UMA FAQ

UMA Frequently Asked Questions

These are questions we have fielded while giving UMA presentations and demos. If you want us to answer one of the empty questions you see appearing here, or have other questions, tweet us!

- **General Questions**
  - What is UMA?
  - What is a typical UMA scenario, and who are the actors in it?
  - What is the UMA protocol flow?
- **UMA's Relationship to Other Efforts**
  - What is UMA's relationship with Kantara and IETF?
  - What is UMA's relationship with the NSTIC Identity Ecosystem effort?
  - Can't you achieve UMA goals just by using OAuth?
  - Is UMA up to date with OAuth development?
  - Why doesn't UMA use OAuth terminology?
  - Does UMA make use of the JSON format or JSON Web Tokens (JWT)?
  - How is UMA related to OpenID and OpenID Connect?
  - How is UMA related to XACML?
- **Data Sharing, User Control, and Privacy Implications**
  - What is the relationship of UMA to the Privacy by Design principles?
  - How can UMA make requesting parties adhere to the user's wishes for privacy and data usage control?
  - Isn't Kantara just a conglomerate of big firms wanting to get hold of my data? Why should I trust them for "user-managed access"?
  - Social networking has made people too willing to share their data. Won't UMA make this worse? How do we get to truly controlled sharing?
  - What is required to make an UMA deployment model "legal" from a privacy, consent, and agency standpoint?
  - UMA seems positioned to be a positive force in managing privacy, but how will it overcome the challenges with Big Data?
- **Trust and Security Implications**
  - How can an UMA resource server be made responsible for incorrect or malicious behavior on its part? How does resource/authorization server trust work?
  - Does UMA come with a trust framework or an accreditation system?
- **UMA Usage in the Real World**
  - What implementations of UMA are there?
  - Have there been any usability studies on UMA?
  - Why externalize authorization?
  - What sort of entity might serve as an UMA authorization server?

---

**General Questions**

**What is UMA?**

User-Managed Access (UMA, pronounced "OOH-mah" like the given name) is an OAuth-based protocol designed to give a web user a unified control point for authorizing who and what can get access to their online personal data (such as identity attributes), content (such as photos), and services (such as viewing and creating status updates), no matter where all those things live on the web.

UMA allows a user to make demands of the requesting side in order to test their suitability for receiving authorization. These demands can include requests for information (such as "Who are you?" or "Are you over 18?") and promises (such as "Do you agree to these non-disclosure terms?" or "Can you confirm that your privacy and data portability policies match my requirements?").

UMA has enterprise implications as well as "user-centric" implications. At least one company has begun using it for coordinating the protection of enterprise APIs in much the way that today's Web Access Management (WAM) systems protect corporate web apps. As well, since it is a system for distributing authorization responsibilities, UMA has contractual and legal implications.

UMA has the following actors and basic architecture, with entities that closely align with core OAuth entities:
A number of historical articles and other materials about UMA are available:

- A comprehensive technical report published under the auspices of Newcastle University called User-Managed Access to Web Resources (also available on ncl.ac.uk site) explains the requirements that drive UMA, analyzes the design features that respond to these requirements, and reviews related work.
- A ReadWriteWeb article, Identity Management and Networks: The Enterprise Considers the Social Way from 23 Sep 2010, discusses UMA's potential impact.
- Group chair Eve Maler has written about UMA and its predecessor, ProtectServe, [here](#).
- Some older [historical materials](#) (may be out of date) explain the original thinking behind UMA and its predecessor, ProtectServe, and a [poster](#) (b) printed on A0-A3 paper; 8.5x11 or 8.5x14 is okay but small) was presented at the IEEE Security and Privacy symposium poster session.

For UMA information in other languages, see:

- Domenico Catalano's UMA introduction in [Italian](#)
- Cordny Nederkoorn's article on UMA in a [Dutch publication](#)
- Tatsuo Kudo's [SlideShare deck](#) covering UMA in Japanese
- Wikipedia information in [Italian](#) and [Spanish](#), thanks to Riccardo Abeti and Domenico Catalano

For external information and thoughts on UMA, see:

- Wikipedia entry for "User-Managed Access" (English)
- WholeChainCom blog entry on selective sharing
- UnboundID blog entry on attribute management and white paper (registration required) on the "identity economy"
- Phil Windley white paper in the Live Web series: From Personal Computers to Personal Clouds
- Oliver Pfaff's "New Trends in Web Security" [SlideShare](#)

UMA-related short links:

- [tinyurl.com/umawg](#): wiki home page
- [tinyurl.com/umafaq](#): this FAQ
- [tinyurl.com/umav1](#): UMA Core home page (with list of breaking/notable changes in drafts)
- [tinyurl.com/umatrust](#): Binding Obligations home page
- [tinyurl.com/umacore](#): latest Core spec
- [tinyurl.com/oauthr3](#): latest Resource Set Registration spec
- [tinyurl.com/umacase](#): UMA case studies page
- [tinyurl.com/umam20](#): Access Management 2.0 case study
- [tinyurl.com/umaiot](#): industrial IoT case study
- [tinyurl.com/uma1op](#): interop home page

Further reading:

- UMA Case Studies
- Latest specification of the UMA profile of OAuth
What is a typical UMA scenario, and who are the actors in it?

Let's use the example of Alice, a typical web user, to introduce UMA terms and concepts. Alice is a "resource owner" who manages her calendar resource online. She might want to share her calendar information with a number of different parties for a variety of purposes, while not making it fully public to the whole world.

The calendar is known as a "protected resource", and Alice manages it at a web application called a "resource server". She could have many resource servers for many different kinds of content she creates, along with other data about herself. In some cases, such as with credit scores, she can't actually control the values of data about herself.

The parties Alice wants to share her calendar with are called "requesting parties". They use "client" applications to attempt access.

In some cases, Alice herself is a requesting party in addition to being a resource owner, such as when she wants to share calendar data out to different calendar applications that she likes to use. We call this "person-to-self" sharing.

In other cases, she wants to allow her friend Bob and her various family members to get calendar access. We call this "person-to-person" sharing.

Finally, she may want to allow non-individuals, such as e-commerce companies and government agencies, to get access. We call this "person-to-organization" sharing.

With UMA, Alice can manage all these types of sharing in a unified way, from a single web-application point of control called an "authorization server". She can set policies that guide the authorization server in allowing or disallowing access by clients to protected resources at resource servers.

Further reading:
- UMA Case Studies
- Latest specification of the UMA profile of OAuth

What is the UMA protocol flow?

Phase 1 of the UMA core protocol involves the resource owner introducing the resource server and authorization server so they can work together. Phases 2 and 3 together involve the requesting party, using a client, making an access attempt, being tested for suitability by the authorization server to receive permission, and ultimately succeeding or failing in the attempt by presenting a token with permissions associated with it.

UMA’s Relationship to Other Efforts

What is UMA’s relationship with Kantara and IETF?

The specifications related to the UMA web protocol are being incubated in the Kantara Initiative, with the intent to contribute the draft work to the IETF. UMA specification draft modules have variously been contributed as IETF individual Internet-Drafts. One such draft so far, covering dynamic client registration, was accepted as an OAuth WG work item, an item that has now progressed.

Further reading:
- Kantara Initiative UMA WG charter
- IETF I-D on UMA profile of OAuth (may not be perfectly up to date compared to internal drafts)
- IETF I-D on Resource Set Registration
- OAuth WG status page

What is UMA’s relationship with the NSTIC Identity Ecosystem effort?

UMA can solve a variety of different access management problems. One of the key problems it was designed to solve was "user-centric" control of access to personal data. A use case was submitted to the NSTIC Identity Ecosystem Steering Group that relates closely to UMA: Delegated Authentication for User Managed Access. The use case is proceeding through the IDESG lifecycle.

Further reading:
- NSTIC Identity Ecosystem Steering Group
- IDESG use case: Delegated Authentication for User Managed Access

Can't you achieve UMA goals just by using OAuth?

In fact, UMA is built on top of OAuth. But typical profiles of OAuth solve a somewhat simpler problem. Here are some features UMA adds to the picture.

OAuth in its typical deployment models solves for person-to-self sharing (that is, Alice is the person using both the client app and the resource server app). UMA, in addition, solves for secure person-to-person sharing and person-to-organization sharing.

OAuth allows for pairwise app-to-app connections. UMA, in addition, defines a hub from which many pairwise sharing connections can be managed, controlled, and revoked.
OAuth authorizes sharing connections mostly based on simple authentication by the requesting side (client). UMA allows a user to craft policies that drive fine-grained claims-based authorization decisions, including getting the requesting party to make promises or have a third party make attestations on its behalf.

OAuth leaves unstated how its "authorization server" and "resource server" components interact (though a recent proposal for "token introspection" is tackling part of this problem). UMA fully defines a standard interface between its enhanced versions of these two components.

Further reading:
* Venn quick-reference slides describing the relationships among OAuth, OpenID Connect, and UMA
* Latest specification of the UMA profile of OAuth

Is UMA up to date with OAuth development?

Yes. The UMA protocol has been defined on top of the RFC version of OAuth 2.0. It now also references the OAuth dynamic client registration specification and a proposed OAuth token introspection specification, and it has defined a modularized proposal for resource set registration that we believe is valuable for OAuth generally and for OpenID Connect usage, as well as UMA usage.

Further reading:
* IETF RFCs 6749 and 6750

Why doesn't UMA use OAuth terminology?

As of late 2012, it does! Some of the old UMA collateral still uses UMA-specific terminology (authorization manager (AM), host, and requester, and prior to that, relationship manager, service provider (SP), and consumer), but we have generally fully moved to a world of authorization servers, resource servers, and clients.

Further reading:
* Latest specification of the UMA profile of OAuth (see the terminology section)
* OAuth's RFCs: 6749 and 6750
* UMA's binding obligations specification (see the terminology discussions)
* UMA historical materials

Does UMA make use of the JSON format or JSON Web Tokens (JWT)?

UMA uses the JSON format heavily in its request and response messaging specifications.

The default, mandatory-to-implement token format for UMA "requesting party tokens" (RPTs, the token that a client presents to a resource server when trying to access a protected resource) is opaque on the wire, and a resource server introspects it at the authorization server at run time. Its format is JWT, with an extension property called "permissions" that takes into account UMA's extended concept of resource set scopes.

Further reading:
* UMA 1.0 Core Protocol
* Draft IETF I-D for token introspection
* Compendium of JWT specification links

How is UMA related to OpenID and OpenID Connect?

UMA's purpose is to ensure that users can control access to their data and content. It does this by allowing a user to set up policies that specify who can get access. UMA has a design principle not to require usage of any one identifier system, such as OpenID identifiers. However, we also recognize that access authorization policies often need to identify other parties, for example, in access control lists (ACLs). UMA thus needs a way to handle identity claims about the requesting party, and for this purpose it integrates the option of using OpenID Connect (which, like UMA, is OAuth-based).

OpenID Connect specifies ways to retrieve claims that identify someone uniquely (for example, with a well-known globally unique identifier) or non-uniquely (such as providing a birth date). UMA policies can dictate requirements for the values of such claims in order for the authorization server to give the requesting party authorization for access. To achieve this, the UMA authorization server can act as an OpenID Connect Relying Party.

Further reading:
* Venn quick-reference slides describing the relationships among OAuth, OpenID Connect, and UMA
* UMA Requirements
* UMA's binding obligations specification (see the terminology discussions)
* OpenID Connect specifications
* UMA and Personal Clouds webinar on 19 June 2014 (slides, SlideShare, recording)

How is UMA related to XACML?
UMA is not formally related to XACML, but we can imagine some patterns of usage that bridge XACML and UMA. For example, UMA does not standardize a policy expression format or its evaluation, and treats an authorization server as a conflated policy decision point (or at least authoritative authorization data source), policy administration point, and policy information point for the purposes of UMA's in-band flows. An authorization server could also provide "raw" authorization data for which a resource server could then seek interpretation at a true XACML PDP. An UMA representative made a presentation to the XACML TC on 19 October 2012 to discuss liaison and technical opportunities. A specialized UMA token profile could also be used to provide a pattern for XACML's ongoing efforts to simplify/RESTify the current XACML standard.

Further reading:
- UMA and XACML slide deck
- Discussion on the XACML TC list
- Proposal for REST profile of XACML

Data Sharing, User Control, and Privacy Implications

What is the relationship of UMA to the Privacy by Design principles?

It is useful to examine how User-Managed Access (UMA) relates to the seven foundational principles of Privacy by Design (PbD). The group has prepared an analysis of each principle, and a companion analysis of how UMA and UMA-enabled solutions can support the principle.

Further reading:
- Privacy by Design website
- Privacy by Design Implications of UMA paper

How can UMA make requesting parties adhere to the user's wishes for privacy and data usage control?

The demands the resource owner may want to make on the requesting side can't be solved only with cryptography and web protocols; you quickly get into the realm of agreements and liability.

UMA stays away from techniques like DRM (digital rights management), where you cryptographically "lock" your data before sending it on (though this could be layered on the data that flows over UMA). For end-user convenience, development simplicity, and web-wide adoption, instead UMA puts the emphasis on user visibility into and control over access by others.

UMA is shooting for a reasonable minimum level of enforceability of authorization agreements, so that if the requesting side goes against your express wishes -- wishes they promised to adhere to -- then you have a meaningful chance of taking them to court over it.

Further reading:
- UMA Requirements
- UMA Binding Obligations framework
- W3C workshop position paper on Controlling Data Usage with UMA

Isn't Kantara just a conglomerate of big firms wanting to get hold of my data? Why should I trust them for "user-managed access"?

It costs no money to participate in Kantara work groups and discussion groups, and typically participation in each group involves a mix of large companies, small companies, government agencies, and individuals. Even among paid members, there is a diversity of individuals and organizations from around the world.

The UMA Work Group conducts various liaison activities with other Kantara groups, such as the Privacy and Public Policy Work Group (P3WG), and also non-Kantara groups, such as the OAuth group in the IETF.

The "UMAnitarians" hail variously from South America, North America, Europe, the UK, Australia, and Japan.

Further reading:
- UMA Participant Roster
- Kantara membership

Social networking has made people too willing to share their data. Won't UMA make this worse? How do we get to truly controlled sharing?

These are the reasons UMA was developed. UMA strives to give individuals control -- and also convenience. For example, it should be possible for someone to easily share financial, health, and other household-related information with the same set of trusted family members, even if the information comes from ten different sources. Today such sharing is piecemeal and usually insecure.

Central to UMA is the notion of an access management console -- a "digital footprint dashboard", if you will. The user can set up sharing rights between parties from this console, which is the user interface of the centralized authorization server component of UMA.

Because individuals are often at a power disadvantage compared to big companies and big websites, truly empowering people to control sharing is a tall order. The UMA Work Group has considered the problems it can tackle, and the ways in which a number of complementary efforts -- such as trust and privacy frameworks -- can be used in combination with it.
What is required to make an UMA deployment model "legal" from a privacy, consent, and agency standpoint?

The embedding of user-dictated policy in UMA makes it an explicit "carrier" for user permission, even if consent is not gathered in real time. Taking into account implementation and UX choices, trust model considerations, and various strengths of "assurance" in any one deployed system, further profiling of UMA may be warranted to reduce unwanted variability.

For example, the core UMA spec leverages OAuth for the resource owner's introduction of the resource server to the authorization server. The core spec allows for both explicit (e.g., authorization code) and implicit (e.g., SAML assertion) forms of user consent for this connection. Profiling may be warranted to require only explicit methods, and even to dictate user experiences for consent and authorization.

As well, wherever various outside-of-UMA terms of service or strong authentication needs might come into play, such as in the authorizing user's mutual agreements forged with authorization or resource server sites, or the strength of "assurance" about a requesting party's claims, profiling may be warranted to require certain terms of service or authentication or verification strengths, or alternatively membership in various trust frameworks that will dictate such answers.

Finally, where interoperability within an ecosystem will demand that certain types of policies about certain claim types must be available, it may be wise to define a mandatory-to-support set of claims and claim assurance strengths.

It should be noted that the UMA Work Group has a proposed a set of minimum contractual obligations that participants in an UMA-enable flow would take on. This set of obligations is by no means complete in terms of what specific parties and deployments would expect, but it attempts to provide a kind of "bedrock of enforceability" behind acts of distributed authorization, so that if something does go wrong, the correct party can be identified and remedial actions can be pursued.

Further reading:
- UMA Binding Obligations framework
- W3C workshop position paper on Controlling Data Usage with UMA

UMA seems positioned to be a positive force in managing privacy, but how will it overcome the challenges with Big Data?

Where personal data has been flowing unfettered in the back channel, entirely unmanaged by the people to whom the data applies, we anticipate that UMA will initially have little effect. It would take some other large force, such as data protection regulation or social/economic pressure around privacy, for that flow to slow or cease and to be pumped through an alternative flow that adds the individual into the mix somehow, such as UMA. UMA is likely to have the biggest near-term effect when it comes to new "data products" that people were formerly unwilling to share because they couldn’t throttle the sharing effectively, or that were formerly unavailable in a machine-readable way but now are accessible through documented web APIs.

Trust and Security Implications

How can an UMA resource server be made responsible for incorrect or malicious behavior on its part?
How does resource/authorization server trust work?

UMA requires a trust "dance" among several parties, who are aligned in some of their interests but who have divergent incentives in many other cases. The UMA Binding Obligations framework explains to what extent trust can be built between pairs of parties at a contractual level.

Further reading:
- UMA and Personal Clouds webinar on 19 June 2014 (slides, SlideShare, recording)
- FAQ: What is required to make an UMA deployment model "legal" from a privacy, consent, and agency standpoint?
- UMA Binding Obligations framework
- Blog post: UMA: Trust in a distributed authorization system

Does UMA come with a trust framework or an accreditation system?

At its current state of development, UMA comes with a draft technical specification and a draft contractual framework, called Binding Obligations, that lays out a set of minimum obligations to help make UMA participants confident about engaging in acts of distributed authorization. This is not a full trust framework in the U.S. FICAM sense, for example, but is meant to complement contracts and umbrella agreements wherever they might be used. The Binding Obligations document currently assumes that those who operate UMA-enabled software services will self-identify as conforming to the technical spec, and by that act, would take on obligations during the normal course of sending and receiving UMA-conforming messages. This is an area the UMA Work Group is actively working on.

Further reading:
- UMA and Personal Clouds webinar on 19 June 2014 (slides, SlideShare, recording)
FAQ: What is required to make an UMA deployment model "legal" from a privacy, consent, and agency standpoint?
- UMA Binding Obligations framework
- Blog post: UMA: Trust in a distributed authorization system

UMA Usage in the Real World

What implementations of UMA are there?
We are aware of several major implementations.

Further reading:
- Implementations page
- UMA and Personal Clouds webinar on 19 June 2014 (slides, SlideShare, recording)
- SMARTAM implementation FAQ
- Fraunhofer AISEC implementation FAQ
- OpenUMA project page

Have there been any usability studies on UMA?
The original SMART project performed formal usability studies at Newcastle University, which led to improvements from SMARTAM 1.0 to SMARTAM 2.0. We anticipate that further usability testing will be done by various parties as the number of interoperable implementations grows.

Further reading:
- UMA and Personal Clouds webinar on 19 June 2014 (slides, SlideShare, recording)
- SMARTAM implementation FAQ

Why externalize authorization?
For a long time, many large enterprises have recognized the value of externalizing and centralizing authorization in securing corporate resources, as exemplified in the "policy decision point vs. policy enforcement point" architecture that is common in access control systems. It allows applications to protect resources not by having to implement extensive authentication and authorization policy infrastructure themselves, but by outsourcing it to a trusted access control "home base". This makes applications simpler, allows them to be more loosely coupled to authentication methods (allowing for stronger authentication based on policy requirements), and more dynamic access decision-making.

For individual people interacting with the web, this model means that they can conveniently reuse any sharing circles (such as "friends and family" lists) they might have among all the Web applications and information they control. They can also set up criteria for access at a single place (the UMA authorization manager) and then go about their lives, not necessarily having to be online and paying attention when someone wants access to their stuff.

Further reading:
- UMA for the Enterprise webinar on 20 March 2014 (slides, SlideShare, recording)

What sort of entity might serve as an UMA authorization server?
An UMA authorization server has a natural affinity with those who operate identity providers (IdPs). It's well known that any one IdP can't provide on-board storage of all information relevant to an individual, and this fact is now being recognized in the new OpenID Connect model of IdPs that mediate access to claims from other authorities. The Open Identity Exchange's Street Identity project shows the value of using an IdP as a "relationship manager" to gather user consent for sharing a claim (in this case, a verified street address) whose authoritative version resides elsewhere. UMA leverages OpenID Connect to enable IdPs to offer this kind of service to users, adding features unique to UMA along the way (such as sharing data with other people and organizations, not just "yourself" using a different app).

Further reading:
- UMA and Personal Clouds webinar on 19 June 2014 (slides, SlideShare, recording)
- OpenID Connect specifications
- Street Identity project
- Google Tech Talk video demonstrating OpenID Connect-enabled UMA flows