Simple Access Authorization Claims

Abstract

This document uses the Claims 2.0 specification to define a small set of basic claims to be used in the process of User-Managed Access (UMA) access authorization.

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. This document is intended to be experimental, and it may be superseded at any time by a more formal or more comprehensive set.

Editors

- Eve Maler
- Paul C. Bryan

Table of Contents

Error rendering macro 'toc'

null

Introduction

This document uses the Claims 2.0 ([Claims20]) specification to define a small set of basic claims to be used in the process of User-Managed Access (UMA) access authorization. UMA uses claims in its process of negotiation for access authorization, in which an authorization manager can require a requester to convey claims on behalf of a requesting party in order to satisfy the polices of an authorizing user.

Terminology

scope: The particular protected resource(s) and method(s) of access being sought by the requester in approaching the authorization manager for an access token.

Claim Format: Requesting Party Policy

URL: http://c2.io/req-party-policy

This claim conveys a URL whose resource representation is a persistent record of the set of policies the requesting party promises to adhere to for the current scope of access. The policy may include privacy, data protection, purpose, data portability, copyright licensing of posted data, or any other relevant policy, and may be in a natural language or may be machine-readable.

Claims-Requested Form

The claims-requested form MAY contain a template requiring one or more issuer attributes. The value attribute for the claims-requested object contains an object as follows:

{

Name	Value	Description
`"policy"`	*URL* (MAY use wildcard and option conventions)	A template for a URL representing the policies to which the requesting party is promising to adhere. If a literal string is provided, requires the requesting party to promise to adhere to a specific set of policies.

}

The following example causes the authorization manager to request a claim from the requester that allows the requesting party to select whatever policy it wants, as long as a persistent URL can be provided for it:

```
{
  "http://c2.io/claims-requested": [
    {
      "type": "http://c2.io/req-party-policy",
      "value": {
        "policy": "*"
      }
    }
  ]
}
```

The following example causes the authorization manager to request a claim from the requester that requires the requesting party to agree to the authorizing user's chosen policy, by virtue of dictating the policy URL string:

```
{
  "http://c2.io/claims-requested": [
    {
      "type": "http://c2.io/req-party-policy",
      "value": {
        "policy": "http://creativecommons.org/licenses/by/2.0/deed.en"
      }
    }
  ]
}
```

Claims Form

The claims form MAY contain an issuer attribute (and may be required to contain one if the claims-requested form requires it). The value attribute for the claims object contains an object as follows:

{

Name	Value	Description
`"policy"`	*URL*	A URL representing the policies to which the requesting party is promising to adhere.

}

The following example represents a claim conveyed by the requester that indicates the requesting party promises to adhere to the policy persistently found at the supplied policy URL:

```
{
  "http://c2.io/claims": [
    {
      "type": "http://c2.io/req-party-policy",
      "value": {
        "policy": "http://creativecommons.org/licenses/by/2.0/deed.en"
      }
    }
  ]
}
```

Claim Format: Self-Asserted Requesting Party Identifier

URL: http://c2.io/self-id

This claim conveys a string that the requesting party wishes to use to identify itself/himself/herself to the authorization manager and authorizing user for the current scope of access. The string is self-asserted.

Note: Such a label may be useful in audit logs and in notifying authorizing users about, and requesting consent to, certain kinds of access requests in real time. Typically a self-asserted identifier is inappropriate for high-sensitivity access.

Claims-Requested Form

The claims-requested form MUST NOT contain a template requiring one or more issuer attributes. The value attribute for the claims-requested object contains an object as follows:

{

Name	Value	Description
`"self-id"`	*string* (MAY use wildcard and option conventions)	A template for a string representing an identifier the requesting party has chosen to use.

}

The following example causes the authorization manager to request a claim from the requester that requires the requesting party to provide a self-asserted identifier of its choice:

```
{
  "http://c2.io/claims-requested": [
    {
      "type": "http://c2.io/self-id",
      "value": {
        "self-id": "*"
      }
    }
  ]
}
```

Claims Form

The claims form MUST NOT contain an issuer attribute. The value attribute for the claims object contains an object as follows:

{

Name	Value	Description
`"self-id"`	*string*	A string representing an identifier the requesting party has chosen to use.

}

The following example represents a claim conveyed by the requester that provides the self-asserted identifier the requesting party has chosen to use:

```
{
  "http://c2.io/claims": [
    {
      "type": "http://c2.io/self-id",
      "value": {
        "self-id": "BelleCare Dental of Bellevue, WA, USA"
      }
    }
  ]
}
```

References

Change History

Version	Date	Comment
Current Version (v. 5)	**Apr 29, 2010 09:31**	**Paul C. Bryan**: Migration of unmigrated content due to installation of a new plugin
v. 4	Apr 29, 2010 09:31	**Paul C. Bryan**: Migrated to Confluence 4.0
v. 3	Apr 29, 2010 09:31	**Paul C. Bryan**
v. 2	Apr 29, 2010 00:21	**Eve Maler**
v. 1	Apr 29, 2010 00:20	**Eve Maler**