User-Managed Access
UMA Work Group

@UMAWG
tinyurl.com/umawg | tinyurl.com/umafaq

IIW 16, May 2013
The “data price” for online service is too high: typing…

- Provisioning by hand
- Provisioning by value
- Oversharing
- Lying!
The “data price” for online service is too high: connecting...

- Meaningless consent to unfavorable terms
- Painful, inconsistent, and messy access management
- Oblivious oversharing
The “data price” for online service is too high: private URLs…

• Handy but insecure
• Unsuitable for really sensitive data
Most data “sharing” today is back-channel and unconsented
Privacy is about context, control, choice and respect – so UMA enables a “digital footprint control console”

- Web 2.0 access control is inconsistent and unsophisticated
- To share with others, you have to list them literally
- You have to keep rebuilding your “circles” in new apps
- You can’t advertise content without giving it away
- You can’t get a global view of who accessed what

- You can unify access control under a single app
- Your access policies can test for claims like “over 18”
- You can reuse the same policies with multiple sites
- You can control access to stuff with public URLs
- You can manage and revoke access from one place
UMA turns online sharing into a privacy-by-design solution

- Historical
- Municipal
- Financial
- Vocational
- Artistic
- Social
- Geolocation
- Computational
- Genealogical
- Biological
- Legal
  ...

Diagram:
- Resource owner
- Authorization server
- Client
- Requesting party
- Resource server
- Manage
- Protect
- Control
- Consent
- Negotiate
- Access
- Manage
UMA turns online sharing into a privacy-by-design solution

I want to share this stuff selectively:
- Among my own apps
- With family and friends
- With organizations

I want to protect this stuff from being seen by everyone in the world.
UMA turns online sharing into a privacy-by-design solution

I want to control access proactively, not just feel forced to consent over and over!
UMA is a profile of OAuth, with bits added for interop and scale.
UMA solves for 1) individual choice and 2) fully modular cloud services
UMA solves for 1) individual choice and 2) fully modular cloud services.
UMA solves for
1) individual choice and
2) fully modular cloud services
Key use cases

http://kantarainitiative.org/confluence/display/uma/Case+Studies

- Subscribing to a friend’s personal cloud
- Sharing accessibility attributes (“GPII”)
- E-transcript sharing (“HEAR”)
- Patient-centric health data access
- Enterprise “access management 2.0”
Key implementations

http://kantarainitiative.org/confluence/display/uma/UMA+Implementations

• SMARTAM.net (running authorization service from Cloud Identity UK)
• Puma (Python libraries for RS- and client-enabling web apps) from ditto
• Fraunhofer AISEC open-source implementation in Java
• Gluu OX open-source implementation for Access Management 2.0 use cases
Next steps

• Work on optimization opportunities when UMA and OpenID Connect are used together
• Issue “Implementor’s Draft”
• Continue to work with AXN, Scalable Privacy, and others in “trusted identities in cyberspace” ecosystem
• Profile UMA for higher ed, accessibility attribute sharing, healthcare use cases
• We welcome your involvement and contributions
  – Become an UMAntarian!
  – Follow @UMAWG on Twitter and UserManagedAccess on FB
Questions?
Thank you

@UMAWG
tinyurl.com/umawg | tinyurl.com/umafaq
IIW 16, May 2013
Phase 1: protect a resource
Phases 2 and 3: get authorization and access resource

Token terminology:
* PAT = protection API token
* AAT = authorization API token
* RPT = requesting party token

Binding obligations terminology, as shown in notes over entities representing obligated parties (see http://docs.kantarainitiative.org/uma/draft-uma-trust.html):
* Subject = Individual or Non-Person Entity
* Authorizing Party = Subject acting as resource owner
* AS Operator = Subject operating authorization server endpoint
* RS Operator = Subject operating resource server endpoint
* Requesting Party = Subject acting as requesting party

Flow scenario:
* Client starts out with no AAT or RPT but is ultimately able to qualify for the required authorization

Client presents no RPT (Sec 3.1.1)

Learn protected resource location and scopes out of band (Sec 3.1)

Attempt access with no RPT
401 with AS location

Look up AS config data (Sec 1.5)

RS Operator-Requesting Party: Give-Accurate-Access

Dynamic client registration if necessary (Sec 3.4, draft-ietf-oauth-dyn-reg)
Phases 2 and 3: get authorization and access resource

2 of 3
Phases 2 and 3: get authorization and access resource

1 of 3

AS Operator-RS Operator:
Follow-Policies-Accurately-and-Timely

403 with AS location and permission ticket (Sec 3.1.2)

RS Operator-Requesting Party:
Give-Accurate-Access

Client asks for authorization data (Sec 3.4.2)

Request authorization, presenting AAT, PAT, permission ticket

Redirect to AS... (Sec 3.5.1)  
(assumes human RqP)

...to provide claims as required by RO's policy

Client presents RPT with sufficient authorization data (Sec 3.1.2)

Attempt access with RPT

Determine RPT status and authorization data (Sec 3.3)  
(depends on RPT profile; may use draft-richer-token-introspection)

200: Give access to resource (3.1.2)

Requesting Party-AuthORIZING Party:
Make-Factual-Representations

Requesting Party-AuthORIZING Party:
Adhere-to-Terms
Spec call tree for the UMA profile of OAuth

UMA core

- OAuth 2
- OpenID Connect
- Token introspection
- OAuth resource set registration
- UMA binding obligations
- Dynamic client registration
- hostmeta

UMA native spec

Required external component

Optional external component

Individual IETF I-D