Federation & Empire

Emmanuel Bouillon
manu@veryopenid.net

DEF CON #19 - 7th August 2011
Prefatory notes

$ whoami

- Having fun in INFOSEC for a while
- SSTIC, PacSec, BlackHat EU, Hack.lu, #Days
- CVE-2010-\{0283,2229,2914,2941,...\}, CVE-2011-\{0001,...\}

Disclaimer
- This expresses my own views and does not involve my previous, current and future employers and thus for seven generations
- Presentation and code provided for educational purpose only
Prefatory notes

$ whoami

- Having fun in INFOSEC for a while
- SSTIC, PacSec, BlackHat EU, Hack.lu, #Days
- CVE-2010-\{0283,2229,2914,2941,...\}, CVE-2011-\{0001,...\}

Disclaimer

- This expresses my own views and does not involve my previous, current and future employers and thus for seven generations
- Presentation and code provided for educational purpose only
Outline

1. Prelude to Federation
   - Introduction

2. Forward the Federation
   - Where we come from

3. Federation
   - What you need to know

4. Federation and Empire
   - Sharpen your weapons

5. Federation’s Edge
   - Design assessment

6. Federation and (down to) Earth
   - Conclusion
What is it about?
SAML = Security Assertion Markup Language

This relates to
- SAML Token and Claims based IAM
- Low level, Pen-tester approach

Won’t discuss
- Formal protocol/API comparison
- Consistent standards study

Standards ”reverse engineering”: Find vulns, see what’s wrong in specs

Take-aways
- Tool to play with SAML protected Web app
- Proven assumption: Standards can be read as an attempt to circumvent SOP
- Important design security considerations
What is it about?
SAML = Security Assertion Markup Language

This relates to
- SAML Token and Claims based IAM
- Low level, Pen-tester approach

Won’t discuss
- Formal protocol/API comparison
- Consistent standards study

Standards ”reverse engineering”: Find vulns, see what’s wrong in specs

Take-aways
- Tool to play with SAML protected Web app
- Proven assumption: Standards can be read as an attempt to circumvent SOP
- Important design security considerations
Why should you care?

- Pervasive
- Cloud
- Joining a federation usually has severe contractual, legal implications.
- It’s coming your way!
The main problem to solve

- User and Administrator friendly cross organization boundaries
  SSO - here for web apps
  - Secure
  - Scalable
  - Manageable
  - Privacy / Anonymity

- Ideally compliant with the Laws of Identity [1]
Historical approaches
The good old time

- Account Replication
  - Manual
  - Automated
- WHAT?
  - Lose control of accounts
  - Have multiple passwords
- "Trust" relationships to be established with other realms / domains
  - All user information shared with federated partners
  - Firewalls need to be opened to allow trust
  - Bilateral $\Rightarrow n^2$ problem - no easy way to establish trust with multiple partners
- Privacy / anonymity
  - Anonymity Support for Kerberos [2]
Outline

1. Prelude to Federation
   - Introduction
2. Forward the Federation
   - Where we come from
3. Federation
   - What you need to know
4. Federation and Empire
   - Sharpen your weapons
5. Federation’s Edge
   - Design assessment
6. Federation and (down to) Earth
   - Conclusion
Federated identity standards - Overview

WS-Policy

WS-Security

WS-Actually Get Some Work Done

WS-Federation

WS-SecureConversation

WS-Trust

XML Encryption

SAML

Kerberos

X.509

Security Token Profiles

XML Digital Signatures

XML, SOAP, WSDL, Schema, WS-Addressing, etc.

HTTP

.Net TCP Channel, Fast InfoSet, etc.

E. Bouillon

Federation & Empire
SAML 101

- transfer of identity information
- between organizations
- that have an established trust relationship

SAML components
- SAML Assertions / Protocols / Bindings / Profiles
  - Web Browser SSO Profile
  - Identity Provider Discovery Profile
What are SAML Assertions?

- Signed XML document containing claims or attributes about a user
- Collected Claims = Identity
- Claims do not need to unambiguously identify user. Only relevant information (e.g. Age > 21, so can buy booze) [5]
What it looks like

Assertion ID=\"e3534d1e-a301-462c-ad72-46fe56c995c8\" IssueInstant=\"2010-11-23T12:14:18.382Z\"
xmlns=\"urn:oasis:names:tc:SAML:2.0:assertion\">
  <Issuer>..Token Issuer..</Issuer>
  <ds:SignedInfo>
    <ds:CanonicalizationMethod Algorithm=\"http://www.w3.org/2001/10/xml-exc-c14n\#\"/>
    <ds:SignatureMethod Algorithm=\"http://www.w3.org/2001/04/xmldsig-more#rsa-sha256\"/>
    <ds:Reference URI=\"#_e3534d1e-a301-462c-ad72-46fe56c995c8\"/>
    <ds:Transforms>
      <ds:Transform Algorithm=\"http://www.w3.org/2000/09/xmldsig#enveloped-signature\"/>
      <ds:Transform Algorithm=\"http://www.w3.org/2001/10/xml-exc-c14n\#\"/>
    </ds:Transforms>
    <ds:DigestMethod Algorithm=\"http://www.w3.org/2001/04/xmlexc#sha256\"/>
    <ds:DigestValue>C4uiwDjuFgPlRf9Eh8G6ssZsVByFp7rSf9Gd+butsd=</ds:DigestValue>
  </ds:Reference>
  <ds:SignatureValue>..Signature Value..</ds:SignatureValue>
  <KeyInfo xmlns=\"http://www.w3.org/2000/09/xmldsig\">
    <ds:X509Data>
    </ds:X509Data>
  </KeyInfo>
</Assertion>

E. Bouillon Federation & Empire
Prelude to Federation
Forward the Federation
What you need to know

What it looks like

```xml
<Subject>
  <SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:holder-of-key">
    <SubjectConfirmationData a:type="KeyInfoConfirmationDataType" xmlns:a="http://www.w3.org/">
      <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
        <e:EncryptedKey xmlns:e="http://www.w3.org/2001/04/xmлenc#">
          <e:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmлenc#rsa-oaep-mgf1p"> 
            <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"></DigestMethod>
          </e:EncryptionMethod>
        </e:EncryptedKey>
        <ds:X509Data xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
          <ds:X509IssuerSerial>
            <ds:X509IssuerName>..Cert Issuer..</ds:X509IssuerName>
            <ds:X509SerialNumber>..Cert Ref..</ds:X509SerialNumber>
          </ds:X509IssuerSerial>
        </ds:X509Data>
      </KeyInfo>
      <e:CipherData>
        <e:CipherValue>..Encrypted Key..</e:CipherValue>
      </e:CipherData>
    </SubjectConfirmationData>
  </SubjectConfirmation>
</Subject>
```

E. Bouillon
Federation & Empire
What it looks like
How is SAML used?

- Standards-based (so widely supported, supposedly interoperable), including:
  - XML Encryption, XML Digital Signatures, X.509
- Relies on standard HTTP (so passes through firewalls and across Internet)
  - Local network (not just for Federation!)
  - Branch offices
  - Remote workers
  - But also supports federation (of which more, later)
- Supports SSO (no need to remember lots of passwords)
- Transparent to user (from web browser or compiled application): a single click, and the magic happens!
How is SAML used?

- Standards-based (so widely supported, supposedly interoperable), including:
  - XML Encryption, XML Digital Signatures, X.509
- Relies on standard HTTP (so passes through firewalls and across Internet)
  - Local network (not just for Federation!)
  - Branch offices
  - Remote workers
  - But also supports federation (of which more, later)
- Supports SSO (no need to remember lots of passwords)
- Transparent to user (from web browser or compiled application): a single click, and the magic happens!
How is SAML used?

- Standards-based (so widely supported, supposedly interoperable), including:
  - XML Encryption, XML Digital Signatures, X.509
- Relies on standard HTTP (so passes through firewalls and across Internet)
  - Local network (not just for Federation!)
  - Branch offices
  - Remote workers
  - But also supports federation (of which more, later)
- Supports SSO (no need to remember lots of passwords)
- Transparent to user (from web browser or compiled application): a single click, and the magic happens!
How is SAML used?

- Standards-based (so widely supported, supposedly interoperable), including:
  - XML Encryption, XML Digital Signatures, X.509
- Relies on standard HTTP (so passes through firewalls and across Internet)
  - Local network (not just for Federation!)
  - Branch offices
  - Remote workers
  - But also supports federation (of which more, later)
- Supports SSO (no need to remember lots of passwords)
- Transparent to user (from web browser or compiled application): a single click, and the magic happens!
How is SAML used?

- Standards-based (so widely supported, supposedly interoperable), including:
  - XML Encryption, XML Digital Signatures, X.509
- Relies on standard HTTP (so passes through firewalls and across Internet)
  - Local network (not just for Federation!)
  - Branch offices
  - Remote workers
  - But also supports federation (of which more, later)
- Supports SSO (no need to remember lots of passwords)
- Transparent to user (from web browser or compiled application): a single click, and the magic happens!
How does it work?
Web Browser SSO Profile (SP-Initiated SSO - Redirect/POST Bindings)

1. User requests access to a claims aware web application
2. Redirected (through 302 Redirection) to IdP
3. Authenticates to IdP (either through Kerberos or Username/Password)
4. Redirected (through HTTP POST) back to web application, including security token
5. Happy User — no passwords to remember
   + Happy Administrator/Developer — much easier to manage
How does it work?

1. User requests access to a claims aware web application
How does it work?

1. Redirected (through HTTP 302) to IdP
How does it work?

3. Authenticates to IdP (either through Kerberos or Username/Password)
How does it work?

- Redirected (through HTTP POST) back to web application, including security token.
How does it work?

5 Happy User — no passwords to remember

+ Happy Administrator/Developer — much easier to manage
In addition to SSO, also supports:

- **Federation** — the sharing of identity between domains (MDSSO)
- **Delegation** — maintenance of identity to backend services
- **Distribution of Directory information to other applications**, which gives us:
  - **ABAC (Attribute Based Access Control)** = **RBAC**
- **Support for Federation** ⇒ **SAML suitable for the cloud**
  - Become ubiquitous
WS-Federation [6]

- Approved OASIS specification
- Defines mechanisms to allow different security realms to federate
  - authorized access to resources managed in one realm
  - can be provided to principals
  - whose identities and attributes are managed in other realms
- Includes mechanisms for brokering of identity, attribute, authentication and authorization assertions between realms
- Chapt 16: Security Considerations
  - Last bullet: compromised services
Federation

What you need to know

Target Domain

Trust

Source Domain

Web Service Provider

Web Application

Client / Browser

Key

- Message Flow
- HTTP Redirection (WS-Federation)
- Request for Security Token
- Request for Delegated Security Token
- Retrieve Directory Information

E. Bouillon

Federation & Empire
Brokered Federation model

- Trust through a central Broker, establishes trust between many IdPs
- But:
  - How is the trust established?
  - Do we trust all of them?
  - How are standards to be maintained?
Multiple Identity Providers

- User establishes account with many IdPs
- Each IdP for different function e.g.
  - Bank
  - Government

Diagram showing interactions between a user, identity providers, and applications.
SAML security seminal papers

On standards

- Security Analysis of the SAML Single Sign-on Browser/Artifact Profile [7]
- SSTC Response to Security Analysis of the SAML Single Sign-on Browser/Artifact Profile: [8]
- Security and Privacy Considerations for the OASIS Security Assertion Markup Language (SAML) V2.0 [9]

On implementations issues

OASIS SAML V2.0 Technical Overview (draft 3 and 10)

[12] Driver of SAML adoption No.1: "Multi Domain SSO ...
However, since browser cookies are never transmitted between DNS domains, ...
SAML solves the MDSSO problem."

True issue, legitimate will but...
Can also be read as: "SOP sucks, let’s build a workaround!"
- Great potential for security issues
- Is it a fail or not?
- E.g. Can a bad guy steal cookies?
  - Be patient ;-)
OASIS SAML V2.0 Technical Overview (draft 3 and 10)


[12] Driver of SAML adoption No.1: "Multi Domain SSO ... However, since browser cookies are never transmitted between DNS domains, ... SAML solves the MDSSO problem."

True issue, legitimate will but...

Can also be read as: "SOP sucks, let’s build a workaround!"

- Great potential for security issues
- Is it a fail or not?
- E.g. Can a bad guy steal cookies?
  - Be patient ;-)
Implementations security

The Good, e.g:
- Token encryption
- Replay attacks usually addressed by default

The Bad, e.g:
- Unsigned LogOut Request accepted
- TargetAudience attribute not verified

The Ugly, e.g:
- Open redirection vulnerability
- Cookie stealing
Implementations security

The Good, e.g:
- Token encryption
- Replay attacks usually addressed by default

The Bad, e.g:
- Unsigned LogOut Request accepted
- TargetAudience attribute not verified

The Ugly, e.g:
- Open redirection vulnerability
- Cookie stealing
Implementations security

The Good, e.g:
- Token encryption
- Replay attacks usually addressed by default

The Bad, e.g:
- Unsigned LogOut Request accepted
- TargetAudience attribute not verified

The Ugly, e.g:
- Open redirection vulnerability
- Cookie stealing
Outline

1. Prelude to Federation
   - Introduction

2. Forward the Federation
   - Where we come from

3. Federation
   - What you need to know

4. Federation and Empire
   - Sharpen your weapons

5. Federation’s Edge
   - Design assessment

6. Federation and (down to) Earth
   - Conclusion
Tools

Tool set usually made of a combination of
- Pro/Community edition of Commercial tools
- FOSS [13]
- Custom scripts

Methodology
- Procedures (+/-) formal (generic or custom)
- Generally accepted best practices [14][15]
- Habits, personal preferences [16]
- Still many manual, ad-hoc, improvised steps
Adapt your toolset

"Don’t be a tool" [20][21] but...

- Properly using the right tools often makes the difference
- Time constraint

Two reasons

- Allow "traditional" assessment of Web apps and services protected by SAML tokens
- Configurations of such architectures is crucial yet complex
  - error prone
  - need tools to assess good configuration settings are effective
Exiting SAML oriented helpers

- **UNINETT beta SAML tracer [17]**
  - Firefox Plugin
  - Tool for viewing SAML messages sent during single sign-on and single logout

- **Feide RnD SAML 2.0 Debugger [18]**
  - Online application to encode/decode SAML message

- **Federation Lab beta (GÉANT3 Identity Federations) [19]**
  - Online automated checks on SP implementation

- **Manual approach**
  - Burp decoder (truncated)
  - Python, ruby
    - `saml = Zlib::Inflate.new(-Zlib::MAX_WBITS).inflate(B...`  
    - `encoded = CGI::escape(Base64::encode64(Zlib::Deflate...`
Fed Lab Service Provider test

Against an out of the box ”Hello world” SP SimpleSAMLphp based

---

**Federation Lab**
Technical resources and tools for exploration of Identity Federation

- **Configure your Service Provider**
- **Connectivity Tests**
- **Running tests**

<table>
<thead>
<tr>
<th>Success (59)</th>
<th>Errors (14)</th>
<th>Warnings (2)</th>
<th>Notices (11)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Session fixation check</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Basic IdP-initiated Logout Test</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SP MUST NOT accept LogoutRequest when Issuer is wrong</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SP MUST NOT accept LogoutRequest when Destination is wrong</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Basic SP-initiated Logout Test</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SP SHOULD find attributes in a second Assertion/AttributeStatement, not only in one of them (test 1 of 2 - attributes in first).</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SP SHOULD find attributes in a second Assertion/AttributeStatement, not only in one of them (test 2 of 2 - attributes in last).</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SP SHOULD NOT accept attributes in unsigned 2nd assertion. (test 1 of 2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SP SHOULD NOT accept attributes in unsigned 2nd assertion. (test 2 of 2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SP SHOULD NOT accept authnstatement in unsigned 2nd assertion. (test 1 of 2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SP SHOULD NOT accept authnstatement in unsigned 2nd assertion. (test 2 of 2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SP SHOULD NOT accept an signed assertion, where the signature is referring to another assertion.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SP SHOULD NOT accept an signed assertion embedded in an AttributeValue inside an unsigned assertion.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SP SHOULD NOT accept an signed assertion embedded in an AttributeValue inside an unsigned assertion. (Signature moved out...)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SP SHOULD find attributes in signed Assertion/AttributeStatement, not only in one of them</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

E. Bouillon  
Federation & Empire
Fed Lab Service Provider test

Against an out of the box "Hello world" SP SimpleSAMLphp based

[Image of the status page of simpleSAMLphp with various error messages related to SP configuration and protocol validation]

- SP MUST accept an LogoutRequest with two session indexes (first valid) (sent in separate session, no session-cookies)
- SP MUST accept an LogoutRequest with two session indexes (second valid) (sent in separate session, no session-cookies)
- SP MUST NOT accept LogoutRequest when NameID content is wrong
- SP MUST NOT accept LogoutRequest when NameID@Format is wrong
- SP MUST NOT accept LogoutRequest when NameID@SPNameQualifier is wrong
- SP MUST NOT logout user when invalid SessionIndex is sent
- SP MUST NOT accept unsigned LogoutRequest
- SP MUST NOT accept a replayed Response. An identical Response/Assertion used a second time. [Profiles]: 4.1.4.3 POST-Specific Processing Rules (test 2 of 2): unsolicited response
- All endpoints in SP metadata SHOULD be HTTPS (not http) (saml2int)
- SP should not accept a Response with a SubjectConfirmationData elements with an incorrect @Address attribute
- SP should not accept a Response with a AuthnStatement missing
Decoding / encoding

[22] “Things humans aren’t good at”
- Decoding / encoding on the fly

Gain of automation
- Easy semantic understanding
- Allows relevant request mangling
- Changes scanner from dumb to smart fuzzer
- Thwarts anti-replay safeguards (e.g. unique random nonce)
- Updates timestamps (long scans can unfold)
Pre & Post processing

- Same approach as [23] for WCF Binary SOAP
- Proxy chaining
  - Preprocessing (decoding requests / encoding responses)
  - Scanning (Fuzz, mangle, do stuff…)
  - Postprocessing (encoding requests / decoding responses)
Illustration with Burp Pro Suite

- Burp Pro Suite [24] Extender
  - Java API to extend Burp Suite functionalities
  - Particularly suitable for Pre & Post processing
  - Bindings for Python and Ruby (Buby [25])
- Buby
  - Ruby based framework to extend Burp Suite
  - Tutorial: [26]
  - Hook either `evt_proxy_message` or `evt_http_message`
- POC
  - Buby modules and sample code at
    `http://code.google.com/p/buby-saml`
  - `buby -r SAML_preprocessing -e ReqTamperer`
  - `buby -r SAML_postprocessing -e ReqTamperer`
Sharpen your weapons

Preprocessing proxy - Original request

E. Bouillon
Federation & Empire
Central Burp instance - Intruder

E. Bouillon  Federation & Empire
Postprocessing proxy - Original request

E. Bouillon  Federation & Empire
## Prelude to Federation

- Forward the Federation
- Federation and Empire
- Federation's Edge
- Federation and (down to) Earth

---

**Sharpen your weapons**

---

**Postprocessing proxy - Edited request**

---

**E. Bouillon**

**Federation & Empire**
Example of vulnerabilities

- Open redirection [27]
  - $\approx$ http://www.vulnerable.com/?redirect=http://www.attacker.com
  - Not critical
  - Built in the standards?

- Cookie theft
  - Works even if the victim has not chosen the "Remember" option
  - Demo: Make the SP leaking idpdisco_saml_lastidp cookie, even if cookie idpdisco_saml_remember = 0
    - If you visit his site, a bad guy can inconspicuously discover your IdP = what is your originating organization
Demo: SimpleSAMLPHP open redirect
When an open redirect leads to cookie theft

- Leveraging an existing live, open to everyone test environment
- Feide [28]: Norwegian academic Federation
- on a dummy account

```plaintext
home_realm_cookie=
https://openidp.feide.no
```
Demo: SimpleSAMLPHP open redirect
When an open redirect leads to cookie theft

1. Victim accesses evil site
2. Contains a crafted get request to the SP
3. Evil site gets the cookie back thanks to the Open Redirection (Google search request for illustration)
Back to the OASIS standard

Identity Provider Discovery Service Protocol and Profile [29]

1. Request Redirected to Discovery Service

2. Discovery service identifies appropriate identity provider(s)

3. Request Redirected back to Service Provider
Identity Provider Discovery Service Protocol and Profile [29]

[sic]

- "This protocol has the potential for creating additional opportunities for phishing…"
- Proposed workaround: use of SP metadata
- "To mitigate this threat, metadata can be used to limit the sites authorized to use a discovery service"
- "A discovery service SHOULD require that the service providers making use of it supply metadata"

- Developers don’t have to implement it to be compliant [30]
Identity Provider Discovery Service Protocol and Profile [29]

[sic]

- "This protocol has the potential for creating additional opportunities for phishing...”
- Proposed workaround: use of SP metadata
- "To mitigate this threat, metadata can be used to limit the sites authorized to use a discovery service”
- "A discovery service SHOULD require that the service providers making use of it supply metadata”

- Developers don’t have to implement it to be compliant [30]
Outline

1. Prelude to Federation
   - Introduction

2. Forward the Federation
   - Where we come from

3. Federation
   - What you need to know

4. Federation and Empire
   - Sharpen your weapons

5. Federation’s Edge
   - Design assessment

6. Federation and (down to) Earth
   - Conclusion
Deployment and trust topologies

Typical situations
New risks?

Previous boundaries become more and more notional

- Network flows
  - Attack surface
  - Management interface
- Users community
  - Insider?
- Data flows

Cost/Benefit not doing it?

- Security policies coherency / comparison / enforcement
Similar flows orchestrated in federated environment
simple federation scenario [6]
Similar flows orchestrated in federated environment
What if OrgC signs a claim for UserA@OrgA.net?
Considerations on deployment architectures

Trust topology

- Previous example follows a direct trust topology [6]
Considerations on deployment architectures

Trust topology

- More complex exist including indirect trust topology [6]
Considerations on deployment architectures

Trust topology

More complex exist including indirect trust topology.
What if OrgC signs a claim for UserA@OrgA.net?
SAML claims laundering
SAML claims laundering

- If STS D signs the token, STS B has no way to see the trick
  - OrgB fully relies on OrgD to properly check SAML claims
  - Policy?
  - Verification?
    - Is auditing permitted? Regular security checks presented?
    - How to prove other parties compliance with relevant requirements?
- Questions usually unasked and even less answered:
  - What about a malicious/compromised IdP in the federation?
    - Can a malicious IdP impersonate another domain users?
    - Are there safeguards in place?
    - Do I own or delegate these safeguards?
  - What about a malicious/compromised SP in the federation?
SAML claims laundering

- If STS D signs the token, STS B has no way to see the trick
  - OrgB fully relies on OrgD to properly check SAML claims
  - Policy?
  - Verification?
    - Is auditing permitted? Regular security checks presented?
    - How to prove other parties compliance with relevant requirements?

- Questions usually unasked and even less answered:
  - What about a malicious-compromised IdP in the federation?
    - Can a malicious IdP impersonate another domain users?
    - Are there safeguards in place?
    - Do I own or delegate these safeguards?
  - What about a malicious-compromised SP in the federation?
SAML claims laundering

WS-Federation [6]

- Situation tersely considered in the specification
- Chapt 16: Security considerations
- **Compromised services**: ”This is of special concern in scenarios like the 3rd party brokered trust where a 3rd party IP/STS is brokering trust between two realms.”

- In practice, by default it works
- No proposed solution on how to prevent that
- In the case of 3rd party brokered trust, how to control the loss of control
  - Whose liability
  - Other parties obligation (accountability)
SAML claims laundering

- All relies on checks made at each relay level
- This MUST be done since default settings are permissive
- Key attributes must be kept or added to avoid turning the situation into blind trust and single point of security failure
- On main federations, this policy is not publicly disclosed, so how to make an educated choice?
Outline

1 Prelude to Federation
   • Introduction
2 Forward the Federation
   • Where we come from
3 Federation
   • What you need to know
4 Federation and Empire
   • Sharpen your weapons
5 Federation’s Edge
   • Design assessment
6 Federation and (down to) Earth
   • Conclusion
Conclusion

Take-aways

- Knowledge and tool to keep on powning SAML protected Web app
- Proven assumption: Standards can be read as an attempt to circumvent SOP
  - Process and tools to get there
- Important design security considerations
  - Without taking care, ”Insecurity by design” is more than likely
  - E.g. Cross domain SSO with AD trust relationships
    - A compromised domain cannot impersonate other domains users
  - With SAML based cross domain SSO, by default, it will
Conclusion

- This applies to other forms of federation
- Developers, marketers ahead of security guys
  - Yet default settings are not secure
  - The "make it working" approach might lead to insecure deployment
  - Need to catch up to avoid big deployment security failure (with probably thorny legal issues)
- Get acquainted with protocols to properly assess designs and deployments
- Adapt our tool set because bad guys will
- Better guidance or improved standards?
Thanks for your attention

- Acknowledgment
  - Isaac Asimov
  - Rui Fiske for his great help and extensive knowledge on SAML
- Q & possibly A
- Buby modules and sample code at http://code.google.com/p/buby-saml

manu@veryopenid.net
References I


[7] T. Groß - IBM Zurich Research Laboratory - Security Analysis of the SAML Single Sign-on Browser/Artifact Profile

References II


References III

[15] OWASP Testing Project -


[17] UNINETT releases public beta of SAML tracer -

[18] Feide RnD SAML 2.0 Debugger -
 https://rnd.feide.no/software/saml_2_0_debugger/


[22] J. Haddix, J. Parish - ToorCon 12 -
References IV


