MetaData

Metadata is a heavily overloaded term, but as used in the federated identity space, it refers to configuration data that can be used to simplify the provisioning of a new partner into a federated identity deployment. Typically it exists in XML form, at least for publishing/interchange. There are several metadata schemas defined by different specifications or software, but Shibboleth is currently designed around the SAML 2.0 Metadata schema now standardized by OASIS.

Shibboleth moved away from its own less functional (but admittedly simpler) format in favor of the SAML 2.0 format, which has been profiled for use with SAML 1.x (the legacy profile is finally nearing standardization as of Summer 2006). In this fashion, Shibboleth can support existing deployments of SAML 1.1 and SAML 2.0 in the future with a single format. We have also profiled it for use with Microsoft's ADFS product.

For specific technical profile information to interoperate with the Shibboleth software, refer to the ShibbolethMetadataProfile.

Use (and non-use) of Metadata

Shibboleth, in its current state, does not offer any tools to import or export SAML metadata. Rather, it consumes the XML directly as a configuration mechanism that enumerates the set of trusted partner sites and tells the software how to communicate securely with them. The IdP software has support for communicating with "unknown" SPs to some degree, as long as the BrowserPOST profile is used. However, there is currently no facility in the SP for accepting assertions from "unknown" IdPs. This will trigger the relatively common "metadata lookup" error.

The IdP consumes metadata by looking for entities that act in SP roles. Conversely, the SP consumes metadata by looking for entities that act in IdP roles. In other words, each type of provider needs metadata about its opposite.

At this time (and this is important to note), providers do **not** configure *themselves* using their own metadata. That is, an IdP does not determine how to behave based on metadata about itself, nor does an SP. So if you're having problems with your own software's behavior, it's probably not because the metadata you gave it is wrong. But if you're having problems accepting data from or communicating with another site, it may be a metadata problem.

API vs. Implementation

The API for obtaining information about sites at run-time is built around the SAML 2.0 metadata schema in terms of the interface design and hierarchy (entities->entity->roles->keys/endpoints, etc.) However, the **implementation** can obtain data from anywhere in any form, as long as the response to lookup requests emulates the proper information. This allows the older metadata format from earlier releases to be supported, for example.

That said, as of ShibOnedotThree, only a single metadata plugin implementation is supplied (and we know of no other implementations in existence). Metadata is obtained out of band and stored in local XML files, usually in aggregated groups based on federations. This only sense in which federations "matter" to the software is in how sites are grouped together, but this is arbitrary. Deployers can install as many sources of metadata as required, even to the extent of defining separate files for each partner, although this is harder to manage.

It is *possible* that ShibTwodotZero MAY explore alternate mechanisms, including dynamically obtaining information directly from other sites, but this is not a certainty. In any event, a stable API will be documented so that other developers can provide value-added implementations to improve scalability.

Bootstrapping Trust

From an API point of view, there is no "questioning" the information served up. It is implicitly trusted, so any rules for evaluating the information have to be supplied inside the plugin, or implemented in some out of band way. This is a particularly important point because of the use of metadata to make TrustManagement decisions about keys. By implicitly trusting metadata, the bootstrap question is addressed. Metadata itself (however it is implemented) bootstraps the TrustManagement layer.