UMA Trust Model

Abstract

This document defines the expectations and responsibilities of various parties interoperating in the User-Managed Access (UMA) context. The overall goal for UMA's trust model is to support legal enforceability of any agreements made between authorizing users and requesting parties in the granting of access authorization. This document's audience includes technologists, legal professionals, and operators of UMA-conforming services.

Status

This document is a product of the User-Managed Access Work Group. It is currently under active development. Its latest version can always be found here. See the Change History at the end of this document for its revision number.

Editors

- Susan Morrow
- Eve Maler

 Intellectual Property Notice

The User-Managed Access Work Group operates under Kantara IPR Policy - Option Patent & Copyright: Reciprocal Royalty Free with Opt-Out to Reasonable And Non discriminatory (RAND) and the publication of this document is governed by the policies outlined in this option.

Table of Contents

- 1 UMA Trust Model
  - 1.1.1 Table of Contents
- 2 Introduction
  - 2.1 Sharing Constellations
  - 2.2 Trust Relationships
    - 2.2.1 Trust Relationships Outside of UMA
    - 2.2.2 Trust Relationships at the Heart of UMA
    - 2.2.3 Trust Relationships Formed by Introducing the Host and AM
    - 2.2.4 Trust Relationships Involving the Protection API
    - 2.2.5 Trust Relationships Involving the Authorization API
- 3 Change History

Introduction

UMA is a Web protocol. As such, it describes a technical "contract" for web-based interactions – standardized request and response messages using standardized data formats – among software entities. The entities fill various roles in order to achieve "user-managed access" to Web resources. The following diagram illustrates the high-level goal of UMA.
The following diagram illustrates the high-level architecture UMA uses to achieve its goal.

Software entities participating in a protocol are known as **endpoints**. The UMA endpoints are:

- **Authorizing user** – the "user" in User-Managed Access
  - NOTE: "User" is often used informally in the UMA spec, where what is really meant is the browser (or other client software application) being operated by this person
Authorization manager (or AM)  
Host (of "protected resources")  
Requester

Software is just a tool; it can’t be held legally responsible for its own actions. Because UMA has a goal of setting legally enforceable conditions for access to resources, we must acknowledge the importance of the natural persons (human beings) and legal persons (such as companies) that run, control, own, contract to use, etc. UMA-conforming software. These are parties (distinguishing them from protocol endpoints). The UMA parties are:

- **Authorizing user** – the person seeking to protect resources stored at the host  
  - NOTE: This could mean a legal person, but Version 1 of UMA focuses primarily on natural persons acting on their own behalf on the Web
- **Host operator** – the natural or legal person responsible for the running host service  
- **Authorization manager operator** (or AM operator) – the natural or legal person responsible for the running AM service  
- **Requesting party operator** – the natural or legal person seeking access to protected resources

---

**Sharing Constellations**

While the essential end-to-end relationship being managed in UMA is the one between the authorizing user and the requesting party, there are subtle variations around the nature of the latter and the nature of the software they might use. Each type of requesting party and interaction style defines a different sharing constellation. Following are examples of each.

**Person-to-self sharing**

This describes most types of today’s OAuth-mediated access, for example, when Alice introduces the Klout service to her Twitter account. She uses both services herself, and wants them to communicate together on her behalf.

**Person-to-person sharing**

Today, many Web 2.0 sites offer some level of this control, but methods, strengths, and interfaces are inconsistent between sites and we’re not able to reuse policies across sites. For example, Alice can share Flickr photos with Bob by adding him to her Flickr “friends” or “family” list or by mailing him a special link to a photo album. Or Alice can add Bob as a “friend” on Facebook.

**Person-to-organization sharing mediated by a human agent of the requesting side**

For example, Alice wants to give her dentist’s office, DentalCare, temporary access to her calendar, to make it easier to schedule a series of root canal appointments. Carl, the office assistant, might be the actual person acting on behalf of the dental practice who sees Alice’s calendar.

**Person-to-organization sharing mediated by a automated web service on the requesting side**

For example, Alice has crafted a “personal request for proposals” (known as a pRFP among the Vendor Relationship Management community) because she’s in the market for a new car, and she’s willing to let car dealerships in her region of the country see the RFP and make her offers. Different car dealerships might use Web crawler services to go out and collect such RFPs, and these services will have to prove in automated fashion that they legitimately represent the right kind of business.

Each constellation’s requirements for successful user-managed access may be distinct. For example, in person-to-self sharing, it’s unlikely Alice will want to impose stringent contract terms on herself. In person-to-organization sharing, if the organization is using a web service client, it makes no sense to present a browser-interaction interface to it. And if the organization is acting through a human agent such as a receptionist or administrator, that person may need to prove they are acting on behalf of the organization.

---

**Trust Relationships**

To trust another party means to expect it to live up to agreed-on responsibilities. If something goes wrong, it must be possible to assign responsibility accurately. For example, if Alice sets policies that she thinks will prevent Zeke from seeing her calendar, and Zeke gets access somehow through some requester software, who exactly was at fault? Since UMA imposes a loose coupling between the setting of policy, the application of policy in assigning authorizations, and the providing of resources, a lot could go wrong. The act of authorization has to be made meaningful through making different parties’ actions enforceable and auditable.

The parties in the UMA picture can potentially have the following expectations of each other. Note that expectations can flow in both directions (though a pair of such expectations is likely not reciprocal):

---
The UMA protocol can only define junctures where entities are required to uphold expectations of technical trust by other entities. For example, when a host sends a request message X to an AM, and the AM replies with response message Y, each of their sets of expectations is defined by the relevant normative interactions provided by the UMA specification. Technical trust is measured in terms of technical conformance (UMA 7). Note that this section may therefore require protocol knowledge on the part of the reader; specific specification references are provided throughout to help with this process.

This Trust Model adds elements of business trust to this picture, defining a behavior standard the parties are required to live up to in order to meet each others' expectations. These elements of business trust build on the exact same interactions defined by the technical spec, so that a technical action by an entity implies that the party operating the entity intends to live up to the matching set of business expectations. These constitute "general rules" (in the contractual or legal sense) that apply to all parties that deploy entities claiming to be UMA-conforming. A business trust expectation is referred to here as a trust relationship.

NOTE: UMA's trust relationships do not have the force of legal liability unless parties explicitly include them in contracts, or unless laws or regulations mandate them.

Following is a summary of the TRs defined by this Trust Model.
Trust Relationships Outside of UMA

Parties are free to define or adhere to additional "special rules". Following are typical pairwise agreements the parties might form between themselves.
Note that UMA neither anticipates nor requires any outside trust relationship between the authorizing user and the requesting party. This is because UMA’s central purpose is to enable this trust relationship itself.

Following are some examples of pairwise terms of service that parties might execute:

- Host operator relies on authorizing user to adhere to the host operator’s TOS. This TR typically gets formed when the user initially registers for an account at the service.
- AM operator relies on operator of requester web app, apart from any requesting party’s usage of it, to adhere to the AM operator’s TOS for API clients. This TR typically gets formed when the client app developer registers for client credentials, or through “API-wrap” TOS (akin to “browse-wrap” that binds a user who merely visits a website).
- Requesting party relies on employee to act as its legitimate agent. The authorizing user’s policy may require the requesting party to prove that the request is being mediated by a legitimate agent for that party. However, the requesting party itself is the one who may impose constraints on its workforce around keeping information learned in the course of business confidential etc.
- Requesting party relies on operator of requester web service to adhere to any contractual agreement governing that service. For example, a car dealership may contract out to use a cloud service that crawls the Web looking for personal RFPs that meet the dealership’s criteria.

Another way of creating rules that go beyond UMA is for parties to join trust frameworks that impose umbrella agreements on their members. Such agreements might contain more specific versions of UMA’s trust relationships, or additional trust relationships among the parties, or might include requirements to use specially profiled technical features of UMA, or impose user experience requirements on parties interacting with human beings – or all of the above. This is where levels of identity or attribute assurance, levels of data protection, and levels of data usage control might come in.

**Trust Relationships at the Heart of UMA**

This is the set of trust relationships that describes the overall purpose of UMA. The authorizing user works through the AM as a proxy for authorization management.

<table>
<thead>
<tr>
<th>TR ID</th>
<th>Expecting party</th>
<th>Responsible party</th>
<th>Contextual parties</th>
<th>Expected behavior of responsible party</th>
<th>When TR is formed</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Authorizing user</td>
<td>Relying party operator</td>
<td>AM operator, host operator</td>
<td>Adhere to promises made in order to get access authorization granted.</td>
<td>When the requesting party successfully receives access to a protected resource by wielding a valid requester permission token with a currently valid permission for the type of access sought (UMA 3.1.3).</td>
<td>Previously, the requesting party asked for the permission from the AM and may have had to provide claims asserting willingness to adhere to data usage constraints imposed by the user. This is precisely the end-to-end access authorization agreement that UMA exists to forge. Accepting access to the protected resource binds the requesting party to the terms it agreed to.</td>
</tr>
</tbody>
</table>
Trust Relationships Formed by Introducing the Host and AM

This is the set of trust relationships formed in UMA phase 1, when the user introduces a host and AM to work together in protecting resources. The user will likely introduce many hosts to the same AM. In some cases, particular hosts may make it possible to protect some resources with one AM and other resources with a different AM. Many of these trust relationships depend on the authorizing user's expression of various preferences and instructions through a user interface, and to this extent, they have a subjective component.

<table>
<thead>
<tr>
<th>TR ID</th>
<th>Expecting party</th>
<th>Responsible party</th>
<th>Contextual parties</th>
<th>Expected behavior of responsible party</th>
<th>When TR is formed</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Authorizing user</td>
<td>AM operator</td>
<td>Host operator</td>
<td>Work with this host operator in protecting the user's resources hosted here.</td>
<td>When the user authorizes the AM to issue a protection API token (UMA 2.3).</td>
<td>Later, the AM will require the host to present the PAT whenever it uses the AM’s protection API on behalf of this user. The user can break this trust relationship by revoking the PAT.</td>
</tr>
<tr>
<td>4</td>
<td>AM operator</td>
<td>Authorizing user</td>
<td>Host operator</td>
<td>Introduce the desired host operator to this AM operator in outsourcing protection of this host's resources.</td>
<td>When the user authorizes the AM to issue a protection API token (UMA 2.3).</td>
<td>How the host collected the AM’s location is out of band for UMA; it is the user's responsibility to check that the user has been redirected to an acceptable AM before authorizing the connection. The user can break this trust relationship by revoking the PAT.</td>
</tr>
<tr>
<td>5</td>
<td>Authorizing user</td>
<td>Host operator</td>
<td>AM operator</td>
<td>Participate in the outsourcing of authorization for protected resources and respect the permissions generated by the AM.</td>
<td>When the user authorizes the AM to issue a protection API token (UMA 2.3).</td>
<td>Once the AM operator becomes the user’s authorization proxy, it begins relying on the host operator in other, more specific ways. The host has the opportunity to inspect permissions in Phase 2, but its responsibility for respecting them begins now. The user can break this trust relationship by revoking the PAT.</td>
</tr>
<tr>
<td>6</td>
<td>Host operator</td>
<td>Authorizing user</td>
<td>AM operator</td>
<td>Introduce the host to the desired AM operator.</td>
<td>When the user authorizes the AM to issue a protection API token (UMA 2.3).</td>
<td>Once the AM operator becomes the user’s authorization proxy, the host operator begins relying on it in other, more specific ways. How the user indicated the desired AM to the host is out of band for UMA; it is the user’s responsibility to check that the user has been redirected to an acceptable AM before authorizing the connection. The user can break this trust relationship by revoking the PAT.</td>
</tr>
<tr>
<td>7</td>
<td>AM operator</td>
<td>Host operator</td>
<td>Authorizing user</td>
<td>a) Register resource sets and applicable actions accurately and timely according to the user's wishes for protection/selective sharing.</td>
<td>a) When the user authorizes the AM to issue a protection API token (UMA 2.3).</td>
<td>The host has the opportunity to register resource sets later in Phase 1, but its responsibility for performing this task begins now. The user can break this trust relationship by revoking the PAT.</td>
</tr>
</tbody>
</table>

Trust Relationships Involving the Protection API
This is the set of trust relationships that allow the protection outsourcing to take place. The host operator and AM operator have two contexts for dealing with each other: one is specific to the authorizing user (the same two operators could participate in protection for different users) and the other is specific to the requesting party (the same two operators could be protecting the same resource on behalf of the same user but fielding requests from different requesting parties). Many of these trust relationships depend on the authorizing user's expression of various preferences and instructions through a user interface, and to this extent, they have a subjective component.

<table>
<thead>
<tr>
<th>TR ID</th>
<th>Expecting party</th>
<th>Responsible party</th>
<th>Contextual parties</th>
<th>Expected behavior of responsible party</th>
<th>When TR is formed</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>Host operator</td>
<td>AM operator</td>
<td>Requesting party</td>
<td>Provide accurate requester permission token status information, including active permissions.</td>
<td>When the host checks the RPT status (UMA 3.3).</td>
<td>This TR involves only UMA token profiles that are defined as part of the UMA spec. Third-party profiles must be covered in externally defined TRs (“special rules”).</td>
</tr>
<tr>
<td>9</td>
<td>Host operator</td>
<td>AM operator</td>
<td>Authorizing user</td>
<td>Represent the user's authorization policies accurately and timely in issuing a permission.</td>
<td>When the host successfully registers a permission at the AM (UMA 3.2).</td>
<td>Later in Phase 2, when a requester approaches the AM seeking that permission, the AM matches user policies to the requested permission to drive any requests for claims and its ultimate authorization decision, but its responsibility for performing this task begins now.</td>
</tr>
<tr>
<td>10</td>
<td>AM operator</td>
<td>Host operator</td>
<td>Requesting party</td>
<td>Respect the status of permissions granted by the AM operator.</td>
<td>When the host checks the RPT status (UMA 3.3).</td>
<td>The host operator, as an autonomous party, carries a great deal of responsibility here not to grant access where the AM has not granted permission and to make every effort to grant access where the AM has granted permission. Its interpretation of scopes and permissions is not directly auditable by the AM. Further, issues can arise from the skew between permission validity periods and actual access. This is a recommended area of exploration for additional UMA token profiles that can effect higher levels of technical trust in order to rely less on business trust.</td>
</tr>
</tbody>
</table>

**Trust Relationships Involving the Authorization API**

This is the set of trust relationships that govern the authorization process. The host operator need not trust the requesting party at all, because it outsources the job entirely to the AM operator. These trust relationships reflect how the requesting party goes from untrusted to trusted for a particular scope of access, as far as the AM operator (acting as a proxy for the authorizing user) is concerned. Many of these trust relationships depend on the authorizing user's expression of various preferences and instructions through a user interface, or on the requesting party's actions, and to this extent, they have a subjective component.

<table>
<thead>
<tr>
<th>TR ID</th>
<th>Expecting party</th>
<th>Responsible party</th>
<th>Contextual parties</th>
<th>Expected behavior of responsible party</th>
<th>When TR is formed</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>AM operator</td>
<td>Requesting party</td>
<td>(none)</td>
<td>Truthfully supply any self-asserted claims required for access authorization at this AM.</td>
<td>When the AM issues an authorization API token (UMA 3.4.3).</td>
<td>Later in Phase 2, the requesting party may be asked to provide claims to support authorization processes at this AM, for accessing all resources protected by this AM, managed by any authorizing users. This is why this TR is formed outside of the context of any specific host operator or authorizing user. The requesting party's responsibility to provide truthful claims in all these cases begins now. The requesting party can revoke this relationship by revoking the AAT.</td>
</tr>
<tr>
<td>12</td>
<td>Requesting party</td>
<td>AM operator</td>
<td>(none)</td>
<td>Request only claims that support the purpose of satisfying the user's policy.</td>
<td>When the AM issues an authorization API token (UMA 3.4.3).</td>
<td>Later in Phase 2, the AM may ask the requesting party to provide claims for specific permission purposes, but its responsibility begins now. The requesting party can revoke this relationship by revoking the AAT.</td>
</tr>
<tr>
<td>AM operator</td>
<td>Requesting party</td>
<td>Host operator</td>
<td>Truthfully supply any claims required for access authorization for the requested permission.</td>
<td>When the requesting party provides claims to satisfy the AM's authorization process (UMA 3.5).</td>
<td>Where claims are supplied that can be verified by the AM, the risk imposed by this need for business trust can be reduced. UMA defines an “OpenID Connect claim profile” that provides one way to collect trusted claims from third-party claim providers.</td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>------------------</td>
<td>---------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
</tbody>
</table>

**Change History**

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current Version (v. 8)</td>
<td>Apr 15, 2012 15:10</td>
<td>Eve Maler</td>
</tr>
<tr>
<td>v. 7</td>
<td>Apr 15, 2012 15:09</td>
<td>Eve Maler</td>
</tr>
<tr>
<td>v. 6</td>
<td>Apr 15, 2012 15:07</td>
<td>Eve Maler</td>
</tr>
<tr>
<td>v. 5</td>
<td>Sep 25, 2011 22:00</td>
<td>Eve Maler: Huge revision incorporating known shape of the forthcoming solution for OpenID Connect integration to solve trusted claims.</td>
</tr>
<tr>
<td>v. 4</td>
<td>Sep 25, 2011 17:38</td>
<td>Eve Maler</td>
</tr>
<tr>
<td>v. 3</td>
<td>Sep 25, 2011 16:39</td>
<td>Eve Maler</td>
</tr>
<tr>
<td>v. 2</td>
<td>Feb 24, 2011 22:49</td>
<td>Eve Maler</td>
</tr>
<tr>
<td>v. 1</td>
<td>Feb 23, 2011 19:03</td>
<td>Eve Maler</td>
</tr>
</tbody>
</table>