IdP Strategy - ADFS

Description

Active Directory Federation Services (ADFS) from Microsoft provides users with SSO capabilities in federated environments. ADFS uses a "claims-based" access control authorization model.

See:

- http://msdn.microsoft.com/en-us/library/bb897402.aspx
- http://en.wikipedia.org/wiki/Active_Directory_Federation_Services
- https://spaces.at.internet2.edu/display/InCFederation/Using+Other+Software
- https://wiki.shibboleth.net/confluence/display/SHIB2/MicrosoftInterop
- http://adfstoolkit.org/

Fact Finder

Scott Koranda and Alex Chalmers

Example Deployments

Ball State University (BSU), an InCommon Participant, is an example of an ADFS deployment used for federation in the InCommon context.

Support for the Recommended Technical Basics for IdPs, including the ability to consume metadata

Metadata Consumption

ADFS does not support direct consumption of InCommon Federation metadata. The product focuses on a more point-to-point federation approach where the details for each entity are entered one at a time using a graphical user interface (GUI) better suited for point-to-point federation than a large identity federation like InCommon. To help work around this limitation ADFS deployers have developed pysfemma, "a script that parses a (Shibboleth) federation metadata XML content and creates a pool of metadata files and a powershell script in order to automatically configure and update an Active Directory Federation Services STS (Security Token Service)." We note, however, that there are issues when using pysfemma such as all entities being deleted and then re-populated with each run, and that a run to consume the InCommon metadata can take more than two hours.

As noted above, ADFS does consume per-entity metadata directly without the use of pysfemma or other third-party tools using a graphical user interfaced (GUI) better suited for point-to-point federation.

Scope in Metadata

ADFS does not support the notion of scope as it is normally used in the SAML context. Specifically when ADFS is acting in an SP role it does not check the scope on (scoped) attributes in an assertion from an IdP. There are, however, techniques that ADFS operators can use in combination with pysfemma and other scripts to approximate the handling of scope. See for example this email thread.

Please note that the above concerns ADFS acting in an SP role and not an IdP role, which is the primary focus here. We include these details in order to answer questions posed by readers.

When acting as an IdP ADFS can assert scoped attributes (or claims), such as eduPersonPrincipalName.

X.509 Certificates in Metadata

Older versions of ADFS would not accept 2 relying parties having the same X.509 signing certificate. This is a problem since some SPs operated by the same entity, often logical SPs sharing the same base Shibboleth installation, use the same X.509 certificate in metadata. ADFS update 3 (also known as 2.1 2012) fixes this issue. Note that if the X.509 certificates are part of a PKI infrastructure (not self-signed) ADFS does check the CRLs and other parts of the certificate chain. If the certificates are self-signed they must still be time valid (not expired).

SAML Protocol Endpoints

ADFS does not support SAML 1.1 but does support SAML2 authentication requests via the SAML V2.0 HTTP-Redirect binding. ADFS can protect all endpoints with SSL/TLS.

Support for Attribute Release

With ADFS the accompanying Active Directory (AD) deployment is configured automatically as an attribute store, so it is only necessary to run through a "wizard" to set up attributes. It is also straightforward to use the AD LDAP functionality as an attribute store. ADFS can also pull/resolve attributes from an associated Microsoft SQL Server easily. Other SQL data stores can also be used but require the writing of a plugin and some scripts.

It is, however, important to note that ADFS releases attributes in an undefined format type, whereas in the InCommon Federation relying parties usually anticipate a format type of URI. The ADFS operator can change the format type from undefined to URI relatively simply, but this must be done for every claim (attribute) that it is issuing. It is not uncommon for this to be scripted. The ADFS claim/attribute rule language has the reputation of being arcane and not well documented.

Support for Entity Attributes/categories (e.g., R&S)

ADFS does not support (without use of third-party tools) entity attributes or categories, such as the InCommon R&S entity category. Configuration is per relying party. So for example it is not possible to configure ADFS to release an attribute (claim) such as eduPersonPrincipalName to a set of SPs based on the SPs being tagged in the consumed (InCommon) metadata with the R&S entity category.

An ADFS operator, however, can use third-party tools such as pysfemma and a combination of PowerShell tools to help configure ADFS so that it effectively does release attributes to SPs based on an entity category like R&S.

Support for Multiple Authentication Contexts for Multi-Factor Authentication and Assurance

ADFS does not support multiple authentication contexts that would allow for easy integration of MFA, InCommon Assurance, and "step up" authentication flows.

Support for ECP (Enhanced Client or Proxy)

ADFS does not support ECP. It only supports the POST, redirect, and artifact bindings and does not support SOAP/PAOS.

Support for User Consent

ADFS does not support "user consent" in the context of a "uApprove like" approach.

Expertise Required

ADFS as a component in a Microsoft Windows server deployment integrates well and does not require expertise beyond that one expects for Windows administrators operating for example AD. Considerable expertise, however, is required when integrating ADFS into a higher education federation like InCommon since it is necessary to "translate" from the language and "culture" of the federation to that of the ADFS environment, and to be able to understand how to use tools like pysfemma to work around some of the limitations of ADFS. This expertise is often hard won and not found with the more "traditional" ADFS administrators.

Resources Required

ADFS is easy to deploy and integrate in an existing AD environment for experienced Windows administrators and does not require any special resources. Note, however, that when federating ADFS with more than 100 relying parties (which is expected in InCommon) it is necessary to use a Microsoft SQL Server database rather than the Windows "internal" database used by default. The SQL server should itself be in a high availability (HA) configuration if ADFS and AD and other components are set up in a HA configuration.

Upkeep and Feeding Required

Once deployed, configured, and integrated with InCommon via pysfemma or the like, ADFS has a reputation for being stable and solid. It is known to scale well.

Applicable Environments

ADFS is used when the existing IdMS is Microsoft Active Directory (AD).

Pros / Benefits

Using ADFS is a natural approach for Microsoft Windows shops and the basic deployment and configuration will be straightforward for experienced Windows administrators. Once deployed ADFS is known to be stable and scale well. High availability (HA) configurations use standard Microsoft Windows server techniques and approaches for HA.

Cons / Risks

Integration with large identity federation environments like InCommon requires additional scripts and supporting infrastructure like pysfemma and a considerable learning curve to acquire the necessary knowledge to "bridge the gaps" between the InCommon environment and the peer-to-peer model or approach that is the norm for ADFS federation.