Cross-domain single sign-on using SAML 2.0 with WebSphere Liberty, Part 3: Integrate Microsoft Windows authentication by using SPNEGO

Cristiano Mauri
Guido Gilardi

March 27, 2017

This 3-part series, "Cross-domain single sign-on using SAML 2.0 with WebSphere Liberty," introduces an end-to-end single sign-on (SSO) solution that uses IBM Cloud in a hybrid cloud environment. Part 3 describes how to integrate the service provider-initiated SSO with the Microsoft Active Directory authentication in a Microsoft domain network. It shows you how to configure the Microsoft Active Directory and the WebSphere Liberty profile to implement the SPNEGO web authentication.

View more content in this series

Introduction

About this series
This 3-part series, "Cross-domain single sign-on using SAML 2.0 with WebSphere Liberty," introduces an end-to-end single sign-on (SSO) solution that uses IBM Cloud in a hybrid cloud environment. It explains how to use WebSphere Liberty to enable Java™ EE standard applications on the cloud to securely invoke services that are exposed in a private network. In this solution, you see how to configure the standard features of IBM WebSphere Liberty (without any customization) to enable Java EE security and to propagate the authenticated subject over different domains.

When a computer that is running Microsoft® Windows® is connected to a domain, and the user logs in to the computer with his domain credentials, the Windows operating system contacts the domain controller to authenticate the user. This scenario is typical in many organizations where users have computers that run the Windows operating system that is connected to a domain controller, and they have a domain user name and password to log in to their computers. If users need to connect to web applications that are running in the company's intranet, often, they need to provide the same domain user name and password.

Part 3 of this tutorial series shows how you can extend the SSO solution that is explained in Part 1 and Part 2 to use your existing Microsoft Active Directory infrastructure for authentication and authorization. You see how to access the cloud applications in a secure way, without challenging users for their credentials.
To complete the solution in this part, you must have read and completed the steps in Part 1 and Part 2. Also, you must ensure that a Windows Server operating system is installed on your network.

Overview of the solution

Suppose that you log in to a computer that is running Windows and is connected to a Microsoft domain. When you enter your domain, user name, and password in the operating system login form, a request is made to a Microsoft Active Directory to authenticate your credentials. From this computer, the browser connects to a web application that is running on an IBM WebSphere Liberty application server.

When you request a resource that is protected by role-based authorization (Java Authentication and Authorization Service) security, if you are not yet authenticated, the browser prompts you to provide your user name and password. You must sign in because the application server detects that you are not authenticated and adds the **WWW-Authenticate: Basic** header to the HTTP response. The browser intercepts this header and shows you the login form. You enter your credentials. The application server then checks them against its user registry, establishes the security context, and performs role-based authorization to see whether you are allowed to access the required resource.

Now, suppose that the application server is configured with a Lightweight Directory Access Protocol (LDAP) user registry that points to the Microsoft Active Directory that authenticated you when you logged in to your computer. In this case, you can use your domain credentials to log in also to the web application.

You can also configure the application server that hosts the application and your browser to negotiate authentication with your computer's operating system API, without prompting you to log in. Specifically, you configure the **Simple and Protected GSS-API Negotiation Mechanism** (SPNEGO) feature on the WebSphere Liberty application server that hosts the web application.

By enabling this feature, when you connect your browser to the web application and request a protected resource, if you are not authenticated, the server adds the **WWW-Authenticate: Negotiate** header to the HTTP response. If your browser is configured to authorize negotiation for the domain to which the resource belongs, it intercepts the header and calls the Windows API to obtain a Kerberos ticket (SPNEGO token) from the Microsoft Active Directory. The SPNEGO token is sent back from the browser to the application server. The SPNEGO feature validates the token and uses it to retrieve the user name and the groups from the LDAP (Microsoft Active Directory) user registry. The application server then uses the user name and groups to create the authenticated subject and to perform the role-based authorization check on the resource that you requested.

To explore the details of the SPNEGO authentication process, see **Single sign-on for HTTP requests using SPNEGO web authentication** in the IBM Knowledge Center.
SPNEGO authentication can coexist with the service provider-initiated SSO process that was explained in Part 1 because they have separate tasks. In the solution in this part, SPNEGO authentication happens only after the service provider redirects the user to the identity provider (IdP) to perform the authentication and finishes his tasks before the identity provider generates the SAML token. The following figure illustrates the SSO sequence that was just described.
Cross-domain single sign-on using SAML 2.0 with WebSphere Liberty, Part 3: Integrate Microsoft Windows authentication by using SPNEGO

[Diagram of the integration process]
In this tutorial, you enable the Microsoft Active Directory Domain Services (AD DS) role on the Windows Server. You then create a Microsoft domain network. (We use samlssosample.net in this example.) The AD DS is supported on Windows 2000 Server, Windows Server 2003, Windows Server 2008, and Windows Server 2012. It is not available for other Microsoft operating systems. In this tutorial, we use Windows Server 2008 R2.

You create the required users and groups and the Service Principal Name (SPN) in the AD DS. The IdentityServer uses the SPN to perform Kerberos authentication to the AD DS. You configure the IdentityServer. You replace the basic user registry from Part 1 with an LDAP user registry that points to the AD DS, and you install and configure the SPNEGO feature.

Finally, you perform single sign-on (SSO) between the application that you created in Part 1 and Part 2 by using the operating system credentials (the Microsoft domain credentials). You set up a Windows computer to connect to the Microsoft domain, and you configure the computer’s browser to negotiate the Kerberos ticket.

The following diagram illustrates the solution architecture that you configure in this part. The domain network (in the red frame in the lower-left corner) and the communication protocols that are used to implement the SSO include:

- HTTPS between all the web applications
- The LDAP between the IdentityServer and the AD DS
- KRB5, the Kerberos protocol that is used by the Windows operating system of the user’s computer to obtain the Kerberos ticket

The IdentityServer is the only server that holds the user registry in our solution, which means:

- You do not need to change the SSO solution setup that you completed in Part 1 and Part 2.
- You do not need to expose the AD DS to the cloud because it needs to communicate only with the IdentityServer (and the user’s computer).
1. Configure the Microsoft Active Directory domain controller

To configure the Active Directory Domain Services, first, you install the AD DS role on your Microsoft Windows Server, if it is not already installed. Then, you create the Microsoft domain network. Next, you configure the required users and groups to run the example and the SPN that is required by the SPNEGO authentication.

1a. Install the Active Directory Domain Services

Before you begin, log in to your Microsoft Windows Server with an administrator account. Then, to install the Active Directory Domain Services:

1. From the Windows desktop, select **Start -> Run**.
2. In the window that opens, type **ServerManager.msc**.
3. In the **Server Manager** window, right-click **Roles** and select **Add Roles**.
4. On the **Before You Begin** tab, click **Next**.
5. On the **Server Roles** tab, select **Active Directory Domain Services** and click **Next**.

6. On the **Active Directory Domain Services** and **Confirmation** tabs, click **Next**.
   At this point, the installation starts. You can monitor it on the **Progress** tab of the wizard.
   When the installation finishes, the **Results** page is displayed where you see that all features installed successfully. In this example, only the AD DS was installed, but you might see other dependencies installed if they are not found (such as the .NET Framework Features).
7. On the **Results** page, click **Close** to return to the **Server Manager** window.

![Installation Results](image)

### 1b. Create the domain

Create a domain for the Microsoft Windows network:

1. In the ServerManager window, expand **Roles** and click **Active Directory Domain Services**.
2. Under **Summary**, click **Run the Active Directory Domain Services Installation Wizard (dcpromo.exe)**.
3. In the first window of the **Active Directory Domain Services Installation Wizard**, click **Next**.
4. In the warning message window about improved security, click **Next**.
5. In the next window, select **Create a new domain in a new forest** and click **Next**.
6. In the **Name the Forest Root Domain** window, enter your domain. You must enter the domain name in lowercase. In this example, we use `samlssosample.net`. Click **Next**.

![Name the Forest Root Domain window](image)

7. In the **Set Forest Functional Level** window, select the functional level. We choose the level that corresponds to the most recent version, which is **Windows Server 2008 R2** in this example. You can see a different level depending on the operating system you are running. Click **Next**.

8. In the next window, select **DNS server**, unless you plan to configure with an existing DNS server. Click **Next**.

9. If your Window Server machine has a dynamic IP assigned, a message informs you about the IP. You can then choose if you want to configure it with a DHCP server or assign a static IP later. We use **DHCP server** in the tutorial, which is acceptable for this example, even though Microsoft does not recommend it. In a production environment, you set it up by using a static IP.

10. In the next window, click **Yes**.

11. Configure the location of the AD DS files. In this example, we accept the default location. Click **Next**.

12. In the next window, enter a password for the **Restore Mode Administrator**. You must write it according to the AD DS security policy, which uses a combination of uppercase letters, lowercase letters, and numbers. We use `Passw0rd` in this tutorial. Click **Next**.

13. In the last window, click **Finish**, and restart your computer.

If you are installing on Windows Server 2012, the wizard is slightly different. For more information, see [Install a New Windows Server 2012 Active Directory Forest (Level 200)].

1c. **Optional: Change the password policies**

1. From the Windows desktop, click **Start -> Run**. Type `gpmc.msc`.

2. In the **Group Policy Management** console, expand **Group Policy Management -> Forest: samlssosample.net -> Domains -> samlssosample.net**. Right-click **Default Domain Policy** and select **Edit**.

4. To change the settings, right-click the policy, select **Properties**, and make your changes. After some time, the policies are loaded.

1d. **Create the groups**

Populate the users and groups in the Microsoft Active Directory. To start, create the required groups:

- IdentityRequestors
- FrontendServiceUsers
- CloudServiceUsers
- LocalServiceUsers

To create the groups:

1. From the Windows desktop, select **Start->Run**.
2. In the window that opens, type `serverManager.msc`.
3. Expand **Roles -> Active Directory Domain Services -> Active Directory Users and Computers**.
4. Right-click `samlsso.sample.net` and select **New -> Group**.
5. In the New object - Group window:
   a. For **Group name**, enter `IdentityRequestors`.
   b. For **Group Scope**, select **Global**.
c. For **Group Type**, select **Security**.
d. Click **OK**.

The IdentityRequestor group is now created.

6. Repeat steps 3 - 5 to create the FrontendServiceUsers, CloudServiceUsers, LocalServiceUsers groups.

### 1e. Create the users

To create the required users, first, you create a technical user that you configure later in the IdentityServer Liberty profile to connect to the AD DS. This user is not required in any of the groups that you previously set up because it is not required to access to the application. This user will belong only to the Domain Users group, which is the default group for new users.

To create the users:

1. From the Windows desktop, click **Start->Run**.
2. In the window that opens, type `ServerManager.msc`, if it is not already opened.
4. Right-click **Users** and select **New -> User**.
5. In the New object - User window, for First name and the User logon name, enter `wasuser`. Click **Next**.
6. Set the password for the new user. We used `Passw0rd` as an easy password to remember. Clear all checkbox options, and then select **Password never expires**. Click **Next**.
7. Click **Finish**.

The user `wasuser` is now created.

8. Repeat the steps 3 - 7 to create four new users: `rob`, `max`, `bobby`, and `alice`.
9. Set up the proper membership for these users to reproduce the configuration that is in the basic user registry of the IdentityServer. Click **Users**. In the
middle pane of the Server Manager window, you see the list of your users.

10. Right-click **rob** and select **Properties**.
11. On the **Member Of** tab, click **Add**.
12. In the Select Groups window, in the **Enter the object names to select** box, enter **IdentityRequestors**. You can also click **Advanced** and **Find now** for a complete list of all the objects. Click **OK**.
13. Repeat steps 10 - 13 to create the memberships for **max**, **bobby** and **alice**, as follows:
   - **max → IdentityRequestors; FrontendServiceUsers**
   - **bobby → IdentityRequestors; FrontendServiceUsers; CloudServiceUsers**
   - **alice → IdentityRequestors; FrontendServiceUsers; CloudServiceUsers; LocalServiceUsers**

**Tip:** We listed the groups separated by semicolon. The semicolon character is a special entry separator in the search object wizard that you can use to copy and paste the groups into the **Enter the object names to select** area.
The Active Directory is configured. Next, you generate the mapping so that WebSphere Liberty can connect to the AD DS.

1f. Create the Service Principal Name mapping

You now prepare the configuration so that the IdentityProvider can log in to the AD DS. You map wasuser to the SPN. The user PC uses the service principal name, which is in the form HTTP/<IdP-full-hostname>, to request a Kerberos ticket. The identity provider also uses this name to log in to the AD DS and retrieve the groups for the user. To do this, you generate a key file that the IdentityProvider uses to log in to the domain (samlssosample.net in the example). If you created wasuser with an expiring password, every time you change the password for wasuser, you must repeat the procedure to regenerate the key and export it to the IdentityProvider machine.

To map the user to the SPN and to generate the key for the machine where the IdentityProvider runs, you use the ktpass command with the -mapuser option:

1. Log on to your Windows server, open a command prompt, and type:
   
   ```
   ktpass -out spnmap.keytab -princ HTTP/win7pro.samlssosample.net@samlssosample.net -mapuser wasuser -pass Passw0rd -ptype KRB5_NT_PRINCIPAL
   ```

   In the command parameters, you see `win7pro.samlssosample.net@samlssosample.net`, which is the full host name of the machine where the IdentityProvider runs. You must use lowercase and uppercase as written in this command. If you do not follow proper capitalization, authentication from WebSphere Liberty will fail.

   When you look at the User logon name field on the Account tab in the wasuser Properties window after you run the ktpass command, you see that it changed from wasuser to HTTP/win7pro.samlssosample.net, which is the SPN.
2. Check the proper SPN configuration:

```bash
setspn -l wasuser
```

In the output, you see the SPN mapped to the `wasuser` distinguished name:

```
Registered ServicePrincipalNames for CN=wasuser,CN=Users,DC=samlsso,DC=sample,DC=net:
HTTP/win7pro.samlsso.sample.net
```

3. Copy the `spnmap.keytab` file to the machine where you installed the IdentityServer, in the path

```bash
<WLP_SERVERS>\IdentityServer\resources\security\spnego\.
```

You create the `spnego` folder because it does not exist.

`<WLP_SERVERS>` is the servers folder of your WebSphere Liberty installation, where you created the IdentityServer profile in Part 1. In this example, the host name of the identity provider machine is `win7pro.samlsso.sample.net`.

2. Configure the identity provider (IdentityServer)

Download the projects and code examples!

This tutorial includes projects and code examples that you can use and refer to as you work through the steps. You can download them from GitHub or this compressed file.

The required projects for the IdentityServer are available on GitHub. You can download the complete set from this compressed file and extract it to the local file system of the machine where you installed the Eclipse workspace.

We refer to the local filesystem paths by using the following placeholders:

- `<WLP_HOME>` for the WebSphere Liberty home path (for example: `C:\programs\wlp-16.0.0.3`)
- `<WLP_BIN>` for the WebSphere Liberty bin path (for example: `C:\programs\wlp-16.0.0.3\bin`)
- `<WLP_SERVERS>` for the WebSphere Liberty server installation path (for example: `C:\programs\wlp-16.0.0.3\usr\servers`
Before you begin this part, ensure that the machine where you run the IdentityServer can connect to the Windows Server machine that is running the AD DS on the LDAP port. (In this tutorial, we use the default of 389.)

2a. Import the sample projects

Open the Eclipse workspace that you created in Part 1, and import the sample projects:

1. From the workspace, select File -> Import.
2. In the Import wizard, select General -> Existing projects into workspace, and click Next.
3. In the Import Projects window:
   a. For Select root directory, click Browse, and select the path in your local file system where you extracted the projects. Click OK.
   b. Under Options, select Copy projects into workspace.
   c. Click Finish.

Your workspace now looks similar to the following example.
Because the IdentityProvider feature is a web bundle, the SPNEGO filter does not work. The two projects you just imported are a workaround to allow SPNEGO authentication and SAML generation to coexist on the same server with minimal effort. Remember that the identity provider is not a fully compliant IdP implementation. If you need to move to production, you might want to adopt a product that provides an IdP implementation and support.

2b. Deploy the sample projects to the IdentityServer
Deploy the projects that you imported previously to your server:

1. Select the Servers view, right-click the IdentityServer, and select Add and Remove.
2. In the Add and Remove window, click IdentityProviderAuthenticator in the Available box on the left side, and click Add to move it to the Configured box on the right side. Then, click Finish.

2c. Configure the LDAP user registry
You configure the new user registry, which points to the Windows Server AD DS, in the IdentityServer server.xml file:

1. Select the Servers view, expand IdentityServer, and double-click Server Configuration.
2. If the server.xml file opens on the Design tab, click the source tab to switch to the XML view.
3. Add the ldapRegistry-3.0 feature to the existing <featureManager> element:
4. Replace the `basicRegistry` configuration with the `ldapRegistry` configuration:

```xml
<ldapRegistry
    baseDN="DC=samlsso,DC=sample,DC=net"
    bindDN="CN=wasuser,CN=users,DC=samlsso,DC=sample,DC=net"
    bindPassword="Passw0rd"
    host="Base-Win2k8x64.samlsso.sample.net"
    ldapType="Microsoft Active Directory"
    port="389">
</ldapRegistry>
```

The `ldapRegistry` configuration element has the following attributes:

- **baseDN** is the domain in the LDAP format. DN (distinguished name) is used for the baseDN and the bindDN, because we are using an LDAP user registry.
- **bindDN** and **bindPassword** are the technical user that you created in the AD DS when you configured the users. The Liberty server uses this user to connect to the AD DS.
- **host** is the Windows Server where you installed the AD DS (Base-Win2k8x64.samlsso.sample.net in this example).
- **port** is the default LDAP port, 389, onto which the AD DS is listening for LDAP requests.

5. Change the `identityProvider` configuration element:

```xml
<identityProvider
    issuer="samlsso.sample.net"
    keyAlias="idpkey"
    keyStoreRef="saml"
    hostname="win7pro.samlsso.sample.net"
    port="443"
    webContextPath="idpweb">
</identityProvider>
```

The `identityProvider` configuration has the following attributes:

- **hostname** is changed to reflect the host name of the identity provider that will be used as the SPN by the Windows client machine to request a Kerberos ticket to the AD DS.
- **webContextPath** is the new web context that is used for the SP-initiated SSO to request a SAML token. You need this workaround because the SPNEGO filter is not called by the identity provider because it is packaged as a web bundle.

### 2d. Install and configure the SPNEGO Liberty feature

To install the `spnego-1.0` Liberty feature, open a command prompt on the machine where you created your IdentityServer:

1. Open a command prompt, and enter `cd <WLP_BIN>`.
2. Run the following command for `spnego-1.0` to download and install the feature that is required to enable SPNEGO web authentication:

   ```bash
   installUtility install spnego-1.0
   ```

   You see the following message after the download and installation are completed successfully:
All assets were successfully installed.
Start product validation...
Product validation completed successfully.
3. Restart the Eclipse workspace to reload the list of available server features.
4. Create a file with the required information so that the SPNEGO feature can negotiate and manage the Kerberos tickets:
   a. Create a new file called krb5.ini in the <WLP_SERVERS>\IdentityServer\resources \security\spnego\ folder, where you previously copied the spnmap.keytab file.
   b. Copy the following snippet in the krb5.ini file that you just created:

```
[libdefaults]
    default_realm = SAMLSSO.SAMPLE.NET
    default_tkt_enctypes = rc4-hmac
    default_tgs_enctypes = rc4-hmac
    kdc_default_options = 0x54800000
[realms]
    SAMLSSO.SAMPLE.NET = {
        kdc = Base-Win2k8x64.samlss.sample.net:88
        default_domain = samlss.sample.net
    }
[domain_realm]
    .samlss.sample.net = SAMLSSO.SAMPLE.NET
```

Note the following points:
• domain. If you have a different domain, replace samlss.sample.net with your domain and use proper capitalization.
• kdc parameter. In this parameter, you enter the host name of your Windows Server that is running the AD DS.
5. Select the Servers view, expand the IdentityServer, and double-click Server Configuration.
6. Add the spnego-1.0 feature to the existing <featureManager> element:

```
<featureManager>
    <feature>javaee-7.0</feature>
    <feature>localConnector-1.0</feature>
    <feature>identityProvider-1.0</feature>
    <feature>ldapRegistry-3.0</feature>
    <feature>spnego-1.0</feature>
</featureManager>
```

7. Add the following snippet to the server.xml file:

```
<spnego canonicalHostName="false" disableFailOverToAppAuthType="false" includeClientGSSCredentialInSubject="false"
    krb5Config="$[server.config.dir]/resources/security/spnego/spnego.keytab"
    krb5Keytab="$[server.config.dir]/resources/security/spnego/spnmap.keytab"
    servicePrincipalNames="HTTP\win7pro.samlss.sample.net"
    trimKerberosRealmNameFromPrincipal="true">
</spnego>
```

The krb5Config and krb5Keytab parameters point to the file system where you saved the required SPNEGO files.
8. Right-click IdentityServer and select Start (or Restart if the server is already running).

**2e. Refresh the idp-metadata.xml file on the FrontendServer**

Because you need to use the full host name of the identity provider to activate the SPNEGO negotiation, you must change the idp-metadata.xml file on the FrontendServer and republish the server configuration to IBM Cloud:
1. Start the **IdentityServer**.
2. From the browser of the machine where your IdentityServer is running, open the `http://localhost/idp/SAMLResource` link.
3. Click the **IdpMetadata.xml** button to download the `idp-metadata.xml` file. Save the file in the `<WLP_SERVERS>\FrontentServer\resources\security\` directory.
4. From your Eclipse workspace, right-click the **WebSphere Application Server Liberty** project, and select **Refresh**.
5. Right-click the **FrontentServer**, and click **Publish**. If the **FrontentServer** is running, right-click it, and select **Stop**.
6. Right-click **IBM Cloud**, and select **Publish**.

You are now done with the LDAP and SPNEGO configurations. The next step is to log in to a computer that is running a Windows client. (We use Microsoft Windows 7 in this tutorial.) You use one of the users (bob, rob, max, or alice) that you configured previously in your AD DS for the final browser configuration and to test the full SSO scenario.

### 3. Configure the Windows client and test the solution

You should have a machine running a Windows client operating system, such as Windows 7, and connected to your Microsoft domain (`samlsso.sample.net` in the tutorial). If you don't have one running, you can configure one as explained in the following steps.

Before you start, from this machine:

- You must able to connect to the IdentityProvider machine by using its full host name (`win7pro.samlsso.sample.net` in this tutorial) on the https port (`443` in the tutorial).
- You must be able to connect to the AD DS machine. You can try a telnet on ports `88`, which is the port where AD DS listens for ticket service requests.

If you already have a machine connected to your existing AD DS, go to step 3b to continue with the browser configuration.

### 3a. Connect the machine to the AD DS domain

1. Log in to the Microsoft Windows client machine by using an administrator account.
2. From the Windows desktop, click **Start**, right-click **Computer**, and select **Properties**.
3. In the Property window, click **Advanced system settings**.
4. In the System Properties window, select the **Computer Name** tab, and click **Change**.
5. In the Computer Name/Domain Changes window, select **Domain**, and enter the domain name in lowercase that you configured in the AD DS (samlssosample.net in this example).

![Computer Name/Domain Changes](image)

Click **OK**.

6. Enter the administrator user name and password of your AD DS machine, and click **OK**.
7. If the credentials are valid, a window opens with the message: **Welcome to the <domain-you-configured> domain.** Click **OK**.
8. Restart the machine.
9. After the machine is restarted successfully, log in by using one of the users that you configured previously in the AD DS, for example, *alice* with *Passw0rd*.

Below the Windows login form, you see the AD DS domain in the old Windows format (*SAMLSO* in this tutorial).

3b. **Configure your browser**

To negotiate the Kerberos ticket with the identity provider, you must first configure your web browser. You can set up Firefox or Internet Explorer, as explained in the following sections, to test the solution.

**Set up Firefox**

1. Open your Firefox web browser.
2. In the address field, type `about:config`.
3. In the Search field, type `network.nego`. (Firefox filters the list of properties while you are typing.)
4. Right-click `network.negotiation-auth.delegation.uris`, and select **Modify**. In the input field, enter your domain (**samlssosample.net** in this tutorial).
5. Right-click `network.negotiation-auth.trusted.uris`, and select Modify. In the input field, enter your domain (`samlsso.sample.net` in this tutorial).

![](about_config.png)

Set up Internet Explorer

1. Open your Internet Explorer web browser.
2. From the menu bar, select Tools->Internet Options.
3. In the Internet Options window, on Security tab, select Local intranet, and click Sites.
4. In the next panel, click Advanced.
5. In the Local intranet panel, in the Add this website to the zone field, enter `*.samlsso.sample.net`. If you have a different domain, select your domain, which is prefixed by `.`. Click Add.

![](local_intranet.png)

Click Close.

6. On the Internet Options panel, click OK.

3c. Test the solution

From your Windows client, open the browser you configured and go to: https://FrontendServices.mybluemix.net/FrontendWeb

Replace the domain `FrontendService.mybluemix.net` with the domain that you created in Part 2 of this series.
Look at the network trace to see the SP-initiated SSO in action. If you are using Firefox, press F12, or from the **Options** menu, select **Developer** to enable the toolbar.

The sequence is similar to the sequence in **Part 1**. You have two requests to the identity provider (win7pro.samlssosample.net in the previous figure). The difference is that because the first response from the server contains the **WWW-Authenticate: Negotiate** header, the browser invokes a service request to the AD DS to obtain the ticket.

If you track the request on the network interface of your Windows client machine or on the AD DS machine, you can see the request between the client Microsoft Windows and the AD DS.

In the next figure, you see **TGS-REQ**, which is the request from the client (192.168.1.101) to the AD DS (192.168.1.198) to obtain the Kerberos ticket. The Server Name is the Service Principal Name that is configured in the AD DS: **HTTP/win7pro.samlssosample.net**.
In the next figure, you can see TGS-REP, which is the response that the AD DS sent back to the client and that contains the Kerberos ticket.

After the browser receives the Kerberos ticket, it sends the ticket to the identity provider in the Authorization header of the request, as shown in the following figure.

### Conclusion

In this tutorial, you implemented a complete and secure SSO architecture, deployed to IBM Cloud. You integrated the SSO solution with the Microsoft Active Directory. You configured SPNEGO web authentication so that users do not have to provide their credentials.

By completing this tutorial series, you can now design and implement an SSO solution based on the SAML assertion, across multiple domains. You can integrate it with your existing security...
infrastructure, if you are using an LDAP user registry. You can even deploy your application in a hybrid cloud, depending on your needs.

Acknowledgments

The authors thank Carlo Randone and Pavel Travnik for reviewing this article and providing their comments.
Related topics

- WebSphere with a side of SPNEGO
- Single sign-on for HTTP requests using SPNEGO web authentication
- Configuring SPNEGO authentication in Liberty
- Getting started with the IBM Cloud
- IBM Cloud Documentation
- WebSphere Liberty (WASDev) Developer Center
- IBM Cloud Developer Center

© Copyright IBM Corporation 2017
Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)