SAS3720-2016

Tips and Best Practices for Configuring Integrated Windows Authentication

Mike Roda, SAS Institute Inc., Cary, NC

ABSTRACT

Since it makes login transparent and does not send passwords over the wire, Integrated Windows Authentication (IWA) is both extremely convenient for end users and highly secure. However, for administrators, it is not easy to set up and rarely successful on the first attempt, so being able to effectively troubleshoot is imperative. In this paper, we take a step-by-step approach to configuring IWA for the SAS® middle tier, explain how and where to get useful debugging output, and share our hard-earned knowledge base of known problems and deciphered error messages.

INTRODUCTION

Integrated Windows Authentication has been a cornerstone of SAS® enterprise security integration, with support from the initial launch of SAS® 9.4 (and earlier versions). The first maintenance release of SAS 9.4 added support for Kerberos delegation, allowing users to launch workspace servers under the identity of the user that authenticated to the middle tier. Novel features such as Fallback Authentication were added and could be combined with other forms of authentication. Through the experience gained from developing and testing of these IWA features in SAS R&D, as well as field deployments and technical support tracks, this author maintained an informal knowledge base of common error messages and solutions. Whenever a new and unexpected error arose, experts would be consulted and eventually a solution found and added to the knowledge base. Eventually this grew into a vital source of information for technical support and installers in the field. Along the way several best practices were developed as well.

This paper is an attempt to share and publish this important information to the wider audience. It is intended as a practical guide and reference for SAS administrators or anyone responsible for configuring and troubleshooting SAS middle tier deployments for IWA. Experience obtained from troubleshooting countless deployments, both internally in SAS R&D as well as in the field, has gone into this paper.

What this paper does not do is provide a technical overview of Kerberos. It is assumed the reader already has a basic understanding of the technology, but it is not required. Those looking for an overview should see *Kerberos and SAS 9.4: A Three-Headed Solution for Authentication* by Stuart Rogers. A link is provided in the references at the end of this paper.

Before delving into the details, let’s consider the key components of IWA. There is the Active Directory Controller, the Kerberos Key Distribution Center (KDC) on the Domain Controller, one or many client PCs with a web browser and/or SAS client software, the middle-tier SAS® Web Application Servers, and the SAS workspace servers.

TIPS FOR KERBEROS SETUP

Plenty of things need to be done before configuring the SAS middle tier for IWA. Some of these are outside the role of the typical SAS administrator but need to be understood so that they can be explained to the responsible parties.

First of all, a service account must be created in Microsoft Active Directory with the service principal name (SPN) that the SAS middle tier will use. It cannot be the Computer object that the SAS middle tier is running on; it must be a separate account. From this account, a Kerberos keytab file is created. The keytab contains the cryptographic keys required by the SAS middle tier to decrypt Kerberos tickets and authenticate users. Creating the service account is outside the scope of this paper, but in the following sections we will look at how to create and verify the keytab and SPNs.

We will also discuss the Unlimited Strength Java Cryptography Extension that adds strong cryptographic ciphers needed in many environments.

MICROSOFT MANAGEMENT CONSOLE

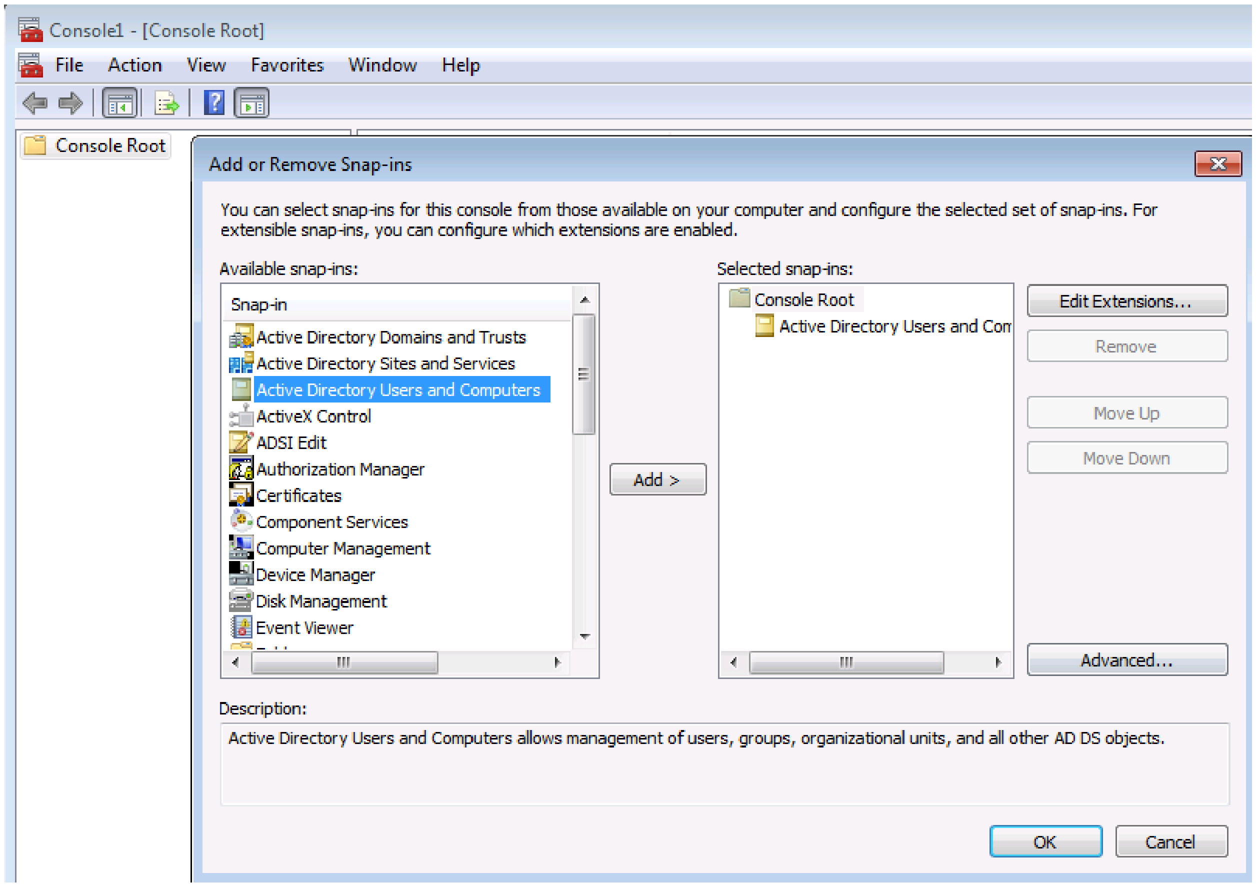
Although it is almost always the case that a different organization in the company is responsible for creating the service account, it is still extremely valuable to have visibility into Active Directory to check the account settings on the service account that the keytab is generated from. Microsoft Management Console (MMC) is a Windows component that can be used to query the Active Directory LDAP and view detailed account information.

Windows 7 and later operating systems should already have MMC but require an additional snap-in that is part of the Remote Server Administration Tools. This can be downloaded from Microsoft at the following address:

http://www.microsoft.com/en-us/download/details.aspx?id=7887

Follow the instructions to install the software. When installation is finished, launch **MMC** from the Windows start menu and select **File > Add/Remove Snap-in**. From the list of snap-ins on the left, select **Active Directory Users and Computers**, click the **Add** button, and click **OK**. This is shown in Display 2.

After that, MMC will display the Active Directory LDAP tree on the left. Drill down into the tree a bit and then select **View** and **Advanced Features** from the top menu. From here you can drill down to the service account, using the Filter icon at the top if necessary to limit the results, and view all the account settings and attributes. Many of these are referred to later in this document.



You can select snap-ins for this console from those available on your computer and configure the selected set of snap-ins. For extensible snap-ins, you can configure which extensions are enabled.

Active Directory Users and Computers allows management of users, groups, organizational units, and all other AD DS objects.

Display 1. Microsoft Management Console Snap-ins

CREATE THE KERBEROS KEYTAB

Creating a good keytab file is usually 90% of the battle. For those challenged with creating the keytab, or helping an administrator to create the keytab, two methods with examples are discussed below.

The Microsoft tool ktpass should be located in C:\Windows\System32 on the domain controller. This is a dangerous little tool that has various side effects in Active Directory, even if you use Ctrl-C to stop it before finishing the prompts. Nevertheless, ktpass is widely used, and it will automatically output the correct key version number (KVNO) associated with the account in Active Directory. Output 1 shows an example of how to use ktpass to create the keytab.

```
C:\>ktpass.exe -princ HTTP/jdtvm01.na.sas.com@NA.SAS.COM -mapUser
CaryNT\jdtvm01-HTTP -pass "*"
               -pType KRB5_NT_PRINCIPAL -out jdtvm01-HTTP.keytab -crypto
All

Targeting domain controller: shqnadc07.na.sas.com
Using legacy password setting method
Successfully mapped HTTP/jdtvm01.na.sas.com@NA.SAS.COM to jdtvm01-HTTP.
Key created.
Key created.
Key created.
Key created.
Key created.
Output keytab to C:\jdtvm01-HTTP.keytab:
Keytab version: 0x502
keysize 68 HTTP/jdtvm01.na.sas.com@NA.SAS.COM ptype 1 (KRB5_NT_PRINCIPAL)
vno 1 etype 0x1 (DES-CBC-CRC) keylength 8
keysize 68 HTTP/jdtvm01.na.sas.com@NA.SAS.COM ptype 1 (KRB5_NT_PRINCIPAL)
vno 1 etype 0x3 (DES-CBC-MD5) keylength 8
keysize 76 HTTP/jdtvm01.na.sas.com@NA.SAS.COM ptype 1 (KRB5_NT_PRINCIPAL)
vno 1 etype 0x17 (RC4-HMAC) keylength 16
keysize 92 HTTP/jdtvm01.na.sas.com@NA.SAS.COM ptype 1 (KRB5_NT_PRINCIPAL)
vno 1 etype 0x12 (AES256-SHA1) keylength 32
keysize 76 HTTP/jdtvm01.na.sas.com@NA.SAS.COM ptype 1 (KRB5_NT_PRINCIPAL)
vno 1 etype 0x11 (AES128-SHA1) keylength 16
```

Output 1. Output from Using ktpass to Generate a Keytab File

Enter any password you like at the prompt. It doesn't have to be the password of the account in Active Directory, but it must meet certain minimum criteria. Note that there are several side effects to using ktpass. First of all, it will change the password on the account in Active Directory to some random value and increment the key version number (we'll discuss key version numbers more later). If you specify a principal name starting with HTTP, it will update the User Logon Name on the Account tab and set the userPrincipalName attribute to this value, and it will add this to the servicePrincipalName attribute if not already there.

The other option is to use ktutil from the MIT Kerberos project. This is the ideal choice for generating keytabs because it does not change any attributes in Active Directory and you have full control over which encryption types are included in the file. Output 2 shows an example of how to use ktutil to create a keytab with key version number 1.

```
$ ktutil
ktutil:  addent -password -p HTTP/jdtvm01.na.sas.com@NA.SAS.COM -k 1 -e
arcfour-hmac
Password for HTTP/jdtvm01.na.sas.com@NA.SAS.COM :
ktutil:  addent -password -p HTTP/jdtvm01.na.sas.com@NA.SAS.COM -k 1 -e
aes128-cts-hmac-sha1-96
Password for HTTP/jdtvm01.na.sas.com@NA.SAS.COM :
```

```
ktutil:  addent -password -p HTTP/jdtvm01.na.sas.com@NA.SAS.COM -k 1 -e
aes256-cts-hmac-sha1-96
Password for HTTP/jdtvm01.na.sas.com@NA.SAS.COM :
ktutil:  wkt jdtvm01-HTTP.keytab
ktutil:  quit
```

Output 2. Output from Using ktutil to Create a Keytab File

Unlike ktpass, with ktutil you must know and specify the existing password of the account in Active Directory.

LIST KEYTAB CONTENTS

Although the previous section discussed how to create the keytab, the reality is that SAS administrators must usually rely on another department in the company to generate the keytab. Therefore, for troubleshooting purposes it is often necessary to list the contents of the keytab. Although simplistic, the ktab program is guaranteed to be available because it is shipped with the Java Runtime Environment. It shows the principal name and KVNO for each entry in the keytab but unfortunately does not show the encryption types. Output 3 shows an example of how to use ktab to list the contents of the keytab.

```
C:\>ktab.exe -l -k jdtvm01-HTTP.keytab

Keytab name: jdtvm01-HTTP.keytab
KVNO Principal
---- ----------------------------------
   1 HTTP/jdtvm01.na.sas.com@NA.SAS.COM
   1 HTTP/jdtvm01.na.sas.com@NA.SAS.COM
   1 HTTP/jdtvm01.na.sas.com@NA.SAS.COM
   1 HTTP/JDTVM01@NA.SAS.COM
   1 HTTP/JDTVM01@NA.SAS.COM
   1 HTTP/JDTVM01@NA.SAS.COM
```

Output 3. Output from Using ktab to List the Contents of a Keytab

Once again, when available, the ktutil program from the MIT Kerberos tools is preferred because it shows the encryption types. Output 4 shows an example of how to use ktutil to list the contents of the keytab.

```
bash-4.1$ ktutil
ktutil:  rkt jdtvm01-HTTP.keytab
ktutil:  list -e
slot KVNO   Principal
---- ----   ---------------------------------------------------------------
   1    1   HTTP/jdtvm01.na.sas.com@NA.SAS.COM (aes256-cts-hmac-sha1-96)
   2    1   HTTP/jdtvm01.na.sas.com@NA.SAS.COM (aes128-cts-hmac-sha1-96)
   3    1   HTTP/jdtvm01.na.sas.com@NA.SAS.COM (arcfour-hmac)
   4    1   HTTP/JDTVM01@NA.SAS.COM (aes256-cts-hmac-sha1-96)
   5    1   HTTP/JDTVM01@NA.SAS.COM (aes128-cts-hmac-sha1-96)
   6    1   HTTP/JDTVM01@NA.SAS.COM (arcfour-hmac)
```

Output 4. Output from Using ktutil to List the Contents of the Keytab

KEY VERSION NUMBERS

The service account in Active Directory will have a key version number (KVNO). You can view this by looking at the msDS-KeyVersionNumber attribute in Microsoft Management Console. This is a constructed attribute, so you have to set the filter to show those. It is important that the KVNO in the keytab matches what is in Active Directory. When the client (for example, Internet Explorer) gets a Kerberos service ticket from the domain controller to call the middle tier, the ticket will indicate the KVNO of the key that was used to encrypt it. The KVNO in the ticket from the client needs to match the KVNO in the keytab, which needs to match the msDS-KeyVersionNumber attribute in Active Directory.

The KVNO in Active Directory is incremented each time the password is changed. This includes each time you run ktpass, since it changes the password. If you are using ktpass, the KVNO in the keytab should be correct. But when using ktutil or ktab, you must specify the KVNO. Alternatively, you can use a KVNO of 0; Java is supposed to ignore the KVNO of the ticket sent by the client if it is 0. We have had some success using this method when unsure of the KVNO.

USING KINIT TO TEST THE KEYTAB

The kinit program that is shipped with the Java Runtime Environment can be used to do a sanity check on the keytab file. This should always be done before attempting to configure the middle tier. You need to provide it the path to the keytab file and specify the user principal name (not the SPN) of the account in Active Directory:

Output 5 shows an example of how to use kinit to test the keytab.

```
C:\> kinit -k –t jdtvm01-HTTP.keytab HTTP/jdtvm01.na.sas.com

New ticket is stored in cache file C:\Users\miroda.CARYNT\krb5cc_miroda
```

Output 5. Output from Using kinit to Test the Keytab

“Pre-authentication information was invalid” Error

Depending on what version of Microsoft Server 2008 the account was created from, AES encryption types might not work. If you are using a keytab with AES encryption types, you might see the error shown in Output 6.

```
Exception: krb_error 24 Pre-authentication information was invalid (24)
Pre-authentication information was invalid
KrbException: Pre-authentication information was invalid (24)
       at sun.security.krb5.KrbAsRep.<init>(KrbAsRep.java:76)
       at sun.security.krb5.KrbAsReqBuilder.send(KrbAsReqBuilder.java:319)
       at
sun.security.krb5.KrbAsReqBuilder.action(KrbAsReqBuilder.java:364)
       at sun.security.krb5.internal.tools.Kinit.<init>(Kinit.java:221)
       at sun.security.krb5.internal.tools.Kinit.main(Kinit.java:113)
Caused by: KrbException: Identifier doesn't match expected value (906)
       at sun.security.krb5.internal.KDCRep.init(KDCRep.java:143)
       at sun.security.krb5.internal.ASRep.init(ASRep.java:65)
       at sun.security.krb5.internal.ASRep.<init>(ASRep.java:60)
       at sun.security.krb5.KrbAsRep.<init>(KrbAsRep.java:60)
       ... 4 more
```

Output 6. Output Showing a “Pre-Authentication information was invalid” Error

To test this theory, try generating a keytab with only arcfour-hmac encryption. If that works, reset the password on the account in Active Directory once or twice. For some reason this resolves the issue. Note you will need to re-create the keytab if this resulted in a new password or if the version number changed.

VERIFY SPN

If you haven't already done so, verify that the service principal name (SPN) is registered in Active Directory using the setspn command. It's a good practice to have both the short and fully qualified hostnames. Also include any aliases for the host that clients might be using.

Output 7 shows an example of how to verify the SPN using setspn.

```
C:\>setspn -F -Q HTTP/jdtvm01.na.sas.com

Checking forest DC=SAS,DC=com
CN=jdtvm01-HTTP,OU=Service
Accounts,OU=RDC,OU=US,OU=Servers,DC=na,DC=SAS,DC=com
```

```
        HTTP/jdtvm01.na.sas.com
        HTTP/JDTVM01

Existing SPN found!
```

Output 7. Output from the setspn Statement

Note: The SPN should only come back to a single account.

KERBEROS CONFIGURATION FILE

A Kerberos configuration file, usually named krb5.conf on UNIX or krb5.ini on Windows, is needed at a minimum on the middle-tier machine running the SASServer1_1 instance, and all machines running SASServer instances when Kerberos delegation is used.

The file typically is located in C:\windows on Windows or the /etc directory on UNIX and might already exist there. If not, use the following sample and modify the KDC host name and realm:

```
[libdefaults]
default_realm = EXAMPLE.COM
forwardable=true

[realms]
EXAMPLE.COM = {
  kdc = domain-controller.com
}

[domain_realm]
example.com= EXAMPLE.COM
.example.com= EXAMPLE.COM
```

The middle tier will be configured to point to this file later.

UNLIMITED STRENGTH JAVA CRYPTOGRAPHY EXTENSION

The Unlimited Strength Java Cryptography Extension (JCE) provides strong cryptographic ciphers that are not shipped with the SAS Private Java Runtime Environment by default due to export restrictions. The Unlimited Strength JCE must be downloaded from Oracle and manually installed if either of the following is true:

- The domain controller will be issuing tickets encrypted with AES 256.
- Users' delegated Kerberos credentials will be used to launch workspace servers.

If you are unsure, just install it to be safe. The files need to be extracted into the SASHome/SASPrivateJavaRuntimeEnvironment/9.4/jre/lib/security directory. At the time of this writing, SAS 9.4 is shipping with Java 7, and you should see three files in that directory with a timestamp of May 31, 2011, as shown in Output 8.

```
Directory of C:\Program
Files\SASHome\SASPrivateJavaRuntimeEnvironment\9.4\jre\lib\security

04/08/2014  09:23 AM    <DIR>          .
04/08/2014  09:23 AM    <DIR>          ..
01/21/2014  12:11 PM             2,770 blacklist
01/21/2014  12:11 PM            82,586 cacerts
01/21/2014  12:11 PM             2,593 java.policy
01/21/2014  12:11 PM            17,985 java.security
01/21/2014  12:11 PM               158 javafx.policy
01/21/2014  12:11 PM                98 javaws.policy
05/31/2011  02:33 PM             2,500 local_policy.jar
```

```
05/31/2011  02:33 PM             7,289 README.txt
01/21/2014  12:11 PM                 0 trusted.libraries
05/31/2011  02:33 PM             2,487 US_export_policy.jar
```

Output 8. Output Showing the Files That Should Be in Place after Installing the Unlimited Strength JCE

BEST PRACTICES FOR MIDDLE TIER CONFIGURATION

With everything set up in Active Directory and a tested keytab file in hand, it's time to configure the middle tier. This can be broken down into the following steps:

1. Configure Java Virtual Machine (JVM) options related to Kerberos.
2. Configure JAAS, telling the Java login module where to find the keytab and what principal name to use. Optionally, turn on Kerberos delegation.
3. Configure SAS Web Application Server for authentication.
4. Enable debugging output for troubleshooting.

Except where otherwise noted, the configuration described in the following sections should be repeated for each SASServer instance. The instructions are for SAS 9.4 maintenance release 3 or later. For earlier releases, refer the *SAS 9.4 Intelligence Platform: Middle-Tier Administration Guide, Third Edition*.

JVM OPTIONS

SAS Web Application Server will look for the Kerberos configuration file created earlier in the server /conf directory with the name krb5.conf. If the file is located elsewhere (for example, in a centralized or more standard location such as /etc), the java.security.krb5.conf JVM option needs to be set. This is also a good time to enable the JVM options for Kerberos debugging output.

On Windows, edit the wrapper.conf file. Find the wrapper.java.additional lines and add the following, using the next available line numbers and changing the path to the file as necessary:

```
wrapper.java.additional.51=-Djava.security.krb5.conf=/etc/krb5.conf
wrapper.java.additional.52=-Dsun.security.krb5.debug=true
wrapper.java.additional.53=-Dsun.security.jgss.debug=true
```

Also find this line and change the loglevel to DEBUG:

```
wrapper.logfile.loglevel=DEBUG
```

On UNIX, add the parameters to the JVM_OPTS in setenv.sh:

```
-Djava.security.krb5.conf=/etc/krb5.conf

-Dsun.security.krb5.debug=true
-Dsun.security.jgss.debug=true
```

JAAS CONFIGURATION

The Java Authentication and Authorization Service (JAAS) plays two roles. It configures the Krb5LoginModule provided by the JVM with important settings such as the file location and the principal entry to use from the keytab. It also configures the SAS OMILoginModule for Kerberos delegation, if needed.

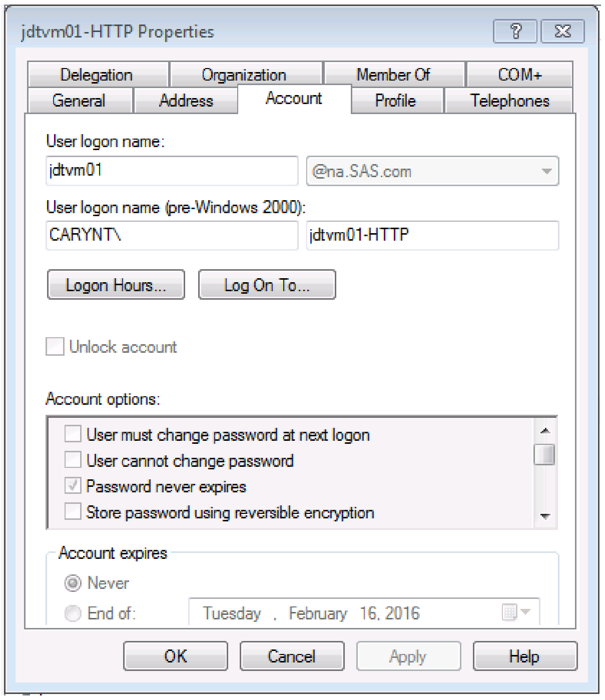
The jaas.config file is located in the server /conf directory. The following example needs to be added on SASServer1_1 at a minimum, and all SASServers when Kerberos delegation is required:

```
com.sun.security.jgss.krb5.accept {
   com.sun.security.auth.module.Krb5LoginModule required
   doNotPrompt=true
```

```
      isInitiator=false
      principal="HTTP/hostname.example.com@EXAMPLE.COM"
      useKeyTab=true
      keyTab="C:/path-to-hostname.keytab"
      storeKey=true;
   };
```

Some important gotchas:

- The principal specified should match the User logon name of the service account in Active Directory (also the userPrincipalName attribute on the account). This might be the same as the service principal name, especially if ktpass was used to generate the keytab, but is commonly not the same. Display 2 is an example where the principal would be jdtvm01@NA.SAS.COM.
- The realm should be all uppercase, regardless of how it is shown in MMC.
- The path to the keytab should use forward slashes, even on Windows systems.



User logon name: jdtvm01 @na.SAS.com

User logon name (pre-Windows 2000): CARYNT\ jdtvm01-HTTP

☐ Unlock account

Account options:

☐ User must change password at next logon
☐ User cannot change password
☑ Password never expires
☐ Store password using reversible encryption

Account expires

(● Never)
○ End of: Tuesday , February 16, 2016

Display 2. User Logon Name in Microsoft Management Console

If Kerberos delegation is required, the following two bolded lines need to be added to the existing PFS entry in the file:

```
   PFS {
     com.sas.services.security.login.OMILoginModule  required
          "host"="jdtvm01.na.SAS.com"
          "port"="8561"
          "repository"="Foundation"
          "domain"="DefaultAuth"
          "trusteduser"="sastrust@saspw"
          "trustedpw"="{sas002}1D5793391C1104E20E3CF4CD2A793E2B"
          "aliasdomain"="DefaultAuth"
          "idpropagation"="sspi"
          "sspisecuritypackagelist"="KERBEROS"
```

```
        "debug"="false";
};
```

AUTHENTICATION

Now we need to configure SAS Web Application Server to authenticate users via Kerberos. There are two parts involved, the authenticator and the realm. Both parts are simplified by software that SAS ships with the servlet container based on Apache Tomcat.

Tomcat comes with a built-in authenticator for the Simple and Protected GSSAPI Negotiation Mechanism (SPNEGO) that handshakes with the user agent to process Kerberos tickets. SAS extends that support to provide Fallback Authentication, allowing SAS FORM authentication to kick in if SPNEGO fails. This should be configured as a best practice. The SasFallbackAuthenticatorValve class, which you configure in the context.xml file located in SASServer1_1/conf, delegates to the Tomcat authenticator by setting authmethod="SPNEGO":

```
<Valve
    className=
      "com.sas.vfabrictcsvr.authenticator.SasFallbackAuthenticatorValve"
    authMethod="SPNEGO" />
```

Placing this in the context.xml file will help with debugging output (discussed later). Later, after everything is working, move this to the SASLogon.xml file under conf/Catalina/localhost/.

A change must also be made to the web.xml file for the SASLogon web application, so that when Kerberos fails and a 401 status code is return by the valve, an error page is returned to the browser that acts as a mechanism to trigger fallback authentication. The file is located under SASServer1_1/sas_webapps/sas.svcs.logon.war/WEB-INF. Uncomment the following section near the bottom of the file:

```
<!--
    <error-page>
        <error-code>401</error-code>
        <location>/WEB-INF/view/jsp/default/ui/401Fallback.jsp</location>
    </error-page>
-->
```

As a best practice, when editing files inside the SAS webapps in the configuration, it is recommended to make the same change in the corresponding file under SASHome so the change is retained if the webapps are rebuilt and redeployed. In this case the file is web.xml.orig under SASHome/SASWebInfrastructurePlatform/9.4/Configurable/wars/sas.svcs.logon/WEB-INF. It is also recommended to maintain a document of all these changes and verify them after applying maintenance.

Some explanation is needed to explain the Tomcat realm. The realm is responsible for taking the credentials collected by the authenticator and authenticating them against some back-end resource such as a database or LDAP server. However, the SPNEGO authenticator is different than the others because it essentially does the authentication also. This could not be left to the realm because Kerberos can involve handshakes that span multiple requests. The realm can still verify that the user exists (since it doesn't have a password) in the back-end database or LDAP, but this is completely redundant.

As a best practice, the GSSContextEstablishedRealm should be used. This realm will immediately authenticate any user who has already established a Kerberos connection. In the server.xml file, replace the existing UserDatabaseRealm or the whole nested LockOutRealm plus UserDatabaseRealm with the GSSContextEstablishedRealm. It should always be used with the allRolesMode="authOnly" option:

```
<Realm className="com.sas.vfabrictcsvr.realm.GSSContextEstablishedRealm"
       allRolesMode="authOnly" />
```

If the GSSContextEstablishedRealm is nested inside a LockOutRealm, the allRolesMode="authOnly" attribute should be put on both of the realms. Otherwise, users will get a 403 message that states "You are not authorized to access applications on this web application server", "Public access is denied", or something similar.

There are additional considerations for SAS Mobile BI. Refer to the *SAS 9.4 Intelligence Platform: Middle-Tier Administration Guide, Third Edition* for more information.

ENABLE DEBUGGING OUTPUT

Rare is the case where an IWA configuration works on the first attempt, so getting good debugging output is absolutely essential to troubleshooting.

SPNEGO Debugging

The Apache Tomcat Catalina servlet container works at the HTTP layer to implement authentication using SPNEGO—Simple and Protected GSSAPI Negotiation Mechanism. This is where most of the debugging takes place, but unfortunately it is not straightforward to set up.

While debugging output is configured via Log4J, the output is split between messages generated by the container that go to the server.log, and messages generated by the web application that go to the individual web app log files under Config/Lev1/Web/Logs/. When an authentication method such as SPNEGO is configured in the web.xml file, a Tomcat authenticator class is instantiated by the container to handle the authentication. However, the authenticators seem to straddle both the container and web app logging, and unfortunately the debugging output falls through the cracks.

But we've already addressed this in configuration by using the SAS fallback valve and placing it at the server level in SASServer1_1/conf/context.xml instead of under the logon web application. This causes debugging output to go to the server.log file. Now all we need to do is add the following to the log4j.xml file under Web/WebAppServer/SASServer1_1/lib:

```
<category name="org.apache.catalina.authenticator">
  <priority value="DEBUG"/>
</category>
```

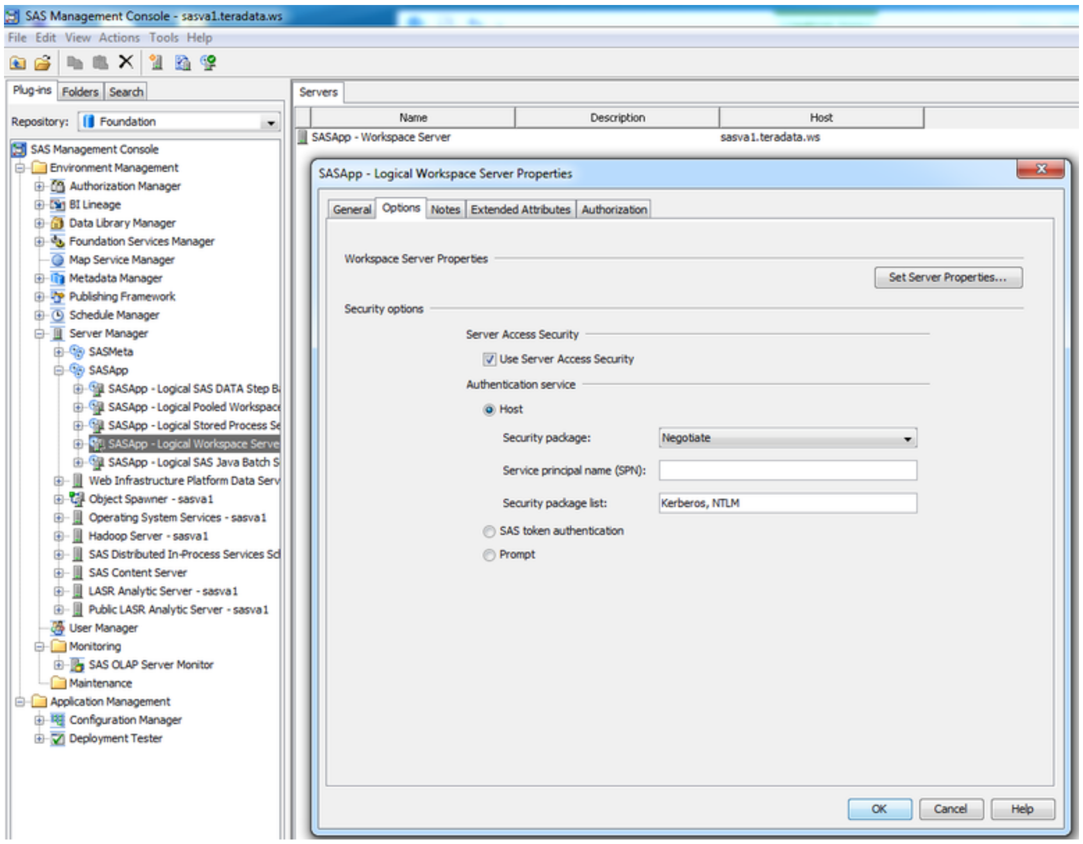
Kerberos Delegation

If Kerberos credential delegation will be used to launch workspace servers, we will want to enable debugging output to verify that credentials are being passed to the web application server during login. Edit the SASLogon-log4j.xml file under Web/Common/LogConfig and add the following:

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

WORKSPACE SERVER BEST PRACTICES

Although the server tier and workspace server are not the focus of this paper, it would be incomplete if it did not include the few simple things required to enable Kerberos delegation. Begin by opening SAS® Management Console. Find the **SASApp – Logical Workspace Server** under **Environment management > Server Manager** and right-click to edit the properties. Change the **Security package** to **Negotiate** and **Security package list** to **Kerberos, NTLM** as shown in Display 3.


SAS Management Console - sasva1.teradata.ws
File Edit View Actions Tools Help
Plug-ins Folders Search
Repository: Foundation
SAS Management Console
Environment Management
Authorization Manager
BI Lineage
Data Library Manager
Foundation Services Manager
Map Service Manager
Metadata Manager
Publishing Framework
Schedule Manager
Server Manager
SASMeta
SASApp
SASApp - Logical SAS DATA Step B
SASApp - Logical Pooled Workspac
SASApp - Logical Stored Process Se
SASApp - Logical Workspace Serve
SASApp - Logical SAS Java Batch S
Web Infrastructure Platform Data Serv
Object Spawner - sasva1
Operating System Services - sasva1
Hadoop Server - sasva1
SAS Distributed In-Process Services Sd
SAS Content Server
LASR Analytic Server - sasva1
Public LASR Analytic Server - sasva1
User Manager
Monitoring
SAS OLAP Server Monitor
Maintenance
Application Management
Configuration Manager
Deployment Tester
Servers
Name
Description
Host
SASApp - Workspace Server
sasva1.teradata.ws
SASApp - Logical Workspace Server Properties
General Options Notes Extended Attributes Authorization
Workspace Server Properties
Set Server Properties...
Security options
Server Access Security
Authentication service
OK Cancel Help

Use Server Access Security: ✓

Authentication service: [✓ Host] / SAS token authentication / Prompt

Security package: Negotiate

Service principal name (SPN):

Security package list: Kerberos, NTLM

Display 3. Workspace Server Properties in SAS Management Console

The **Service principal name** field can be left blank in most cases. However, if the server tier is in a different Kerberos realm from the middle tier, the full SPN should be specified with realm in the format SAS/<hostname>@<realm>.

As a good practice, check the workspace server's ability to use Kerberos by right-clicking it again and selecting **Validate**.

Metadata Accounts

Normally, the User object in metadata should have at least two accounts, one with the domain and one without it. Neither of these should have a password, but they both need to be there. The account without the domain is the one used when authenticating with the middle tier, since the Tomcat realm strips the Kerberos realm from the user name by default. The account with the domain is used when the workspace server receives a Kerberos connection from the middle tier and is attempting to look up the User associated with the ticket.

TROUBLESHOOTING

Troubleshooting is categorized here by the types of issues that are commonly encountered. The information here is assembled from experience troubleshooting countless deployments within SAS R&D for testing IWA and assisting with technical tracks and on-site deployments.

KEYTAB ISSUES

Verify That the Keytab Is Being Read

Upon the first attempt by a user to authenticate with the middle tier, the Tomcat SpnegoAuthenticator will call the com.sun.security.auth.module.Krb5LoginModule to log in. The login module should read in the keytab file and connect to the KDC. Output 9 shows an example of the messages that are output when the keytab file is read.

```
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTabInputStream,
readName(): NA.SAS.COM
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTabInputStream,
readName(): HTTP
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTabInputStream,
readName(): jdtvm01.na.sas.com
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTab: load() entry length:
93; type: 18
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTabInputStream,
readName(): NA.SAS.COM
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTabInputStream,
readName(): HTTP
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTabInputStream,
readName(): jdtvm01.na.sas.com
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTab: load() entry length:
77; type: 17
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTabInputStream,
readName(): NA.SAS.COM
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTabInputStream,
readName(): HTTP
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTabInputStream,
readName(): jdtvm01.na.sas.com
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTab: load() entry length:
77; type: 23
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTabInputStream,
readName(): NA.SAS.COM
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTabInputStream,
readName(): HTTP
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTabInputStream,
readName(): JDTVM01
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTab: load() entry length:
82; type: 18
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTabInputStream,
readName(): NA.SAS.COM
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTabInputStream,
readName(): HTTP
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTabInputStream,
readName(): JDTVM01
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTab: load() entry length:
66; type: 17
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTabInputStream,
readName(): NA.SAS.COM
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTabInputStream,
readName(): HTTP
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTabInputStream,
readName(): JDTVM01
INFO   | jvm 1    | 2014/04/08 10:22:25 | >>> KeyTab: load() entry length:
66; type: 23
```

Output 9. Output Showing Example Messages Output from Reading the Keytab

If these messages are not output, check the path of the keytab file in jaas.config. The path should use forward slashes, even on Windows. For example:

```
com.sun.security.jgss.krb5.accept {
   com.sun.security.auth.module.Krb5LoginModule required
   doNotPrompt=true
   principal="HTTP/d77793.na.sas.com@NA.SAS.COM"
   useKeyTab=true
   keyTab="C:/SAS/Config/Lev1/Web/WebAppServer/SASServer1_1/conf/jdtvm01-
HTTP.keytab"
   storeKey=true;
};
```

Java does not appear to use the keytab path in krb5.ini, but this file would use backslashes when on Windows.

Verify the Principal Name

Make sure that the correct principal name has been specified in jaas.config, corresponding to the name of the service account in Active Directory, and that it matches an entry in the keytab. Output 10 shows what happens if it is incorrect.

```
ERROR org.apache.catalina.authenticator.SpnegoAuthenticator - Unable to
login as the service principal
javax.security.auth.login.LoginException: Unable to obtain password from
user
```

Output 10. Output Showing an Error When the Principal Name in jaas.config Is Incorrect

Encryption Types

Some messages should be output when Java reads the Kerberos configuration file and attempts to find keys matching those in the keytab. Output 11 shows the messages regarding the encryption types.

```
INFO   | jvm 1    | 2014/04/08 10:22:25 | Java config name:
C:\SAS\Config\Lev1\Web\WebAppServer\SASServer1_1\conf\krb5.ini
INFO   | jvm 1    | 2014/04/08 10:22:25 | Loaded from Java config
INFO   | jvm 1    | 2014/04/08 10:22:25 | Added key: 23version: 1
INFO   | jvm 1    | 2014/04/08 10:22:25 | Added key: 17version: 1
INFO   | jvm 1    | 2014/04/08 10:22:25 | Added key: 18version: 1
INFO   | jvm 1    | 2014/04/08 10:22:25 | Ordering keys wrt
default_tkt_enctypes list
INFO   | jvm 1    | 2014/04/08 10:22:25 | default etypes for
default_tkt_enctypes: 23 18 17.
```

Output 11. Output Showing the Encryption Types Found in the Keytab

The Kerberos configuration file might contain default encryption types. Remove these lines to avoid problems with the encryption types not matching what is in the keytab:

```
default_tkt_enctypes = arcfour-hmac-md5,aes256-cts-hmac-sha1-96,aes128-cts-
hmac-sha1-96
default_tgs_enctypes = arcfour-hmac-md5,aes256-cts-hmac-sha1-96,aes128-cts-
hmac-sha1-96
```

JAVA ISSUES

The SAS middle tier, as well as some “rich” clients, rely on the Kerberos support built into Java. This introduces some issues specific to Java.

SAS® Enterprise Miner, SAS® Forecast Server, and SAS® Model Manager

These Java rich clients all share a common module for authenticating to the middle tier that supports IWA. That code relies on the SPNEGO support built into Apache HTTP Client, which in turn relies on Kerberos support in Java.

On Windows, Java does not use the native SSPI library to obtain Kerberos tickets and instead uses the Kerberos V5 protocol to communicate directly to the KDC. To do this, it needs access to the user's TGT and a session key to decrypt it. Unfortunately, by default Windows does not allow applications to access the session key associated with the current TGT in the cache. This might cause the Java client to prompt the user for their password, or it might just refuse to authenticate with the server altogether. There are two workarounds available.

The preferred workaround is to edit the windows registry with regedit, navigate to HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Lsa\Kerberos\Parameters, and add a new DWORD with the name allowtgtsessionkey and a value of 1.

If the registry cannot be changed, open a command shell and run “kinit -f" to get a new TGT and session key from the KDC, and cache it on the filesystem. Note that the TGT has a limited lifespan, so this would need to be done periodically.

Cannot Locate Default Realm

This problem is likely caused by either a misconfigured Kerberos configuration file or Java not being able to find the Kerberos configuration file, and consequently not being able to determine the default realm and KDC server. Normally it would be able to fall back on DNS and auto-discover this information, but in some networks this might not work. Output 12 shows the messages displayed when Java is attempting to use DNS to locate the default Kerberos realm.

```
INFO   | jvm 1    | 2015/10/06 10:48:38 | >>> KrbAsReq creating message
INFO   | jvm 1    | 2015/10/06 10:48:38 | getRealmFromDNS: trying
na.sas.com
INFO   | jvm 1    | 2015/10/06 10:48:38 | getRealmFromDNS: trying
unx.sas.com
INFO   | jvm 1    | 2015/10/06 10:48:38 | getRealmFromDNS: trying sas.com
INFO   | jvm 1    | 2015/10/06 10:48:38 | getRealmFromDNS: trying
emea.sas.com
INFO   | jvm 1    | 2015/10/06 10:48:38 | getRealmFromDNS: trying
apac.sas.com
```

Output 12. Output When Java Is Attempting to Locate the Default Kerberos Realm from DNS

You can tell Java where to find the Kerberos configuration file by setting the 'java.security.krb5.conf' system property. By default, the Tomcat code that does SPNEGO authentication automatically sets the system property, if it is not already set, to point to a file named 'krb5.ini' located in the server 'conf' directory. You can override this by setting the system property in the wrapper.conf or setenv.sh file. Since SPNEGO authentication happens only with SASLogon, this property gets set automatically only for SASServer1_1. For other servers, particularly servers involved with launching workspace servers, you need to set the property.

TICKET VALIDATION ISSUES

Output 13 shows the ominous error that occurs when the user agent has supplied a Kerberos ticket, but it could not be validated.

```
Failed to validate client supplied ticket…
```

Output 13. Output When the Supplied Kerberos Ticket Cannot Be Validated

That message should be followed by more detailed information about why the ticket could not be validated. Note that SAS 9.4 maintenance release 3 fixed a bug preventing the detailed GSS exception from displaying. A hotfix is available for earlier maintenance releases.

Specified Version of Key Is Not Available

Output 14 shows the error that is output when the keytab does not have an entry matching the ticket received.

```
Failed to validate client supplied ticket [GSSException: Failure
unspecified at GSS-API level
(Mechanism level: Specified version of key is not available (44))]
```

Output 14. Output Showing KVNO Error

The most common cause of this is that the KVNO was recently incremented (for example, by running ktpass), and the client PC is using a cached a ticket with the old KVNO. On the client PC, run the “klist purge” command to clear the cached tickets. It is a good idea to run this whenever changing something in Active Directory.

You might also get this error if another account is using the same SPN or principal name. You can use the setspn command to check the SPN, but that won't tell you if another account has this name as its user principal name. We have even seen cases where another account with the same principal name continued to be a problem even after it was deleted. This is evident if you edit the service account in Active Directory and change the User Logon Name to be the same as the SPN. Microsoft Management Console will complain if another account is using that name already.

Defective Token Detected

A common problem is that the client attempts to do NTLM with the middle tier instead of Kerberos. Output 15 shows the error that is displayed when the client attempts to authenticate with NTLM.

```
Failed to validate client supplied ticket [GSSException: Defective token
detected
(Mechanism level: GSSHeader did not find the right tag)]
```

Output 15. Output showing the error that is displayed with NTLM

This usually happens when using Internet Explorer from the same machine that the middle tier is running on. Internet Explorer will detect that the server is on the same machine and try to do NTLM instead of Kerberos when challenged with the Negotiate response. If this is the case, Internet Explorer should pop up a window and the user should still be able to enter credentials, and then it will do Kerberos.

This might also happen when there are problems with the SPN. After receiving a 401 status code and "WWW-Authenticate: Negotiate" response from the server, the web browser will construct what it thinks the SPN should be and will ask the domain controller for a Kerberos ticket for that SPN. The SPN is constructed by taking HTTP/ and appending the host name of the server being called. Some browsers might do a reverse lookup on the IP to get the correct name. If the domain controller cannot find an SPN by that name, it refuses to give a ticket, and the browser reverts to NTLM instead, which fails with this error. Make sure the service account in Active Directory (from which you created the keytab) has all versions of the SPN in its servicePrincipalName attribute, including short names, fully qualified names, aliases, and so on. Note that SPNs always start with uppercase HTTP/ even if the actual protocol is https.

Checksum Failure

Checksum errors are some of the hardest errors to troubleshoot, since the error doesn’t give any clues what is wrong. Output 16 shows the error that is output when the checksum doesn’t match what is expected.

```
Failed to validate client supplied ticket [GSSException: Failure unspecified
at GSS-API level (Mechanism level: Checksum failed)]
```

Output 16. Output indicating a checksum error

Incorrect or Missing SPNs

With a checksum failure, the first thing to do is check that the SPNs are registered correctly in Active Directory, especially if you are using DNS aliases. Use the setspn command to check each one. Internet Explorer apparently does a reverse DNS lookup to find the machine name and gets a ticket for that. The safest thing to do is add all the SPNs that might be used, including DNS aliases and machine names.

Microsoft PAC with AES Encryption

There is a known issue with Java handling Kerberos tickets encrypted with AES and containing Microsoft PAC information, which results in checksum errors.

The domain controller can be told to not include PAC information in the ticket (Java doesn't use it anyway) via the userAccountControl attribute on the service account in Active Directory. This field is a bit mask, and one of the bits is called NO_AUTH_DATA_REQUIRED. Take the existing value of this attribute and use OR to join it with 33554432. Alternatively, you can resort to using the default RC4-HMAC encryption instead of AES by editing the Account Options for the service account in Active Directory and deselecting the boxes for AES 128 and 256 encryption.

The middle tier does not need to be restarted for either of these options to take effect, but "klist purge" should be run on the client PCs to purge any existing tickets.

JNDI REALM ISSUES

Configurations that use a Java Naming and Directory Interface (JNDI) realm might have problems with the LDAP connection settings and search criteria. The realm is called after the Kerberos exchange, so the message shown in Output 17 rules out a Kerberos problem and causes one to suspect the realm next.

```
>>> KrbApReq: authenticate succeed.
```

Output 17. Output Indicating That Kerberos Authentication Was Successful

Unfortunately, the Tomcat JNDI realm outputs very little debugging information. If incorrect connection information was specified, an appropriate error message will be output in the logs. But if the user lookup/search fails, only the 401 response is returned to the browser. Output 18 shows what will be displayed in the log.

```
[org.apache.catalina.authenticator.AuthenticatorBase]  Failed
authenticate() test
```

Output 18. Output Showing the Message That Is Displayed When the LDAP Lookup Fails

If the user lookup/search is successful, but the realm does not find the appropriate role, the browser will receive a 403 response and display a message "You are not authorized to access applications on this web application server", "Public access is denied", or something similar. Output 19 shows the error message that is displayed in the log.

```
[org.apache.catalina.authenticator.AuthenticatorBase] Authenticated
'miroda' with type 'SPNEGO'
[org.apache.catalina.authenticator.AuthenticatorBase]  Calling
accessControl()
[org.apache.catalina.realm.RealmBase]   Checking roles
GenericPrincipal[miroda(*Dept: JDT (Java Development
[org.apache.catalina.authenticator.AuthenticatorBase]  Failed
accessControl() test
```

Output 19. Output Showing the Error When the User Does Not Have the Required Role

Either of these errors will require going back and checking the LDAP configuration on the JNDIRealm, or replacing it with the GSSContextEstablishedRealm.

KERBEROS DELEGATION ISSUES

When debugging Kerberos delegation, the absolute first thing that should be done is to make sure users can log in to the middle tier with IWA using a web browser and that credentials are being delegated. If the debugging output was enabled as described earlier, messages like that shown in Output 20Output 20. Output Showing the Message Displayed in the Log When Credentials Have Been should appear in the log file for SASLogon.

```
... INFO
com.sas.svcs.security.authentication.gss.GSSCredentialCachingFilter -
Received credentials for 'miroda'.
```

Output 20. Output Showing the Message Displayed in the Log When Credentials Have Been Delegated

If these messages do not appear, the user's browser is not sending credentials to the application server. Refer to the *SAS 9.4 Intelligence Platform: Middle-Tier Administration Guide, Third Edition* for instructions on configuring Internet Explorer, Chrome, and Firefox for delegation.

If the previous test passes, the problem is likely on the back-end server tier launching the workspace server using Kerberos. An error should be displayed in the Object Spawner logs under Config/Lev1/ObjectSpawner/Logs/.

Server Not Found in Kerberos Database

A bug in SAS 9.4 maintenance release 3 misconfigures the Apache HTTP client used by Java rich clients such as SAS Enterprise Miner such that they include the port number in the service principal name. This results in an error like that shown in Output 21.

```
>>>KRBError:
         sTime is Wed Aug 19 13:10:15 EDT 2015 1440004215000
         suSec is 196533
         error code is 7
         error Message is Server not found in Kerberos database
         realm is RACE.SAS.COM
         sname is HTTP/pdcesx08007.race.sas.com:8343
         msgType is 30
KrbException: Server not found in Kerberos database (7)
```

Output 21. Kerberos Exception in the SAS Enterprise Miner Log

The workaround to this problem is to register the SPNs in Active Directory with and without port numbers.

CONCLUSION

Configuring Integrated Windows Authentication is not difficult, but when it doesn't work it can be a real challenge to determine the problem.

It all starts with a good Kerberos keytab, associated with a service account in Active Directory with the right service principal names, and this paper has shared tips for achieving that. Then there is the middle-tier configuration, where the best practices described here are designed to minimize complexity and maximize debugging output. Finally, even with the most careful effort, most administrators will find themselves referring to the last part of the paper where all of the issues and common pitfalls that this author has encountered over the last few years are explained.

I hope that you find this paper a valuable resource. Additional documentation is provided in the following references.

REFERENCES

Massachusetts Institute of Technology. 2015. "MIT Kerberos Distribution." Available http://web.mit.edu/Kerberos/dist/. Accessed on January 26, 2016.

Microsoft Corporation. 2016. "An update is available that introduces the NO_AUTH_REQUIRED flag to the UserAccountControl property in Windows Server 2003 and in Windows 2000." Available https://support.microsoft.com/en-us/kb/832572. Accessed on January 26, 2016.

Microsoft Corporation. 2016. "Remote Server Administration Tools for Windows 7 with Service Pack 1 (SP1)." Available https://www.microsoft.com/en-us/download/details.aspx?id=7887. Accessed on January 26, 2016.

Microsoft TechNet. 2016. “Ktpass.” Available https://technet.microsoft.com/en-us/library/cc753771.aspx. Accessed on January 26, 2016.

Olougouna, Edgar A. “Understanding Microsoft Kerberos PAC Validation.” Last modified April 24, 2009. Available http://blogs.msdn.com/b/openspecification/archive/2009/04/24/understanding-microsoft-kerberos-pac-validation.aspx. Accessed on January 26, 2016.

Oracle Inc. “Maximum Key Sizes Allowed by ‘Strong’ Jurisdiction Policy Files.” Available at http://docs.oracle.com/javase/7/docs/technotes/guides/security/SunProviders.html#importlimits.

Oracle Technology Network. 2016. “Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files 7 Download.” Available http://www.oracle.com/technetwork/java/javase/downloads/jce-7-download-432124.html. Accessed on January 26, 2016.

Rogers, Stuart J. 2013. “Kerberos and SAS 9.4: A Three-Headed Solution for Authentication.” *Proceedings of the SAS Global Forum 2013 Conference*. Cary, NC: SAS Institute Inc. Available http://support.sas.com/resources/papers/proceedings13/476-2013.pdf.

SAS Institute Inc. 2014. *SAS 9.4 Intelligence Platform: Middle-Tier Administration Guide, Third Edition*. Cary, NC: SAS Institute Inc. Available http://support.sas.com/94administration.

ACKNOWLEDGMENTS

Each of the following people at SAS has made a substantial contribution to the knowledge base that this paper draws from: Brian English, Stuart Rogers, and Adam Smith. But I am especially grateful to Evan Kinney for his extensive knowledge of Kerberos and the peculiarities of Microsoft implementation.

CONTACT INFORMATION

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

Mike Roda
100 SAS Campus Drive
Cary, NC 27513
SAS Institute Inc.
919-531-5891
mike.roda@sas.com