<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Appendix 1 / Hadoop Deployment: Configuring SAS Access to Hadoop and SAS Data Connector to Hadoop</td>
<td>49</td>
</tr>
<tr>
<td>Pre-deployment Hadoop Tasks</td>
<td>49</td>
</tr>
<tr>
<td>Configure SAS/ACCESS to Hadoop and SAS Data Connector to Hadoop</td>
<td>51</td>
</tr>
<tr>
<td>Remove with Kubernetes</td>
<td>47</td>
</tr>
</tbody>
</table>
Steps for a Successful Deployment

Before You Begin

Because the contents of this guide are subject to continual updates, make sure that you have the latest guide. You can always access the latest release of this guide from the following site:

SAS Viya Deployment Guides

If you accessed this guide directly from the Software Order Email (SOE), you are viewing the latest guide. If you are viewing a saved copy of the PDF version of this guide, the content might be outdated.

Follow the instructions in this guide to deploy a predefined SAS Viya image, which is a Docker image. Also, instructions for adding Jupyter Notebook to the image are included.

Instead of using the predefined SAS Viya image, you can create a custom Docker image that includes SAS Viya software of your choosing. This method of deployment is called using a recipe. For information about using recipes and what is supported, see the following websites:

- SAS for Containers
- SAS Product Support for Virtualization Environments

To use this guide successfully, you should have a working knowledge of Docker or Kubernetes.

Make sure that you have the SOE, which includes information about the software order and file attachments.

The SAS Viya image contains a programming-only deployment. A programming-only deployment supports data scientists and programmers who use SAS Studio or direct programming interfaces such as Python or REST APIs. Understand that this type of deployment does not include SAS Drive, SAS Environment Manager, and the complete suite of services that are included with a full deployment.

Step 1 — Prepare for the Deployment

Perform one of the following tasks:
To update the container, go directly to "Update Your Software" on page 37.

To deploy a new instance, continue with the following the steps.

2 Go to “System Requirements” on page 5 to learn about requirements for the virtual infrastructure, data sources, security, and more.

3 Go to “Pre-deployment Tasks” on page 13 to prepare your environment for the deployment.

**Step 2 — Deploy the Software**

1 Go to “Deploying SAS Viya” on page 19 to deploy the image.

2 Go to “Post-deployment Tasks” on page 27 to configure your environment.

3 Go to “Validating the Deployment” on page 31 to verify that services are running and that you can access the software.

Note: To locate documentation for administrative tasks and usage information, see “Completing the Deployment” on page 35.

**Step 3 — Configure Access to Hadoop**

If you deployed SAS to access data in Hadoop, see “Hadoop Deployment: Configuring SAS Access to Hadoop and SAS Data Connector to Hadoop” on page 49. This configuration is most likely performed by a Hadoop administrator.

---

**SAS Products and Supporting Components**

This guide provides information for deploying the following software:

| SAS Cloud Analytic Services for Viya 3.4 | SAS/ACCESS Interface to Hadoop (on SAS Viya) |
| SAS Econometrics 8.3 | SAS/ACCESS Interface to ODBC (on SAS Viya) |
| SAS In-Database Technologies for Hadoop (on SAS Viya) | SAS/ACCESS Interface to PC Files (on SAS Viya) |
| SAS Optimization 8.3 | SAS/ACCESS Interface to PostgreSQL (on SAS Viya) |
| SAS Studio 4.4 | SAS/CONNECT (on SAS Viya) |
| SAS Visual Analytics (on SAS Viya) 8.3 | SAS/IML (on SAS Viya) |
| SAS Visual Data Mining and Machine Learning 8.3 | SAS/QC (on SAS Viya) |
| SAS Visual Statistics (on SAS Viya) 8.3 |

Note: Consider the following to understand what gets deployed:

- Your Software Order Email (SOE) lists products and supporting components to which you are entitled. Other software to which you are entitled, such as SAS Studio, might not be listed in the SOE.

- The predefined SAS Viya image contains a programming-only deployment, which includes the SAS procedures and programming functionality for the products listed. The graphical user interfaces of SAS Visual Analytics and SAS Visual Statistics are not included.
Contact SAS Technical Support

Technical support is available to all customers who license SAS software. However, you are encouraged to engage your designated on-site SAS support personnel as your first support contact. If your on-site SAS support personnel cannot resolve your issue, have them contact SAS Technical Support to report your problem.

Before you contact SAS Technical Support, explore the SAS Support website at support.sas.com/techsup/. This site offers access to the SAS Knowledge Base, as well as SAS communities, Technical Support contact options, and other support materials that might answer your questions.

When you contact SAS Technical Support, you are required to provide information, such as your SAS site number, company name, email address, and phone number, that identifies you as a licensed SAS software customer.
System Requirements

Virtual Infrastructure Requirements .................................................. 6
   Overview of Infrastructure Requirements .................................. 6
   General Host Machine Requirements ..................................... 6
   Hardware Requirements for the Host Machine .......................... 6
   Runtime Environment .......................................................... 6
   Orchestration Requirements ..................................................... 7

Third-Party Software Requirements .............................................. 7
   Java .............................................................................. 7

Data Source Requirements .......................................................... 7
   Overview of Data Source Requirements ............................... 7
   Supported Data Sources ....................................................... 8
   Hadoop Requirements ......................................................... 8
   Requirements to Transfer Data from SAS 9.4 ......................... 8
   Requirements for SAS/ACCESS Interface to Hadoop ............... 9
   Requirements for SAS In-Database Technologies for Hadoop .... 9
   Requirements for SAS/ACCESS Interface to ODBC .................. 9
   Requirements for SAS/ACCESS Interface to PC Files ............... 9
   Requirements for SAS/ACCESS Interface to PostgreSQL .......... 9

Security Requirements ............................................................... 10
   Authentication Options ....................................................... 10
   Default Security Settings ..................................................... 10

User and Group Requirements ..................................................... 10
   Requirements for the User Who Performs the Deployment .......... 10
   CAS Administrator Account Requirements ............................ 10
   User Account Requirements ............................................... 11
   User Accounts .................................................................. 11

Client Requirements ................................................................. 12
   Web Browsers .................................................................... 12
   Mobile Platform and Touchscreen Support ............................ 12
   Database Drivers .............................................................. 12
   Screen Resolution ............................................................. 12
Virtual Infrastructure Requirements

Overview of Infrastructure Requirements
Requirements for the deployment of SAS Viya for Containers are minimal because most of the required components are included.

The topics in this section will assist you in preparing your environment for the deployment.

General Host Machine Requirements
The Docker host (the machine where the container will be created) must be running Linux. Windows is not supported for the deployment. This machine must be a physical or virtual machine. In this release, SAS Viya for Containers cannot be installed on a managed container service such as AWS Elastic Container Service (ECS).

At least 10 GB of free disk space must be available on the Docker host machine.

For a deployment with Kubernetes, reserve at least 10 GB of free disk space on at least one of the Kubernetes worker nodes. You will run commands to create the images on these machines.

Hardware Requirements for the Host Machine
Hardware requirements for SAS products are typically dependent on the number of users who access the SAS Viya for Containers environment and the amount of data that is imported and analyzed.

Here are the minimum hardware requirements for the host machine where the container is running:

<table>
<thead>
<tr>
<th>Resource</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>VMs</td>
<td>1</td>
</tr>
<tr>
<td>VCPUs</td>
<td>4 cores</td>
</tr>
<tr>
<td>RAM</td>
<td>1 - 4 GB</td>
</tr>
<tr>
<td></td>
<td>The amount of RAM required depends on the workload and number of concurrent users.</td>
</tr>
<tr>
<td>Disk Space</td>
<td>10 - 11 GB</td>
</tr>
<tr>
<td></td>
<td>This value assumes that data is not stored in the file systems inside the running container.</td>
</tr>
</tbody>
</table>

Runtime Environment
SAS Viya for Containers can be deployed using either Docker or Kubernetes. The following versions are supported:

- Docker version 17.05.0 or later
- Kubernetes 1.10 or later
Both Docker Community Edition (CE) and Docker Enterprise Edition (EE) are supported. SAS Viya for Containers has been tested with Docker 17.05.0-ce.

To check the Docker version on your machine:

```bash
docker -v
```

To determine your version of Kubernetes:

```bash
kubectl version
```

SAS Viya for Containers includes a CentOS 7 distribution for Docker.

**Orchestration Requirements**

SAS Viya for Containers supports deployments in private Docker registries.

You can use Docker and a shell script to perform the deployment. SAS recommends using `docker run` as the deployment method for deployments that use Docker.

For deployments of SAS Viya for Containers with Kubernetes, `kubectl` is the recommended method. You can deploy the SAS Viya for Containers image in a private registry using `kubectl run`.

**Third-Party Software Requirements**

**Java**

A Java Runtime Environment (JRE) must be available in your deployment. Only the JRE is required; the full JDK is not required. SAS provides a recent release of the OpenJDK JRE in the container.

You have the option to provide your own JRE in a custom-built Docker image. For a list of supported JRE distributions and other requirements, see: [https://support.sas.com/en/documentation/third-party-software-reference/viya/34/support-for-jre.html](https://support.sas.com/en/documentation/third-party-software-reference/viya/34/support-for-jre.html).

Third-party distributions of the JRE are supported as long as the version matches the one that is listed on the SAS Support website. However, IBM SDK, Java Technology Edition is not supported.

The current JRE options for SAS Viya for Containers have been tuned for OpenJDK and Oracle JRE. If you use a JRE from another vendor and experience performance issues, SAS might recommend using OpenJDK or Oracle JRE.

**Data Source Requirements**

**Overview of Data Source Requirements**

The SAS Viya image includes software to enable data retrieval from one of the supported data sources. The software that is included depends on your software order and consists of one or more SAS/ACCESS products or a SAS In-Database Technologies products.

You might be required to provide the following additional software:

- The database client for your associated database software. You might need to add a layer to the SAS Viya image that includes the database client.
- Drivers or other requirements for the SAS data connector that is included with SAS/ACCESS for use with your data source. The appropriate SAS data connector is provided on the SAS Viya image.
Refer to the section that corresponds to your SAS/ACCESS product or SAS In-Database Technologies product for additional system requirements that apply.

**Supported Data Sources**

SAS Viya for Containers supports the following data sources:

- Apache Hive
- Data sources that are accessible with an ODBC driver
- PostgreSQL

**Note:** The required SAS/ACCESS interface products might have individual system requirements.

SAS Viya for Containers also supports the following data sources, which use data connectors that are automatically included with CAS and are not separately licensed or configured.

- SASHDAT on UNIX
- LASR Analytic Server (SAS 9.4)
- SAS Scalable Performance Data Engine (SPDE)
- SAS Data Sets

SAS Viya for Containers also supports CSV files. Files of this type do not require a SAS data connector and can be accessed directly.

For Hadoop, HDFS is not supported.

**Hadoop Requirements**

**Supported Distributions and Connection Requirements**

SAS Viya for Containers supports multiple third-party distributions of Hadoop.


For Hive, SAS/ACCESS Interface to Hadoop and possibly SAS In-Database Technologies for Hadoop are required. These products have individual system requirements, which are documented below.

**SAS Support for Alternative Releases of Hadoop Distributions**

SAS identifies the specific set of Hadoop distributions that are supported with each SAS product release. The SAS policy that applies to alternative releases or distributions of Hadoop is documented at the following website: [http://support.sas.com/resources/thirdpartysupport/9–4/hadoop/alternative-hadoop-distributions.html](http://support.sas.com/resources/thirdpartysupport/9–4/hadoop/alternative-hadoop-distributions.html). The same policy that applies to SAS 9.4 also applies to SAS Viya for Containers.

**Requirements to Transfer Data from SAS 9.4**

For SAS 9.4 deployments that are earlier than SAS 9.4 TS1M5 (SAS 9.4M5), SAS/CONNECT is required in the environment in order to transfer data from other SAS deployments and operating systems to SAS Viya. SAS/CONNECT can convert data from a non-UTF-8 encoded SAS session to the UTF-8 format that SAS Viya requires.

SAS/CONNECT is not included with a standard SAS Viya order, and must be separately licensed.

By contrast, SAS 9.4M5 is integrated with SAS Viya directly. As a result, SAS/CONNECT is no longer required in order to transfer data from SAS 9.4M5. All SAS programming clients in a 9.4M5 environment can call procedures that are enabled in SAS Viya and submit DATA step code, operating directly on CAS data sources.
Examples of SAS programming clients are SAS Studio, SAS Enterprise Guide, SAS Data Integration Studio, and SAS Data Management Studio.

SAS/CONNECT is still supported, but if you are running SAS 9.4M5, it is no longer required in order to transfer data into SAS Viya.

**Requirements for SAS/ACCESS Interface to Hadoop**

SAS/ACCESS Interface to Hadoop (on SAS Viya) includes SAS Data Connector to Hadoop.


**Requirements for SAS In-Database Technologies for Hadoop**

SAS In-Database Technologies for Hadoop includes SAS Data Connect Accelerator for Hadoop. SAS/ACCESS Interface to Hadoop is required and is separately licensed.

The SAS Embedded Process is included with SAS In-Database Technologies for Hadoop. The SAS Embedded Process must be installed on the Hadoop cluster. SAS recommends installing the latest version of the SAS Embedded Process.

For more information about supported Hadoop versions and additional requirements, see: [https://support.sas.com/en/documentation/third-party-software-reference/viya/34/support-for-databases.html](https://support.sas.com/en/documentation/third-party-software-reference/viya/34/support-for-databases.html).

**Requirements for SAS/ACCESS Interface to ODBC**

SAS/ACCESS Interface to ODBC (on SAS Viya) includes SAS Data Connector to ODBC. SAS/ACCESS Interface to ODBC enables access to multiple data source types by means of a generic ODBC driver.


**Requirements for SAS/ACCESS Interface to PC Files**

SAS/ACCESS Interface to PC Files (on SAS Viya) includes SAS Data Connector to PC Files.

SAS/ACCESS Interface to PC Files enables access to the following file formats:

- `.jmp`
- `.spss`
- `.stata`
- `.xlsx` or `.xls`

No additional software is required.

**Requirements for SAS/ACCESS Interface to PostgreSQL**

SAS/ACCESS Interface to PostgreSQL (on SAS Viya) includes SAS Data Connector to PostgreSQL.

Security Requirements

Authentication Options

By default, a containerized deployment of SAS Viya is secured by requiring host authentication to SAS Studio. Additional security, including LDAP authentication, requires a new container, which you can build based on the SAS Viya image in the Docker repository. Using this method, you can add an authentication layer, such as the Linux System Security Services Daemon (SSSD), and integrate the authentication scheme with LDAP. For more information, see "(Optional) Enable Host Authentication" on page 16.

SAS Viya for Containers includes a template that configures SSSD to integrate with the identity and authentication provider at your organization. Other methods are also supported for integration with an authentication provider. The template provides guidance for setting the required SSSD properties.

Default Security Settings

In a programming-only deployment, TLS is not enabled by default. As a result, network connections from the Apache HTTP Server to backend services are not encrypted.

User and Group Requirements

Requirements for the User Who Performs the Deployment

The Docker process always runs as a user with root privileges. As a result, the user who performs the deployment requires sudoers privileges in order to run many of the required commands.

Instead of granting sudoers privileges to any user who will run Docker commands, you can create a Linux group named docker on the Docker host machine. Any users that you add to this group will automatically have Read/Write ownership of the Docker process. They will be able to run Docker commands without using sudo.

For more information about the docker group, see https://docs.docker.com/install/linux/linux-postinstall/.

CAS Administrator Account Requirements

An administrative user is the only type of user that can log on to the CAS Server Monitor user interface in order to manage CAS settings. Therefore, a CAS administrator must be configured when the Docker image is launched. You can designate a user account for the CAS administrator by specifying the desired user ID for the CASENV_ADMIN_USER variable.

A user account with password access to the SAS Studio user interface can be created automatically when the container is launched. This user account, named sasdemo, is created only if the image is started with the option RUN_MODE=developer. In addition, the user name sasdemo must be specified for the CASENV_ADMIN_USER variable. The sasdemo user automatically becomes the designated CAS administrator and does not require additional configuration.

The password for the sasdemo user is randomly generated and cannot be changed. You can retrieve it from the deployment log files. For information about finding the password for the sasdemo user, see "Deploy with Docker" on page 19.
During the deployment process, you can specify a different user account and password for the CASENV_ADMIN_USER variable. Do not define RUN_MODE when launching the container if you want to designate a user other than sasdemo as the CAS administrator.

For any alternative user that you designate for CASENV_ADMIN_USER, the user ID and an identical password must also be configured in LDAP. A valid LDAP account is required by the CAS server.

**User Account Requirements**

End users will log on to the SAS Studio user interface. SAS Studio supports pluggable authentication modules (PAM) and requires host (operating system) accounts for authentication. By default, the SAS Viya for Containers image does not contain any host authentication software. For information about how to enable host authentication, see “(Optional) Enable Host Authentication” on page 16.

Here are the requirements for SAS Studio users:

- Accounts that exist only on the LDAP server cannot log on to SAS Studio by default.
- Each SAS Studio user must have a valid host account for the virtual machine on which the SAS Studio web application runs.
- SAS Studio users also require an LDAP account in order to access CAS. The passwords for the LDAP and host accounts must be identical.
- Each user must log on with an account that has a home directory. SAS Studio requires home directories.

If you have added a host authentication layer to the container using the System Security Services Daemon (SSSD), multiple users will be able to log on to the container using their LDAP credentials. You can designate one of these users as the CAS administrator by specifying the LDAP user ID of the selected user as the value of the CASENV_ADMIN_USER variable. For more information, see “CAS Administrator Account Requirements” on page 10.

**User Accounts**

This table identifies and describes SAS Viya for Containers user accounts.

<table>
<thead>
<tr>
<th>Account Name and Group</th>
<th>Description</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>sas; member of sas group</td>
<td>A service account without user restrictions. The sas group is an administration group and is not a general user group.</td>
<td>Required for the installation and is created automatically. The installation process sets user and group ownership permissions on all the installation files. This user must exist to enable ownership. It enables required components to run. The sas group is intended to allow access to administrative features, such as logs and backup. It is the group owner of many files on disk. If you create additional user accounts, restrict membership in this group to administrators.</td>
</tr>
<tr>
<td>cas; member of sas group</td>
<td>The process owner of CAS processes.</td>
<td>Required for managing and enabling CAS and is created automatically.</td>
</tr>
</tbody>
</table>
### Account Name and Group

<table>
<thead>
<tr>
<th>Account Name and Group</th>
<th>Description</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>sasdemo; member of sas group</td>
<td>The default CAS administrator. It also enables logon.</td>
<td>If the RUN_MODE variable is set to developer when the container is started, this user account is created automatically. This user is provided to enable a rapid start-up and does not require any LDAP configuration to be able to log on to CAS Server Monitor.</td>
</tr>
</tbody>
</table>

### Client Requirements

#### Web Browsers

End users can access the product user interfaces for SAS Viya applications from a desktop computer, using a supported web browser. Because SAS software is not installed on this machine, the requirements are minimal. UNIX and 64-bit Windows operating systems are supported.

Some SAS Viya user interfaces include some advanced features that require recent versions of popular web browsers. For information about supported web browsers and the corresponding platforms to access SAS user interfaces, see: [https://support.sas.com/en/documentation/third-party-software-reference/viya/34/support-for-web-browsers.html](https://support.sas.com/en/documentation/third-party-software-reference/viya/34/support-for-web-browsers.html).

#### Mobile Platform and Touchscreen Support

The SAS Visual Analytics Apps run natively on iOS, Android, and Windows 10, and provide the ability to view and explore reports using a touchscreen.

Some SAS Viya user interfaces are not currently supported on mobile devices.


#### Database Drivers

Make sure that each client where users will access SAS software has the required database drivers already installed.

#### Screen Resolution

The minimum screen resolution for each client machine that will access the SAS Viya user interfaces is 1280 x 1024.
Pre-deployment Tasks

**Obtain the Required Files**

When you order SAS software, SAS sends a Software Order Email (SOE) to your business or organization that includes information about the software order. The SOE directs you to save its attached ZIP file and the license file to a directory on your local machine. If you have not already done so, you must save those files before performing any steps in this section.

In the same directory where you have saved the ZIP file, uncompress the file.

```
unzip SAS_Viya_deployment_data.zip
```

The following file structure is created in the directory where the ZIP file was uncompressed:

- `licenses/SASViyaV0300_order-number_Linux_x86-64.txt`
- `licenses/SASViyaV0300_order-number_site-number_Linux_x86-64.jwt`
- `ca-certificates/SAS_CA_Certificate.pem`
- `ca-certificates/SAS_CA_Certificate.p7b`
- `entitlement-certificates/entitlement_certificate.pem`
- `entitlement-certificates/entitlement_certificate.pfx`
Download the SAS Viya Image

Access the SAS Repository

The SAS Viya image is located in a repository that is hosted by SAS. To access the SAS repository, you use SAS Mirror Manager, which is a command-line utility that you run locally.

1. In the directory where you uncompressed the ZIP file from your SOE, download SAS Mirror Manager from the SAS Mirror Manager download site.

2. Uncompress the downloaded file.

3. List the name of the image that is in the SAS repository:

   ```bash
   mirrormgr list remote docker tags --deployment-data $(pwd)/SAS_Viya_deployment_data.zip --latest
   ```

   Here is the format of the output:

   ```
   sas-repository/docker-namespace/image-name:tag
   ```

   Here is typical output:

   ```
   ses.sas.download/va-104.0.0-x64_redhat_linux_7-docker/sas-viya-programming:3.4.7-20180702.1530552574483
   ```

   Note: The values from the preceding output are used in the examples throughout this guide.

Download the SAS Viya Image

1. To download the image from the SAS repository, configure Docker to use the certificates that were saved from the ZIP file from your SOE:

   ```bash
   Note: Enter each command on a single line.
   Run the commands as root or as a user with elevated privileges.
   ```

   ```bash
   mkdir /etc/docker/certs.d/ses.sas.download
cp -v $(pwd)/ca-certificates/SAS_CA_Certificate.pem /etc/docker/certs.d/ses.sas.download/ca.crt
cp -v $(pwd)/entitlement-certificates/entitlement_certificate.pem /etc/docker/certs.d/ses.sas.download/client.cert
cp -v $(pwd)/entitlement-certificates/entitlement_certificate.pem /etc/docker/certs.d/ses.sas.download/client.key
   ```

2. Pull and inspect the image:

   ```bash
   Note: Enter each command on a single line.
   ```

   ```bash
   docker pull ses.sas.download/va-104.0.0-x64_redhat_linux_7-docker/sas-viya-programming:3.4.7-20180702.1530552574483
docker inspect ses.sas.download/va-104.0.0-x64_redhat_linux_7-docker/sas-viya-programming:3.4.7-20180702.1530552574483
   ```

Tag and Push the Image to a Local Docker Registry

The SAS Viya image is a Docker image that can be tagged and stored in a local Docker registry. Pushing the image to a local Docker registry might save time and can protect against download limits because you can pull from a local copy instead of the remote SAS repository.
If you plan to deploy SAS Viya using Docker, using a local Docker registry and performing the following steps are optional.

If you plan to deploy SAS Viya using Kubernetes, using a local Docker registry and performing the following steps are required.

1. Tag the pulled image with the Docker registry's address:
   
   ```
   docker tag sas-repository/docker-namespace/image-name:tag \
   docker-registry-host-name:port/docker-namespace/image-name:tag
   ```

   Here is an example of the tag command:

   ```
   docker tag ses.sas.download/va-104.0.0-x64_redhat_linux_7-docker/sas-viya-programming:3.4.7-20180702.1530552574483 \
   localhost:5000/va-104.0.0-x64_redhat_linux_7-docker/sas-viya-programming:3.4.7-20180702.1530552574483
   ```

   Note: You can keep the same tag that is used for the image that was downloaded, or you can change it using the tag command. In the preceding example, the tag for the image in the local Docker registry is the same as the tag for the image that was downloaded from the SAS repository.

   Also, in the preceding example, you can run the command on two lines as shown.

2. Push the image to the local Docker registry:

   ```
   docker push docker-registry-host-name:port/docker-namespace/image-name:tag
   ```

   Here is an example of the push command:

   ```
   docker push localhost:5000/va-104.0.0-x64_redhat_linux_7-docker/sas-viya-programming:3.4.7-20180702.1530552574483
   ```

   After you push the image, users can pull the image and run it from the Docker registry.

   Note: In the example commands and file names throughout this guide, the Docker image in the SAS repository is used.

---

**Create the Required Directories**

1. In the same directory where you uncompressed the SAS_Viya_deployment_data.zip file, create the following directories:

   - `sasinside`
   - `sasdemo`
   - `cas`

2. Ensure that `_usermods` files are created in the `sasinside` directory.

   ```
   touch sasinside/autoexec_usermods.sas sasinside/cas_usermods.settings sasinside/init_usermods.properties sasinside/workspaceserver_usermods.sh
   ```

3. Copy the license file from the `licenses` directory to the `sasinside` directory and name the file `license.sas`.

   ```
   cp licenses/SASV* sasinside
   mv sasinside/SASViyaV0300_XXXXXX_Linux_x86-64.txt sasinside/license.sas
   ```

   Note: Over the lifetime of the deployment, the `sasinside` directory can contain the following files:

   - the SAS license
   - autoexec_usermods.sas
   - casconfig_usermods.lua
   - cas_usermods.settings
init_usermods.properties
sasv9_usermods.cfg
workspaceserver_usermods.sh
sasstudio_usermods.properties

Note: This file contains the SAS Studio configuration properties. For a container deployment of SAS Viya, the sasstudio_usermods.properties file is a symbolic link to the init_usermods.properties file in /opt/sas/viya/config/etc/sasstudio/default. In other SAS Viya documentation, such as SAS Viya Administration, the instructions for changing the SAS Studio configuration refer to the file as init_usermods.properties.

Note: Except for the license file, the files are symbolically linked to the files at /opt/sas/viya/config/etc/product/default when the container is started. Having the files in this directory provides a way to modify settings. For example, you can log off from SAS Studio, modify the files, and then log back in to SAS Studio to start a new SAS Workspace Server session or create a new CAS session with the new settings.

(Optional) Enable Host Authentication

By default, the SAS Viya image does not include support for host authentication, such as connecting to an LDAP server to authenticate users. To enable host authentication, you must add a software layer to the SAS Viya image that includes support for System Security Services Daemon (SSSD). Here is an example:

1. Create a directory named sssd_layering.
2. In the sssd_layering directory, create the following files.
   a. Create a file named host_auth.sh with the following content:

   ```bash
   #!/bin/bash -e

   sssd_pid_file="/var/run/sssd.pid"
   if [[ -f $sssd_pid_file ]]; then
     echo "[INFO] sssd already running."
     sssd_pid=$(cat $sssd_pid_file)
     rm -f $sssd_pid
   else
     /usr/sbin/sssd --config /etc/sssd/sssd.conf
     set +e;/usr/sbin/sssd -g --config /etc/sssd/sssd.conf;set -e
     /usr/sbin/sssd --config /etc/sssd/sssd.conf
   fi
   ```

   Note: After saving the file, run chmod +x to make sure that the file is executable. Here is an example:

   ```bash
   chmod +x host_auth.sh
   ```

   b. Create a Dockerfile:
      i. Copy the content from the auth_sssd Dockerfile that is located in the SAS Viya Container Recipes GitHub Project.

         ```bash
         ```

         ii. Comment out the two ARG lines.
             ```bash
             sed -i 's|ARG BASE|ARG BASE|g' Dockerfile
             ```

         iii. Change the FROM line to reference the SAS Viya image. Here is an example:
             ```bash
             sed -i 's|FROM.*|FROM ses.sas.download/va-104.0.0-x64_redhat_linux_7-docker/sas-viya-programming:3.4.7-20180702.1530552574483|' Dockerfile
             ```
Note: The example indicates that the SAS Viya image is accessed from the remote SAS repository. If you tagged and pushed the image to a local Docker registry, be sure to reference the Docker registry host name and tag.

iv  Change the last COPY line to reference the host_auth.sh file that you created in Step 2a on page 16. Here is an example:

   sed -i 's|^COPY sssd_pre_deploy.sh.*$|COPY host_auth.sh /tmp/host_auth.sh|' Dockerfile

c  Create a file named sssd_mkhomedir_helper.sh with the following content:

   #!/bin/bash
   [ -z "$PAM_USER" ] && exit 0
   /sbin/mkhomedir_helper $PAM_USER

   Note: After saving the file, run chmod +x to make sure that the file is executable. Here is an example:

   chmod +x sssd_mkhomedir_helper.sh

d  Create an sssd.conf file for your organization’s configuration. For more information about the sssd.conf file, contact your system administrator.

e  (Optional) If you plan to use security certificates, determine whether you need an sssd.cert file. If you do, perform the following steps:

   i  Move the sssd.cert file to the same location as the sssd.conf file, and make sure that the certificate file is named sssd.cert.

   ii  Make sure that the sssd.conf file is configured to point to /etc/ssl/certs/sssd.

   If you do not plan to use security certificates, remove the sssd information from the Dockerfile that you created in Step 2 on page 16.

3  From the sssd_layering directory, build the new image and save the image to a Docker repository.

   docker build -t docker-registry-host-name:port/docker-namespace/new-image-name:tag .

   Here is an example:

   docker build -t localhost:5000/va-104.0.0-x64_redhat_linux_7-docker/company-sas-server:latest .

   Because the image has been renamed in the last step, you should replace all instances of sas-viya-programming in the commands throughout the rest of this guide with company-sas-server:latest.

---

(Optional) Lock Down SAS Workspace Servers

Using a whitelist file and the LOCKDOWN option, you can limit access to files in a SAS Workspace Server session when code is submitted in batch mode.

Note: For an example of using a whitelist, see Example 2: Hiding the Whitelist By Locating the Path outside the Whitelist in SAS Viya Administration: Programming Run-Time Servers.

To configure the lockdown feature before you start the SAS container:

1  Create a whitelist file that includes the paths that should be available to any SAS Workspace Server.

   Note: A path in the whitelist does not mean that any user can read any file in that path. Host permissions on physical files and directories always take precedence over the whitelist. By default, SAS adds certain predefined paths from the SAS configuration file.
2 In the `sasinside/workspaceserver_usermods.sh` file, set the `WORKSPACESERVER_LOCKDOWN_ENABLE` variable to 1. Setting this variable enables the LOCKDOWN option.

3 If your site uses SAS Studio, in the `sasinside/init_usermods.properties` file, set the `webdms.showSystemRoot` variable to `false`.

---

**Configure External Access for Kubernetes Services**

If you plan to deploy your software using Kubernetes, you must establish a method for external access. There are two methods of external access: using an Ingress or creating a NodePort service. Regardless of the method that you choose, external access must be configured before you deploy your software.

For information about setting up an Ingress, see Ingress, specifically the Ingress Controllers section.

For information about preparing for a NodePort service, see Type NodePort.
Deploying SAS Viya

Deploy with Docker .......................................................... 19

Deploy with Kubernetes ....................................................... 20
  Set Up External Access .................................................. 20
  Deploy the Software ..................................................... 22

Differentiate between Multiple SAS Environments .................. 25
  Docker ........................................................................... 25
  Kubernetes .................................................................... 25

Deploy with Docker

1 In the same directory where you uncompressed the SAS_Viya_deployment_data.zip file, create a shell script named launchsas.sh with the following content for creating and running the container.

   Note: If you added host authentication using the steps in "(Optional) Enable Host Authentication" on page 16, change the instance of sas-viya-programming in the script to company-sas-server:latest.

   #! /bin/bash -e

   IMAGE=ses.sas.download/va-104.0.0-x64_redhat_linux_7-docker/sas-viya-programming:tag
   SAS_HTTP_PORT=8081

   run_args="
   --name=sas-programming
   --rm
   --hostname sas-programming
   --env RUN_MODE=developer
   --env CASENV_ADMIN_USER=sasdemo
   --env CASENV_CAS_VIRTUAL_HOST=$hostname -f
   --env CASENV_CAS_VIRTUAL_PORT=${SAS_HTTP_PORT}
   --env CASENV_CASDATADIR=/cas/data
   --env CASENV_CASPERMSTORE=/cas/permstore
   --publish-all
   --publish 5570:5570
   --publish ${SAS_HTTP_PORT}:80
   --volume ${PWD}/sasinside:/sasinside
   --volume ${PWD}/sasdemo:/data
   --volume ${PWD}/cas/data:/data
   --volume ${PWD}/cas/cache:/cache
   --volume ${PWD}/cas/permstore:/cas/permstore"

   # Run in detached mode
```bash
docker run --detach ${run_args} $IMAGE "$@

# For debugging startup, comment out the detached mode command and uncomment the following

#docker run --interactive --tty ${run_args} $IMAGE "$@

The *tag* is a descriptive name used to identify this deployment of SAS Viya. If you are using host authentication, the tag used in the Dockerfile should be the tag that is used here.

For more information about the `CASENV_ADMIN_USER` variable, see "CAS Administrator Account Requirements" on page 10.

The lines beginning with `--volume` describes the locations where data will be persisted.

2 From the directory where the `launchsas.sh` script is located, run the script to launch the image.
   ```bash
   ./launchsas.sh
   ```

3 Retrieve the `sasdemo` password from the logs.
   ```bash
   docker logs sas-programming 2>&1 | grep Password=
   ```

---

**Deploy with Kubernetes**

**Set Up External Access**

If you are deploying with Kubernetes, you must establish a method for external access. You can either use an Ingress or set up a NodePort service. For more information, see "Configure External Access for Kubernetes Services" on page 18.

**Create an Ingress**

1 Create a file named `sas-analytics-ingress.yml`.

2 Add the following content to the `sas-analytics-ingress.yml` file.

   **Note:** Replace every instance of `K8SUSER` with a unique value for the Kubernetes user.

   ```yaml
   apiVersion: extensions/v1beta1
   kind: Ingress
   metadata:
     name: user-K8SUSER-ing-sasprogramming
     namespace: user-K8SUSER
   spec:
     rules:
     - host: K8SUSER-sas-programming.company.com
       http:
         paths:
         - backend:
             serviceName: sas-programming
             servicePort: 80
     - host: K8SUSER-sas-cas.company.com
       http:
         paths:
         - backend:
             serviceName: sas-programming
             servicePort: 5570
   ```
3 Create the Ingress.

kubectl apply -f sas-analytics-ingress.yml

4 Confirm the creation of the Ingress.

Note: Replace $K8SUSER$ with the Kubernetes user you used in step 2.

kubectl describe ing user-$K8SUSER$-ing-sasprogramming

The output should look like the following. $K8SUSER$ will be replaced by the Kubernetes user you used in step 2.

<table>
<thead>
<tr>
<th>Namespace:</th>
<th>user-$K8SUSER$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address:</td>
<td></td>
</tr>
<tr>
<td>Default backend:</td>
<td>default-http-backend:80 (&lt;none&gt;)</td>
</tr>
<tr>
<td>Rules:</td>
<td></td>
</tr>
<tr>
<td>Host</td>
<td>Path</td>
</tr>
<tr>
<td>K$8SUSER$-sas-programming.company.com</td>
<td>sas-programming:80 (&lt;none&gt;)</td>
</tr>
<tr>
<td>K$8SUSER$-sas-cas.company.com</td>
<td>sas-programming:5570 (&lt;none&gt;)</td>
</tr>
</tbody>
</table>

Create a NodePort Service

1 Create a file named sas-analytics-nodeport.yml.

2 Add the following contents to the sas-analytics-nodeport.yml file.

```yaml
apiVersion: v1
kind: Service
metadata:
  name: sas-programming
spec:
  selector:
    app: sas-programming
  ports:
  - name: http
    port: 80
    protocol: TCP
    targetPort: 80
  - name: cas
    port: 5570
    protocol: TCP
    targetPort: 5570
  type: NodePort
```

3 Create the NodePort service.

kubectl apply -f sas-analytics-nodeport.yml

4 Confirm the NodePort service is working.

kubectl describe svc sas-programming

The output will look like the following.

<table>
<thead>
<tr>
<th>Namespace:</th>
<th>user-jogate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Labels:</td>
<td>&lt;none&gt;</td>
</tr>
<tr>
<td>Annotations:</td>
<td>&lt;none&gt;</td>
</tr>
<tr>
<td>Selector:</td>
<td>app=sas-programming</td>
</tr>
<tr>
<td>Type:</td>
<td>NodePort</td>
</tr>
<tr>
<td>IP:</td>
<td>10.254.50.19</td>
</tr>
<tr>
<td>Port:</td>
<td>http 80/TCP</td>
</tr>
</tbody>
</table>
Use the port that is mapped to port 80 as the value for the CASENV_CAS_VIRTUAL_PORT in “Deploy the Software” on page 22. In the example above, that port would be 30962.

**Deploy the Software**

1. Create a file named sas-analytics.yml.

2. Add the following content to the sas-analytics.yml file.

```yaml
---
apiVersion: v1
kind: ConfigMap
metadata:
  name: poac-config
data:
  poac.RUN_MODE: "developer"
  poac.DEPLOYMENT_NAME: "K8SUSER"
  poac.CASKEY: "unique-key"
  poac.SAS_DEBUG: "0"
  poac.SETINIT_TEXT: |
    contents-of-license-file
  poac.PRE_DEPLOY_SCRIPT: ""
  poac.POST_DEPLOY_SCRIPT: ""
  poac.CASENV_CAS_VIRTUAL_HOST: "see-vitual-host-note"
  poac.CASENV_CAS_VIRTUAL_PORT: "see-virtual-port-note"
---
apiVersion: v1
kind: Service
metadata:
  name: sas-programming
spec:
  selector:
    app: sas-programming
  ports:
  - name: http
    port: 80
    protocol: TCP
    targetPort: 80
  - name: cas
    port: 5570
    protocol: TCP
    targetPort: 5570
  sessionAffinity: None
  clusterIP: None
---
```
apiVersion: apps/v1beta1
type: Deployment
metadata:
  name: sas-programming
spec:
  replicas: 1
strategy:
  rollingUpdate:
    maxSurge: 1
    maxUnavailable: 1
type: RollingUpdate
template:
  metadata:
    labels:
      app: sas-programming
      hostname: sas-programming
spec:
  containers:
  - name: sas-programming
    image: location-of-docker-image:tag
    imagePullPolicy: Always
    ports:
    - containerPort: 5570
    - containerPort: 80
    env:
    - name: SERVICE_NAME
      value: cascontroller
    - name: CASCFG_MODE
      value: "smp"
    - name: SAS_DEBUG
      valueFrom:
        configMapKeyRef:
          name: poac-config
          key: poac.SAS_DEBUG
    - name: RUN_MODE
      valueFrom:
        configMapKeyRef:
          name: poac-config
          key: poac.RUN_MODE
    - name: DEPLOYMENT_NAME
      valueFrom:
        configMapKeyRef:
          name: poac-config
          key: poac.DEPLOYMENT_NAME
    - name: CASKEY
      valueFrom:
        configMapKeyRef:
          name: poac-config
          key: poac.CASKEY
    - name: SETINIT_TEXT
      valueFrom:
        configMapKeyRef:
          name: poac-config
          key: poac.SETINIT_TEXT
    - name: PRE_DEPLOY_SCRIPT
      valueFrom:
Consider the following issues while adding the content:

- When you insert the license information, ensure that you maintain the indentation provided in this sample.
- For **poac.CASENV_CAS_VIRTUAL_HOST**, use the host value of the Ingress you’ve created, such as `K8SUSER-sas-programming.company.com`. If you are not using an Ingress, use the IP address or DNS of a Kubernetes worker node.
- For **poac.CASENV_CAS_VIRTUAL_PORT**, use the port the Ingress is running on. If you are not using an Ingress, use the port that port 80 is mapped to for the NodePort service.
- Replace the text of `location-of-docker-image:tag` with the local Docker registry where the Docker image was stored. See “Tag and Push the Image to a Local Docker Registry” on page 14 for more information about the local Docker registry.
For more information about the CASENV_ADMIN_USER variable, see "CAS Administrator Account Requirements" on page 10.

Mounting the sasinside directory is for supporting configuration of the SAS/ACCESS engines associated with your order.

3 Deploy the image.

   kubectl apply -f sas-analytics.yml

4 Retrieve the sasdemo password from the logs.

   kubectl logs sas-analytics -f | grep "Password="

---

Differentiate between Multiple SAS Environments

To avoid the confusion of deploying multiple SAS environments to a single Docker or Kubernetes environment, create a new directory for each additional environment. In each new directory, create the sasinside, sasdemo, and cas directories as described in "Create the Required Directories" on page 15.

Docker

For Docker environments, edit the launchsas.sh script to run each new instance.

   IMAGE=ses.sas.download/va-104.0.0-x64_redhat_linux_7-docker/sas-viya-programming:tag
   SAS_HTTP_PORT=new-port-value
   SAS_CAS_PORT=new-port-value
   run_args="
   --name=sas-unique-value
   --rm
   --hostname sas-unique-value
   --env RUN_MODE=developer
   --env CASENV_ADMIN_USER=sasdemo
   --env CASENV_CAS_VIRTUAL_HOST=$(hostname -f)
   --env CASENV_CAS_VIRTUAL_PORT=${SAS_HTTPD_PORT}
   --env CASENV_CASDATADIR=/cas/data
   --env CASENV_CASPERMSTORE=/cas/permstore
   --publish-all
   --publish ${SAS_CAS_PORT}:5570
   --publish ${SAS_HTTP_PORT}:80
   --volume ${PWD}/sasinside:/sasinside
   --volume ${PWD}/sasdemo:/data
   --volume ${PWD}/cas/data:/cas/data
   --volume ${PWD}/cas/cache:/cas/cache
   --volume ${PWD}/cas/permstore:/cas/permstore"

Kubernetes

For Kubernetes environments, assign a unique name to the SAS invocation in the Service and Deployment section in the sas-analytics.yml file:

   ---
   apiVersion: v1
kind: Service
metadata:
  name: sas-unique-value
spec:
  selector:
    app: sas-unique-value
...
--
apiVersion: apps/v1beta1
dependent:
  kind: Deployment
metadata:
  name: sas-unique-value
spec:
  replicas: 1
  template:
    metadata:
      labels:
        app: sas-unique-value
        hostname: sas-unique-value
...
Post-deployment Tasks

- Configure SAS/ACCESS Interface to Hadoop .................................................. 27
- Configure ODBC ................................................................. 28
- Configure SAS/ACCESS Interface to PostgreSQL ..................................... 29

Configure SAS/ACCESS Interface to Hadoop

Note: The information in this section is applicable only if you ordered SAS/ACCESS Interface to Hadoop (on SAS Viya).

The collected configuration and JAR files should be stored under the `sasinside/hadoop` directory in the `sasinside/hadoop/config` and `sasinside/hadoop/jar` directories, respectively.

If the environment needs to be configured further, to manually configure the variables:

1. On your machine, use a text editor to edit the `workspaceserver_usermods.sh` file:
   ```bash
   sudo vi sasinside/workspaceserver_usermods.sh
   ```

2. `JAVA_HOME` will default automatically to the Java version that is in the container. If you installed your own version of Java, add the following lines:
   ```bash
   export JAVA_HOME=location-of-your-Java-8-JRE
   export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$JAVA_HOME/lib/amd64/server
   ```

3. If you are using MapR, add the following line:
   ```bash
   export MAPR_HOME=/opt/mapr
   ```

4. Save and close the `workspaceserver_usermods.sh` file.

5. On your machine, use a text editor to edit the `cas_usermods.settings` file:
   ```bash
   sudo vi sasinside/cas_usermods.settings
   ```

6. `JAVA_HOME` will default automatically to the Java version that is in the container. If you installed your own version of Java, add the following lines:
   ```bash
   export JAVA_HOME=location-of-your-Java-8-JRE
   export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$JAVA_HOME/lib/amd64/server
   ```

7. If you are using MapR, add the following line:
   ```bash
   export MAPR_HOME=/opt/mapr
   ```

8. Save and close the `cas_usermods.settings` file.
Configure ODBC

Note: This information is applicable only if you ordered SAS/ACCESS Interface to ODBC (on SAS Viya).

1 In the same directory where you uncompressed the SAS_Viya_deployment_data.zip file and created the sasinside directory, create the sasinside/odbc directory:

2 On your machine, use a text editor to create and edit the sasinside/odbc/odbc.ini file in order to configure data sources.

Some vendors of ODBC drivers might provide support for system administrators to maintain a centralized copy of the odbc.ini file via the environment variable ODBCINI. Refer to your ODBC driver’s vendor documentation for more specific information.

Assign the location of the shared libraries to one of the system environment variables in order to enable the ODBC drivers to be loaded dynamically at run time. The ODBC drivers are ODBC API-compliant shared libraries, which are referred to as shared objects in UNIX.

3 On your machine, use a text editor to edit the sasinside/workspaceserver_usermods.sh file:

   sudo vi sasinside/workspaceserver_usermods.sh

4 Add the following lines, depending on the version of ODBC that you are using.

   For DataDirect:

   export ODBCHOME=/sasinside/odbc
   export ODBCINI=/sasinside/odbc/odbc.ini
   export ODBCINST=/sasinside/odbc/odbcinst.ini
   export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$ODBCHOME/lib

   For iODBC:

   export ODBCINI=/sasinside/odbc/odbc.ini
   export ODBCINST=/sasinside/odbc/odbcinst.ini
   export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/sasinside/odbc/lib

   For unixODBC:

   export ODBCSYSINI=/sasinside/odbc
   export ODBCINI=${ODBCSYSINI}/odbc.ini
   export ODBCINST=${ODBCSYSINI}/odbcinst.ini
   export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:${ODBCSYSINI}/lib

   Note: For unixODBC, if ODBCSYSINI is not set in your environment, then ODBCINI and ODBCINSTINI should be specified as full paths to the respective files, including the file names.

5 Save and close the /sasinside/workspaceserver_usermods.sh file.

6 On your machine, use a text editor to edit the sasinside/cas_usermods.settings file:

   sudo vi sasinside/cas_usermods.settings

7 Add the following lines, depending on the version of ODBC that you are using.

   For DataDirect:

   export ODBCHOME=/sasinside/odbc
   export ODBCINST=/sasinside/odbc/odbc.ini-file-name
   export ODBCINST=/sasinside/odbc/odbcinst.ini-file-name
   export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$ODBCHOME/lib
For iODBC:

```bash
export ODBCINI=/sasinside/odbc/odbc.ini-file-name
export ODBCINSTINI=/sasinside/odbc/odbcinst.ini-file-name
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/sasinside/odbc/lib
```

For unixODBC:

```bash
export ODBCSYSINI=/sasinside/odbc
export ODBCINI=${ODBCSYSINI}/odbc.ini
export ODBCINSTINI=${ODBCSYSINI}/odbcinst.ini
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:${ODBCSYSINI}/lib
```

**Note:** For unixODBC, if ODBCSYSINI is not set in your environment, then ODBCINI and ODBCINSTINI should be specified as full paths to the respective files, including the file names.

8. Save and close the `sasinside/cas_usermods.settings` file.

---

**Configure SAS/ACCESS Interface to PostgreSQL**

**Note:** This information is applicable only if you ordered SAS /ACCESS to PostgreSQL (on SAS Viya).

The SAS/ACCESS Interface to PostgreSQL is used by the workspace server and the SAS Cloud Analytics Services.

Create a file with information about the database connection:

1. In the same directory where you uncompressed the `SAS_Viya_deployment_data.zip` file and created the `sasinside` directory, create the `sasinside/odbc` directory.

2. Choose between the following two options for providing connection information to enable SAS/ACCESS Interface to PostgreSQL.
   - **Reference a Data Source Name (DSN).**
     
     On your machine, use a text editor to create and edit the `sasinside/odbc/odbc.ini` file. Here is an example of an `odbc.ini` file that supports DSN:
     ```ini
     [postgresql_data_source_name]
     Driver=/opt/sas/viya/home/lib64/psqlodbcw.so
     ServerName=host-name-where-docker-is-running
     username=user-name
     password=password
     database=database
     port=5432
     ```
   - **Specify connection information in your code.**
     
     On the host system, use a text editor to create and edit the `sasinside/odbc/odbcinst.ini` file. Here is an example:
     ```ini
     [ODBC Drivers]
     PostgreSQL=Installed
     [PostgreSQL]
     Description=ODBC for PostgreSQL
     Driver=/opt/sas/viya/home/lib64/psqlodbcw.so
     ```

Configure SAS and CAS to use the database connection information.

1. On your machine, use a text editor to edit the `sasinside/workspaceserver_usermods.sh` file:
2 Add the following lines:
   
   export ODBCINI=/sasinside/odbc/odbc.ini
   export ODBCINST=/sasinside/odbc/odbcinst.ini
   export PGCLIENTENCODING=encoding-for-the-PostgreSQL-client
   export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:path-to-PostgreSQL-client
   
   For bulk loading, add the following line:
   
   export PATH=$PATH:path-to-PostgreSQL-bulk-loading

3 Save and close the `sasinside/workspaceserver_usermods.sh` file.

4 On the host system, use a text editor to edit the `sasinside/cas_usermods.settings` file:
   
   sudo vi sasinside/cas_usermods.settings

5 Add the following lines:
   
   export ODBCINI=/sasinside/odbc/odbc.ini
   export ODBCINST=/sasinside/odbc/odbcinst.ini
   export PGCLIENTENCODING=encoding-for-the-PostgreSQL-client
   export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:path-to-PostgreSQL-client

6 Save and close the `sasinside/cas_usermods.settings` file.
Validating the Deployment

Validate a Docker Deployment

To verify that the container is running, run the following command:

```
docker ps --filter name=sas-programming --format "table {{.ID}}	{{.Names}}	{{.Image}}	{{.Status}}	{{.Size}}"*
```

**Note:** Enter the command on a single line.

Here is example output:

<table>
<thead>
<tr>
<th>CONTAINER ID</th>
<th>NAMES</th>
</tr>
</thead>
<tbody>
<tr>
<td>9436fbaef7d7</td>
<td>sas-programming</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IMAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ses.sas.download/va-104.0.0-x64_redhat_linux_7-docker/sas-viya-programming:3.4.7-20180702.1530552574483</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>STATUS</th>
<th>SIZE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Up 2 hours</td>
<td>3.54MB (virtual 9.14GB)</td>
</tr>
</tbody>
</table>

If the container is not running, perform the following steps:

1. In the launchsas.sh script (see "Deploy with Docker" on page 19), comment out the first command by adding a number sign (#) at the beginning of the line. Uncomment the second command by removing the number sign.

   ```
   # Run in detached mode
   docker run --detach ${run_args} $IMAGE "$@
   # For debugging startup, comment out the detached mode command and uncomment the following
   docker run --interactive --tty ${run_args} $IMAGE "$@
   ```

2. (Optional) In the launchsas script, add a new line to enable debugging.

   ```
   ... --env CASENV_CASDATADIR=/cas/data --env CASENV_CASPERMSTORE=/cas/permstore --env SAS_DEBUG=1 --publish-all
   ```
3 Run the script in interactive mode to start the container.
   ./launchsas.sh

4 The software is deployed with enhanced messaging. You might be able to find the error. If you cannot find
   the error, the output from this command will be useful if you contact SAS Technical Support.

---

**Validate a Kubernetes Deployment**

To verify that the container is running, run the following command:

```
kubectl get -f sas-programming.yml
```

Here is example output:

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>CLUSTER-IP</th>
<th>EXTERNAL-IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>sas-programming</td>
<td>NodePort</td>
<td>10.254.224.149</td>
<td>&lt;none&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PORT(S)</th>
<th>AGE</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>NAME</th>
<th>DESIRED</th>
<th>CURRENT</th>
<th>UP-TO-DATE</th>
<th>AVAILABLE</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>sas-programming</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>11d</td>
</tr>
</tbody>
</table>

If you do not get the results that you expect, run the following command to enable the DEBUG option to provide
more information:

```
kubectl get configmap poac-config -o yaml | \sed 's|poac.SAS_DEBUG: "0"|poac.SAS_DEBUG: "1"|' | \kubectl replace -f -
```

**Note:** Enter the command on a single line.

After you run the command, delete the sas-viya-programming pod. Removing the pod forces the creation of a
new sas-viya-programming pod. The new pod will provide additional information about the pod start, which can
be reviewed by using the kubectl logs command.

---

**Access CAS Server Monitor**

If you deployed SAS Viya using Docker, perform the following steps to access CAS Server Monitor.

1 Type one of the following URLs in a web browser:
   - If you deployed SAS Viya using Docker:
     http://host-name-where-docker-is-running:sas_http_port/cas-shared-default-http
     **Note:** The sas_http_port value is defined in the launchsas.sh script that starts the container.
   - If you deployed SAS Viya using Kubernetes:
     http://K8SUSER-sas-programming.company.com/cas-shared-default-http

2 Log on using the sasdemo account or the alternative administrator that you designated in the installation
   script. For more information, “CAS Administrator Account Requirements” on page 10.
Log On to Your Version Of SAS Studio

To ensure that your default version of SAS Studio has been deployed correctly and is working, log on to it:

1. Type one of the following URLs in a web browser:
   - If you deployed SAS Viya using Docker:
     http://host-name-where-docker-is-running:sas_http_port/SASStudio
     Note: The sas_http_port value is defined in the launchsas.sh script that starts the container.
   - If you deployed SAS Viya using Kubernetes:
     http://K8SUSER-sas-programming.company.com/SASStudio

2. Log on using the credentials for your operating system account.

Note: To log off from SAS Studio, click **Sign Out** on the toolbar. Do not use the **Back** button on your web browser.
Completing the Deployment

Refer to Additional Documentation

After you validate the deployment, you can perform administrative tasks.

- The following tasks, which are specific to running SAS Viya for Containers in Docker or Kubernetes environments, are provided in this guide. For more information, see “Managing Your Software” on page 37.
  - Add to the SAS Viya image
  - Back up and recover
  - Configure CAS
  - Apply a new license
  - Access logs

- For information about other administrative tasks, see SAS Viya Administration: Initial Tasks.

  Note: The SAS Viya for Containers image is a programming-only deployment, which does not include SAS Environment Manager. Therefore, topics about SAS Environment Manager that are covered in SAS Viya Administration are not supported.

For usage information, refer to the Help that is available from the SAS Viya product and administrative interfaces.

You can also refer to the appendixes in this guide for additional tasks that you might perform based on your environment. For example, the appendixes include information to help you configure your Hadoop infrastructure.
Managing Your Software

Update Your Software

Overview
The update process brings your deployed software up-to-date with the latest compatible software. You will perform the update with the same commands that were used to install SAS Viya, and you will use the same software order.

You will need the location of the directory where you stored deployment and maintenance files. For more information about this directory, see “Obtain the Required Files” on page 13.

Update Your Software

Update with Docker
Updating the SAS Viya environment requires an update to the Docker image and a restart of the image. To update a deployment using Docker:

1 Run the following command to find the latest Docker tag.
Note: Multiple lines are used to improve readability. However, in your environment, make sure that you enter the command on a single line.

```
mirrormgr list remote docker tags --deployment-data
$(pwd)/SAS_Viya_deployment_data.zip --latest
```

Here is an example of the output:

```
ses.sas.download/va-104.0.0-x64_redhat_linux_7-docker/sas-viya-programming:3.4.7-20180702.1530552574483
```

2 Obtain the latest image.

```
docker pull output-from-step-1
```

3 In the launchsas.sh file, edit the tag portion of the value for IMAGE to use the tag of the latest image.

From the example output in step 1, here is the tag portion:

```
3.4.7-20180702.1530552574483
```

See “Deploy with Docker” on page 19.

4 To determine whether the container is running, run the following command:

```
docker ps
--filter name=sas-programming
--format "table {{.ID}}	{{.Names}}	{{.Image}}	{{.Status}}	{{.Size}}"
```

Here is an example of the output:

```
CONTAINER ID        IMAGE               COMMAND             CREATED ...
```

In the example, the table is empty. If the table is empty, skip to step 6.

5 Stop the container.

```
docker stop sas-programming
```

6 Remove the container.

```
docker rm sas-programming
```

7 Run launchsas.sh to start the new image.

```
./launchsas.sh
```

8 If there are problems, edit ./launchsas.sh to set the tag to the last working value, and then perform steps 5 through 7 again.

**Update with Kubernetes**

To update a deployment using Kubernetes:

1 Run the following command to find the latest Docker tag.

```
mirrormgr list remote docker tags --deployment-data
$(pwd)/SAS_Viya_deployment_data.zip --latest
```

Here is typical output:

```
ses.sas.download/va-104.0.0-x64_redhat_linux_7-docker/sas-viya-programming:3.4.7-20180702.1530552574483
```

In the example output, the tag is the portion after `sas-viya-programming`:

```
3.4.7-20180702.1530552574483
```

2 Update the image.
Add to the SAS Viya Image

The SAS Analytics image can be used as the basis for a customized Docker image that meets the needs of the user. Users could add Jupyter Notebook or R Studio to the image in order to extend the functionality of the SAS Analytics image.

Here is an example of how to build a customized Docker image, using Jupyter Notebook.

1 Create a directory named `jupyter_layering`.

2 In `jupyter_layering`, create the following files.
   a Create a file named Dockerfile with the following content:

   ```bash
   FROM ses.sas.download/va-104.0.0-x64_redhat_linux_7-docker/sas-sas-server:tag
   RUN yum install --assumeyes --noplugins epel-release && 
       yum clean all && 
       rm -rf /tmp/rpms && 
       rm -rf /root/.cache /var/cache/yum
   RUN yum install --assumeyes --noplugins gcc gcc-c++ openssl python34-devel python34-pip && 
       yum clean all && rm -rf /tmp/rpms && 
       rm -rf /root/.cache /var/cache/yum
   RUN pip3 --no-cache-dir install notebook saspy sas_kernel \
       https://github.com/sassoftware/python-swat/releases/download\/
       /v1.4.0/python-swat-1.4.0-linux64.tar.gz && \
       jupyter nbextension install --py sas_kernel.showSASLog && \
       jupyter nbextension enable sas_kernel.showSASLog --py && \
       jupyter nbextension install --py sas_kernel.theme && \
       jupyter nbextension enable sas_kernel.theme --py
   # Uncomment the following and add '&& ' to the previous line to disable the Python kernel
   #pip3 uninstall -y terminado && 
   #jupyter kernelspec remove python3 -f
   COPY post_deploy.sh /tmp/post_deploy.sh
   RUN chmod +x /tmp/post_deploy.sh
   COPY jupyter_proxy.conf /etc/httpd/conf.d/jupyter_proxy.conf
   
   b Create a file named post_deploy.sh with the following content:
#!/bin/bash

# Run Jupyter notebook

export SASSERVICENAME="jupyter"
export JPY_COOKIE_SECRET=`openssl rand -hex 32`
export SSLCALISTLOC="${SASHOME}/SASSecurityCertificateFramework/cacerts/trustedcerts.pem"
export CAS_CLIENT_SSL_CA_LIST="/data/casconfig/sascas.pem"

# create it here also (instead of just in sasdemo user creation)
runuser --shell "/bin/sh" --login ${RUN_USER} \ 
  --command "mkdir -p ~/jupyter"

cp /usr/lib/python3.4/site-packages/saspy/sascfg.py\ 
/usr/lib/python3.4/site-packages/saspy/sascfg_personal.py
& & sed -i -e "s#opt/sasinside/SASHome/SASFoundation\ 
9.4/bin/sas_u8#opt/spre/home/SASFoundation/sas#g"\ 
/usr/lib/python3.4/site-packages/saspy/sascfg_personal.py

echo "Starting ${SASSERVICENAME}..."

#docker_create_pid_file

_jupyterpid=${DOCKERPIDFILE}

# create jupyter config file
runuser --shell "/bin/sh" --login ${RUN_USER} \ 
  --command "jupyter notebook --generate-config"

runuser --shell "/bin/sh" --login ${RUN_USER} \ 
  --command "JPY_COOKIE_SECRET=${JPY_COOKIE_SECRET} \ 
  SSLCALISTLOC=${SSLCALISTLOC} \ 
  CAS_CLIENT_SSL_CA_LIST=${CAS_CLIENT_SSL_CA_LIST} \ 
  jupyter notebook \ 
  --ip='*' \ 
  --no-browser \ 
  --NotebookApp.token='${JUPYTER_TOKEN}' \ 
  --NotebookApp.terminals_enabled=False \ 
  --NotebookApp.base_url=/jupyter \ 
  --KernelSpecManager.ensure_native_kernel=False \ 
  --notebook-dir=~/jupyter &"

# pgrep jupyter > ${_jupyterpid}

Create a file named jupyter_proxy.conf with the following content:

<Location /Jupyter>
ProxyPass        http://localhost:8888/Jupyter
ProxyPassReverse http://localhost:8888/Jupyter
RequestHeader set Origin "http://localhost:8888"
</Location>

<Location /Jupyter/api/kernels>
ProxyPass        ws://localhost:8888/Jupyter/api/kernels/
ProxyPassReverse ws://localhost:8888/Jupyter/api/kernels/
</Location>
3 From the `jupyter_layering` directory, build the new image, and save the image to a local Docker repository.

```bash
docker build -t docker-repository/company-sas-server:latest .
```

Since the image has been renamed in the last step, you should replace all instances of `sas-viya-programming` in the commands throughout the rest of this guide with `company-sas-server:latest`.

---

**Backup and Recovery**

**Guidelines**

This topic provides information about backup and recovery tasks that are specific for running SAS Viya for Containers. Consider the following guidelines:

- Persistent storage attached to the containers can ensure that data sources that are not handled by the backup utilities can still be retained and recovered.
- Assets that retain information or the state between CAS sessions or SAS sessions (for example, CAS controls and caslib information) should be backed up on persistent disks. User home directories must also be NFS mounts and capable of being backed up through your organization’s standard persistent disk backup procedures.

**Kubernetes: Retrieve a Copy of the Manifest**

For backup and archival purposes, retrieve a copy of the running deployment manifest from the Kubernetes environment.

1 Make sure that `kubectl` is pointed to the correct environment.

2 Run the following command to get a list of the deployed pods:

```bash
cubectl get pods
```

Look for a pod that begins with `sas-programming`.

3 Run the following command:

```bash
cubectl get pod -o yaml name-of-pod > local-file-name
```

4 Copy the manifest file to an archive location for backup purposes.
CAS Configuration

Before You Begin

- This topic includes CAS configuration tasks that are specific for running SAS Viya for Containers in Docker or Kubernetes environments.
- For other CAS configuration tasks, see SAS Cloud Analytic Services in SAS Viya Administration: SAS Cloud Analytic Services.

About Caslib Management Privileges

Caslib management privileges are stored in a directory known as the permstore. For SAS Viya for Containers, the default permstore should be stored on persistent storage and attached to the container so that the permstore persists if the container is redeployed.

Specify CAS Configuration Options

For SAS Viya for Containers, CAS configuration options are specified as environment variables or are recorded in a file on disk that is mounted to the container.

Note: In the following examples, the CAS configuration option MODE is set. Because SAS Viya on Containers supports the CAS server for symmetric multi-processing (SMP) only, the MODE setting would always be SMP. The MODE setting is used here as an example of how a CAS configuration can be set.

To specify a CAS configuration option as an environment variable:

- Prefix the CAS configuration option with CASCFG_ and capitalize the option name. In the following examples, the MODE option is prefixed with CASCFG_ and is set to smp.
- For Docker environments, use the Docker run command to set the CAS configuration option. Here is an example:

  ```bash
  docker run \
  --detach \
  --rm \
  --env CASCFG_MODE=smp \
  --hostname sas.viya.programming \
  --name sas-viya-programming \
  sas-viya-programming
  ```

- For Kubernetes environments, an addition is made to the manifest file (sas-analytics.yml). Here is an example:

  ```yaml
  - name: CASCFG_MODE
    value: "smp"
  ```

To specify the CAS configuration options in a file that is mounted to the container:

1 In the directory where you unpacked the SAS_Viya_deployment_data.zip file, run Docker to create a directory named sasinside, and then create a file in that directory named casconfig_usermods.lua.
2 Edit the casconfig_usermods.lua file to specify the CAS configuration options, such as cas.mode='smp', and then save the file.
3 Perform one of the following steps:
For Docker environments, run the container to link the location of the casconfig_usermods.lua file to where the software expects it.

```bash
docker run \
  --detach \
  --rm \ 
  --volume ${PWD}/sasinside:/sasinside \
  --hostname sas.viya.programming \
  --name sas-viya-programming \
  sas-viya-programming
```

For Kubernetes environments, edit the sas-analytics.yml file to mount the volume. In the following example, an NFS server is used.

```yaml
apiVersion: apps/v1beta1
class: Deployment
metadata:
  name: sas-programming
spec:
  replicas: 1
template:
  metadata:
    labels:
      app: sas-programming
  spec:
    containers:
      - name: sas-programming
        <content removed intentionally>
        volumeMounts:
          - name: sasconfig
            mountPath: /sasinside
        volumes:
          - name: sasconfig
            nfs:
              server: sasconfig.company.com
              path: "*/vol"
```

**Note:** To pick up configuration changes, a user must connect to a new session of CAS.

### Specify CAS Environment Options

For SAS Viya for Containers, CAS environment options are specified as environment variables or are recorded in a file on disk that is mounted to the container.

To specify a CAS environment option as an environment variable:

- Prefix the CAS environment option with CASENV_ and capitalize the option name. In the following examples, the ADMIN_USER option is prefixed with CASENV_ and is set to sasdemo.
- For Docker environments, use the Docker run command to set the CAS environment option. Here is an example:

```bash
docker run \
  --detach \
  --rm \ 
  --env CASENV_ADMIN_USER=sasdemo \
  --hostname sas.viya.programming \
  --name sas-viya-programming \
  sas-viya-programming
```
For Kubernetes environments, an addition is made to the manifest file (sas-analytics.yml). Here is an example:

- name: CASENV_ADMIN_USER
  value: "sasdemo"

To specify the CAS environment options in a file that is mounted to the container:

1. In the directory where you unpacked the SAS_Viya_deployment_data.zip file, run Docker to create a `sasinside` directory, and then create a file in that directory named `casconfig_usermods.lua`.

2. Edit the `casconfig_usermods.lua` file to specify the CAS configuration options, such as `Env.ADMIN_USER='sasdemo'`, and then save the file.

3. Perform one of the following steps:
   - For Docker environments, run the container to link the location of the `casconfig_usermods.lua` file to where the software expects it.
     ```bash
     docker run \
     --detach \
     --rm \
     --volume ${PWD}/sasinside:/sasinside \
     --hostname sas.viya.programming \
     --name sas-viya-programming \
     sas-viya-programming
     ```
   - For Kubernetes environments, edit the `sas-analytics.yml` file to mount the volume. In the following example, an NFS server is used.
     ```yaml
     apiVersion: apps/v1beta1
     kind: Deployment
     metadata:
       name: sas-programming
     spec:
       replicas: 1
     template:
       metadata:
         labels:
           app: sas-programming
       spec:
         containers:
         - name: sas-programming
           <content removed intentionally>
           volumeMounts:
             - name: sasconfig
               mountPath: /sasinside
           volumes:
             - name: sasconfig
               nfs:
                 server: sasconfig.company.com
                 path: "/vol"
     ```

Note: To pick up configuration changes, a user must connect to a new session of CAS.
Apply a New License

Before You Begin

To apply a new license for SAS Viya for Containers running in Docker or Kubernetes environments, perform the steps in this topic.

SAS Viya uses a single licensing file. The contents of the license file resides in an environment variable or in a file on disk that is mounted to the container. You apply a new license to enable new products or to extend expiration dates on existing products.

Both SAS Cloud Analytic Services (CAS) and SAS Foundation use the same license. During the deployment, a license is applied to both the CAS in-memory compute engine and the SAS Foundation compute engine.

For additional information about licensing, see Licensing in SAS Viya Administration: Licensing.

Apply a New License

Perform one of the following steps to apply a new license.

For Docker environments:

1. Copy the new license file to the sasinside directory that was mounted to the running container. Make sure that it is named license.sas.

2. Run `docker restart sas-programming` to stop and start the running container, which applies the new license.

   Note: To pick up a new license, the Docker container must be restarted.

For Kubernetes environments:

1. Use the `kubectl get configmaps` command to get the defined ConfigMap. Look for `sas-programming` in the resulting list.

2. Run the following command to get the sp_configmap.yml file:

   ```
kubectl get configmap -o yaml sas-programming > sp_configmap.yml
   ```

3. Edit the sp_configmap.yml file to add the contents of the new license file.

   Important: When you insert the license information, ensure that you maintain the indentation from the license file.

   Here is an example of the license information:

   ```
   poac.SETINIT_TEXT: |
   PROC SETINIT RELEASE='V03';
   SITEINFO NAME=your-site-name'
   SITE=site-number OSNAME='LIN X64' RECREATE WARN=47 GRACE=45
   ```

4. Run the following command to update the ConfigMap.

   ```
kubectl replace -f sp_configmap.yml
   ```

5. Restart the container.

   a. Run the `kubectl get pods` command to get a list of the running pods. Look for the pod that begins with `sas-programming`. 
Run the following command to stop the current pod:

```bash
kubectl delete pod name-of-pod
```

Kubernetes creates a new pod that has the new license applied.

## Access Logs

To access logs, use one of the following commands:

- `docker logs`
- `kubectl logs`
Removing SAS Viya

Remove with Docker .................................................................................................................. 47
Remove with Kubernetes .............................................................................................................. 47

Remove with Docker

1. Stop the container.
   ```bash
docker stop sas-programming
   ```

2. Determine if the container is still listed.
   ```bash
docker ps -a \ 
   --filter name=sas-programming --format "table {{.ID}} \t{{.Names}} \t{{.Image}}" | \ 
   grep sas-programming
   ```
   Here is an example of the output:
   ```
   0180fa635ac2        sas-programming
   ses.sas.download/va-104.0.0-x64_redhat_linux_7-docker/sas-viya-programming:3.4.7-20180702.1530552574483
   ```
   If the output is empty, skip to step 4.

3. Remove the container.
   ```bash
docker rm sas-programming
   ```

4. (Optional) If the image is no longer needed, then remove it. Here is an example:
   ```bash
docker rmi ses.sas.download/va-104.0.0-x64_redhat_linux_7-docker/sas-viya-programming:3.4.7-20180702.1530552574483
   ```

Remove with Kubernetes

1. Verify that the images are still running.
   ```bash
kubectl get pods | grep sas
   ```
   The output is a list of pods and their status.

2. Remove the service and pod.
   ```bash
   kubectl delete -f sas-programming.yml
   ```
   As the service and pod are removed, messages that indicate success are displayed. Here is an example:
   ```bash
   service "sas-programming" deleted
   ```
statefulset "sas-programming" deleted

3 (Optional) Make sure that the service and pod are removed.

    kubectl get pods | grep sas
    kubectl get svc | grep sas

No results indicate a successful removal of the service and pod.
Pre-deployment Hadoop Tasks

Pre-deployment Checklist

Before you install SAS Viya software that interacts with Hadoop and Hive, it is recommended that you verify your Hadoop environment. Use the following checklist:

- Ensure that you have configured SAS Data Connector to Hadoop and, if required, SAS Data Connector Accelerator for Hadoop. For details, see “Configure SAS/ACCESS Interface to Hadoop” on page 27.
- Understand and verify your Hadoop user authentication.
- Have sudo access on the NameNode.
- Enable the HDFS user with Write permission to the root of HDFS.
  The HDFS user home directory, /user/user-account, must exist and must have drwxrwxrwx permissions for the HDFS user directory.
- Verify that the Hadoop master node can connect to the Hadoop slave nodes using passwordless SSH. For more information, see the Linux manual pages about ssh-keygen and ssh-copy-id.
- Understand and verify your security setup.
  - Verify that you can use your defined security protocol to connect from your client machine, which is outside of the SAS Viya environment) to your Hadoop cluster.
  - It is highly recommended that you enable Kerberos or another security protocol for data security. If your cluster is secured with Kerberos, you must obtain a Kerberos ticket. You also must have knowledge of any additional security policies.
For clusters that have Kerberos security enabled, verify that you have a valid ticket on the node on which the Hive2 service is running.

- Gain working knowledge about the Hadoop distribution that you are using (for example, Cloudera or Hortonworks).
  You also need working knowledge about the HDFS, MapReduce 2, YARN, and Hive services. For more information, see the Apache website or the vendor’s website.
  For MapR, you must install the MapR client. The installed MapR client version must match the version of the MapR cluster that SAS Viya connects to. For more information, see the MapR documentation.
- Verify that the HCatalog, HDFS or Hive, MapReduce, and YARN services are running on the Hadoop cluster. SAS Viya software uses these various services, which ensure that the appropriate JAR files are located during the configuration.

- For the Hive server:
  - Identify the machine on which the Hive server is running. If the Hive server is not running on the same machine as the NameNode, note the server and port number of the Hive server for future configuration.
  - Know the host name of the Hive server and the host name of the NameNode.

- For MapReduce:
  - Know the location of the MapReduce home directory.
  - Request permission to restart the MapReduce service.
  - Verify that you can run a MapReduce job successfully.

### Security

#### Kerberos Security

SAS Data Connector to Hadoop can be configured for a Kerberos ticket cache-based logon authentication by using MIT Kerberos 5 Version 1.9.

**Note:** SAS Viya must be configured for pluggable authentication module (PAM) support.

If you are using Advanced Encryption Standard (AES) encryption with Kerberos, you must manually add the Java Cryptography Extension local_policy.jar file to each instance of JAVA_HOME on the Hadoop cluster. If you are located outside the United States, you must also manually add the US_export_policy.jar file. The addition of these files is governed by the United States import control restrictions.

If you are using the Oracle JRE or the IBM JRE, the appropriate JAR file must also replace the existing local_policy.jar file and the US_export_policy.jar file in your JRE location. This location is typically the JAVA_HOME/jre/lib/security/ directory. You can obtain the appropriate file from the Oracle website or the IBM website.

It is recommended that you back up the existing local_policy.jar file and the US_export_policy.jar file first in case they need to be restored.

If you are using the OpenJDK, the files do not need to be replaced.

#### JDBC Read Security for Hive

SAS Data Connector to Hadoop can access Hadoop data through a JDBC connection to Hive. Depending on your release of Hive, Hive might not implement Read security. A successful connection from SAS Viya can allow Read access to all data that is accessible to the identity that is used to access the Hive server. Hive can be secured with Kerberos. SAS Data Connector to Hadoop supports Kerberos 5 Version 1.9 or a later release.
Configure SAS/ACCESS to Hadoop and SAS Data Connector to Hadoop

Requirements to Deploy JAR Files for Viya 3.4 on Docker

- Hadoop cluster manager:
  - host name and port number
  - credentials (account name and password)
- Hive service host name
- SSH credentials of the Linux account that has access to the machine on which the Hive service has been installed and is running.
- If your deployment includes MapReduce users from Windows clients, after you run the hadoop_extract.sh script, you must follow the instruction to edit the mapred-site.xml file and set the mapreduce.app-submission.cross-platform property to true.

Install the Hadoop JAR Files on the Docker Host

Obtain and Install the Hadoop Tracer Script Manually

To run the Hadoop tracer script successfully:

- Ensure that the user running the script has passwordless SSH access to all of the Hadoop services.
- Ensure that Python 2.6 or later and strace are installed. Contact your system administrator if these packages are not installed on the system.
- Ensure that the user running the script has authorization to issue HDFS and Hive commands.
- If Hadoop is secured with Kerberos, obtain a Kerberos ticket for the user before running the script.

To obtain and run the Hadoop tracer script

1. On the Hadoop server, create a temporary directory to hold a ZIP file that you download later. An example would be /opt/sas/hadoopfiles/tmp.
2. Download the hadooptracer.zip file from the following FTP site to the directory that you created in step 1. ftp.sas.com/techsup/download/blind/access/hadooptracer.zip
3. Using a method of your choice (for example, PSFTP, SFTP, SCP, or FTP), transfer the ZIP file to the Hive node on your Hadoop cluster.
4. Unzip the file. The hadooptracer.py file is included in this ZIP file.
5. Change permissions on the file to have EXECUTE permission.
   ```
   chmod 755 ./hadooptracer.py
   ```
6. Run the tracer script.
   ```
   python ./hadooptracer.py --filterby=latest
   ```
Note: The filterby=latest option ensures that if duplicate JAR or configuration files exist, the latest version is selected. If you want to pull the necessary JAR files without filtering, use filterby=none or do not use the filterby argument at all.

This tracer script performs the following tasks:

- Pulls the necessary Hadoop JAR and configuration files and places them in the /tmp/jars directory and the /tmp/sitexmls directory, respectively.
- Creates a hadooptracer.json file in the /tmp directory. If you need a custom path for the JSON output file, use this command instead:
  python ./hadooptracer.py -f /your-path/hadooptracer.json
- Creates a log in the /tmp/hadooptracer.log directory.

Note: Some error messages in the console output for hadooptracer.py are normal and do not necessarily indicate a problem with the JAR and configuration file collection process. However, if the files are not collected as expected or if you experience problems connecting to Hadoop with the collected files, contact SAS Technical Support and include the hadooptracer.log file.

7 On the Docker host machine, create the hadoop/config and hadoop/jars directories under your sasinside directory.

8 Using a method of your choice (for example, PSFTP, SFTP, SCP, or FTP), copy the files in the /tmp/jars and /tmp/sitexmls directories on the Hadoop server to the hadoop/config and hadoop/jars directories respectively on the Docker host machine that you created in step 7.

Note: If you connect to the Hadoop server with an HTTP REST API, you do not need the Hadoop JAR files on the SAS client machine.

Set Up Multiple Hadoop Versions for Multiple Hadoop Servers

If you have multiple Hadoop servers that run different Hadoop versions:

- The version of the JAR files in the hadoopJarPath directory on the CAS server must match the version of the JAR files on the Hadoop server to which CAS connects.
- Each CAS session can connect only to Hadoop clusters of one configured hadoopJarPath version.
- Separate concurrent CAS sessions can independently connect to different versions of Hadoop clusters.

To support multiple Hadoop versions:

1 Create and populate separate directories with version-specific Hadoop JAR files for each Hadoop version.
2 Start separate CAS sessions, and point each separate CAS session to one of the hadoopJarPath versions.

Upgrading your Hadoop server version might involve multiple active Hadoop versions. The same multi-version instructions apply.

Verify SAS Data Connector to Hadoop

To verify that the software has been successfully deployed:

1 From SAS Studio, edit and run the following SAS code to verify the SAS/ACCESS to Hadoop LIBNAME statement:

```sas
option set=SAS_HADOOP_CONFIG_PATH="path-to-config-files";
option set=SAS_HADOOP_JAR_PATH="path-to-jar-files";
libname hdplib hadoop server="hive-hadoop-host-name"
user=user-ID password=user-password;
```
Note: Do not use the USER= argument or PASSWORD=argument if your Hadoop cluster is secured by Kerberos.

For more information, see LIBNAME Statement Specifics for Hadoop.

If SAS/ACCESS to Hadoop was successfully deployed, the execution of the LIBNAME statement will return results without error.

2 From SAS Studio, edit and run the following SAS code to verify SAS Data Connector to Hadoop:

```sas
caslib hdlib datasource=srctype="hadoop", dataTransferMode="serial", username="user-ID",
server="hadoop-host-name",
hadoopjarpath="path-to-jar-files",
hadoopconfigdir="path-to-config-files",
schema="Hadoop-schema-name");

proc casutil;
list files incaslib="hdlib";
run;
```

If the data connector was successfully deployed, the results are the names of the tables in Hive.

If an error was returned on the execution of the LIBNAME statement or no table information was returned for the data connector, you should perform the configuration steps again.