Claims 2.0

Abstract

This document defines a JSON-based format for expressing claims and requests for claims. The primary driver for Claims 2.0 is the process of negotiation for access authorization defined by the User-Managed Access (UMA) core protocol, but this document is defined as a modular building block that can be used by other protocols.

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

- Eve Maler
- Paul C. Bryan

Intellectual Property Notice

Table of Contents

Error rendering macro 'toc'

null

Introduction

This document defines a JSON-based format ([JSON]) for expressing claims and requests for claims, that is, statements in the sense of [IDCclaim]. The primary driver for Claims 2.0 is the process of negotiation for access authorization defined by the User-Managed Access (UMA) core protocol, 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. This document is defined as a modular building block that is intended to be used by other protocols.

The following are design goals of this specification:

- Allows for efficient, convenient, and consistent processing, marshalling, and unmarshalling
- Lightweight way to express both self-asserted claims and third-party-signed claims
- Allows for variability in the characteristics of claims being requested

Following are not goals of this specification:

- Complete replacement or equivalent for other claims and signature formats, such as SAML, IMI, or XML Signature
- Protocol for carrying or securing claims and requests for claims for any particular purpose

Terminology

claim: A statement in the sense of [IDCclaim].

claims document: A top-level JSON object containing one or more claims.

claims-requested document: A top-level JSON object containing a request for one or more claims, conforming to this specification. (Claims can be requested in ways that do not involve a claims-requested document.)

signed claim: A claim object that contains a signature attribute and related data.

Conventions

Following are conventions used in this specification for documenting constraints on claims-related JSON objects:

- @@Do we need a proper BNF, or will this do?
- **Type:** *Italic text* stands for a scalar value, array, or object of the indicated type. @@This one can't currently be used in actual Requested Claims object, unless we import JSON-schema or something. Do we want to go that far?

- **Options:** `"(`*`type`*`|`*`type`*`)"` represents a literal string with a series of two or more exclusive options of the shown type. When a literal parenthesis mark or vertical bar must appear in such a string, use @@what? instead.
- **Wildcards:** `"*"` or `"`*`string*string`*`"` represents a literal string that can have any non-null value at the point indicated by the asterisk. When a literal asterisk must appear in such a string, use @@what? instead. @@Slippery slope question: the claims-requested example below shows why it would be valuable to allow zero-or-more in addition to one-or-more. What to do?

Definition of Claims-Using Protocols and Specific Claim Formats

It is anticipated that any number of specific claim formats will be defined separately from this specification. A definer of a specific claim format MUST document the syntax (including a URL identifying the claim format) and semantics of both the form in which claims must be supplied and any corresponding form in which claims must be requested.

A higher-level protocol using this specification for claims exchange MUST document where claims documents appear in the protocol. Without further specification in a higher-level protocol, a requester of a claim is the sole judge of whether a claim supplied in response is acceptable for the purpose to which the requester intends to put it.

When these documents are exchanged by means of HTTP, their content type MUST be `application/json`.

Additional requirements on the definition of specific claim formats and higher-level claims exchange protocols appear below.

Claims-Related Documents

This specification defines a partial protocol made up of a JSON object that represents a request for claims, known as a claims-requested document, and a JSON object that represents claims, known as a claims document. It is OPTIONAL for a higher-level protocol to use an explicit claims-requested document, rather than an implicit, statically defined, or explicit but alternate-format request for claims.

This specification does not dictate how to interpret the subject of a claim; it is dependent on the particulars of the higher-level protocol in which claims are being exchanged. In some cases, the subject may be context-dependent, for example, it may be the claim supplier or an entity in a defined relationship with the claim supplier. @@Is this sufficient to support our needs at the UMA level?

claims-requested Document

{

Name	Value	Value Description
`"http://c2.io/claims-requested"`	Array of one or more *Requested Claim* objects	A set of requested claims.

}

For example:

```
{
"http://c2.io/claims-requested": [
    {...},
    {...}
  ]
}
```

Each array element is a *Requested Claim* object that represents a template for a corresponding claim being sought, where the template MAY use any or all of the order, type, option, and wildcard conventions defined by this specification to indicate constraints on the corresponding claim, along with any other conventions.

@@Need to express claims that are alternatives for each other? optional vs. required claims?

claims Document

{

Name	Value	Description
`"http://c2.io/claims"`	Array of one or more *Claim* objects	A set of claims.

}

For example:

```
{
  "http://c2.io/claims": [
    {...},
    {...}
  ]
}
```

Each array element is a *Claim* object that represents a claim being delivered on request.

Requested Claim Object

{

Name	Value	Description
`"type"`	`"`*`claimURL`*`"`	The identifying URL of the claim format being requested.

`"issuer"`
(optional)
	`"`*`URL`*`"`
	or
	`"(`*`URL1`*`|`*`URL2`*`)"`
	or
	`"*"`
		Constraint on the value of the issuer of a supplied claim. If absent, it is acceptable for the corresponding claim to supply no issuer. (The presence of an issuer value in a supplied claim has additional implications, as noted below.)

`"value"` (optional)	Object containing attributes that are specific to the claim type	The literal values of the attributes MAY use the **option** and/or **wildcard** conventions to indicate desired constraints on the claim.

}

For example:

```
{
  "http://c2.io/claims-requested": [
    {
      "type": "http://www.example.com/ABC",
      "value": {
        "param": 18
      }
    }
  ]
}
```

This example requests a claim conforming to the `http://www.example.com/claimsformats/ABC` format, whose value is an object that must contain at least an attribute named `param` that has a number value of `18`.

Claim Object

A supplier of a claim MUST construct it to conform to the following format, the constraints of the named claim type, and any constraints dictated by the claims-requested document that it is responding to.

{

Name	Value	Description
`"type"`	`"`*`claimURL`*`"`	The identifying URL of the format of the claim being supplied.
`"issuer"` (optional)	`"`*`URL`*`"`	The issuer of the claim. The issuer URL can be used by the recipient of the claim to retrieve the public key of the signer of a signed claim. The issuer MUST be present if a signature is present.
`"signature"` (optional)	*Signature* object	A signature over the entire *Claim* object (see the Signed Claims section, below). The signature MUST be present if an issuer is present.

`"value"` (optional)	Object containing attributes that are specific to the claim type	

}

For example:

```
{
  "http://c2.io/claims": [
    {
      "type": "http://www.example.com/ABC",
      "value": {
        "param": 18,
        "param2": "foo"
      }
    }
  ]
}
```

This example represents a claim supplied in response to the previous example, with a `param` attribute that conforms to the template and an additional attribute `param2`.

Signed Claims

Any *Claim* object can contain a signature attribute. The signer of the claim represents the issuer of the claim about the subject of the claim. A claim with a signature attribute MUST also have an issuer attribute, and vice versa.

The *Signature* object is as defined in [CouchSign]. @@Need to fully define, and extend, in this spec.

A claim recipient that wishes to verify the signature of a signed claim MAY use the issuer value to construct and dereference a well-known location for the signer's "host-meta" metadata (as defined in [hostmeta]) and then obtain the signer's public key from this location by retrieving the href value an XRD `<Link>` element with a `rel` attribute whose value is `http://c2.io/public-key`.

For example, if the signer value is "https://signer.example.com", the claim recipient can construct the well-known metadata location "https://signer.example.com/.well-known/host-meta" and perform an HTTP GET on that location to retrieve the XRD resource and obtain the public key.

References

Normative References

[JSON]
http://www.ietf.org/rfc/rfc4627.txt

[hostmeta]
http://tools.ietf.org/html/draft-hammer-hostmeta

[CouchSign]
http://wiki.apache.org/couchdb/SignedDocuments

Non-Normative References

[IDCclaim]
http://wiki.idcommons.net/Claim

Change History

Version	Date	Comment
Current Version (v. 8)	**Apr 28, 2010 16:37**	**Paul C. Bryan**: Migration of unmigrated content due to installation of a new plugin

v. 7	Apr 28, 2010 16:37	**Paul C. Bryan**: Migrated to Confluence 4.0
v. 6	Apr 28, 2010 16:37	**Paul C. Bryan**
v. 5	Apr 28, 2010 15:11	**Eve Maler**
v. 4	Apr 28, 2010 14:48	**Paul C. Bryan**
v. 3	Apr 28, 2010 14:43	**Paul C. Bryan**
v. 2	Apr 28, 2010 14:11	**Eve Maler**
v. 1	Apr 26, 2010 20:47	**Eve Maler**