# Federation & Empire

Emmanuel Bouillon
manu@veryopenid.net

DEF CON #19 - $7^{th}$ 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

# 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!

# 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

# 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

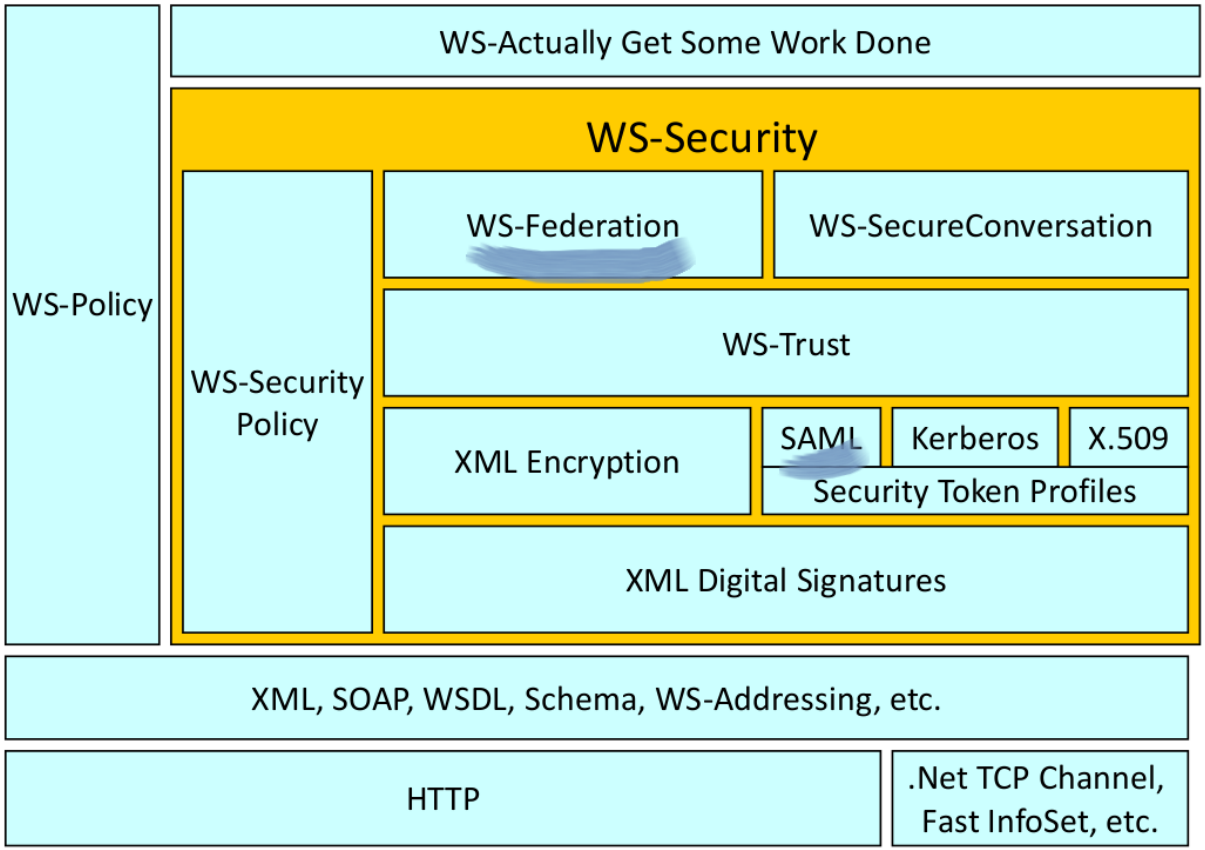
3 Federation
- What you need to know

4 Federation and Empire
- Sharpen your weapons

5 Federation's Edge
- Design assessment

6 Federation and (down to) Earth
- Conclusion

# Federated identity standards - Overview




[3]

# SAML 101

## Security Assertion Markup Language [4]

- 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: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:Can
      <ds:SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"></ds:
      <ds:Reference URI="#_e3534d1e-a301-462c-ad72-46fe56c995c8">
        <ds:Transforms>
          <ds:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"></d
          <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"></ds:Transform>
        </ds:Transforms>
        <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"></ds:DigestMetho
        <ds:DigestValue>C4uizWDjuFgPlRf9Eh8G6ssZsVByFp7rSf9Gd+butds=</ds:DigestValue>
      </ds:Reference>
    </ds:SignedInfo>
    <ds:SignatureValue>..Signature Value..</ds:SignatureValue>
    <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
      <ds:X509Data>
        <ds:X509Certificate>..Base64 Encoded Issuer Certificate..</ds:X509Certificate>
      </ds:X509Data>
    </KeyInfo>
  </ds:Signature>
  <Subject>
    <SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:holder-of-key">
```

# What it looks like

```
<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/xmlenc#">
          <e:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p">
            <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"></DigestMethod>
          </e:EncryptionMethod>
          <KeyInfo>
            <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>
        </e:EncryptedKey>
      </KeyInfo>
    </SubjectConfirmationData>
  </SubjectConfirmation>
</Subject>
<Conditions NotBefore="2010-11-23T12:14:18.368Z" NotOnOrAfter="2010-11-23T13:14:18.368Z">
```

# What it looks like

```
</Subject>
<Conditions NotBefore="2010-11-23T12:14:18.368Z" NotOnOrAfter="2010-11-23T13:14:18.368Z">
  <AudienceRestriction>
    <Audience>..Relying Party URI..</Audience>
  </AudienceRestriction>
</Conditions>
<AttributeStatement>
  <Attribute Name="http://schemas.xmlsoap.org/claims/UPN">
    <AttributeValue>
      ..Value from Directory..
    </AttributeValue>
  </Attribute>
  <Attribute Name="http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name">
    <AttributeValue>
      ..Value from Directory..
    </AttributeValue>
    <AttributeValue>
      ..Value from Directory..
    </AttributeValue>
  </Attribute>
  <Attribute Name="http://schemas.xmlsoap.org/claims/EmailAddress">
    <AttributeValue>
      ..Value from Directory..
    </AttributeValue>
  </Attribute>
</AttributeStatement>
<AuthnStatement AuthnInstant="2010-11-23T12:14:18.315Z">
  <AuthnContext>
    <AuthnContextClassRef>urn:federation:authentication:windows</AuthnContextClassRef>
  </AuthnContext>
</AuthnStatement>
</Assertion>
```

# 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!

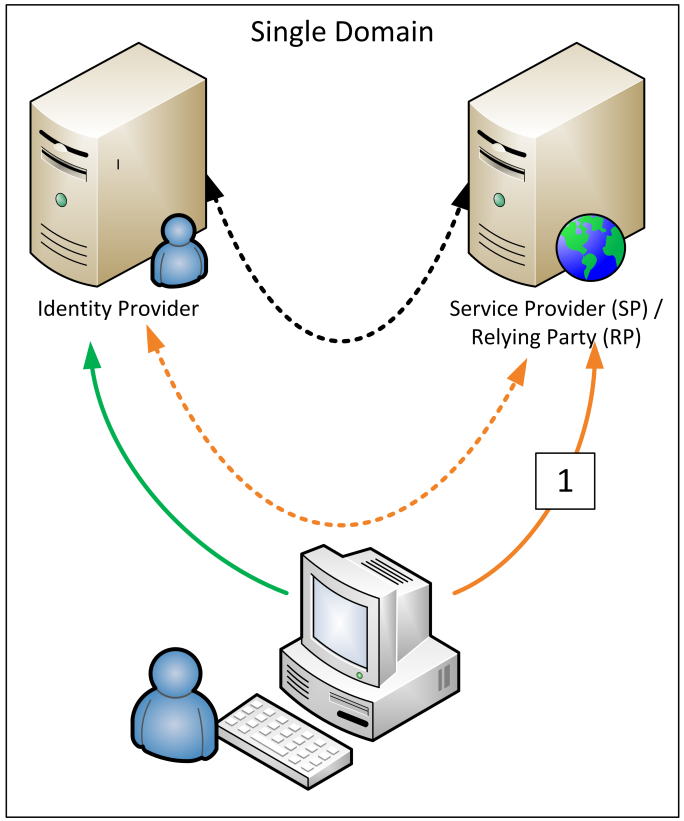
# How does it work?

## 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

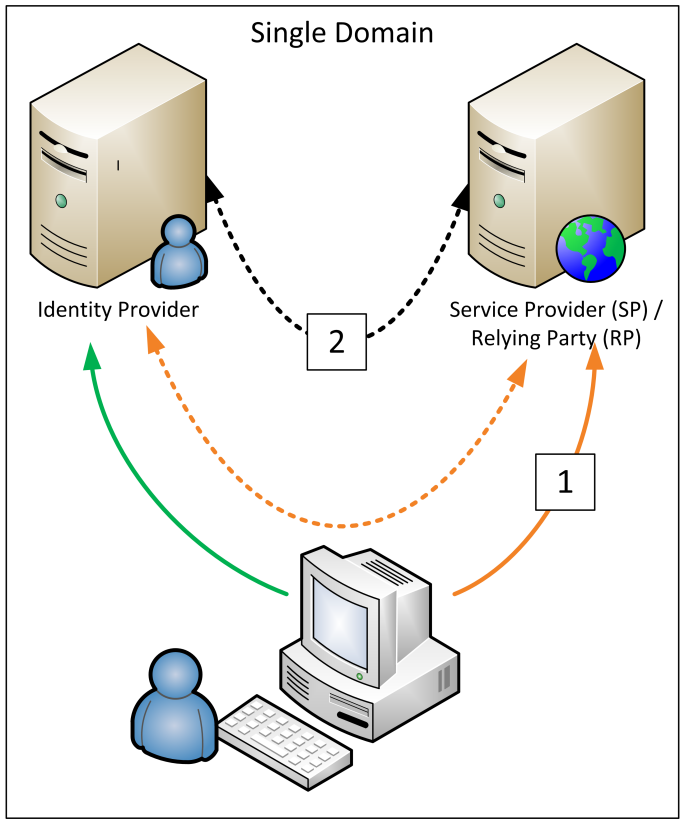
# How does it work?

1. User requests access to a claims aware web application



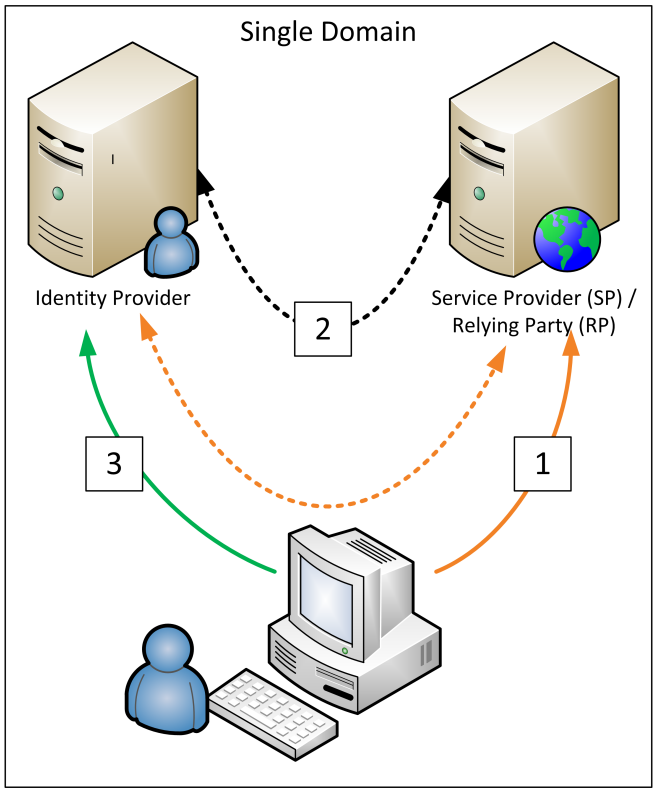
# How does it work?

2. Redirected (through HTTP 302) to IdP



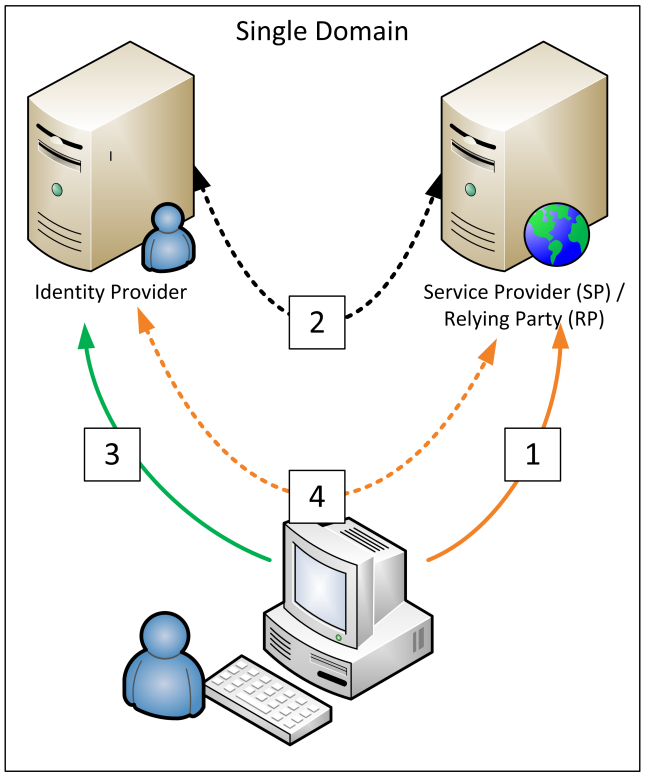
# How does it work?

3. Authenticates to IdP (either through Kerberos or Username/Password)



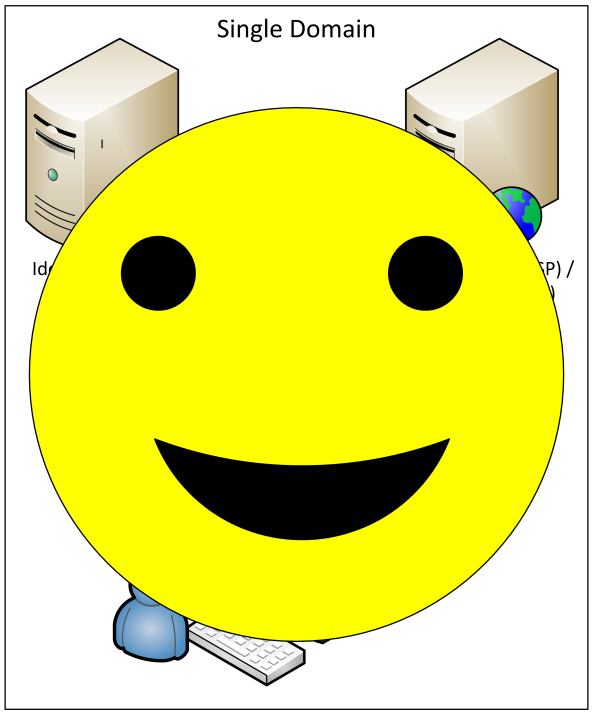
# How does it work?

4. 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



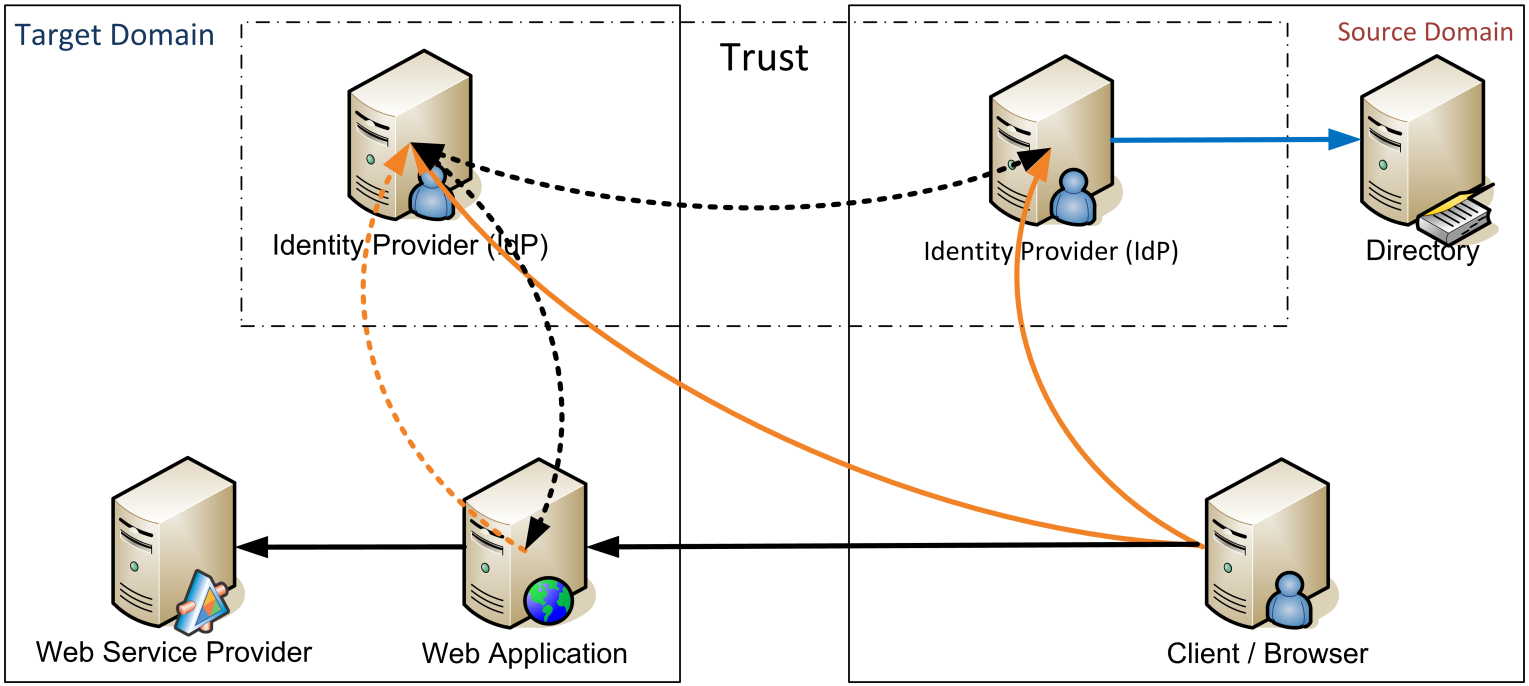
# So what?

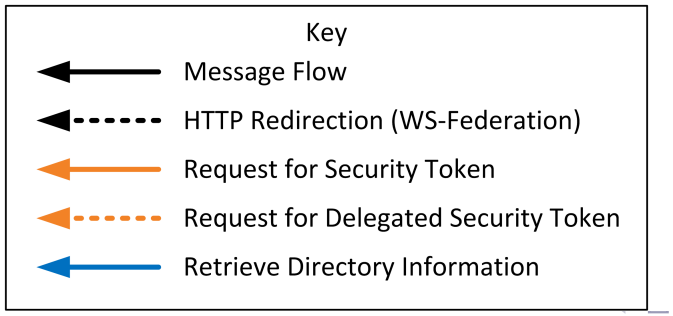
- In addition to SSO, also supports:
  - Federation – the sharing of identity between domains (MDSSO)
  - Delegation – maintenance of identity to backend services
  - Distribution of Directory information to other applications, which gives us:
    - ABAC (Attribute Based Access Control) = $\text{RBAC}^{+}$
- Support for Federation $\Rightarrow$ 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

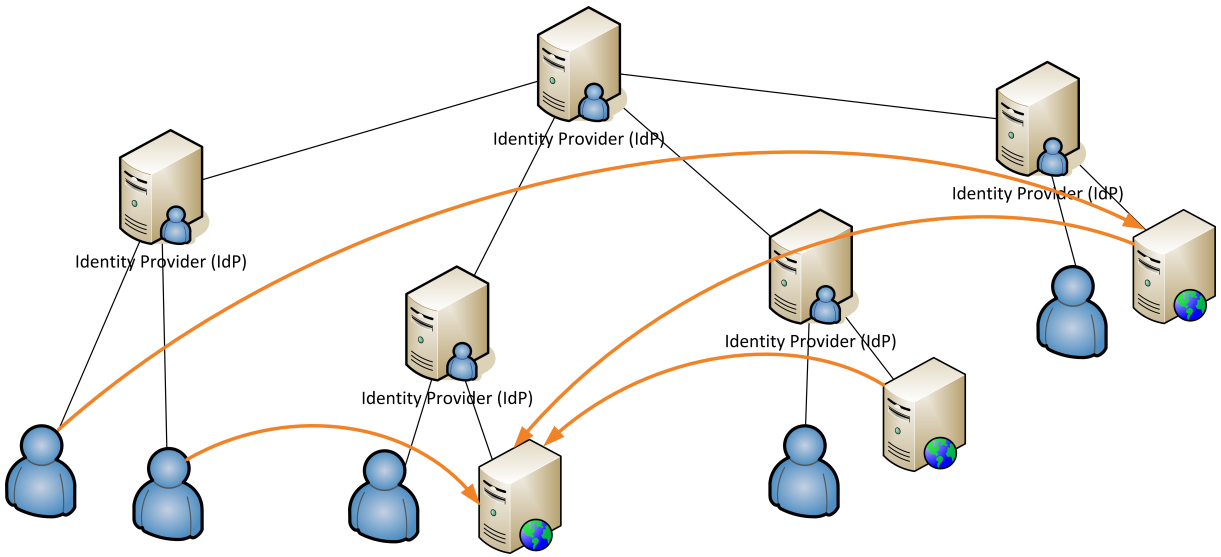
# Federation


Target Domain
Trust
Source Domain
Identity Provider (IdP)
Identity Provider (IdP)
Directory
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

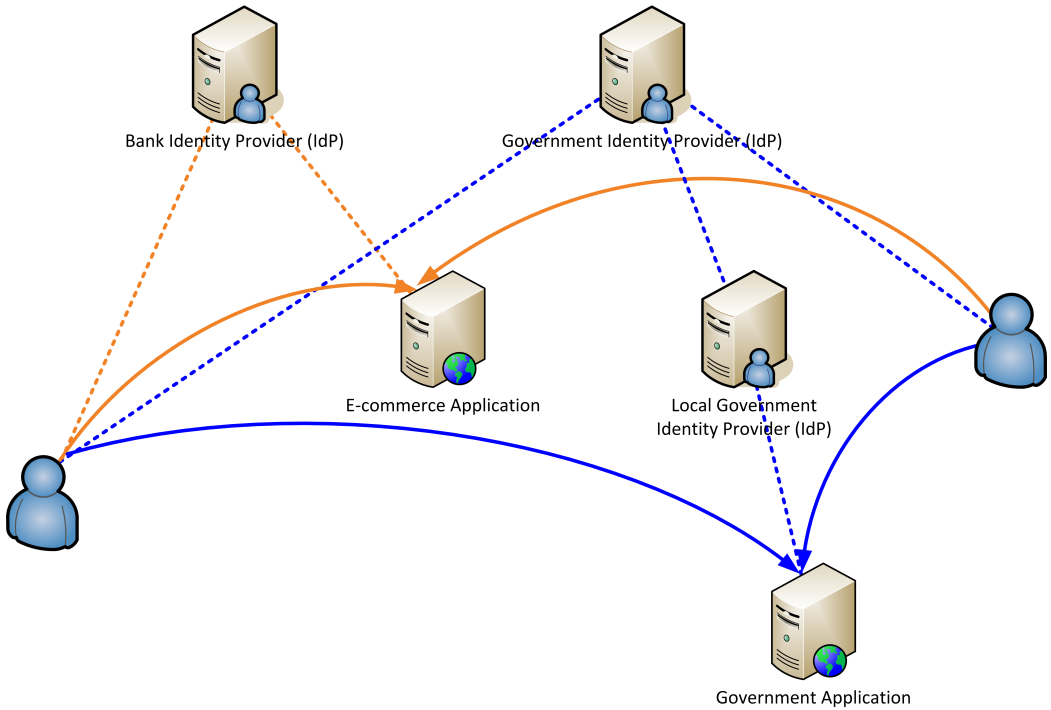
# Brokered Federation model

- 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



# 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

- Armando & Al - Breaking the SAML-based Single Sign-On for Google Apps [10]
- B. Hill - Attacking XML Security - Black Hat US 2007 [3]

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

## [sic]

- [11] SAML use case No.1: *"Limitations of Browser cookies"*
- [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)

## [sic]

- [11] SAML use case No.1: *"Limitations of Browser cookies"*
- [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

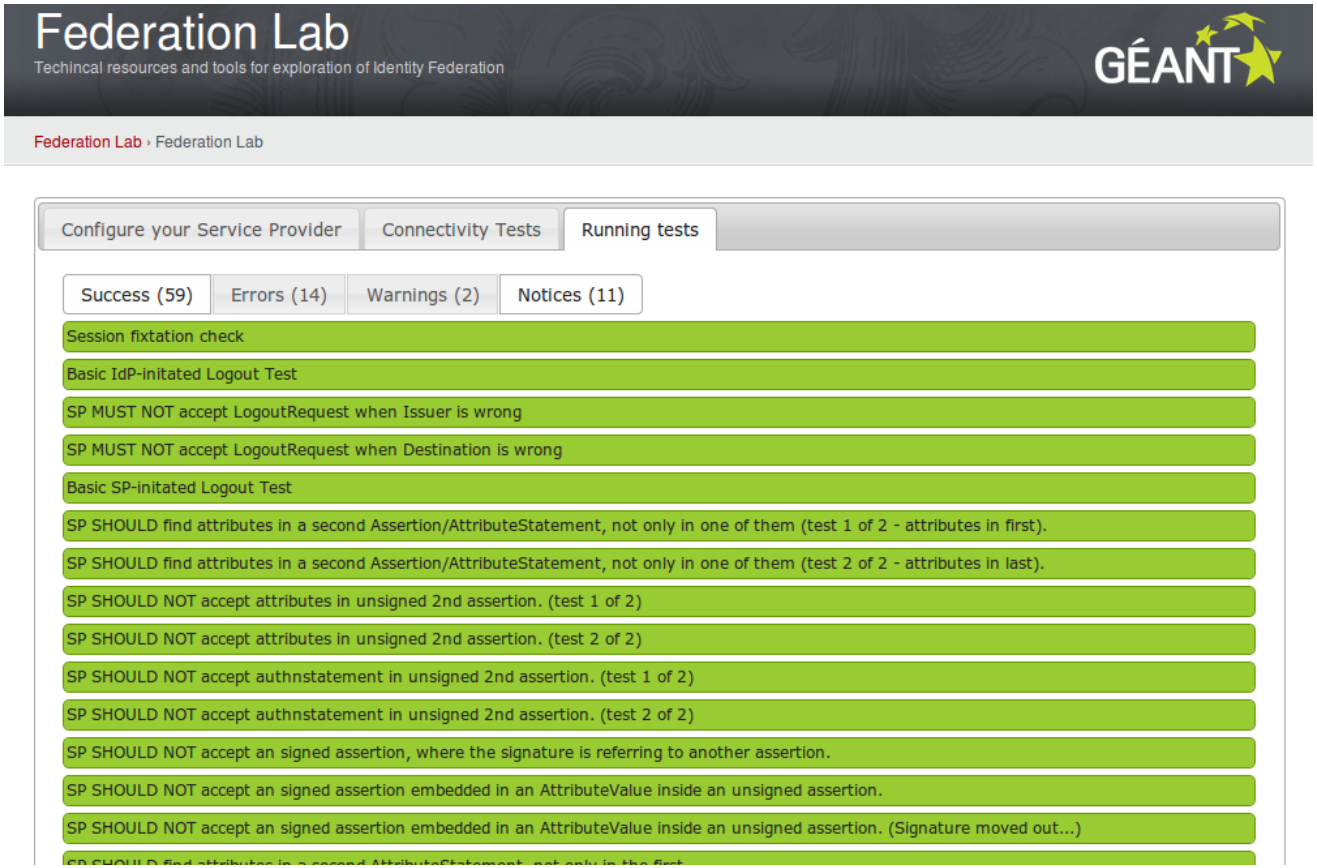
- UNINETT beta SAML tracer [17]
  - Firefox Plugin
  - Tool for viewing SAML messages sent during single sign-on and single logout




- 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...`

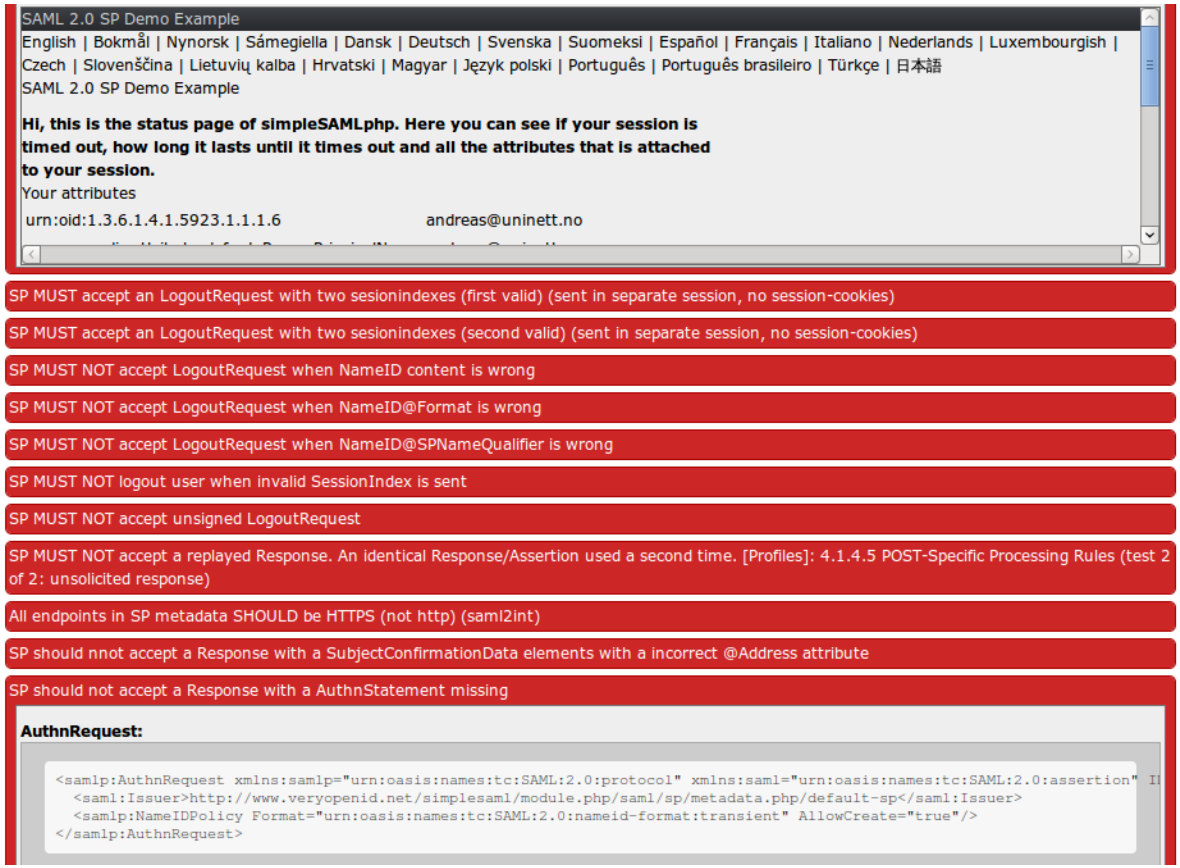
# Fed Lab Service Provider test

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



# Fed Lab Service Provider test

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



# Decoding / encoding

## [22] "Things humans aren't good at"

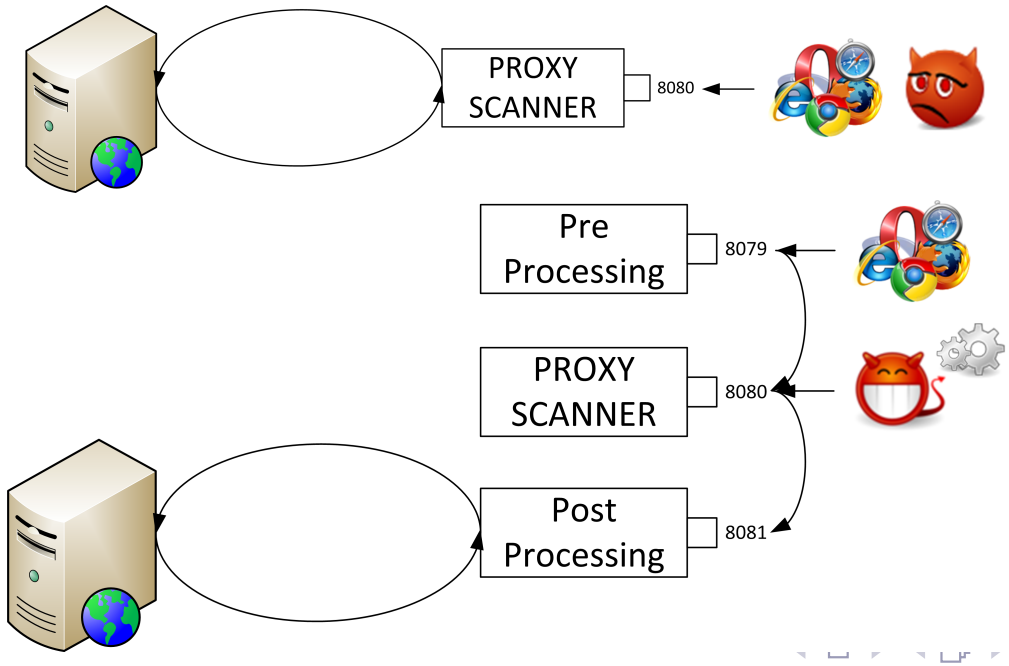
- Decoding / encoding on the fly

## 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

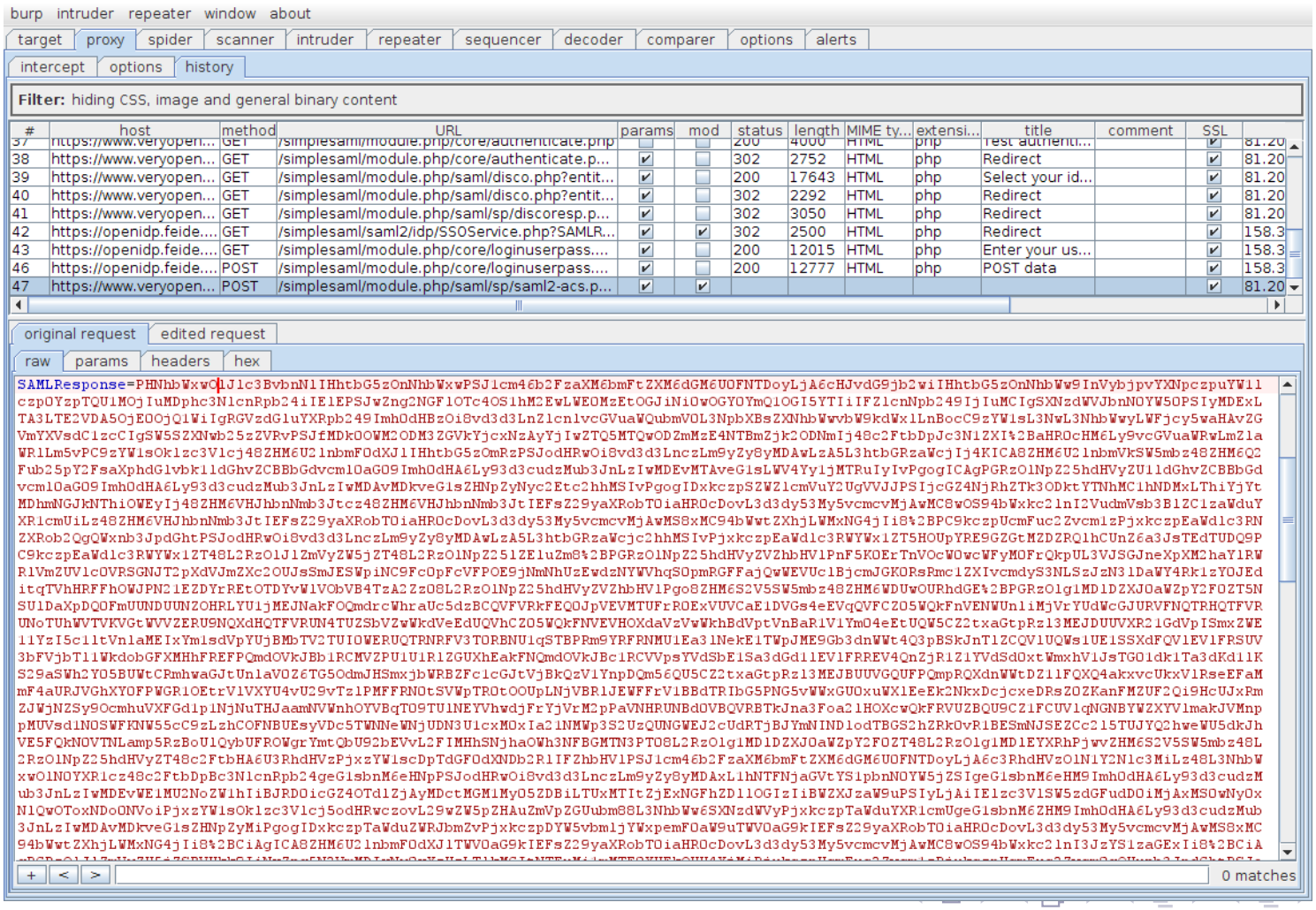
- Same approach as [23] for WCF Binary SOAP
- Proxy chaining
  - Preprocessing (decoding requests / encoding responses)
  - Scanning (Fuzz, mangle, do stuff...)
  - Postprocessing (encoding requests / decoding responses)



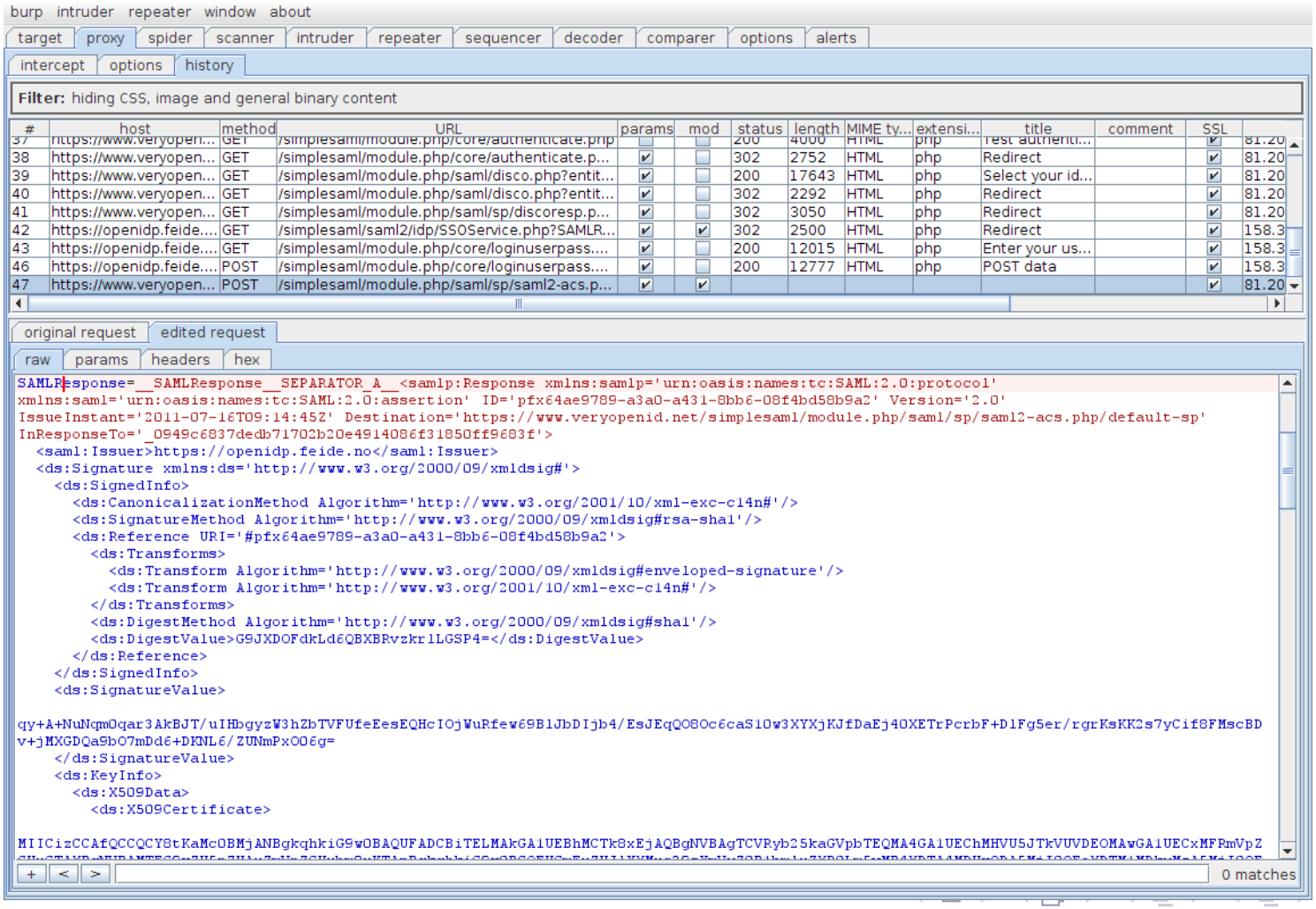
# Illustration with Burp Pro Suite

- 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`

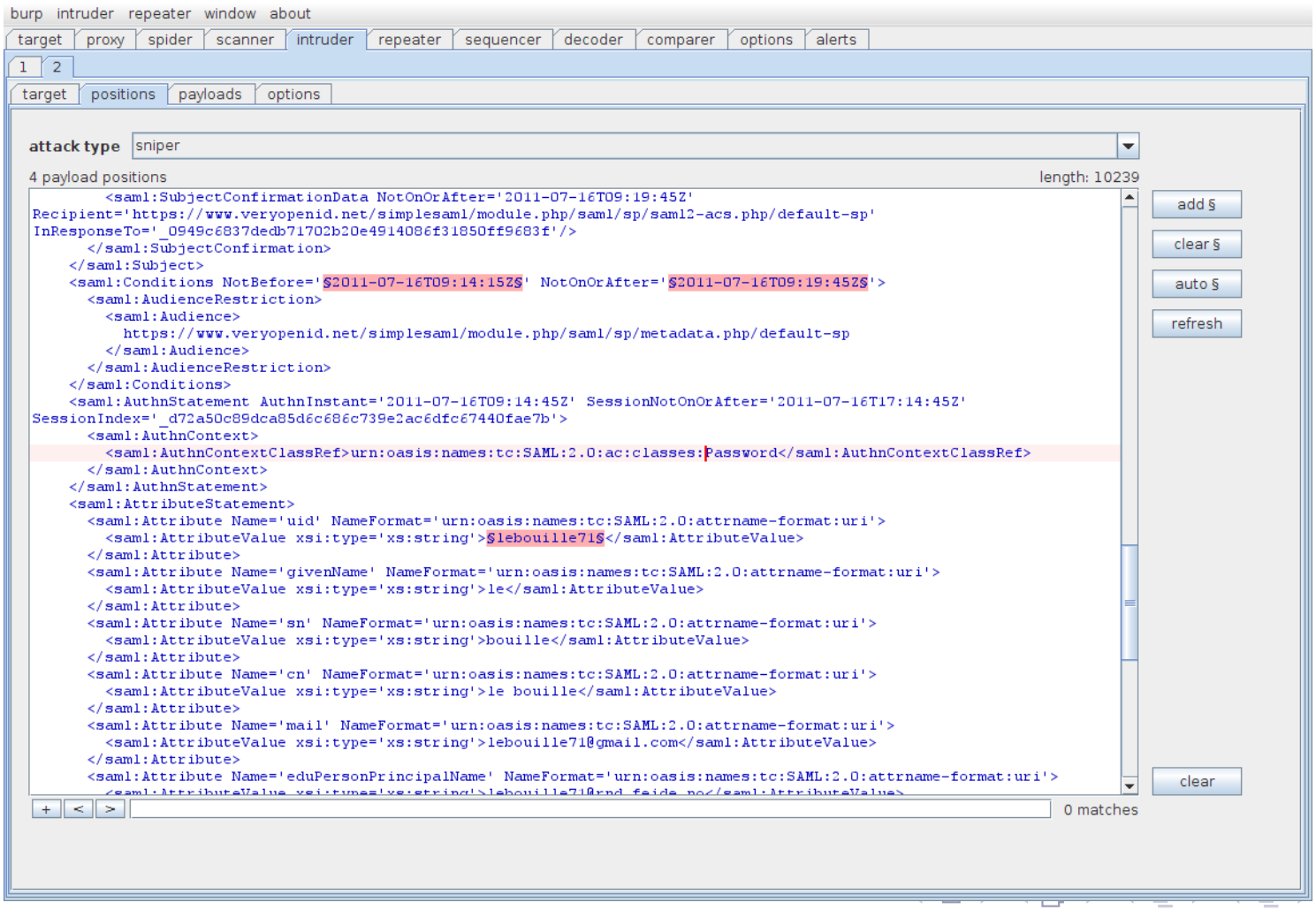
# Preprocessing proxy - Original request



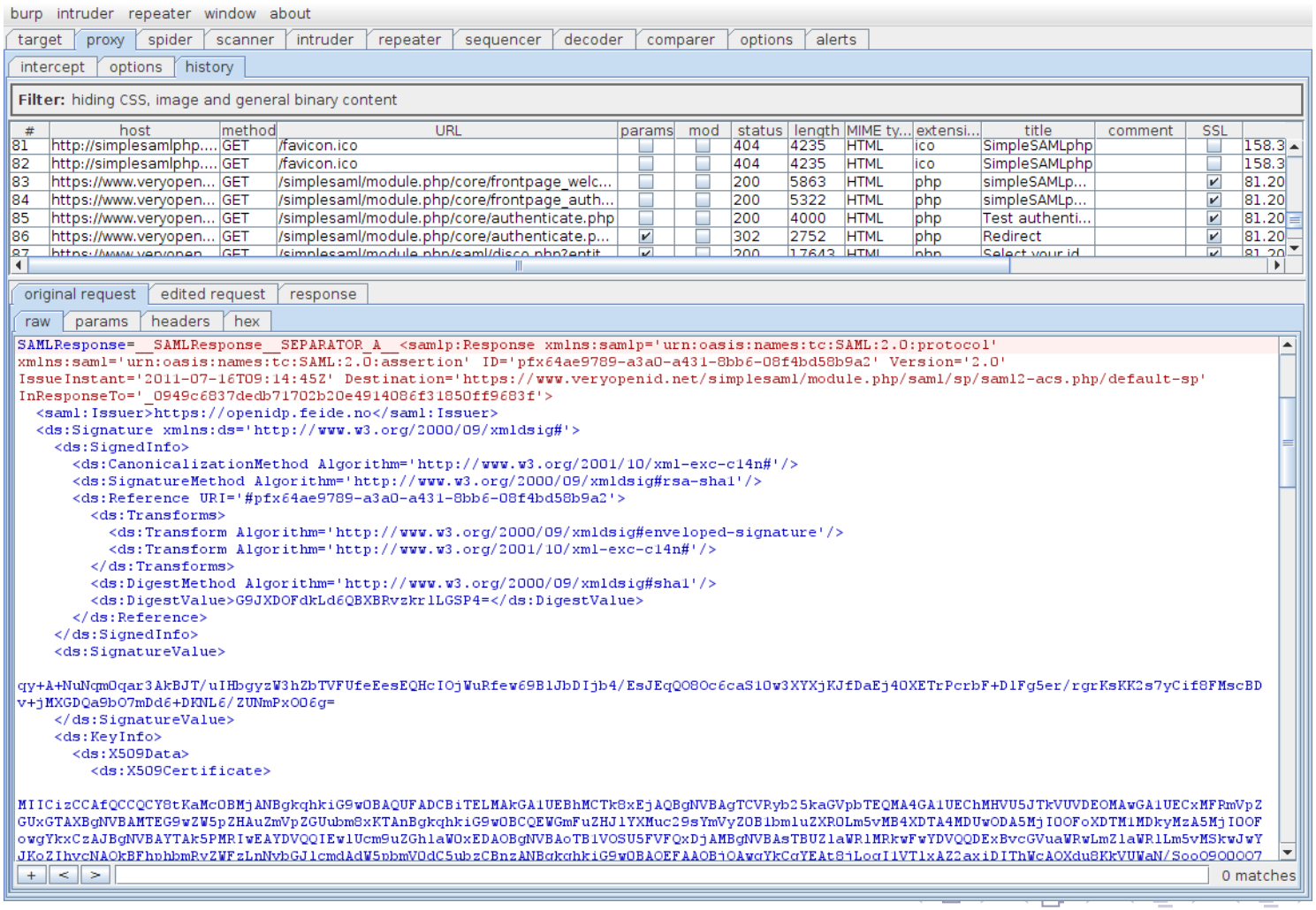
# Preprocessing proxy - Edited request



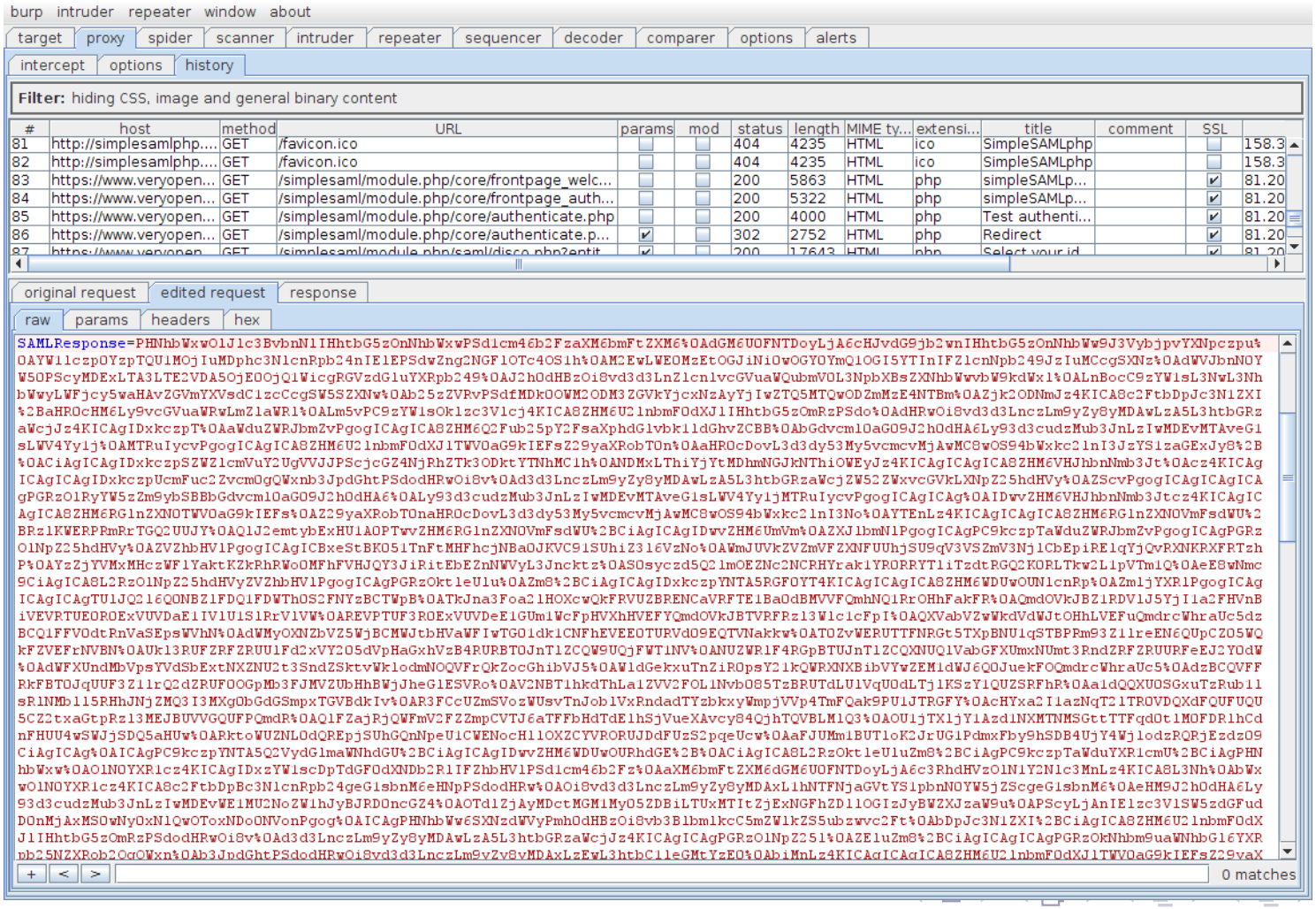
# Central Burp instance - Intruder



# Postprocessing proxy - Original request



# Postprocessing proxy - Edited request



# Example of vulnerabilities

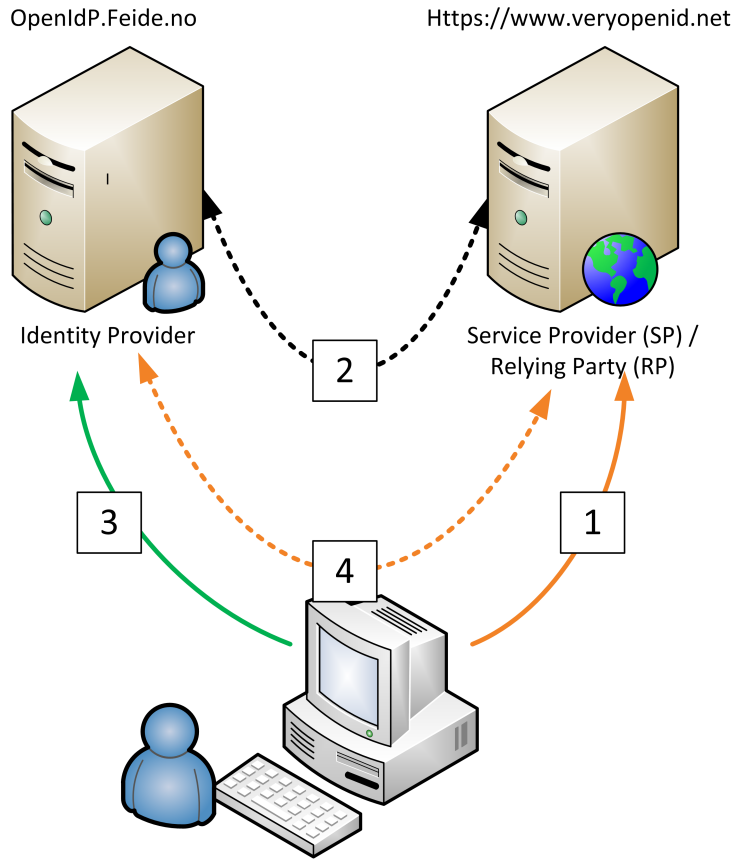
- Open redirection [27]
  - $\simeq$ `http://www.vulnerable.com/?redirect=http://www.attacker.com`
  - Not critical
  - Built in the standards?
- Cookie theft
  - Works even if the victim has not chosen the "Remember" option
  - Demo: Make the SP leaking $idpdisco_saml_lastidp$ cookie, even if cookie $idpdisco_saml_remember = 0$
    - 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

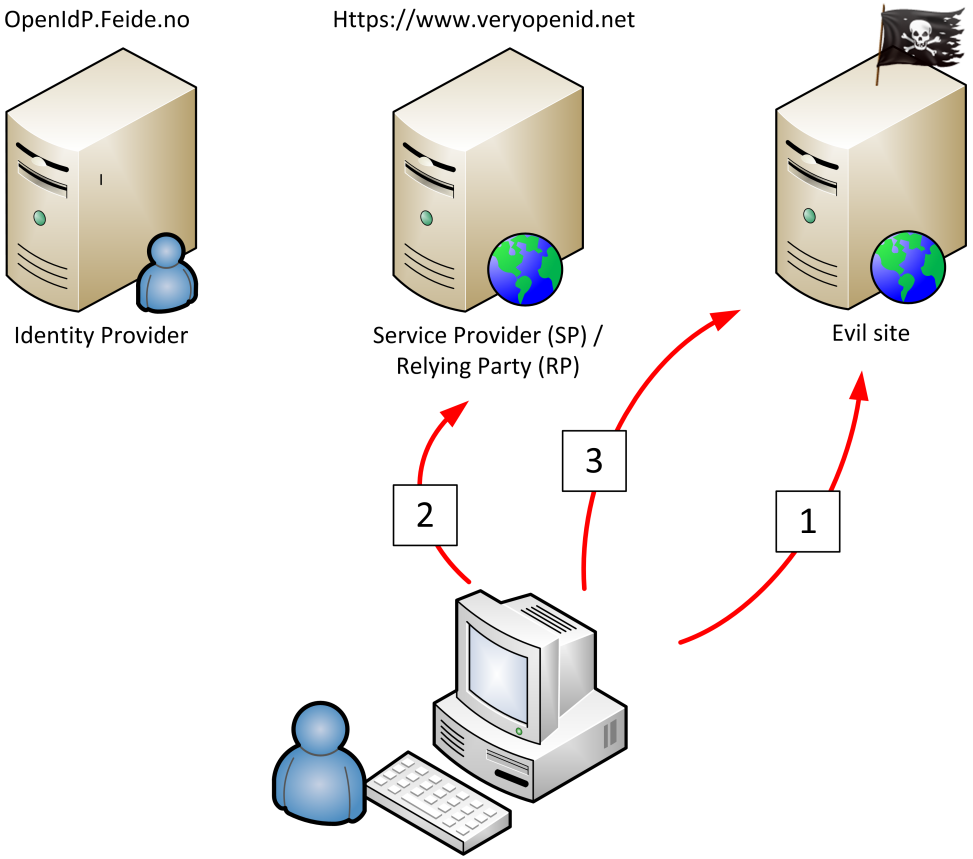
- Leveraging an existing live, open to everyone test environment
- Feide [28]: Norwegian academic Federation
- on a dummy account

home_realm_cookie=
`https://openidp.feide.no`



# Demo: SimpleSAMLPHP open redirect

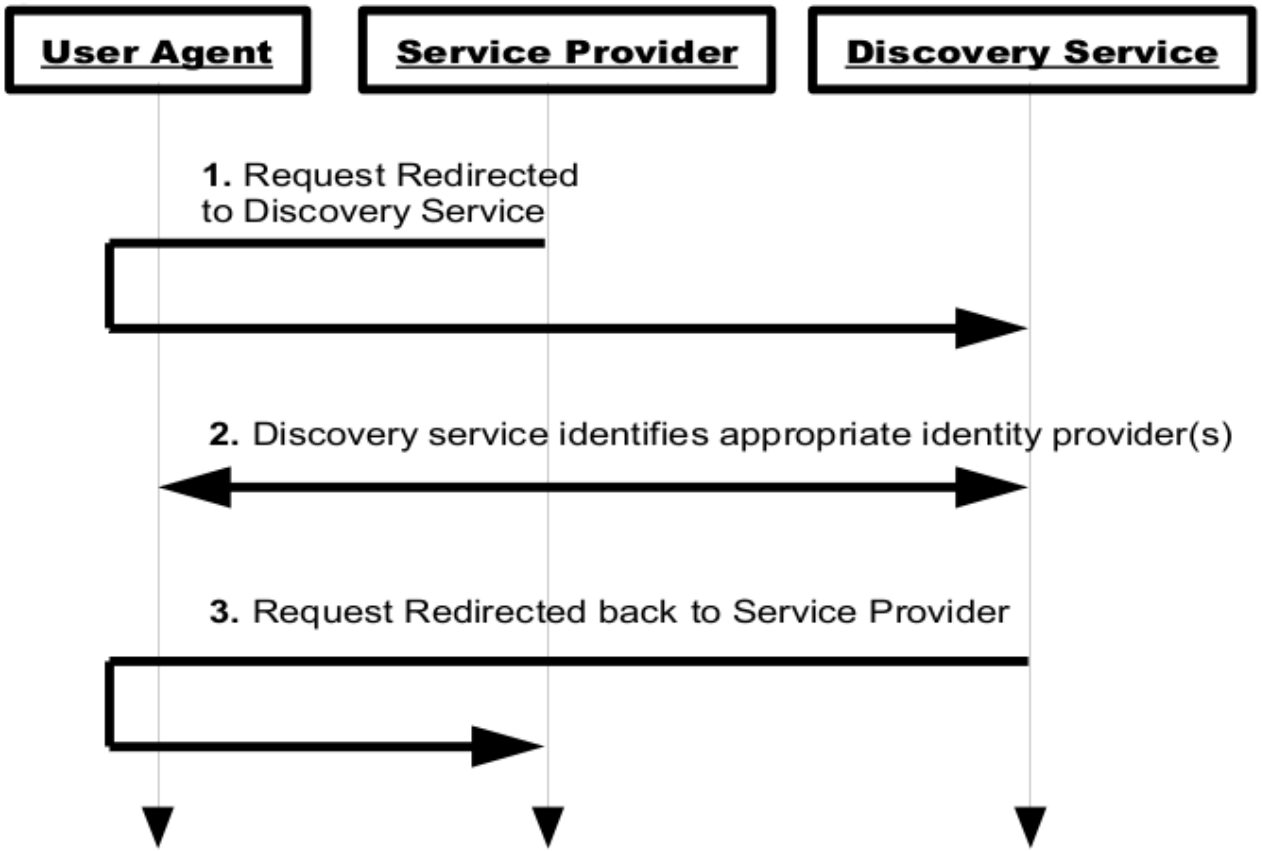
## When an open redirect leads to cookie theft




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]



# 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"*

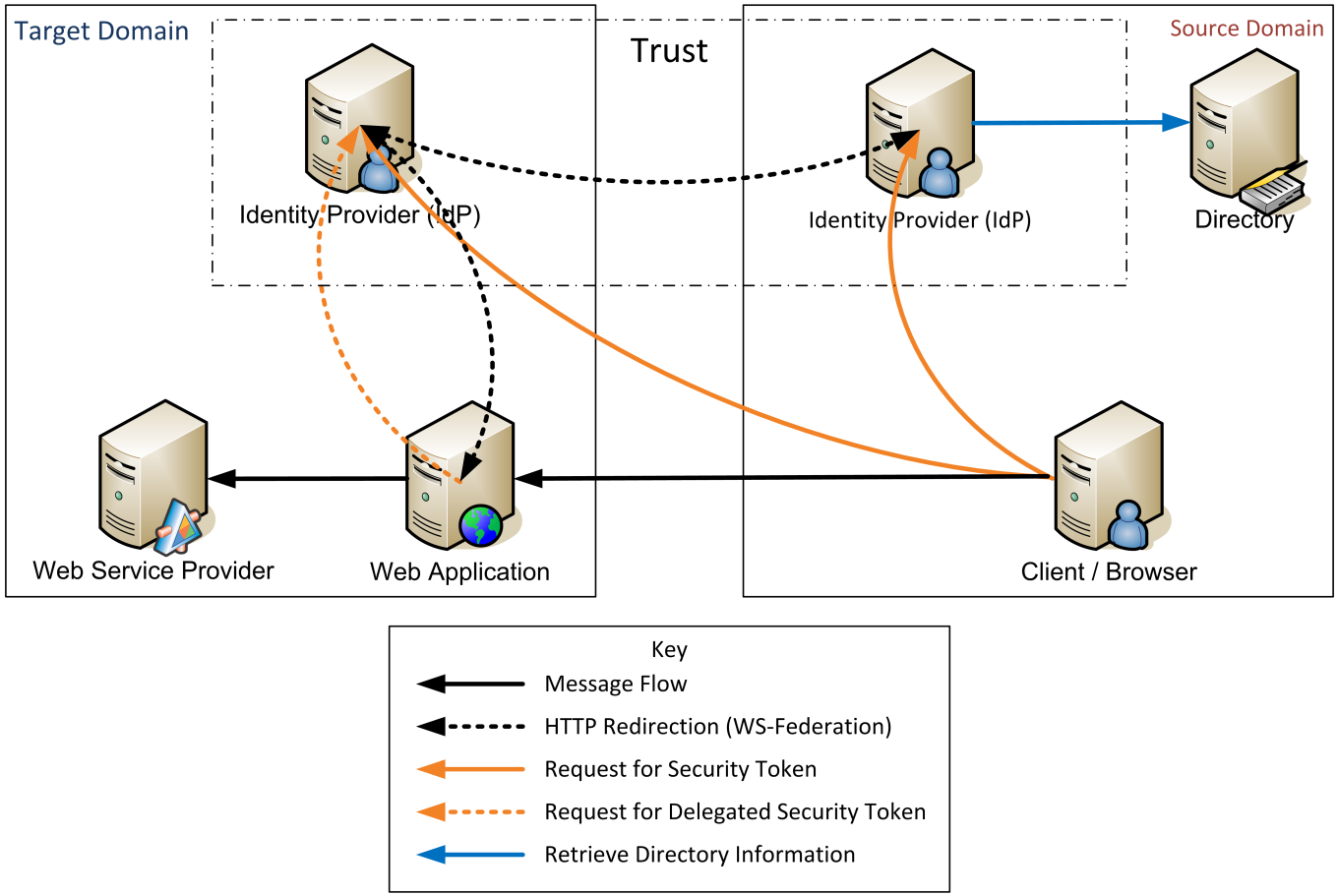
- Developers don't have to implement it to be compliant [30]

# 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

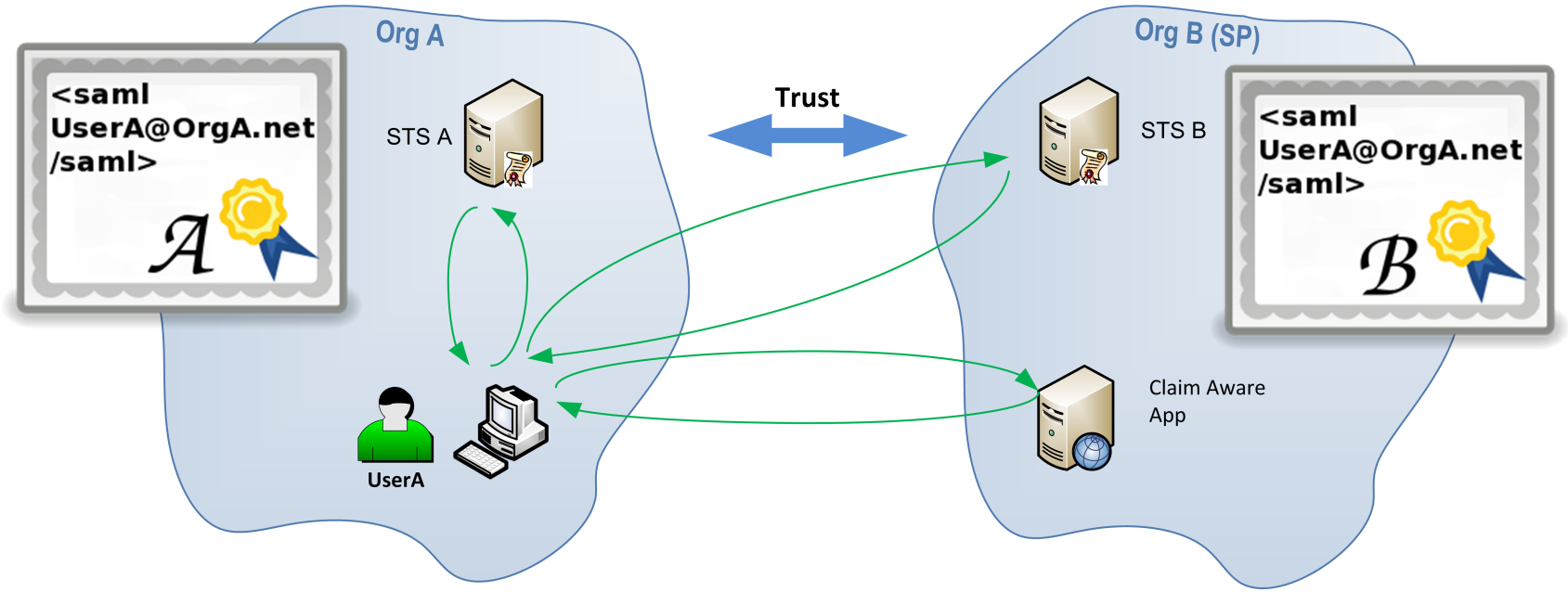
- Network flows
  - Attack surface
  - Management interface
- Users community
  - Insider?
- Data flows

## Cost/Benefit not doing it?

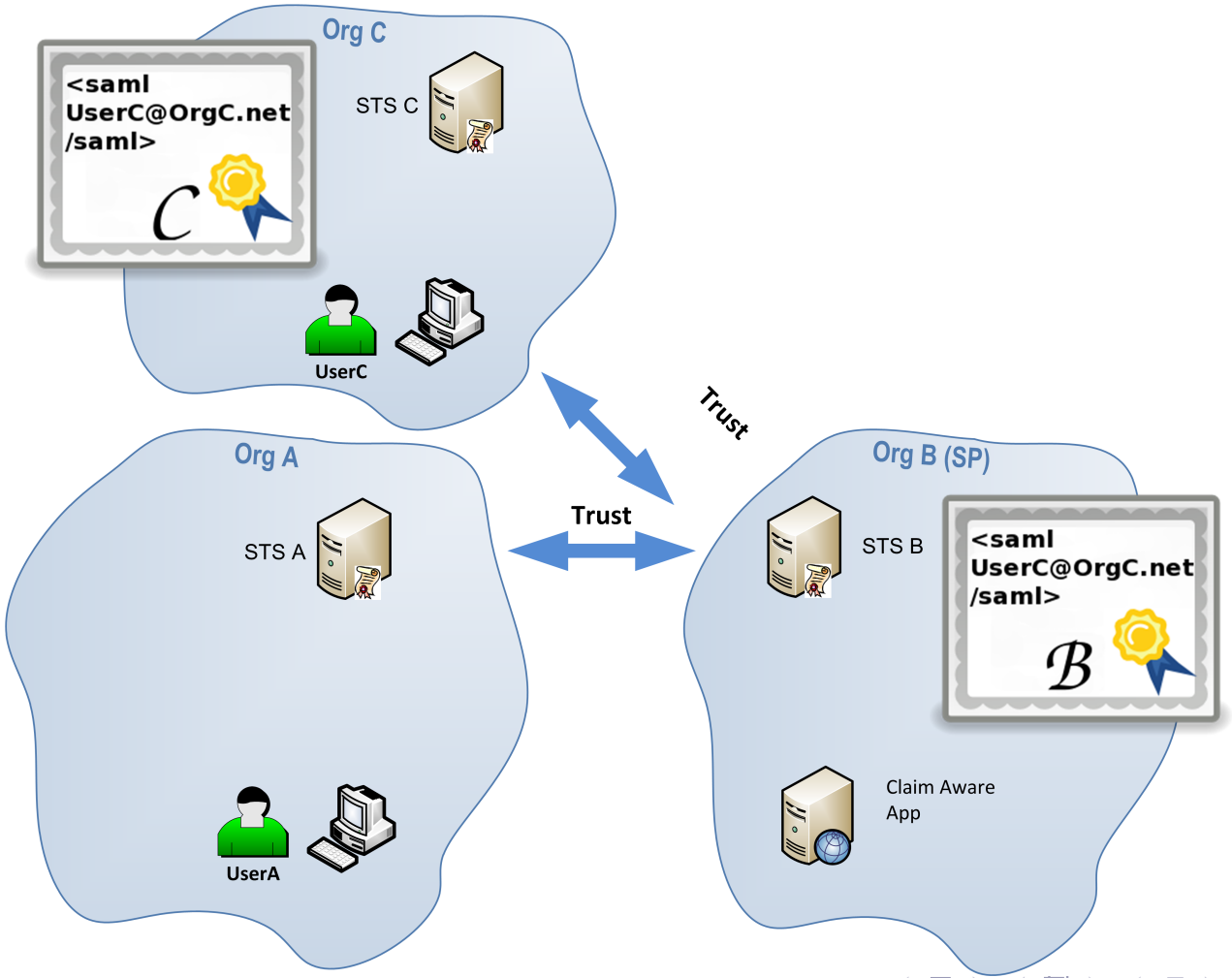
- Security policies coherency / comparison / enforcement

# Similar flows orchestrated in federated environment

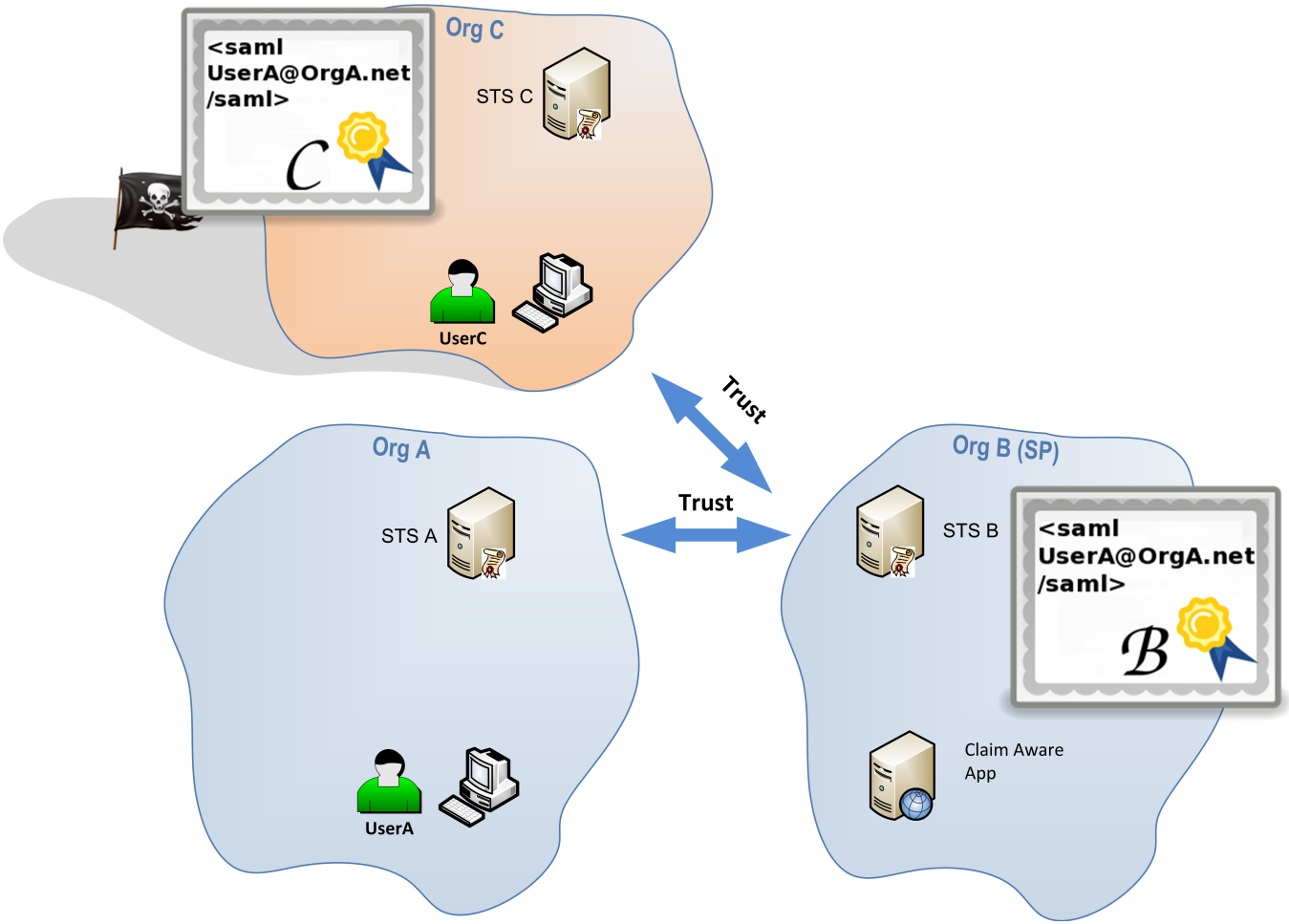
## simple federation scenario [6]



# Similar flows orchestrated in federated environment



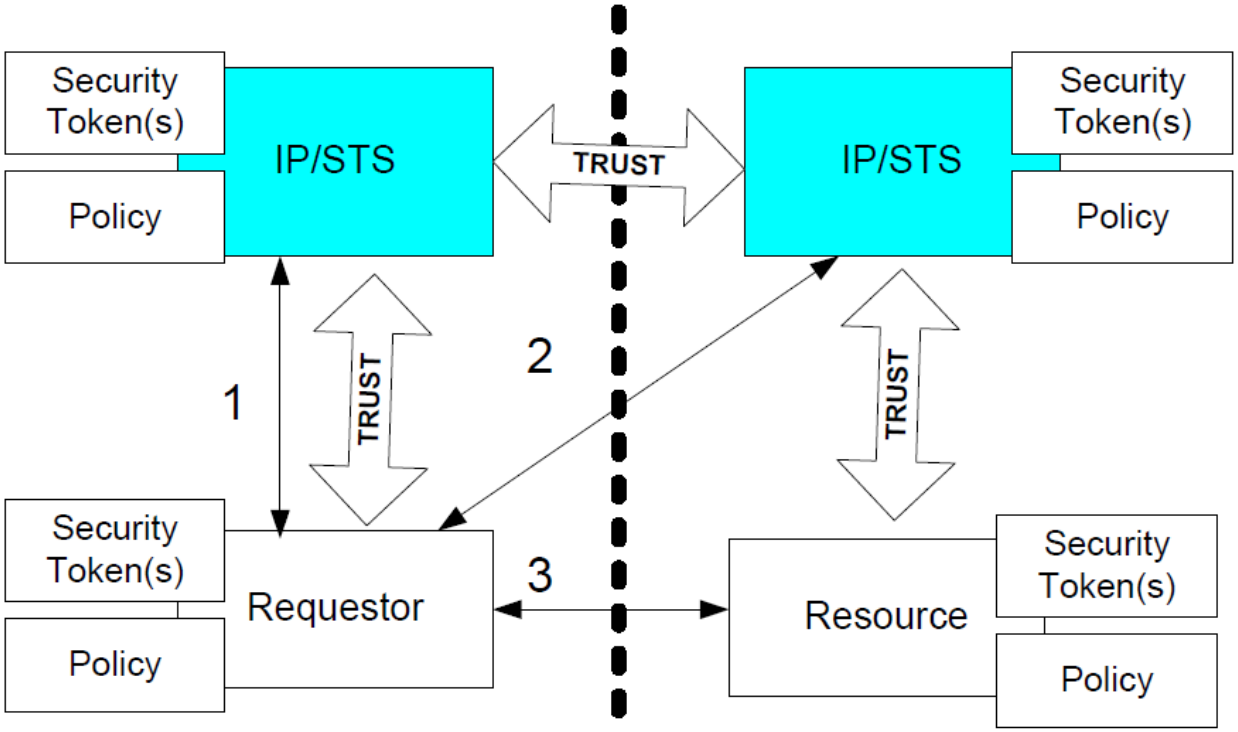
# What if OrgC signs a claim for UserA@OrgA.net?



# Considerations on deployment architectures

## Trust topology

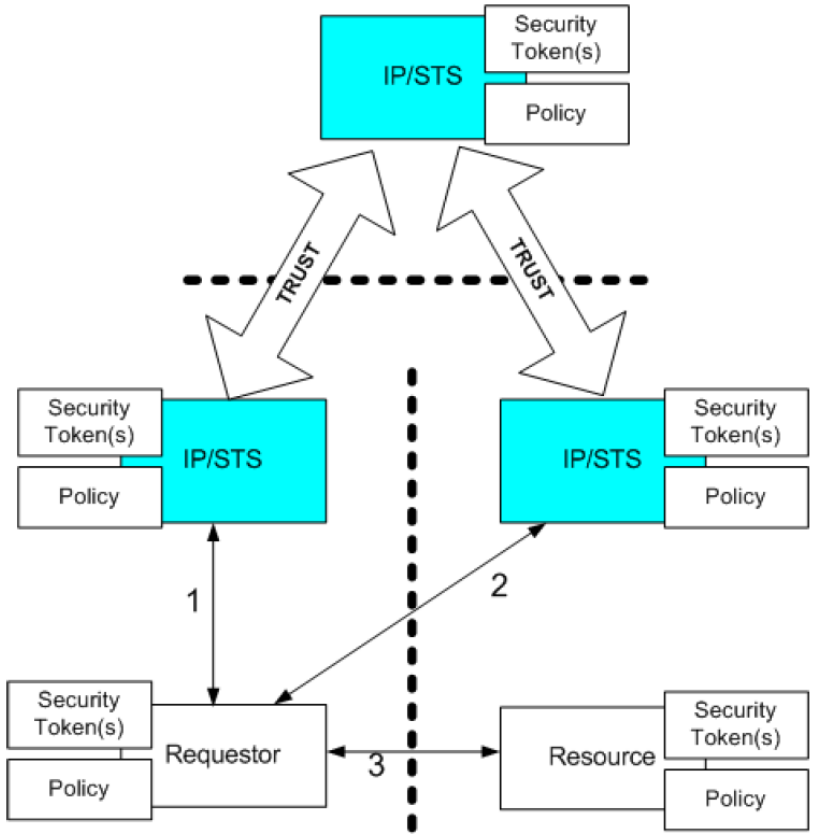
- Previous example follows a direct trust topology [6]



# Considerations on deployment architectures

## Trust topology

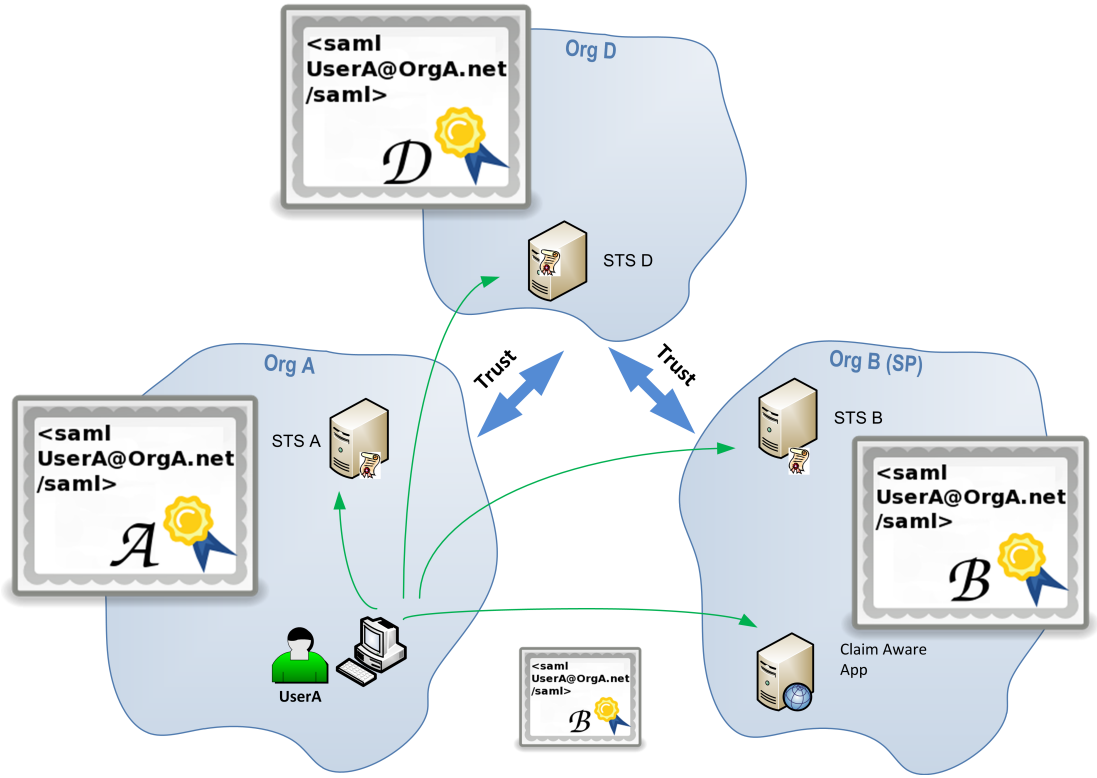
- More complex exist including indirect trust topology [6]



# Considerations on deployment architectures

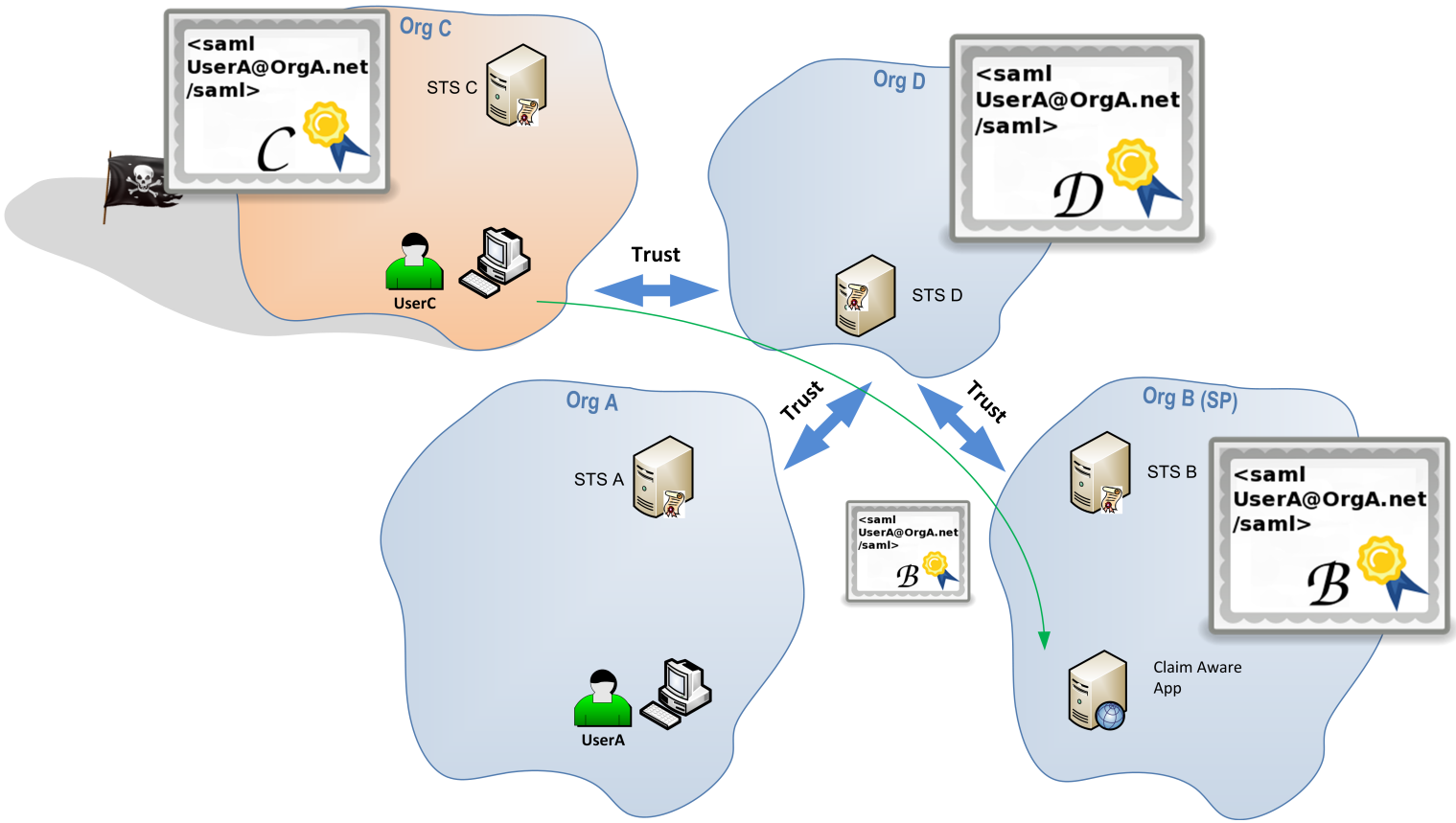
## Trust topology

- 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

[1] K. Cameron - The Laws of Identity - http://www.identityblog.com/stories/2005/05/13/TheLawsOfIdentity.pdf

[2] Anonymity Support for Kerberos - draft-ietf-krb-wg-anon-04 - Kerberos extension

[3] B. Hill - Attacking XML Security - Black Hat Briefings USA 2007 - http://www.isecpartners.com/files/iSEC_HILL_AttackingXMLSecurity_bh07.pdf

[4] OASIS - SAML XML.org - http://saml.xml.org/

[5] D. Hardt - Identity 2.0 - OSCON 2005 Keynote - http://identity20.com/media/OSCON2005/

[6] Web Services Federation Language (WS-Federation) Version 1.2 - OASIS - http://docs.oasis-open.org/wsfed/federation/v1.2/ws-federation.pdf

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

[8] OASIS - SSTC Response to :Security Analysis of the SAML Single Sign-on Browser/Artifact Profile - Working Draft 01, 24 January 2005

# References II

[9] OASIS - Security and Privacy Considerations for the OASIS Security Assertion Markup Language (SAML) V2.0 - OASIS Standard, 15 March 2005

[10] Armando and Al - Breaking the SAML-based Single Sign-On for Google Apps - http://www.ai-lab.it/armando/GoogleSSOVulnerability.html

[11] Security Assertion Markup Language (SAML) 2.0 Technical Overview (draft 3) - OASIS - http://www.oasis-open.org/committees/download.php/11511/sstc-saml-tech-overview-2.0-draft-03.pdf

[12] Security Assertion Markup Language (SAML) 2.0 Technical Overview (draft 10) - OASIS - http://www.oasis-open.org/committees/download.php/20645/sstc-saml-tech-overview-2%200-draft-10.pdf

[13] Myth Breaker - The Best Open Source Web Application Vulnerability Scanner - http://sectooladdict.blogspot.com/2011/01/myth-breaker-best-open-source-web.html

[14] OSSTMM - Open Source Security Testing Methodology Manual - http://www.isecom.org/osstmm/

# References III

[15] OWASP Testing Project - https://www.owasp.org/index.php/OWASP_Testing_Project

[16] Web Application Scanner Benchmark (v1.0) http://sectooladdict.blogspot.com/2010/12/web-application-scanner-benchmark.html

[17] UNINETT releases public beta of SAML tracer - https://addons.mozilla.org/en-US/firefox/addon/saml-tracer/

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

[19] Federation Lab beta - https://fed-lab.org/

[20] J. Haddix, J. Parish - Bsides Chicago 2011 - http://www.securityaegis.com/wp-content/uploads/2011/04/bsides_final.ppt

[21] Pentest John - http://www.securityaegis.com/pentest-john-memes

[22] J. Haddix, J. Parish - ToorCon 12 - http://www.securityaegis.com/burp_preso.pdf

# References IV

[23] WCF Binary Soap Plug-In for Burp - Gotham Digital Science - http://www.gdssecurity.com/l/b/2009/11/19/wcf-binary-soap-plug-in-for-burp/

[24] Burp Suite - http://portswigger.net

[25] Buby's homepage - http://emonti.github.com/buby

[26] Buby tutorial - K. Johnson - http://carnal0wnage.attackresearch.com/2011/05/buby-script-basics-part-1.html

[27] OWASP Open Redirect - https://www.owasp.org/index.php/Open_redirect

[28] Feide - http://www.feide.no

[29] Identity Provider Discovery Service Protocol and Profile - OASIS - http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-idp-discovery.pdf

[30] Support metadata DiscoveryResponse for discovery service - SimpleSAMLphp issue 363 - http://code.google.com/p/simplesamlphp/issues/detail?id=363