Shibboleth 2:
A Guide for Deployers

Scott Cantor
cantor.2@osu.edu
Internet2 / The Ohio State University

Outline

- Introduction to Shibboleth and Related Topics
- Software Architecture
- Deploying an Identity Provider
- Service Providers
- Futures

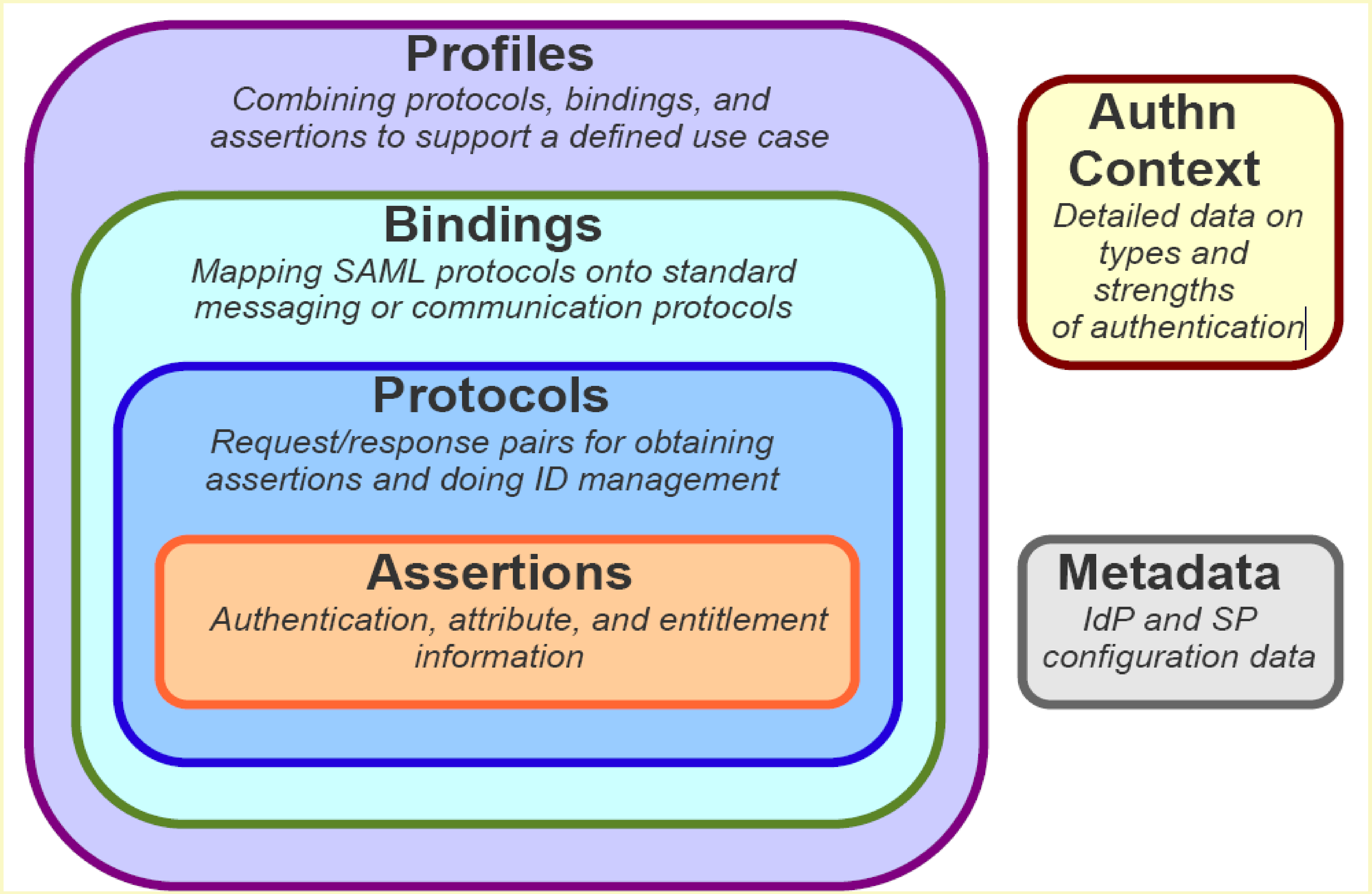
What is Shibboleth?

- Began life as an experimental project in standards-based inter-organizational access control, primarily in the library access management space.
- Stable, second-generation SSO platform for web applications:
  - Open Source (Apache-licensed)
  - Standards-Based (de jure, not de facto)
  - Multi-Protocol (SAML 2, SAML 1, WS-Federation RP)
  - Federated (Multi-Domain)
  - Attribute-Based, Privacy-Enabling

What is SAML?

- Security Assertion Markup Language
- Application of XML for the exchange and use of security-related information
- OASIS standard since 2003
- (Final?) 2.0 revision approved in 2005, integrating capabilities from several federation efforts and profiles, including Shibboleth
- Widely adopted commercially (now including Microsoft), less so non-commercially due to XMLitis

SAML 2.0 Structure


Profiles
Combining protocols, bindings, and assertions to support a defined use case
Bindings
Mapping SAML protocols onto standard messaging or communication protocols
Protocols
Request/response pairs for obtaining assertions and doing ID management
Assertions
Authentication, attribute, and entitlement information
Authn Context
Detailed data on types and strengths of authentication
Metadata
IdP and SP configuration data

Why Federations?

- "Noun" usage somewhat unique to higher education
- (Lack of) "Vision" of companies behind standard was to enable point to point exchanges, an "identity VPN"
- Shibboleth project goal was to scale to hundreds of organizations, potentially thousands of services
- Federations evolved out of a need for policy and technology to support NxN "mesh" of SAML-enabled peers
- The "I" in the SAML equivalent of a PKI

Federations

- Can range from the very simple to the very complex (can you guess which ones do better?)
- Commonly organized around geo-political boundaries
- Policy:
  - Membership obligations
  - Legal agreements to enforce obligations
  - Identity management obligations
- Technology:
  - Technical standards and attribute definitions
  - Enumerating participants and facilitating reliable operations
  - Brokering trust in the keys used at runtime for authentication and encryption

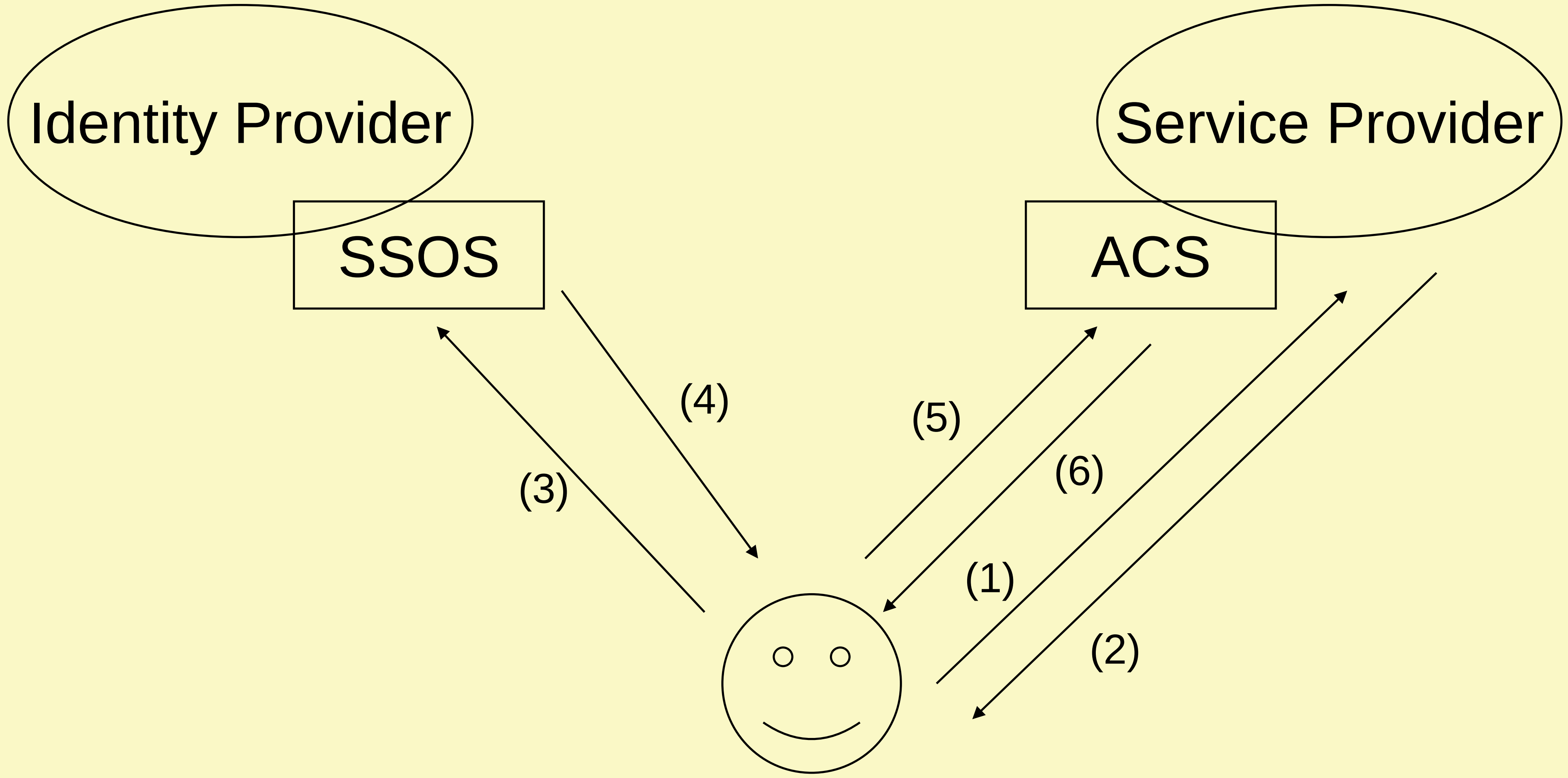
Metadata

- XML document format defined in SAML
- Used by Shibboleth software to:
  - identify the systems allowed to request or provide authentication
  - identify capabilities and endpoints to allow successful interoperation across multiple protocols
  - identify public keys belonging to particular trusted systems
- Basis of virtually all runtime software behavior
- Metadata often signed by federation keys that act as trust roots much like CAs were intended to

Other Terms

- Identity Provider – the login service
  - Single Sign-On Service – place you're sent to login by SP
  - Attribute Service – SOAP endpoint for attribute queries
- Service Provider – the application end
  - Assertion Consumer Service – place for response to SP
- entityID – unique identifiers for IdPs and SPs
- discovery – figuring out which IdP to use

Basic Example



1. HTTP Request to SP
2. **<samlp:AuthnRequest> encoded in URL Redirect**
3. **<samlp:AuthnRequest> passed to SSO Service via HTTP GET**
4. **<samlp:Response> in HTML FORM (action=POST) containing <saml:Assertion>**
5. **HTTP POST to Assertion Consumer Service**
6. HTTP Response

Complications

- Discovery is assumed / out of scope, but some possible mechanisms are defined
- Example assumes Redirect/POST bindings, others exist and can be mixed/matched, smaller implementations unlikely to support them all
- Encryption of assertion and inclusion of attributes can be adjusted in combination to achieve desired flow/compatibility/security
- Metadata exchange usually assumed to be out of band, though not required to be

Why Shibboleth?

- Federated access to external/outsourced services
- Single infrastructure for a larger range of needs
- Identity alone is insufficient, and LDAP requires additional effort in some cases
- Application portability
- Commercial interoperability
- Deployable for large-scale federated use

Why Not Shibboleth?

- IdP isn't self-contained or plug and play
- Java skills not necessarily common
- SP is NOT designed for developers, but for administrators with necessary rights
- SP will NOT provide an application API; if that's your perspective on SSO, it's not a good fit
- Configuration and operation is heavily XML-based

Commercial Alternatives

Shibboleth

- End to end, like the Internet's supposed to be
- Metadata-driven, focused on scaling of basic trusted exchanges
- Integration with infrastructure
- Loose coupling between identities and your application's use of them

Commercial

- Gatewayed, like the Internet actually is
- GUI-driven, focused on explicit "deep" relationships
- Sell you their infrastructure
- Tightly controlled account linking behavior

Federating Opportunities

- Outsourced services
  - Financial Aid / Scholarship information
  - Travel Reservations
  - Discounted Purchasing
- Government Agencies (NIH, NSF)
- Grids
- Collaboration platforms
- iTunesU
- Library resource access

FAQs

- Which version should I start with?
  - Always use the latest version available
- Does the new version interoperate with older versions?
  - Completely, dating back to 1.3
- Do I have to use SSL?
  - Only if you care about security
- How do I get started?
  - If you're playing, there's www.testshib.org
  - If you're serious, you need both an IdP and SP
- Does a test system need to be accessible, registered in DNS, etc?
  - New defaults assume no back-channel between IdP and SP, so all contact is via your browser

Skill Sets

Identity Provider

- Administration of web server / OS
- XML, XML, XML, and some XML
- Java web deployment
- Java development if customizing
- Basics of SSL / PKI concepts
- Integration skills for authentication and attribute back-end(s)
- Understanding of session handling in a web environment

Service Provider

- Administration of web server, especially plugins/extensions
- XML, XML, and some XML
- Basics of SSL / PKI concepts
- Basics of web development environment (e.g. accessing headers, redirection)
- Understanding of session handling in a web environment

Prerequisites

Identity Provider

- Source(s) of authentication
- Source(s) of attributes
- Means of managing policies on attribute release
- Means of managing metadata for SPs
- At least one SP you control for testing

Service Provider

- Supported web server with at least one hosted application
- Means of managing metadata for IdP(s)
- Strategy for IdP discovery

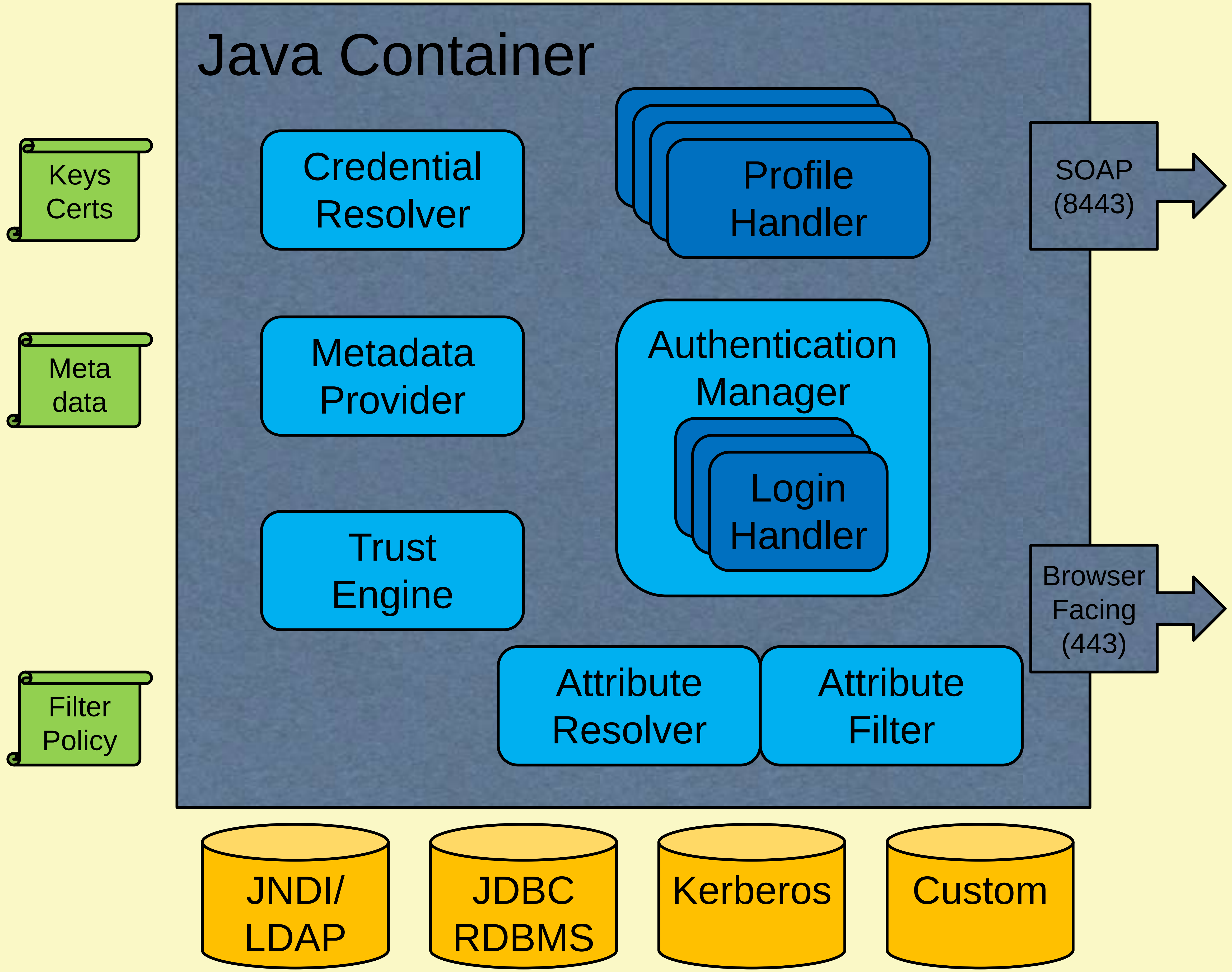
Resources

- http://shibboleth.internet2.edu/
- http://shibboleth.internet2.edu/shib-which-version.html
- http://shibboleth.internet2.edu/downloads.html
- http://shibboleth.internet2.edu/lists.html
- https://spaces.internet2.edu/display/SHIB2/Home
- http://www.incommonfederation.org/
- http://net.educause.edu/CampusArchitecturalMiddllewarePlanning(CAMP)Workshops/1607

Software Architecture

Identity Provider

- Written in Java, requires a container with Servlet 2.4 support
    - Requires very non-buggy JAXP parser, often causes problems with commercial containers
- "Yet another multi-protocol software platform"
    - Spring-based component framework and extension model
- Does NOT include database/directory or identity management tools
- Has no UI / direct user controls at the moment


Java Container
Keys Certs
Meta data
Filter Policy
Credential Resolver
Profile Handler
SOAP (8443)
Metadata Provider
Authentication Manager
Login Handler
Trust Engine
Browser Facing (443)
Attribute Resolver
Attribute Filter
JNDI/ LDAP
JDBC RDBMS
Kerberos
Custom

Profile Handlers

- Implement protocols for SSO, attribute query, eventually logout, diagnostics, others
- SAML 2.0
    - Browser SSO Profile
    - Attribute Query
- SAML 1.1
    - Shibboleth SSO Extension
    - Attribute Query

Authentication Manager

- Incoming request determines selection of a LoginHandler
  - Requested authentication class
  - IsPassive
  - ForceAuthn
- LoginHandler interacts with user to authenticate user
  - PreviousSession – handles SSO based on previous login
  - UsernamePassword – simple JAAS-backed option
  - REMOTE_USER – relies on container / web server

Attribute Resolver

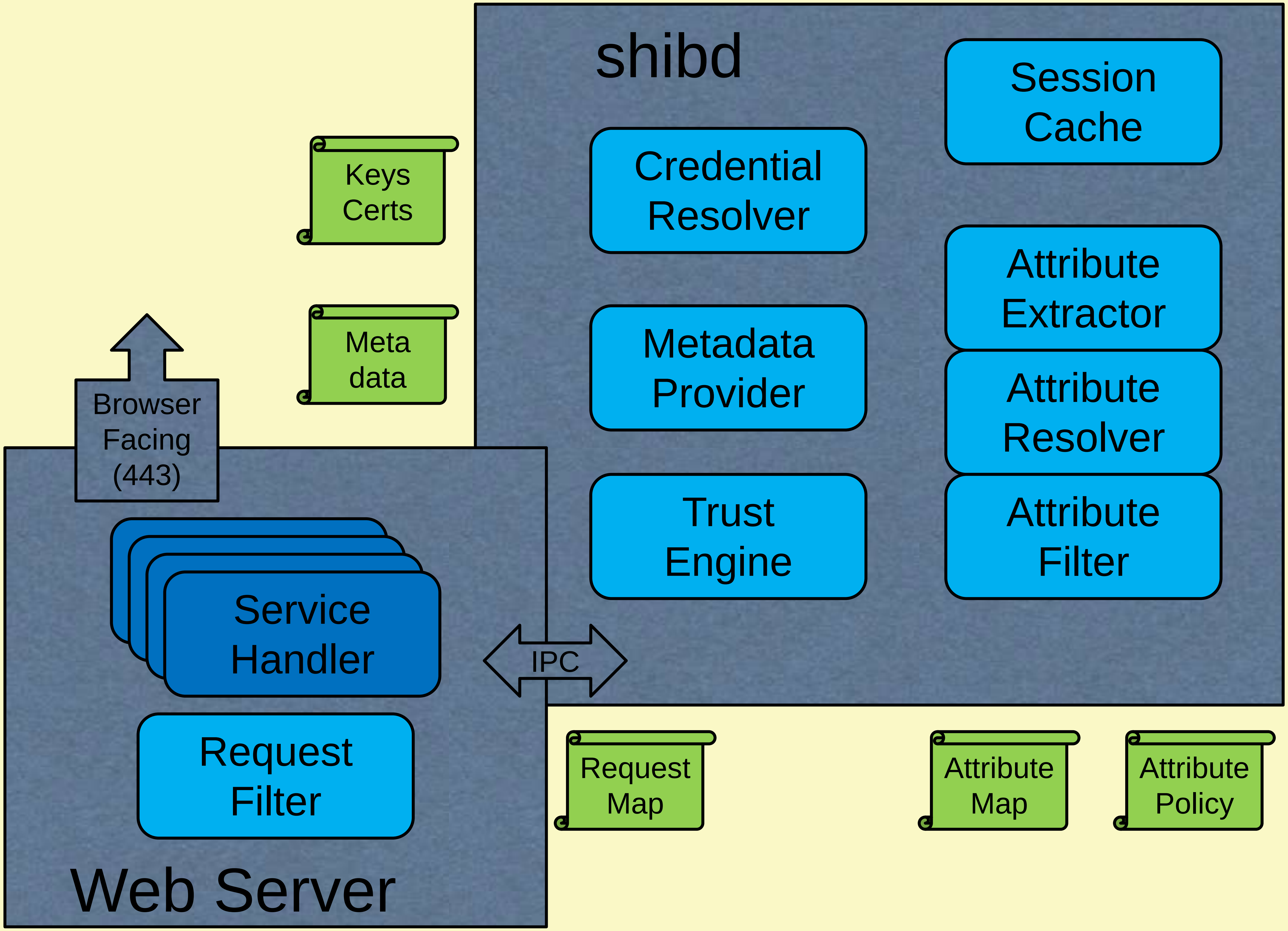
- Directed dependency graph of plugins that performs like a virtual directory
- Data Connectors – load attributes from data stores
- Attribute Definitions – transform/combine/expose connector attributes for encoding into SAML
- Principal Connectors – reverse map SAML subjects into local users (for queries)
- Connectors can failover for redundancy or consolidating distributed data stores

Attribute Filter

- Policy engine for rules governing which attributes are released to SPs
- Policies can apply based on a variety of criteria, and rules are run to permit or deny each attribute value
- Policies and rules for:
  - Requesting SP
  - Comparisons or regular expressions over values
  - User identity
  - Authentication method
  - Other attribute values (e.g. FERPA release flags)

Service Provider

- Written in C++ for Apache, IIS, or Sun web servers
  - Apache can front-end other web servers: Java containers, Zope, etc.
- "Yet another multi-protocol software platform"
  - DLL-based component framework and extension model
- Does NOT include database/directory or identity management tools


shibd
Session Cache
Credential Resolver
Attribute Extractor
Metadata Provider
Attribute Resolver
Trust Engine
Attribute Filter
Keys Certs
Meta data
Browser Facing (443)
Service Handler
IPC
Request Filter
Web Server
Request Map
Attribute Map
Attribute Policy

General Design

- No formal API; logged-in-user information available through protected headers or env. variables
    - Requires identity to be externalized from application
    - Applications can interact with SP using declarative configuration or simple redirection protocols
- Module inside web server filters requests, applies policy, attaches data from active sessions
- Services requests to special URLs for processing SAML messages and performing functions on behalf of applications
- Heavy lifting and state management delegated to a separate process

Service Handlers

- Implement protocols for SSO, logout, diagnostics, others
- SAML 2.0
  - Browser SSO Profile
  - Browser SLO Profile
  - NameID Management Profile
- SAML 1.1 Browser SSO Profile
- WS-Federation Passive Requester Profile
- Metadata generation, status, and session dump

Sessions

- Sessions tracked by cookie specific to SP, used to recover user identity data and attach to requests, honor logout, enforce timeouts
- Applications use or ignore SP session as they choose, choices vary widely by scenario
- Storage plugins provided for in-memory, ODBC, and memcache

Attribute Handling

- Extensive plugin architecture for turning SAML assertion data into attributes for application to consume
- Environment or header names all configurable
- Data can be filtered analagously to IdP (controls "release" to applications)
- Additional attributes can be "resolved" via plugins that query SAML authorities or local sources

Deploying an Identity Provider

A Real-World Example

- Access to a federated wiki run by Internet2
  - Requires the eduPersonPrincipalName attribute to identify and auto-provision users
  - Sets email address and a display name locally if available from IdP
- Demonstrate an example using LDAP as an authentication/attribute store
- Identify some deployment choices as we go along

Basic Steps

- Install / setup Java container and web server if necessary
    - Tomcat or Apache + Tomcat
- Download and install IdP software
- Set entityID, publish IdP metadata, supply SP metadata
- Configure Authentication
- Configure Attribute Resolution
- Release attributes to SP

Container / JVM

https://spaces.internet2.edu/display/SHIB2/IdPApacheTomcatPrepare

- Endorse XML libraries in container
- Install JCE stub as JVM security extension
- Setup SSL connectors on 443/8443
- Adjust JVM heap size
- Create context deployment fragment

Initial Installation

- Download and unzip…
  - UI customizations currently maintained in source tree, applied with reinstallation
- Run install.sh or install.bat script
  - installation location (default highly advised)
  - hostname
  - password for keystore
- Sanity check, start Tomcat:
  - https://hostname/idp/profile/Status
  - https://idp.example.org:8443/idp/profile/Status

Add SP Metadata to IdP

- This particular SP hosts and self-signs its metadata, obtaining the key is "out of scope"
- Install a reference to its metadata, verified by the signing key
- Most SP-oriented maintenance performed with relying-party.xml
- Typically rely on the DefaultRelyingParty, but if you need specific rules for an SP, remember that nothing is inherited from the default element settings

Add SP Metadata to IdP

```
<MetadataProvider id="URLMD" xsi:type="FileBackedHTTPMetadataProvider"
    xmlns="urn:mace:shibboleth:2.0:metadata"
    metadataURL="https://spaces.internet2.edu/shibboleth"
    backingFile="C:\opt\shibboleth-idp/metadata/spaces-metadata.xml">
    <MetadataFilter xsi:type="ChainingFilter">
        <MetadataFilter xsi:type="RequiredValidUntil"
                        maxValidityInterval="0" />
        <MetadataFilter xsi:type="SignatureValidation"
                        trustEngineRef="spacesMetadataTrustEngine"
                        requireSignedMetadata="true" />
        <MetadataFilter xsi:type="EntityRoleWhiteList">
            <RetainedRole>samlmd:SPSSODescriptor</RetainedRole>
        </MetadataFilter>
    </MetadataFilter>
</MetadataProvider>
…
<security:TrustEngine id="spacesMetadataTrustEngine"
    xsi:type="security:StaticExplicitKeySignature">
    <security:Credential id="spacesCredentials"
    xsi:type="security:X509Filesystem">
        <security:Certificate>
         C:/opt/shibboleth-idp/credentials/spaces.crt
        </security:Certificate>
    </security:Credential>
</security:TrustEngine>
```

Add IdP Metadata to SP

- Procedurally could be similar, with the IdP hosting a self-signed file with OOB key exchange
- With lack of connectivity to example machine, the file will be copied manually to the SP and referenced locally
- Copy /opt/shibboleth-idp/metadata/idp-metadata.xml to /etc/shibboleth on SP
- Add reference to shibboleth2.xml:

```
<MetadataProvider type="XML" path="idp-example-metadata.xml"/>
```

Authentication

- Primary settings for authentication in handler.xml
- Default configuration is for authentication by container / web server, switch to UsernamePassword handler, which points to a JAAS configuration file

```
<!-- Username/password login handler -->
<LoginHandler xsi:type="UsernamePassword"
     jaasConfigurationLocation="file://C:\opt\shibboleth-idp/conf/login.config">
         <AuthenticationMethod>
         urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport
         </AuthenticationMethod>
</LoginHandler>
```

- LoginHandler identifies the technical authentication methods supported by the handler
- Can be annotated as to whether they support advanced SAML features

JAAS Configuration

- For simple name/password handling, the included handler relies on JAAS
  - Good composition options and flexibility of back-ends
  - Atrociously bad error handling
- LDAP example already in place, just uncomment and adjust settings

```
edu.vt.middleware.ldap.jaas.LdapLoginModule required
   host="idp.example.org"
   port="10389"
   base="ou=people,dc=example,dc=org"
   ssl="false"
   serviceUser="uid=admin,ou=system"
   serviceCredential="secret"
   userField="uid";
```

Attribute Resolution

- eduPerson / LDAP examples already in attribute-resolver.xml
- Need at least one DataConnector to pull raw bits from directory or database
- Need AttributeDefinition per-attribute to expose outside of IdP
    - Variety of plugins available for non-trivial transforms and scripting to generate attributes
- Each AttributeDefinition needs AttributeEncoder for each SAML version to generate wire format

LDAP Data Connector

- Uncomment and adjust
- Returns all attributes in entry by default
- Can build filter around variety of request data, including other internal attributes

```
<!-- Example LDAP Connector -->
<resolver:DataConnector id="myLDAP" xsi:type="LDAPDirectory"
        xmlns="urn:mace:shibboleth:2.0:resolver:dc"
        ldapURL="ldap://idp.example.org:10389"
        baseDN="ou=people,dc=example,dc=org"
        principal="uid=admin,ou=system"
        principalCredential="myServicePassword">
 <FilterTemplate>
        <![CDATA[
        (uid=$requestContext.principalName)
        ]]>
 </FilterTemplate>
</resolver:DataConnector>
```

eduPersonPrincipalName

- Can be computed by appending a "scope" to a directory attribute, using the "Scoped" plugin
- Here, pull directly from LDAP using the "Prescoped" plugin

```
<resolver:AttributeDefinition id="eduPersonPrincipalName"
        xsi:type="Prescoped" xmlns="urn:mace:shibboleth:2.0:resolver:ad"
        sourceAttributeID="edupersonprincipalname">

        <resolver:Dependency ref="myLDAP" />

        <resolver:AttributeEncoder xsi:type="SAML1ScopedString"
               xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
               name="urn:mace:dir:attribute-def:eduPersonPrincipalName"/>

        <resolver:AttributeEncoder xsi:type="SAML2ScopedString"
               xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
               name="urn:oid:1.3.6.1.4.1.5923.1.1.1.6"
               friendlyName="eduPersonPrincipalName" />

</resolver:AttributeDefinition>
```

mail

- Pulled directly from directory using "Simple" plugin

```
<resolver:AttributeDefinition id="email" xsi:type="Simple"
      xmlns="urn:mace:shibboleth:2.0:resolver:ad"
      sourceAttributeID="mail">

      <resolver:Dependency ref="myLDAP" />

      <resolver:AttributeEncoder xsi:type="SAML1String"
            xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
            name="urn:mace:dir:attribute-def:mail" />

      <resolver:AttributeEncoder xsi:type="SAML2String"
            xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
            name="urn:oid:0.9.2342.19200300.100.1.3"
            friendlyName="mail" />

</resolver:AttributeDefinition>
```

displayName

- Computed within IdP from the givenName and sn directory attributes using the "Template" plugin

```
<resolver:AttributeDefinition id="displayName"
xsi:type="Template" xmlns="urn:mace:shibboleth:2.0:resolver:ad">

       <resolver:Dependency ref="myLDAP" />

       <resolver:AttributeEncoder xsi:type="SAML1String"
             xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
             name="urn:mace:dir:attribute-def:displayName" />

       <resolver:AttributeEncoder xsi:type="SAML2String"
             xmlns="urn:mace:shibboleth:2.0:attribute:encoder"
             name="urn:oid:2.16.840.1.113730.3.1.241"
             friendlyName="displayName" />

       <Template><![CDATA[${givenname} ${sn}]]></Template>
       <SourceAttribute>givenname</SourceAttribute>
       <SourceAttribute>sn</SourceAttribute>

</resolver:AttributeDefinition>
```

Release the Attributes

- Default is to release nothing, so a policy in attribute-filter.xml is added for the SP

```
<AttributeFilterPolicy>
        <PolicyRequirementRule
               xsi:type="basic:AttributeRequesterString"
               value="https://spaces.internet2.edu/shibboleth" />

        <AttributeRule attributeID="eduPersonPrincipalName">
               <PermitValueRule xsi:type="basic:ANY" />
        </AttributeRule>

        <AttributeRule attributeID="mail">
               <PermitValueRule xsi:type="basic:ANY" />
        </AttributeRule>

        <AttributeRule attributeID="displayName">
               <PermitValueRule xsi:type="basic:ANY" />
        </AttributeRule>
</AttributeFilterPolicy>
```

A Note About NameIDs

- Shibboleth favors using SAML Attributes in place of the SAML NameID construct that appears in the assertion subject
- Commercial interop often requires reconfiguration to allow use of a non-transient NameID format
- Configured and released exactly like Attributes; only difference is the type of AttributeEncoder used

Clustering

- DNS is not a redundancy tool
- IdP session management is private to IdP unless container-managed authentication is used
- Officially supports clustering using Terracotta, in use by several large deployers
- At least one example of JBoss Groups known
- Back-channels make sticky sessions an incomplete solution, though possibly usable in some contexts

Service Providers

Basic Setup

- Binary install, generates self-signed keypair
- In shibboleth2.xml:
  - Set entityID
  - Supply IdP metadata source(s) and enter an IdP's entityID in the default <SessionInitiator>
- Extract metadata from /Shibboleth.sso/Metadata to enroll in federation or supply to IdP
- Define protection rules for content

More Setup

- Customizing error templates: don't be **that** guy
- Discovery
  - In principal, a per-SP exercise, but you might choose to run a shared DS on behalf of campus apps
  - DS software available from Internet2 (Java) and SWITCH (PHP), or there's always the simple web page of links
- Customizing attribute handling
  - Uncomment or create mapping definitions and choose header names via attribute-map.xml
  - Map selected attributes into REMOTE_USER if needed

Session Strategies

- Simple/small apps
  - protect entire scope of app with SP session
  - consume REMOTE_USER and attributes on each request
  - allows easy portability to any proper SSO design
- More complex scenarios
  - usual stategy is to protect an "entry/setup" script with SP session to establish a typical application session
  - can time-out SP session quickly to save resources, or preserve for logout handling
  - requires application changes for logout to work

Supporting a Campus

- Documentation Site:
  - Overview / FAQ
  - Formal documentation on supported attributes
  - Instructions and configuration files for campus and federated use
- Message to deployers:
  - install software in the usual manner
  - follow instructions, copying in campus-specific files
  - send registration request, with certificate
  - test SP, check logs

Registration Approaches

- Email or web submission, the latter allows authenticated submissions of certificates
- Typical information:
  - Service name / description
  - Contact information
  - Technical/platform details
  - Requested attributes
  - Web hostname(s)
  - Certificate
  - Verifiable phone number
- Rigor of processes depend on institutional culture

External Services

- Existing federation SPs:
  - Timeline likely driven by your ability to get approval to release attributes and ensure users are supported
  - Technical effort / involvement often quite low
  - For some services, might be literally the 5 minutes to add a filter policy
- A typical outsourced SP coming online:
  - You won't be the bottleneck
  - If it's not in the RFP/contract, chances are low
  - Use of a federation strongly advisable

Metadata Management

- External use
  - Federations rely on MAD – if I ask for bilateral vetting, others can ask me for it
  - Pay a relative pittance and you solve a bunch of ongoing management and security problems
- Internal use
  - Maintaining IdP metadata for SPs easy, but if you're in a federation anyway, just point them there
  - Maintaining SP metadata has a range of solutions from the very manual to the highly automated
  - Tools for this not a current project focus

Help Desk Implications

- Most problems either with authentication or application after initial testing
- Rule of thumb: if you're stopped at the IdP, treat centrally, otherwise application owner should take responsibility
- Strongly advise against SPs, especially externals, delegating triage responsibility to IdPs
- Biggest problem by far is application developers and owners refusing to own their part

Common Errors

- Clock skew
- Client address mismatches between IdP/SP
- Looping and other errors when SSL not used
  - could make this easier for people, somewhat stubborn about it
- Hostname issues causing failures at IdP
- IIS file permissions

Futures

Logout

- SP includes extensive support for SAML 2 logout features, but leaves UI to deployer
- IdP currently lacks logout support, likely to offer back-channel profile eventually
- Front-channel profile requires substantial UI/design study
- Note most SPs unlikely to support logout in the medium term
- Seems to be largely an intranet concern; consider the implications when logout is usually partial

SSO Delegation

https://spaces.internet2.edu/display/ShibuPortal/Home

- Similar to CAS proxying, but with additional requirements: privacy, federation, standards-based
- Starting point is to achieve SSO-level security via bearer tokens between sites, improve later
- Can be transparent to applications but not usable without explicit opt-in via SP configuration
- Policy controls at IdP likely to evolve

User Consent

- Typical use of Shibboleth relies on administrative control over attribute release
- Consequence? Useless for collaborative / one-off scenarios
- Solution? Default to user approval of attribute release:
  - Hide non-privacy-sensitive "technical" attributes users can't understand anyway
  - Disclose and require approval on initial access if "intelligible" attributes are being requested
  - Whitelist contractual arrangements where consent is OOB
- Major issue: requires SPs to disclose attribute requirements and federations to track them