint` option on the requests to /SASLogon/oauth/authorize. In most cases, the SAS Viya environment is accessed over HTTPS. This means the changes that need to be made should be made in the SSL configuration file. For Red Hat Enterprise Linux this would be the `/etc/httpd/conf.d/ssl.conf` file. The new content is added to the end of the file before the closing `</VirtualHost>` tag.

      Note: For deployments that are using HTTP, this should be placed in a new .conf file. The Apache HTTP Server processes the .conf files in alphabetical order and this one needs to occur before the proxy configuration in proxy.conf, so it should be named accordingly (for example, login_hint.conf). Putting the redirect rule in both places is fine too. The Apache HTTP Server must be restarted after making any changes to the configuration.

   b. Enable `login_hint` for all authorize requests made to SAS Logon Manager, using one of the following:
To add `login_hint` to all authorize requests, use the following code. The SAS Logon Manager sign-in page is not displayed for any users.

```bash
# SSO for SAS Viya set login_hint option
RewriteEngine On
RewriteCond "%{QUERY_STRING}" !login_hint
RewriteRule "SASLogon/oauth/authorize" "SASLogon/oauth/authorize?login_hint=email_domain" [QSA, PT]
```

**Note:** Enter `/SASLogon/login` in the web browser to access the sign-in page directly and to sign in using LDAP credentials or the sasboot account.

To enable the SAS Logon Manager sign-in page when accessed from a specific IP Address, use the following example code:

```bash
# SSO for SAS Viya set login_hint option except for specific IP
RewriteEngine On
RewriteCond "%{QUERY_STRING}" !login_hint
RewriteCond expr "! -R 'ip_address'"
RewriteRule "SASLogon/oauth/authorize" "SASLogon/oauth/authorize?login_hint=email_domain" [QSA,PT]
```

Replace `ip_address` with the value for your environment.

To enable the SAS Logon Manager sign-in page to be available to more than a single IP address, change the RewriteCond statement to use CIDR notation:

```bash
# SSO for SAS Viya set login_hint option except for specific IP Range
RewriteEngine On
RewriteCond "%{QUERY_STRING}" !login_hint
RewriteCond expr "! -R 'ip_address/24'"
RewriteRule "SASLogon/oauth/authorize" "SASLogon/oauth/authorize?login_hint=email_domain" [QSA,PT]
```

Replace `ip_address` with the CIDR value for your environment.

To add conditions to the RewriteRule, you can change the behavior based on the browser rather than the IP address. To exclude a web browser, use the following:

```bash
# SSO for SAS Viya set login_hint option except for browser_type
RewriteEngine On
RewriteCond "%{QUERY_STRING}" !login_hint
RewriteCond "%{HTTP_USER_AGENT}" !.*browser_type.*
RewriteRule "SASLogon/oauth/authorize" "SASLogon/oauth/authorize?login_hint=email_domain" [QSA,PT]
```

To exclude Google Chrome, specify `Edg.` in place of `browser_type`. To exclude Microsoft Edge, specify `Chrome`.

---

**Configure Authentication Options with SAS 9.4**

**Configure the SAS 9.4 Deployment**

1. Log on to SAS Management Console and navigate to Plug-ins ➔ Application Management ➔ Configuration Manager.
2. Right-click **SAS Application Infrastructure** and select **Properties**.
3 Click **Advanced**, and then set the following property value:

ServiceUrl.Allowed

Specifies the address to where tickets should be sent on SAS Viya. The format of the address should be similar to the following: http://hostname/SASLogon/**.

**Note:** For SAS deployments prior to SAS 9.4M3, the ServiceUrl.Allowed property is not required.

4 Click **OK**.

5 Restart all instances of SASServer1 to pick-up the new property.

**Configure the SAS Viya Deployment**

1 From SAS Environment Manager, navigate to the SAS Logon Manager configuration definitions. For more information, see “Edit Authentication Configuration Instances” on page 37.

2 In the **Definitions** list, select **sas.logon.sas9**.

3 In the top right corner of the window, click **New Configuration**.

4 In the New sas.logon.sas9 Configuration dialog box, enter values for the required fields, based on your environment. The following table provides guidance on what information needs to be provided for the listed fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>autoLink</td>
<td>Specifies whether to automatically open the link to SAS 9.4 when the sign-in page is displayed.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> If the autoLink property is enabled, then the SAS Logon Manager in SAS Viya sign-in page is not displayed. End users are automatically redirected to SAS Logon Manager in SAS 9.4 to authenticate. End users cannot use the LDAP provider.</td>
</tr>
<tr>
<td>enabled</td>
<td>Specifies whether to enable users to sign in using SAS 9.4 credentials.</td>
</tr>
<tr>
<td>linkText</td>
<td>Specifies the hyperlink to display on the sign-in page.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> By default, the end user is presented with a link at the bottom of the standard SAS Logon Manager in SAS Viya sign-in page. The text of the link is controlled by the linkText property. This default behavior means that end users can choose to either use SAS 9.4 to authenticate or use the LDAP provider.</td>
</tr>
<tr>
<td>showLinkText</td>
<td>Specifies whether to display the link text on the sign-in page.</td>
</tr>
<tr>
<td>single.signOn.enabled</td>
<td>Specifies whether to redirect to SAS 9.4 for single sign-on.</td>
</tr>
</tbody>
</table>
### Session Management Using SAS Environment Manager

#### Overview

The following sections provide information about customizing SAS Logon Manager and the user’s session experience.

#### Edit Authentication Configuration Instances

1. Log on to SAS Environment Manager, using your user ID or the ID of a user who is a member of the SAS Administrators group.

2. In the applications menu (☰), select **Administration ⇄ Manage Environment**.

3. In the navigation bar, click ⚙.

4. Select **Definitions** from the drop-down box.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>single.signOut.enabled</td>
<td>Specifies whether the local sign-out should also sign the user out of SAS 9.4.</td>
</tr>
<tr>
<td>viyaLogonUrl</td>
<td>Specifies the URL of the SAS Logon Manager in SAS Viya (for example, https://SASViya_hostname/SASLogon).</td>
</tr>
</tbody>
</table>
See Also

- *SAS Viya Administration: Configuration Properties*

### Customize Sign-in, Sign-out, and Session Time-out Content

You can configure customized content that is displayed when users of SAS web applications sign in, sign out, or the session reaches the time-out interval. To enable the display of customize content, follow these steps:

1. In the *Definitions* list, select *sas.logon.custom*.
2. In the top right corner of the window, click 
3. In the New *sas.logon.custom* Configuration dialog box, specify the URI that contains the custom content that you want to display. Here are the available fields:
   - **login**
   - **logout**
   - **timedout**
   For a description of the properties, see “*sas.logon.custom*” in *SAS Viya Administration: Configuration Properties*.
4. Click *Save*.

### Customize Concurrent Sign-in Sessions

1. In the *Definitions* list, select *sas.logon.sessions*.
2. In the top right corner of the window, click 
3. In the New *sas.logon.sessions* Configuration dialog box, you can set the following properties:
   - **maxConcurrentSessions**
   - **rejectNewSessionsIfMaxExceeded**
   Set this property to limit users to a certain number of concurrent sessions.
   When sessions are limited, the default behavior is to cause an existing session to expire and grant a new session to the user attempting to authenticate. To override this behavior and prevent a new session from being granted, set this property to *true*.
4. Click *Save*.

### Configure the HTTP Session Time-out Interval

1. In the *Definitions* list, select *server*.
2. In the top right corner of the window, click 

---

**Note:**
- For detailed descriptions of the properties, refer to the *SAS Viya Administration: Configuration Properties* documentation.
- The steps provided are for guidance and may vary based on the specific version of SAS Viya you are using.
- Ensure that you have the necessary permissions to configure these settings in your environment.
In the New server Configuration dialog box, complete the following:

a. Select **SAS Logon Manager** from the **Services** drop-down list.

b. Click +.

c. In the **Name** field, enter **session.timeout**.

d. In the **Value** field, enter the amount of time a session has to be idle before it times out, in seconds.

e. Click **Save**.

4. Click **Save**.

5. Restart all services to reflect the new time-out interval. For more information, see “Start and Stop All Servers and Services” in *SAS Viya Administration: General Servers and Services*.

### Disable Sign-ins

As a SAS administrator, you can disable sign-ins through operating system firewall rules or using LDAP. This disables new sessions, ends current sessions, and prevents others from using the deployment. For more information, see the appropriate documentation for your operating system.

### Additional Authentication Topics

#### Configure Cross-Origin Resource Sharing

By default, SAS Viya allows only same-origin requests. If cross-origin requests are needed, complete the following steps:

1. In SAS Environment Manager, edit the CORS configuration instance. For details, see “Edit Authentication Configuration Instances” on page 37.

2. Select **sas.commons.web.security.cors**.

3. In the top right corner of the window, click **New Configuration**.

4. In the New sas.commons.web.security.cors Configuration dialog box, specify values that correspond to your environment. For a description of each field, see “sas.commons.web.security.cors” in *SAS Viya Administration: Configuration Properties*.

   **Note:** The specified value for the **allowedOrigins** field must be a comma-delimited list of URIs or an asterisk (‘*’) to accept all origins. Partial wildcards are not supported (for example, `https://*.example.com`).

5. Click **Save**.

6. Restart the SAS Logon Manager Service.

   - For Red Hat Enterprise Linux 6.7:
sudo service sas-viya-saslogon-default restart

- For Red Hat Enterprise Linux 7.x or later and SUSE Linux:
  sudo systemctl restart sas-viya-saslogon-default

On Windows, in Windows Services Manager, right-click the **SAS Logon Manager Service** and select Restart.

**Note:** It might take several minutes to restart SAS Logon Manager.

---

## Register a New Client ID

You can register a client manually or using the command-line interface (CLI). You need to register a client only once.

### Register a Client Manually

1. Set an environment variable, containing the content of the Consul token. Here is an example:

   ```
   export CONSUL_TOKEN=token_from_file
   ```

   The Consul token is located in the following directory:

   - **On Linux:** `SAS-Viya-configuration-directory/etc/SASSecurityCertificateFramework/tokens/consul/default/client.token`
   - **On Windows:** `SAS-Viya-configuration-directory\etc\SASSecurityCertificateFramework\tokens\consul\default\client.token`

2. Obtain a token to register the new client ID and secret.

   ```
   curl -k -X POST "https://localhost/SASLogon/oauth/clients/consul?callback=false&serviceId=client-id" 
   -H "X-Consul-Token: $CONSUL_TOKEN"
   ```

   **Note:** The initial line of the curl command must be entered on one line. It is shown on more than one line for display purposes only.

3. Set an environment variable, containing the content of the Bearer token.

   ```
   export BEARER_TOKEN=bearer_access_token
   ```

4. Use the token to register the new client ID and secret.

   The following curl command registers a client to obtain tokens using a password grant. For a listing of the other grant parameters, see **Table 13 on page 41**.

   ```
   curl -k -X POST "http://localhost/SASLogon/oauth/clients" \
   -H "Content-Type: application/json" \
   -H "Authorization: Bearer token-from-previous-step" \
   -d '{
     "client_id": "client-id",
   }'
   ```

---
Table 13 Password Grant Parameters and Descriptions

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Constraints</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>client-id</td>
<td>String</td>
<td>Required</td>
<td>Specifies a unique client identifier.</td>
</tr>
<tr>
<td>client-secret</td>
<td>String</td>
<td>Required if the client uses the authorization_code or client_credentials grant types</td>
<td>Specifies a secret string used for authenticating as this client.</td>
</tr>
<tr>
<td>scope</td>
<td>Array</td>
<td>Optional (defaults to &quot;uaa.none&quot;)</td>
<td>Specifies a list of scopes and groups that you might request when obtaining a token. We recommend that you always use &quot;*&quot;, because authorization rules may exist to specifically DENY access to certain groups. Therefore, you want the tokens to have a complete picture of all the group memberships the user has.</td>
</tr>
<tr>
<td>resource_ids</td>
<td>Array</td>
<td>Optional</td>
<td>Specifies resources the client is allowed to access.</td>
</tr>
<tr>
<td>authorities</td>
<td>Array</td>
<td>Optional (defaults to &quot;uaa.none&quot;)</td>
<td>Specifies the scopes that tokens get using the client_credentials grant_type.</td>
</tr>
<tr>
<td>authorized_grant_types</td>
<td>Array</td>
<td>Required</td>
<td>Specifies the list of grant types that can be used to obtain a token with this client. The types include authorization_code, password, implicit, and client_credentials.</td>
</tr>
<tr>
<td>redirect_uri</td>
<td>Array</td>
<td>Optional</td>
<td>Specifies the allowed URI pattern for redirect during authorization. Wildcard patterns can be specified using the Ant-style pattern.</td>
</tr>
<tr>
<td>autoapprove</td>
<td>[Boolean, Array]</td>
<td>Optional</td>
<td>Specifies the scopes that do not require user approval. Boolean values true and false apply to all scopes, otherwise a list can be provided.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Type</td>
<td>Constraints</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------</td>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>access_token_validity</td>
<td>Number</td>
<td>Optional</td>
<td>Specifies the time, in seconds, to access token expiration after it is issued.</td>
</tr>
<tr>
<td>refresh_token_validity</td>
<td>Number</td>
<td>Optional</td>
<td>Specifies the time, in seconds, to refresh token expiration after it is issued.</td>
</tr>
<tr>
<td>allowedproviders</td>
<td>Array</td>
<td>Optional</td>
<td>Specifies a list of origin keys (alias) for identity providers to which the client is limited. Null implies any identity provider is allowed.</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>Optional</td>
<td>Specifies a human readable name for the client.</td>
</tr>
<tr>
<td>token_salt</td>
<td>String</td>
<td>String</td>
<td>Specifies a random string used to generate the client’s revocation key. Change this value to revoke all active tokens for the client.</td>
</tr>
</tbody>
</table>

Register a Client Using the CLI

1. Set an environment variable, containing the content of the Consul token. Here is an example:

   ```bash
   export CONSUL_TOKEN=token_from_file
   ``

   The Consul token is located in the following directory:
   
   - On Windows: `SAS-Viya-configuration-directory\etc\SASSecurityCertificateFramework\tokens\consul\default\client.token`

2. Register the client, by running the OAuth command-line interface (CLI).

   The following CLI registers a client to obtain tokens using a password grant. Run the command without the `--grant-password` to display all of the grant parameters. For a listing of the other parameters, see Table 13 on page 41.

   ```bash
   
   Note: You should change `localhost` to specify the host name for the endpoint in your environment, and `secret` should be a unique secret value for registering the client.

Obtain an Access Token

SAS Viya APIs have a security layer that requires an access token. The token authenticates the user and also contains the group memberships of the user, which is used to make authorization decisions and determine if the user can access the endpoint or resource.

You can obtain an access token using an authorization code, a password, or client credentials.

Obtain an Access Token Using an Authorization Code

During this method, the end user is not required to give their password credentials to the client application. Instead, the user goes to SAS Logon Manager to get the client application an authorization code, which it can use to obtain an access token.

Note: A prerequisite is that the client ID is registered with the authorization_code, grant_type, and redirect_uri urn:ietf:wg:oauth:2.0:oob.

1  The end user approves access to get the authentication code.
   a  Enter the following URL in your web browser and substitute the host name and client ID from your environment:

   https://localhost/SASLogon/oauth/authorize?client_id=client-id&response_type=code

   The SAS sign-in page is displayed.

   b  Log on with your SAS user credentials.

   c  On the Authorize Access window, select the openid option, and any other required groups. Then, click Authorize Access.

   d  On the Authorization Code window, copy the authorization code.

2  The client application gets an access token using the authorization code.
   a  Run the following command:

   curl -k https://localhost/SASLogon/oauth/token \
   -H "Accept: application/json" \
   -H "Content-Type: application/x-www-form-urlencoded" \
   -d "grant_type=authorization_code&code=auth-code" \
   -u "client-id:secret"

   Note: The access token is created once and needs to be refreshed when the token expires.

   b  Create environment variables for the tokens, since they might be long:

   export ACCESS_TOKEN="access_token"
   export REFRESH_TOKEN="refresh_token"

3  Use the access token to call SAS Viya APIs. Once you have the access token, you can call any API, as long as your access token is valid and the user has access to the endpoint.

   curl -k https://localhost/folders/folders?filter=isNull(parent) \
   -H "Authorization: Bearer $ACCESS_TOKEN"
4 Use the refresh token to get a new access token. A prerequisite is that the client ID is registered with the `refresh_token` grant_type.

Here is an example:

```bash
curl -k https://localhost/SASLogon/oauth/token
-H "Accept: application/json" \
-H "Content-Type: application/x-www-form-urlencoded"
-u "myclientid:myclientsecret"
-d "grant_type=refresh_token&refresh_token=$REFRESH_TOKEN"
```

The access token is new, and the refresh token remains static. Use the new token for future REST calls. Make sure to replace the ACCESS_TOKEN variable with the new token. Also, the access token has a default life of ten hours before it expires. Most applications deal with expiring and refreshing tokens programmatically. If you want to change the default expiry of an access token in SAS, see “sas.logon.jwt” in SAS Viya Administration: Configuration Properties.

**Obtain an Access Token Using Password Credentials**

During this method, the end user gives their password credentials to the client application.

*Note: A prerequisite is that the client ID is registered with the `password` grant_type.*

1 Get an access token using password credentials. Here is an example:

```bash
curl https://localhost/SASLogon/oauth/token \
-H "Content-Type: application/x-www-form-urlencoded" \
-d "grant_type=password&username=username&password=password" \
-u "client-id:secret"
```

2 Copy the value of the `access_token` attribute from the JavaScript Object Notation (JSON) in the response and save it.

**Obtain an Access Token Using Client Credentials**

Clients may obtain an access token using the client's credentials alone. This type of access token is not associated to any end user and is scoped according to the authorities registered to the client. This topic is covered in the next section. An end user is not involved in this flow, so there is no resource owner except the client itself.

*Note: A prerequisite is that the client ID is registered with the `client_credentials`, `grant_type`, and any authorities that are required.*

1 The client requests an access token using the client credentials.

```bash
curl -k https://localhost/SASLogon/oauth/token \
-H "Content-Type: application/x-www-form-urlencoded" \
-d "grant_type=client_credentials" \
-u "client-id:secret"
```

2 SAS Logon Manager authenticates the client and issues an access token.
Create an Authinfo File

The authinfo file supplies a user name and password that is sent to CAS for authentication. For information about how to create an authinfo file, see Create an Authinfo File.

Authentication: Concepts

Authentication Architecture

In a full deployment, authentication services are provided by SAS Logon Manager. SAS Logon Manager is based on the Cloud Foundry User Account and Authentication (UAA) server. The security architecture is built around Open Authorization (OAuth) and OpenID Connect. By default, authentication is performed via a Lightweight Directory Access Protocol (LDAP) provider. Authentication support is also available for Kerberos, OAuth 2.0 with OpenID Connect, and Security Assertion Markup Language (SAML).

Note: For Windows deployments, Kerberos is the only supported authentication mechanism for SAS Viya visual interfaces and configuration of the middle tier environment.

In a programming-only and full deployment, host authentication is supported on both Linux and Windows systems. On Linux systems, you can configure the host to use only pluggable authentication modules (PAM).

See Also

"About Authentication in Multi-tenancy" in SAS Viya Administration: Multi-tenancy

Authentication and SAS Viya Services

The following table lists the key services that are used in authentication in SAS Viya:
### In-bound and Out-bound Authentication

#### In-bound Authentication

In-bound authentication is the authentication of the end user to the environment. In-bound authentication provides an internal OAuth token and group membership information in the OAuth token. If Kerberos authentication is used, a delegated Kerberos credential is also stored.

The client browser must be configured to delegate credentials. Otherwise, an error message is displayed. To prevent the message, set the `disableDelegationWarning` option. For more information, see the steps to configure the Kerberos authentication properties on page 5.

#### Out-bound Authentication

Out-bound authentication is the authentication of the SAS process to a downstream process. Out-bound authentication occurs after the end user is initially authenticated to SAS Logon Manager.
bound authentication occurs to SAS Cloud Analytic Services, SAS Compute Server (through SAS Launcher Service), and then onto external resources, such as Secured Hadoop environments.

Authentication Options

Authentication for Visual Interfaces

With visual interfaces, users are authenticated through SAS Logon Manager. SAS Logon Manager is a web application that handles all authentication requests for SAS web applications and is accessed via the Apache HTTP Server.

The following figure shows how a user is authenticated on Linux to SAS Logon Manager and the supported authentication mechanisms.
The following protocols are available for you to configure for authentication:

- The first option is a Lightweight Directory Access Protocol (LDAP) provider. This is the default configuration. In this configuration, SAS Logon Manager displays a sign-in page and submits the entered credentials to LDAP. The identity service verifies users in LDAP. For more information, see “LDAP Authentication (Full Deployment)” on page 52.

- The second option is Kerberos. In this configuration, SAS Logon Manager uses SPNEGO to authenticate users against the Kerberos Key Distribution Center (KDC). The identity service verifies users in LDAP. For more information, see “Kerberos Authentication (Full Deployment)” on page 52.

Note: SAS Cloud Analytic Services sessions run as the end user only when using Kerberos delegation. On Linux systems, the user must be a member of the CASHostAccountRequired custom group. On Windows systems, users are automatically delegated.
The third option is OAuth 2.0 and OpenID Connect. In this configuration, SAS Logon Manager uses OAuth 2.0 and OpenID Connect to authenticate users. The identity service verifies users in LDAP. For more information, see “OAuth and OpenID Connect Authentication (Linux Full Deployment)” on page 62.

The fourth option is Security Assertion Markup Language (SAML). In this configuration, SAS Logon Manager uses a SAML provider to authenticate users. The identity service verifies users in LDAP. For more information, see “SAML Authentication (Linux Full Deployment)” on page 63.

The fifth option is SAS 9.4. In this configuration, SAS Logon Manager supports single sign-on and single sign-off with SAS 9.4. The identity service verifies users in LDAP. For more information, see “SAS 9.4 Authentication” on page 66.

The sixth option is pluggable authentication module (PAM) to support multi-factor authentication. In this configuration, SAS Logon Manager uses the operating system PAM stack. The identity service verifies users in LDAP. For more information, see “PAM Authentication (Linux)” on page 67.

With all six options, the connection to SAS Cloud Analytic Services (CAS) environment is performed using internal OAuth tokens that are generated by SAS Logon Manager. In most cases, the session that is started by the CAS controller runs on the operating system as the same user who launched the CAS operating system service. This defaults to the cas account.

### Authentication for Programming Interfaces

#### Overview of Programming Interfaces

The following figure shows how a user is authenticated on Linux while using programming interfaces.
In a deployment with programming interfaces, the user’s credentials are entered into SAS Studio via the Apache HTTP Server. Then SAS Object Spawner uses pluggable authentication module (PAM) configuration files on the host to validate the user ID and password. The user ID and password can be a local account on the host or, depending on the PAM configuration, an account in the LDAP provider. Once the user is authenticated, SAS Workspace Server is started. The PAM configuration file for SAS Studio is sasauth and includes the password module.

SAS Workspace Server connects to the CAS environment using the user ID and password that were used to start SAS Workspace Server. However, if the AUTHINFO= option is specified, it is used to find credentials to connect to CAS. For more information about the AUTHINFO= option, see AUTHINFO= SAS system option or AUTHINFO= CAS statement option.

The CAS controller uses its own PAM configuration to validate the user’s credentials and launch the session process as the user. The PAM configuration file for CAS is cas and includes the password module.
The CAS controller uses the user ID and password to obtain an internal OAuth token from SAS Logon Manager. This requires the user ID and password to be valid in the LDAP provider that is configured for SAS Logon Manager. Otherwise, CAS cannot obtain an OAuth token, and the session fails. Therefore, PAM for SAS Studio (sasauth), PAM for CAS (cas), and SAS Logon Manager should all use the same or equivalent LDAP providers. These three components should be sending the user ID and password that was entered into SAS Studio to the same provider. Otherwise, errors might be generated when trying to connect.

Programming Interfaces with Symmetric Multiprocessing CAS Server

In a symmetric multi-processing (SMP) environment, a CAS server consists of a controller and runs on a single machine. The following details the authentication process:

1. The end user connects to the SAS Studio 5.2 (Basic) application and enters their user name and password in the sign-in page. SAS Studio is proxied by the Apache HTTP Server.
2. SAS Studio 5.2 (Basic) passes the user name and password to SAS Object Spawner to start the SAS Workspace Server for the end user.
3. SAS Object Spawner uses the PAM configuration that is defined in /etc/pam.d/sasauth to validate the user name and password and launches SAS Workspace Server as the end user.
4. The end user enters code to start their CAS session. SAS Workspace Server passes the user name and password to the CAS controller.
5. The CAS controller connects to SAS Logon Manager to obtain an OAuth token presenting the end user’s user name and password.
6. SAS Logon Manager validates the user name and password against the defined LDAP Provider. SAS Logon also connects to the Identities service to obtain group information to include in the OAuth token.
7. The Identities service connects to the LDAP provider with a simple BIND operation. It uses stored credentials for a service account and regularly connects to refresh the cache of users and groups, which is stored in SAS Infrastructure Data Server.
8. SAS Logon Manager sends the OAuth token back to the CAS controller.
9. The CAS controller uses the PAM configuration in /etc/pam.d/cas to validate the user name and password and launches the CAS controller as the end user.

See Also

- “Single-machine CAS Server” in SAS Viya Administration: SAS Cloud Analytic Services
- “Multiple CAS Servers” in SAS Viya Administration: SAS Cloud Analytic Services

Programming Interfaces with Massively Parallel Processing CAS Servers

In a massively parallel processing (MPP) environment, a distributed CAS server consists of one controller, one or more workers, and one backup controller (optional). Each component runs on a separate machine. The authentication process for MPP SAS Cloud Analytic Services is essentially the same as for SMP CAS, with the following key differences:
Initial communication between the CAS controller and CAS workers is via Secure Socket Shell (SSH).

On-going communication does not use SSH.

A worker process is launched on each CAS worker as the end user.
- The CAS controller authenticates the end user with PAM.
- The CAS controller generates an internal identity token after authenticating the end user.
- The internal identity token is used to launch the CAS worker processes.
- PAM is not used on the CAS worker nodes.

See Also

“Multiple CAS Servers” in SAS Viya Administration: SAS Cloud Analytic Services

Authentication Mechanisms

LDAP Authentication (Full Deployment)

Overview of LDAP

In SAS Viya, LDAP is used for identifying and authenticating users. Third-party LDAP server implementations are supported, including Microsoft Active Directory and OpenLDAP.

How It Works in SAS Viya

LDAP is the default authentication mechanism. The Identities service always makes a direct connection to LDAP to obtain user and group information. By default, SAS Logon Manager authenticates users using a direct connection to the configured LDAP provider. To ensure that network connections are secure, the connection between the browser and the Apache HTTP Server can be secured with HTTPS. In addition, the connection between SAS Logon Manager and the LDAP provider can be secured with LDAPS.

For information about configuring LDAP, see Configure the Connection to Your Identity Provider.

Kerberos Authentication (Full Deployment)

Overview of Kerberos

Kerberos is a network authentication protocol that is used to verify user or host identity. The Kerberos protocol uses strong cryptography so that a client can prove its identity to a service (and vice versa) across an insecure network connection. During Kerberos authentication, a user’s credentials (user ID and password) are not sent over the network. Instead, both the client and the service use the credentials that were supplied as a key in an encryption algorithm to encrypt the message that is sent between the client and the service. If the client sends an encrypted message, and the service uses
the same key to decrypt the message, it is proven that the credential is known without having to transmit the credentials.

In SAS Viya, the visual interfaces are SAS Environment Manager and CAS Server Monitor. SAS Environment Manager can be enabled to support Kerberos authentication. Conversely, CAS Server Monitor does not support Kerberos authentication.

Key Terms

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client</td>
<td>An application that is attempting to connect to and access a resource, on behalf of a user. Resources include reports that are viewed, services that are accessed, and databases that are queried. In SAS Viya, the client is the web browser.</td>
</tr>
<tr>
<td>Service</td>
<td>A service, or server, that hosts a resource the user wants to connect to. The service must be able to validate the service tickets presented by the client.</td>
</tr>
<tr>
<td>Key Distribution Center</td>
<td>A trusted third party within Kerberos that verifies the authenticity of the client and service. Both the client and service must trust the KDC. In addition, end users and services must register with the KDC.</td>
</tr>
<tr>
<td>Service Principal Name</td>
<td>A unique name that is used to identify a web service that is running on a server. Before a service principal name (SPN) can be used, it must be registered. Every web service that uses Kerberos authentication needs to have an SPN set for it so that clients can identify the server on the network. An SPN usually matches the pattern of HTTP/hostname.example.com.</td>
</tr>
<tr>
<td>Keytab File</td>
<td>A file containing pairs of Kerberos principals and encrypted keys. The keys are associated with a password for the principal. The principals are SPNs. Keys can use different encryption algorithms. For a single principal, you might have several entries that correspond to each encryption type.</td>
</tr>
<tr>
<td>Ticket-granting ticket</td>
<td>An encrypted identification file that is valid for a limited amount of time. After a user is authenticated, this file is granted to a user for data traffic protection by the KDC. The TGT file contains the session key, its expiration date, and the user's IP address.</td>
</tr>
</tbody>
</table>

How It Works in SAS Viya

In addition to using the LDAP provider to obtain user and group information, you can configure SAS Logon Manager for Kerberos authentication. This option replaces the option to use the default LDAP provider for authentication to SAS Logon Manager. Kerberos provides the user with single sign-on capabilities from the browser on their desktop. Single sign-on allows the user to access the SAS Viya visual interfaces without being prompted to enter their credentials.

For information, see “Configure Kerberos (Linux Full Deployment)” on page 3.
Integrated Windows Authentication

Integrated Windows Authentication (IWA) uses Kerberos authentication and is a Microsoft technology that is used in an environment where users have Windows domain accounts. With IWA, the credentials are hashed before being sent across the network. The client browser proves its knowledge of the password through a cryptographic exchange with the web application server. When IWA is used in conjunction with Kerberos, IWA enables the delegation of security credentials. The Kerberos protocol uses strong cryptography so that a client can prove its identity to a server (and vice versa) across an insecure network connection.

Kerberos Authentication with CAS Scenarios

There are different scenarios in which user’s credentials are used to access a Hadoop environment that is secured by Kerberos. The following table provides an overview of each use case and links to additional information.

<table>
<thead>
<tr>
<th>End-User Client</th>
<th>Connection to SAS Viya</th>
<th>Who Runs CAS Session</th>
<th>Connection to Hadoop</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAS Viya visual interface</td>
<td>Kerberos (delegation)</td>
<td>End user</td>
<td>End user</td>
</tr>
<tr>
<td>See “Kerberos in SAS Viya Visual Interface (Delegation)” on page 55.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAS Viya visual interface</td>
<td>Kerberos</td>
<td>Service account (cas)</td>
<td>Service account (sascas)</td>
</tr>
<tr>
<td>See “Kerberos in SAS Viya Visual Interface (Outbound from CAS)” on page 57.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAS Viya visual interface</td>
<td>Kerberos</td>
<td>End user</td>
<td>End user</td>
</tr>
<tr>
<td>See “Kerberos in SAS Viya Visual Interface (Leveraging Stored Credentials)” on page 57.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAS Viya programming interface</td>
<td>User ID and password</td>
<td>End user</td>
<td>End user</td>
</tr>
<tr>
<td>See “Kerberos in SAS Viya Programming Interface with User Credentials” on page 58.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAS 9.4</td>
<td>User ID and password</td>
<td>End user</td>
<td>End user</td>
</tr>
<tr>
<td>See “Kerberos in SAS 9.4 with User Credentials” on page 60.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Kerberos in SAS Viya Visual Interface (Delegation)

The following figure illustrates this scenario:
Kerberos delegation to CAS, or user delegation, is a feature that allows a SAS Viya application to reuse the end-user credentials to access Kerberized systems. Delegation allows a server to forward a user’s credentials to the CAS server where they can be used to access other Kerberized services, such as Hadoop. By default, user delegation is not enabled and must be configured.

In this scenario, membership in the CASHostAccountRequired group notifies CAS that the user session needs to be launched under the user's operating system account and Kerberos delegation needs to take place. The user’s delegated Kerberos ticket can then be used for user access to Kerberized services as himself or herself.

Note: On Windows, user sessions are launched under the user identity. However, there are times when this is not possible. A session can also be launched under the CAS service account if a session cannot be launched under the user identity and the user requesting the session is able to assume the Superuser role.
See Also

- “Configure Kerberos for SAS Cloud Analytic Services” on page 7
- “The CASHostAccountRequired Custom Group” in SAS Viya Administration: Identity Management

Kerberos in SAS Viya Visual Interface (Outbound from CAS)

Note: This scenario is currently not supported on Windows.

In this scenario, the inbound authentication is a mechanism other than Kerberos, but the outbound authentication is performed using Kerberos. For inbound authentication, you can use any authentication mechanism that is supported by SAS Logon Manager, such as SAML or OAuth and OpenID Connect.

A registered principal for SAS Cloud Analytic Services (CAS) is required. This is not the account that is running the CAS server. The default principal name is sascas/cas_controller_hostname. An alternate principal name must be specified using the CAS_SERVER_PRINCIPAL environment variable. In addition, the principal must be mapped to a valid Hadoop user and permission granted in Hadoop.

A Kerberos keytab file for the service account is also required. It should contain only the credentials of the service account. The default location for the keytab is /etc/sascas.keytab. An alternative location must be defined using the KRB5_KTNAME environment variable.

Here is a list of the implications of outbound Kerberos with any other authentication mechanism used by SAS Logon Manager:

- Access to the secured Hadoop cluster is as the principal provided to CAS. No end-user credentials are available for access to Hadoop.

- Kerberos credentials can be automatically renewed by SAS. CAS initializes the credentials using the keytab.

- Authorizations that are set in SAS Environment Manager still apply to the end user. These authorizations are the only permissions applied since all access to Hadoop is as the service account.

Kerberos in SAS Viya Visual Interface (Leveraging Stored Credentials)

Note: This scenario is currently not supported on Windows.

In this scenario, the inbound authentication is a mechanism other than Kerberos. Stored credentials are used for outbound authentication, which is performed using Kerberos. For inbound authentication, you can use any authentication mechanism that is supported by SAS Logon Manager, such as SAML or OAuth and OpenID Connect.

A credential must be stored for the end user. This can either be individually or from a group membership. A custom group must also be defined in SAS Environment Manager. The group can be named anything, but the ID must be CASHostAccountRequired. In addition, the operating system must generate the Kerberos credentials through PAM. It is recommended that you use the System Services Security Daemon (SSSD).
Here is a list of the implications of outbound Kerberos with any other authentication mechanism used by SAS Logon Manager and a stored credential:

- Access to Secured Hadoop cluster is as the stored credentials. This could be an individual account or a shared account.

- Kerberos credentials might be automatically renewed by CAS. CAS attempts to renew the credentials. Alternatively, the operating system can provide options for renewal of credentials. For example, SSSD monitors and renews credentials before they expire.

- Authorizations set in SAS Environment Manager still apply to the end user.

Kerberos in SAS Viya Programming Interface with User Credentials

The following figure illustrates this scenario:
In this scenario, there is a full deployment and a user that provides his or her user ID and password to CAS. CAS uses its own pluggable authentication modules (PAM) configuration to validate the user’s credentials and launch the CAS controller process running as the user. In addition, the CAS controller also uses the user ID and password to obtain an OAuth token from SAS Logon Manager. The OAuth token provides the user’s group memberships from the Identities service. These memberships are essential in enforcing access control.

The PAM stack is configured to generate a Kerberos credentials cache during authentication. The resulting cache can be used to access Hadoop as the user.

Depending on the deployment options that you chose, users who access both the programming interface and the visual interface might have different access to Hadoop.
Kerberos in SAS 9.4 with User Credentials

In this scenario, end users provide their credentials to access SAS 9.4. SAS Workspace Server running SAS 9.4 is launched using a user ID and password, which are cached when SAS is launched. This enables SAS Workspace Server to use these cached credentials when connecting to CAS. The user credentials can also be provided by other sources in a SAS 9.4 environment, such as SAS Metadata Server or an authinfo file in the user’s home directory, because the process on the CAS controller is the same.

The user ID and password are validated through the PAM stack on the CAS controller and is used to generate an internal OAuth token from SAS Logon Manager running SAS Viya. The PAM stack is responsible for initializing the Kerberos credentials for the end user. These Kerberos credentials are placed into a Kerberos Ticket cache, which makes them available to the CAS session for the connection to the secured Hadoop environment. The different sessions within SAS 9.4, SAS Viya, and the secured Hadoop environment run as the end user.

Kerberos in SAS 9.4 with Delegation

In this scenario, SAS 9.4 is configured for Kerberos authentication. SAS Workspace Server running SAS 9.4 is launched using Kerberos credentials and the service principal for SAS Object Spawner running SAS 9.4 must be trusted for delegation. A Kerberos credential for the end user is available to SAS Workspace Server, which can be used to request a service ticket for the connection to CAS. CAS is provided with a Kerberos keytab and principal that it can use to validate this service ticket. Validating the service ticket authenticates the SAS 9.4 end user to CAS. The principal for CAS must also be trusted for delegation. CAS session must have access to the Kerberos credentials of the SAS 9.4 end user.

The Kerberos credentials that are made available to CAS are used to make a Kerberized connection to SAS Logon Manager running SAS Viya to obtain the SAS Viya internal OAuth token. Therefore, SAS Logon Manager running SAS Viya must be configured to accept Kerberos connections. For information about the configuration property that must be configured, see “sas.logon.kerberos” in SAS Viya Administration: Configuration Properties. In addition, the Kerberos credentials for the SAS 9.4 end user are used to connect to the secure Hadoop environment.

Since all the principals are trusted for delegation, the SAS 9.4 end user can be authenticated using Kerberos with each component in the SAS Viya and SAS 9.4 integrated environment. Through the use of Kerberos authentication, the SAS 9.4 end user is authenticated in to CAS and out to the secure Hadoop environment.

Kerberos in SAS 9.4 with One-Time Password

In this scenario, the SAS 9.4 session can be a SAS Stored Process Server, SAS Pooled Workspace Server, or SAS Workspace Server using server launch credentials. The SAS 9.4 session is not running as the end user and does not have access to the end-user credentials. You can still connect to CAS and to the secured Hadoop environment by configuring one-time passwords generated by SAS Metadata Server running on SAS 9.4. SAS Metadata Server running on SAS 9.4 must be aware of CAS. This is done by creating a CAS server definition in SAS Metadata Server, using the AUTHDOMAIN= argument. For more information, see CAS Statement Arguments.

SAS Logon Manager running on SAS Viya then passes the one-time password to SAS Web Infrastructure Platform running on SAS 9.4 to validate the password.
After the one-time password is validated, a SAS Viya internal OAuth token is generated and passed back to CAS.

CAS does not have access to the end-user credentials. Therefore, the session that is created is run using the account that is used to launch the controller process. By default, this account is cas. Since the end-user credentials are not available, the Kerberos credentials that are initialized for the session are from the Kerberos keytab provided to CAS. The connection to the secured Hadoop environment is made using those Kerberos credentials of the principal assigned to CAS.

### Fallback Authentication

Fallback authentication is available when SAS Logon Manager is configured for Kerberos authentication. Fallback authentication is enabled by default. Once Kerberos authentication is configured, no additional configuration is required to use fallback authentication.

Fallback occurs when the browser is unable to perform Kerberos authentication. At this point, other authentication mechanisms (such as LDAP, SAML, PAM, or OAuth and OpenID Connect) that are supported by SAS Logon Manager can be attempted. Multiple authentication methods can be used concurrently, in the same environment.

SAS Viya supports a custom fallback authentication security module. When the browser is not configured to perform Kerberos authentication, it falls back to the standard sign-in page and any other authentication mechanisms that are configured.

**Note:** Google Chrome and Microsoft Internet Explorer attempt NTLM authentication before falling back to another authentication mechanism. A dialog box might be displayed for the user to enter their credentials. If this happens, they can cancel out of the box to initiate fallback to the sign-in page.

If the user accesses the /SASLogon/login page in their browser (instead of being redirected to it), they will always get the sign-in page. The only way to initiate Kerberos authentication is to be redirected from another application to SASLogon. Therefore, if a user does not want to use Kerberos (for example, their browser does not support it), they can bookmark the sign-in page to always fallback without the additional prompt.

### See Also

- “Configure Kerberos (Linux Full Deployment)” on page 3
- “Authentication for Visual Interfaces” on page 47

### Kerberos Constrained Delegation

SAS Viya 3.5 introduces support for Kerberos constrained delegation. Kerberos constrained delegation can be used to authenticate in the following ways:

- to both CAS and SAS Compute Server from the SAS Viya 3.5 web applications
- from CAS to authenticate to data sources or from SAS Compute Server to CAS or data sources
- through a SAS 9.4 environment to CAS and onto your data source

Constrained delegation differs from unconstrained delegation in the following ways:

- The service does not require the user to forward either the Ticket-Granting Ticket (TGT) or the proxy ticket.
- The user does not need to authenticate using Kerberos, and the user does not need to have a TGT or a proxy service ticket.
Windows Local Group policy can be used to limit the services that can be delegated.

The client has no control over whether a service can delegate on behalf of the user. The client does not request delegation, nor does it pass a forwardable TGT to the service.

The client cannot detect that delegation is, or has been, performed.

Kerberos constrained delegation, or Service for User (S4U), is a Microsoft extension to the Kerberos protocol. S4U provides two extensions to the Kerberos protocol. Together, these extensions allow a service to obtain a Kerberos service ticket on behalf of a user.

- The Service for User to Proxy (S4U2proxy) extension allows a service to obtain a service ticket on behalf of a user to a different service.
- The Service for User to Self (S4U2self) extension allows a service to obtain a Kerberos service ticket to itself.

The following table compares the two types of Kerberos constrained delegation:

<table>
<thead>
<tr>
<th>Types of Kerberos Constrained Delegation</th>
<th>Constraints</th>
<th>Kerberos Realm</th>
</tr>
</thead>
<tbody>
<tr>
<td>traditional</td>
<td>The constraints are defined against the front-end service and control what back-end services the front-end service can delegate credentials to.</td>
<td>Both the front-end and back-end services must be in the same Kerberos realm.</td>
</tr>
<tr>
<td>resource-based</td>
<td>The constraint is configured on the back-end service. This enables the resource owner of the back-end service to configure what front-end services can delegate credentials to the back-end service.</td>
<td>The front-end services and back-end service do not have to be in the same Kerberos realm.</td>
</tr>
</tbody>
</table>

For information about configuring on Linux, see “Configure Kerberos Constrained Delegation in Active Directory” on page 10 and on Windows see “Configure Kerberos Constrained Delegation in Active Directory” on page 20.

OAuth and OpenID Connect Authentication (Linux Full Deployment)

Overview of OAuth and OpenID Connect

Open Authorization (OAuth) is a token-based authorization standard on the internet. OAuth 2.0 acts as an intermediary on behalf of the user, giving the third-party service an access token that authorizes specific account information. OpenID Connect is an extension to OAuth 2.0, which provides authentication support.
Key Terms

Table 18  Term Definitions

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access token</td>
<td>Specifies identifying information for a user, including the user’s credentials, groups, and privileges.</td>
</tr>
<tr>
<td>OpenID Connect</td>
<td>An authentication layer built on top of OAuth 2.0.</td>
</tr>
<tr>
<td>Flow</td>
<td>The process for obtaining an OAuth token.</td>
</tr>
</tbody>
</table>

How It Works in SAS Viya

An OAuth 2.0 and OpenID Connect provider can be internal to the customer’s environment, or it can be an external provider, such as Google Authenticator or Facebook. When the OAuth 2.0 option is configured, this does not completely replace the default LDAP provider. Instead, when users access SAS Logon Manager, they are presented with a link to authenticate using OAuth 2.0 and the standard sign-in page using the LDAP provider. Users can select which to use. The user identity and group membership information is looked up in LDAP. OAuth 2.0 can provide single sign-on from the OAuth 2.0 provider. For example, when a user signs in to his or her Google account, the user can access the visual interfaces of SAS Viya without being prompted any further for credentials.

Fallback authentication is available when SAS Logon Manager is configured for Kerberos authentication. If Kerberos authentication fails, OAuth and OpenID Connect authentication can be attempted. For more information, see “Fallback Authentication” on page 61.

Also available is simplified configuration of identity provider discovery for logins. The email address of the end user is used for redirection to the OpenID Connect Identity Provider. For more information, see "Configure Identity Provider Discovery for OIDC" on page 28 and "Identity Provider Discovery for OpenID Connect and SAML" on page 69.

See Also

- “Configure OIDC” on page 26
- “Configure Single Sign-On with Automatic Redirect” on page 34

SAML Authentication (Linux Full Deployment)

Note: Security Assertion Markup Language (SAML) is currently not supported for SAS Visual Analytics App (formerly called SAS Mobile BI) and SAS Add-in for MS Office connections.
Overview of SAML

The SAML standard defines a framework for exchanging security information about users between an identity provider and service provider. This security information is packaged in the form of portable XML assertions that applications working across security domain boundaries can trust. SAML allows for single sign-on to web browser applications.

Key Terms

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Federation</td>
<td>Allows multiple identity management systems to work together and establish trust.</td>
</tr>
<tr>
<td>Assertion</td>
<td>A package of information, in the form of an XML document, that is created and sent during a federated access request.</td>
</tr>
<tr>
<td>Claims</td>
<td>Information that a federation member is asserting to be true.</td>
</tr>
<tr>
<td>Identity provider</td>
<td>A federation member that authenticates users and keeps track of their information. Creates assertions for the users, and sends them to service providers.</td>
</tr>
<tr>
<td>Service provider</td>
<td>A federation member that consumes assertions to make access control decisions for its applications.</td>
</tr>
<tr>
<td>Metadata</td>
<td>An XML document that is produced by a SAML provider to describe its service endpoint URLs, x.509 certificate, and other information in a standard way for consumption by partners in the federation.</td>
</tr>
<tr>
<td>Relying party</td>
<td>A server providing access to secure software.</td>
</tr>
</tbody>
</table>

How It Works in SAS Viya

SAML supports configuring SAS Logon Manager to be integrated with an external SAML identity provider. This identity provider can be internal or external to the customer’s environment. If it is internal, a tool similar to Oracle Access Manager can be used. If it is external, something like salesforce.com can be used. SAML does not completely replace the default LDAP provider. End-users accessing SAS Logon Manager can choose SAML authentication or the default LDAP provider. The user identity and group membership information is looked up in LDAP. This option also provides single sign-on with the third-party SAML provider.

When a user attempts to access a service URL, the service provider initiates the exchange with an authentication request. The service provider is SAS Logon Manager. The identity provider sends a response that contains the assertion. The SAML protocol defines the structure and content of these request and response messages. When the user logs on to a service or system, the service provider trusts the identity provider to validate the credentials, instead of providing credentials to the service provider. Therefore, users do not have to provide their credentials directly to anyone but the identity provider.
The following functionality is available:

- Fallback authentication when SAS Logon Manager is configured for Kerberos authentication. If Kerberos authentication fails, SAML authentication can be attempted. For more information, see “Fallback Authentication” on page 61.

- Simplified configuration of identity provider discovery for logins. In prior releases, custom code and JavaScript were needed to configure this feature. Now, the email address of the end user is used for redirection to the SAML Identity Provider. For more information, see “Configure Identity Provider Discovery for SAML” on page 33 and “Identity Provider Discovery for OpenID Connect and SAML” on page 69.

- Identity provider initiated sign-on. This enables you to redirect into the SAS Viya environment and specify where you want the browser to go. The RelayState parameter is designed to be a state that the service provider can pass to the identity provider with the authentication request and get back in the response. The parameter can also be specified by the identity provider to indicate where you want the browser to go after redirecting to SAS Viya. Here is an example:
  
  - /SASVisualAnalytics/
  - /SASDrive/
  - /SASEnvironmentManager/

For more information, see “Identity Provider Initiated Logon for SAML” on page 70.

See Also

- “Configure SAML (Linux Full Deployment)” on page 29
- “Configure Single Sign-On with Automatic Redirect” on page 34

Single Sign-On with Automatic Redirect

When single sign-on (SSO) with automatic redirect is used with SAML or OpenID Connect, the configuration relies on part of the OpenID Connect specification. Since SAS Logon Manager is an OpenID Connect provider to the rest of the SAS web applications, this functions for both SAML and OpenID Connect third-party authentication.

Since the end user will not interact with SAS Logon Manager, they will not see the standard sign-in page. This means that end users will not be able to log in with LDAP credentials or the sasboot account. In the configuration, we will provide some options to enable certain clients to still have access to the sign-in page. Alternatively, end users can directly access the /SASLogon/login URL to access the sign-in page, they will then need to enter the URL for the application that they want to access after logging in.

The SSO configuration uses the login_hint option. The login_hint option is part of the OIDC specification and supported by SASLogon. This query string option is passed in the authorize request, which is /SASLogon/oauth/authorize. SAS Logon expects to receive an email domain in the hint, and this value is compared against a list of email domains configured for each SAML or OIDC identity provider (IdP). If there is a match, the user is redirected automatically to that provider, bypassing the sign-in page.

See Also

“Configure Single Sign-On with Automatic Redirect” on page 34
SAS 9.4 Authentication

Overview of SAS 9.4 Authentication

This option enables integration between SAS Viya and an existing SAS 9.4 environment. The authentication to the SAS Viya visual interfaces is performed by the SAS Logon Manager in SAS 9.4. None of the authentication occurs with the SAS Logon Manager in SAS Viya. Any authentication mechanism supported by SAS 9.4 is supported by this configuration. For more information about the supported authentication mechanisms, see SAS Intelligence Platform: Security Administration Guide.

Note: All versions of SAS 9.4 support this configuration. The SAS 9.4 deployment does not have to be running the latest maintenance release.

How It Works in SAS Viya

Here is a sample scenario:

1. The client’s web browser connects to SAS Logon Manager in SAS Viya.
   a. If the request to SAS Logon Manager in SAS Viya does not have an existing session, the SAS Logon Manager in SAS Viya displays the sign-in page, which contains a link to perform SAS 9.4 authentication and the page to do LDAP authentication.
   b. If the end user selects the link, SAS Logon Manager in SAS Viya constructs an authentication request and redirects the client’s web browser to the SAS 9.4 middle tier.

2. The client authenticates to SAS 9.4, receives a service ticket, and is redirected to SAS Logon Manager on SAS Viya.

3. The client’s web browser connects to SAS Logon Manager on SAS Viya, including the SAS 9.4 service ticket in the request.

4. SAS Logon Manager on SAS Viya connects to SAS 9.4 middle tier to validate the service ticket and the end user.

5. SAS Logon Manager on SAS Viya connects to the Identities service to get the custom and LDAP group information for the validated end user.

6. The Identities service either looks up the validated end user in its cache or connects to Active Directory using the LDAP service account to update the cache.

The SAS 9.4 authentication configuration impacts only the SAS Viya 3.4 visual interfaces using the SAS Logon Manager. An LDAP provider is still required by the Identities service. For authentication to SAS Logon Manager in SAS Viya that is not through a browser, the credentials are first passed to the SAS Logon Manager in SAS 9.4. If they fail, the credentials are tried against LDAP. Therefore, authentication with the administration command-line interface (CLI) and SAS Visual Analytics App (previously called SAS Mobile BI) is still authenticated against the SAS Logon Manager in SAS Viya first. SAS Studio 5.2 (Basic) is not impacted by this configuration.

If you want to configure TLS for either the SAS 9.4 or SAS Viya deployment, the Apache HTTP server certificate must be trusted. You need to import the certificate of the one deployment into the SAS certificate framework of the other deployment. For more information, see “Configure SAS 9.4 Clients to Work with SAS Viya” in Encryption in SAS Viya: Data in Motion.
Compatibility of User Names

The Identities service must be able to take the authenticated user name from SAS 9.4 and correctly search for it in the SAS Viya LDAP provider. You can sign on to SAS 9.4 using an internal account (which includes the @saspw suffix), but such accounts cannot exist in the LDAP provider. Therefore, these accounts do not work with SAS Viya.

Also, you can sign in to SAS 9.4 with an account that does not exist in any LDAP provider, such as a Google account. This does not work with SAS Viya unless the Google account is the accountId property that is used by the Identities service. For more information about the accountId property, see “sas.identities.providers.ldap.group (Field Mappings)” in SAS Viya Administration: Configuration Properties.

Finally, domain qualified user names cannot be used with SAS Viya. Even if the SAS 9.4 environment passed the domain qualified user name, the domain is stripped.

Single Sign-On and Single Sign-Out

Single sign-on and single sign-out is supported between SAS Viya and SAS 9.4. During single sign-on, a user with an active SAS 9.4 session can access SAS Viya applications without being required to sign on to SAS Viya.

Single sign-out is initiated from SAS Viya. If a user has two browser tabs open, one with a SAS Viya web application and the other with a SAS 9.4 web application, selecting the sign-out option in SAS Viya also signs the user out of SAS 9.4. However, the reverse is not true. If the user signs out from the SAS 9.4 web application, he or she is not signed out from the SAS Viya web application.

PAM Authentication (Linux)

Overview of PAM

PAM enables you to determine how applications use authentication to verify the identity of a user. It is an industry-standard technology that extends UNIX host authentication to recognize additional authentication providers. PAM uses modules or libraries to access multiple authentication methodologies. SAS Viya supports host authentication.

Account modules are required when SAS Logon Manager is configured for PAM. This ensures that the user is not authenticated with an expired password. Some authentication providers allow a user to use an expired password and address this in the account modules.

How It Works in SAS Viya

Default PAM configuration files, SAS-Viya-configuration-directory/etc/pam.d/service, are installed as a part of the SAS Viya deployment process.

Note: For SAS Cloud Analytic Services (CAS) server, service is cas. For SAS Studio, service is sasauth.

For sasauth to perform authentication, entries must be made in the PAM configuration files that are provided by SAS. These entries describe the authentication services that are used when sasauth performs an authentication. This includes the account and auth modules. The session and password modules are not supported.
In a multi-machine deployment, configure PAM on the host with SAS Object Spawner and the host with CAS controller.

Fallback authentication is available when SAS Logon Manager is configured for Kerberos authentication. If Kerberos authentication fails, PAM authentication can be attempted. For more information, see “Fallback Authentication” on page 61.

Authinfo File

Authentication is used to control access to the CAS server and its resources. Your identity must be successfully authenticated before your session is created. SAS Studio authenticates the connection to CAS by using your user credentials. When password information is not available, an attempt is made to find an authinfo file (.authinfo is the default file name on Linux). The authinfo file provides a user name and password to CAS for host authentication. It is an alternative to including passwords in programs.

You can also force the use of the authinfo file by specifying authinfo= in the CAS statement. An alternative method is to use the CAS_AUTH_METHOD environment variable.

The authinfo file is required when you are using the command line to submit commands for the following tasks:

- Run programs in batch mode. The USER= option in the CAS statement or SAS system option CASUSER= can be specified.
- Perform limited server administration using the casadmin command.
- Run commands in line mode.
- Sign on to SAS/CONNECT and specify the casuser in the RSUBMIT block of code. This action is performed when the casuser is different from the SAS Viya user or when the user is the same for both SAS Viya and CASUSER, but the password is different.

Note: SAS Studio user credentials are used to authenticate your connection to CAS. SAS Studio does not use the authinfo file for authentication.

Typically, the authinfo file resides in the $HOME directory.

The authinfo file format is based on the .netrc file specification. The .netrc file format is an older format. You can see the file specification at Netrc Format. In addition to the standard .netrc file standards, the authinfo specification allows for putting commands in the file as well as using quoted strings for passwords. The quoted strings allow for spaces within passwords.

If the authinfo file contains values that match the host, port, or user name. The information contained in the authinfo file is used to connect to CAS.

The following system options and environment variables can be used to override the authinfo file. These options point to authinfo files that are located in a different directory or are named differently.

Here are the ways that the AUTHINFO system option, environment variable, and the statement option can be used to override the authinfo file:

- Environment variable AUTHINFO takes precedence over the authinfo file.
- SAS system option AUTHINFO= (alias CASAUTHINFO=) overrides the AUTHINFO environment variable as well as the authinfo file.
- AUTHINFO= option in the CAS statement overrides the AUTHINFO= system option, the AUTHINFO environment variable, and the authinfo file.

For more information, see the following documents:
Multi-Factor Authentication

Multi-Factor authentication (MFA) is a security system that requires more than one method of authentication from independent categories of credentials to verify the user’s identity for a sign-on or other transaction.

MFA combines two or more of the following independent credentials:

- what the user knows – their password
- what the user has – a security token
- what the user is – biometric verification

The goal of MFA is to create a layered security defense, making it more difficult for an unauthorized person to access a target such as a physical location, computing device, network, or database.

Typical MFA scenarios include the following:

- swiping a card and entering a PIN
- logging on to a website and being requested to enter an additional one-time password (OTP) that the website's authentication server sends to the requester's phone or email address
- downloading a virtual private network (VPN) client with a valid digital certificate and logging on to the VPN before being granted access to a network
- swiping a card, scanning a fingerprint, and answering a security question
- attaching a universal serial bus (USB) hardware token to a desktop that generates a one-time passcode and using the one-time passcode to sign on to a VPN client

See Also

“Configure PAM (Linux)” on page 28.

Additional Authentication Topics

Identity Provider Discovery for OpenID Connect and SAML

By default, when one or more external identity providers (IdP) are configured, they are listed on the sign-in page under **Or sign in with:**. Each IdP is listed as a hyperlink that can be configured to display your choice of text. You can choose not to display the IdPs and instead configure IdP discovery. When configured, IdP discovery uses the domain of the user's email address to automatically select which IdP to use.
When IdP discovery is configured, the sign-in screen displays a **User ID or Email** field. If a user enters an email address, the domain from the email address is compared to the list of email domains that are configured for each IdP. If a match is found, the user is automatically redirected to that IdP, in the same way they would be if they clicked on the standard sign-in page. If no match is found, the standard **Sign In** page is displayed with the **User ID** and **Password** fields.

### Identity Provider Initiated Logon for SAML

You can update the SAML identity provider to redirect to the SAS Viya website that you want the user to go to, using the **RelayState** parameter.

IdP discovery streamlines logging in when multiple identity providers are configured. Instead of a sign-in page with the standard user name and password fields followed by a list of SAML providers, users are prompted to enter their user name or email address and click **Next**. If the emailDomain option is configured for an external identity provider, the user is automatically redirected to the provider. Otherwise, the user is prompted for a password. The emailDomain option is configured for SAML in **Step 1c** on page 31.

In the identity provider-initiated flow, use the **RelayState** parameter to specify the relative URL of a SAS web application to redirect to post-authentication. You must also ensure that the correct links are available for redirection to the SAS Viya environment.

**Note:** The relative URL requires a trailing slash (for example, /SASDrive/).

### SAS/CONNECT Authentication

As an administrator, you might want to enable SAS Viya to accept connections for existing SAS 9 environments. SAS/CONNECT enables that connection, and passes credentials that can be used in the SAS Viya environment.

With SAS Viya, your credentials are used to authenticate to CAS when you are using SAS/CONNECT. When additional SAS/CONNECT servers are spawned, SAS/CONNECT forwards your credentials to the spawned SAS/CONNECT server session.

Here are the ways that SAS/CONNECT and CAS authenticate your user credentials:

- The spawner passes the SIGNON credentials to the SAS/CONNECT server where the credentials can be used to connect to CAS in the following situations:
  - when the user is using any environment that is not a SAS Viya environment
  - when the user is connecting to SAS Viya via the SAS/CONNECT spawner
- When the user is in the SAS Viya environment using SAS Studio and starting SAS/CONNECT server sessions (using SASCMD SIGNON or the CONNECT Spawner), the CAS credentials (if they exist) are passed to the SAS/CONNECT server in SAS Viya.
- When running SAS Viya in batch or line mode, the authinfo file is used to authenticate to CAS. If you specified the USER= option in the CAS statement, CASUSER= system option, or if you specified the CAS_AUTH_METHOD environment variable, authinfo file authentication is used.

For more information, see the following documents:

- USER=user-ID
- CAS AUTH_METHOD environment variable on page 86
- SAS/CONNECT 9.4 User’s Guide
Single Sign-On (Full Deployment)

Single sign-on (SSO) is an authentication model that enables users to access a variety of computing resources without being repeatedly prompted for their user IDs and passwords. For example, SSO can enable a user to access SAS servers that run on different platforms without interactively providing the user’s ID and password for each platform. SSO can also enable someone who is using one application to launch other applications based on the authentication that was performed when the user initially logged on.

SAS Logon Manager is the central point for handling changes to authentication mechanisms, such as the addition of third-party SSO products. SAS Viya supports Kerberos, SAML, and OpenID Connect. For information about configuring these products, see:

- Kerberos (on Linux or on Windows)
- SAML
- OpenID Connect

Dual Authentication

Linux

In a dual authentication environment on Linux, users are validated against the LDAP server and the host authentication mechanism. The following conditions exist:

- If PAM is configured to use local accounts and those users also sign on to the visual components, then those local accounts must match the LDAP server used for SAS Logon Manager.
- If PAM is configured to use an LDAP server, SAS Logon Manager should be configured to use the same LDAP server.
- When directly connecting to the CAS server using SAS Studio or a batch job, the user ID and password that are supplied are authenticated against both the LDAP server and PAM.

Windows

In a dual authentication environment on Windows, users are validated against the LDAP server and the host authentication mechanism. The following conditions exist:

- The LDAP server should be configured to use the same Active Directory server that the Windows host is using.
- When directly connecting to the CAS server using SAS Studio or a batch job, the user ID and password that are supplied are authenticated against the LDAP server and host authenticated.
Authentication: Guest Access (Linux)

About Guest Access

Note: Guest access is currently not supported in a Windows deployment.

Guest access is an optional feature that provides anonymous Read-Only access to a subset of resources and functionality in participating applications. Guest access is supported for viewing reports in SAS Visual Analytics and SAS Visual Analytics App (previously called SAS Mobile BI).

For information about multi-tenancy, see "Enable Guest Access" in SAS Viya Administration: Multi-tenancy.

Enable Guest Access

Note: In a multi-tenancy environment, the following steps must be repeated for each tenant that supports guest access.

1 Set the sas.logon.provider.guest configuration property, using SAS Environment Manager:
   a In the applications menu (), select Administration ⇒ Manage Environment. In the navigation bar, select 🏷.
   b Create a new configuration instance for sas.logon.provider.guest, ensuring that you enable the guest access option. For more information, see “Create Configuration Instances” in SAS Viya Administration: Configuration Properties.

2 Add rules that provide the necessary access to functionality:
   a From the SAS Viya machine where the command line interfaces are installed, create a default profile, if you have not already created one, and sign on. For more information, see “Command-Line Interface: Preliminary Instructions” in SAS Viya Administration: Using the Command-Line Interfaces.
   b Modify the authorization rules.
      ■ For a new SAS Viya 3.5 installation, run the following command:
        sas-admin authorization facilitate-guest
      ■ For an upgrade from SAS Viya 3.4 to SAS Viya 3.5 in which guest access was not previously configured, run the following command:
        sas-admin authorization facilitate-guest
      ■ For an upgrade from SAS Viya 3.4 to SAS Viya 3.5 in which guest access was previously configured, complete the following steps:
Run the `facilitate-guest` command.

```
sas-admin authorization facilitate-guest
```

Output similar to the following is displayed:

The jsonPatch was not valid.
Http Status: 400
ErrorCode: 1177

Detailed Messages:
```
correlator: e607fd5d-c4c8-4548-ad2d-b9e608ccf41a
traceId: 49f41d99e62595f2
path: /authorization/rules
FieldError: Rule [id=<defined_id>, type=GRANT, permissions=[READ], principal=null, principalType=guest, containerUri=null, objectUri=/identities/users/@currentUser, mediaType=null, condition=null, filter=null, reason=null, description=Guest Access: XXX, isEnabled=true, matchParams=false, isShare=false]: Provided authorization rule is a duplicate of this rule.
```

Remove the rule ID that is specified in the output of the previous step:

```
sas-admin authorization remove-rule --id=<defined_id>
```

Run the `facilitate-guest` command again. If an error message is displayed stating "Provided authorization rule is a duplicate of this rule", repeat the previous step to remove the rule ID.

Repeat this step until the `facilitate-guest` command runs successfully.

For an upgrade from a release prior to SAS Viya 3.4 to SAS Viya 3.5, run the following command:

```
sas-admin authorization facilitate-guest
```

If you have a preexisting source file, modify the direct access controls for the predefined caslibs on the server. Use the controls that are defined in the specified source file. The following command must be executed by a user who is a member of the Superuser role.

```
sas-admin cas facilitate-guest --source-file path-to-controls-file --server CAS-server-name --superuser
```

If you do not have a preexisting source file, you can generate one that contains the default access controls, make modifications to it, and use it as the source file. The following command must be executed by a user who is a member of the Superuser role.

```
sas-admin cas generate-guest-controls --output-location /path/
sas-admin cas facilitate-guest --source-file path-to-controls-file --server serverA --superuser
```

Generate a source file from the default access controls. The generated source file is named `facilitate-guest-controls.txt`.

Make the desired modifications to access controls to the source file that you just generated.

Modify the direct access controls using the source file that you just modified.

Add access controls that provide Read access to caslibs that should be accessible to guest users:

- From the SAS Viya machine, if you have not already signed in to SAS Viya, sign on using the default profile that was created in the previous step.

- Run the following commands as a user who is a member of the Superuser role:

```
sas-admin cas caslibs add-control --server server-name --caslib caslib-name --grant readInfo --guest --superuser
```
sas-admin cas caslibs add-control --server server-name --caslib caslib-name
--grant select --guest --superuser
sas-admin cas caslibs add-control --server server-name --caslib caslib-name
--grant limitedPromote --guest --superuser

6 Use SAS Environment Manager to grant Read access to folders and reports that should be accessible to guest users:

   a From the **Content** page, identify the folder to which you want to grant Read access to guest users.

   b Right-click and select **Edit authorization**.

   c Click ⌁ and select **Add Guest**. Grant Read and Read (convey) access. For more information, see “General Authorization: How To (Authorization Window)” in *SAS Viya Administration: General Authorization*.

   d Click **Save**.

---

**Note:** From the **Content ➔ Users ➔ guest** page, you can move folders and objects into the **My Folder** folder for the guest user. You can also create and add folder and report shortcuts into the **My Favorites** and **My Folder** folders. For more information, see “Folders: How To” in *SAS Viya Administration: Folders*.

---

**Connect as Guest Users**

Once guest access is enabled, guest users can view reports using SAS Visual Analytics and SAS Visual Analytics App. SAS Visual Analytics displays a guest sign-in button. SAS Visual Analytics App displays a guest sign-in button when a mobile connection is established.

**See Also**

- SAS Report Viewer 8.3 Documentation
- SAS Visual Analytics: Viewing Reports
- SAS Visual Analytics App Documentation

---

**Generate Custom Links to Reports**

You can create a custom web link for guest users, allowing them to access a specific report. If guest access is enabled, the custom link is configured to bypass the sign-in page and automatically connect the user as guest. If guest access is disabled, a sign-in page is displayed, where users can choose to connect as a guest or sign-in with their credentials.
Generate Custom Links to Reports Using SAS Report Viewer

1. From SAS Report Viewer, open the report to which you want to generate a link.
2. Click ‹ and then select Share report ➔ Link.
3. In the Generate Link window, customize the link, if necessary, in the Link field.
4. Click Copy Link. You can paste the link and distribute to guest users.

See Also
SAS Report Viewer 8.3 Documentation

Generate Custom Links to Reports Using SAS Visual Analytics

1. From SAS Visual Analytics, open the report to which you want to generate a link.
2. Click ‹ and then select Copy Link.
3. In the Copy Link window, customize the link, if necessary, using the Options selections.
4. Click Copy Link. You can paste the link and distribute to guest users.

See Also
SAS Visual Analytics: Viewing Reports

Disable Guest Access

1. Set the sas.logon.provider.guest configuration property, using SAS Environment Manager:
   a. In the applications menu (☰), select Administration ➔ Manage Environment. In the navigation bar, select 📋.
   b. From the Definitions view, select sas.logon.provider.guest.
   c. Click ✎. In the Edit sas.logon.provider.guest Configuration window, select the option to disable guest access.
      
      Note: The sas.logon.provider.guest option is tenant-specific and must be disabled for each tenant.
      
   d. Click Save.
2  (Optional) Remove the rules that provide the necessary access to functionality:
   a  From the SAS Viya machine, navigate to the `SAS-Viya-installation-directory/home/bin` directory.
   b  At the command prompt, create a default profile and sign on by entering the following commands:
      ```
sas-admin profile init
sas-admin auth login
```
   c  Modify the authorization rules by running the following command:
      ```
sas-admin authorization disable-guest-access
```

Note: This command removes the rules that were automatically loaded by the `facilitate-guest` command. If you manually created any custom rules, using either SAS Environment Manager or the command-line interface, you must manually remove those rules. A list of the remaining guest rules can be viewed on the SAS Environment Manager Rules page.

3  (Optional) Run the following commands as a user who is a member of the Superuser role to remove CAS Access grants:
   ```
sas-admin cas sessions create --server server-name --name clisession --superuser

sas-admin cas caslibs remove-control --server server-name --caslib VAModels
     --grant readInfo --guest --session-id session-id

sas-admin cas caslibs remove-control --server server-name --caslib VAModels
     --grant select --guest --session-id session-id

sas-admin cas caslibs remove-control --server server-name --caslib VAModels
     --grant limitedPromote --guest --session-id session-id

sas-admin cas caslibs remove-control --server server-name --caslib ReferenceData
     --grant readInfo --guest --session-id session-id

sas-admin cas caslibs remove-control --server server-name --caslib ReferenceData
     --grant select --guest --session-id session-id

sas-admin cas caslibs remove-control --server server-name --caslib ReferenceData
     --grant limitedPromote --guest --session-id session-id

sas-admin cas caslibs remove-control --server server-name --caslib AppData
     --grant readInfo --guest --session-id session-id

sas-admin cas caslibs remove-control --server server-name --caslib AppData
     --grant select --guest --session-id session-id

sas-admin cas caslibs remove-control --server server-name --caslib AppData
     --grant limitedPromote --guest --session-id session-id

sas-admin cas caslibs remove-control --server server-name --caslib Formats
     --grant readInfo --guest --session-id session-id

sas-admin cas caslibs remove-control --server server-name --caslib Formats
     --grant select --guest --session-id session-id
```
sas-admin cas caslibs remove-control --server server-name --caslib Formats --grant limitedPromote --guest --session-id session-id
sas-admin cas sessions delete --server server-name --session-id session-id

Note: These commands remove the grants that were automatically defined by the facilitate-guest command. If you manually created any custom grants, using either SAS Environment Manager or the command-line interface, you must manually remove those grants.

You can also remove guest access with the `cas remove-guest-controls` command. However, this command removes only the default set of direct access controls and not any other guest access controls that you might have applied. To view which direct access controls are removed, see the `facilitate-guest-controls.txt` file that is generated by the `cas generate-guest-controls` command.

### Authentication: OpenID Connect with ISAM Scenario (Linux Full Deployment)

In the following tasks, OpenID Connect uses IBM Security Access Manager (ISAM) WebSEAL reverse proxy server as the single sign-on entry point for initial user authentication. Other providers can be used, but configuration instructions are not provided here. To configure the OAuth and OpenID Connect, complete the following sections:

#### Configure OpenID Connect Provider Properties for IBM Security Access Manager

1. From SAS Environment Manager, navigate to the SAS Logon Manager configuration definitions. For more information, see “Edit Authentication Configuration Instances” on page 37.

2. In the Definitions list, select `sas.logon.oauth.providers.external_oauth`.

3. In the top right corner of the window, click New Configuration.

4. In the New `sas.logon.oauth.providers.external_oauth` Configuration dialog box, enter values for the required fields, based on your environment. Table 20 on page 77 provides guidance about the information needed for the listed fields.

<table>
<thead>
<tr>
<th>Configuration Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>addShadowUserOnLogin</td>
<td>A local shadow user should be added once authentication is successful.</td>
</tr>
<tr>
<td>attributeMapping.user_name</td>
<td>The attribute claim to use as the user name. For ISAM, use sub.</td>
</tr>
<tr>
<td>Configuration Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>authUrl</td>
<td>The URL to the authorization endpoint (for example, <a href="https://hostname.example.com/isam/oidc/endpoint/amapp-runtime-ISAMOP/authorize">https://hostname.example.com/isam/oidc/endpoint/amapp-runtime-ISAMOP/authorize</a>).</td>
</tr>
<tr>
<td>emailDomain</td>
<td>Specifies a comma separated list of email domains for users that can sign in with the OpenID Connect provider. It is used with identity provider discovery and is optional.</td>
</tr>
<tr>
<td>issuer</td>
<td>The principal that issued the token, specified as a case-sensitive string or URI. This is your WebSEAL instance (for example, the reverse proxy entry point, <a href="https://oidcidp.example.com">https://oidcidp.example.com</a>).</td>
</tr>
<tr>
<td>linkText</td>
<td>The text that should be displayed on the sign-in page for the provider (for example, OpenID Connect Login Using ISAM Reverse Proxy [WebSEAL]).</td>
</tr>
<tr>
<td>relyingPartyId</td>
<td>The client ID that is registered with the provider.</td>
</tr>
<tr>
<td>relyingPartySecret</td>
<td>The secret that is registered with the provider for the client ID.</td>
</tr>
<tr>
<td>scopes</td>
<td>The comma-delimited list of scopes for the authorization request. The list should contain openid. Note: SAS Viya does not process any additional scopes that are returned in the token.</td>
</tr>
<tr>
<td>showLinkText</td>
<td>The link text should show on the sign-in page.</td>
</tr>
<tr>
<td>tokenUrl</td>
<td>The URL to the token endpoint.</td>
</tr>
<tr>
<td>type</td>
<td>The protocol type. By default, the value is oidc1.0. Note: SAS Viya requires an id_token in the authorization response from the provider. However, some providers return an id_token when the scope in the authorization request is openid and response_type=token. For those providers, use type oauth2.0.</td>
</tr>
<tr>
<td>tokenKey</td>
<td>Specifies the HMAC key or RSA public key that is used to sign tokens.</td>
</tr>
<tr>
<td>tokenKeyUrl</td>
<td>Specifies the URL to obtain the token key.</td>
</tr>
</tbody>
</table>

5 Click Save.

6 Restart the SAS Logon Manager Service.

- For Red Hat Enterprise Linux 6.7:
  ```
sudo service sas-viya-saslogon-default restart
  ```

- For Red Hat Enterprise Linux 7.x or later and SUSE Linux:
  ```
sudo systemctl restart sas-viya-saslogon-default
  ```

Note: It might take several minutes to restart SAS Logon Manager.
Configure OpenID Connect Provider in IBM Security Access Manager

For basic steps to configure OpenID Connect in ISAM 9.0.3.1, search for the Access Manager Federation Cookbook 9.0.0.0- 9.0.3.0 at IBM Security Community.

To configure OpenID Connect Provider, complete the following steps:

1. In the ISAM 9.0.3.x admin console, create the WebSEAL reverse proxy instance as a single sign-on entry point.

2. Configure an OpenID Connect Provider and its partner.
   An OpenID Connect Provider on ISAM is a federation. First create a federation that represents the OpenID Connect Provider. Then, create a partner that represents the SAS Viya application under it.

3. Create a federation for OpenID Connect Provider. Table 21 on page 79 displays the values that you should provide while creating the new federation.

   Table 21 Create New Federation Values

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Federation Name</td>
<td>ISAMOP</td>
</tr>
<tr>
<td>Protocol for this federation</td>
<td>OpenID Connect</td>
</tr>
<tr>
<td>Role</td>
<td>OpenID Connect Provider</td>
</tr>
<tr>
<td>Issuer Identifier</td>
<td><a href="http://www.oidcidp.example.com">www.oidcidp.example.com</a></td>
</tr>
<tr>
<td></td>
<td>Note: This is your WebSEAL instance.</td>
</tr>
<tr>
<td>Signature Algorithm</td>
<td>HS256</td>
</tr>
<tr>
<td>Grants</td>
<td>Authorization Code</td>
</tr>
<tr>
<td>Identity Mapping</td>
<td>Do not perform identity mapping. The same user name exists both in ISAM LDAP and SAS Viya LDAP.</td>
</tr>
</tbody>
</table>

4. Create an OpenID Connect Provider Partner for SAS Viya (SASLogon). Table 22 on page 79 displays the values that you should provide while creating the new partner.

   Table 22 Create New Partner Values

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>ISAM-to-SASViya</td>
</tr>
<tr>
<td>Field Name</td>
<td>Value</td>
</tr>
<tr>
<td>------------------------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>Enabled</td>
<td>Yes</td>
</tr>
<tr>
<td>Connection Template</td>
<td>OIDC</td>
</tr>
<tr>
<td>Client ID</td>
<td>isamClientID</td>
</tr>
<tr>
<td>Client Secret</td>
<td>isamClientSecret</td>
</tr>
<tr>
<td>Client Display Name</td>
<td>SAS Viya Client</td>
</tr>
<tr>
<td>Response Types</td>
<td>code, id-token token, and token</td>
</tr>
<tr>
<td>Allow Refresh Token Grant</td>
<td>Enabled</td>
</tr>
<tr>
<td>Redirect URIs</td>
<td><a href="https://sas-viya-host/SASLogon/login/callback/external_oauth">https://sas-viya-host/SASLogon/login/callback/external_oauth</a></td>
</tr>
<tr>
<td>Scope</td>
<td>openid</td>
</tr>
</tbody>
</table>

5 Test your configuration by accessing SAS Environment Manager. The text that you specified in the *linkText* field in *Step 4 on page 77* should be displayed

---

OpenID Connect and IBM Security Access Manager

The following diagram depicts the IBM Security Access Manager reverse proxy components and process flow.
In this figure, the numbered arrows correspond to the following activities:

1. A client browser (user agent) accesses SAS Logon Manager (OAuth client).
2. SAS Logon Manager redirects the client browser to the SAS Logon Manager sign-in page. The end user clicks **OpenID Connect logon using ISAM Reverse Proxy (WebSEAL)**.
3. The client browser sends an authentication request to WebSEAL (resource owner).
4. WebSEAL redirects the client browser to the IBM Security Access Manager (ISAM) sign-in page. The end user provides their authentication information.
5. The client browser sends the authentication information to the Lightweight Directory Access Protocol (LDAP) server.
6. The LDAP server authenticates the user with IBM Security Access Manager.
7. WebSEAL sends an authorization request to the ISAM federation run time (WebSphere Application Server).
8. The ISAM federation run time sends the authorization request to the authorization server.
9. The authorization server sends an authorization code to the ISAM federation run time.
10. The ISAM federation run time sends the authorization code to WebSEAL.
11. WebSEAL sends the authorization code to the client browser.
12. The client browser sends the authorization code to SAS Logon Manager.
13. SAS Logon Manager sends a request to the ISAM federation run time to convert the authorization code to an access token.
The ISAM federation run time sends the request to the authorization server.
The authorization server sends the access token to the ISAM federation run time.
The ISAM federation run time sends the access token to SAS Logon Manager.

---

Authentication: OpenID Connect with Okta Scenario (Linux Full Deployment)

The following tasks provide one way to configure Okta as an OpenID Connect provider. In this scenario, OpenID Connect uses Okta as the single sign-on entry point for initial user authentication. Other providers can also be used, but configuration instructions are not provided here. To configure the OAuth and OpenID Connect, complete the following sections:

Create the Web Application Using the Okta Admin Console

1. Log on to Okta Developer Console and click **Admin**.
2. In the Okta Dashboard, navigate to **Applications** ⇒ **Add Application**.
   a. In the Create New Application window, select **Web**, and then click **Next**.
   b. Under Application Settings, enter values for the fields, based on your environment. **Table 23 on page 82** provides guidance about the information needed for the listed fields.

**Table 23  Web Application Settings and Suggested Values**

<table>
<thead>
<tr>
<th>Setting</th>
<th>Suggested Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Provide a name that identifies your application with Okta.</td>
</tr>
<tr>
<td>Base URIs</td>
<td>Leave this field blank.</td>
</tr>
<tr>
<td>Login redirect URIs</td>
<td>Specifies the URI of the SAS Logon Manager OAuth configuration, where Okta redirects after authentication. Use <code>https://hostname.example.com/SASLogon/login/callback/external_oauth</code>.</td>
</tr>
<tr>
<td>Logout redirect URIs</td>
<td>Specifies the URI of the SAS Logon Manager OAuth configuration. This value must match the Login redirect URI.</td>
</tr>
<tr>
<td>Group assignments</td>
<td>Select the users to whom you want to grant access.</td>
</tr>
<tr>
<td>Grant type allowed</td>
<td>Select <strong>Authorization Code</strong>.</td>
</tr>
</tbody>
</table>
3 In the Okta Dashboard, navigate to API ⇒ Authorization Servers.

Note: The following steps might differ, depending on your Okta configuration.

a In the API window, select default ⇒ Claims and click Add Claim.

b In the Add Claim dialog box, enter values for the fields, based on your environment. Table 24 on page 83 provides guidance about the information needed for the listed fields.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Suggested Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Provide a name for your claim. For Okta, use sub_sas.</td>
</tr>
<tr>
<td>Include in token type</td>
<td>Select ID Token.</td>
</tr>
<tr>
<td>Value type</td>
<td>Select Expression.</td>
</tr>
<tr>
<td>Value</td>
<td>Specify the following sting: String.substringBefore(user.email,&quot;@&quot;).</td>
</tr>
<tr>
<td>Disable claim</td>
<td>Do not select this setting.</td>
</tr>
<tr>
<td>Include in</td>
<td>Select Any scope.</td>
</tr>
</tbody>
</table>

c Click Create.

Configure OpenID Connect Provider Properties for Okta

1 Display the configuration settings for your Okta authorization server.

   a Log on to Okta Developer Console and click Admin.

   b In the Okta Dashboard, navigate to API ⇒ Authorization Servers.

   c In the API window, select default.

   d Under Settings, note the value of the Metadata URI setting.

2 In a web browser, navigate to URL that was noted in Step 1d on page 83 (for example, https://hostname.example.com/oauth2/default/.well-known/oauth-authorization-server).

   The configuration settings for your authorization server are displayed. You will specify these settings in SAS Environment Manager.

3 Log on to SAS Environment Manager.
Note: In a multi-tenant environment, the following must be configured for the tenant.

a From SAS Environment Manager, navigate to the SAS Logon Manager configuration definitions. For more information, see “Edit Authentication Configuration Instances” on page 37.

b In the Definitions list, select sas.logon.oauth.providers.external_oauth.

c In the top right corner of the window, click New Configuration.

d In the New sas.logon.oauth.providers.external_oauth Configuration dialog box, enter values for the required fields, based on your environment. Table 25 on page 84 provides guidance about the information needed for the listed fields. Some of your values might differ, depending on your Okta configuration.

Table 25 OpenID Connect Configuration Fields and Descriptions

<table>
<thead>
<tr>
<th>Configuration Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>addShadowUserOnLogin</td>
<td>A local shadow user should be added once authentication is successful.</td>
</tr>
<tr>
<td>attributeMapping.user_name</td>
<td>The attribute claim to use as the user name. For Okta, use sub_sas. Note: The value sub_sas can be substituted with another claim that was created during the Okta configuration. This value must be the same as the Name specified in Step 3b on page 83.</td>
</tr>
<tr>
<td>authUrl</td>
<td>The URL to the authorization endpoint (for example, <a href="https://hostname.example.com/oauth2/default/v1/authorize">https://hostname.example.com/oauth2/default/v1/authorize</a>).</td>
</tr>
<tr>
<td>emailDomain</td>
<td>Specifies a comma-separated list of email domains for users that can sign in with the OpenID Connect provider. It is used with identity provider discovery and is optional.</td>
</tr>
<tr>
<td>issuer</td>
<td>The principal that issued the token, specified as a case-sensitive string or URI. This is your Okta instance (for example, the entry point, <a href="https://oidcidp.example.com">https://oidcidp.example.com</a>). To see the value for your server, see Step 2 on page 83.</td>
</tr>
<tr>
<td>linkText</td>
<td>The text that should be displayed on the sign-in page for the provider (for example, OpenID Connect Login Using Okta).</td>
</tr>
<tr>
<td>relyingPartyId</td>
<td>The client ID that is registered with the provider. This value was noted in Step 2c on page 83.</td>
</tr>
<tr>
<td>relyingPartySecret</td>
<td>The client secret that is registered with the provider for the client ID. This value was noted in Step 2c on page 83.</td>
</tr>
<tr>
<td>scopes</td>
<td>The comma-delimited list of scopes for the authorization request. The list should contain openid. Note: SAS Viya does not process any additional scopes that are returned in the token.</td>
</tr>
<tr>
<td>showLinkText</td>
<td>The link text should show on the sign-in page.</td>
</tr>
</tbody>
</table>
### Configuration Field

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tokenUrl</td>
<td>The URL to the token endpoint.</td>
</tr>
<tr>
<td>type</td>
<td>The protocol type. By default, the value is <strong>oidc1.0</strong>. Note: SAS Viya requires an id_token in the authorization response from the provider. However, some providers return an id_token when the scope in the authorization request is <code>openid</code> and <code>response_type=token</code>. For those providers, use type <strong>oauth2.0</strong>.</td>
</tr>
<tr>
<td>tokenKey</td>
<td>Specifies the HMAC key or RSA public key that is used to sign tokens.</td>
</tr>
<tr>
<td>tokenKeyUrl</td>
<td>Specifies the URL to obtain the token key.</td>
</tr>
</tbody>
</table>

4. Click **Save**.

5. Test your configuration by accessing SAS Environment Manager. The text that you specified in the `linkText` field in **Step 3d on page 84** should be displayed.

---

### Authentication: Passwords

#### Update Account Passwords on Windows

#### SAS Cloud Analytic Services

If you change the SAS Cloud Analytic Services (CAS) service account password on the Windows domain, you must complete the following:

1. If you use a single account for the HTTP service and CAS service accounts, re-create the keytab file.

2. For any Windows service that is running as the CAS user, update the password.
In Windows Services Manager, right-click the service name and select **Properties**.

b In the Properties dialog box, select the **Log On** tab.

c Update the password and click **OK**.

3 Run the `encryptCasUser.bat` command to update the deployment scripts. For more information, see “Specify Credentials for the cas User Account” in *SAS Viya for Windows: Deployment Guide*.

### SAS Infrastructure Data Server

If you change the PostgreSQL user account password on Windows, you must complete the following:

1 For any Windows service that is running as the PostgreSQL user, update the password.
   a In Windows Services Manager, right-click the service name and select **Properties**.
   b In the Properties dialog box, select the **Log On** tab.
   c Update the password and click **OK**.

2 Regenerate the credentials file each time the password changes. For more information, see “Specify Credentials for the postgres User Account” in *SAS Viya for Windows: Deployment Guide*.

### Authentication: Reference

### CAS Environment Variables for Clients

The environment variables in this section are set on the client and affect how the client authenticates with the CAS server.

**CAS_AUTH_METHOD=authinfo | kerberos**

specifies the authentication method that CAS clients use.

<table>
<thead>
<tr>
<th>Valid in</th>
<th>operating system command line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category</td>
<td>Security</td>
</tr>
<tr>
<td>Linux specifics</td>
<td>Environment variables on Linux are case-sensitive.</td>
</tr>
</tbody>
</table>

**See**

[Authinfo File Authentication](#)

**Examples**

In these examples, the CAS client is forced to authenticate using the credentials in the authinfo file (Kerberos authentication is not attempted). Here are two examples of specifying the command for Linux.

- `export CAS_AUTH_METHOD=authinfo`
- `set CAS_AUTH_METHOD=kerberos`
CAS Environment Variables for Administrators

The environment variables in this section affect authentication with the CAS server.

**env.CASSTRIPAUTH=1**
strips the @domain that is a part of the user name when authenticating a user name and password with SASLogon.

This environment variable helps in resolving a mismatch between host authentication and LDAP authentication when host authentication requires "user@domain" format user names, and LDAP authentication requires "user" format user names without the @domain part with SASLogon.

Valid in  cas_usermods.settings
Category  Environment
Default  Does not strip the domain name that is specified after the @ character in the user name.
Example  Here is an example where the user name is user01@company.com. It is authenticated as: 'user01'

**CASUSERIGNORECASE='ON'**
when in effect (specified using any value), causes the CAS server to ignore the letter casing for user names during authentication, group lookup, and process launch. Always specify CASUSERLOWERCASE whenever specifying CASUSERIGNORECASE, unless instructed otherwise by SAS Technical Support.

The typical scenario for declaring CASUSERIGNORECASE is when users run their CAS sessions under their own host account and the user authentication system is configured to be case-insensitive and contains uppercase or mixed case user names. For more information, see “The CASHostAccountRequired Custom Group” in SAS Viya Administration: Identity Management.

Valid in  cas_usermods.settings file
Category  Administration
Default  off
Restrictions  Applies to Linux only.
Requirement  Use with CASUSERLOWERCASE.
Note  To turn off CASUSERIGNORECASE, remove its definition.
Example  In this example, CASUSERIGNORECASE is in effect:

```bash
export CASUSERIGNORECASE='on'
```

**CASUSERLOWERCASE='ON'**
when in effect (specified using any value), causes the CAS server to convert user names to lowercase during group lookup. CASUSERLOWERCASE is typically used in conjunction with CASUSERIGNORECASE.
The typical scenario for declaring `CASUSERLOWERCASE` is when users run their CAS sessions under their own host account and the user authentication system is configured to be case-insensitive and contains uppercase or mixed case user names. For more information, see "The CASHostAccountRequired Custom Group" in SAS Viya Administration: Identity Management.

<table>
<thead>
<tr>
<th>Valid in</th>
<th><code>cas_usermods.settings</code> file</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category</td>
<td>Administration</td>
</tr>
<tr>
<td>Default</td>
<td>Off</td>
</tr>
<tr>
<td>Restrictions</td>
<td>Applies to Linux only.</td>
</tr>
</tbody>
</table>
| Requirement    | Use with `CASUSERIGNORECASE`.
| Note           | To turn off `CASUSERLOWERCASE`, remove its definition. |
| Example        | In this example, `CASUSERLOWERCASE` is in effect: `export CASUSERLOWERCASE='on'` |

**CASUSERFORCELOWER=\"ON\"**

when in effect (specified using any value), causes the CAS server to convert user names to lowercase. Using `CASUSERFORCELOWER` eliminates the need for either `CASUSERIGNORECASE` or `CASUSERLOWERCASE`.

The typical scenario for declaring `CASUSERFORCELOWER` is when users run their CAS sessions under their own host account and the user authentication system is configured to be case-insensitive and contains uppercase or mixed case user names. For more information, see "The CASHostAccountRequired Custom Group" in SAS Viya Administration: Identity Management.

<table>
<thead>
<tr>
<th>Valid in</th>
<th><code>cas_usermods.settings</code> file</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category</td>
<td>Administration</td>
</tr>
<tr>
<td>Default</td>
<td>Off</td>
</tr>
<tr>
<td>Restriction</td>
<td><code>CASUSERFORCELOWER</code> is case sensitive.</td>
</tr>
<tr>
<td>Note</td>
<td>To turn off <code>CASUSERFORCELOWER</code>, remove its definition.</td>
</tr>
<tr>
<td>Example</td>
<td>In this example, <code>CASUSERFORCELOWER</code> is in effect: <code>export CASUSERFORCELOWER='on'</code></td>
</tr>
</tbody>
</table>

---

See Also

“Where Do I Set CAS Environment Variables?” in SAS Viya Administration: SAS Cloud Analytic Services
Authentication: External Languages Package

Overview

Because the functionality of the EXTLANG package might conflict with your site’s security policies, the package includes mechanisms that enable administrators of SAS Cloud Analytic Services (CAS) to control users access to installed external-language interpreters and usable storage areas. Administrators can do the following:

- Specify whitelisted directories, which are directories that users can write to or that contain source code that users are allowed to load.
- Control whether users can specify paths to the executables for interpreting and compiling external-language programs.
- Control whether users can insert external-language source code directly into their SAS programs.
- Control whether users can specify environment variables for their external-language programs.

By taking advantage of the hierarchical nature of XML, users can specify attributes as concisely as possible on a global, per-group, per-language, or per-group-per-language basis. The following sections describe the configuration file that is used for access control. The path to this XML-formatted file is specified using the CAS environment variable, SAS_EXTLANG_SETTINGS. For information about where to set environment variables for your particular CAS deployment, see "CAS Environment Variables" in SAS Viya Administration: SAS Cloud Analytic Services.

External Languages Access Control Configuration

By default, the EXTLANG package is disabled, all text attributes are set to an empty string, and all Boolean values are set to 'BLOCK'. The following XML schema can be configured in the configuration file. The EXTLANG tag is required, but the other tags are optional.

**EXTLANG**

Contains attributes that control the default settings for nested tags. Users can specify the following attributes in the EXTLANG tag:

- **version=level**
  Specifies the XML document’s version number. This attribute exists to accommodate future changes to the XML definition.

- **mode=’ALLOW’ | ’ANARCHY’ | ’BLOCK’**
  Specifies the access mode for the EXTLANG package. Possible values for this attribute are:

  - **’ALLOW’**
    Allows use of the EXTLANG package and initializes all Boolean properties to 'BLOCK'. (The properties can be explicitly overridden).
'ANARCHY'  
Allows use of the EXTLANG package and initializes all Boolean properties to 'ALLOW'. (the properties can be explicitly overridden).

'BLOCK'  
Does not allow use of the EXTLANG package.

allowAllUsers='ALLOW' | 'BLOCK'  
Controls which users can run external-language programs. Possible values for this attribute are:

'ALLOW'  
Allows all users to use the EXTLANG package. As the administrator, you can still specify GROUP blocks to specify group-specific overrides. For users who are not specified in a GROUP block, the attributes from the DEFAULT tag are applied.

'BLOCK'  
Allows only users specified in a GROUP block to use the EXTLANG package.

The following tags are all optional:

DEFAULT  
Enables users to specify global default settings. Any attributes that are specified in the DEFAULT block override the values that are initialized according to the mode= attribute that is specified in the EXTLANG start tag. These settings cascade down to all LANGUAGE and GROUP blocks that are nested in the EXTLANG block. The following attributes can be specified in the DEFAULT tag:

scratchDisk=location  
Specifies a location on the file system in which to store temporary files that are used to enable support for external languages. This location must have enough space to store shared variables (that is, variables that are transferred between the SAS and external-language environments). Environment variables are expanded. By default, the system-defined default temporary directory is used.

diskWhitelist=paths  
Specifies one or more file system paths that users can use for the following:

- Loading source code from files.
- Specifying as their scratchDisk.
- Specifying as options specific to certain objects. For example, the 'TEMPDIR' and 'EXECPATH' options of the PYTHON2, PYTHON3, and R objects of the EXTLANG package

Users can specify multiple paths by separating them with the platform-specific path separator, which is ":" on Linux systems. Environment variables are expanded. The escape character is the "\". A literal "\" can be inserted using "\\". Attempting to push a file that is not inside a whitelisted directory or its subdirectories will result in an error. The diskWhiteList= attribute does not affect the ability to insert external-language source code in other ways. For example, users can still insert code inline. Attempting to specify a path for the scratchDisk= attribute that is not inside this directory will result in an error.

userSetScratchDisk='ALLOW' | 'BLOCK'  
Controls whether users are allowed to specify the path to the location that is specified in the scratchDisk= attribute. By default, userSetScratchDisk='BLOCK'.

'ALLOW'  
Allows users to specify the scratch disk.

'BLOCK'  
Prevents users from specifying the scratch disk. A run-time error will occur if a user program attempts to set the scratch disk.
userSetEnv='ALLOW' | 'BLOCK' | 'UNDERSCORE'
Controls whether users are allowed to specify environment variables to be passed to the external-language program via the AddEnvVariable method. By default, userSetEnv='BLOCK'. The following values can be specified:

'ALLOW'
Allows users to specify environment variables whose names consist of a string of ASCII characters.

'BLOCK'
Does not allow users to specify environment variables.

'UNDERSCORE'
Allows users to specify environment variables whose name consists of an underscore followed by a string of ASCII characters.

textVersion='ALLOW' | 'BLOCK'
Controls whether users can specify the external-language executable’s path. By default, userSetInterpreter='BLOCK'.

'ALLOW'
Allows users to specify the path to the external-language interpreter.

'BLOCK'
Does not allow users to specify the path to the -language interpreter. An error will occur if a user attempts to specify the path.

LANGUAGE
Enables users to override default settings on a per-language basis. Users can nest one or more LANGUAGE tags directly in the DEFAULT block or within a GROUP block. Any attribute specified in a LANGUAGE tag becomes the default value for that language (that is, those attributes cascade down for all GROUP blocks that do not specify them for that language). If a LANGUAGE tag is nested within a GROUP block, the specified attributes apply only to users within that group.

name='PYTHON2' | 'PYTHON3' | 'R'
Specifies the name of the language being configured. Users must specify this attribute.

Users can also specify the following optional attributes in the LANGUAGE tag. Unspecified attributes inherit the corresponding attribute value from the DEFAULT block, which inherits default values according to the mode attribute of the EXTLANG start tag:

interpreter=path
Specifies the default path to the external-language executables. This path will be used by all workers in the CAS cluster. Attempting to use objects of the EXTLANG package will result in an error if the interpreter and userSetInterpreter attributes are not specified.

userInlineCode='ALLOW' | 'BLOCK'
Controls whether users can add external-language code from within their SAS programs.

'ALLOW'
Allows users to insert inline code.

'BLOCK'
Does not allow users to insert inline code. By default, userInlineCode='BLOCK'.

userSetEnv='ALLOW' | 'BLOCK' | 'UNDERSCORE'
Controls whether users can specify environment variables. Acceptable values are:

'ALLOW'
Allows users to specify environment variables whose names consist of a string of ASCII characters.

'BLOCK'
Does not allow users to specify environment variables. By default, userSetEnv='BLOCK'.
'UNDERSCORE'  
Allows users to specify environment variables whose name consists of an underscore followed by a string of ASCII characters.

userSetInterpreter='ALLOW' | 'BLOCK'  
Controls whether users can specify the path to the language executable.

   'ALLOW'  
   Allows users to set the interpreter executable.

   'BLOCK'  
   Does not allow users to set the interpreter executable. An error will occur if a user attempts to specify the path to the interpreter.

ENVIRONMENT  
Enables users to set environment variables. Users can nest one or more ENVIRONMENT tags within a LANGUAGE tag. Environment variables are set in the external-language interpreter’s running environment. In order for environment variables to be passed, the value of the userSetEnv attribute of the enclosing LANGUAGE block must be 'ALLOW' or 'UNDERSCORE'.

Note: A run-time error will occur under these conditions:

- if userSetEnv='BLOCK' and a user attempts to set an environment variable
- if userSetEnv='UNDERSCORE' and a user attempts to set an environment variable whose name does not begin with an underscore

Administrators can specify variables that begin with underscore regardless.

Users must specify the following attributes in the ENVIRONMENT tag:

   name=variableName  
   Specifies the environment variable name, which must be a string of ASCII characters

   value=variableValue  
   Specifies the value of the variable, which must be a string of Unicode characters

GROUP  
Specifies group-specific overrides. Every user must be specified in a GROUP block, unless allowAllUsers='ALLOW' in the EXTLANG tag. Users cannot belong to multiple groups. Any languages that do not contain a LANGUAGE block that is enclosed by the GROUP block inherit the default language attributes that are defined in the corresponding LANGUAGE block that is enclosed by the DEFAULT block. The default language attributes can themselves be implicitly defined according to the mode attribute of the EXTLANG start tag.

   name=groupName  
   Specifies a string that identifies the group. If no users attribute is specified in the GROUP tag, the attributes that are defined in this GROUP block are applied to the operating system account whose user name is groupName. Users must specify the following attribute in the GROUP tag.

Users can also specify the following attributes of the GROUP start tag:

   users=list  
   Specifies a comma-delimited list of user names to which the settings that are defined in this GROUP block will apply. Each user must belong to only one group. The escape character is "\". A literal "\" can be inserted using "\\".

   scratchDisk=directory  
   Specifies the temporary working directory for this group. By default, the location that is specified in the scratchDisk= attribute in the DEFAULT block is used.
diskWhiteList=paths

Specifies a list of whitelisted paths. Users can specify multiple paths by using the operating system's native path separator (which is ':' on Linux). If this attribute is specified, source code that the users of this group push from a file must reside under a path in this list. The location that is specified in the scratchDisk= attribute must also reside under a path in this list.

userSetScratchDisk='ALLOW' | 'BLOCK'

Specifies whether the users in this group are allowed to specify the scratch disk.

'ALLOW'

Allows users in this group to set the scratchDisk location.

'BLOCK'

Does not allow users to set the scratchDisk location. An error will occur if a user attempts to specify the scratchDisk location.

Sample Access Control File for the EXTLANG Package

Content similar to the following can be defined in the access control configuration file:

```xml
<EXTLANG version="1.0" mode="ALLOW" allowAllUsers="BLOCK">
  <DEFAULT  scratchDisk="/smalldisk/sas/scratch"
            diskWhiteList="/secure/sas/allowed_scripts:/whitelist"
            userSetScratchDisk="BLOCK"
            userSetEnv="BLOCK"
            userSetInterpreter="BLOCK">
    <LANGUAGE name="PYTHON2"
              interpreter="/some/path/python2"
              userSetEnv="ALLOW"
              userSetInterpreter="BLOCK">
      <ENVIRONMENT name="PYTHONPATH" value="/some/path1:/some/path2" />
      <ENVIRONMENT name="LDLIBRARYPATH" value="/some/ldpath1:/some/ldpath2" />
    </LANGUAGE>
    <LANGUAGE name="PYTHON3"
              interpreter="/some/path/python3"
              userSetEnv="BLOCK"
              userSetInterpreter="BLOCK">
      <ENVIRONMENT name="PYTHONPATH" value="/some/path1:/some/path2" />
      <ENVIRONMENT name="LDLIBRARYPATH" value="/some/ldpath1:/some/ldpath2" />
    </LANGUAGE>
    <LANGUAGE name="R"
              interpreter="/some/path/Rscript"
              userSetEnv="BLOCK"
              userSetInterpreter="BLOCK">
      <ENVIRONMENT name="LDLIBRARYPATH" value="/some/ldpath1:/some/ldpath2" />
    </LANGUAGE>
  </DEFAULT>
  <GROUP name="DanDLyons"  scratchDisk="$HOME/scratch" />
  <GROUP name="SassySean"
          userSetInterpreter="ALLOW"
          userSetEnv="UNDERSCORE" />
</EXTLANG>
```
The following provides details about the above sample code:

- In the EXTLANG tag:
  - Specifying mode='ALLOW' enables the EXTLANG package and initializes all Boolean attributes to 'BLOCK'.
  - Specifying allowAllUsers='BLOCK' prohibits users whose user name does not appear in a GROUP from using the package.

- In the DEFAULT tag, restrictive settings initialized in the EXTLANG tag are overridden. Five settings are overridden in this block:
  - The scratchDisk= attribute is set to /smalldisk/sas/scratch. All temporary files that the EXTLANG package creates will go here; it must be large enough to accommodate all data sets that every user must work with.
  - The diskWhitelist= attribute specifies that users can push code files that reside only in the /smalldisk/sas/scratch and /whitelist directories.
  - The userSetScratchDisk, userSetEnv, and userSetInterpreter attributes are set to 'BLOCK'; this is done for readability since they were all initialized by mode='ALLOW' in the enclosing EXTLANG tag.

- In the LANGUAGE tags, default language settings are entered.
  - For PYTHON2, the interpreter path is specified and users are not allowed to change this setting in their program (unless the setting is overridden in a GROUP block). Users are not able to set environment variables. Default values are then defined for 'PYTHONPATH' and 'LDLIBRARYPATH' environment variables.
  - The settings for PYTHON3 are the same as for PYTHON2, except the interpreter executable path is different.
  - A similar configuration is specified for R, but only the 'LDLIBRARYPATH' environment variable is set.

- GROUP tags are used to override default settings for users.
  - The first GROUP block does not have a users= attribute, so the settings that it overrides will apply to user DanDLyons. All external-language programs run by DanDLyons will be stored in the scratch subdirectory within DanDLyons’ home directory.
  - The next GROUP block does not have a users= attribute, so the settings that it overrides apply only to user SassySean. This user is assigned a different default Python 3 interpreter, which SassySean can change programmatically because userSetInterpreter='ALLOW'. The default
The final GROUP tag specifies multiple users to which the specified attributes apply. These users' temporary data will be stored in /pan1 and they can push only scripts that are under their home directory. The userInLineCode= attribute is overridden to 'ALLOW', so these users can add code in their SAS program. The remainder of the settings are inconsequential because they match the defaults.

Authentication: Troubleshooting

After configuring Kerberos for SAS Logon Manager, you are unable to log on to a visual interface, such as SAS Environment Manager.

Resolution:
You must use a web browser on a different machine. Once Kerberos is enabled on Windows, a browser running on the same machine where the services are deployed cannot connect to SAS Viya visual interfaces.

The Kerberos authentication handshake fails and a session is not launched.

Resolution:
Users can store their credentials from the My Credentials page. Then, if the Kerberos handshake fails, authentication will fallback to the stored credentials in DefaultAuth. For more information, see “Add New Credentials” in SAS Viya Administration: External Credentials.

After Kerberos is configured for SAS Logon Manager, no one is able to log on to SAS Environment Manager.

Resolution:
If the information that you specified while adding Kerberos to the active profile, profiles.active, is incorrect or missing, the only way to change the information is by using the SAS Bootstrap Config CLI.

Run the following command on a single line. Multiple lines are used for each command to improve readability.

```
/opt/sas/viya/home/bin/sas-bootstrap-config --token-file $consul-token kv write --force config/SASLogon/spring/profiles.active ldap,postgresql
```

For more information, see “Use SAS Bootstrap Config CLI on Consul to Manage the KV Store and ACL Tokens” in Encryption in SAS Viya: Data in Motion.

Linux group lookup fails when user names are uppercase or mixed case.

Resolution:
This problem typically occurs when Active Directory is used as the back-end user store. Use the CASUSERIGNORECASE environment variable to force SAS Cloud Analytic Services (CAS) to ignore letter casing during authentication and session launch. In addition, use the CASUSERLOWERCASE environment variable to force CAS to use the lowercase version of the user's name when doing the group lookup. For more information, see “CASUSERIGNORECASE="ON"” on page 87.