Shibboleth 2.0 Overview

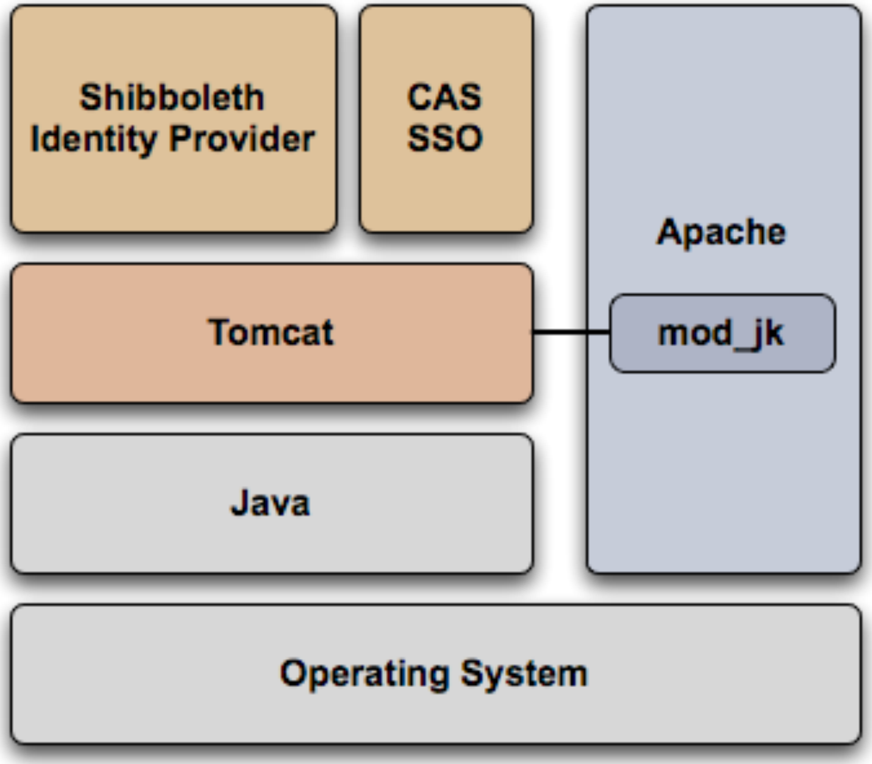
Serving Swiss Universities

Chad La Joie
chad.lajoie@switch.ch

Agenda

- New IdP Architecture & Features
- Upcoming 2.1 Shibboleth IdP Release
- New SP Features
- New Discovery Service
- Upcoming Installfest Events

IdP: Architecture



Shibboleth Identity Provider

CAS SSO

Apache

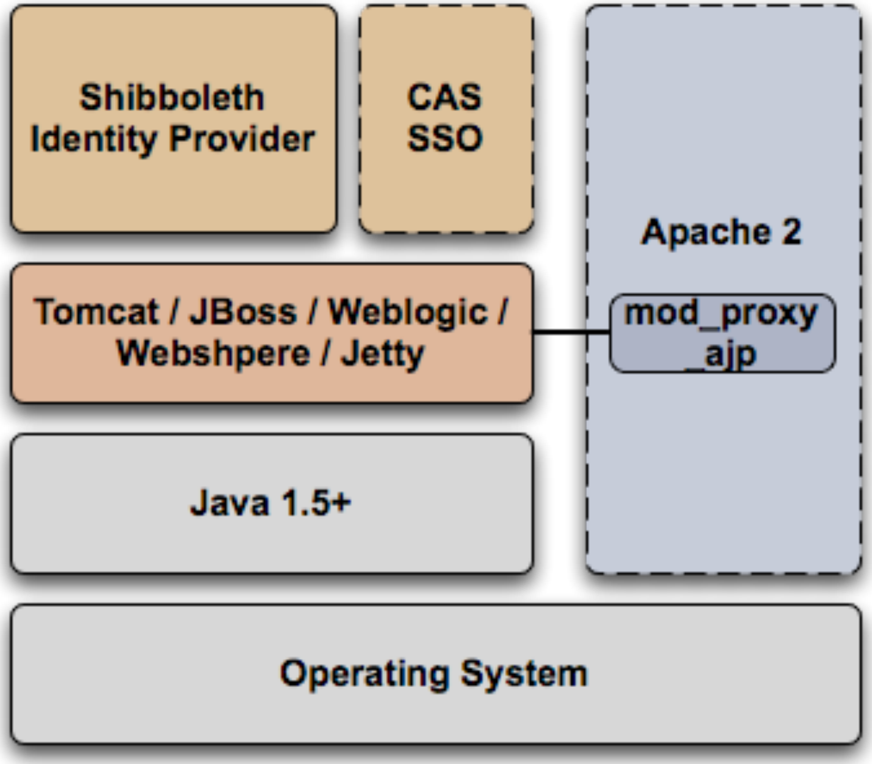
Tomcat

mod_jk

Java

Operating System

Shibboleth 1.3



Shibboleth Identity Provider

CAS SSO

Apache 2

Tomcat / JBoss / Weblogic / Webshpere / Jetty

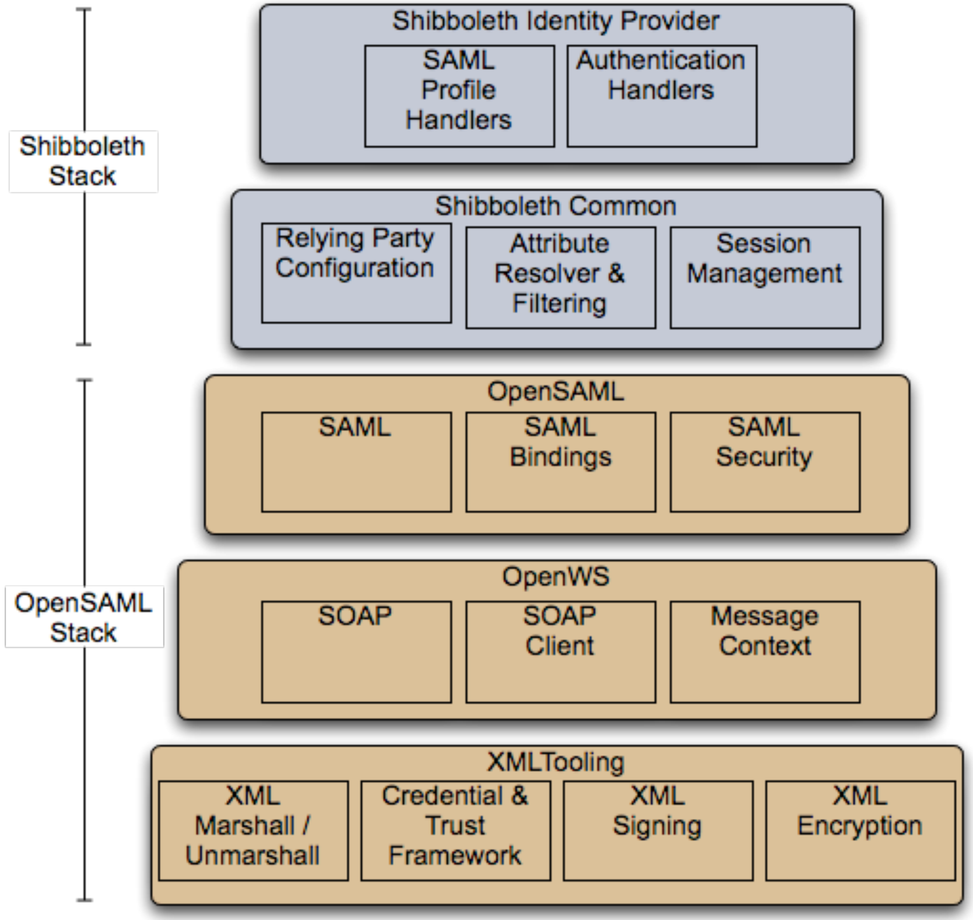
mod_proxy _ajp

Java 1.5+

Operating System

Shibboleth 2.0

IdP Architecture



- Shibboleth software composed of two stacks
- OpenSAML stack provides XML processing
- Shibboleth stack provides multi-protocol platform

IdP: New Features

- New installation process
- New configuration files
- SAML 2 support
- Authentication method support
- New attribute resolution and filtering engine
- Improved logging including access and audit logs

IdP: New Installation Process

- IdP installation process is still command line based
- New Features:
  - Generation of self-signed certificate for use with Attribute Authority, signing, and encryption
  - Generation of IdP metadata including scope and entity ID
  - Generation of relying-party.xml with populated entity ID and credential information
  - Very basic attribute resolver and filter policy
- The result is a configuration that can be used for SSO if the deployers add a metadata and authentication source.

IdP: New Configuration Files

- All configuration files located in $IDP_HOME/conf
  - *relying-party.xml*: per relying party controls
    - ❑ supported profiles, metadata, credentials, security settings
  - *attribute-resolver.xml*: attribute resolver configuration
  - *attribute-filter.xml*: attribute filtering policy
  - *logging.xml*: process, access, and audit log configuration
  - *login.config*: JAAS authentication policy
  - *handler.xml*: controls IdP endpoints
  - *internal.xml* & *service.xml*: only used for very advanced options
- Most configuration files may be reloaded
  - login.config, internal.xml, service.xml are the exceptions

IdP: New Configuration Files

- “Type” based configuration
  - <JNDIDataConnector> => <DataConnector xsi:type=“LDAP”>
  - Each type has its own configuration attributes/elements
  - Each type represents an extension point within the IdP
- Most configuration validated at start up time
  - Prevents IdP from starting with bad config but crashing weeks later when a request happens to trigger the mis-configured section

IdP: SAML 2 Support

- Configured in relying-party.xml
- Profiles:
  - Supported: SSO, Artifact Resolution, Attribute Query
  - Unsupported: Single Logout, NameID management & mapping
- Encryption:
  - Supported: NameID, Assertion
  - Unsupported: Attribute
- Single Sign-On defaults to attribute push with encryption
  - Requires SP key be embedded in metadata
- Attribute Query expected to be unused in most SAML 2 deployments

IdP: Authentication Method/Mechanism

- SAML 2 introduces the idea that service providers may request the method by which a user is authentication.
- A particular authentication method is represented by a URI
- The IdP maps all support authentication methods to concrete mechanisms (e.g. LDAP, Kerberos).
- So:
  - **Authentication Method**: A URI used by the SP to describe how it wants the user authenticated
  - **Authentication Mechanism**: The actual process used by the IdP to authentication the user
  - **Login Handler**: The IdP plugin that actually does the authentication

IdP: Authentication Method

- Each authentication mechanism may timeout separately
  - Controlled by `authenticationDuration` attribute, 30min default
- Login handlers may support one, or more, methods
  - Configured in handler.xml
  - Types:
    - ❑ Remote User: delegates authentication to container or external SSO system
    - ❑ Username/Password: username/password checked against LDAP or Kerberos
    - ❑ IP Address: checks IP address of client browser
- A default authentication method may be configured otherwise the IdP randomly chooses a method if the SP does not request one.

IdP: Attribute Resolver

- Configured in attribute-resolver.xml
- 4 kinds of things:
  - **Data connector**: pulls data from some data source (LDAP, RDBMS)
  - **Attribute definition**: creates attributes from environmental and data connector retrieved information
  - **Attribute encoder**: transforms attributes into XML
  - **Principal connector**: connects a SAML name identifier with a principal
- Connectors and definitions may get information from each other

IdP: Attribute Resolver - Data Connector

- Retrieves a set of attributes from a data store
- Connector may specify a failover connector that is invoked in the event of an error
- Information gathered by a data connector **never** leaves the resolver
- Included data connectors
  - Static: a statically configured set of attributes for every user
  - RelationalDatabase: pulls data from a SQL database
  - LDAPDirectory: pulls data from an LDAP directory
  - ComputedId: creates a value by hashing current state information
  - StoredId: creates and retrieves a value from a SQL database
  - Computed and Stored ID connectors are almost always used to create long lived name identifiers

IdP: Attribute Resolver - Attribute Definition

- Create **one** attribute with a unique ID and set of encoders
- Can specify a source attribute for incoming values
- Attributes created by a definition leave the resolver and may be released to a service provider
- Included Attribute Definitions:
  - Simple: provides values as-is from data connector
  - Scoped: adds a static scope to all values of an attribute
  - Prescoped: splits all values of an attribute in an value and a scope
  - PrincipalName: user’s login ID as value
  - PrincipalAuthenticationMethod: authentication method as value
  - Regex: keeps only the portion of a value that matches a regular expression
  - Script: runs a script to produce an attribute; default language: ecmascript
  - Mapped: maps one, or more, values to a different value
  - Template: creates value by filling in a template string
  - SAML 1 NameIdentifier: creates a SAML 1 NameIdentifier from all value
  - SAML 2 NameID: creates a SAML 2 NameID from all values

IdP: Attribute Resolver - Attribute Encoder

- New in Shibboleth 2
- Protocol and value-type specific
- SAML 1 Encoders
  - `SAML1String`: operates on string values
  - `SAML1ScopedString`: operates on scoped values
  - `SAML1Base64`: operates on binary blobs
  - `SAML1XMLObject`: operates on XMLObject values
- SAML 2 Encoders
  - `SAML2String`: operates on string values
  - `SAML2ScopedString`: operates on scoped values
  - `SAML2Base64`: operates on binary blobs
  - `SAML2XMLObject`: operates on XMLObject values

IdP: Attribute Resolver - Principal Connector

- New in Shibboleth 2
- Name format and, optionally, relying party specific
- Included Principal Connectors:
  - `Direct`: provides no mapping, assumes name identifier is already principal ID
  - `TransientId`: map from ID created by TransientID attribute definition
  - `StoredId`: map from ID created by StoredID data connector

IdP: Attribute Filtering

- New filtering policy language used by IdP and SP
- One policy file may have multiple policies
- Each policy contains
  - One requirement rules that triggers the application of the policy
  - One or more attribute rules that filter the value of an attribute
    - ❑ Attributes identified by the same ID given in the resolver configuration
    - ❑ Each attribute rule contains a permit value rule
    - ❑ Only values that are permitted are released
    - ❑ No way to expressly deny the release of value
- Policy requirement and permit value rules use functors to determine if they are active

IdP: Attribute Filtering: Rule Functors

- Boolean operations:
  - And, Or, Not, Any
- String or regular expression matching:
  - Attribute Issuer, Attribute Requester, Principal Name, Authentication Method, Attribute Value, Attribute Scope
- SAML metadata matching:
  - Attribute Issuer In Entity Group, Attribute Requester In Entity Group
- Misc:
  - Script

IdP: Improved Logging

- All IdP errors are logged, even those that occur before the logging framework is initialized
- Three types of logs; all located in $IDP_HOME/logs
  - idp-process.log – same as the IdP 1.3 log
  - idp-access.log – machine parsable apache style access log
  - idp-audit.log – machine parsable log providing transactional information: type of request, attributes released, etc.
- Logging configuration can be changed during runtime
- More performant logging framework; still shouldn't run production machines on debug

IdP: Upcoming 2.1.0 Release

- First, a word about version numbers:
  - Patch releases increment the third component of the version number and are bug fixes.
  - Minor releases increment the second component of the version number and represent new functionality with backwards compatibility.
  - Major releases increment the first component of the version number, represent new functionality, but generally are **not** compatible with any previous release.
  - No new major release is in the plans at this time.

http://shibboleth.internet2.edu/java-versioning.html

IdP: Upcoming 2.1.0 Release

- New Features:
  - **Subversion and HTTP URL Configuration File Locations**
    Read configurations files directly from Subversion or an HTTP(S) URL (like the resource registry).
  - **Resource Configuration File Filters**
    Perform some processing after the IdP has fetched a configuration file but before it loads the file. First filter to ship with the IdP replaces macros with values from a property file
  - **Attribute Filter DenyValueRule**
    Explicitly deny the release of an attribute to a service provider
  - **Support for New Scripting Languages**
    PHP, Python, Ruby, Beanshell (what 1.3 IdP used)
- Also includes some bug fixes and updated versions of dependant libraries.

SP: New Features

- SAML 2 support
- New management endpoints
- Additional server integration
- Performance and high availability

SP: SAML 2 Support

- Supported Profiles:
  - Single Sign On, Attribute Query, Artifact Resolution, Single Logout, NameID mapping, NameID management
  - NameID mapping, NameID management require user developed scripts to integrate with protected application(s)
- Encryption:
  - NameID, Assertion, Attributes

SP: Management Endpoints

- `/Metadata` endpoint auto generates SAML metadata for SP
- `/Session` provides information on active session
  - Client address, IdP, AuthN method and timeout, attributes recieved
- `/GetAssertion` provides access to full assertions

SP: Performance & Availability

- RPC layer removed from calls to shibd
- Reduced number of calls to shibd
  - Shib 1.3 required two calls per request, Shib 2 uses 1 call
- Enhanced caching of data on web server side
  - Further reduces calls to shibd
- ODBC plugin allows state to be stored in a database
  - Allows state to be shared allowing SPs to be load balanced
  - Quicker recovery in the event of a restart

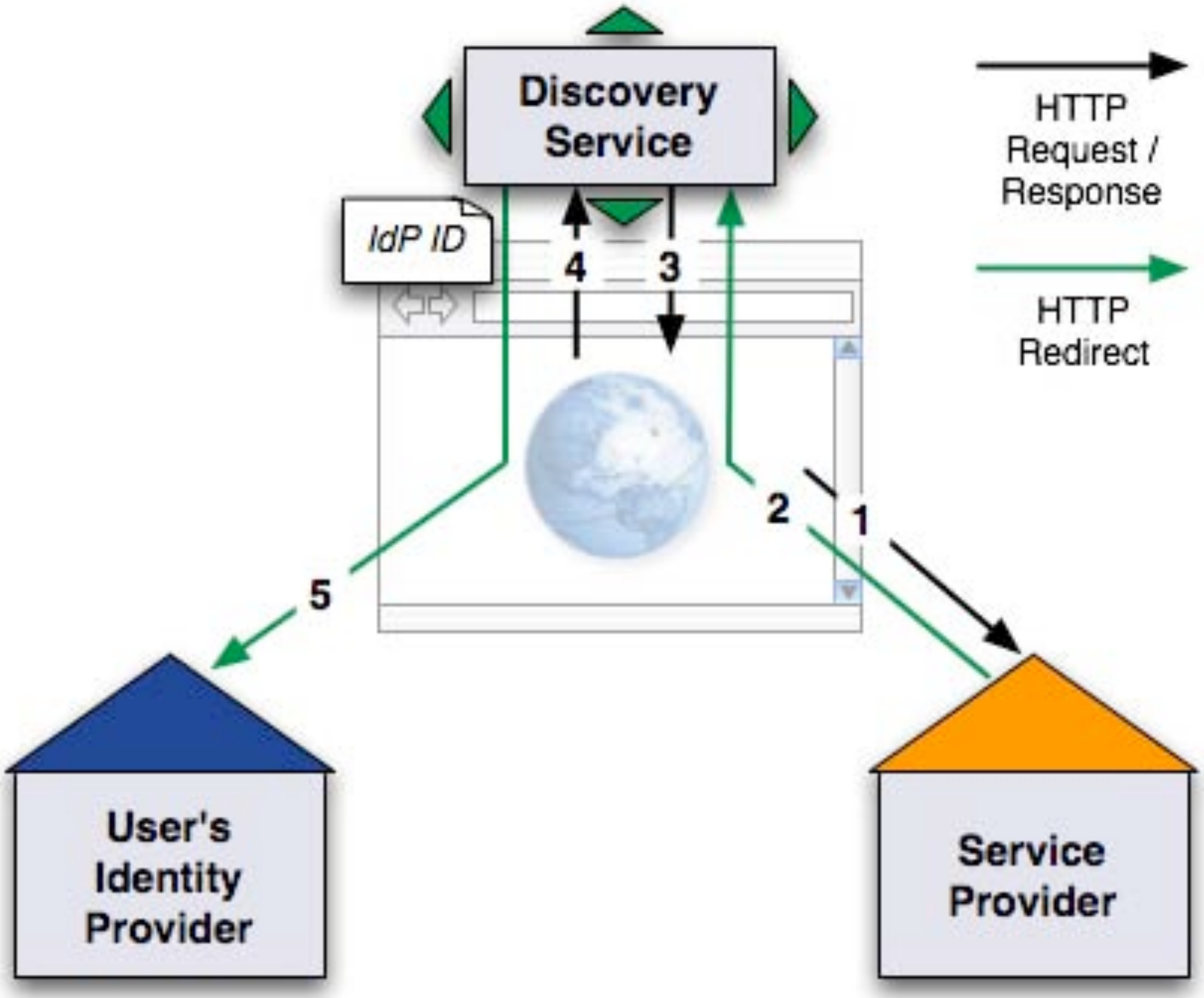
SP: Miscellaneous

- MacPorts build
- FastCGI support
- Use of environment variables within Apache
  - Protects against any spoofing, appears the same to applications

New Discovery Service

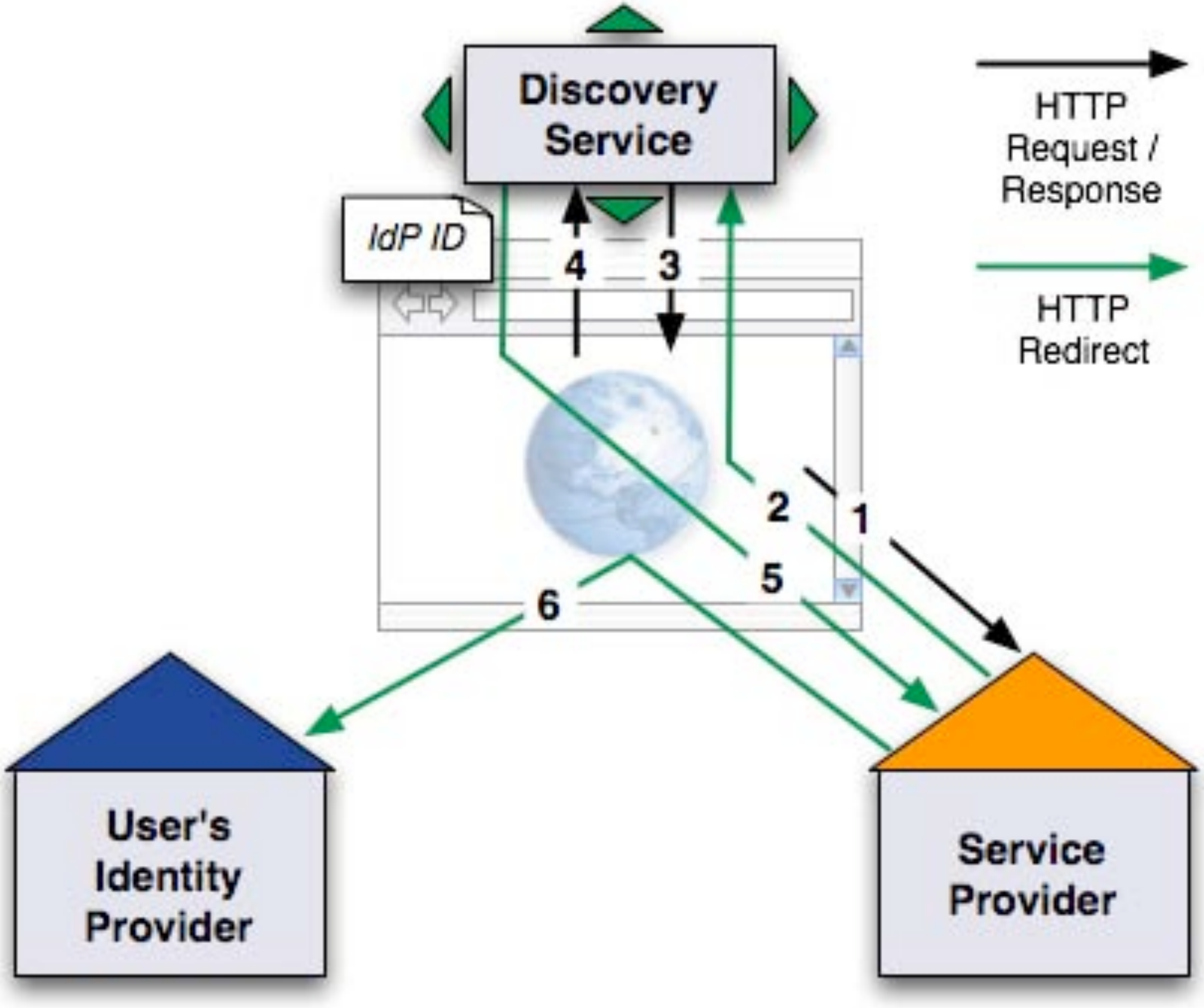
- Previously known a the WAYF service; still looks the same.
- New Features:
  - Supports new SAML discovery protocol
  - Multi-federation and multi-view support
  - Support for ordering/filtering plugins to sort/narrow list of selectable entities
- SAML Discovery Protocol is necessary when multiple protocols (SAML 1 and SAML 2) are used within a federation
  - Allows the service provider to construct/tailor a message to the identity provider it will be communicating with

New Discovery Service: WAYF Flow



© 2008 SWITCH

New Discovery Service: SAML DS Flow



Upcoming Installfests

- Service Provider Installfests
  - Zürich: June 23, 24
  - Lausanne: August 19, 20
- Identity Provider & Service Provider Installfest
  - Zürich: July 16-18