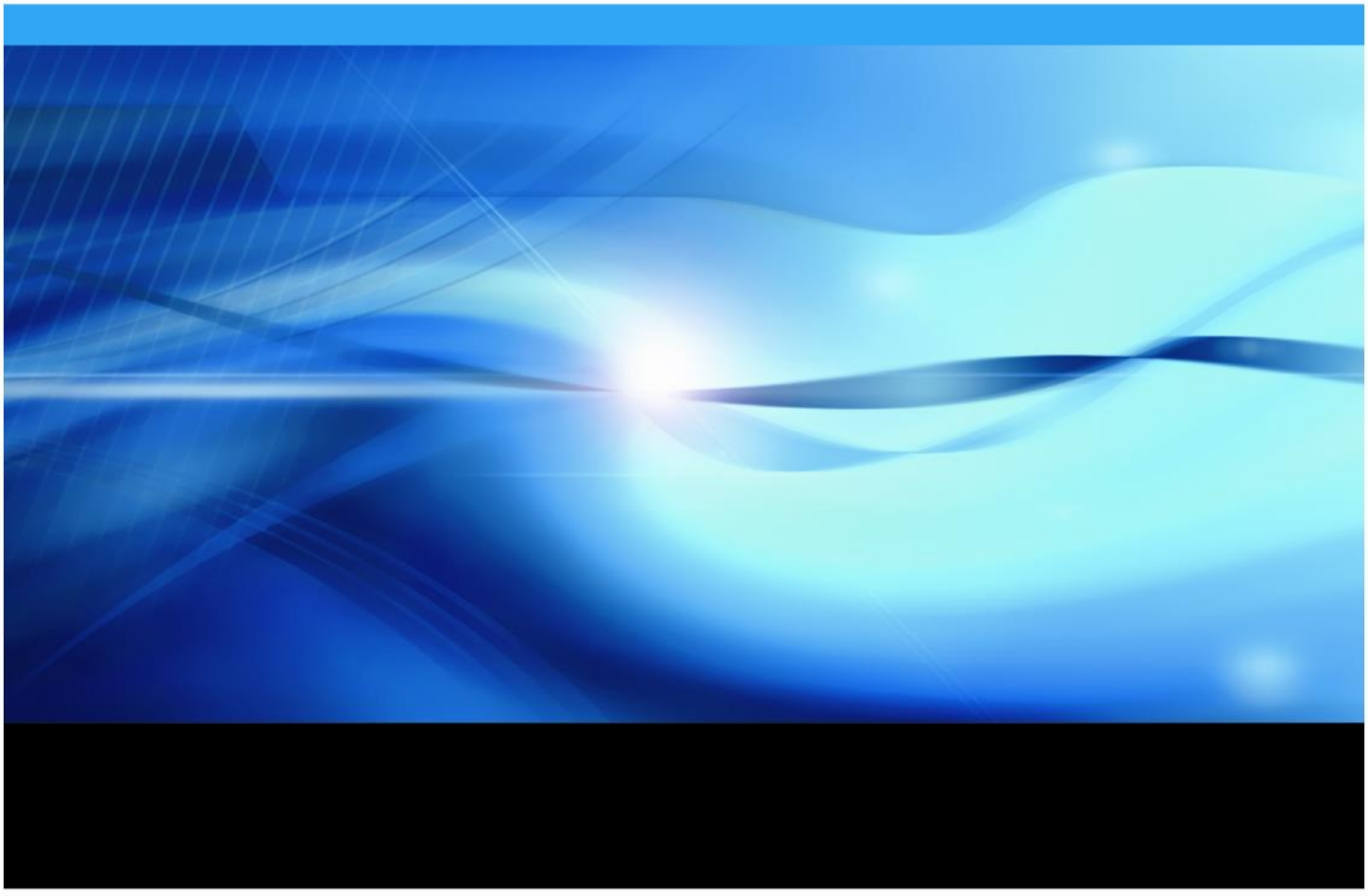
Configuring the Hadoop Cluster for Use by SAS® Grid Manager for Hadoop



The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2017. *Configuring the Hadoop Cluster for Use by SAS® Grid Manager for Hadoop*

Cary, NC: SAS Institute Inc.

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

January 2017

SAS® Publishing provides a complete selection of books and electronic products to help customers use SAS software to its fullest potential. For more information about our e-books, e-learning products, CDs, and hard-copy books, visit the SAS Publishing Web site at **support.sas.com/bookstore** or call 1-800-727-3228.

Table of Contents

Contents

Chapter 1 — Introduction

The purpose of this document is to help the reader successfully deploy a SAS Grid Manager for Hadoop deployment into an existing Linux Hadoop cluster.

Supported Linux Hadoop Distributions

- Cloudera 5.2.x and higher
- Hortonworks 2.1.x and higher. Hortonworks 2.2 and higher is required to have the REST API for job submission
- MapR 4.0.x and higher

Note: *MapR 4.1.x and higher is recommended. MapR introduced support for cron syntax in the coordinator's frequency attribute with Oozie. This functionality s required in order to schedule flows with recurring time events.*

Chapter 2 - Before Running the SAS Deployment Wizard

Set Up Shared File System

While SAS Grid Manager for Hadoop reduces the need for a shared file system, a shared file system is still required for certain functionality and facilitates the following:

- Sharing SAS installation and configuration information.
- Sharing batch processing job information between clients and grid machines.
- Allowing SAS batch processing to restart processing at the last checkpoint.
- Allowing user home directories to be shared across all nodes.
- Sharing project/data directories required by other SAS products running on the grid (for example, SAS Enterprise Miner and SAS Forecast Server).

Set Up Users

All users who access the cluster through SAS must have the same UID/GID on every machine in the cluster. This includes the SAS installer account.

Install Hadoop Services

For SAS Grid Manager for Hadoop to install, configure, and work correctly, the following Hadoop services are required:

- HDFS or MapR-FS for file storage
- HTTP-FS for WebHDFS REST API on MapR-FS
- YARN for resource management
- Oozie for workflow scheduling
- Hue, if the Oozie web GUI is surfaced through Hue
- Hive, for the SAS Hadoop Configuration component in the SAS Deployment Wizard to be able to access the required Hadoop configuration and jar files

Enable Kerberos

You must enable Kerberos authentication for the Hadoop cluster so that the YARN jobs can be executed as the person submitting the job. You enable Kerberos authentication by setting a number of service properties in the cluster (see Appendix 1 - Hadoop Kerberos Properties). Most Hadoop distributions provide a mechanism for enabling Kerberos authentication in the cluster.

To test whether Kerberos is set up properly in your cluster, run the PI MapReduce sample with and without a credential cache. If the credential cache is valid, it will run successfully.

With Kerberos enabled, the following test should be successful:

```
1. kinit
2. yarn jar <path_to_YARN_examples>/hadoop-mapreduce-examples.jar pi 16 100000
```

With Kerboros enabled, the following test should NOT be successful:

```
1. kdestroy
2. yarn jar <path_to_YARN_examples>/hadoop-mapreduce-examples.jar pi 16 100000
```

If both of the above tests work, Kerberos is not enabled.

Set Up SAS and Kerberos Libraries

SAS dynamically loads Kerberos libraries and searches for libraries that do not have versions in the library name. This means that the files `libkrb5.so` and `libgssapi_krb5`.so must exist for SAS to work properly. Most Linux systems have Kerberos libraries installed, but they are the versioned type of library. Therefore, you must do one of the following:

- Create symbolic links from non-versioned names to the versioned libraries:
  - `ln –s /lib64/libkrb5.so.3 /usr/lib64/libkrb5.so`
  - `ln –s /lib64/libgssapi_krb5.so.2 /usr/lib64/libgssapi_krb5.so`
- Install the Kerberos developer RPM.

Creating Kerberos Tickets For Grid-launched Servers

When the object spawner launches a server in YARN, a module used by the object spawner will execute the 'kinit' command to generate a credential cache to access the cluster. The call to kinit will not specify the principal name or realm, but rather lets kinit default to the principal being the user requesting the server, the realm being the default realm for the object spawner host, and the password being the password sent to the object spawner to start the server. If any of these values are different, the customer will need to write a script named 'kinit' that gets executed in place of the normal kinit where it can perform the kinit with the proper principal and realm and password. The content of the script will be specific to the way each customer performs authentication.

Enable HTTP Authentication

If the cluster is based on Hadoop version 2.6 or higher, you must enable HTTP authentication for YARN.

With HTTP authentication enabled, the following should work:

```
1. kinit
2. curl --negotiate -u :  http://<RM_host>:<RM_web_port>/cluster
```

With HTTP authentication enabled, the following should NOT work:

```
1. kdestroy
2. curl --negotiate -u :  http://<RM_host>:<RM_web_port>/cluster
```

MapR clusters have their own form of authentication which require that you run the "`maprlogin`" command to access the cluster. In the examples above, the "`maprlogin kerberos`" command should be run after every "`kinit`" and "`maprlogin logout`" should be run after every "`kdestroy`".

Enable SSL

If a secure transmission of information and data is required, you must configure SSL in the cluster. After you configure SSL in the cluster, SAS products must be able to verify the SSL certificate sent to SAS from all the hosts in the cluster.

Update YARN parameters

The following parameter is not specified in a default YARN configuration and is needed for some of the SAS Grid Manager for Hadoop functionality:

- yarn-site.xml

    - `yarn.nodemanager.user-home-dir`
        - Required for grid-launched foundation servers (such as a workspace server) to work.
        - Required for running SAS through the SAS Grid Manager for Hadoop Client Utility (SASGSUB).

The MapR distribution also requires some additional properties for SAS products to work properly. Add the following files and properties need to the Hadoop cluster configuration files:

- core-site.xml
    - Set `fs.defaultFS` to '`maprfs:///`'
- yarn-site.xml
    - Set `yarn.resourcemanager.hostname` to the host name of the resource manager
    - Set `yarn.http.policy` to HTTP_ONLY or HTTPS_ONLY depending on whether SSL is enabled or not in the cluster
    - Set either `yarn.resourcemanager.webapp.address` or `yarn.resourcemanager.webapp.https.address` to the Web address of the resource manager
        - If SSL is not used (and HTTP_ONLY is set), set the `yarn.resourcemanager.webapp.address` property to '${yarn.resourcemanager.hostname}:*<port>*' where port is the port of the resource manager's web interface.
        - If SSL is used (and HTTPS_ONLY is set), set the `yarn.resourcemanager.webapp.https.address` property to '`${yarn.resourcemanager.hostname}:<ssl_port>`' where `ssl_port` is the SSL port of the resource manager's web interface.

For examples of these setting, see Appendix 4 - Example Properties.

Set Up HDFS Directories

SAS Grid Manager for Hadoop requires several directories in HDFS. If the directories do not exist, you must create the appropriate directories as follows:

- /tmp
    - SAS Hadoop products will use `/tmp` and expect the `sticky` bit to be turned off. SAS Hadoop Configuration will complain if the bit is set.
    - SAS job launching will use `/tmp` and create `/tmp/SASGrid` in that directory. If the SASGrid directory is deleted, it will be re-created. The directory will be created as open to all users and groups.
- /user/<user_name>
    - SAS Scheduling needs to have a user directory for every user that will submit scheduled jobs.

Chapter 3 - Run the SAS Deployment Wizard

To install and configure a SAS Grid Manager for Hadoop Control Server, do the following:

1. Log on as the SAS install user to the machine that contains Resource Manager.
2. Run the "`kinit`" command to initialize the Kerberos credential cache.
3. Run the SAS Deployment Wizard (SDW).

Verify Hadoop Configuration and Jar Files

During the configuration process, the SDW will connect to the cluster, determine the configuration files and jar files that will be needed, and place them into the configuration directory for SAS to use. To do this, SDW will need the administrator password for Cloudera Manager or Ambari along with a username and password of someone that can run Hive jobs on the cluster. Once configuration has completed, environment variables that SAS uses to determine the location of those files will be set in the `hadoop_env` script file located in the `LevX` subdirectory under the configuration directory.

If for some reason this step fails, you can re-run from the SAS Deployment Manager (SDM) using the "Configure Hadoop Client Files" under the "Hadoop Configuration" task.

Validate SAS/ACCESS

During configuration, you have the option to validate that SAS/ACCESS is working.

If you want to validate SAS/ACCESS while configuring and use higher levels of encryption, you must run the SDW once to install, then update the policy files, and then run the SDW again to do the configuration and validation.

Note: *If you perform an install and configuration all in one pass of the SDW, you may not be able to validate SAS/ACCESS. This is because the SAS Private Java Runtime Environment (JRE) is missing security policy files that it may need to use higher levels of encryption for Kerberos.*

Specify Scheduling Directories

In SAS Grid Manager for Hadoop, scheduling consists of both of the following:

- the Data Step Batch Server
- the Oozie Scheduling server

The Data Step Batch Server contains information about SAS processing including the command to run, the directory to put the log files, and the directory to put the output files. You can specify the log directory in HDFS by prepending the path with '`hdfs://`' (or '`maprfs://`' if using MapR).

The Oozie Scheduling Server contains information including the coordinator directory and the workflow directory. Both of the Oozie directory paths are assumed to be in HDFS already so nothing needs to be prepended to the path value.

Verify the Oozie Server Host

Since you are running the SDW on the host that has Resource Manager installed, the grid control server will be the Resource Manager host. This does not necessarily mean that the Oozie server host will be correct. When the Oozie Scheduling Server dialog asks for the Oozie host, make sure the host name is the host name that the Oozie server is running on.

Chapter 4 - After Running the SAS Deployment Wizard

Create a SAS Grid Policy File

The SAS Grid Policy file defines the application types that will be running on the SAS Grid. An application type consists of a set of attributes that should apply to a SAS Grid job of the specified type. These attributes can include the memory the job requires, the number of virtual cores the job requires, the queue to use, the priority of the job, the hosts the job can run on, and others. If no policy file is created, SAS job launching will use the maximum memory allowed by YARN along with the fewest cores allowed by YARN. The location of the policy file is determined in the following order:

1. The location is specified by the `policyFile` option in the grid options for the logical grid server.
2. The file `sasgrid-policy.xml` exists in the same path as the Hadoop configuration files (which is specified by the `SAS_HADOOP_CONFIG_PATH` environment variable)
3. The file `sasgrid-policy.xml` exists in the `HttpFS` directory `/tmp/SASGrid`.

Once the policy file is created, you can set the `appType` grid option in metadata to associate a SAS application server context or grid option set with an application type defined in the policy file. The `appType` may also be used on the job options macro variable for the `grdsvc_enable` statement and the `GRIDJOBOPTS` option for SAS Grid Manager for Hadoop Client Utility (SASGSUB).

A sample SAS Grid policy file is listed in Appendix 2 - Sample Grid Policy File.

Update Java Policy Files

If your Kerberos implementation is using keys that are encrypted with 256 bit AES, Java will require updated security policy jars to work properly with Kerberos. To determine what keys your implementation is using, initialize your credential cache using `kinit` and then run the `klist` command with the "`-e`" option to list the encryption type. If the encryption type has "`aes256`" in the string, you need to update the policy files.

The policy files live in `<JAVA_HOME>/lib/security` and have the names `local_policy.jar` and `US_export_policy.jar`. Oracle provides unlimited security policy jars for each version of Java, which can be downloaded from the following locations:

Java 6: http://www.oracle.com/technetwork/java/javase/downloads/jce-6-download-429243.html

Java 7: http://www.oracle.com/technetwork/java/javase/downloads/jce-7-download-432124.html

Java 8: http://www.oracle.com/technetwork/java/javase/downloads/jce8-download-2133166.html

To determine what version of Java you are using, run the "`java --version`" command. Version 1.6.xxxx is Java 6, 1.7.xxxx is Java 7 and 1.8.xxxx is Java 8. By default, SAS will use the SAS Private JRE unless told to use some other Java JRE during installation. If you are using the SAS Private JRE, the Java 7 policy files need to be placed in `<SASROOT>/SASPrivateJavaRuntimeEnvironment/9.4/jre/lib/security`.

Verify the Object Spawner Keytab

When the Object Spawner needs to access the cluster to get information for grid load-balancing or grid launched workspace servers, it may be required to authenticate to the cluster using Kerberos if the cluster has HTTP authentication enabled. To do this authentication, the spawner requires a

keytab file to be placed in the ObjectSpawner directory. The file, "`objspawn.keytab`" must contain the Kerberos credentials of the service account used to run the spawner. The file should be owned by the service account user and have permissions set so that only that account may read the file ('400'). For default Linux distributions that use MIT Kerberos, the `ktutil` command can be used to create a keytab file.

Extract the SSL Certificates

If you have used SSL to protect data-in-motion on the cluster, the certificate that can verify the various servers' certificates must be put into the SAS Security Certificate Framework. This is either a Certificate Authority certificate or a machine's self-signed certificate. Unfortunately, the certificate needed is usually in a Java `truststore` but the certificate format needed for the SAS Security Certificate Framework is a human-readable format called "PEM" (Privacy Enhanced Mail). The Java `keytool` command and OpenSSL's `openssl` command can be used to convert one to the other.

To extract the certificate out of a Java `truststore` into a DER (Distinguished Encoding Rules) format file:

```
keytool -export -keystore mytruststore -alias myCertAlias -file myCert.der
```

Now the file can be converted from DER to PEM using the openssl command:

```
openssl x509 -inform der -in myCert.der -out myCert.pem
```

Install the MapR Client

If you are using a MapR cluster, the MapR client software must be installed on any machine that will be accessing the MapR cluster. Since MapR requires SSL communication, the MapR cluster's `ssl_truststore` must be copied from the cluster to the client machine.

Deploy SAS to the Grid Nodes

In a typical SAS Grid deployment, a shared directory is used to share the installation and configuration directories between machines in the grid. With SAS Grid Manager for Hadoop, you can either use NFS to mount a shared directory on all cluster hosts or use the SAS Deployment Manager (SDM) to work with the cluster manager to distribute the deployment to the cluster hosts. The SDM has the ability to create Cloudera parcels and Ambari packages to enable the distribution of the installation and configuration directories from the grid control server to the grid nodes. Start SDM and under the "Hadoop Configuration" task, select the "Deploy SAS Grid Manager for Hadoop" option.

The SSH user that you select has to have permission to write to `/opt/Cloudera` for Cloudera Manager, and `/var/lib/ambari-server` for Ambari. If the owner of these directories is root, someone with password-less sudo privileges will work also.

Since the size of a SAS installation and configuration directory can be large, creating the parcel/package and distributing it to nodes in the cluster can be time consuming.

Set up Scheduling Directories

If you changed any of the following directories, you must set up those directories in HDFS before you deploy or schedule jobs:

- the default directory for the Data Step Batch Server log output to an HDFS location
- the default Oozie server coordinator directory

- the default workflow directory

Verify the Cluster

Verify the cluster as follows:

1. Validate the workspace server in SAS Management Console.
    - A workspace server is required to validate the grid server.
    - Initially, it will be easier to validate a non-grid-launched workspace server. After the grid server has been validated, you can change it to grid-launched load balancing and run validate again.
        - Grid-launched workspace servers will have Kerberos credentials generated.
        - Non-grid-launched workspace servers require PAM changes to create Kerberos credentials. Otherwise, you need to generate Kerberos credentials wherever the workspace server is going to run.
2. Validate the grid server in SAS Management Console.
3. Run the test program included in Appendix 3 – Grid Test Program. This program will get a list of hosts available to a SAS Application Server context and try to SIGNON to each of those hosts using YARN. To ensure each host it tries to use is available, the code should be run in a normal (non-grid-launched) workspace server rather than a grid-launched workspace server.

Appendix 1 – Hadoop Kerberos Properties

This section contains a list of the services that need to have Kerberos authentication enabled and the properties that need to be set. To configure authentication and properties, please consult the documentation for your specific distribution.

- Hadoop Core (in `core-site.xml`)
  - hadoop.security.authentication
  - hadoop.security.authorization
  - If using HTTP authentication (required for YARN on Hadoop 2.6 or higher)
    - hadoop.http.authentication.type
    - hadoop.http.authentication.kerberos.principal
    - hadoop.http.authentication.kerberos.keytab
    - hadoop.http.authentication.token.validity
    - hadoop.http.authentication.signature.secret.file
- HDFS (in hdfs-site.xml)
  - dfs.block.access.token.enable
  - Web Interface
    - dfs.web.authentication.kerberos.principal
    - dfs.web.authentication.kerberos.keytab
  - Name Node
    - dfs.namenode.kerberos.principal
    - dfs.namenode.keytab.file
  - Secondary Name Node
    - dfs.secondary.namenode.kerberos.principal
    - dfs.secondary.namenode.keytab.file
  - Data Node
    - dfs.datanode.kerberos.principal
    - dfs.datanode.keytab.file
    - dfs.datanode.data.dir.perm (allow only principal to manipulate files)
    - dfs.datanode.address (must be less than 1024)
    - dfs.datanode.http.address (must be less than 1024)
- HTTP-FS (in httpfs-site.xml if HDFS not being used)
  - Web Interface
    - httpfs.authentication.type
    - httpfs.authentication.kerberos.principal
    - httpfs.authentication.kerberos.keytab
    - httpfs.authentication.kerberos.name.rules
  - HTTPFS interface to Hadoop
    - httpfs.hadoop.authentication.type
    - httpfs.hadoop.authentication.kerberos.principal
    - httpfs.hadoop.authentication.kerberos.keytab
- YARN (in yarn-site.xml)
  - Resource Manager
    - yarn.resourcemanager.principal
    - yarn.resourcemanager.keytab

    - Node Manager
        - yarn.nodemanager.principal
        - yarn.nodemanager.keytab
    - Web Interface
        - yarn.web-proxy.principal
        - yarn.web-proxy.keytab
    - Timeline Server
        - yarn.timeline-service.principal
        - yarn.timeline-service.keytab
- Oozie (in oozie-site.xml)
    - local.realm
    - Web Interface
        - oozie.authentication.type
        - oozie.authentication.kerberos.principal
        - oozie.authentication.kerberos.keytab
        - oozie.authentication.kerberos.name.rules
    - Oozie interface to Hadoop
        - oozie.service.HadoopAccessorService.kerberos.enabled
        - oozie.service.HadoopAccessorService.kerberos.principal
        - oozie.service.HadoopAccessorService.keytab.file

Appendix 2 – Sample SAS Grid Policy File

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<GridPolicy  defaultAppType="normal">

    <GridApplicationType name="normal">
        <priority>20</priority>
        <nice>10</nice>
        <memory>1024</memory>
        <vcores>1</vcores>
        <runlimit>120</runlimit>
        <queue>normal_users</queue>
        <hosts>
            <host>myHost1.mydomain.com</host>
            <hostGroup>development</hostGroup>
        </hosts>
    </GridApplicationType>

    <GridApplicationType name="priority">
        <jobname>High Priority!</jobname>
        <priority>30</priority>
        <nice>0</nice>
        <memory>1024</memory>
        <vcores>1</vcores>
        <runlimit>120</runlimit>
        <queue>priority_users</queue>
        <hosts>
            <host>myHost2.mydomain.com</host>
        </hosts>
    </GridApplicationType>

    <HostGroup name="development">
        <host>myHost3.mydomain.com</host>
        <host>myHost4.mydomain.com</host>
    </HostGroup>

</GridPolicy>
```

Appendix 3 – Grid Test Program

```
options nosource;           /* Do not show source when running...  */
options linesize=MAX;       /* Try not to wrap output              */

/*****************************************************************
** Set the name of the SAS Application Server context to use.
*/
%let appsrv=SASApp;

/*****************************************************************
** Setup metadata values if not running in a workspace server
options metaserver="myServer.myDomain.com";
options metaport=8561;
options metauser="myUser";
options metapass="myPassword";
*/


/*****************************************************************
** Make sure we can use the grid.
*/
%put
rc=%sysfunc(grdsvc_enable(_ALL_,"server=&appsrv;jobname=jobNameVar;jobopts=jobOptVar;"));

%let jobNameVar=SAS Grid Manager for Hadoop Test Program;

/*****************************************************************
** Get a list of the hosts we want to verify
*/
%let hostList = %sysfunc(grdsvc_hostlist(server=&appsrv));

%put ==>>GRIDTEST<<=======================================================;
%put ==>>GRIDTEST<<== Host List is "&hostList";
%put ==>>GRIDTEST<<=======================================================;

/*****************************************************************
** Macro that will perform the SIGNONs and RSUBMITS
*/
%macro runjobs(hostList);

  %let i=1;

  /***************************************************************
  ** For each host in the list, perform an asynchronous SIGNON
  */
  %do %while (&i > 0);

    %let host=%scan(%BQUOTE(&hostList),&i,',');

    %if ("&host" ne "") %then

      %do;
        %global host&i.Name;
        %global host&i.Signon;
        %global host&i.Rsubmit;
        %let host&i.Name=&host;
        %let host&i.Signon=0;
        %let host&i.Rsubmit=0;
```

```
        %put ==>>GRIDTEST<<==========================================================;
        %put ==>>GRIDTEST<<== Performing SIGNON to host "&host";
        %put ==>>GRIDTEST<<==========================================================;

        %let jobOptVar=testHost=&host;

        SIGNON host&i SIGNONWAIT=NO CMACVAR=host&i.Signon;

        %let hostCount=&i;
        %let i=%eval(&i+1);
      %end;

    %else
      %let i=0;
  %end;

  %put There are &hostCount hosts in the grid;

  /*****************************************************************
  ** Wait for each host to finish the SIGNON and when one finishes,
  ** RSUBMIT code for it to report its name
  */
  %let hostRsubmits=0;
  %let waitList=;
  %let loops=0;

  %do %while (&hostRsubmits < &hostCount and &loops < 10);

    %let i=1;

    %do %while (&i le &hostCount);

      %* %put Host &i (&&host&i.Name) Submit is &&host&i.Signon and Rsubmit is
&&host&i.Rsubmit;

      %if (&&host&i.Signon eq 0) and (&&host&i.Rsubmit eq 0) %then
        %do;
          %put ==>>GRIDTEST<<==========================================================;
          %put ==>>GRIDTEST<<== Submitting SAS code to "&&host&i.Name";
          %put ==>>GRIDTEST<<==========================================================;

          %SYSLPUT remoteHost=&&host&i.Name /remote=host&i;

          RSUBMIT host&i WAIT=NO;
            %macro doit;
              %put ==>>GRIDTEST<<======================================================;
              %put ==>>GRIDTEST<<== This code is expected to run on  "&remoteHost";
              %put ==>>GRIDTEST<<== This code is actually running on "&SYSTCPIPHOSTNAME";
              %put ==>>GRIDTEST<<======================================================;
            %mend;
            %doit;
          ENDRSUBMIT;

          %let hostRsubmits=%eval(&hostRsubmits+1);
          %let host&i.Rsubmit=1;
          %let waitList=&waitList host&i;
        %end;
```

```
      %let i=%eval(&i+1);
    %end;

    %if (&hostRsubmits < &hostCount 0) %then
      %let rc=%sysfunc(sleep(2,1));

    %let loops=%eval(&loops+1);
  %end;

  /*****************************************************************
  ** Wait for all RSUBMITs to all hosts to complete.
  */
  waitfor _all_ &waitList;

%mend;

/*****************************************************************
** Perform a signon to each host in the host list.
*/
%runjobs(%BQUOTE(&hostList));

/*****************************************************************
** Terminate all connections in the grid.
*/
signoff _all_;
```

Appendix 4 – Example Properties

core-site.xml

```
  <property>
      <name>fs.defaultFS</name>
      <value>maprfs:///</value>
      <description>
          The name of the default file system.
      </description>
  </property>
```

yarn-site.xml

```
  <property>
      <name>yarn.resourcemanager.hostname</name>
      <value>myResourceManagerHost.myDomain.com</value>
      <description>
          The hostname of the ResourceManager
      </description>
  </property>

  <property>
      <name>yarn.http.policy</name>
      <value>HTTPS_ONLY</value>
      <description>
          This configures the HTTP endpoint for Yarn Daemons.
          The following values are supported:
          - HTTP_ONLY : Service is provided only on http
          - HTTPS_ONLY : Service is provided only on https
      </description>
  </property>

  <property>
      <name>yarn.resourcemanager.webapp.address</name>
      <value>${yarn.resourcemanager.hostname}:8088</value>
      <description>
          The http address of the RM web application.
      </description>
  </property>

  <property>
      <name>yarn.resourcemanager.webapp.https.address</name>
      <value>${yarn.resourcemanager.hostname}:8090</value>
      <description>
          The https adddress of the RM web application.
      </description>
  </property>
```

SAS is the leader in business analytics software and services, and the largest independent vendor in the business intelligence market. Through innovative solutions, SAS helps customers at more than 65,000 sites improve performance and deliver value by making better decisions faster. Since 1976 SAS has been giving customers around the world THE POWER TO KNOW®.