IAM, Enterprise Directories and Shibboleth (oh my!)

Gary Windham
Senior Enterprise Systems Architect
University Information Technology Services
windhamg@email.arizona.edu
What is IAM?

*Identity and Access Management (IAM)* is a framework—consisting of technical, policy, and governance components—that allows an organization to:

- identify individuals
- link identities with roles, responsibilities and affiliations
- assign privileges, access, and entitlements based on identity and associations

IAM permits data stewards and service providers to control access to information and/or services, according to an individual’s identity, roles and responsibilities.
What is IAM? (cont)

- A middleware layer, used by many services
- Comprised of four main areas:
  - **Credentials** (assignment of a unique token to an entity needing access to resources)
  - **Authentication** (act of validating proof of identity)
  - **Authorization** (act of affording access to only appropriate resources and functions)
  - **Accountability** (ensuring against illegitimate utilization of an entity’s authority...flows from the first 3 functions)
IAM Functions

- **Consolidates information** about a person and their roles (identities) across an organization or organizations.

- **Makes this information available**, in appropriate and **policy-guided ways**, to services and applications.

- **Allows for integration of services** and authority management which can grant, change, or rescind access based on status or affiliation with the organization.

- **Provides the mechanism for appropriate, auditable access** to services, critical for security architectures and to ensure compliance.
IAM Example Scenario

- “Hi! I’m Lisa.” (Identity)
- “...and here’s my NetID / password to prove it.” (Authentication)
- “I want to open the Portal to check my email.” (Authorization 😊: Allowing Lisa to use the services for which she’s authorized)
- “And I want to change my grade in last semester’s Physics course.” (Denial of Authorization ☹️: Preventing her from doing things she’s not supposed to do)

Source: Keith Hazelton - UW-Madison/Internet2 MACE
Functional View of IAM environment

Source: Internet2 Middleware Initiative
The State of UA IAM Today

- The UA NetID service was designed as an authentication mechanism, not authorization.
- Together, the UA NetID authentication service and WebAuth (UA’s Web Single Sign-On Environment) provide a solid middleware foundation for authentication
  - ...but more is needed for a full-fledged IAM environment
- Some campus applications/services rely on NetID authentication as implicit authorization.
  - The move to a “permanent” NetID will only exacerbate this problem
- Applications/services requiring more granular authorization typically use one (or more) of the following approaches:
  - Run queries (canned or custom) against UIS
  - Create and maintain local, application-specific repositories of authorization data
  - Query the LDAP “phonebook” directory (inclusion in which is not guaranteed)
Enterprise Directory Service (EDS)
What is an “Enterprise Directory”?  

- A “lightweight directory services” (LDAP) repository containing core bio/demo data for students, staff, faculty, and other University affiliates
  - Groups, roles, etc, can be easily represented as well

- Contains key institutional and person data reflected from systems of record, represented as name-value pairs that can be easily retrieved and utilized by a variety of applications

- The enterprise directory does not replace the databases supporting the institutional systems of record
  - Rather, it provides a unified view of selected subsets of these records or other information maintained by departments at the institution

- Optimized for reads—can service hundreds of requests per second
What does the EDS contain?

- The EDS contains a select subset of bio/demo attributes related to employees, students and other affiliates.

- EDS also provides a subset of attributes specified by the \textit{eduPerson} LDAP schema. \textit{EduPerson} specifies a common set of attributes related to affiliates of higher education institutions, and was developed by EDUCAUSE and Internet2.
  - \textit{eduPersonAffiliation} and \textit{eduPersonPrimaryAffiliation} are of particular interest.

- Many attributes, particularly those carrying term-specific or position/title-specific data are \textit{multi-valued}, meaning that one attribute name may be associated with multiple values.

- Some attributes group multiple, co-related pieces of information into a single, token-delimited string.
  - reduces “attribute bloat” resulting from several pieces of information which need to be represented multiple times (e.g. term-specific data, employee position information, etc).
  - examples of such attributes include \textit{studentAcademicProgram}, \textit{studentTermStatus} and \textit{employeeIncumbentRecord}. 

Where EDS fits in...
Provisioning and Consumption of EDS data

- EDS is currently provisioned with SIS and PSOS data via a daily UIS batch feed

- The future data provisioning process will incorporate daily feeds from EPM in conjunction with real-time event notifications from PeopleSoft to reflect status changes (e.g. a student record transitions from “admitted” to “enrolled”)

- Employee- and student-specific codes (e.g. PSOS employee type/status codes, SIS majors, colleges, etc) go into EDS unfiltered
  - assumes that the consuming application understands meaning of codes
  - See the EDS attribute documentation (http://sia.uiits.arizona.edu/eds) for full details

- Because the EDS attribute names are (mostly) divorced from SIS/PSOS/UIS schemas and nomenclature, populating them with PeopleSoft data should not be a huge migration effort
  - SIS- and PSOS-specific codes that change during the PeopleSoft migration will obviously change in EDS as well
  - Attributes composed of multiple, co-related data elements (e.g. `employeeIncumbentPosition`, which contains the PCN, budget department and position end-date) may incur format changes depending on how PeopleSoft models these elements
Inclusion in EDS

Data for the following populations are included in the EDS directory:

- **Students** (must meet one of the following criteria)
  - admitted for a future term
  - registered for a future term orientation session
  - enrolled in a current/future term
  - has been enrolled within the past academic year

- **Employees**
  - incumbent in a budgeted position within the last 100 days

- **Departmental Sponsored Visitors**
  - currently sponsored, non-expired
EDS Schema and Attributes

- Entries in the EDS consist of attributes belonging to the following LDAP object classes:
  - `person`
  - `inetOrgPerson`
  - `eduPerson`
  - `arizonaEduPerson`
  - `arizonaEduStudent`
  - `arizonaEduEmployee`

- All entries will contain the first 4 object classes listed above.
  - These contain base information about the person and his/her affiliation with the University.

- If a person is active in a student and/or employee role, attributes from the relevant object classes will be present as well.

- A complete list of attribute names and descriptions is available on the EDS documentation site: [http://sia.uits.arizona.edu/eds](http://sia.uits.arizona.edu/eds)
The EDS consists of a flat namespace for person data:

```
ou=People,dc=eds,dc=arizona,dc=edu
```

Entries are uniquely identified via the `uald` attribute.

- This attribute will most likely contain the PeopleSoft EMPLID value after the migration to Mosaic.

Group data (coming soon) will occupy a separate branch:

```
ou=Groups,dc=eds,dc=arizona,dc=edu
```

- The group branch may incorporate sub-branches in order to reflect organization structure and permit delegation of group membership and management functions.
Sample EDS data (Employee)

dn: uaid=116528548655,ou=people,dc=eds,dc=arizona,dc=edu
employeeType: F
employeeIsFerpaTrained: Y
eduPersonAffiliation: member
eduPersonAffiliation: employee
eduPersonAffiliation: staff
uaid: 116528548655
cn: Gary D Windham
dateOfBirth: xxxxxxxx
eduPersonPrimaryAffiliation: staff
employeePrimaryDept: 09504
employeeBldgName: COMPUTER CENTER
employeeInfoReleaseCode: Y
employeeIncumbentRecord: 862988:02601:20090630
employeePhone: 5206265981
mail: windhamg@email.arizona.edu
employeeStatus: A
employeeId: xxxxxxxx
employeeTitle: SENIOR ENTERPRISE SYSTEMS ARCHITECT, COMPUTING INFRASTRUCTURE SVCS - COMPUTING SVCS
uid: windhamg
employeeBldgNum: 00073
employeeRosterDept: 09504
employeePoBox: 210073
objectClass: arizonaEduEmployee
objectClass: top
objectClass: person
objectClass: eduPerson
objectClass: inetOrgPerson
objectClass: arizonaEduPerson
objectClass: organizationalPerson
employeeEmail: windhamg@email.arizona.edu
employeeStatusDate: 20010102
sn: Windham
givenName: Gary D
employeeRoomNum: 205
Sample EDS data (Student)

dn: uaid=109619083395,ou=people,dc=eds,dc=arizona,dc=edu
eduPersonAffiliation: member
eduPersonAffiliation: employee
eduPersonAffiliation: student
uid: 109619083395
cn: Mohammad Fazel
eduPersonPrimaryAffiliation: student
studentInfoReleaseCode: B
dateOfBirth: xxxxxxxxx
studentMinor: 091:BS:A6:SOC
mail: morox5@email.arizona.edu
studentTermStatusPast: 074:U:JR:F:RM
studentTermStatusPast: 081:U:SR:F:RM
studentTermStatusPast: 084:U:SR:F:RM
studentAcademicProgramPast: 074:BS:A6:MCB:
studentAcademicProgramPast: 081:BS:A6:MCB:
studentAcademicProgramPast: 084:BS:A6:MCB:
uid: morox5
studentAcademicProgram: 091:BS:A6:MCB:
studentid: xxxxxxxxx
studentEmail: morox5@email.arizona.edu
objectClass: arizonaEduStudent
objectClass: top
objectClass: person
objectClass: eduPerson
objectClass: inetOrgPerson
objectClass: arizonaEduPerson
objectClass: organizationalPerson
studentTermStatus: 091:U:SR:F:RM
studentAPDesc: Junior - Bachelor of Science - Molecular & Cellular Biology - 2007 Fall
studentAPDesc: Junior - Bachelor of Science - Sociology - 2007 Fall
studentAPDesc: Senior - Bachelor of Science - Molecular & Cellular Biology - 2008 Spring
studentAPDesc: Senior - Bachelor of Science - Sociology - 2008 Spring
studentAPDesc: Senior - Bachelor of Science - Molecular & Cellular Biology - 2008 Fall
studentAPDesc: Senior - Bachelor of Science - Sociology - 2008 Fall
studentAPDesc: Senior - Bachelor of Science - Molecular & Cellular Biology - 2009 Spring
studentAPDesc: Senior - Bachelor of Science - Sociology - 2009 Spring
studentMinorPast: 074:BS:A6:SOC
studentMinorPast: 081:BS:A6:SOC
studentMinorPast: 084:BS:A6:SOC
sn: Fazel
givenName: Mohammad
Sample EDS data (Student & Employee)

dn: uaid-101010892681,ou=people,dc=eds,dc=arizona,dc=edu
employeeIsFerpaTrained: N
employeeType: F
eduPersonAffiliation: member
eduPersonAffiliation: employee
eduPersonAffiliation: student
uid: Matthew D Munsey
cn: Matthew D Munsey
dateOfBirth: xxxxxxxxxx
studentInfoReleaseCode: B
eduPersonPrimaryAffiliation: staff
employeeBldgName: MUSIC
employeePrimaryDept: 03501
employeeInfoReleaseCode: Y
employeeIncumbentRecord: 858612:03501:20090630
employeePhone: 5206211272
mail: munsey@email.arizona.edu
employeeStatus: A
employeeId: xxxxxxxxxxx
employeeTitle: WEBMASTER - FINE ARTS ADMIN
studentAcademicProgram: 091:BFA:A0:STD:VIC
uid: munsey
employeeBldgNum: 000004
employeePbBox: 210004
employeeRosterDept: 03501
studentId: xxxxxxxxxxx
studentEmail: munsey@email.arizona.edu
objectClass: arizonaEduStudent
objectClass: arizonaEduEmployee
objectClass: top
objectClass: person
objectClass: eduPerson
objectClass: inetOrgPerson
objectClass: arizonaEduPerson
objectClass: organizationalPerson
studentTermStatus: 091:U:JR:P:RM
employeeEmail: munsey@email.arizona.edu
employeeStatusDate: 20080623
studentAPDesc: Junior - Bachelor of Fine Arts - Studio Art - 2009 Spring
sn: Munsey
givenName: Matthew D
employeeRoomNum: 137
EDS Access Mechanisms

**REST/DSML**

- Data returned in an XML format (specifically, DSMLv1)
- Takes a simple HTTP GET request as input
- Caller can use any standard identifier (NetID, SID/EID or UA_ID) to retrieve attribute values for the desired individual
- Each person is treated as a discrete resource with a globally unique URI endpoint. An example request, for the fictitious NetID “johndoe”, would look like this:
  
  https://eds.arizona.edu/people/johndoe

- Access to this interface requires authentication, using the application username and password provided obtained during the registration process
  - Username/password are transmitted to the REST service via standard HTTP Basic authentication
  - Credentials and data are encrypted (in transit) via HTTPS
<?xml version="1.0" encoding="UTF-8"?>
<dsml:dsml xmlns:dsml="http://www.dsml.org/DSML">
  <dsml:directory-entries>
    <dsml:entry dn="uid=11222333444,ou=People,dc=eds,dc=arizona,dc=edu">
      <dsml:attr name="studentId">
        <dsml:value>889999999</dsml:value>
      </dsml:attr>
      <dsml:attr name="studentAcademicProgram">
        <dsml:value>084:BS:AI:MICR</dsml:value>
      </dsml:attr>
      <dsml:attr name="uaid">
        <dsml:value>11222333444</dsml:value>
      </dsml:attr>
      <dsml:attr name="objectclass">
        <dsml:objectclass>
          <dsml:oc-value>arizonaEduStudent</dsml:oc-value>
        </dsml:objectclass>
        <dsml:objectclass>
          <dsml:oc-value>top</dsml:oc-value>
        </dsml:objectclass>
        <dsml:objectclass>
          <dsml:oc-value>person</dsml:oc-value>
        </dsml:objectclass>
        <dsml:objectclass>
          <dsml:oc-value>eduPerson</dsml:oc-value>
        </dsml:objectclass>
        <dsml:objectclass>
          <dsml:oc-value>inetOrgPerson</dsml:oc-value>
        </dsml:objectclass>
        <dsml:objectclass>
          <dsml:oc-value>organizationalPerson</dsml:oc-value>
        </dsml:objectclass>
      </dsml:objectclass>
      <dsml:attr name="uid">
        <dsml:value>johndoe</dsml:value>
      </dsml:attr>
      <dsml:attr name="studentTermsStatus">
      </dsml:attr>
      <dsml:attr name="eduPersonPrimaryAffiliation">
        <dsml:value>student</dsml:value>
      </dsml:attr>
      <dsml:attr name="email">
        <dsml:value>johndoe@email.arizona.edu</dsml:value>
      </dsml:attr>
      <dsml:attr name="sn">
        <dsml:value>Do</dsml:value>
      </dsml:attr>
      <dsml:attr name="givenName">
        <dsml:value>John X Doe</dsml:value>
      </dsml:attr>
      <dsml:attr name="studentInfoReleaseCode">
        <dsml:value>5</dsml:value>
      </dsml:attr>
      <dsml:attr name="o">
        <dsml:value>John X Doe</dsml:value>
      </dsml:attr>
      <dsml:attr name="eduPersonAffiliation">
        <dsml:value>member</dsml:value>
      </dsml:attr>
      <dsml:attr name="student">
        <dsml:value>
      </dsml:attr>
    </dsml:entry>
  </dsml:directory-entries>
</dsml:dsml>
EDS Access Mechanisms (cont)

**LDAP**
- The Enterprise Directory Service is based on an LDAPv3-compliant directory
- Can be accessed via the LDAPS (LDAP-over-SSL) protocol
- Access to the directory server provided via common registration process used for both REST/DSML and LDAP access
  - uses the same access credentials (username/password) required to access the REST interface
- Attributes provided via the LDAP interface are identical to those provided via the REST/DSML interface
  - However, results not in XML format
  - BER encoding format requires use of LDAP API client library
- LDAP interface offers more flexibility to application programmers at the cost of increased complexity
  - Can perform searches based on combinations of different attributes (search filters)
  - Can retrieve multiple entries in a result, rather than only a single person entry
- For connection details and programming examples, please refer to the EDS documentation: [http://sia.uiits.arizona.edu/eds](http://sia.uiits.arizona.edu/eds)
<?php

/*
 * for our example, the username/password used to access
 * the REST service is "user:password"
 */
$username = 'user';
$password = 'password';
$url = 'https://eds.arizona.edu/people/111222333444';
$cred = sprintf('Authorization: Basic %s', base64_encode($username.':'.$password));
$opts = array( 'http' => array ( 'method' => 'GET', 'header' => $cred));
$ctx = stream_context_create($opts);

// send our request and retrieve the DSML response
$dsml = file_get_contents($url, false, $ctx);

// create SimpleXMLElement from response
$xml = new SimpleXMLElement($dsml);

// set namespace context for XPath query
$xml->registerXPathNamespace('dsml', 'http://www.dsml.org/DSML');

// retrieve all values for 'eduPersonAffiliation' attribute
$query = "//dsml:entry/dsml:attr[@name='eduPersonAffiliation']/dsml:value";
$vals = $xml->xpath($query);

// examine values
foreach($vals as $node) {
    echo 'eduPersonAffiliation: ', $node, "\n";
}
?>
<?php

// setup LDAP parameters
$ldapUrl = "ldaps://eds.arizona.edu";
$bindDn = "uid=user,ou=App Users,dc=eds,dc=arizona,dc=edu";
$bindPw = "password";
$searchBase = "ou=People,dc=eds,dc=arizona,dc=edu";
$searchFilter = "(uaid=111222333444)";

// establish LDAP connection
$ldap = ldap_connect($ldapUrl);
if (!$ldap) {
    error_log("Could not connect to LDAP server");
}

// bind as app user
if (!ldap_bind($ldap, $bindDn, $bindPw)) {
    error_log(ldap_error($ldap));
}

// perform search
if (($sr = ldap_search($ldap, $searchBase, $searchFilter)) == FALSE) {
    error_log(ldap_error($ldap));
}

// retrieve entry
$entry = ldap_first_entry($ldap, $sr);
if ($e) {
    $vals = ldap_get_values($ldap, $entry, "eduPersonAffiliation");

    // examine values
    foreach($vals as $v) {
        echo 'eduPersonAffiliation: ', $v, "\n";
    }
}

ldap_close($ldap);
?>
Availability and Registration

- EDS has been generally available since mid-February
- FERPA training is a prerequisite for requesting EDS access
  - FERPA training verified against UIS table during registration process
- EDS access is granted to *applications*, which are associated with one-or-more departmental points of contact
- EDS access is requested via a self-service registration application, available at [https://eds.arizona.edu/access](https://eds.arizona.edu/access)
  - Access credentials expire, and must be renewed annually
  - Registered points of contact will receive notification well in advance of credential expiration
Shibboleth
What is Shibboleth?

- An open software system for web single sign-on
  - Developed by Internet2
- Enables web applications deployed in most typical web server environments to authenticate and authorize users via a single protocol
- Facilitates *federated identity*
- Enables fine-grained assertion of identity data to federated and external partners
  - privacy and security are key elements
Where Shibboleth fits in...
Key Concept #1: Federated Identity

- **Federated identity** supplies user information to applications offered by different organizations, enabling:
  - single sign-on
  - one identity for common access across applications and organizations
  - provisioning of authoritative data

- Identity information can include anything from the user's full identity, role information, academic or employment information to simply the fact that the user has successfully authenticated, leaving the user anonymous

There are several major advantages of federated identity:

1. It delivers authoritative user attributes directly from the institution responsible for the credentials
2. Organizations do not have to maintain credentials for inter-institutional affiliates in order to provision application access
3. User data is protected. Storage at a single, hardened location and stringent release policies minimize the chance of privacy violation
4. Users across organizations and institutions can utilize their local authentication mechanisms to access remote resources—enhances end-user experience and scales easily to new participants
Key Concept #2: Attributes

- The “currency” of the Shibboleth software is attributes.
  - named set of values about an authenticated user
  - values are typically strings, but can be more complex XML-based data.

- When a user logs into your service provider software, Shibboleth obtains a set of attributes for that user and maps them (based on rules you create) into environment variables and/or HTTP headers for your application to consume.

- Attributes not stored within Shibboleth itself
  - pulled from other sources (e.g. LDAP directory or database)

- Attribute data retrieved from sources can be enriched/transformed by both identity and service providers.

- Shibboleth is capable of using arbitrary, different attribute names for each interface, decoupling the name in any protocol from all other systems.

- Identity providers and consumers have unlimited flexibility in choosing what attributes to provide and consume.
How does it work?

- Shibboleth is an implementation of the SAML (Security Assertion Markup Language) specifications for web single sign-on and attribute exchange
  - adds additional layers of public-key trust management and configuration features specifically designed for web-based deployment
  - designed to interoperate with other open and proprietary implementations of SAML and with applications, portals, etc that offer SAML support
How does it work (cont)?

- Shibboleth is comprised of two major components:
  - *Identity Provider* (IdP) - supplies information about users to services
  - *Service Provider* (SP) - gathers information about users to protect resources (static content, application functionality, etc)

- Interaction between the IdP and SP are governed by the Shibboleth and SAML specifications

- IdP’s are typically centrally managed by the institution’s IT organization

- SP’s are installed in a service’s web application container
  - Apache and Microsoft IIS environments are supported
  - SP runtime environment consists of both an Apache module (or ISAPI filter) and a standalone daemon (or service, in Windows environments)
  - Java application servers (e.g. Tomcat, Jboss, Weblogic, etc) can be accommodated by front-ending with Apache and mod_jk, mod_wl (or similar)
IdP and SP components

- While the IdP and SP software are typically implemented as discrete, monolithic services, internally they are composed of multiple services—some are externally addressable by distinct URI endpoints, others are internal components that handle discrete phases of the SAML authentication/attribute exchange process.

- **IdP**
  - Authentication Authority
  - Attribute Authority
  - SSO
  - Artifact Resolution Service

- **SP**
  - Attribute Requester
  - Assertion Consumer Service
  - Resource Manager

- **WAYF** - “Where Are you From” service (more on this later)
The Shibboleth Protocol (intra-institutional use case)

1. User requests resource

2. You are not authenticated, redirect to IdP SSO

3. I don’t know you. Authenticate using WebAuth

4. I know you now. Send client (via form POST) to resource’s ACS

5. I don’t know your attributes. Ask the attribute authority

6. Return the attributes allowed by release policy

7. Based on attribute values, allow access to resource

Source: Kathryn Huxtable, Internet2
1. User requests resource
2. Where are you from?
   2a. You are not authenticated, redirect to federation WAYF
   2b. You are not authenticated, redirect to federation WAYF
   2c. Redirect to your home institution’s IdP
3. I don’t know you. Authenticate using org’s WebSSO
   3a. I don’t know you. Authenticate using org’s WebSSO
   3b. You are not authenticated, redirect to federation WAYF
   3c. Redirect to your home institution’s IdP
4. I know you now. Send client to resource’s ACS
   4a. I know you now. Send client to resource’s ACS
   4b. You are not authenticated, redirect to federation WAYF
5. I don’t know your attributes. Ask the attribute authority
   5. I don’t know your attributes. Ask the attribute authority
6. Return the attributes allowed by release policy
   6. Return the attributes allowed by release policy
7. Based on attribute values, allow access to resource
   7. Based on attribute values, allow access to resource

Source: Kathryn Huxtable, Internet2
Where Are You From Service (WAYF)

Select an Identity Provider

In order to fulfill the request for a web resource you have just attempted to access, information must be obtained from your identity provider. Please select the provider with which you are affiliated.

Choose from a list:

- Brown University
- Carleton College
- Case Western Reserve University
- Clemson University
- Columbia University
- Cornell University
- Dartmouth College
- Duke University
- Florida State University
- InCommon Operations
- Indiana University
- Internet2
- James Madison University
- Johns Hopkins
- Lafayette College
- Lawrence Berkeley National Laboratory
- Massachusetts Institute of Technology
- MCNC
- Medical University of South Carolina
- Miami University
- Michigan State University
- Moss Landing Marine Laboratories
- National Institutes of Health
- New York University
- NITLE (National Institute for Technology and Liberal Education)
- North Carolina State University
- Northwestern University

[Select] [Remember for session]
Wow, that’s all really complex... what does it mean to me?

- The complexity of the protocol is handled transparently by the IdP and SP software components.

- Shibboleth security is applied *declaratively* to resources within the web application container (e.g. via Apache `<Location>` directives or IIS virtual URLs).

- Complex access control rules involving multiple attributes can be declaratively configured via XML.

- The Shibboleth SP facilitates direct access to the attributes in the SAML assertion via the application environment.
Attributes released to a particular SP depend on an attribute release policy (ARP) maintained by the IdP
- ARPs usually are written to release the same set of attributes to all members of a particular federation (identified via federation metadata)

- UA maintains an “internal” federation consisting of campus service providers who wish to utilize Shibboleth for intra-campus authentication and authorization
  - The ARP for this federation releases all the attributes contained in the EDS
  - Membership in this federation is governed by the same policies as EDS access

- UA is also a member of the InCommon federation—an identity federation made up of North American higher education institutions and partners
  - The ARP for this federation releases very basic information by default: `eduPersonAffiliation`, `eduPersonPrimaryAffiliation` and `eduPersonTargetedID`
Really...show me the attributes

- Attributes are provisioned to consuming applications, by the SP software, via environment variables (or HTTP headers in some cases).

- In most cases, retrieving an attribute is as simple as referencing an environment variable or an object property. Examples for retrieving the `employeePrimaryDept` attribute in a few different environments follow:
  - **Apache/PHP**
    
    ```
    $_SERVER('Shib-employeePrimaryDept')
    ```
  - **Java**
    
    ```
    HttpServletRequest.getHeader("Shib-employeePrimaryDept")
    ```
  - **ASP**
    
    ```
    Request.ServerVariables("HTTP_SHIB_EMPLOYEEPRIMARYDEPT")
    ```
  - **Cold Fusion**
    
    ```
    CGI.SHIB_EMPLOYEEPRIMARYDEPT
    ```
Really...show me the attributes (cont)

- The EDS attribute list represents, for the most part, the set of attributes available via Shibboleth.

- UITS provides an UA-specific SP `attribute-map.xml` file, which provisions these attributes using the same names as described in the EDS attribute list:
  - but prefixed with the string "Shib-" (in order to avoid any potential HTTP environment/request header namespace collisions)
  - some exceptions, due to attributes that don’t map one-to-one with EDS attributes; see the documentation site for details.

- Federated use cases require coordination between the SP and the IdP organizations to ascertain what attributes will be released by the IdP and how they should be mapped at the SP.
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shib-Application-ID</td>
<td>default</td>
</tr>
<tr>
<td>Shib-Session-ID</td>
<td>96d5577a3d6c7dbb0c08655d13fd5000</td>
</tr>
<tr>
<td>Shib-Identity-Provider</td>
<td>urn:oasis:names:tc:SAML:2.0:ac:classes:PreviousSession</td>
</tr>
<tr>
<td>Shib-Authentication-Instant</td>
<td>2009-01-30T07:07:43.938Z</td>
</tr>
<tr>
<td>Shib-affiliation</td>
<td><a href="mailto:member@arizona.edu">member@arizona.edu</a>;<a href="mailto:employee@arizona.edu">employee@arizona.edu</a>;<a href="mailto:staff@arizona.edu">staff@arizona.edu</a></td>
</tr>
<tr>
<td>Shib-cn</td>
<td>Gary D Windham</td>
</tr>
<tr>
<td>Shib-dateOfBirth</td>
<td>20010102</td>
</tr>
<tr>
<td>Shib-employeeBldgName</td>
<td>COMPUTER CENTER</td>
</tr>
<tr>
<td>Shib-employeeBldgNum</td>
<td>00073</td>
</tr>
<tr>
<td>Shib-employeeEmail</td>
<td><a href="mailto:windhamg@email.arizona.edu">windhamg@email.arizona.edu</a></td>
</tr>
<tr>
<td>Shib-employeeId</td>
<td></td>
</tr>
<tr>
<td>Shib-employeeIncumbentRecord</td>
<td>002</td>
</tr>
<tr>
<td>Shib-employeeIsFerpaTrained</td>
<td>Y</td>
</tr>
<tr>
<td>Shib-employeePhone</td>
<td>5206265981</td>
</tr>
<tr>
<td>Shib-employeePoBox</td>
<td>210073</td>
</tr>
<tr>
<td>Shib-employeePrimaryDept</td>
<td>09504</td>
</tr>
<tr>
<td>Shib-employeeRoomNum</td>
<td>205</td>
</tr>
<tr>
<td>Shib-employeeRosterDept</td>
<td>09504</td>
</tr>
<tr>
<td>Shib-employeeStatus</td>
<td>A</td>
</tr>
<tr>
<td>Shib-employeeStatusDate</td>
<td>20010102</td>
</tr>
<tr>
<td>Shib-employeeTitle</td>
<td>SENIOR ENTERPRISE SYSTEMS ARCHITECT, COMPUTING INFRASTRUCTURE SVCS - COMPUTING SVCS</td>
</tr>
<tr>
<td>Shib-employeeType</td>
<td>F</td>
</tr>
<tr>
<td>Shib-epnn</td>
<td><a href="mailto:windhamg@email.arizona.edu">windhamg@email.arizona.edu</a></td>
</tr>
<tr>
<td>Shib-givenName</td>
<td>Gary D</td>
</tr>
<tr>
<td>Shib-persistent-id</td>
<td>urn:oasis:names:tc:SAML:2.0:ac:classes:PreviousSession</td>
</tr>
<tr>
<td>Shib-primary-affiliation</td>
<td>staff</td>
</tr>
<tr>
<td>Shib-sn</td>
<td>Windham</td>
</tr>
<tr>
<td>Shib-uaid</td>
<td>116528548655</td>
</tr>
<tr>
<td>Shib-uid</td>
<td>windhamg</td>
</tr>
<tr>
<td>Shib-uncscoped-affiliation</td>
<td>member;employee;staff</td>
</tr>
</tbody>
</table>
Lazy Sessions

- Application developers who wish to delay the Shibboleth SSO process can utilize an advanced feature called “lazy sessions”

- The Shibboleth SP registers a series of virtual URIs with the underlying web application container

- Application developers can simply issue an HTTP Redirect to the Shibboleth “session initiator” URI (typically, /Shibboleth.sso/Login), with query-string parameters indicating the return point, in order to launch the Shibboleth SSO process
InCommon Federation

• UA is an InCommon federation member
• InCommon is a higher education federation of identity and service providers
• Provides common policies, practices and framework for sharing identity information across institutions
• Enables collaboration and enhances trust
• The de-facto standard for identity federation in higher ed
### Current InCommon Participants

A community of more than **2.2 million end users**.

(October 2008. Source: Higher Education Students, Faculty, and Staff, Integrated Postsecondary Education Data System.)

<table>
<thead>
<tr>
<th>Higher Education Participants (83)</th>
<th>Government and Nonprofit Laboratories, Research Centers, and Agencies (5)</th>
<th>Sponsored Partners (33)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brown University</td>
<td>Lawrence Berkeley National Laboratory</td>
<td>Apple - iTunes U</td>
</tr>
<tr>
<td>California State University, Office of the Chancellor</td>
<td>Moss Landing Marine Laboratories</td>
<td>Burton Group</td>
</tr>
<tr>
<td>Carleton College</td>
<td>National Institutes of Health</td>
<td>Cengage Learning, Inc.</td>
</tr>
<tr>
<td>Case Western Reserve University</td>
<td>TeraGrid</td>
<td>e2Campus by Omnilert, LLC</td>
</tr>
<tr>
<td>Clemson University</td>
<td></td>
<td>EBSCO Publishing</td>
</tr>
<tr>
<td>College of William and Mary</td>
<td></td>
<td>Elsevier</td>
</tr>
<tr>
<td>Colorado State University</td>
<td></td>
<td>Houston Academy of Medicine - Texas Medical Center Library</td>
</tr>
<tr>
<td>Columbia University</td>
<td></td>
<td>Identit-e</td>
</tr>
<tr>
<td>Cornell University</td>
<td></td>
<td>Internet2</td>
</tr>
<tr>
<td>Dartmouth</td>
<td></td>
<td>JSTOR</td>
</tr>
<tr>
<td>Duke University</td>
<td></td>
<td>Kuali Foundation</td>
</tr>
<tr>
<td>Emory University</td>
<td></td>
<td>MCNC</td>
</tr>
<tr>
<td>Florida State University</td>
<td></td>
<td>Microsoft</td>
</tr>
<tr>
<td>George Mason University</td>
<td></td>
<td>National Institute for Technology and Liberal Education (NITLE)</td>
</tr>
<tr>
<td>Georgetown University</td>
<td></td>
<td>National Student Clearinghouse</td>
</tr>
<tr>
<td>Hampden-Sydney College</td>
<td></td>
<td>NG Web Solutions</td>
</tr>
<tr>
<td>Indiana University</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Availability and Registration

- Shibboleth has been generally available since mid-February
- FERPA training is a prerequisite for requesting Shibboleth access
  - FERPA training verified against UIS table during registration process
- Shibboleth access is granted to applications, which are associated with one-or-more departmental points of contact
- Shibboleth access is requested via a self-service registration application, available at https://eds.arizona.edu/access
  - Upon registration, the point-of-contact will receive instructions to complete the set-up process (including certificate generation, inclusion in UA metadata, etc)
  - Federated SP deployments require additional set-up...please contact UITS-IIA@listserv.arizona.edu if you have a need to establish federated service with InCommon members or external entities
Resources

- uits-iia@listserv.arizona.edu
- http://sia.uits.arizona.edu/eds
- http://sia.uits.arizona.edu/shibboleth
- This presentation available online at: http://sia.uits.arizona.edu/iam
Q & A