Foundstone
Professional Services
A DIVISION OF McAFEE

» Fuzzing XML Based Protocols (SAML)

Hacks-In-Taiwan 2006

Yen-Ming Chen
Senior Principal Consultant
Foundstone, A Division of McAfee

Agenda

- Introduction
  - SAML
  - OpenSAML
- Scenarios
- Implementation
- Conclusion

SAML

- Security Assertion Markup Language (SAML)
- Codified by OASIS with participation from MACE and others
- Defines XML Schema for AuthN and attribute assertions, queries, responses, and use profiles such as Web SSO.
- Defines bindings to protocols for transport
- V2.0 expands SAML and includes definitions from Shibboleth and the Liberty Alliance

SAML in a Nutshell

- An XML-based framework for exchanging security information
  - XML-encoded security assertions
  - XML-encoded request/response protocol
  - Rules on using assertions with standard transport and messaging frameworks
- An OASIS standard (1.0, 1.1, and 2.0)
  - Vendors and users involved
  - OpenSAML implementation available
  - Codifies current system outputs *vs*. creating new technology

OpenSAML

- OpenSAML for the message and assertion formats, and protocol bindings which is based on Security Assertion Markup Language (SAML)
- SAML (Security Assertion Markup Language) is a standard for the formation and exchange of authentication, attribute, and authorization data as XML. It describes various kinds of messages and standard ways of transporting them.
- OpenSAML is a set of open-source libraries in Java and C++ which can be used to build, transport, and parse SAML messages.

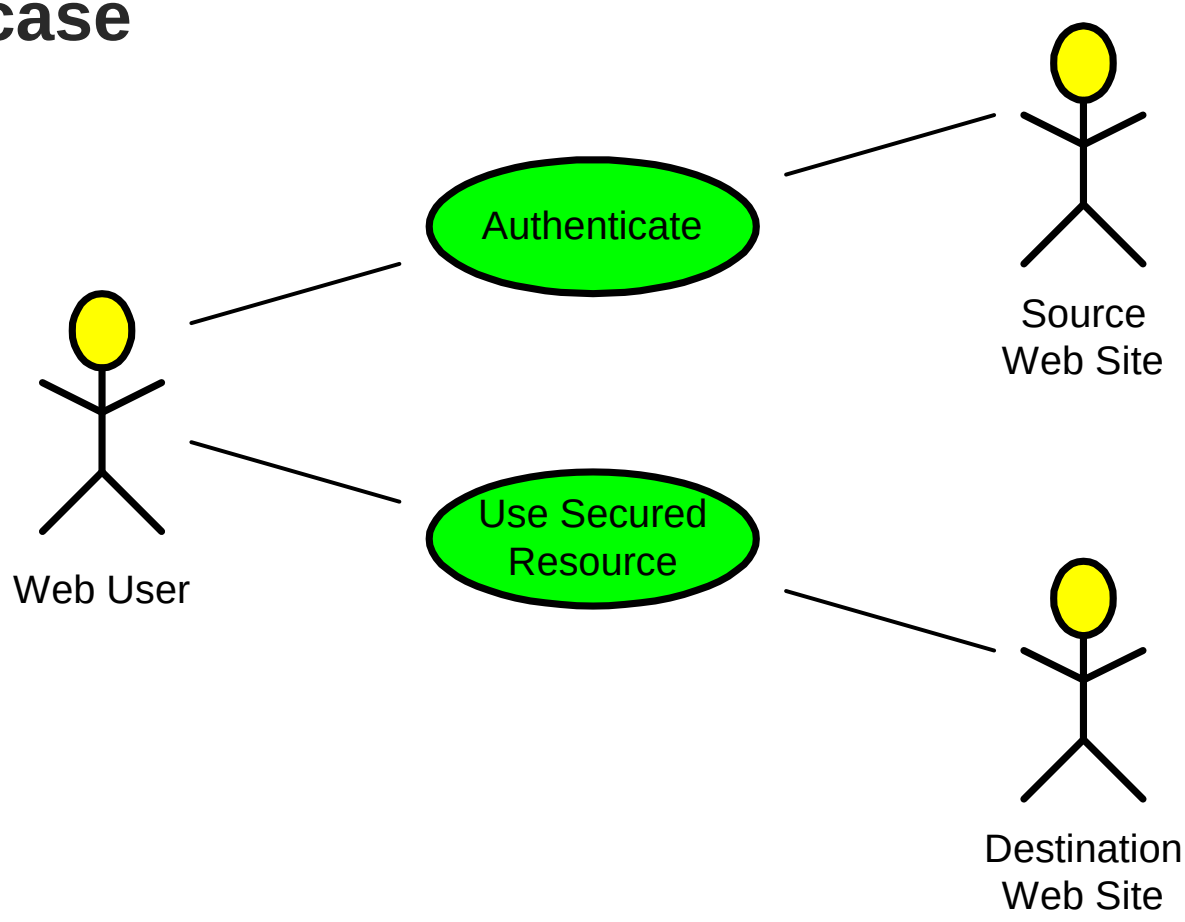
Technology

- Basic concepts
  - Subject/principal
    - User or application requesting access to a resource
  - Assertion
    - Set of statements about a subject
  - Authority
    - Entity that produces and/or consumes assertions
  - Binding
    - Specification for transporting assertions as protocol payloads
  - Profile
    - Specification describing rules for embedding, transferring, extracting, and processing assertions

Technology

- Use cases
  - Web single sign-on (SSO)
    - User logs onto source site and implicitly requests brokered logon to one or more destination sites with pre-existing trust relationships to source site
  - Authorization
    - Once having logged onto trusted destinations via SSO, user requests authorized access to various resources controlled by destinations
  - Back-office transactions
    - User attaches assertions to electronic business document and transmits to relying party

SSO use case



Assertion Title Syntax

Assertion

--Identifier
--Issuer
--Issuance timestamp
--Conditions
--Advice

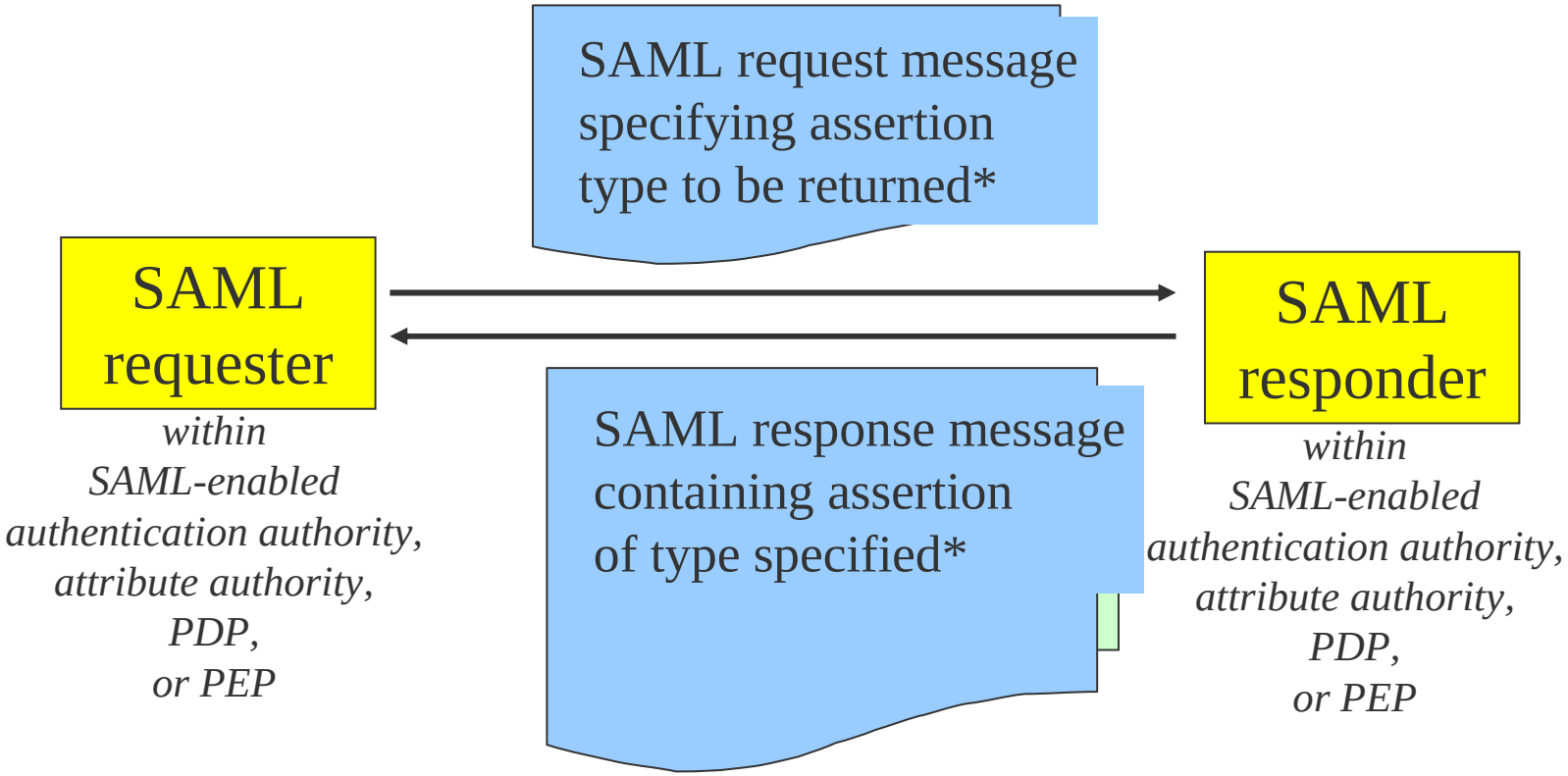
Statement

Authentication Statement

Attribute Statement

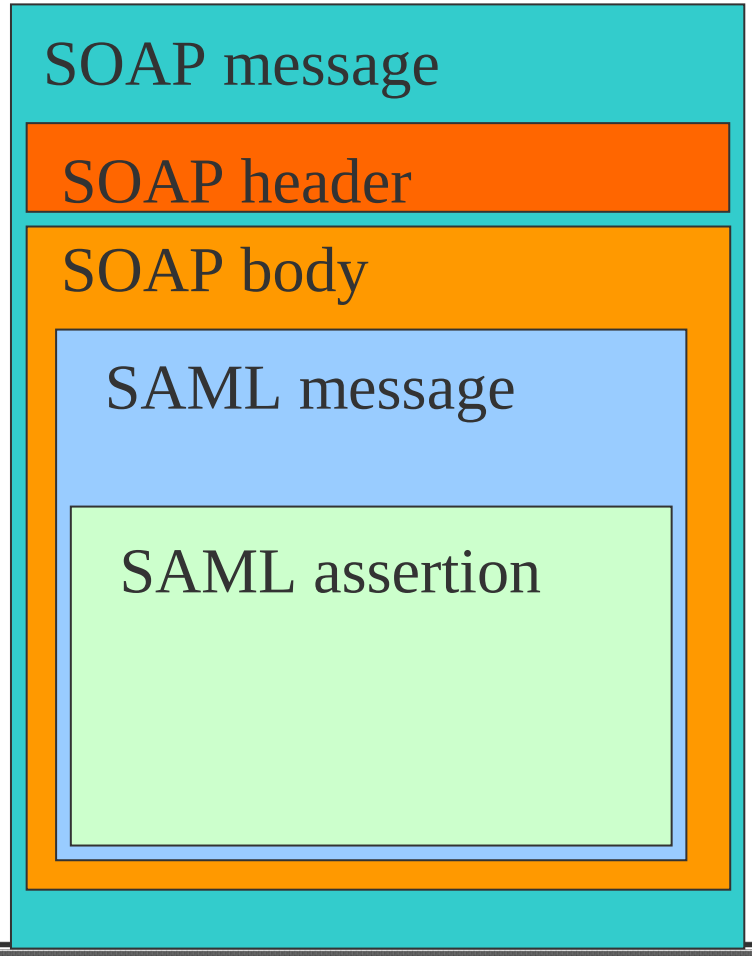
Authorization Decision Statement

Message Exchange Protocol



*optionally, SAML messages may be digitally signed via XML Signatures, or sent over secure Transport Layer Security (TLS) channels

Binding with SOAP



SAML assertions

- An assertion is a declaration of fact about a subject, e.g. a user
  - (according to some assertion issuer)
- SAML has three kinds, all related to security:
  - Authentication
  - Attribute
  - Authorization decision
- You can extend SAML to make your own kinds of assertions
- Assertions can be digitally signed

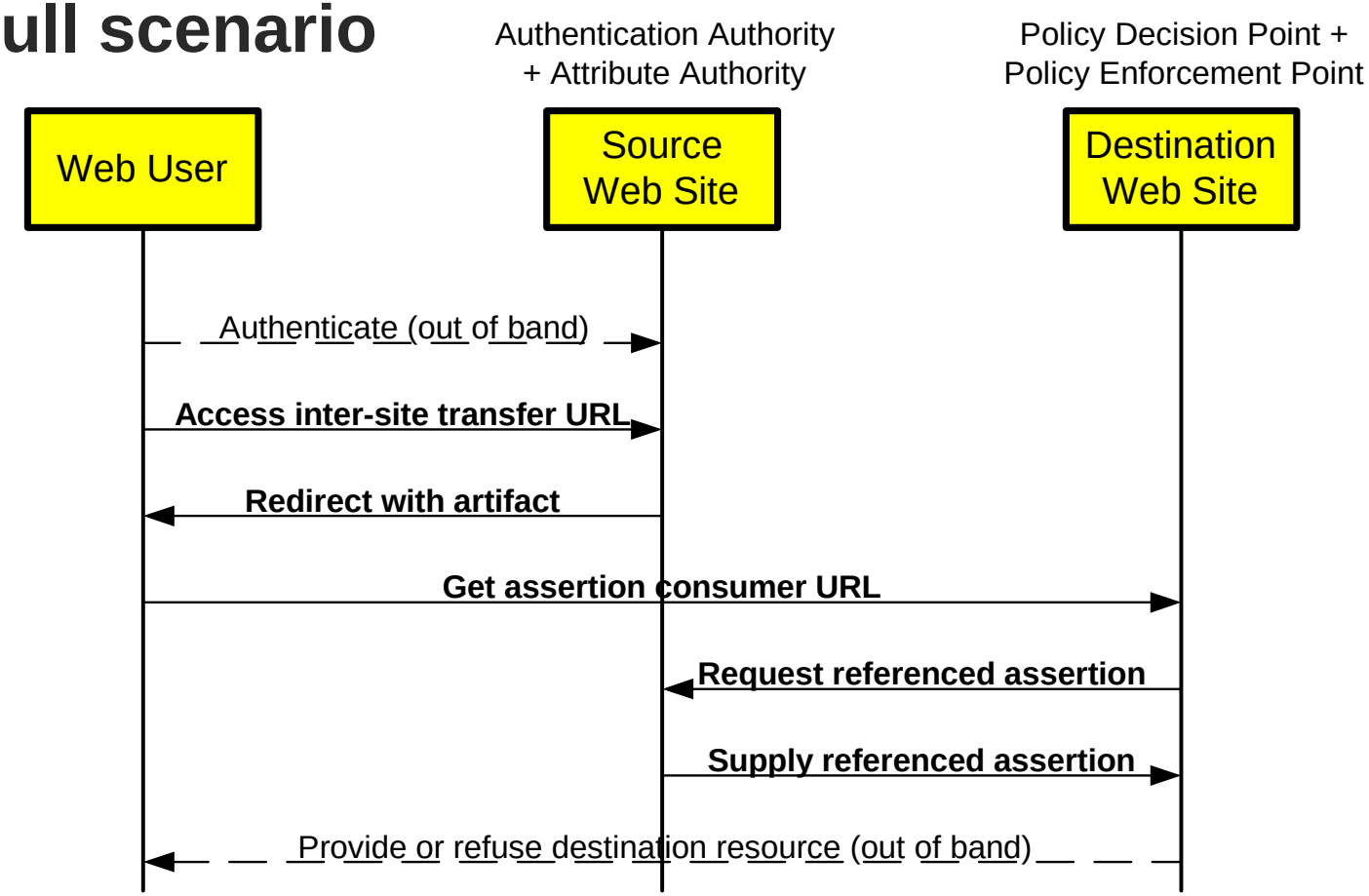
All assertions have some common information

- Issuer and issuance timestamp
- Assertion ID
- Subject
  - Name plus the security domain
  - Optional subject confirmation, e.g. public key
- "Conditions" under which assertion is valid
  - SAML clients *must reject* assertions containing unsupported conditions
  - Special kind of condition: assertion validity period
- Additional "advice"
  - E.g., to explain how the assertion was made

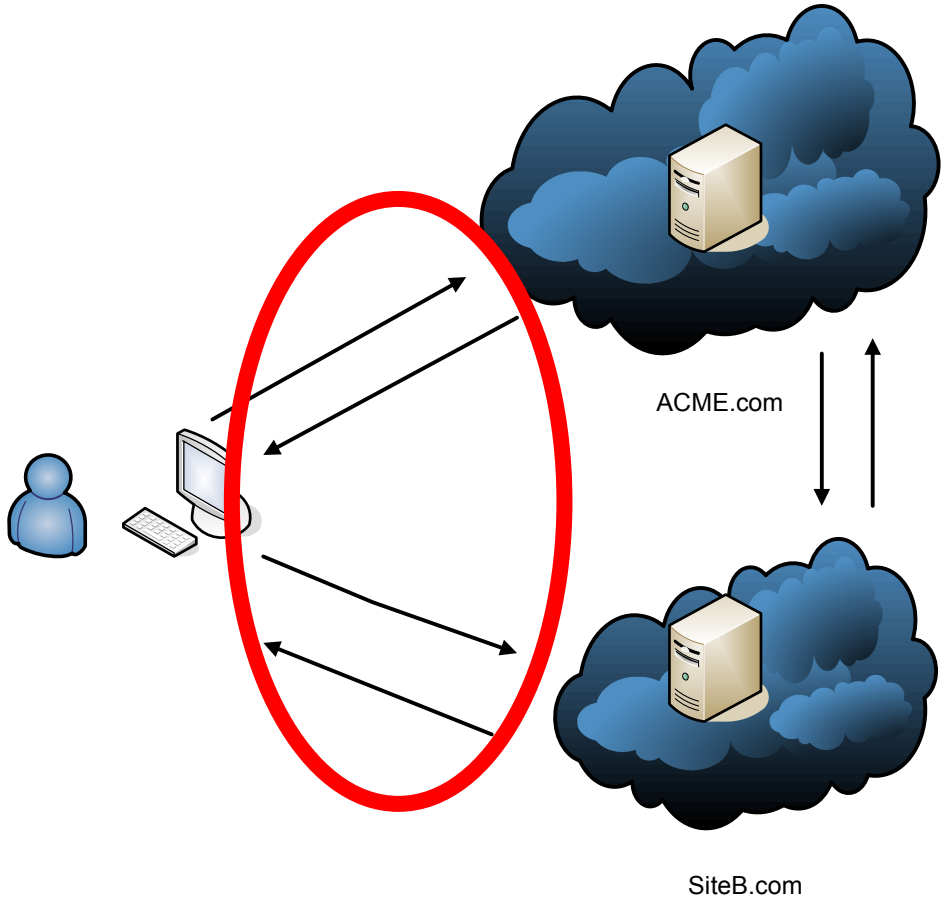
Authentication assertion

- An issuing authority asserts that:
  - subject S
  - was authenticated by means M
  - at time T
- **Caution:** Actually checking or revoking of credentials is not in scope for SAML!
  - Password exchange
  - Challenge-response
  - Etc.
- It merely lets you link back to acts of authentication that took place previously

SSO pull scenario



Our Scenario



Login

POST https://www.acme.com/app/loginSubmit.jspx HTTP/1.1

Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, */*

Referer: https://www.acme.com/app/login.jspx

Accept-Language: en-us

Content-Type: application/x-www-form-urlencoded

User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; .NET CLR 1.1.4322) Paros/3.2.10

Host: www.acme.com

Content-Length: 118

Connection: Keep-Alive

Cache-Control: no-cache

referer=&userName=ymchen&password=ymchen&x=16&y=9

Login Response (Set-Cookie)

HTTP/1.1 302 Moved Temporarily

Cache-Control: no-cache,no-store,max-age=0

Pragma: No-cache

Content-Type: text/html

Expires: Thu, 01 Jan 1970 00:00:00 GMT

Location: https://www.acme.com/app/welcome.jspx

Set-Cookie:
JSESSIONID=Gkfbl3YJ9MBdxzVLkRtPpXkYD6gMQkCQMCJVz3dYld7kPcdJG1LJ!239153226; path=/

Date: Sat, 15 Jul 2006 23:17:15 GMT

Connection: close

Get SAML Assertion from ACME.com for SiteB

GET https://www.acme.com/app/loginToSiteB.jspx HTTP/1.1

Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, */*

Cookie: CP=null*;
JSESSIONID=Gkfbl3YJ9MBdxzVLkRtPpXkYD6gMQkCQMCJVz3dYld7kPcdJG1LJ!239153226

User-Agent: Mozilla/4.0 (compatib MSIE 6.0; Windows NT 5.1; .NET CLR 1.1.4322) Paros/3.2.10

Host: www.acme.com

Connection: Keep-Alive

Accept-Language: en-us

Content-length: 0

Using ONLY JSESSIONID to get SAML Assertion

Response from ACME.com

```
<form name="samlform"
    action="https://www.siteb.com/actionb.dll?cmd=sson&pid=1234
    5" method="POST">
<input type="hidden" name="SAMLResponse"
    id="SAMLResponse" value="Base64 Encoded SAML
    Response">
</form>
```

SAML Response -- Header


```
<Response xmlns="urn:oasis:names:tc:SAML:1.0:protocol"
    xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
    xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    IssueInstant="2006-06-29T23:23:20.559Z" MajorVersion="1"
    MinorVersion="1"
    Recipient="https://www.siteb.com/actionb.dll?cmd=sson&amp;
    pid=12345" ResponseID="_c875208d11f9daa014770c0cf7812418">
```

SAML Response -- Digital Signature


```
<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
<ds:SignedInfo>
<ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-
      c14n#"></ds:CanonicalizationMethod>
<ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"></ds:SignatureMethod>
<ds:Reference URI="#_c875208d11f9daa014770c0cf7812418">
<ds:Transforms>
<ds:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"></ds:Transform>
<ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"><ec:InclusiveNamespaces
      xmlns:ec="http://www.w3.org/2001/10/xml-exc-c14n#" PrefixList="code ds kind rw saml samlp typens
      #default xsd xsi"></ec:InclusiveNamespaces></ds:Transform>
</ds:Transforms>
<ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"></ds:DigestMethod>
<ds:DigestValue>QNVCOOOsXzCDyl2mp6wZGhUBUCI=</ds:DigestValue>
</ds:Reference>
</ds:SignedInfo>
<ds:SignatureValue>
SgT0UDeIhUk2KYPk/N6TA2STerwDOTL/4paQ39odRhbngUwzfCizJwLCvZKHCqCwSY3btv9aj/kz
1i0180VCnpMtytVR0UWWM8kzRf1AuPEB3gm5gCZkX1zp/UOnWyEkpdSRNGSquFilrMt9q7JoE7Cq
QjR1uDqdBwPsOGlmkcw=
</ds:SignatureValue>
</ds:Signature>
```

SAML Response – Status

<Status><StatusCode Value="samlp:Success"></StatusCode></Status>

<Assertion xmlns="urn:oasis:names:tc:SAML:1.0:assertion"
AssertionID="_b3360dd260d9c4f7215554869a12044c"
IssueInstant="2006-06-29T23:23:20.559Z"
Issuer="http://www.acme.com" MajorVersion="1" MinorVersion="1">

SAML Response -- Condition

```
<AuthenticationStatement AuthenticationInstant="2006-06-
    29T23:23:20.559Z"
    AuthenticationMethod="urn:oasis:names:tc:SAML:1.0:am:password">
<Conditions NotBefore="2006-06-29T23:23:20.559Z"
    NotOnOrAfter="2006-06-29T23:28:20.559Z">
    <AudienceRestrictionCondition>
        <Audience>http://www.siteb.com</Audience>
    </AudienceRestrictionCondition>
</Conditions>
```

This SAML Assertion is only valid for 5 minutes!!!

SAML Response -- Subject

```
<Subject>
<NameIdentifier>123456789054321</NameIdentifier>
    <SubjectConfirmation>
    <ConfirmationMethod>urn:oasis:names:tc:SAML:1.0:cm:bearer</Confir
    mationMethod>
    </SubjectConfirmation>
</Subject>
<SubjectLocality IPAddress="10.50.45.23">
</SubjectLocality>
</AuthenticationStatement>
```

SAML Response -- Attributes

```
<AttributeStatement>
    <Subject>
        <NameIdentifier>123456789054321</NameIdentifier>
        <SubjectConfirmation>
            <ConfirmationMethod>urn:oasis:names:tc:SAML:1.0:cm:bearer
            </ConfirmationMethod>
        </SubjectConfirmation>
    </Subject>
<Attribute AttributeName="member_id"
    AttributeNamespace="urn:oasis:names:tc:SAML:1.0:assertion">
    <AttributeValue>123456789054321</AttributeValue>
</Attribute>
</AttributeStatement>
</Assertion>
</Response>
```

Posting SAML Response

POST https://www.siteb.com/actionb.dll?cmd=sson&pid=12345 HTTP/1.1

Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, */*

Referer: https://www.acme.com/app/loginToSiteB.jspx

Content-Type: application/x-www-form-urlencoded

Host: www.siteb.com

Connection: Keep-Alive

Cache-Control: no-cache

SAMLResponse=<Base64 encoded>

Response from SiteB

HTTP/1.1 200 Ok

Server: Microsoft-IIS/5.0

Date: Thu, 29 Jun 2006 23:23:58 GMT

P3P: policyref="/w3c/p3p.xml", CP="CAO DSP IND COR ADM CONo CUR CUSi DEV PSA PSD DELi OUR COM NAV PHY ONL PUR UNI"

Connection: close

Set-Cookie: RID=BLAHBLAH; path=/

Content-Type: text/html

Content-length: 12345

Implementation

- Read the XML File
- Parse all elements and attributes
- Put in attack patterns
- Results and problems

Read XML File

- Save the base 64 decoded file as an XML file
- Using System.XML to read the XML file like this:
  - XmlReader reader = XmlReader.Create(filename, settings);
  - Other ways like DOM or DataSet can be used too
- Determine NodeType (Element or Attribute)

Attack Patterns

- Only buffer overflow was tested.
- Patterns like ‘Z’ x 1024, ‘Z’ x 4096 or random data pattern
- After you generate the XML file,
  - Base 64 encode
  - Generate HTTP POST request
- File name convention
  - <element>-<attribute>-<test>.xml
  - E.g.: ds:Signature-value-50k.xml
- Coverages
  - 15 elements and their attributes
  - Hundreds of test cases

Issues

- How do we determine results automatically?
- By three conditions:
  - Comparing HTTP Response Code from the server
  - Comparing HTTP Response Content-Length header
  - Time out (in case the server died)
- Looking for anomolies (like an IDS)
  - Send normal request first
  - Send test case to compare results

Results

- We found one buffer overflow:
  - <ds:Signature>
  - The program did not handle the signature verification correctly, therefore when you feed a large amount of data, it crashed.
- Flawfinder found 29 potential problems on OpenSAML
  - Our test application was ‘based’ on OpenSAML implementation
  - We can’t test what we don’t see!

Future Works

- Need to add more attack
  - XPATH Injection
  - XML memory corruption test
  - Authorization test
    - If you have another user's account, can you become that user?
- Need to correlate with source code review results
  - Can you 'prove'/'disprove' flawfinder's result?
- Can similar tests been done in unit testing?
  - Even earlier, in TDD
- We have not touched the backend process part

Reference

- PROTOS -- http://www.ee.oulu.fi/research/ouspg/protos/
- SAML -- http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security
- OPENSAML – http://www.opensaml.org/

» Question & Answer

Thank You!

Yen-Ming Chen
ychen@foundstone.com