Liberty Conformance Testing Tips

Version 1.0-09

This document attempts to provide some guidance to Liberty Conformance Test participants, and should help in preparing for and successfully passing the tests.

Revisions:

Version	Date	Changes
1.0-06	5-May-04	Clarified LECP Tip #1 to convey correct concept.
1.0-07	10-Aug-04	Updated contact info, expanded certificate description
1.0-08	1-May-05	Added list of cipher suites that should be used
1.0-09	25-May-05	Fixed up some formatting and converted to OpenOffice

1 Testing Setup

1.1 Location and Contacts

Info about the location can be obtained at the registration page for the event. Scroll to the bottom.

When responding to event email sent to the event list, remember that not everyone on the list is under NDA and that by responding you may be exposing the fact of your participation to anyone who might be CC'd.

The best way to contact the conformance team is to send inquiries to cert-admin@projectliberty.org.

The main Liberty contact is:

Roger Sullivan
Chair of Conformance Expert Group
roger.sullivan@oracle.com

The ISTO contacts are

Eric Tiffany
+1-413-458-3743
+1-413-627-1778 (GSM mobile)
eric@projectliberty.org

Debbie Mac
+1 (646) 221-1100
debbie@ieee-isto.org

1.2 Equipment

1.2.1 Shipping

If you have equipment that you need to ship to the event, see the shipping information on the registration page or elsewhere.

1.2.2 Electricity

Remember that your equipment may use different plugs, voltage and frequency than what will be available at the testing event. This means that whatever you bring needs to work with any voltage, and has a plug adapter. We won't be able to provide everyone with plug adapters, so bring your own. We won't be able to provide **anyone** with a transformer such as you might need for a desktop computer or workstation. It's possible that the event venue will have transformers, but I wouldn't plug my workstation into one.

1.2.3 Security

If you are concerned about losing your laptop, use a security cable. Lock your screen when you are not sitting in front of it.

1.3 DNS and addresses

The event will use special domains for the hosts: `liberty-iop.org` for the actual servers, and `cot.projectliberty.org` for the "introduction cookie common domain".

We will be using static addresses which will be assigned in advance. You machines should be configured with the correct names and addresses.

In order to preserve anonymity outside the event, each team will be given a letter (e.g. "A") and this will be used to construct the host names: A-SPA, A-SPB, A-IDP, A-LECP.

1.4 Certs and Keys

Certificates and Keys are provided along with the CA certificate. The certs are provided in both JKS and PKCS12 format; corresponding keys in the two formats use the same private key. If you need another format, you should convert using openssl or other tools.

The keys and certs all have a password of "changeit" and the alias in the JKS file is "tomcat". These are conveniently the defaults used by some web servers. Note that the CA certificate used to sign these keys and certs is a "bogus" self-signed cert, which you must add to your keychain or keystore and then "trust".

The certs/keys come in several different flavors:

- "Regular" SSL certificates, named (e.g.) a-wsp-ssl.jks and a-wsp-ssl.p12. These should be installed as the SSL/TLS certificate for the appropriate servers.
- "Signing" certs for signing XML, named (e.g.) a-wsp-sign.jks and a-wsp-sign.p12 (these certs have the x509 extension `keyUsage` attribute `digitalSignature`)
- "Encryption" certs for EncryptedResourceID encryption, named (e.g.) a-wsp-enc.jks and a-wsp-enc.p12 (these certs have the x509 extension `keyUsage` attribute `keyEncipherment`)
- "Introduction" certs for use in the "common-domain" IDP introduction scenarios, named (e.g.) a-idp-intro.jks and a-idp-intro.p12.

The CA, keys and certs will be sent prior to event.

1.5 Metadata

Send your metadata information (URLs where your metadata can be retrieved) to cert-admin@projectliberty.org. These URLs will be available on a website at the event. Note that the URLs need to be expressed in terms of your dns names for the event (i.e., https://A-SP.liberty-iop.org:8443/metadata or some such).

1.6 Testing in Advance

[Note: The option for testing in advance may not be available at all events]

Initial test pairings will be determined (to some degree) by the order in which the signed NDA agreements are received, to the extent this will result in valid pairings. Once both parties have submitted their NDAs, they will be introduced privately so that they may perform preliminary testing if they wish to.

The keys and certs should be usable if the testing machines are suitably configured to use /etc/hosts (or equivalent) for name resolution and the relevant testing host names are assigned the correct IP addresses.

2 Network

The network will consist of approximately the following:

- Direct internet access without firewall or NAT for testing.
- Dumb 10 Mbit/sec hubs connected together (about 48 ports total). We will use dumb hubs so that all traffic is visible to the sniffer machine.
- Wireless 802.11b network for non-event traffic (NAT to internet). The wireless router may have some Ethernet ports for attachment of wired connections that need it.

The network setup details will be provided as soon as they are finalized by the event facility

3 Recording

Testing interactions will be recorded using an SSL network sniffer. This will provide a log of the transactions over the wire, and will be retained by Liberty. It may also be used to resolve any disputes during or after testing. **Note that in order for the ssldump logging program to decipher your streams, you must configure your SSL cipher to be one of the following:**

```
TLS_RSA_WITH_NULL_MD5
TLS_RSA_WITH_NULL_SHA
TLS_RSA_EXPORT_WITH_RC4_40_MD5
TLS_RSA_WITH_RC4_128_MD5
TLS_RSA_WITH_RC4_128_SHA
TLS_RSA_EXPORT_WITH_RC2_CBC_40_MD5
TLS_RSA_WITH_IDEA_CBC_SHA
TLS_RSA_EXPORT_WITH_DES40_CBC_SHA
TLS_RSA_WITH_DES_CBC_SHA
TLS_RSA_WITH_3DES_EDE_CBC_SHA
TLS_DH_DSS_EXPORT_WITH_DES40_CBC_SHA
TLS_DH_DSS_WITH_DES_CBC_SHA
TLS_DH_DSS_WITH_3DES_EDE_CBC_SHA
TLS_DH_RSA_EXPORT_WITH_DES40_CBC_SHA
TLS_DH_RSA_WITH_DES_CBC_SHA
TLS_DH_RSA_WITH_3DES_EDE_CBC_SHA
TLS_DHE_DSS_EXPORT_WITH_DES40_CBC_SHA
TLS_DHE_DSS_WITH_DES_CBC_SHA
TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA
TLS_DHE_RSA_EXPORT_WITH_DES40_CBC_SHA
TLS_DHE_RSA_WITH_DES_CBC_SHA
TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA
TLS_DH_anon_EXPORT_WITH_RC4_40_MD5
TLS_DH_anon_WITH_RC4_128_MD5
TLS_DH_anon_EXPORT_WITH_DES40_CBC_SHA
TLS_DH_anon_WITH_DES_CBC_SHA
TLS_DH_anon_WITH_3DES_EDE_CBC_SHA
TLS_RSA_EXPORT1024_WITH_RC4_56_MD5
TLS_RSA_EXPORT1024_WITH_RC2_CBC_56_MD5
TLS_RSA_EXPORT1024_WITH_DES_CBC_SHA
TLS_DHE_DSS_EXPORT1024_WITH_DES_CBC_SHA
TLS_RSA_EXPORT1024_WITH_RC4_56_SHA
TLS_DHE_DSS_WITH_RC2_56_CBC_SHA
TLS_DHE_DSS_WITH_RC4_128_SHA
TLS_DHE_DSS_WITH_NULL_SHA
SSL2_CK_RC4
SSL2_CK_RC4_EXPORT40
SSL2_CK_RC2
SSL2_CK_RC2_EXPORT40
SSL2_CK_IDEA
SSL2_CK_DES
SSL2_CK_RC464
SSL2_CK_3DES
TLS_ECDH_ECDSA_WITH_3DES_EDE_CBC_SHA
TLS_ECDH_ECDSA_WITH_RC4_128_SHA
SSL_RSA_WITH_RC2_CBC_MD5
TLS_ECDH_ECDSA_WITH_DES_CBC_SHA
TLS_ECDH_ECDSA_EXPORT_WITH_RC4_56_SHA
TLS_ECDH_ECDSA_EXPORT_WITH_RC4_40_SHA
```

Note that TLS_RSA_WITH_RC4_128_MD5 is a proven good choice.

It is also extremely advisable that testing teams record their own log output, as this may provide more immediately useful information. Such logs should also be submitted at the conclusion of testing.

Team-specific test records will be distributed on CD shortly after the event. Each CD will contain the records from each test in which a team participated.

4 Testing Tips

4.1 General Issues

These are some frequently made mistakes based on the experience at the dry run:

1) It is a **very good idea** to have a web-based interface for changing testing configuration. There are many changes that must be made during a single testing run, such as switching from front to back channel (i.e. HTTP-redirect to SOAP) and this should be as convenient and error-free as possible. The happiest testers are those that can change configuration with a click, rather than recompiling or editing config files.
2) The correct version number for Liberty 1.1 protocol messages and assertions is 1.0 [ProtSchema, Sect 3.1.2, line 202]. People had various other values for Major/MinorVersion in their messages, probably due to changes made to support phase 2 testing. If you are testing 1.1 specs, the version number should be 1.0.
3) For Liberty IDFF 1.2 protocol messages, the correct version number depends on the situation (from [ProtSchema 1.2, section 3.1.2, line 165]:
   - If the XML element containing the MajorVersion and MinorVersion attributes is in the Liberty ID-FF namespace (urn:liberty:iff:2003-08) or has an xsi:type attribute value in this namespace, then the MajorVersion MUST be 1 and the MinorVersion MUST be 2.
   - If the element or its type is in a SAML namespace (urn:oasis:names:tc:SAML:1.0:assertion or urn:oasis:names:tc:SAML:1.0:protocol), then the values MUST be 1 and 1 respectively.
4) Set your time correctly. The best choice is to use NTP. The event may provide an NTP server; the details will be provided if this happens.
5) Install the CA certificate provided, and trust it so that your browser and servers are not unhappy.
6) The SSL keys are provided in JKS and PKCS12 formats. Make sure to convert these to whatever form your servers need **before** you get to the event. Note that the JKS and P12 files share the same private key.
7) Make sure you will have GPRS roaming available for the network you plan to use, if you need to test using a mobile device.
8) Signature problems are difficult to debug. Make sure you sign the correct element, with the correct signing cert, and apply the signature correctly.
9) Kill and restart browser before starting testing and recording. In addition, make sure to reload pages to be sure you are not seeing a cached page.
10) Don't send personal or any other traffic using the SP, IDP, or LECP addresses, unless you want this traffic recorded by the network sniffer. You should be able to use the wireless/NAT network. Make sure you configure your IP settings correctly to achieve this. For example, by default the WiFi interface typically has a higher metric than Ethernet, so you will need to tweak this lower if you want you machine to use WiFi for "normal" traffic.

4.2 LECP Issues

This section is based on email from Robert Aarts, "Frequently Made Mistakes in LECP implementations for an SP or IdP".

1) The IdP should be able to handle the SOAP-enveloped AuthnRequest from an LECP on its **SingleSignOnURL**. Often implementers seem to think that as the LECP sends SOAP to the IdP, it will send it to the soap endpoint and don't expect the AuthnRequest envelope to arrive at the SingleSignonURL.
2) MIME types are not set correctly. Care is needed to ensure that e.g. a SOAP stack on an IdP does not set the content-type of the SOAP response to the LECP to "text/xml". Instead it should be "application/vnd.liberty-response+xml".
3) Not really a mistake, but not very clever, is the extensive use of redirects. Typically an IdP that receives a AuthnRequest from a LECP who's user is not yet authenticated responds with a redirect to another page on the IdP. This is not against the specs but in a mobile deployment this is an expensive redirect. Likewise often after the user fills out a login form and submits it to the IdP, the IdP again redirects the user agent back to itself, to a page (servlet) that will respond with the AuthnResponseEnvelope. I realize that in some environments the use of such redirects is justified, so this is not a black and white issue.

4) Related to the above mentioned redirects is a design where many of the transaction parameters are placed in the redirect URLs. This results in sometimes really long URLs, that will cause many (and not only mobile!) browsers to fail. Note that the maximum length of URLs in "action" etc. attributes can be shorter than the length of a URL in the main UI of the browser. In WML decks I recommend the use of <postfield> elements combined with a POST of the actual form.
5) Many web application environments put some default content in 302 redirect responses along the lines of "This page has been moved". I've encountered many problems with wml browser that would barf because the MIME type of such 302 redirects is not set to something that the browser can accept. If the actual "body" of the HTTP response is empty the MIME type doesn't seem to matter for Nokia browsers, I don't know about others. WAP GWs can also often be configured to "protect" the browser from unacceptable content types. Most web application environments can be configured to do particular 302 responses, so check it out.
6) Problems with cookies. It is almost impossible to properly support all cookie scenario's on a LEP. We actually do a quite much about it in our WAP GW but nevertheless I strongly recommend the following:
    a) try to avoid reliance on cookies for Liberty transactions. Use e.g. RelayState and URL-rewriting for things like local login pages. For true SSO the IdP needs to be able to set a cookie and our LEP supports that. It is impertinent though that your IdP sets the cookie on the application/vnd.liberty-response+xml response, i.e. the HTTP response that has the AuthnResponseEnvelope. That is because the Liberty code in the LEP will be invoked by the MIME type and it knows that it should keep that cookie and send it along with a future AuthnResponse to that same IdP.

       Now it so happens that most web application servers do set cookies "greedily". I.e. if your IdP throws a login page, it will likely set a cookie. This cookie comes in a non-Liberty response, so it will go to the phone. Which will nicely send it back to the IdP when the user submits the form. Hence your web-app/IdP will NOT send a cookie on the AuthnResponse, unless you explicitly do something about it. So I strongly recommend that you do something about it ;). Our WAP GW LEP will save and send the IdP-cookie, even if the user's phone has cookies turned off, or doesn't support cookies at all.
    b) On a real LEC cookies are no issue, this applies only to LEP implementations. In that sense you can feel pretty good about working with our LEP, it is the toughest case.
7) Most difficulties in testing the LECP support at an IdP or SP occur because our LEP is based upon WAP and WML. I'll try to arrange a setup that will facilitate easier testing of LECP implementations. The two hurdles are WAP (more precisely WTP) and WML. Our 3.1 Nokia WAP Gateway allows only for WTP between the browser and the gateway. Forum Nokia offers several phone browser simulators, that will do WTP. Many of these will only handle WML content but for example the Nokia Mobile Browser is rather flexible and can work with decent (x)html, i.e. html that is conformant to xml. So all elements are closed etc. Unfortunately we discovered a problem with the Nokia Mobile Browser: when it receives a 302 as a response to a POST over https it will prompt the user for confirmation of the redirect. That's fine but when the user selects "OK", the browser doesn't seem to do anything. I'll try to get a fix for that and/or find another browser. Note that this problem only occurs when issue 3) above applies and when we use https. It would be really nice if somebody could do a LECP plugin for Opera, Mozilla, or IE ;).
8) I will try to keep my "mini LEP" up and running over the next weeks and post contact details. As I don't want to configure everybody's metadata it is pertinent that in your testing the SP includes an IDPList in the AuthnRequestEnvelope that contains the a <ProviderID> and <Loc> element with the SingleSignOnURL of the IdP that you want to test. For the actual certification I recommend the same approach as we otherwise would have to change the "default" IdP that the LECP will use all the time.
9) In this setting you probably should not do https unless your SP and IdP use SSL certs that point to a very well known root CA cert. My WAP GW has only 20 of the most common root certs. For testing I can turn of the reverse DNS verification of the name on the cert.
10) Perhaps I come to think about more later, but this is enough for now.
11) Finally, of course the certification does NOT require that your LECP implementation supports WML. That's why I want a fix for that mobile browser as you then don't have to do WML. But likewise the certification doesn't require that a LECP implementation supports html. So an xhtml browser such as the Nokia Mobile Browser is I believe a reasonable "compromise".

4.2.1 WAP MIME Types

Here is some generic MIME type information for WAP browsers. Please note that Robert included the MIME type for SOAP response to a LECP above.

File Type	MIME Type	Description
.wml	text/vnd.wap.wml	An unencoded WML file.
.wmlc	application/vnd.wap.wmlc	An encoded WML file.
.gif	image/gif	Either a static (GIF87a) or animated (GIF89a) image.
.wbmp	image/vnd.wap.wbmp	A wireless bitmap (WBMP) image.

Other MIME types that MAY work on WAP phones are:

```
text/vnd.wap.wmlscript                  application/vnd.wap.wmlscriptc
application/xhtml+xml                   text/xml
application/vnd.wap.xhtml+xml           application/vnd.wap.coc
application/vnd.wap.sic                 application/vnd.wap.slc
application/vnd.wap.wbxml               application/vnd.wap.multipart.mixed
application/vnd.wap.multipart.related   application/vnd.wap.multipart.alternative
```