Global Information Assurance Certification Paper

Shibboleth SSO With 2-Factor on CentOS 6

GIAC GWEB Gold Certification

Author: Rich Graves, rgraves@carleton.edu
Advisor: Rob VandenBrink

Accepted: September 22nd 2014

Abstract

Shibboleth is a free, open-source web single sign-on solution (SSO) for complex federated environments based on the Security Assertion Markup Language (SAML). Installation is voluminously documented by the Shibboleth Consortium, but requires considerable time, expertise, and site-specific integration. To help system administrators and security analysts who are new to SAML and Shibboleth get started, a VMware image is provided with CentOS, OpenLDAP, Apache, Tomcat, and Shibboleth identity and service providers preconfigured to work together. Deployment of the virtual machine is discussed, including integration with Duo Security's two-factor authentication service. SSO security considerations are discussed in the context of Adam Shostack's STRIDE thread modeling framework.

Audience:

- System administrators asked to install Shibboleth and join a federation
- Penetration testers and security architects evaluating SAML installations
- Others interested in the theory and practice of web single sign-on

1. Introduction

Secure authentication and authorization across organizational boundaries is a hard problem. Consider an academic publisher that wishes to make scientific journals available to currently enrolled students, but not staff, faculty, or alumni, at universities that have paid a site license fee. Students could register with a site-specific username and password – though such credentials are likely to be shared or forgotten, diminishing security and increasing user frustration and support burden. Each school would also need to supply lists of authorized students – with obvious problems of ongoing maintenance, interoperability, and privacy. Perhaps the service provider (SP) could connect to the university's Active Directory to compare usernames, passwords, and organizational unit/group memberships in real time, but this does not scale well. Moreover, security-savvy enterprises are loath to allow external parties to process cleartext passwords and interact with core authentication servers.

SAML, the Security Assertion and Markup Language (OASIS, 2005a), is an international standard that attempts to address authentication across system and organizational boundaries in a very general way. It defines Extensible Markup Language (XML) schemata and application-layer protocols for communication between service providers (SPs), such as our hypothetical academic journal publisher, and identity providers (IdPs). IdPs at customer sites assert user identity and other attributes that SPs may use to make access control decisions. The glossary section of the SAML specification (OASIS, 2005b) is a good reference for those not familiar with the IdP/SP/assertions vocabulary. Superficially, SAML appears similar to OpenID Connect (OpenID Foundation, 2014), with which web developers may be more familiar because Facebook and Google use it. OpenID is relatively easy to deploy, being based on lightweight JSON and popular interpreted languages rather than the XML and (usually) C++ or Java of SAML implementations. However, SAML has been around far longer, has a more mature security model, and offers more features for distributed environments (Hodges, Technical Comparison: OpenID and SAML - Draft 07a, 2009). Thus, generalizing, SAML is widely adopted by business-to-business services and OpenID by consumer-facing web applications. Major enterprise service providers supporting SAML

Rich Graves, rgraves@carleton.edu

include ServiceNow (ServiceNow, 2014), Google Apps (Google, 2013), Salesforce (Salesforce, 2010), and Amazon AWS (Brauer, 2013). SAML becomes most interesting, though, in a massively multipolar world, such as academic research, where a large and fluid set of IdPs interact with a large and fluid set of SPs.

The high-level flow of an SP-Initiated single sign-on (SSO) exchange with the common Redirect and POST bindings is illustrated below. Note that unlike some SSO schemes, such as Cosign (University of Michigan, 2014), there is no out-of-band communication between the SP and the IdP. All messages pass through the browser.

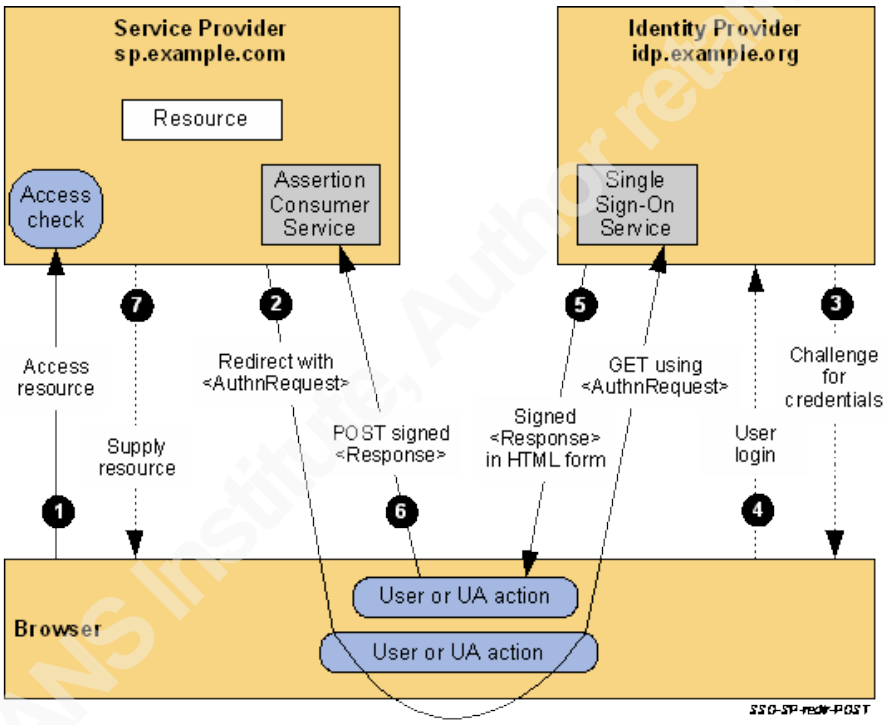


Figure 1: SP-Initiated SSO with Redirect and POST Bindings (OASIS, 2008)

Although the design feature of passing all messages through the user-agent potentially leaves room for mischief, it allows the system to scale better and has security benefits in some environments. For example, suppose the IdP is behind the enterprise firewall, unreachable from the SP or any other Internet site. Provided that public key encryption keys have been shared, SAML allows the isolated IdP to authenticate users to a public SP. Likewise, a public IdP could authenticate users to a private SP. The fact that the IdP-to-SP assertions are digitally signed and, optionally, encrypted end-to-end at the

XML level[1] addresses tampering and interception concerns at the browser. Unfortunately, the lack of a back channel (in most cases) does make single sign-out problematic (Shibboleth Consortium, 2013a) and puts a premium on the secrecy of the encryption keys used to sign assertions. What ultimately authenticates the user to the SP is the digitally signed POST response at step 6. If an attacker can obtain an IdP's secret key, they effectively become that IdP, able to sign on to any SP as any user without notice to the legitimate IdP. Potential issues with SAML trust models are further discussed in section 3, SSO Threat Modeling.

Shibboleth (Shibboleth Consortium, 2014b) is a free, open-source web single sign-on (SSO) suite implementing SAML and other standards. It was conceived by Internet2 and is supported by the international Shibboleth Consortium. Features that distinguish Shibboleth from other web SSO implementations include special attention to privacy and distributed, federated authentication. The Shibboleth Consortium provides an IdP as a Java web application, typically run under Apache Tomcat, and a "native SP" as C code that plugs into Apache HTTPD and Microsoft IIS. However, installing the IdP and SP is merely the start of the process. The IdP must get attributes (user names, email addresses, affiliations and levels of entitlement) from systems of record, often implemented as Lightweight Directory Access Protocol (LDAP) servers.

The Shibboleth trust model requires SPs and IdPs to exchange public keys (Internet2, 2014b), either bilaterally or through a trusted third party. A particular design goal and strength of Shibboleth is support for identity federations like InCommon (InCommon LLC, 2014). Federations set technical standards and serve a role similar to certification authorities. It is neither practical nor particularly secure for each IdP to keep track of the cryptographic keys of each SP, and vice versa, though this is effectively how SAML integration is done in ServiceNow and Salesforce. Delegating a certain amount of trust to the federation enables IdP/SP networks to scale up without administrative overhead. An organization's users may discover resources of which their IdP managers

[1] Alas, as discussed in the SSO threat modeling section, many commercial service providers lack support for XML encryption. Assertions should still be signed, protecting from tampering, but without end-to-end encryption, attributes are readable by the user (or malware running as the user).

are not even aware. For example, the Internet2 Filesender service, supporting secure transfer of terabyte-sized files, is available to any federated institution without registration (Internet2, 2012).

Installing and configuring a complete Shibboleth environment is a major endeavor. The Shibboleth wiki provides an intimidating list of technical and non-technical skills required (Klingenstein, 2009). In order to give the reader the opportunity for hands-on learning, a fully functional CentOS+OpenLDAP+Shibboleth+Duo virtual machine has been uploaded to http://go.carleton.edu/shibcentos6. Login and customization details are in Appendix A.

Centralized, federated SSO may have both positive and potential negative effects on security and usability. Reducing the number, frequency, and diversity of sites asking for passwords is thought to reduce user susceptibility to phishing attacks. Stanford's SSO web site attempts to inoculate users with the text, "Caution: Never enter your password on a web page unless that page's address bar points to weblogin.stanford.edu or unless you are establishing network connectivity" (Stanford University, 2014). Driving all authentication to one well-managed site may help concentrate system administrator effort. Users may be more willing to tolerate burdensome multi-factor authentication systems if they encounter them infrequently and with session persistence (Bonneau, Herley, Oorschot, & Stajano, 2012, p. 11). The balance of this paper covers security considerations for the specific case of a Shibboleth 2.4.1 IdP+SP combination installed on CentOS Linux 6.

2. Architectural Choices

As a broadly portable open-source software package, Shibboleth allows great freedom in deployment. Some decisions come down to personal or organizational preference, while others have real security impact.

A deployable VMWare image implementing the choices discussed in this section is available at http://go.carleton.edu/shibcentos6, with logon details in Appendix A.

2.1. Operating System, Packages, and Java Servlet Container

Shibboleth IdP should run on any platform that can run a compatible Java servlet container. There is even a 32-bit Windows QuickInstall (Widdowson, 2012), though it should not be used in production. In this project we used CentOS 6, a free recompile of RedHat Enterprise Linux 6, because it's an organizational standard. Ubuntu Linux is another common choice. We use OpenJDK 7 and Apache Tomcat 6 because they are the CentOS defaults and support Shibboleth well. Should you decide to install on your own CentOS system rather than use the provided VM (see Appendix A), please follow the step-by-step directions from the InCommon Shibboleth Workshop Series (Woodbeck, 2014) rather than struggling through the upstream Shibboleth documentation. As Internet2 architect Nate Klingenstein wrote when I posted that I was working to improve the documentation and provide a Linux demonstration VM, "Fantastic that you managed to get that much out of the distribution documentation. I couldn't and I helped to write it..." (personal communication, June 26, 2014).

There are a variety of philosophies and practices in the system administration community regarding software deployment. An organization with a strong DevOps focus (Garnichaud, 2012) should take a look at Elliot Kendall's Chef cookbook (Kendall, 2014). The Shibboleth developers, interested in cross-platform support, might recommend installing the latest (or specific prescribed) versions of Shibboleth and its many dependencies from source code. However, for a stable production deployment in small enterprise environments where Shibboleth is but one of a large constellation of applications, it makes most sense to maximize the use of packages provided by the Linux distribution. Even though CentOS/RHEL6 and the included Tomcat 6 are several years old, they remain fully supported. Digitally signed security and functional updates will be available until November 2020 (Red Hat, Inc., 2014). Another advantage of vendor-supplied packages is integration with distribution-specific security features like SELinux, as discussed in section 3.6.2.

2.2. Public-Facing Tomcat or Apache HTTPD Proxy?

Years ago, it was common to install multi-tier web application servers with Apache HTTPD as the front-end web server, with Tomcat or other servlet container

behind a reverse proxy. This is because Tomcat had limited features and a questionable security record as a web server. Today, it is more common to run Tomcat or Jetty "naked" on the Internet. InCommon's Shibboleth Workshop Series (Internet2, 2014c) documents a "quick and dirty" install where Tomcat runs as root, with no intermediate proxy. Clearly, it is dangerous to run Tomcat with unlimited access. There are several good ways to listen on the "low" ports 80 and 443 without running as root (Apache Software Foundation, 2014a), but that is not the only issue; other security architectural considerations below favor use of a separate Apache HTTPD front-end server.

A few files and directories need ownership and permissions changed when running Tomcat as an unprivileged user (default "tomcat").

```
# chgrp tomcat /opt/shibboleth-idp/logs /opt/shibboleth-idp/metadata
# chmod 770 /opt/shibboleth-idp/logs
# chmod 775 /opt/shibboleth-idp/metadata
# chgrp tomcat /etc/tomcat6/server.xml
# chmod 640 /etc/tomcat6/server.xml
```

These files and directories need to be changed to the "tomcat" group because the Shibboleth ./install.sh leaves them writable by root alone. The metadata directory is world-readable because it contains only public keys and generally available configuration information. In this demonstration, at least the local SP running as user "shibd" needs read access. Logs and server.xml, which contains the passphrase for SSL certs, should only be readable by root and tomcat.

The initiating and planning phases of this project happened to coincide with public disclosure of the Heartbleed vulnerability (Codenomicon, 2014). Although the upstream Java SSL implementation was not vulnerable to Heartbleed attacks, production Tomcat deployments, especially those serving HTTPS on port 443 while running as a non-root user, tend to use a native Apache Portable Runtime library that *was* vulnerable (Apache Software Foundation, 2014b). A successful Heartbleed attack on a Shibboleth IdP running as a monolithic Java/Tomcat process could expose not only user passwords and SSL server keys, but also the SAML signing keys, allowing forgery of arbitrary identity assertions. The serious implications of key compromise are discussed in section 3.1.3. Therefore, Shibboleth's Heartbleed security advisory recommends careful consideration of key replacement (Shibboleth Consortium, 2014a).

Rich Graves, rgraves@carleton.edu

The fully featured Apache HTTPD front end also supports header rewriting and other features that are not so easily supported in Tomcat alone. A web application firewall such as mod_security (Trustwave, 2014) could also be considered. Several examples of using Apache features to defend against potential Tomcat or Shibboleth security issues are presented in section 3.

2.3. Two-Factor Authentication with MCB and Duo

Shibboleth has an open plugin architecture. Any authentication scheme can be supported simply by writing, compiling, and deploying custom Java servlet code. For people for whom the idea of maintaining locally written sensitive security code in Java is not attractive, the Internet2 Multi Context Broker (MCB) (Wessel & Langenberg, 2014) and its Duo plugin (Langenberg, 2014) allow two-factor authentication with a fairly well documented binary distribution. The vendor Duo Security offers an open-source Shibboleth+Duo plugin, which is used at sites including Boston University (DuoSecurity, 2014) (Grundig, 2014). The demonstration virtual machine and the production deployment at Carleton College use MCB+Duo instead of Duo's own plugin because they are available as binary releases, not just source code (no small consideration at a small institution). In addition, the MCB layer allows the two-factor requirement to be imposed by an enterprise LDAP directory. In the case of the demonstration VM, a "description" attribute set to http://dev522.org/duo forces enrollment. A production deployment would extend the LDAP schema and use eduPersonAssurance for this purpose, but LDAP design is well beyond the scope of this paper.

3. SSO Threat Modeling

Before delving into specific threats, it is worthwhile to consider high-level risks to web single sign-on systems. What might an attacker seek to do? Based on a brainstorming scenario analysis (Shostack, Threat Modeling: Designing for Security, 2014, pp. 31-33), Figure 2 enumerates the main things that an enterprise IdP security administrator needs to worry about.

Risk Category	Example Attacker Scenarios
R1. Abuse system resources	Launch denial of service attack on a third party. Post links for search engine optimization. (IdP function is irrelevant.)
R2. Log on to specific SP as any user	Read expensive academic journals. Send spam from a webmail server.
R3. Log on to specific SP as specific user	Steal a specific user's email. Log on to ERP system to breach private data.
R4. Steal user passwords	Potentially access systems not even covered by SSO. Most often achieved through phishing, but sniffing and breach of credential stores are also options.
R5. Log on to any SP as any user	Subvert or replace IdP functionality. Find a way to make the IdP sign arbitrary SAML assertions, or steal private keys.

Figure 2: High level risks to enterprise IdPs

The purpose of an IdP is to securely vouch for the identity and attributes of logged-on principals so that downstream SPs can make access control decisions. Therefore, the very worst thing that can happen to an IdP is theft or abuse of the signing keys that are the foundation of trust.

Now, we look at more specific threats and vulnerabilities according to Adam Shostack's STRIDE framework (Shostack, Threat Modeling: Designing for Security, 2014, pp. 62-63). STRIDE does not attempt to provide a complete taxonomy, but is a developer-friendly mnemonic with an interactive card game (Shostack, Elevation of Privilege: Drawing Developers into Threat Modeling, 2012, p. 6) that plays a part in Microsoft's Security Development Lifecycle (SDL). Previous work has applied STRIDE to Shibboleth SPs supporting collaborators from a large number of research institutions (Meizner, Malawski, Naqvi, & Bubak, 2008). The well-known OWASP Top 10 checklist was also followed while building the Shibboleth demonstration VM, but STRIDE proved a better fit to Shibboleth's security model. Moreover, six points are easier to digest in a paper of this size than ten. STRIDE stands for:

Spoofing: Impersonating something or someone else.

Tampering: Modifying data or code.

Repudiation: Claiming to have not performed an action.

Information Disclosure: Exposing info to someone not authorized to see it.

Denial of Service: Deny or degrade service to users.

Elevation of Privilege: Gain capabilities without proper authorization.

Figure 3: The STRIDE framework (Shostack, Threat Modeling: Designing for Security, 2014)

3.1. Spoofing

Shibboleth is all about identity and access management, so spoofing is our primary concern. There are many ways that spoofing could come into play.

3.1.1. An attacker could spoof the login web site and collect credentials

SSL certificates are the primary technical means to counter spoofing. As previously mentioned, it is common for single sign-on pages to include text like "Caution: Never enter your password on a web page unless that page's address bar points to weblogin.stanford.edu or unless you are establishing network connectivity" (Stanford University, 2014). This has two problems. First, technically speaking, it places a lot of trust on the domain name system (DNS) and the public certification authority (CA) system, which may be misplaced (Marlinspike, 2011). Second, it relies on the user to examine the URL bar, to understand any browser security warnings that may appear, and to respond accordingly. It may not be realistic or even advisable to educate users to browse so carefully; the cost of their time and attention may outweigh the expected reduction in risk (Herley, 2009).

Another approach to mutual authentication, that is, proving to the user that web site is authentic, is epitomized by SiteKey (Bank of America, 2014). With SiteKey and similar site authentication image schemes, the server first asks only for username (identification), then presents a personal picture or phrase that is supposed to prove to the user that the site is genuine (server authentication), and only then asks for a password (client authentication). This is an appealing scheme that appears to provide more usable security than SSL. Many financial institutions have adopted SiteKey or similar in

response to regulatory guidance (Federal Financial Institutions Examination Council, 2005). However, a 2007 study showed that neither traditional browser security warnings nor site authentication images were effective in practice (Schechter, Dhamija, Ozment, & Fischer, 2007). One would hope that users have grown more savvy and wary of phishing since 2007, but a 2013 study specific to single sign-on systems largely reproduced those 2007 results, showing that 71% of participants were susceptible to phishing or man-in-the-middle network attacks (Yue, 2013).

The fact remains that web sites *can* be spoofed. Security should not be predicated on users' submitting passwords only to authentic sites.

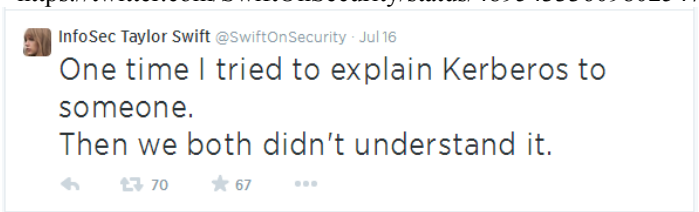
3.1.2. An attacker who gets a password can reuse it

At first this seems like no more than a syllogism, but Adam Shostack identifies this as a spoofing issue (Shostack, Threat Modeling: Designing for Security, 2014, pp. 501-502), and rightly so. Reusable passwords are not the only choice for authenticating users. IdPs that provide access to high-impact data or transactions should use stronger forms of authentication, such as Kerberos, TLS mutual authentication, or two-factor authentication. It is possible to authenticate to Shibboleth via Kerberos/SPNEGO[2] (Shibboleth Consortium, 2010a) and X.509 client certificates (Shibboleth Consortium, 2010b), but those approaches require client-side software that is difficult to use or simply unavailable on some devices. This paper discusses, and the associated virtual machine image implements (Appendix A), two-factor authentication with Duo Security.

Although it is not the purpose of this paper to endorse or review Duo, a little background is in order. Duo was founded by Dug Song, the author of dsniff (Song, 2001), who knows a thing or two about credential harvesting. In a typical deployment, the web application, in our case the Shibboleth MCB's UserPassword authenticator, first

---

[2] https://twitter.com/SwiftOnSecurity/status/489543336098025472



checks a primary local username/password authentication source such as LDAP or Kerberos. If that succeeds, then the application redirects to Duo's cloud service, which prompts for verification. For the second factor, Duo supports smartphone apps, SMS or pre-printed passcodes, telephone call back, and traditional hardware tokens. If the second stage succeeds, the user is redirected back to the web application with a digitally signed assertion that is a function of username, time, and application-specific public and private key components. The application checks the signature and allows or denies access accordingly. See Appendix B-C for some screen shots of Duo in action. Duo Security's web site (Duo Security, 2014) offers extensive documentation and marketing materials.

3.1.3. An attacker could spoof another IdP, Federation, or SP

Site spoofing and password reuse could apply to any application. There is also a class of attacks specific to SAML and Shibboleth. Due to the high level of complexity involved – each of Java, XML, and Public Key Infrastructure (PKI) is a complex discipline in itself, and Shibboleth relies heavily on all three – sites may be satisfied, even relieved, when they finally get an IdP or SP "working." This can be dangerous.

Look back at Figure 1: SP-Initiated SSO with Redirect and POST Bindings on page 3. The integrity of the authentication process depends entirely on digitally signed assertions backed by public key cryptography. A basic assumption is that the IdPs and SPs know each other's public keys. How is that done in practice? Shibboleth and other SAML software allows unvalidated PKI includes in relying-parties.xml, and some production deployments are running with something like this:

```
<MetadataProvider id="URLMD" xsi:type="FileBackedHTTPMetadataProvider"
   xmlns="urn:mace:shibboleth:2.0:metadata"
   minRefreshDelay="PT30M" maxRefreshDelay="PT2H"
   metadataURL=http://md.incommon.org/InCommon/InCommon-metadata.xml
   backingFile="/opt/shibboleth-idp/metadata/InCommon-metadata.xml" />
```

This XML fragment tells the IdP to fetch a new copy of the entire InCommon key ring every two hours. It trusts DNS to return the correct IP address of md.incommon.org.[3] It trusts the network to return valid content. It trusts that the server has not been compromised. The transfer does not use HTTPS, which in this case is more important for

[3] Incommon.org does use DNSSEC, but few organizations' DNS resolvers validate end-to-end.

server certification than encryption. There is an XML signature on InCommon-metadata.xml, but the above configuration never checks it. In the course of working on this paper, a number of IdPs and SPs were discovered that failed to validate metadata. The responsible parties were contacted, and the issues have been addressed to the best of their ability.

The proper way to fetch and validate federation metadata is demonstrated in the VM (Appendix A), and shown here:

[This content is taken from /opt/shibboleth-idp/conf/relying-parties.xml]

```
        <!-- InCommon Federation metadata provider. -->
        <!-- Reads metadata from a URL, stores a copy on the file system. -->
        <!-- Validates the signature of the metadata and filters out all
             but SP entities in order to save memory -->
        <metadata:MetadataProvider id="InCommonMD"
              xsi:type="metadata:FileBackedHTTPMetadataProvider"
              metadataURL=http://md.incommon.org/InCommon/InCommon-metadata.xml
              backingFile="/opt/shibboleth-idp/metadata/InCommon-metadata.xml"
              maxRefreshDelay="PT1H">
            <metadata:MetadataFilter xsi:type="metadata:ChainingFilter">
                <metadata:MetadataFilter xsi:type="metadata:SignatureValidation"
                                         trustEngineRef="InCommonTrust"
                                         requireSignedMetadata="true" />
                <metadata:MetadataFilter xsi:type="metadata:RequiredValidUntil"
                                         maxValidityInterval="P14D" />
                <metadata:MetadataFilter xsi:type="metadata:EntityRoleWhiteList">
                    <metadata:RetainedRole>samlmd:SPSSODescriptor</metadata:RetainedRole>
                </metadata:MetadataFilter>
            </metadata:MetadataFilter>
        </metadata:MetadataProvider>
```

[Much XML content skipped…]

```
        <!-- Trust engine used to evaluate the signature on loaded metadata. -->
        <security:TrustEngine id="InCommonTrust"
              xsi:type="security:StaticExplicitKeySignature">
            <security:Credential id="InCommonCredentials"
                  xsi:type="security:X509Filesystem">
                <security:Certificate>
                    /opt/shibboleth-idp/credentials/inc-md-cert.pem
                </security:Certificate>
            </security:Credential>
        </security:TrustEngine>
```

This more complete configuration improves security in two ways. It still fetches the metadata via HTTP, because InCommon only publishes this public key data via HTTP. But the fetched data's digital signature is verified against the public key stored in the inc-md-cert.pem file (two sections highlighted in blue) and digital signatures valid for more than 14 days are ignored (highlighted in yellow). For additional background on metadata signing and trust issues, see the Shibboleth and InCommon documentation wikis (Scavo T. , 2014a) (Internet2, 2014c).

Rich Graves, rgraves@carleton.edu

3.1.4. An attacker could steal and reuse credentials stored on the client

Within the STRIDE framework, this is classified as a spoofing risk because an attacker is spoofing the client's identity to the server. Two-factor authentication, with Duo or some other solution, helps to address this threat. Some may quibble that if the second factor is a smartphone or a "Trusted Device" browser cookie, the second factor may be vulnerable alongside the first. This is true. Evaluate your entire threat environment and usability factors[4] and choose authentication technologies accordingly. The quest to replace passwords is long and arduous (Bonneau, Herley, Oorschot, & Stajano, 2012).

3.2. Tampering

With the scope of this paper limited to IdPs and their relationship with SPs, "tampering" mostly applies to the alteration of SAML messages. As shown in Figure 1: SP-Initiated SSO with Redirect and POST Bindings, all messages pass through the client web browser. They may also be vulnerable in transit.

3.2.1. An attacker can manipulate data because there's no integrity protection for data on the network

SAML standards offer both integrity and confidentiality protection. Further, most IdPs and SPs run over SSL. Avoid disabling attribute signing/encryption unless a vendor insists that their IdP or SP software will work no other way.

3.2.2. An attacker can change parameters over a trust boundary and after validation (for example, important parameters in a hidden field in HTML, or passing a pointer to critical memory)

This tricky attack pattern did affect some SAML implementations, including Shibboleth, Salesforce, and IBM XS40. Security researchers discovered and responsibly disclosed a XML Signature wrapping (XSW) attack against the underlying OpenSAML library (Somorovsky, Mayer, Schwenk, Kampmann, & Jensen, 2012). In essence, the attack smuggles extra assertions inside a signed XML document without invalidating the

[4] In most environments, it is not realistic to ban the storage of passwords on smartphones.

signature. The issue was resolved in Shibboleth IdP version 2.3.2 and Shibboleth SP version 2.4.3 (Shibboleth Consortium, 2011). Shibboleth code quality is quite high and the features intentionally sparse – the IdP has never used the problematic Apache Struts library (Apache Software Foundation, 2014c), for example – but it is important to stay up to date.

3.3. Repudiation

For this paper, we are only concerned with the authentication phase, not specific transactions. It may be good to know that if you are interested in "step up" authentication, where basic username/password is sufficient for most applications and transactions but a few SPs or actions require second-factor verification, Shibboleth can do that (University of Michigan, 2013). There is one specific bug or architectural flaw from the STRIDE framework to keep in mind.

3.3.1. An attacker can make a log lose or confuse information

A July thread on the Shibboleth users mailing list discussed strange log entries possibly attributable to "The Jester," an independent hacktivist whose activities were the subject of a SANS research paper (O'Connor, 2011). The thread ended ambiguously, with Shibboleth developer Scott Cantor writing, "I'm still fascinated by the log message. It suggests to me that the Java container is not doing canonical naming. That's not a huge exposure, but it bears some thought and it definitely should be done if possible" (Cantor, 2014). The extent to which this odd artifact indicates a vulnerability is ripe for future research. What is clear is that it is possible to inject log entries that could send the IdP administrator on a wild goose chase. Continuing to maintain Apache logs, containing the "real" browser requests, as distinct from the possibly subvertable Tomcat and IdP logs, may mitigate this issue.

3.4. Information Disclosure

In the Shibboleth context, Information Disclosure issues comprises both encryption and application-level logic.

3.4.1. Basic SSL server configuration and security X-headers

The demonstration VM (Appendix A) strives to implement best practices for key storage, cryptographic ciphers, and headers, sufficient to score an A+ on the SSL Labs (Appendix D). Basic Apache server hardening is beyond the scope of this paper, but a few highlights of the configuration follow.

We use this block in /etc/httpd/conf.d/ssl.conf so that Apache renders all cookies secure and HTTP-only, even when the internal Java application server does not (Kehlet, 2012):

```
Header edit Set-Cookie "(?i)^((?:(?!;\s?HttpOnly).)+)$" "$1; HttpOnly"
Header edit Set-Cookie "(?i)^((?:(?!;\s?Secure).)+)$" "$1; Secure"
```

Selected headers for the initiation of a representative SAML redirect challenge/response are reproduced in Figure 4:

```
GET /idp/profile/SAML2/Redirect/SSO?SAMLRequest=<long encoded string> HTTP/1.1
Host: login.dev522.org

HTTP/1.1 302 Moved Temporarily

Content-Length: 0
Strict-Transport-Security: max-age=15768000; includeSubDomains
X-Frame-Options: deny
X-UA-Compatible: IE=edge
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Set-Cookie: JSESSIONID=<LongString>; Path=/idp; Secure; HttpOnly
Set-Cookie: _idp_authn_lc_key=<LongString>; Version=1; Path=/idp; Secure; HttpOnly
Location: https://login.dev522.org:443/idp/AuthnEngine
Content-Security-Policy: connect-src 'self' https://login.dev522.org; script-src
'self' https://login.dev522.org https://*.duosecurity.com 'unsafe-inline'; style-
src 'self' 'unsafe-inline' https://login.dev522.org https://*.duosecurity.com;
report-uri http://login.dev522.org/csp-report-test.php
Content-Type: text/plain; charset=UTF-8
```

Figure 4 HTTP Security Headers

HTTP Strict-Transport-Security tells modern browsers[5] to remember the fact that a site uses HTTPS, and to refuse to connect via HTTP (Hodges, Jackson, & Barth, RFC 6797, 2012). Note that all cookies are set "Secure," requiring HTTPS transport, and "HttpOnly," not available to JavaScript. The other headers will be discussed in the next section.

X-Frame-Options: deny stops many click-jacking attacks (OWASP, 2014). The X-UA-Compatible, X-Content-Type-Options, and X-XSS-Protection headers tell Internet

[5] As of this writing, all major browsers except Microsoft Internet Explorer support HSTS.

Explorer to use its latest and most standards-compliant rendering engine, to use only the Content-Type provided by the web server (and not to guess HTML based on file content), and to activate strict cross-site scripting protection.

Content-Security-Policy (CSP) merits a few more words. CSP is a relatively new standard that allows a server to ban a variety of content types from unexpected sources, thereby preventing many classes of injection attack (Barth, Veditz, & West, 2013). Unfortunately, both Shibboleth and the Internet2 MCB plugin generate inline JavaScript, so until they are re-architected to support CSP, the "unsafe-inline" token is required. However, even with this lax policy, interesting and useful results are obtained. One unexpected finding from real-world CSP deployment at Carleton College is that the Chrome and Safari browsers apply CSP to spyware browser plug-ins. For example, this was submitted to our report-uri by a browser infected with the potentially unwanted "Savings Slider":

```
{"csp-report":{"document-uri":"https://login.carleton.edu/idp/Authn/UserPassword",
"violated-directive":"script-src 'self' https://login.carleton.edu",
"blocked-uri":"https://savingsslider-a.akamaihd.net",
"source-file":"https://savingsslider-a.akamaihd.net",
"line-number":57,"column-number":400,"status-code":0}}
```

This one appears to be the potentially unwanted Conduit toolbar (herdProtect, 2014):

```
{"csp-report":{"document-uri":"https://login.carleton.edu/idp/Authn/UserPassword",
"violated-directive":"script-src 'self' https://login.carleton.edu",
"blocked-uri":"",
"source-file":"chrome-extension://plmlpkfpkijnlijgalnjaacllnjmoamo",
"line-number":106,"column-number":10,"status-code":0}}
```

Although the effectiveness of the CSP policy is limited by the requirement to support inline JavaScript, it is still helpful.

3.4.2. An attacker can read content because messages (say, an email or HTTP cookie) aren't encrypted even if the channel is encrypted

The idea that SAML messages should be signed and encrypted has already been mentioned several times in this paper. Barring an XSW bug (section 3.2.2), not much mischief is enabled by the knowledge that an SAML assertion contains an email address and the fact that the subject is an enrolled student. The stakes are raised if the assertions contain more sensitive information that the user would not ordinarily know (class rank?),

or if there is malware in the browser. Therefore, best practice is to sign and encrypt assertions so that only the IdP and SP may read them.

3.4.3. An attacker can discover the fixed key being used to encrypt

Protection of the web server-level HTTPS certificate is important, but as mentioned in the spoofing section, the SAML assertion signing certificate is even more critical. If that key is compromised, whether by means of a Heartbleed-style bug, poor permissions, or exposed backups, then an attacker can forge signed SAML assertions and log on to any SP as any user.

3.4.4. Disclosing too much information to an untrustworthy SP

Shibboleth's design aspirations emphasize privacy. In a way, it offers a form of David Chaum's credentials without identification (Chaum, 1986). Users sign on to their local IdP, then are given cryptographic bearer tokens with assertions like "is an active student at Example University" that allow access to resources without the need for site-specific identification. Alas, this utopian ideal is seldom reached due to issues at the SP and IdP. It is rare for service providers to serve exclusively Shibboleth or SAML users. Their web applications are often designed with a local authentication option that requires a username, email address, and full name, with SAML as an optional pre-authentication layer on top. In order to satisfy the application's data model, SAML must provide all identifiers – even though there is little business need. On the IdP side, it can be onerous to keep track of which SPs have a legitimate need for which attributes. Organizations may adopt an attribute release policy that is too lax. Helpful tip: Shibboleth IdP includes an Attribute Authority, Command Line Interface tool (aacli.sh) that shows, for a given SP, what attributes would be released (Shibboleth Consortium, 2013b).

3.5. Denial of Service

This study found no significant denial of service vulnerabilities specific to Shibboleth. All web applications may be attacked. As with the phishing threat, centralization brings additional risks and opportunities. If an attacker takes down a single central IdP, then all users or SPs that rely on that IdP are affected. It might seem to be wise to distribute the authentication tasks more widely. On the other hand, applying well-

understood denial of service protection techniques including web application firewalls, horizontal scaling, and IP anycast to a focused, well-defended IdP cluster could be better.

3.6. Elevation of Privilege

Finally, we come to the last major heading within the STRIDE framework, elevation of privilege. The sections on spoofing and tampering already discussed attacking the core purpose of the IdP as an arbiter of identity and privilege, as bestowed by attribute resolution and release policies. Previously discussed methods of impersonating another user *are* elevation of privilege. There remain, though, a few additional points to cover.

3.6.1. An attacker can force data through different validation paths which give different results (SAML dialects)

"Different validation paths" is how Shostack frames the general pattern. As applied to Shibboleth, it makes sense to reduce the complexity and attack surface of IdP and SPs alike by limiting the protocols and protocol versions supported to the minimum necessary. Recently, Tom Scavo mooted a "Recommended Protocol Support for New IdPs" document with key points (Scavo T. , 2014b):

- **DO** support SAM2 Web Browser SSL on the front channel
- **DON'T** support back-channel SAML protocols

Up to this point, this paper has glossed over the possibility of back-channel, out-of-band IdP-to-SP SAML communication and SAML version 1. This was intentional. Neither should be needed in 2014. Neither the demonstration VM (Appendix A) nor Carleton College's production IdP implement these protocols. Whether specific vulnerabilities can be identified at this time is in some sense irrelevant. They are deprecated, not needed, and not getting a lot of developer attention, so they should be turned off.

3.6.2. Security Enhanced Linux (SELinux)

SELinux is another complicated piece of software, originally sponsored by the US National Security Agency, which is enabled by default on RHEL6 and CentOS 6. It implements a form of mandatory access control (MAC) in the Linux kernel. Even if there is a flaw in the web application or server, and even if discretionary access control (DAC)

permissions in the file system would allow access, SELinux constrains the activities of daemons and their child processes to prescribed actions only. An excellent introduction to the topic is the perennial RedHat presentation "SELinux for Mere Mortals" (Cameron, 2013). By way of example, the default SELinux policy happens to break the Shibboleth SP by preventing Apache HTTPD from communicating with the privilege-separated shibd over the UNIX domain socket /var/run/shibboleth/shibd.sock. This access is actually needed, so the login.dev522.org virtual machine contains this policy in /root/shibd-httpd.te:

```
module shibd-httpd 1.0;

require {
        type var_run_t;
        type httpd_t;
        type initrc_t;
        class sock_file write;
        class unix_stream_socket connectto;
}

#============= httpd_t ==============
allow httpd_t initrc_t:unix_stream_socket connectto;
allow httpd_t var_run_t:sock_file write;
```

This allows HTTPD to connect to UNIX domain sockets (normally denied) and to write (send data) only to socket files labeled var_run_t, such as /var/run/shibboleth/shibd.sock. This policy does not need to grant read of files labeled var_run_t, because that's already allowed by default. Granting these exceptions with semodule –i shib-httpd.pp allows Shibboleth SP to work unhindered while still running SELinux in full enforcing mode.

4. Conclusion

Shibboleth has a well-deserved reputation as a complex, intimidating bit of software. Adding two-factor authentication and securing the surrounding operating system and web server environment makes it more so. Participating in the Shibboleth user community, updating their wiki, developing a demonstration VM, and presenting preliminary findings to peers has been an immensely rewarding experience. I hope that

the work embodied in Appendix A continues to be useful to the community. Adam Shostack's STRIDE framework helped put this work in a theoretical context and brought to the surface some potential and actual vulnerabilities that might otherwise have escaped notice. As with so many cryptographic applications, the devil is in the key management and implementation details.

Rich Graves, rgraves@carleton.edu

5. References

Apache Software Foundation. (2014a, April 17). *HowTo*. Retrieved July 20, 2014, from Tomcat Wiki: http://wiki.apache.org/tomcat/HowTo#How_to_run_Tomcat_without_root_privileges.3F

Apache Software Foundation. (2014b). *Security/Heartbleed*. Retrieved July 20, 2014, from Tomcat Wiki: https://wiki.apache.org/tomcat/Security/Heartbleed

Apache Software Foundation. (2014c). *Announcements*. Retrieved July 21, 2014, from Struts: http://struts.apache.org/announce.html

Bank of America. (2014). *SiteKey Security from Bank of America*. Retrieved August 30, 2014, from Bank of America: https://www.bankofamerica.com/privacy/online-mobile-banking-privacy/sitekey.go

Barth, A., Veditz, D., & West, M. (2013, June 4). *Content Security Policy 1.1*. Retrieved September 7, 2014, from World Wide Web Consortium (W3C): http://www.w3.org/TR/2013/WD-CSP11-20130604/

Bonneau, J., Herley, C., Oorschot, P. C., & Stajano, F. (2012, March). *The quest to replace passwords: a framework for comparative evaluation of Web authentication schemes*. Retrieved from Computer Laboratory, University of Cambridge: http://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-817.pdf

Brauer, B. (2013, November 11). *AWS Identity and Access Management Using SAML*. Retrieved July 20, 2014, from Amazon Web Services: http://aws.amazon.com/blogs/aws/aws-identity-and-access-management-using-saml/

Cameron, T. (2013, June 13). *SELinux for Mere Mortals*. Retrieved September 8, 2014, from YouTube: http://youtu.be/bQqX3RWn0Yw

Cantor, S. (2014, July 22). *SAML Spoofing*. Retrieved from users@shibboleth.net Gmane archive: http://comments.gmane.org/gmane.comp.web.shibboleth.user/35088

Chaum, D. (1986, January). Showing credentials without identification: Transferring signatures between unconditionally unlinkable pseudonyms. (F. Pichler, Ed.) *Proc. of a workshop on the theory and application of cryptographic techniques on Advances in cryptology---EUROCRYPT '85*, 241-244.

Codenomicon. (2014). Retrieved July 20, 2014, from Heartbleed Bug: http://heartbleed.com/

Duo Security. (2014). Retrieved August 30, 2014, from Two-Factor Authentication Made Easy - Duo Security: https://www.duosecurity.com/

DuoSecurity. (2014, June 10). *Duo two-factor authentication for Shibboleth*. Retrieved August 30, 2014, from GitHub: https://github.com/duosecurity/duo_shibboleth

Federal Financial Institutions Examination Council. (2005, October 12). *Authentication in an Internet Banking Environment*. Retrieved from FFIEC: http://www.ffiec.gov/press/pr101205.htm

Garnichaud, N. (2012, December 24). *What Exactly is DevOps?* Retrieved September 11, 2014, from Dr. Dobb's: http://www.drdobbs.com/architecture-and-design/what-exactly-is-devops/240009147

Rich Graves, rgraves@carleton.edu

Google. (2013, August 19). *SAML Single Sign-On (SSO) Service for Google Apps*. Retrieved July 20, 2014, from Google Developers: https://developers.google.com/google-apps/sso/saml_reference_implementation

Grundig, T. (2014). Duo at BU – Our plan for enabling two factor authorization for all employees. *BU Security Camp*. Boston. Retrieved August 30, 2014, from http://www.bu.edu/tech/services/security/services/security-events-training/camp/archives/sc2014/talk-descriptions-and-slides/#SC14_TG

herdProtect. (2014, April 20). *plmlpkfpkijnlijgalnjaacllnjmoamo.crx*. Retrieved from herdProtect: http://www.herdprotect.com/plmlpkfpkijnlijgalnjaacllnjmoamo.crx-f1e2c465f781426df4d6522e5b6048bc8d3467eb.aspx

Herley, C. (2009). So long, and no thanks for the externalities: the rational rejection of security advice by users. *Proceedings of the 2009 Workshop on New Security Paradigms* (pp. 133-144). New York: ACM. doi:10.1145/1719030.1719050

Hodges, J. (2009, July 3). *Technical Comparison: OpenID and SAML - Draft 07a*. Retrieved from IdentityMeme.org: http://identitymeme.org/doc/draft-hodges-saml-openid-compare.html

Hodges, J., Jackson, C., & Barth, A. (2012, November). *RFC 6797: HTTP Strict Transport Security (HSTS)*. Retrieved from IETF Tools: http://tools.ietf.org/html/rfc6797

InCommon LLC. (2014). *InCommon Federation*. Retrieved June 25, 2014, from InCommon: Security, Privacy and Trust for the Research and Education Community: http://www.incommon.org/federation/

Internet2. (2012, July 16). *FileSender*. Retrieved July 20, 2014, from Internet2: http://www.internet2.edu/products-services/filesender/

Internet2. (2014a, March 18). *Metadata Trust Models*. Retrieved June 25, 2014, from Internet2 Wiki: https://spaces.internet2.edu/display/InCFederation/Metadata+Trust+Models

Internet2. (2014b, January 1). *X.509 Certificates in Federation Metadata*. Retrieved June 25, 2014, from Internet2 Wiki: https://spaces.internet2.edu/display/InCFederation/X.509+Certificates+in+Metadata

Internet2. (2014c, July 24). *Shibboleth Workshop Series - Linux Identity Provider (Centos 6.5)*. Retrieved September 3, 2014, from Internet2 Wiki: https://spaces.internet2.edu/display/ShibInstallFest/Shibboleth+Workshop+Series+-+Linux+Identity+Provider+%28Centos+6.5%29

Kehlet, S. (2012, July 25). *Add HttpOnly flag to cookies on the fly with Apache?* Retrieved September 4, 2014, from Stack Overflow: http://stackoverflow.com/a/11660223

Kendall, E. (2014). *chef-shibboleth_idp*. Retrieved July 20, 2014, from elliotKendallUCSF: https://github.com/elliotkendallUCSF/chef-shibboleth_idp

Klingenstein, N. (2009, July 30). *IdPSkills*. (P. Schober, Ed.) Retrieved July 20, 2014, from Shibboleth Documentation: https://wiki.shibboleth.net/confluence/display/SHIB2/IdPSkills

Langenberg, D. (2014, June 24). *DUO Security Module for the Shibboleth Multi-Context Broker*. Retrieved August 30, 2014, from GitHub: https://github.com/uchicago/mcb-duo/

Rich Graves, rgraves@carleton.edu

Marlinspike, M. (2011, April 11). *Blog >> SSL and the Future of Authenticity*. Retrieved August 30, 2014, from Moxie Marlinspike: http://www.thoughtcrime.org/blog/ssl-and-the-future-of-authenticity/

Meizner, J., Malawski, M., Naqvi, S., & Bubak, M. (2008, October 9). *Threat Model for MOCCA*. Retrieved from Maciej Malawski homepage: http://home.agh.edu.pl/~malawski/p17-tmd.pdf

Morris, J., & Brindle, J. (2014). Retrieved July 22, 2014, from SELinux Project Wiki: http://selinuxproject.org/page/Main_Page

OASIS. (2005a, March 15). *Security Assertion Markup Language (SAML) v2.0*. Retrieved from Organization for the Advancement of Structured Information Standards (OASIS): http://docs.oasis-open.org/security/saml/v2.0/saml-2.0-os.zip

OASIS. (2005b, March 15). *Glossary for the OASIS Security Assetion Markup Language (SAML) V2.0*. Retrieved from Organization for the Advancement of Structured Information Standards (OASIS): http://docs.oasis-open.org/security/saml/v2.0/saml-glossary-2.0-os.pdf

OASIS. (2008, March 25). *Security Assertion Markup Language (SAML) V2.0 Technical Overview*. Retrieved July 21, 2014, from OASIS Docs: http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-tech-overview-2.0.html

O'Connor, T. (2011, December 30). *The Jester Dynamic: A Lesson in Asymmetric Unmanaged Cyber Warfare*. Retrieved September 4, 2014, from SANS Reading Room: http://www.sans.org/reading-room/whitepapers/attacking/jester-33889

OpenID Foundation. (2014, February 14). *OpenID Connect*. Retrieved from OpenID Foundation website: http://openid.net/connect/

OWASP. (2014, July 23). *Clickjacking Defense Cheat Sheet*. Retrieved September 7, 2014, from Open Web Application Security Project: https://www.owasp.org/index.php/Clickjacking_Defense_Cheat_Sheet

Red Hat, Inc. (2014). *Red Hat Enterprise Linux Life Cycle*. Retrieved July 20, 2014, from Red Hat: https://access.redhat.com/support/policy/updates/errata/

Salesforce. (2010). *About Identity Providers and Service Providers*. Retrieved July 20, 2014, from Salesforce.com Help Portal: https://help.salesforce.com/apex/HTViewHelpDoc?id=identity_provider_about.htm&language=en

Scavo, T. (2014a, January 26). *Refresh InCommon Metadata*. Retrieved September 3, 2014, from Shibboleth Documentation: https://wiki.shibboleth.net/confluence/display/SHIB2/IdPMetadataProviderExamples

Scavo, T. (2014b, April 20). *Protocol Support for New IdPs*. Retrieved September 8, 2014, from InCommon Federation: https://spaces.internet2.edu/display/InCFederation/Protocol+Support+for+New+IdPs

Schechter, S., Dhamija, R., Ozment, A., & Fischer, I. (2007). The Emperor's New Security Indicators. *IEEE Symposium on Security and Privacy* (pp. 51-65). Berkeley: IEEE. doi:10.1109/SP.2007.35

Schwartz, M. (2013, October 25). *Two factor authentication is not the answer*. Retrieved from Gluu: http://www.gluu.org/blog/2fa_not_enough/

Rich Graves, rgraves@carleton.edu

ServiceNow. (2014, June 26). *SAML 2.0 Web Browser SSO Profile*. Retrieved July 20, 2014, from ServiceNow Product Documentation: http://wiki.servicenow.com/index.php?title=SAML_2.0_Web_Browser_SSO_Profile

Shibboleth Consortium. (2010a). *Kerberos Login Handler*. Retrieved September 3, 2014, from Shibboleth Documentation: https://wiki.shibboleth.net/confluence/display/SHIB2/Kerberos+Login+Handler

Shibboleth Consortium. (2010b). *X.509 Login Handler*. Retrieved September 3, 2014, from Shibboleth Documentation: https://wiki.shibboleth.net/confluence/display/SHIB2/X.509+Login+Handler

Shibboleth Consortium. (2011, July 11). *Shibboleth Security Advisory*. Retrieved from Shibboleth: http://shibboleth.net/community/advisories/secadv_20110725.txt

Shibboleth Consortium. (2013a, November 7). *SLOIssues*. Retrieved July 21, 2014, from Shibboleth Wiki: https://wiki.shibboleth.net/confluence/display/SHIB2/SLOIssues

Shibboleth Consortium. (2013b, October 15). *AACLI*. Retrieved September 8, 2014, from Shibboleth Documentation: https://wiki.shibboleth.net/confluence/display/SHIB2/AACLI

Shibboleth Consortium. (2014a, April 9). *Shibboleth Security Advisory*. Retrieved from Shibboleth: http://shibboleth.net/community/advisories/secadv_20140409.txt

Shibboleth Consortium. (2014b, May 30). *What's Shibboleth?* Retrieved from Shibboleth: http://shibboleth.net/about/

Shibboleth Consortium. (2014c). *IdPConfigConfig*. Retrieved July 20, 2014, from Shibboleth: https://wiki.shibboleth.net/confluence/display/SHIB2/IdPConfigConfig

Shostack, A. (2012, December 18). *Elevation of Privilege: Drawing Developers into Threat Modeling*. Retrieved from Microsoft Download Center: http://download.microsoft.com/download/F/A/E/FAE1434F-6D22-4581-9804-8B60C04354E4/EoP_Whitepaper.pdf

Shostack, A. (2014). *Threat Modeling: Designing for Security*. Indianapolis: John Wiley & Sons.

Somorovsky, J., Mayer, A., Schwenk, J., Kampmann, M., & Jensen, M. (2012, August 23). *On Breaking SAML: Be Whoever You Want to Be*. Retrieved from USENIX: https://www.usenix.org/conference/usenixsecurity12/technical-sessions/presentation/somorovsky

Song, D. (2001). Retrieved September 3, 2014, from dsniff: http://www.monkey.org/~dugsong/dsniff/

Stanford University. (2014). Retrieved June 25, 2014, from Stanford WebLogin: https://webmail.stanford.edu/

Trustwave. (2014). Retrieved September 5, 2014, from ModSecurity: Open Source Web Application Firewall: https://www.modsecurity.org/

University of Michigan. (2013, December). *Configuring Your Service Provider for Step-Up Two-Factor Authentication*. Retrieved September 4, 2014, from ITS Information System: http://www.itcs.umich.edu/itcsdocs/s4397/

University of Michigan. (2014, January 15). *Cosign Overview*. Retrieved July 21, 2014, from CoSign: web single sign-on: http://weblogin.org/overview.shtml

Rich Graves, rgraves@carleton.edu

Wessel, K., & Langenberg, D. (2014). Multifactor Authentication, Assurance, and the Multi-Context Broker. *IAM Online Webinar Series*. InCommon. Retrieved April 30, 2014, from http://www.incommon.org/iamonline/

Widdowson, R. (2012, June 22). *IdPQuickInstall*. (S. Hopkins, Ed.) Retrieved July 20, 2014, from Shibboleth Documentation: https://wiki.shibboleth.net/confluence/display/SHIB2/IdPQuickInstall

Woodbeck, D. (2014, May 5). *Shibboleth Workshop Series - Linux Service Provider (CentOS 6.5)*. (N. Klingenstein, Ed.) Retrieved July 20, 2014, from Internet2 Wiki: https://spaces.internet2.edu/x/LoLNAQ

Yue, C. (2013). The Devil is Phishing: Rethinking Web Single Sign-On Systems Security. *Proceedings of the 6th USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET)* (pp. 1-4). Washington, D.C.: USENIX. Retrieved August 30, 2014, from https://www.usenix.org/conference/leet13/workshop-program/presentation/yue

Rich Graves, rgraves@carleton.edu

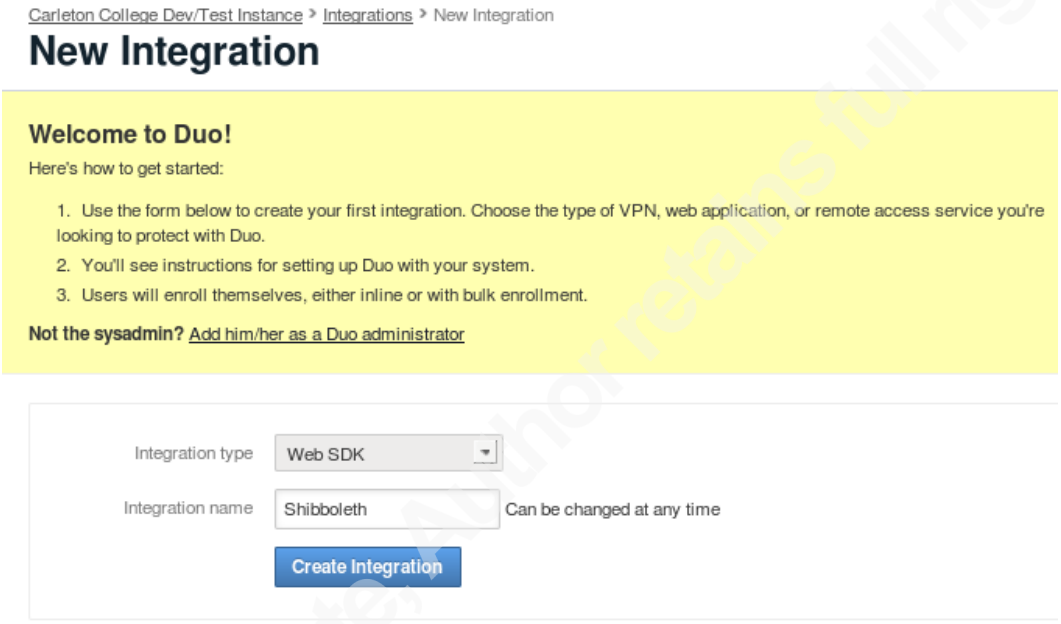
Appendix A: Shibboleth+MCB+Duo VM README

```
The virtual machine is available at http://go.carleton.edu/shibcentos6
Should this link become invalid, you can find me on Twitter @richgraves

This is a fully functional demo system pre-integrated with CentOS 6.5,
Shibboleth IdP 2.4.1, Shibboleth SP 2.5.3, the Internet2 Multi Context
Broker 1.1.4, and the Duo MCB plugin 2.0.1. The IdP is configured to
authenticate to a local OpenLDAP server.

HowTo:

1. Install the OVF in VMWare Workstation 9+, ESX 5+, or other
virtualization platform. NAT networking is sufficient.

2. The root password is "shibboleth." Log on. You might want to change
the root password, but since the firewall is limited to RFC1918 space
and you are probably NATed, you don't have to.

3. Check the IP addres (ifconfig). Firefox, Burp proxy, and various web
analysis plugins are available in the VM, but if you want to use a non-
virtual web browser, enter "<ip address>  shib-centos6.dev522.org" into
your /etc/hosts file (Windows: C:\windows\system32\drivers\etc\hosts).

4. Navigate to https://shib-centos6.dev522.org/secure/

5. Log on with username "user1" and password "1" (the single digit 1).

6. Other users (all but user1) are configured to require Duo 2-factor
authentication. To set this up, you need to go to
https://signup.duosecurity.com/ and create a generic "web SDK
integration." The "Shibboleth integration" offered refers to Duo's own
demo plugin, which lacks some handy features of the Multi Context
Broker. (See Appendix B for some Duo screenshots.)

7. The parameters obtained from Duo, and a random APPKEY, go in
/opt/shibboleth-idp/conf/mcb-spring.xml

8. /sbin/service tomcat6 restart (or touch
/usr/share/tomcat6/webapps/idp.war, slightly faster)

9. Log on as "user2" with password "2". You should be prompted to
enroll with Duo. The users actually go all the way to 200, so that you
can experiment fully.

10. If you want to view/edit the LDAP directory, the root account is
preauthenticated locally. Use commands like: ldapsearch –Y EXTERNAL –H
ldapi:/// uid=user2; ldapmodify -Y EXTERNAL -H ldapi:///

11. Setting a user's "description" attribute to http://dev522.org/duo
forces 2-factor. "urn:oasis:names:tc:SAML:2.0:ac:classes:Password" sets
password authentication only.

Comments and corrections to rcgraves@gmail.com or Twitter @richgraves
```

Rich Graves, rgraves@carleton.edu

Appendix B: Setting Up a New Duo Integration

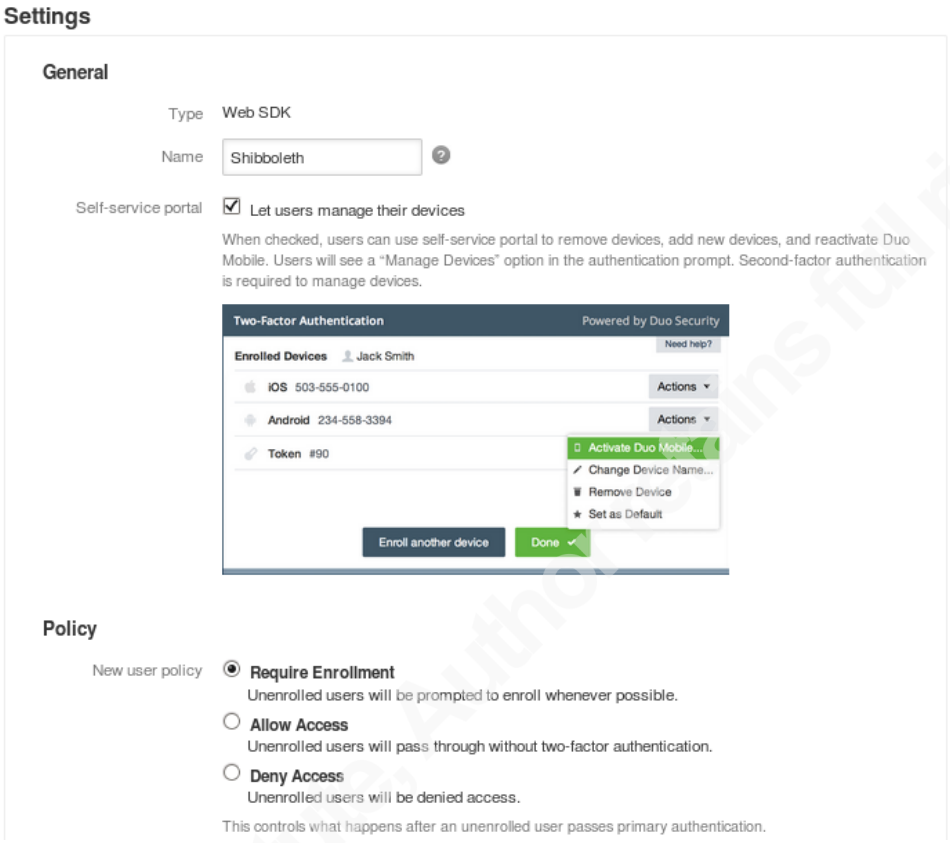
The Duo Security administration user interface is fairly clear and has integral online help. These screenshots are provided as an illustration of functionality as of August 2014. Creating a new integration:



When you "Create Integration" above, Duo provides identification and secret strings.

Rich Graves, rgraves@carleton.edu

There are many client options that can be configured:



More options to "remember" devices or opt out certain IP ranges:

Rich Graves, rgraves@carleton.edu

Appendix C: Duo Authentication Log

On August 28, 2014, I presented a webinar on this Shibboleth+MCB+Duo project to a closed information security community. In addition to distributing the VM (Appendix A), I hosted a public instance. Below is a sample of what the Duo service logged. Focusing on the Event, Factor, and Result columns:

- "Enrollment" indicates that a user passed primary LDAP authentication and was required to use a second factor, but had not yet configured any. This site was configured to allow end-user self-service enrollment.
- "Phone Call" and "SMS Passcode" should be self-explanatory.
- "Duo Push" is their smartphone app, which uses device-specific public-key encryption to prevent attacks like the RSA breach.
- "Trusted Device" means that an authenticated user previously checked the "Remember this device for 30 days" box, which saves a browser cookie.
- "Fraud" in the "Result" column means that the user explicitly repudiated a logon attempt, a choice offered by the smartphone app and phone call back options.

Carleton College Dev/Test Instance › Authentication Log

Authentication Log

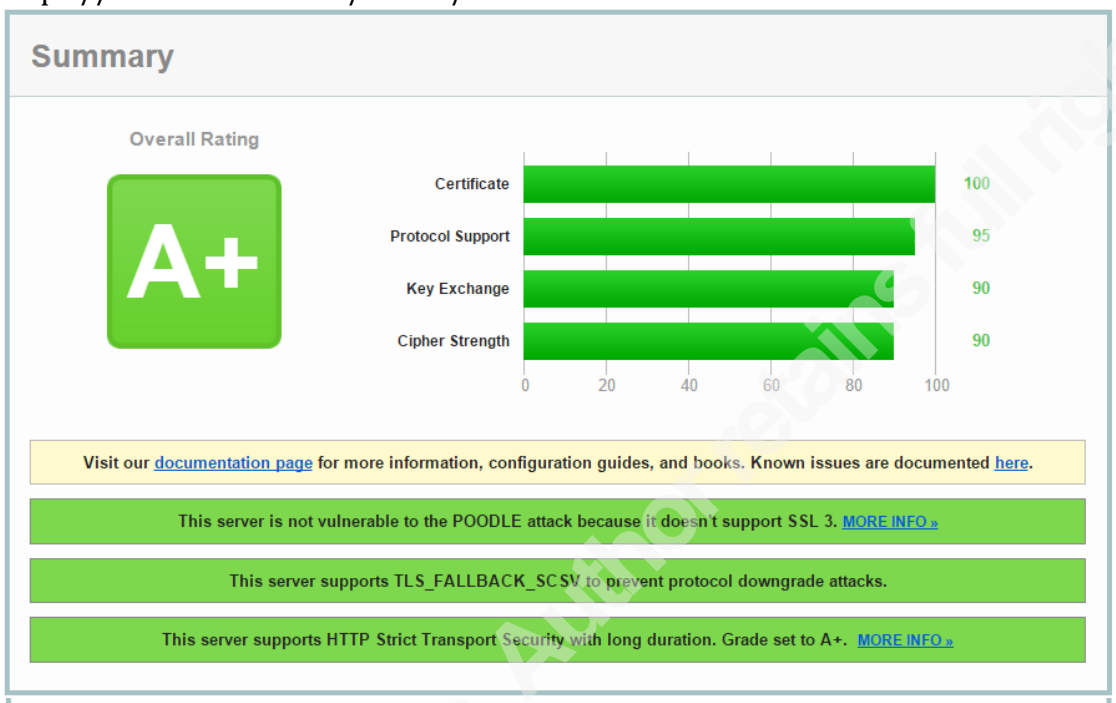
CSV JSON

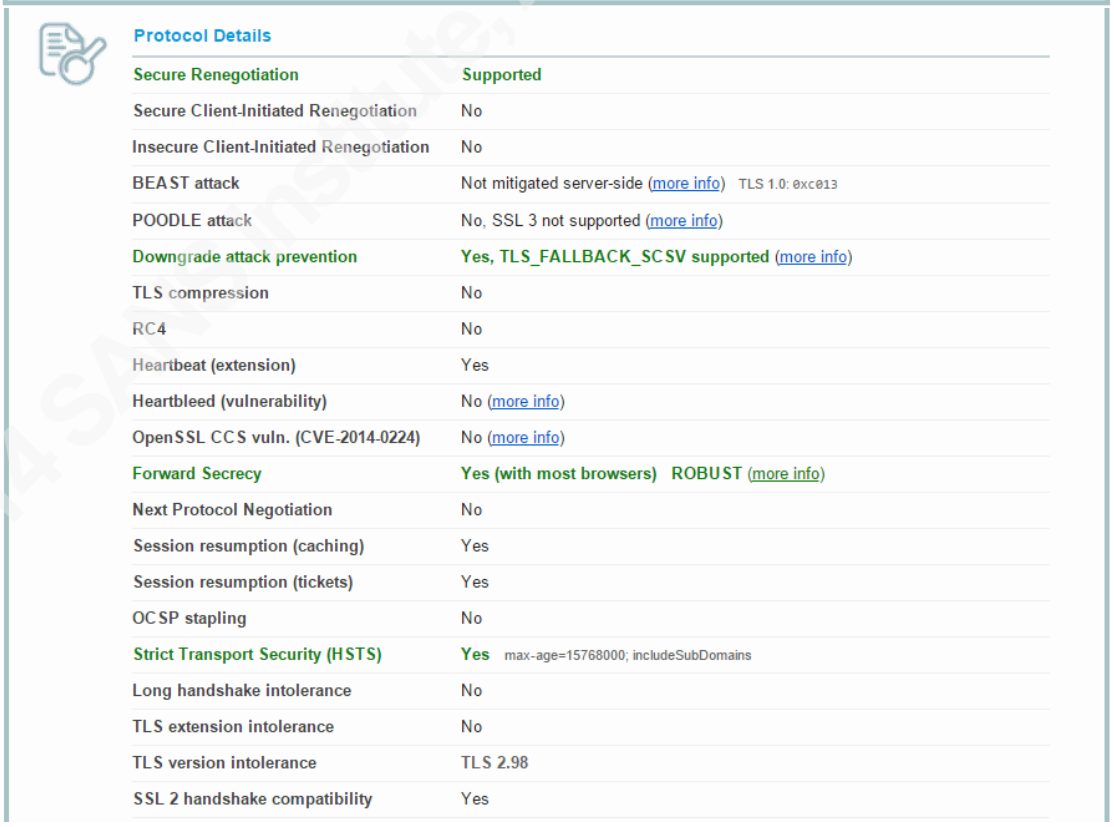
Timestamp	User	Integration	Event	Factor	Result	IP Address	Location
Sep 1, 2014 12:13 PM CDT	user111	Shibboleth	Authentication	Phone Call	Success	Redacted	, PA, United States
Aug 29, 2014 11:05 AM CDT	user175	Shibboleth	Authentication	Duo Push	Success		, ID, United States
Aug 28, 2014 12:07 PM CDT	user13	Shibboleth	Authentication	Duo Push	Success		ID, United States
Aug 28, 2014 12:07 PM CDT	user13	Shibboleth	Authentication	Duo Push	Fraud		, ID, United States
Aug 28, 2014 12:06 PM CDT	user13	Shibboleth	Authentication	Duo Push	Failure		, ID, United States
Aug 28, 2014 12:04 PM CDT	user13	Shibboleth	Authentication	Duo Push	Success		, ID, United States
Aug 28, 2014 12:03 PM CDT	user13	Shibboleth	Authentication	Duo Push	Failure		, ID, United States
Aug 28, 2014 12:01 PM CDT	user13	Shibboleth	Authentication	Phone Call	Success		, ID, United States
Aug 28, 2014 12:01 PM CDT	user13	Shibboleth	Authentication	Phone Call	Failure		, ID, United States
Aug 28, 2014 12:00 PM CDT	user32	Shibboleth	Authentication	SMS Passcode	Success		, NC, United States
Aug 28, 2014 11:58 AM CDT	user166	Shibboleth	Authentication	Duo Push	Success		, IA, United States
Aug 28, 2014 11:57 AM CDT	user166	Shibboleth	Enrollment		Success		, IA, United States
Aug 28, 2014 11:56 AM CDT	user114	Shibboleth	Authentication	Duo Push	Success		C, United States
Aug 28, 2014 11:54 AM CDT	user32	Shibboleth	Authentication	Duo Push	Success		, NC, United States
Aug 28, 2014 11:54 AM CDT	user34	Shibboleth	Authentication	Phone Call	Success		, WI, United States
Aug 28, 2014 11:54 AM CDT	user34	Shibboleth	Authentication	Phone Call	Failure		, WI, United States
Aug 28, 2014 11:52 AM CDT	user100	Shibboleth	Authentication	Duo Push	Success		, NY, United States
Aug 28, 2014 11:51 AM CDT	user34	Shibboleth	Authentication	Phone Call	Success		, WI, United States
Aug 28, 2014 11:51 AM CDT	user100	Shibboleth	Enrollment		Success		NY, United States
Aug 28, 2014 11:51 AM CDT	user176	Shibboleth	Authentication	Phone Call	Success		, MT, United States
Aug 28, 2014 11:50 AM CDT	user114	Shibboleth	Authentication	Duo Push	Success		, NC, United States
Aug 28, 2014 11:47 AM CDT	user13	Shibboleth	Authentication	Duo Push	Failure		, ID, United States
Aug 28, 2014 11:47 AM CDT	user114	Shibboleth	Enrollment		Success		, NC, United States
Aug 28, 2014 11:46 AM CDT	user175	Shibboleth	Enrollment		Success		, ID, United States
Aug 28, 2014 11:45 AM CDT	user13	Shibboleth	Authentication	Duo Push	Success		, ID, United States
Aug 28, 2014 11:43 AM CDT	user101	Shibboleth	Enrollment		Success		, OH, United States
Aug 28, 2014 11:42 AM CDT	user1	Shibboleth	Authentication	Trusted Device	Success		, UT, United States
Aug 28, 2014 11:41 AM CDT	user1	Shibboleth	Authentication	Trusted Device	Success		UT, United States
Aug 28, 2014 11:41 AM CDT	user1	Shibboleth	Authentication	Duo Push	Success		, UT, United States
Aug 28, 2014 11:40 AM CDT	user49	Shibboleth	Authentication	Duo Push	Success		, ID, United States

Rich Graves, rgraves@carleton.edu

Appendix D: SSL Labs Report

The demonstration VM scores an A+ on the Qualys SSL Labs test, https://www.ssllabs.com/ssltest/





Protocol Details

Secure Renegotiation	Supported
Secure Client-Initiated Renegotiation	No
Insecure Client-Initiated Renegotiation	No
BEAST attack	Not mitigated server-side (more info) TLS 1.0: 0xc013
POODLE attack	No, SSL 3 not supported (more info)
Downgrade attack prevention	Yes, TLS_FALLBACK_SCSV supported (more info)
TLS compression	No
RC4	No
Heartbeat (extension)	Yes
Heartbleed (vulnerability)	No (more info)
OpenSSL CCS vuln. (CVE-2014-0224)	No (more info)
Forward Secrecy	Yes (with most browsers) ROBUST (more info)
Next Protocol Negotiation	No
Session resumption (caching)	Yes
Session resumption (tickets)	Yes
OCSP stapling	No
Strict Transport Security (HSTS)	Yes max-age=15768000; includeSubDomains
Long handshake intolerance	No
TLS extension intolerance	No
TLS version intolerance	TLS 2.98
SSL 2 handshake compatibility	Yes

Rich Graves, rgraves@carleton.edu

```
# From /etc/httpd/conf.d/ssl.conf
SSLProtocol all -SSLv2 -SSLv3
# From https://wiki.mozilla.org/Security/Server_Side_TLS
SSLCipherSuite "ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-
SHA256:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-SHA384:DHE-
RSA-AES128-GCM-SHA256:DHE-DSS-AES128-GCM-SHA256:kEDH+AESGCM:ECDHE-RSA-
AES128-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:ECDHE-
ECDSA-AES128-SHA:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES256-
SHA384:ECDHE-RSA-AES256-SHA:ECDHE-ECDSA-AES256-SHA:DHE-RSA-AES128-
SHA256:DHE-RSA-AES128-SHA:DHE-DSS-AES128-SHA256:DHE-RSA-AES256-
SHA256:DHE-DSS-AES256-SHA:DHE-RSA-AES256-SHA:AES128-GCM-SHA256:AES256-
GCM-SHA384:AES128-SHA256:AES256-SHA256:AES128-SHA:AES256-
SHA:AES:CAMELLIA:DES-CBC3-
SHA:!aNULL:!eNULL:!EXPORT:!DES:!RC4:!MD5:!PSK:!aECDH:!EDH-DSS-DES-CBC3-
SHA:!EDH-RSA-DES-CBC3-SHA:!KRB5-DES-CBC3-SHA"
SSLHonorCipherOrder on
```

Rich Graves, rgraves@carleton.edu

Appendix E: Automatically Reloading IdP Config Files

The Shibboleth IdP configuration comprises several XML files, which in the case of the demonstration VM are stored in /opt/shibboleth-idp/conf. A common operational problem is that adding a new service provider or altering the attribute release policy requires a restarting at least the IdP servlet, which means about one minute of user-affecting downtime. The Shibboleth developer documentation does address this issue, but obliquely and without enthusiasm (Shibboleth Consortium, 2014c). Their preferred answer is that all changes should be made on a test system and rolled to production during a change window. More explicit directions for enabling configuration reloading are included here because it can be immensely useful.

The parent configuration file for the demonstration VM is /opt/shibboleth-idp/conf/service.xml. Adding the configurationResourcePollingFrequency attribute to srv:Service nodes in this file tells the Shibboleth IdP servlet to check the included files for changes and, if needed, reload without downtime. A particularly nice feature is that there is a try/catch around the reload, so if a change introduces an XML syntax error, the previous working configuration stays. For example, this service.xml fragment will poll the attribute filter policy file every 5 minutes:

```
<srv:Service
         id="shibboleth.AttributeFilterEngine"
         xsi:type="attribute-afp:ShibbolethAttributeFilteringEngine"
         configurationResourcePollingFrequency="PT5M"
         configurationResourcePollingRetryAttempts="5">
    <srv:ConfigurationResource
              file="/opt/shibboleth-idp/conf/attribute-filter.xml"
              xsi:type="resource:FilesystemResource"/>
</srv:Service>
```

If we edit attribute-filter.xml with the typo PolicyRequirement**Rool** for PolicyRequirement**Rule**, the log at /opt/shibboleth-idp/logs/idp-process.log will say:

```
00:05:34.178 - DEBUG
[org.opensaml.util.resource.ResourceChangeWatcher:204] - Publishing
update event for resource: /opt/shibboleth-idp/conf/attribute-filter.xml
00:05:34.179 - INFO
[edu.internet2.middleware.shibboleth.common.config.BaseService:158] -
Loading new configuration for service shibboleth.AttributeFilterEngine
00:05:37.748 - ERROR
[edu.internet2.middleware.shibboleth.common.config.BaseService:188] -
Configuration was not loaded for shibboleth.AttributeFilterEngine
service, error creating components.  The root cause of this error was:
```

Rich Graves, rgraves@carleton.edu

```
org.xml.sax.SAXParseException: cvc-complex-type.2.4.a: Invalid content
was found starting with element 'PolicyRequirementRool'. One of
'{"urn:mace:shibboleth:2.0:afp":PolicyRequirementRule,
"urn:mace:shibboleth:2.0:afp":PolicyRequirementRuleReference}' is
expected.
00:05:38.175 - ERROR
[edu.internet2.middleware.shibboleth.common.config.BaseReloadableService:
197] - Error reloading configuration, upon configuration resource update,
for service shibboleth.AttributeFilterEngine
edu.internet2.middleware.shibboleth.common.service.ServiceException:
Configuration was not loaded for shibboleth.AttributeFilterEngine
service, error creating components.
[stack trace follows…]
```

And when corrected:

```
00:10:34.179 - INFO
[edu.internet2.middleware.shibboleth.common.config.BaseService:158] -
Loading new configuration for service shibboleth.AttributeFilterEngine
00:10:36.603 - DEBUG
[edu.internet2.middleware.shibboleth.common.config.attribute.filtering.At
tributeFilterPolicyGroupBeanDefinitionParser:64] - Parsing attribute
filter policy group ShibbolethFilterPolicy
00:10:36.618 - INFO
[edu.internet2.middleware.shibboleth.common.config.attribute.filtering.At
tributeFilterPolicyBeanDefinitionParser:72] - Parsing configuration for
attribute filter policy releaseTransientIdToAnyone
[various policy parsing snipped…]
00:10:37.425 - INFO
[edu.internet2.middleware.shibboleth.common.config.BaseService:180] -
shibboleth.AttributeFilterEngine service loaded new configuration
```

Although the Shibboleth developers' point about carefully staging change windows is well taken, dynamic reload is a time-saver for test systems and, thanks to the try/catch, a failsafe for production systems as well. A full servlet restart is riskier because in case of error, the service stays down. Just be careful that on-the-fly configuration reloading doesn't become an enabler for poor change control processes.

Rich Graves, rgraves@carleton.edu