Change Notify Proposal

Internet Identity Workshop XI
Nov 2-4, 2010

Phil Hunt, Oracle Corporation
@independentid
phil.hunt@oracle.com

Notice

- The following information reflects work being carried out at the OASIS Security Services Working Group. The opinions expressed are my own and do not necessarily reflect those of my employer or OASIS.

Agenda

- Introduction
  - What is the problem?
  - The use case?
- Status
- Working Draft 04
  - Basic features
  - Schema / Protocol
- Discussion

Characterizing Today's Fed Id Systems

- Federation has grown, but identity information is treated as largely static, or focused exclusively on SSO
- Many applications need limited claims but…
- Many applications need as many as 70 to 200 attributes
- Entity state firewalled

Challenges

- Control of State
  - Federated scenarios block full knowledge of entity state between parties
  - Control of entities by multi-party consent (e.g. user!)
  - Federated SPs do not return detailed errors if any
    - E.g. is a user really present in an IDP or not?
  - ☛Enterprise ‘style’ provisioning is difficult across domains

Challenges

- Limitations of Claims
  - – Is sign-on the appropriate time for claims transfer?
  - – Not all claims come from a single provider
  - – Applications are claims providers
  - – Applications often need to retain claims
    - Offline processing
    - Workflows
    - Multi-tier architectures

☛One-way, SSO flow, is not often possible

Use Cases

- Application Workflow
  - An application facilitates the transfer of a user from one IDP to another
  - The application wants to provide a “warm introduction” of a new user to an IDP
  - User might have prior relationship with new IDP
  - Application needs to know user is registered with new IDP
  - After confirming new IDP, application wants to notify old IDP
  - E.g. transferring user applications, accounts between telcos

Use Cases

- Enterprise Cloud Services
  - Service provider needs to know when employee is retired, or de-registered (offline update?)
  - Service provider needs to differentiate between first time registration, vs. ongoing SSO
  - Service provider may need to update enterprise with cloud service generated claims
  - E.g. Legal service provides proof of legal residency or visa status

New-User Registration

- How to distinguish between first time registration vs. iterative updates
- Warm introductions – combining SSO with a notification
- Identifier handling
- Are IDP vs. RP roles always clear in a community of applications and services?

Updates

- Transferring information as-needed vs. at sign-on time
- Some SPs both retain and generate claims
- If information isn’t always transferred at sign-on, then how are updates transferred?
- How do SPs who generate claims update IDPs?
- How do IDPs notify SPs who retain data?

Removals

- What happens when a user cancels a service?
- De-provisioning vs. de-federation
  - Deleting / disabling vs. de-coupling
  - E.g. an enterprise notifies a service provider of a employee retirement

Why Not Use Provisioning Systems?

- Often the recommended approach today, but…
- Enterprise provisioning only works when
  - Assumption of ownership
  - Tight control over entity “state”
- Yet, federation assumes
  - Independent organizations
  - User-control options
  - Firewalled knowledge
  - Entity “state” cannot be presumed

Status of Proposal

- Currently a SAML Protocol Proposal
- Written by NSN and Oracle as part of the OASIS Security Services Technical Committee
- Working Draft 04
  - http://www.oasis-open.org/committees/document.php?document_id=40036
- Heading towards committee draft status

Change Notify Protocol

- Adds update capability to federation protocols
- Terms: Notify Issuer and Notify Targer
- 2-step approach
  - Notification step
  - Action step
- Notification provides context to an action step that can be push or pull
- E.g. SAML Attribute Query can now be used for attribute modification

Notification Step

- Types
  - NewSubject – One or more identifiers which the notifier believes to be “new”
  - ModifySubject – One or more identifiers listing one or more attributes that are to be “changed”
  - RemoveSubject – One or more identifiers to be “removed”
- Notifications contain only identifiers
- Boxcarring – use of one or more identifiers allows message traffic to be reduced
- Can be used in online, front-channel profiles
- No claims / values transferred (except idenitifers)
- Message SHOULD be signed

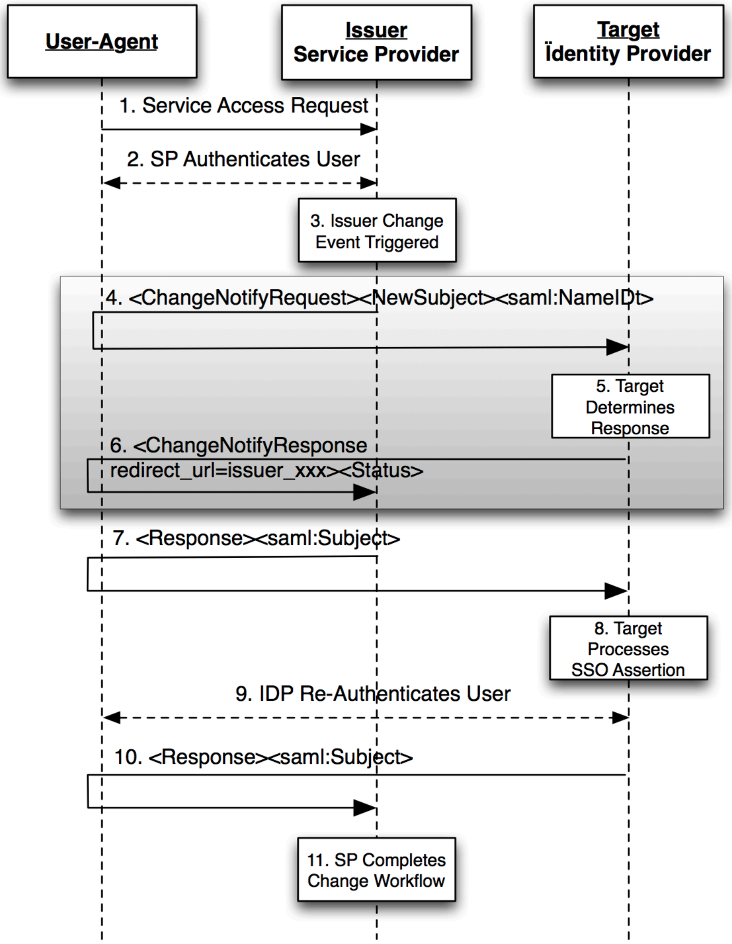
Action Step

- Uses existing protocols to facilitate claims transfers
- Protocol could be almost anything:
  - SAML, OpenID, LDAP, SPML, PortableContacts, …
- E.g. NewSubject notification is followed by Web SSO profile to facilitate transfer of user information, ‘in-context’, and provide ‘warm introduction’

History

- Working Drafts 01, 02
  - Exploration of push model: Add, Modify, Remove
  - Primary problems became
    - Error handling
    - Need to quantify entity state
- Working Draft 03
  - Evolution to 2-step
  - Push notification followed by negotiated multi-protocol action step
  - Boxcarring permitted
  - Issues
    - How to handle name identifiers for multiple protocol choices
    - Too much negotiation
- Working Draft 04
  - 2-step
  - Push notification followed by pre-negotiated protocol step
  - Simplification
    - No in protocol negotiation of "action" step – but can be achieved
    - Per protocol end-points
    - Identifier handling
    - Single multi-purpose front-channel and back-channel profile

SP Initiates 'Warm' Registration



User-Agent

Issuer
Service Provider

Target
İdentity Provider

1. Service Access Request

2. SP Authenticates User

3. Issuer Change Event Triggered

4. <ChangeNotifyRequest><NewSubject><saml:NameID>

5. Target Determines Response

6. <ChangeNotifyResponse redirect_url=issuer_xxx><Status>

7. <Response><saml:Subject>

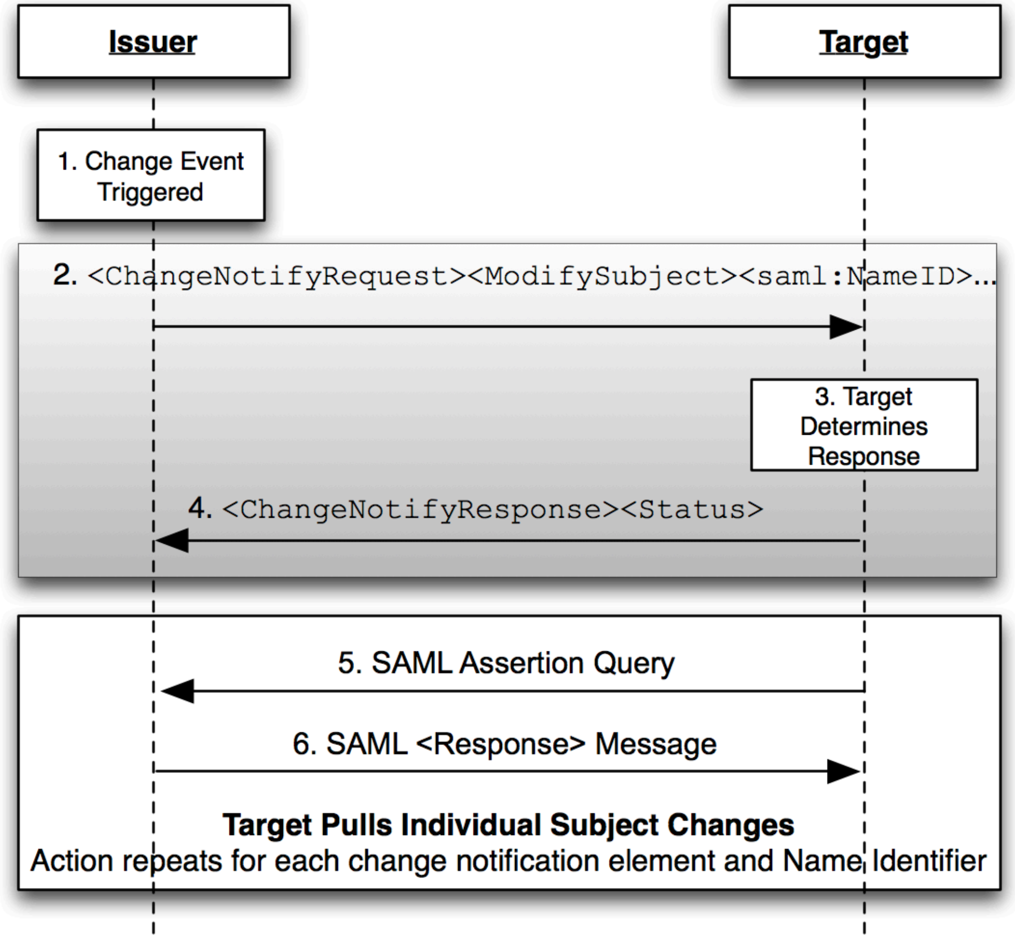
8. Target Processes SSO Assertion

9. IDP Re-Authenticates User

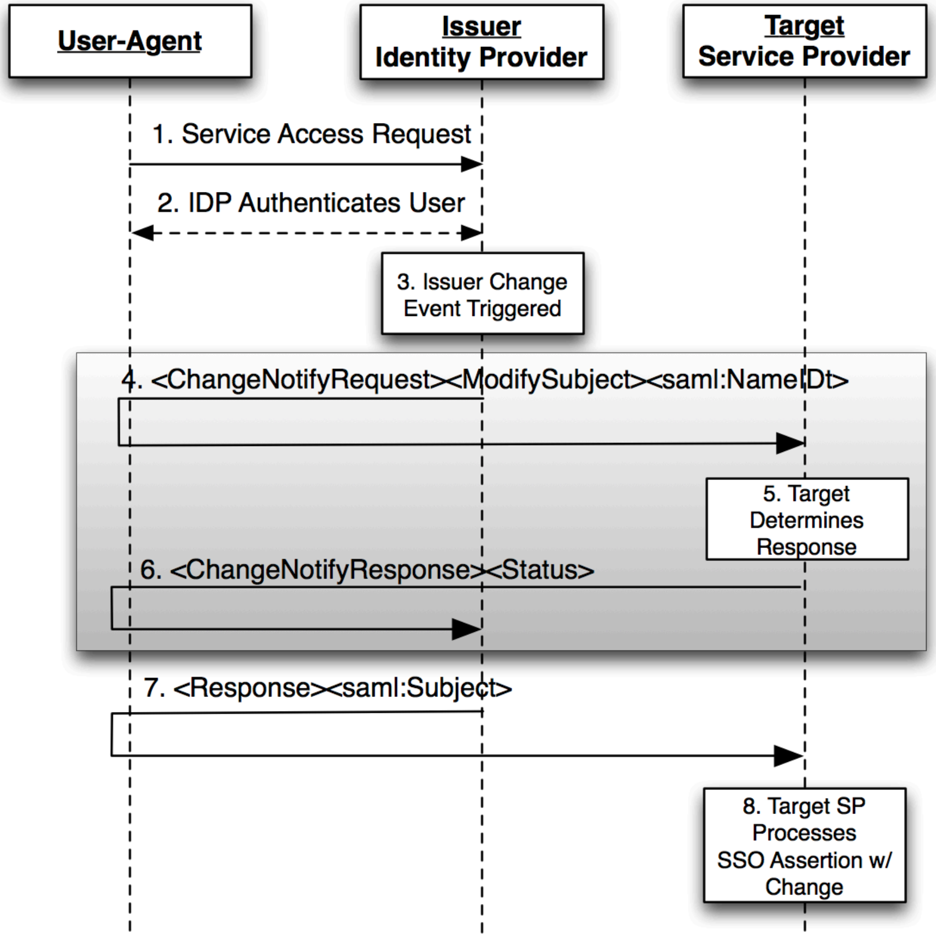
10. <Response><saml:Subject>

11. SP Completes Change Workflow

Backchannel Update


Issuer
Target
1. Change Event Triggered
2. <ChangeNotifyRequest><ModifySubject><saml:NameID>…
3. Target Determines Response
4. <ChangeNotifyResponse><Status>
5. SAML Assertion Query
6. SAML <Response> Message
Target Pulls Individual Subject Changes
Action repeats for each change notification element and Name Identifier

IDP Initiated Change


User-Agent
Issuer Identity Provider
Target Service Provider
1. Service Access Request
2. IDP Authenticates User
3. Issuer Change Event Triggered
4. <ChangeNotifyRequest><ModifySubject><saml:NameIDt>
5. Target Determines Response
6. <ChangeNotifyResponse><Status>
7. <Response><saml:Subject>
8. Target SP Processes SSO Assertion w/ Change

Example SAML Notify Request

```
<samln:ChangeNotifyRequest xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
    xmlns:samln="urn:oasis:names:tc:SAML:2.0:notify"
    ID="aaf23196-1773-2113-474a-fe114412ab72" Version="2.0"
    IssueInstant="2006-07-17T20:31:40Z"
    protocol="urn:oasis:names:tc:SAML:2.0:notify:protocol:saml:FrontChannel" >
    <NewSubject>
     <saml:NameID
           Format="urn:oasis:names:tc:SAML:1.1:nameidformat:X509SubjectName">
           C=US, O=NCSA-TEST, OU=User, CN=john.doe@corp.com
     </saml:NameID>
     <saml:Attribute
           xmlns:x500="urn:oasis:names:tc:SAML:2.0:profiles:attribute:X500"
           x500:Encoding="LDAP" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
           Name="urn:oid:2.5.4.42" FriendlyName="givenName">
     </saml:Attribute>
     <saml:Attribute
           xmlns:x500="urn:oasis:names:tc:SAML:2.0:profiles:attribute:X500"
           x500:Encoding="LDAP" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
           Name="urn:oid:1.3.6.1.4.1.1466.115.121.1.26" FriendlyName="mail">
     </saml:Attribute>
    </NewSubject>
</samln:ChangeNotifyRequest>
```

Response

```
<samln:ChangeNotifyResponse xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
    xmlns:samln="urn:oasis:names:tc:SAML:2.0:notify"
    xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
    ID="aaf23196-1773-2113-474a-fe114412ab72" Version="2.0"
    IssueInstant="2006-07-17T20:31:40Z">

    <samlp:Status>
     <samlp:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Success" />
    </samlp:Status>
</samln:ChangeNotifyResponse>
```

Future

- Should Change Notify exclusively be a SAML Protocol?
- Is there interest in exploring a lightweight variant?

Discussion