The Iam Lotus User Group

Your Auth is open!

Oversharing with OpenAuth & SAML

Andrew Pollack
Northern Collaborative Technologies

IamLUG The Iam Lotus User Group

IBM®

teamstudio®

psc
It's all in the way we listen.®

IamLUG 2013 Sponsors

IamLUG

Who Am I?

- Andrew Pollack – President of NCT

- Administrator & Developer since version 2
- Products
  - NCT Search, NCT Compliance Search,
  - and NCT Simple Sign On, and Second Signal
- Services
  - Site Performance Reviews
  - Application Development
  - Administrative Overhaul
  - Security Review & Penetration Testing
- IBM Lotus Beacon Award Winner

- Structural Firefighter – Lieutenant of Engine Company 1 in Cumberland, Maine

SSO & Shared Authentication Topics

- SSO vs. Shared Authentication
- Use Cases
  - Why not just use LTPA or Active Directory?
- Specific Security Concerns
  - What to consider when planning
- Creating your own simple specification
- Emerging Standards
  - SAML (Security Assertion Markup Language)
  - OpenID
  - OAuth
- A Real World Example

SSO vs. Shared Creds - What’s the Difference?

Single Sign On

- Enter Credentials Once and you are signed in at multiple sites
- Wikipedia lists dozens of projects
  - http://bit.ly/WRaxPq

Shared Authentication

- May need to re-enter the same credentials at each site
- LDAP
- Domino HTTP Password Sync with Notes ID

Why not just use LTPA or Active Directory?

USE CASES

Why not just use LTPA, AD, or similar?

- Incompatible Technology
  - Not every server fully supports these methods in the same way
  - Not all users of your application come from the same source
    - Different AD Forests
- Incompatible Organizations
  - Your application may not serve only users in your company
    - 3rd Party Service Providers
    - Portals making use of 3rd Party Providers
- Someone else is setting the specification
  - You don’t always get to pick your favorite protocols
    - SAML is hip and cool right now

What to look out for when you implement – Things that your users and customers will expect that you've considered

SPECIFIC SECURITY CONCERTS

How much do you trust the credential provider?

- Their security weaknesses are now yours
  - If someone can bypass or otherwise game their authentication process, that person can now access your site
- Your site's availability is now subject to theirs
  - If they go down, users cannot access your site
    - Depending on the schema, it may look like your fault
- Is Your Privacy Policy Is Still Accurate?
  - Can you really guarantee the safety of user data if you're not providing the authentication?
- Can you protect your administrative logins?
  - What prevents the remote site from passing someone to you with an administrative user id?

Users will still expect common services

- You may no longer be managing a users credentials but your users will still expect some things to work well
  - If you don’t provide a method to handle these, your phone will ring frequently.
- Log Off
  - Will the log off button on your page work?
    - …or will they redirect to the auth provider and bounce back to you already logged in?
- Password Change / Reminder
  - Make sure you provide links back to wherever the user has to go for making these changes
- What happens when authentication or authorization fails?
  - Will you create a redirection loop?
- Help & Support Links

How can user access be revoked?

- Is there a way for the authentication provider to notify your server to log off a user?
  - If several systems are sharing authentication, does logging off one of them log off the rest?

- If a “Problem” user is accessing your system but authenticating somewhere else, can you lock them out?

- Can you block certain user login ids from being passed from the provider?

Are you hack resistant?

- Can the authentication provider be spoofed
  - Are you sure the credentials you're being sent really represent the user connecting to you?
- Can the credential data being passed to you be altered?
  - When the credentials are passed to you, are they visible or even editable by the user?
- Can a link to your site generated by the provider by bookmarked and re-used by the user?
  - Is there a time limit built into the secure credential data?
- Does your site expose data from the credential provider that can be used to access other sites?

Authentication is not Authorization

- Who you are does not tell us what you can do
- Many SSO implementations also require a back end data integration phase
  - Pre-Shared user data to pre-populated access groups
    - Authorization is ready when the user hits the site
    - Requires significantly more data integration
    - Requires a matching key between data and login id
  - First time access questionnaires
    - Often require a validation step
    - User access to content or services may be delayed
    - May result in duplication of data from difference sources
      - *Which eventually means a time and cost intensive reconciliation project*

Opportunities to add some control

- Consider putting all SSO logins in a specific "organization" or "Organizational Unit"
  - E.g. "SSOName/SSO" or "SSOName/SSO/MyOrg"
    - Prevents the credentials from using your admin accounts
    - Allows you to use wildcards in group and ACL entries
      - *E.g. */sso or */sso/MyOrg*
- Make full use of the "Maximum Internet Name and Password" database ACL setting
  - Just in case the credential provider provides credentials which would have admin level access

Roll your own or user a standard, either way you need a schema –

We'll talk about rolling your own first, because it will explain why some things are done in the OAUTH and SAML standards when we get there.

SSO SCHEMAS

Creating Your Own Schema – What You Need

- **Minimum Requirements**
  - A way to know that the credentials came from the provider and were not counterfeited
  - A way to know when the credentials were last authorized by the provider
- **Additional Requirements**
  - User meta data
  - Authorization Criteria

Creating Your Own Schema – The Encrypted Packet

- This can work both ways – with Domino as the authentication provider, or consumer
  - Is it your portal using a 3rd party, or are you the 3rd parth?
- Simple Idea – A signed and/or encrypted packet of data is included as a URL parameter
  - http://your.server/landingdb.nsf/landingagent?openagent&userdata=[packet]
    - The URL is generated at the remote side as a link
    - http/https request can be done as a redirect or link
- Why not use a form action POST and put the data in a field value?
  - Form submissions require the user to click a link to post the data, so redirection becomes far more difficult
  - May raise security warnings at the browser side

Creating Your Own Schema – What should be in the packet

- The user id itself
  - Do you have a standard user id format?
    - Domino doesn’t like an “@” in a username
    - You may have unusual issues with hierarchical names
  - You really should include a time stamp
    - Allows you to invalidate a packet after a given time
      - *Prevents bookmarking or sharing links with credentials*
    - Make sure you agree on the time zone (just use GMT)
  - You may also want to include meta-data
    - Allows you to assign authorization as well

Creating Your Own Schema – Protecting the Cred Packet

Digital Signature

- Uses another parameter to provide the signature itself
- Requires pre-exchange of data (public keys or hash salt value)
- May use current x.509 standards or older technology
- Does not prevent the data from being visible
- Open source libraries are available, but can be very complex to use

Encrypting the Data

- May use current x.509 standards or older encryption schemas like Blowfish
- Requires pre-exchange of data (public keys or hash salt value)
- Makes data unreadable to end users or man-in-the-middle
- Open source libraries are available, but can be very complex to use

Creating Your Own Schema – Protecting the Cred Packet

- **Encrypting with Blowfish**
  - Easy to find open source implementations for most languages
  - Simple password allows decryption and proves source
    - If you can decrypt it, you know the other end had the password to encrypt it.
    - Agree on a password change if you need to re-secure
  - May not be up to the most current security requirements
    - Still adequate for most uses
  - Not the way the “cool kids” do things any more

Creating Your Own Schema – Protecting the Cred Packet

- **Encrypting with x.509**
  - Currently very much in fashion
    - Support the latest encryption standards
  - Open source libraries available, but can be complex to use
    - Not just in Domino – Accessing the “Keystore” on an IIS server is very tricky as well.
  - May require paying for recognized certificates
    - Some library stacks do not like self signed certificates
  - Requires exchange of public keys
    - Never trust the key sent with the packet
  - Certificates are revocable

Creating Your Own Schema – Encoding the packet

- You have to encode the packet for URLs
  - Encrypted or not, it will contain characters that can’t be stuck in a url without problems.
- HEX encoding
  - Two hex digits for each digit of encrypted data
  - Can handle pretty much any data
  - Results in very long URLs
- Base64 encoding
  - Open source libraries for most languages
  - Results in shorters URLs
  - Padding “=“ at the end can interfere with URL parsing
- URL “Escape” sequence encoding
  - Very cumbersome – looks like someone vomited % characters
  - Results in very long URLs

Openid – common, cheap, and not very secure

EMERGING STANDARDS

OpenID Overview

- Useful for low security public facing sites like blog comments and discussion boards
- Because OpenID is so open, the level of trust you can place in credentials is very limited.
- Many well known OpenID providers
  - Google, Yahoo! LiVE JOURNAL, Blogger, Aol
- You can create your own provider
  - But not all sites that accept OpenID will use it
  - Many sites just use specific buttons to authenticate using known OpenID providers
- Not directly supported by the Domino Web Server
  - But it can be done
- For more: http://openid.net/

OAuth – The standard that isn't standard

EMERGING STANDARDS

OAuth Overview

- Complementary to OpenID
- OpenID provides Authentication while OAuth provides for Authorization
- OAuth works like a “valet key”, authorizing third party applications to do things under your credentials on a site.
- Major split between version 1 and version 2
  - Original author no longer involved
  - Version 2 implementations “unlikely to be compatible” with each other.

OAuth Terminology

- Resource Owner: Who’s Content Is it?
- Client: Who wants to access the content?
- Server: Where does the content live?

OAuth Credential Types

- Client Credentials
  - Typically the user’s server login
- Temporary Credentials
  - May be used to track the authorization request between the client and the server
- Token Credentials
  - Issued by the server to the client as a stand-in for the client credentials without giving those away
  - Can usually be revoked at the server by the resource owner (e.g. Remove this application’s authorization)

OAuth Request Types

- **Two Legged Request**
  - Where the Client and the Resource Owner are the same
- **Three Legged Request**
  - Where the Client is a third party (like an app) acting with authorization from the Resource Owner
- **N-Legged Request**
  - Used when “re-delegation” is allowed (works like a three legged request)

OAuth Use Cases

- Third party web site apps
  - E.g. Facebook Games
- RSS Feed Aggregators
- Third Party Client Software
  - E.g. Twitter Applications
- Notes 9
  - Integration with Connections

SAML – All the cool kids are using this one

EMERGING STANDARDS

SAML Terminology

- Security Assertion Markup Language
- IdP – Identity Provider
- SP – Service Provider
- Assertion – What the IdP tells the SP

SAML Overview

- SAML is a very rich and detailed spec which provides for passing identity along with meta data between an Identity Provider and one or more Service Providers
- Data is passed in XML packages
  - Generally using http protocols, but not necessary always. The XML can be passed almost any way.
- Packaged XML can be signed, encrypted, both, or neither
- Communication can be made directly between the SP and the IdP or the XML packages can be passed by the requesting client.
  - Usually, the packets are passed by the requesting client as part of the http GET or POST data

SAML Benefits/Use Cases

- A single trusted, authoritative source is used to authenticate users who then need access to resources on multiple servers
  - often outside the control sphere of the authoritative source.
- Allows third parties to provide services to your user community, while management of that community remains centralized.
- Highly flexible security and meta data capabilities allow a wide range of interoperability
  - We'll talk about "Assertions" in a minute

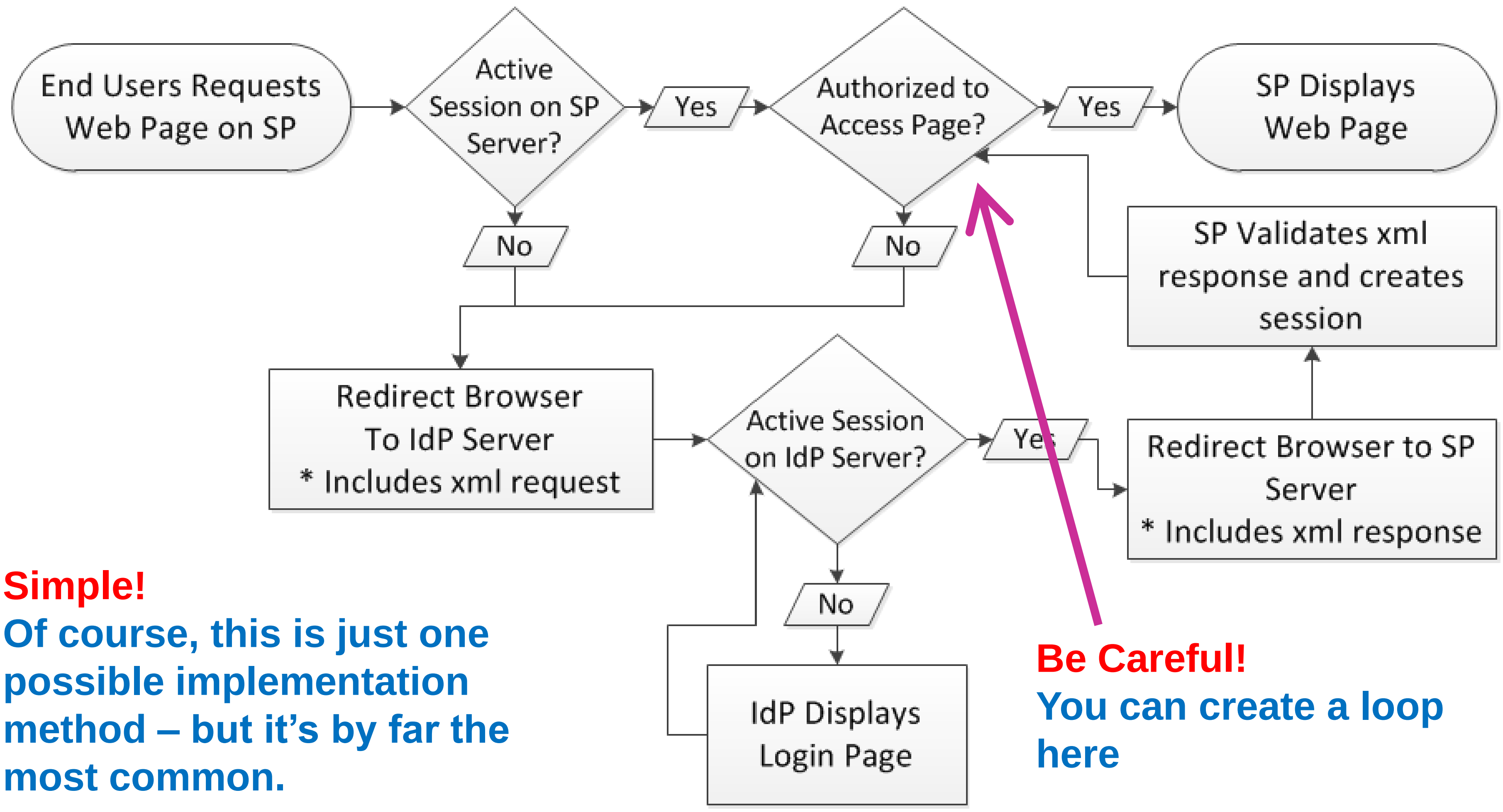
SAML Setup

- The IdP and the SP MUST establish a trust relationship for exchanging credentials and keys outside this process
  - Typically by exchanging official x509 certificates to be used for signature validation and decryption
  - Public keys are commonly also transferred inside the xml transactions, however these cannot be trusted unless the SP and IdP servers are in direct, verified, secure communication
- The IdP provides set up information in an “IdP.xml”
  - Contains the resource locations, Identifiers, requirements and defined services for all future transactions between the IdP and the SP
- The SP imports that data and responds with an “SP.XML”
  - Contains the SP identifier, resource locations, and defined services for this service provider
- These set up files are usually exchange manually, during the project implementation phase.

The Assertion is The Heart of SAML

- The IdP "Asserts" specific information to the SP
  - The UserID and other metadata attributes
  - The format of the userid and each attribute
  - The timespan in which the assertion is valid
  - Other conditions placed on this use of this info
    - Audience Restriction, One Time Use, Proxy Use, etc.
  - Assertions are usually signed and may be encrypted as well

Typical SAML Process Flow


End Users Requests Web Page on SP
Active Session on SP Server?
Yes
Authorized to Access Page?
Yes
SP Displays Web Page
No
No
SP Validates xml response and creates session
Redirect Browser To IdP Server * Includes xml request
Active Session on IdP Server?
Yes
Redirect Browser to SP Server * Includes xml response
No
IdP Displays Login Page

Simple!
Of course, this is just one possible implementation method – but it's by far the most common.

Be Careful!
You can create a loop here

SAML in Domino 9

- Domino acts as an SP only, not an IdP
- Currently only supports two IdP Products
  - Microsoft Active Directory
  - Tivoli Federated Identity Manager
- There are reports of it working with others
  - Most common IdP I’ve seen is Oracle Federated Identity (add on to Oracle Identity Manager)
- Requires a Notes ID and Person Document for all federated Notes Client users
  - but not necessarily browser access users
- Requires the use of ID Vault if used for Notes Client federated login

Game Over

Thank you for playing

Insert .25 (or ask a question) to Continue

Contact me:

andrewp@thenorth.com

@FireFighterGeek

http://www.thenorth.com

IamLUG