Paper SAS1775-2018

The Ins and Outs of Internal and External Host Names with SAS® Grid Manager

Paula Kavanagh, SAS Institute Inc., Cary, NC

ABSTRACT

It is common to have network topologies introduce an external host name known only outside the firewall, while having a different host name that is used only internally or behind the firewall. Often, third-party network products are added after the deployment of SAS® that require internal and external URLs, such as load balancers and firewalls. The SAS web applications and SAS® Environment Manager components that are needed for SAS® Grid Manager administration require additional configuration when different internal and external host names are used. It is important to understand the components of SAS Grid monitoring and management in SAS Environment Manager and where you should use each host name. This paper clarifies the configuration for successful communication between the components for a SAS Grid in SAS Environment Manager and external URLs.

INTRODUCTION

The purpose of this paper is to illustrate the flow of events that occur between the SAS Grid components in SAS Environment Manager and how an added network device might cause the need for additional configuration of your SAS Grid. Particularly, the flow related to authentication and validation is discussed in depth.

There are various reasons why multiple host names might be required in an environment. Listing those reasons is not the purpose of this paper. Rather, this paper identifies the issues that might be encountered if there are separate internal and external host names, and discusses how to address them.

This paper applies to the capabilities of SAS Grid in SAS Environment Manager for SAS 9.4M4 and later releases. However, there is a hotfix available for SAS 9.4M3 to allow the same capabilities.

COMPONENTS FOR SAS GRID MANAGER IN SAS ENVIRONMENT MANAGER

SAS GRID MANAGER MODULE

The SAS Grid Manager component that is responsible for monitoring, management and configuration of the Grid is the SAS Grid Manager Module, which is referred to as the “module” in this paper. The server side of the module is a Java web application running in a SAS Web Application Server in the SAS Middle Tier. The client side is a JavaScript application that runs in a web browser. The web application for the module is deployed to the SASServer14 server instance by default.

SAS GRID MANAGER AGENT

The SAS Grid Manager component that is responsible for periodic data collection of the SAS Grid is the SAS Environment Manager Agent for SAS Environment Manager. This is referred to as the “agent” in this paper. The data collected by the agent is used to generate graphs, alerts and reports in SAS Environment Manager.

PLATFORM WEB SERVICES

Platform Web Services (PWS) is a third party web application provided by Platform Computing that is deployed in a SAS Web Application Server. PWS is a back-end RESTful web service that communicates with Platform LSF on behalf of clients such as the SAS Grid Manager Module and Agent. The Platform Web Services web application is deployed to the SASServer14 server instance by default.

SAS LOGON MANAGER AND CENTRAL AUTHENTICATION SERVICE

The SAS Logon Manager is a SAS web application that runs in a SAS Web Application Server, which is deployed in the SAS middle tier. SAS Logon Manager handles the authentication requests for various applications including the Platform Web Services web application. By default, the SAS Logon Manager web application is assigned to the SASServer1 SAS Web Application Server instance.

SAS Logon Manager uses the Central Authentication Service (CAS) that is available from Jasig to support a single sign-on authentication model. In a default 9.4 SAS deployment, the Central Authentication Service defers to the SAS Metadata Server to authenticate credentials. By default, the SAS Metadata Server is configured to use host-based authentication. That means by default when a client application logins through SAS Logon Manager, it uses host-based authentication of the SAS Metadata Server.

COMMUNICATION FLOW

Both the grid module and the agent communicate with Platform Web Services to perform its tasks with the underlying LSF grid.

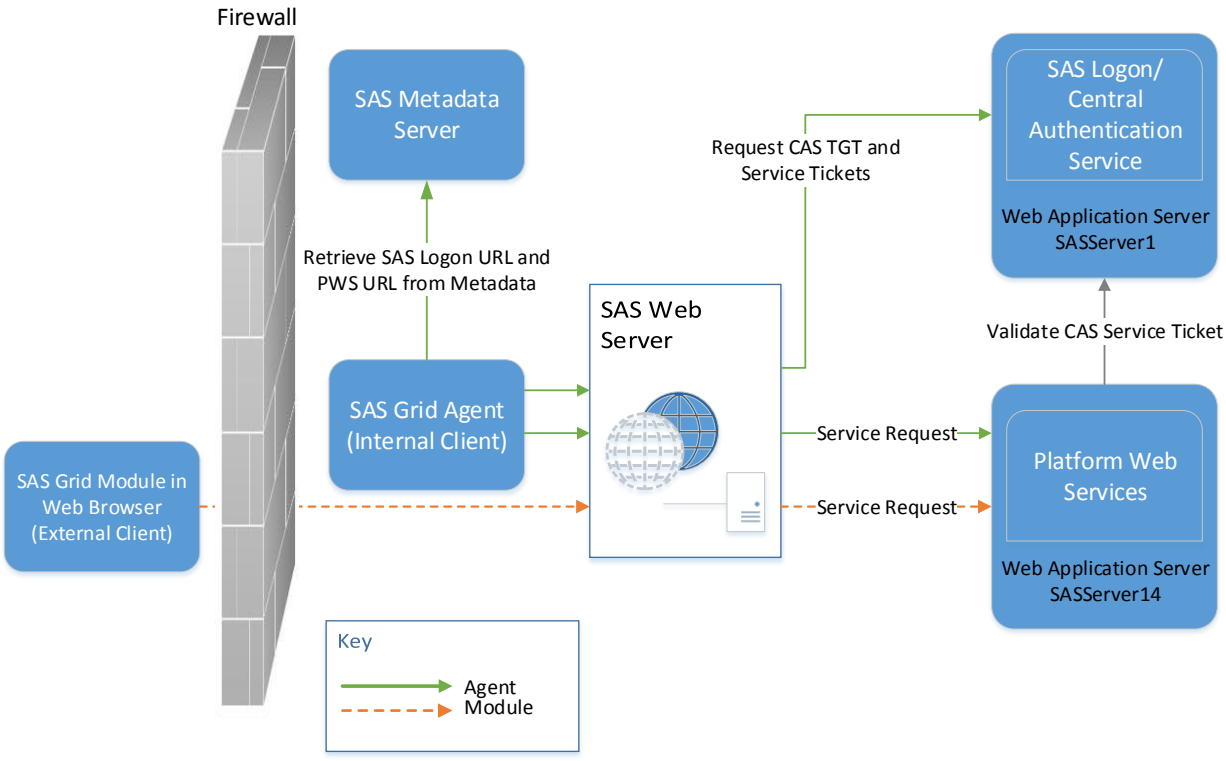

Figure 1: SAS Grid Module and Agent Overview

As shown in Figure 1, the module runs in a web browser, which will likely be running outside of any firewalls that exist in the environment. The module is considered to be an **external** client to the SAS middle tier. On the other hand, the grid agent runs on the same host as the SAS Environment Manager Server. Because it resides behind the firewall, it is considered to be an **internal** client to the middle tier. The agent and module interact with the following SAS Web Applications in the middle tier:

- Platform Web Services
- SAS Logon Manager (Central Authentication Service)

The agent and module both make service requests via HTTP to Platform Web Services REST endpoints. The incoming HTTP requests need to come from a client that has permission to access the requested resource. Permission is determined based on tickets issued by and checked by the Central Authentication Service. The client authenticates to the Central Authentication Service by providing a user ID and password to SAS Logon Manager in exchange for a Ticket Granting Ticket (TGT). This TGT is then used to obtain service tickets for accessing Platform Web Services. Once initial permission is granted for an identity and a service ticket is issued and validated, an HTTP session is established. Service tickets are then no longer needed for subsequent requests for the remainder of the valid session. The session is valid until the ticket granting ticket expires. See Session (Ticket Granting Ticket) Timeout. This minimizes the number of service tickets that must be generated and prevents the user having to provide their user ID and password for every request.

The entry point for checking whether the requesting client is authorized is the filter layer for the Platform Web Services web application.

FILTER CHAIN

The Platform Web Services web application has a series of filters that incoming HTTP requests have to pass through before reaching the actual application. The filters for the Platform Web Services web application are defined in its web.xml deployment descriptor:

```
<SASConfig>/Levn/Web/WebAppServer/SASServer14_1/sas_webapps/platform.web.
services.war/WEB-INF/web.xml
```

There are two request filters that are affected when using different host names:

- CAS authentication filter
- CAS validation filter

The Platform Web Services filters rely on the Jasig CAS client library (cas-client-core.jar) to perform the client authentication and CAS ticket validation. The entries in the web.xml file configure these filters and add them to the filter chain, but the Java implementation provided within the cas-client-core.jar is called to do the work. The cas-client-core.jar is packaged with the Platform Web Services web application WAR file.

The filter declaration contains the name of the Java class in the cas-client-core.jar that is called for that filter, as well as the input parameters to that class. This can be seen in the excerpts from the web.xml file in Figure 2 and Figure 3.

CAS Authentication Filter

The CAS authentication filter detects whether a client needs to be authenticated or not. Authentication is needed if there is no HTTP session established or if there is no CAS service ticket associated with the incoming request. If a client needs to be authenticated, the filter redirects the client to the SAS Logon page (CAS server) with an HTTP 302 response. When accessing the module in a browser, the browser handles this redirect to SAS Logon Manager automatically, where the user can provide their user ID and password. If the agent receives the redirect response, it will attempt to do a log on to the CAS server internally with the user ID and password it has from Environment Manager.

The module and agent handle a redirect response differently, which will be explained in later sections.

On the other hand, if there is an HTTP session already established or if a service ticket is provided with the request then the authentication filter will just proceed to the next filter in the chain, which is the CAS validation filter.

Figure 2 shows the CAS authentication filter, which is one of several filters that the Platform Web Services web application declares in its web.xml file.

```
<filter>
<filter-name>CAS Authentication Filter</filter-name>
<filter-class>
    org.jasig.cas.client.authentication.AuthenticationFilter
</filter-class>
<init-param>
    <param-name>casServerLoginUrl</param-name>
    <param-value>${CAS_SERVER_LOGIN_URL}</param-value>
</init-param>
<init-param>
    <param-name>serverName</param-name>
    <param-value>${PWS_SERVER_URL}</param-value>
</init-param>
</filter>
```

Figure 2: Authentication Filter

In Figure 2, we see that these initialization parameters are passed to the `org.jasig.cas.client.authentication.AuthenticationFilter` filter class:

- casServerLoginUrl
- serverName

The parameters obtain their values from:

- ${CAS_SERVER_LOGIN_URL}
- ${PWS_SERVER_URL}

These variables are defined in the setenv.sh file for the Platform Web Services web application. See the Filter Parameter Values in setenv.sh section.

CAS_SERVER_LOGIN_URL is used by the CAS authentication filter to redirect the client browser to the SAS Logon Manager (CAS server), where the user can log on and obtain a ticket granting ticket.

PWS_SERVER_URL is used by the CAS authentication filter to construct the service URL that the CAS service ticket will be requested for when the client gets redirected to the SAS Logon Manager. This service URL is also used after the service ticket is obtained to redirect the client back to the requested service along with the newly obtained service ticket.

The filters are executed in the order in which they are declared in the web.xml file. That leads us to the CAS validation filter, which is next in the filter chain after the CAS authentication filter.

CAS Validation Filter

Once a client request passes the CAS authentication filter, it proceeds to the CAS validation filter. The purpose of the validation filter is to validate that the service ticket matches the service URL for which the ticket was issued. The CAS client in Platform Web Services sends a validation request to the CAS server, containing both the service ticket from the incoming request and the service URL.

Figure 3 shows the declaration for the validation filter used by Platform Web Services.

```
<filter>
<filter-name>CAS Validation Filter</filter-name>
<filter-class>
    org.jasig.cas.client.validation.Cas20ProxyReceivingTicketValidationFilter
</filter-class>
<init-param>
    <param-name>casServerUrlPrefix</param-name>
    <param-value>${CAS_SERVER_URL_PREFIX}</param-value>
</init-param>
```

```
<init-param>
    <param-name>serverName</param-name>
    <param-value>${PWS_SERVER_URL}</param-value>
</init-param>
<init-param>
    <param-name>acceptAnyProxy</param-name>
    <param-value>true</param-value>
</init-param>
<init-param>
    <param-name>exceptionOnValidationFailure</param-name>
    <param-value>false</param-value>
</init-param>
<init-param>
    <param-name>redirectAfterValidation</param-name>
    <param-value>${CAS_VALIDATION_REDIRECT}</param-value>
</init-param>
<init-param>
    <param-name>encodeServiceUrl</param-name>
    <param-value>${CAS_ENCODE_SERVICE_URL}</param-value>
</init-param>
</filter>
```

Figure 3: Validation Filter

Here we see that these initialization parameters are passed to the `org.jasig.cas.client.validation.Cas20ProxyReceivingTicketValidationFilter` filter class:

- casServerUrlPrefix
- serverName

The parameters obtain their values from:

- ${CAS_SERVER_URL_PREFIX}
- ${PWS_SERVER_URL}

These variables are defined in the setenv.sh file for the Platform Web Services web application. See the Filter Parameter Values in setenv.sh section.

The CAS_SERVER_URL_PREFIX is used by the validation filter to connect to the CAS server to send the validation request.

The PWS_SERVER_URL is used to construct the service URL sent by the validation filter to the CAS server for validation against the service ticket.

The filter-mapping for both the authentication and validation filters use the ***/**** url-pattern, which indicates that all requests to the Platform Web Services web application are routed through these filters.

```
<filter-mapping>
    <filter-name>CAS Validation Filter</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>
```

Upon successful validation from the CAS server, the security assertion is saved for the session and an HTTP session is established. The effect of this is that the next time a request comes in to Platform Web Services, the authentication filter will see that there is a valid session, then move on to the validation filter. The validation filter will see there is no service ticket (since the session is used instead of a service ticket) and no validation request will be sent to the CAS server (since there is no service ticket with the request to validate).

FILTER PARAMETER VALUES IN SETENV.SH

The values that are used for the filters are provided in the web application's setenv.sh file:

```
<SASConfig>/Levn/Web/WebAppServer/SASServer14_1/bin/setenv.sh
```

The setenv.sh file contains the properties that are required by the authentication and validation filters:

- CAS_SERVER_LOGIN_URL="http://internal.example.com:<port>/SASLogon/login"
- CAS_SERVER_URL_PREFIX="http://internal.example.com:<port>/SASLogon"
- PWS_SERVER_URL="http://internal.example.com:<port>"

Note: By default, the **internal** host name will be set in these values, since this is the host name that the SAS deployment knows about.

Changing Filter Parameter Values in setenv.sh

When the SAS Web Application Server starts up, the properties in CATALINA_OPTS in the setenv.sh file are passed to the Java application as JVM properties.

This is why modifications to the setenv.sh require a restart of the SAS Web Application Server in order for the new values to take effect. To restart the SAS Web Application Server that contains Platform Web Services, execute the following command:

```
<SASConfig>/Levn/Web/WebAppServer/SASServer14_1/bin/tcruntime-ctl.sh
restart
```

MODULE AUTHENTICATION

PROBLEMS

A network topology might be present that prevents host names that are used by SAS from being used by external clients (those outside the firewall). The grid module, being an external client (since it runs in the web browser), will have trouble accessing the SAS middle tier when trying to make a service request to Platform Web Services.

The grid module client needs to be authenticated by the Platform Web Services filters. By default, the setenv.sh configuration file that PWS uses, contains the internal host name for the URL values that are used for redirecting the client browser to SAS Logon Manager and for creating the service URL for which a service ticket is requested. This results in several problems.

Problem 1:

The redirect from the authentication filter in Platform Web Services to SAS Logon Manager for the user to log on will not work, since the internal host name in the CAS_SERVER_LOGIN_URL is not accessible externally.

Problem 2:

Even if the CAS_SERVER_LOGIN_URL were changed to use the external host name, if SAS Logon Manager does not know about the external host name, it will fail to provide a service ticket for the service requested by the grid module client.

Problem 3:

The service ticket granted by the CAS server will be for the service endpoint based on the internal host name that is in PWS_SERVER_URL. The CAS server will also use that internal URL to redirect the browser to the Platform Web Services service endpoint, but this will not work, since the internal host name is not known externally in this case.

SOLUTION

The external host name URL must be configured throughout the components that the grid module interacts with, because the module can only access the components through the external host name.

Configure Platform Web Services to Use External Host Name

In the following file:

```
<SASConfig>/Levn/Web/WebAppServer/SASServer14_1/bin/setenv.sh
```

Change from using the internal host name to the external host name based URL, for the following two properties needed by the Platform Web Services request filters:

- CAS_SERVER_LOGIN_URL="http://**external**.example.com:<port>/SASLogon/login"
- PWS_SERVER_URL=http://**external**.example.com:<port>

See, the Changing Filter Parameter Values in setenv.sh section.

Note: CAS_SERVER_URL_PREFIX is used to communicate only between internal components (the CAS client in Platform Web Services and the CAS server), so this value should not need to be changed.

After this change, the browser that is running the module should now be automatically redirected to the SAS Logon page, where the user can enter their user ID and password to obtain the ticket granting ticket.

Add the External Host Name to Metadata

Add the external host name to metadata so that the CAS server will be able to issue service tickets based on that host name. The CAS server will not issue service tickets for a service that is based on a host name that it does not know about.

1. Log on to SAS Management Console
2. On the **Plug-ins** tab, select **Application Management →Configuration Manager → SAS Application Infrastructure**
3. Right-click **Platform Web Svc for SAS**, and select **Properties**.
4. Click the **External Connection** tab.
5. Clear the **Use internal connection information** check box.
6. Specify the external host name in the **Host Name** field.

The following figure shows the external connection information required for the Platform Web Services web application.

Figure 4: External Connection Properties for Platform Web Services

The internal and external connection properties will now be different, so the following step is required:

7. Add the following JVM option to the web application server setenv.sh JVM_OPTS:

```
-Dsas.retry.internal.url=true
```

For Platform Web Services modify:

```
<SASConfig>/Levn/Web/WebAppServer/SASServer14_1/bin/setenv.sh
```

8. Repeat the above steps 1-6 for the SAS Grid Manager Module and steps 1-7 for the SAS Logon Manager. In step 3, use the following SAS Web Applications selections:
   a. **Environment Mgr Mid-Tier →Grid Mgr Module Env Mgr**
   b. **Logon Manager**

   In step 7, for SAS Logon Manager, modify:

```
<SASConfig>/Levn/Web/WebAppServer/SASServer1_1/bin/setenv.sh
```

9. Restart the SAS Web Application Servers using these commands:

```
<SASConfig>/Levn/Web/WebAppServer/SASServer1_1/bin/tcruntime-ctl.sh
restart
<SASConfig>/Levn/Web/WebAppServer/SASServer14_1/bin/tcruntime-ctl.sh
restart
```

After you make these changes, and after successful log on to SAS Logon Manager, the module running in the web browser should be able to obtain the service ticket and be successfully redirected to the Platform Web Services service endpoint.

AGENT AUTHENTICATION

The agent is an internal client and should not have a problem communicating with the SAS middle tier using the internal host name. However, the agent is affected by some of the configuration changes made to Platform Web Services to solve the problems encountered by the module.

The agent does not follow redirects from the authentication filter. Instead, when it receives a redirect (HTTP 302), it just uses the SAS Logon URL from metadata if it decides it has to obtain a new ticket granting ticket from the CAS server. Therefore, the agent is not affected by the change to the CAS_SERVER_LOGIN_URL value in the setenv.sh file for Platform Web Services. This is mentioned because some network topologies prevent the external host name from being known internally.

The agent differs from the module in that it uses the base URL it gets directly from metadata for Platform Web Services. By default, the agent uses this internal base URL for acquiring service tickets and making REST calls to Platform Web Services.

PROBLEMS

Problem 1:

Due to the changes needed for the module to communicate with Platform Web Services, the PWS_SERVER_URL value is now based on the external host name. However, at this time the agent uses the internal URL to acquire the CAS service ticket and make REST calls to Platform Web Services.

All requests to Platform Web Services, whether originating from the module or the client, have to pass through its filters. Remember that PWS_SERVER_URL is used as input to the CAS validation filter. The problem occurs when the request from the agent to Platform Web Services gets to the validation filter. The PWS_SERVER_URL is used to construct the service URL that is sent to the CAS server for validating the service ticket that is also sent.

On the CAS server, since the service URL will now be based on the external URL but the service ticket is for the internal URL, the CAS server will reject the validation. In other words, the CAS server will see that the CAS service ticket was obtained for a different URL.

This problem will result in no metrics collected or reported, and graphs in SAS Environment Manager for SAS Grid will be missing.

The agent log will show the failed validation as an "HTTP 403 Forbidden" error.

Problem 2:

Changing the **pws.ticket.url** agent property in SAS Environment Manager to the external host name will not work if the steps for the module in Add the External Host Name to Metadata have not been completed.

In the agent, a ticket granting ticket will most likely be obtained successfully (with valid credentials) because it uses the internal CAS Logon URL directly from metadata. But it might have a problem getting the service ticket for Platform Web Services. The CAS server will not issue a service ticket for a requested service if the service is based on a host name that it does not know about. If a service ticket cannot be issued due to an unknown host name, the CAS server will respond with a "400 Bad Request" error. The agent will not be able to obtain the grid metrics from Platform Web Services, and Grid Manager will appear as red (unavailable) in SAS Environment Manager.

SOLUTION

First, add the external host name to metadata if not done so as described in Add the External Host Name to Metadata.

Configure the Agent to Use the External Host Name for Service Ticket Acquisition

The CAS service ticket obtained by the agent must match the external URL specified in PWS_SERVER_URL in the setenv.sh file for Platform Web Services. The URL used to obtain the CAS service ticket by the agent can be overridden by specifying it in the **pws.ticket.url** property field in SAS Environment Manager.

1. In SAS Environment Manager, select **Resources → Browse**.
2. Select **Servers**.
3. Select the **SAS Grid Manager** entry.
4. Select **Tools Menu → Configure Server**.
5. Specify the external URL in the **pws.ticket.url** field.

For example, if the PWS_SERVER_URL variable in the setenv.sh contains:

```
PWS_SERVER_URL="http://host.external.com:7980"
```

Then enter the following in the **pws.ticket.url** field for the agent in SAS Environment Manager:

```
http://host.external.com:7980/PlatformWebServices
```

The agent uses the credentials provided by the user in SAS Environment Manager. Figure 5 shows the **Configuration Properties** fields for SAS Grid Manager in SAS Environment Manager. The user ID and password can also be added here.

The agent sends the user ID and password supplied on this page to obtain a ticket granting ticket from the CAS server. The agent performs this action only on the initial request or if the ticket granting ticket is expired.

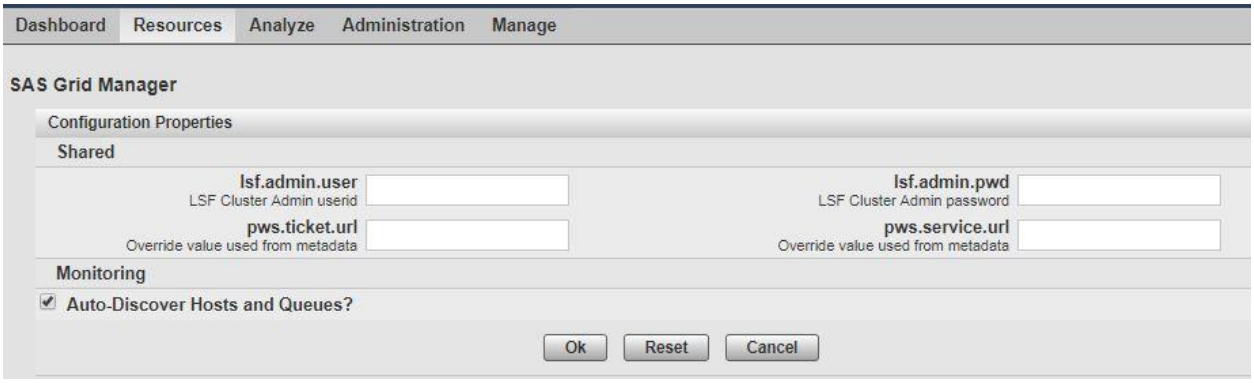

Figure 5: SAS Grid Manager Agent Input Fields

This change allows the agent to get the metrics required for the graph generation from Platform Web Services.

Note: The **pws.service.url** field is used to override the URL used for contacting the Platform Web Services REST endpoint. By default, the internal value from metadata is used.

SESSION (TICKET GRANTING TICKET) TIMEOUT

The default time for the ticket granting ticket to expire is 12 hours. The agent and module use session cookies instead of service tickets once the HTTP session is established (after successfully validating the initial service ticket). If a ticket granting ticket expires or a CAS logout is done, the HTTP sessions that are associated with that ticket granting ticket are invalidated. From then on, attempting to use that session would receive an HTTP 302 response code. If this response is received, the agent and module need to re-authenticate in order to acquire a new ticket granting ticket. The default of 12 hours can be changed by specifying a new time out using this JVM option for SAS Logon Manager.

```
-Dsas.tgt.expiration.period=inverval-in-milliseconds
```

The option is set in this file.

```
<SASConfig>/Levn/Web/WebAppServer/SASServer1_1/bin/setenv.sh
```

DEBUG LOGGING

Even after following all of the instructions in this paper you might still find the need to turn on debugging. Turning on debug level logging in the following components might be helpful. You must restart SASServer1 and SASServer14 web application servers in order to pick up the debug settings.

SAS LOGON MANAGER

Add the following to the `<SASConfig>/Levn/Web/Common/LogConfig/SASLogon-log4j.xml` file.

```
<category additivity="false" name="org.jasig.cas">
    <priority value="DEBUG"/>
    <appender-ref ref="SAS_CONSOLE"/>
    <appender-ref ref="SAS_FILE"/>
</category>
<category additivity="false" name="com.sas.svcs.security.authentication">
    <priority value="DEBUG"/>
    <appender-ref ref="SAS_CONSOLE"/>
    <appender-ref ref="SAS_FILE"/>
</category>
```

PLATFORM WEB SERVICES

Add the following to the `<SASConfig>/Levn/Web/WebAppServer/SASServer14_1/sas_webapps/platform.web.services.war/WEB-INF/classes/log4j.xml` file.

```
<category additivity="false" name="org.jasig.cas.client">
    <priority value="DEBUG"/>
    <appender-ref ref="PWS_FILE_APPEND"/>
</category>
```

In the same file, change the priority value for "`com.platform`" to "`DEBUG`".

GRID MODULE

Add the following to the `<SASConfig>/Levn/Web/Common/LogConfig/SASGridManager-log4j.xml` file.

```
<category additivity="false" name="org.jasig.cas">
    <priority value="DEBUG"/>
    <appender-ref ref="SAS_CONSOLE"/>
    <appender-ref ref="SAS_FILE"/>
</category>
<category additivity="false" name="org.springframework.security">
    <priority value="DEBUG"/>
    <appender-ref ref="SAS_CONSOLE"/>
    <appender-ref ref="SAS_FILE"/>
</category>
```

GRID AGENT

Add the following to the `<SASConfig>/Levn/Web/SASEnvironmentManager/agent-5.8.0-EE/conf/agent.properties` file.

```
log4j.logger.com.sas.grid.hyperic=DEBUG
log4j.logger.com.sas.svcs.security.authentication.client=DEBUG
```

CONCLUSION

This paper dives into the ins and outs of internal and external host names. It clarifies where the components live that are involved with monitoring, managing and configuring the SAS Grid in SAS Environment Manager. When you understand the communication flow between these components, you can see that configuration allows for network topologies that require separate host names.

ACKNOWLEDGMENTS

Thanks to Scott Parrish and Casey Hadden for their input.

RECOMMENDED READING

SAS Institute, Inc. 2016. *SAS 9.4 Intelligence Platform: Middle-Tier Administration Guide*. 4th ed. Cary, NC: SAS Institute Inc. Available: http://go.documentation.sas.com/?cdcId=bicdc&cdcVersion=9.4&docsetId=bimtag&docsetTarget=titlepage.htm&locale=en

“Configuring the Jasig CAS Client for Java in the web.xml” Apereo. Available https://wiki.jasig.org/display/CASC/Configuring+the+Jasig+CAS+Client+for+Java+in+the+web.xml. Accessed on February 14, 2018.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Paula Kavanagh
100 SAS Campus Drive
Cary, NC 27513
Paula.Kavanagh@sas.com
http://www.sas.com