TAS³ Workshop
Architecture Using ZXID and PERMIS

Sampo Kellomäki (sampo@symlabs.com), Symlabs
25-26-27.8.2009 Lisboa
Notes from Buda

IdP integration items * AMQP or SAWS * OCT support, for generating tokens

PEP integration items * Method profiles (stored on PEP machine as configuration, profiles written by app developer) to describe attributes to feed

ID Mapper integration * Trust and Privacy Negotiator mechanics
TalkTo

- Jutta:
  - how to integrate the workflow (to mod_auth_saml?)
  - discovery
  - PIP
  - Stack
- Brecht: profiles, interop
**Overall Outline for 3 Days**

https://portal.tas3.eu/trac/wiki/Meeting/2009-08-25

**Venue:** R. Padre Damian 6B, Lisboa (behind Centro Cultural Belém)  
**Sampo:** +351-918.731.007

**Tue** Setup, infra, and demo  
**Wed** ZXID  
**Thu** PERMIS

- Travel arrangements? Usability of Thu and Fri?  
- 9 am to 19 pm  
- Coffee and tea provided  
- Lunch at the near by restaurant  
- Dinner plan?
Attendance

- Jeroen
- Marc S.
- Jens
- Tom
- Brian
- Marc Van Collie
- David
- Stijn
- George
- Sampo
Setup, Infra, and Demo Outline (Tue)

1. WiFi connectivity, firewall (full out, nothing in), etc.
   - WPA: ssid="BNGWIZI_Adsl" psk="72JTHPK5ACNA9"
   - Use DHCP (netmask 24 bits, gateway: 192.168.1.1)
     - After DHCP gives you address, use that as fixed address
   - DNS: OpenDNS 208.67.222.222 208.67.220.220 for external
     - Use /etc/hosts for peers after fixed IPs

2. Concrete architecture we are trying to setup
   - Feedback and planning on objectives of each participant

3. Demo of what is there already: SSO and Az

4. CA and setup certs for everybody, Connectivity Test

5. Compile / Package Install for ZXID and PERMIS

6. Output documents from this event?
ZXID Outline (Wed) (1/2)

1 Create your own SP
   1. Dummy using ZXID standalone code
   2. Hookup to CoT, metadata
   3. See it work
   4. Integrate to your own code
   5. See it work

2 Triggering Az from SSO

3 Using SSO attributes

4 Creating your own WSC
   1. Demo of actual web service call, with traces
   2. Integrating ZXID code to call existing service
ZXID Outline (Wed) (2/2)

5 Providing your own WSP
   1. Integrating ZXID code
   2. Service Registration Step
   3. Association Step
   4. Making web service call: your WSC to your WSP

6 Interop
   1. Discovering other people’s WSPs
   2. Your WSC calling other people’s WSP
   3. Your WSP being called by other people’s WSC

7 mod_auth_saml tutorial
PERMIS Outline (Thu)
Supplied separately by Kent.
Homework Prior to Event (1/2)

https://portal.tas3.eu/trac/wiki/Meeting/2009-08-25/ZXID

The workshop is intended to be on developer or poweruser deployer level. Therefore

- You MUST have C development environment (gcc, ld, make, sed, perl, tar, gunzip) installed. Be sure to install headers as well. You will also need OpenSSL and libCurl development packages. On Windows, install Cygwin with the above (and below) components.
- If you plan to use perl, php, Java, or other scripting solution, be sure to have full development environment for whatever you do. If you do Java, have your Tomcat figured out and working.
- Have a web server (Apache 2.2 recommended) installed and functioning on your laptop.
Homework Prior to Event (2/2)

• Practise creating X509v3 certificates with your tools.
• Have wireshark or similar installed and know how to use it. Also browser plugins like "Tamper Data" for analyzing http traffic may come handy.
• Compile zxid downloaded from zxid.org
• Compile PERMIS
CA and Certs

- Jeroen’s CA
  - Jeroen to supply more material
- Configuring trust on new root CA at Browser and OpenSSL level
- Self signed certs, openssl command line tutorial
- PEM format (and other formats)
- Role of Metadata, Circle-of-Trust, and Auto-CoT Metadata Exchange based on WKL
Example PEM Cert

—–BEGIN CERTIFICATE—–
MIIGWTCCBcKgAwIBAgIDAjEBMA0GCSqGSib3DQEBBBQUAMIIIBEjELMAkGA1UEBhMC
RVMxEjAQBgNVBAgTCUJhcmNiGbG9uYTESMBAGA1UEBxMJQmFyY2Vsb25hM2HetQQK
VQQKEyBJU0MwEQYDVQQKEwpTeW1sYWJzIFNBMRQwEgYDVQQLEwtJVCBTZXJzaWdu
MRAwHgYJKoZIhvcNAQkBFhFmZWxpeEBzeW1sYWJzLmNvbTCBnzANBgkqhkiG9w0BAQ
EFAAOBjQAwgYkCgYEA06uxx5ZjAl06CZcSMVtjoaS2sCbrBq/whwWnuVgbD6gAM9EO9q
DDEs9eB5n4lHGY8S94iFTWuZy9jdxL5wNgr2Zk8NxytyaznQgAddKLCSqPZh7Dd+U3Z5hoGtLe

25.6.2009
Sampo Kellomäki: TAS3 ZXID PERMIS 01
25.6.2009 Sampo Kellomäki: TAS3 ZXID PERMIS 01

---END CERTIFICATE---
Example PEM Cert And Private Key As Used by ZXID

---BEGIN CERTIFICATE---
MIIGWTCCBcKgAwIBAgIDA-JEBMA0GCSqGSIb3DQEBBQUAMIGIIBejELMAkGA1UEBhMC
RVMxEDjAQBgNVBAgTCUJhcmNlbG9uYTESMBAGA1UEBxMJQmFyY2Vsb25hMk
VQQKEyBJUFMgQ2VydGlmaWNhdGlvbiBBdXRob3JpdHkgcy5sLjEuMCwGA1UEChQl
(snip) 8lb3DQEBBQUAA4GBACan4TGRFHayR38xPkJMabzwz9VmCbm0u
jakSenPwpvomvNfp4G0WJdavd7KnZBbMbnKx1qTMgge/ftBnuqcncn6wa
ahftQ+r2gFYiVX4HEa6NU5AgpiQjme0Vh3Hzs228lVIlgsFqv6YbdlyTYIU
---END CERTIFICATE---

---BEGIN RSA PRIVATE KEY---
MIICXQIBAAKgQDTq7HHImMCXToJlxloOMSz14HmfUcZjxL3iIVNa5nL2N3EvnA2CvZmTw3HK3JrOdCAB10os.8nEpzyUJWXpCs9K+kuuJAkJAm0b523XnsJmsipA+ZDdyqrUjKDo6WH3/GeJEXxqlwfcj2IZLp/iRvG7ICjN/rdWoNImF3HVBRS ---END RSA PRIVATE KEY---

25.6.2009 Sampo Kellomäki: TAS3 ZXID PERMIS 01
Example Metadata

<md:EntityDescriptor xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
  entityID="https://idp1.zxidp.org:8443/zxididp?o=B">
  <md:IDPSSODescriptor
    WantAuthnRequestsSigned="1" errorURL="https://idp1.zxidp.org:8443/zxididp?o=E"
    protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">
    <md:KeyDescriptor use="encryption">
      <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig">
        <ds:X509Data>
          <ds:X509Certificate>
            MIIGWTCCBcKgAwIBAgIDA-
            JEBMA0GCSqGSIb3DQEBBQUAMIIIBEjELMAkGA1UEBhMC RVMxEx-
            jAQBgNVBAgTCUJhcmNlbG9uYTESMBAGA1UEBxMJQmFyY2Vsb25hM
            VQQKEyBjcmNlbG9uYTESMBAGA1UEBxMJQmFyY2Vsb25hM
          </ds:X509Certificate>
          (snip) jkSenPwpvomvNfp4G0WJdavd7KnZBbMbnKx1qTMgge/ftBnuq
          aHftQ+r2gFYiVX4HEa6NU5AgpiQjme0Vh3Hzs228lVlIgsFqv6YbdIyTYIU
        </ds:X509Data>
      </md:KeyDescriptor>
      <md:KeyDescriptor use="signing">
        <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig">
          <ds:X509Data>
            <ds:X509Certificate>
              MIIGWTCCBcKgAwIBAgIDA-
            </ds:X509Certificate>
            (snip) jkSenPwpvomvNfp4G0WJdavd7KnZBbMbnKx1qTMgge/ftBnuq
            aHftQ+r2gFYiVX4HEa6NU5AgpiQjme0Vh3Hzs228lVlIgsFqv6YbdIyTYIU
          </ds:X509Data>
        </md:KeyDescriptor>
  </md:KeyDescriptor>
</md:IDPSSODescriptor>
</md:EntityDescriptor>
25.6.2009 Sampo Kellomäki: TAS3 ZXID PERMIS 01
<md:KeyDescriptor use="encryption"> <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig"> <ds:X509Data> <ds:X509Certificate> MIIGWTCCBcKAwIBAgIDAJEBMA0GCSqGSIb3DQEBBQUAMIGA1UEBxMJQmFyY2Vsb25hMC RVMxEjAQBgNVBAsGSIb9uYTESMBAGA1UEBxMJQmFyY2Vsb25hM... (snip) jkSenPwpvomvNfp4G0WJdavd7KnZBbMbnKx1qTMgge/ftBnuQaHftQ+r2gFYiVX4HEa6NU5AgpiQjme0Vh3Hzs228lVIlgsFqv6YbdLyTYIU <md:KeyDescriptor use="signing"> <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig" xmlns:ds="http://www.w3.org/2000/09/xmldsig"

<ds:X509Data> <ds:X509Certificate> MIIGWTCCBcKAwIBAgIDAjEBMA0GCSqGSIb3DQEJBQUAMIIjEjELMAkGA1UEBhMCVVMxETQjABIBgNVBAgTCUJhcmNlbiBGa1UEUzMBV0NKS0BTRQYDVQQKEwFtYXVsYXRpb25hMRgwFg0lMCQGA1UdIwQjMCwwNDApMQ0tMTQxOTI2OTYxNDQ4MTQxOTI2OTYxNDQ4MA0GA1UdIwQjMCwwNDApMQ0tMTQxOTI2OTYxNDQ4MTQxOTI2OTYxNDQ4MADTAQGCZMUwNzA1Nzg1MDkxMTQxNjgyODA0MDU0MzE1NzYxMDMwMSAwCgYIKwYBBQUHAwEwggD6AMEQQKFDQSC8yMDEzMTc5Mzc5MzI5NzU5MjYwOTgwMDQwNTQzMDkyMQ0wCQYDVQQIEw9HUmNlbi5ldGggUmVvdXVsdCBTUlN属于ZXID内部的内部网络,该网络保证了数据的机密性和完整性。通过这种方式,ZXID能够为用户提供一个安全、可靠、可扩展的环境,支持业务流程的自动化和信息化。随着业务的不断扩大,ZXID将持续完善其内部网络和安全策略,为用户提供更为优质的服务。

25.6.2009 Sampo Kellomäki: TAS3 ZXID PERMIS 01
Key Concepts

- **SP - IdP**
- **SP/PEP - PDP**
- **SP/WSC - WSP**
- **WSP registering itself**
- **SP/WSP creating an association for the user**
- **SP/WSC discovering WSP**

**Metadata**
- End Point URLs
- Signing certificate
- XML-Enc certificate
- **TLS/SSL certificate**

**Metadata import**
- WKL method

**Direct Trust by Listing Metadata**
Demo

IdP https://idpdemo.tas3.eu:8443/zxididp?o=B
IdP Selection

ZXID SP Federated SSO (user NOT logged in, no session)

Login Using New IdP

A new IdP is one whose metadata we do not have yet. We need to know the IdP URL (aka Entity ID) in order to fetch the metadata using the well known location method. You will need to ask the administrator of the IdP to tell you what the Entity ID is.

IdP URL: http://r90s.germany.sun.com:8080/amserver-fep-sym-saml-idp
Entity ID of this SP (click on the link to fetch the SP metadata): https://sp1.zxidsp.org:8443/zxidhlo?o=B

Login Using Known IdP

Login to https://lolo:8681/idp.xml (A2)  Login to https://lolo:8681/idp.xml (P2)
Login to https://idp.symdemo.com:8880/idp.xml (A2)  Login to https://idp.symdemo.com:8880/idp.xml (P2)

Technical options

- Create federation, NID Format: Persistent

zxid.org, 0.18 1178728139 libz nid (zxid.org)
Login at IdP

Welcome to Id Provider "IdP3 A" Home Login

You may login using various methods (pick your poison)

(be sure browser accepts cookies from the same domain)

1. Cookie login

Username: sue
Password: ***

If any web site (SP) asks...

The IdP URL (Provider ID/Entity ID) of this IdP is https://a-idp.liberty-idp.org:8881/idp.xml

You can cut and paste the above URL to any web site that allows Single Sign-On using IdP URL or "Any IdP" or "Other IdP". This mechanism allows the web site (SP) to dynamically join the Circle of Trust of this IdP. This is called Auto-CoT.
SSO Successful: Protected Page
SAML Hello World in PHP

- 38 lines of PHP code of which only 22 do something (rest are comments or HTML)
- Complete
  - All profiles are handled
  - Single Logout handled
  - Well Known Location (WKL) metadata exchange handled
- Hides SAML protocol details
- This Hello World can be cut-and-pasted into any PHP application
Initialization once

```php
01 <?
02 dl("php_zxid.so");  // Pull in module (.so file)
03 // CONFIG: You must have created /var/zxid directory hierarchy
04 // CONFIG: You must edit the URL to match your domain
05 $conf = "PATH=/var/zxid/
       &URL=https://sp1.zxidsp.org:8443/zxidhlo.php"
06 $cf = zxid_new_conf_to_cf($conf);
07 ?>
```

- **PATH** configuration means multiple instances of ZXID can coexist (e.g. virtual hosting of web sites)
- **URL** configuration determines provider ID, can also be configured via `/var/zxid/zxid.conf`
Per protected page or until session is bootstrapped

08 <?
09 $qs = $_SERVER['REQUEST_METHOD'] == 'GET'
10 ? $_SERVER['QUERY_STRING']
11 : file_get_contents('php://input');
12 $res = zxid_simple_cf($cf, -1, $qs, &ses, 0x1814);
13
14 switch (substr($res, 0, 1)) {
15 case 'L': header($res); exit;
16 case '<': header('Content-type: text/xml'); echo $res;
     • Read input and call zxid_simple() to handle SAML protocol details
     • Act on outcome of zxid_simple() as indicated by the first letter
       - L: protocol requires redirect, perform it
       - <: Send out XML data (such as Metadata or SOAP response)
The IdP Selection Page

17 case 'n': exit;  # Already handled, do nothing further
18 case 'e':
19 ?>
20 <title>Please Login Using IdP</title>
21 <h1>Please Login Using IdP</h1>
22 <?=zxid_idp_select_cf($cf, null, 0x1800)?>
23 <?
24 exit;

• e: indicates that IdP Selection page needs to be rendered
• zxid_idp_select() generates the ZXID standard form
• Alternatively you could supply your own HTML for the form as long as you respect the form field naming convention
Login Successful Case

25 case 'd': break;  # Logged in case -- continue after
26 default:  die("Unknown zxid_simple() res($res)" );
27 }

28

29 # Parse the LDIF in $res into a hash of attributes $attr
30
31 foreach (split("\n", $res) as $line) { 
32    $a = split(": ", $line);
33    $attr[$a[0]] = $a[1];
34 }
35 ?>

• d: login successful, return data is LDIF entry with attributes of SSO
Protected Content with Single Logout and Defederate Buttons

36 <title>Protected content, logged in</title>
37 <h1>Protected content, logged in as <?=$attr['cn']?>?</h1>
38 <?=zxid_fed_mgmt_cf($cf, null, -1, $attr['sesid'], 0.61)?>

- `zxid_fed_mgmt()` generates the Single Log-Out buttons
- This is the place to bootstrap your application’s own session
Login Successful: Returned LDIF

dn: idpnid=Pa45XAs2332SDS2asFs,affid=https://idp.demo.com/idp.xml
objectclass: zxidsession
affid: https://idp.demo.com/idp.xml
idpnid: Pa45XAs2332SDS2asFs
authnctxlevel: password
sesid: S12aF3Xi4A
cn: Joe Doe

• The LDIF entry is used as convenient format for passing attribute-value pairs from `zxid_simple()` to application
• Some "attributes" are synthesized, others come actually from assertion
Thank You
Sampo Kellomäki (sampo@symlabs.com)
+351-918.731.007

<table>
<thead>
<tr>
<th>Product</th>
<th>License</th>
<th>Platform</th>
<th>SAML SP</th>
<th>SAML IdP</th>
<th>WSC</th>
<th>WSP</th>
<th>Disco</th>
<th>People</th>
<th>Interact</th>
<th>Account</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZXID.org</td>
<td>Apache2</td>
<td>C + SWIG</td>
<td>Full</td>
<td>y</td>
<td>y</td>
<td></td>
<td></td>
<td>TBA</td>
<td>TBA</td>
<td>TBA</td>
<td></td>
</tr>
<tr>
<td>mod_auth_saml</td>
<td>Apache2</td>
<td>C + SWIG</td>
<td>Full</td>
<td>-</td>
<td>y</td>
<td>y</td>
<td>WSC</td>
<td>TBA</td>
<td>TBA</td>
<td>TBA</td>
<td></td>
</tr>
<tr>
<td>Lasso</td>
<td>GPL2+</td>
<td>C + SWIG</td>
<td>Cert</td>
<td>-</td>
<td>y?</td>
<td>y?</td>
<td>WSC</td>
<td>TBA</td>
<td>TBA</td>
<td>TBA</td>
<td></td>
</tr>
<tr>
<td>Authentic</td>
<td>GPL2+</td>
<td>C + SWIG</td>
<td>Certified</td>
<td>-</td>
<td>-</td>
<td>WSP</td>
<td>-</td>
<td>TBA</td>
<td>TBA</td>
<td>TBA</td>
<td></td>
</tr>
<tr>
<td>OpenSSO</td>
<td>Java?</td>
<td>pure PHP</td>
<td>Partial</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OpenSSO</td>
<td>Java?</td>
<td>Java</td>
<td>Cert</td>
<td>Cert</td>
<td>1.1</td>
<td>1.1</td>
<td>1.1</td>
<td>-</td>
<td>1.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OpenSAML</td>
<td>?</td>
<td>Java?</td>
<td>Partial</td>
<td>Partial</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OpenLiberty</td>
<td>Apache2</td>
<td>Java</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ConorCli</td>
<td>BSD?</td>
<td>C++</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ConorSvc</td>
<td>BSD?</td>
<td>Java</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**TBA** To be announced, on the road map, but not here yet  

**1.1** Only supports older ID-WSF 1.1 version of the services  

**C + SWIG** C library with language bindings generated using SWIG
tool. SWIG supports among others C, C++, Perl, PHP, Python, Ruby, and Java language bindings. Generally open source products can be compiled for all popular operating systems such as Unix and Windows.

Additional info available on openliberty.org