UMA for SDS

Eve Maler, Kantara UMA WG chair
tinyurl.com/umawg
1 Oct 2020
UMA Demo

USER MANAGED ACCESS (UMA)
UMA and Consent

**Consent** (and consent to contract) legally require **Manifestation**, **Knowledge**, and **Voluntariness** – more often honored in the breach.

Cookie consent
App permissions
Marketing preferences
Third-party permissions
ToS agreements

Digital consent has serious practical challenges achieving revocability, contract meeting of the minds, choice in relationship building, and consent seeker good faith.

**UMA** enables permissioning that is asynchronous:
- Share with parties, with groups, by relationship
- Respond to pending requests
- Monitor all current shares across sources
- Modify one or more shares
- (Respond to request at run time à la consent)

It is a technology that can enable **right-to-use licensing** within a Me2B framework of mutual agency and value exchange.

---

OAuth and UMA

“ALICE-TO-SELF” SHARING

OAuth enables constrained delegation of access to apps on request.

Alice can agree to app connections and also revoke them.

---

OAuth enables constrained delegation of access to apps on request.

Alice can agree to app connections and also revoke them.
OAuth and UMA

“ALICE-TO-BOB” SHARING

UMA adds control of cross-party sharing, letting Alice be absent when Bob uses a client to attempt access.

Alice controls trust between resource hosts and authorization services – enabling a wide ecosystem of resource hosts, so Alice can manage sharing across them.

UMA2 GRANT

UMA2 FEDAUTHZ

can be in different domains
UMA Technical and UMA BLT

Key:
- Lowercase = tech (specs)
- Uppercase = Biz/Legal
- = Permissions
- = Licenses

No trust required; “negative trust” is an option
**UMA and New Work**

**Policy Manager extension:** AS can delegate policy handling; RO can choose how to manage policy; RO can aggregate management across AS’s at one trusted place.

**Manage API extension (TBD):** RO can manage details of resource registration in an interoperable way.

**Resource definitions (extension? TBD):** RS can register API resource and scope templates for UMA clients to follow, to increase interop as well as extent of AS abilities to manage client communities of trust.

**Trusted claims (TBD):** AS delegates claims collection about RqP to other AS’s in an interoperable way, with predictable set math.
**P*P and (OAuth and) UMA**

**XACML (AND SIMILAR) ASSUMPTIONS**
- PEP “proxies” access request for requester (client) [2-3]
- Access response is yes/no answer vs. access token potentially introspected later [12]
- Policy language is standard vs. entitlements
- Extensive policy at-rest and in-motion handling therefore
- PEP trust in PDP is implied
- There is a single enterprise “resource owner”
- Subject is the implied “requesting party”

**OAUTH IMPLICATIONS**
- OAuth entitlement approach improves on cloud scale
- OAuth resource owner authorizes/denies (consents) at run time but enterprise can use XACML for access control

**UMA IMPLICATIONS**
- UMA AS/RS relationship is akin to PDP/PEP but trust is explicit, in the context of the RO
- Entitlement model and resource registration transfer more control to RS
- Explicit resource owner and requesting party roles standardize flexible access control without standardizing policy language
  (UMA2 token endpoint errors map to XACML responses)