Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: February 19, 2017

P. Hunt, Ed.
Oracle
W. Denniss
Google
M. Ansari
Cisco
M. Jones
Microsoft

August 18, 2016

Security Event Token (SET)
draft-hunt-idevent-token-03

Abstract

This specification defines the Security Event token, which may be distributed via a protocol such as HTTP. The Security Event Token (SET) specification profiles the JSON Web Token (JWT) and may be optionally signed and/or encrypted. A SET describes a statement of fact that may be shared by an event publisher with event subscribers.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on February 19, 2017.

Copyright Notice

Table of Contents

1. Introduction and Overview

This specification defines an extensible Security Event Token (SET) format which may be exchanged using protocols such as HTTP. The specification builds on the JSON Web Token (JWT) format [RFC7519] in order to provide a self-contained token that can be optionally signed using JSON Web Signature (JWS) [RFC7515] and/or encrypted using JSON Web Encryption (JWE) [RFC7516].

For the purpose of this specification, an event is a statement of fact by a publisher (also known as the event issuer) that the state of a security subject (e.g., a web resource, token, IP address) it controls or is aware of, has changed in some way (explicitly or implicitly). A security subject may be permanent (e.g., a user account) or temporary (e.g., a login session) in nature. A state change may include direct changes of entity state, implicit changes to state or other higher-level security statements such as:

- The creation, modification, removal of a resource.

- The resetting or suspension of an account.

- The revocation of a security token prior to its expiry.

- The logout of a user session. Or,

- A cumulative conclusion such as to indicate that a user has taken over an email identifier that may have been used in the past by another user.

Based on some agreed upon criteria for an event Feed, the publisher distributes events to the appropriate subscribers. While an event may be delivered via synchronous means (e.g., HTTP POST), the distribution of the event often happens asynchronously to the change of state which generated the security event. As an example, an OAuth2 Authorization Server [RFC6749], having received a token revocation request [RFC7009], may issue a token revocation event to downstream web resource providers. Having been informed of a token revocation, the OAuth2 web resource service provider may add the token identifier to its local revocation list assuming the token has not already expired.

A subscriber having received an event, validates and interprets the event and takes its own independent action, if any. For example, having been informed of a personal identifier now being associated with a different security subject (i.e., is being used by someone else), the subscriber may choose to ensure that the new user is not granted access to resources associated with the previous user. Or it may not have any relationship with the subject, and no action is taken.

While subscribers will often take actions upon receiving one or more events, events MUST NOT be assumed to be commands or requests. To do so requires complex bi-directional signals and error recovery mechanisms that fall outside the scope of this specification. The intent of this specification is to define a way of exchanging statements of fact that subscribers may interpret for their own purposes. Since events are typically historical statements by a publisher and are not commands, idempotency or lack thereof, does not apply.

Unless otherwise specified, this specification uses example events intended as non-normative examples showing how an event may be used. It is expected that other specifications will use this specification to define normative events.

This specification is scoped to security and identity related events. While event tokens may be used for other purposes, the specification only considers security and privacy concerns relevant to identity and personal information.

1.1. Notational Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119]. These keywords are capitalized when used to unambiguously specify requirements of the protocol or application features and behavior that affect the inter-operability and security of implementations. When these words are not capitalized, they are meant in their natural-language sense.

For purposes of readability, examples are not URL encoded. Implementers MUST percent encode URLs as described in Section 2.1 of [RFC3986].

Throughout this document, all figures MAY contain spaces and extra line-wrapping for readability and space limitations. Similarly, some URIs contained within examples have been shortened for space and readability reasons.

1.2. Definitions

The following definitions are used with SETs:

Feed Publisher
: The Feed Publisher creates SETs to be distributed to registered subscribers. In JWT terminology, the Feed Publisher is also known as the issuer ("iss").

Security Event Token (SET)
: An SET is a JWT that is to be distributed to one or more registered subscribers. A SET MAY be signed or encrypted using JWS and/or JWE for authentication and confidentiality reasons.

Feed
: A Feed is a logical grouping of SETs or a context under which SETs may be issued. A Subscriber registers with the Feed Publisher to subscribe to SETs associated with a Feed. How a Feed is defined or the method for subscription is out-of-scope of this specification.

Subscriber
: A Subscriber registers to receive SETs from a Feed Publisher using a protocol such as HTTP. The method of registration and delivery is out-of-scope of this specification.

Security Subject

A Security Subject is the entity to which a SET refers. A Security Subject may be a principle (e.g., Section 4.1.2 [RFC7519]), a web resource, or other thing such as an IP address that a SET might reference.

2. The Security Event Token (SET)

A SET conveys a statement (in the form of a JWT [RFC7519]) about a single security event in relation to a Security Subject that may be of interest to a Subscriber or set of Subscribers receiving SETs from a Feed Publisher.

The schema and structure of a SET follows the JWT [RFC7519] specification. A SET has the following characteristics:

- An outer JSON structure that acts as the event envelope. The envelope contains a set of attributes common to every SET. The attributes are used to validate the event and determine the event data included. The envelope includes an "events" attribute describing the type of event, and

- JSON [RFC7159] sub-objects, that act as event payload, that contain attributes associated with the event URIs values provided in the envelope "events" attribute.

- While a SET may have more than one URI value for "events", the intent is that the additional URIs are to provide additional attributes related to the same event in the form of extensions to the primary event.

SET payload objects are added to the envelope by adding an attribute to the top-level JSON object (the envelope) whose name corresponds to a value from "events". The payload object contains the attributes relevant to the specified event URI. For example, SET event payloads may include "iss" attribute to distinguish between the issuer of the event and the issuer of a Security Subject or "sub".

The following is a non-normative example showing a hypothetical SCIM password reset SET. The example also shows an example where the issuer has provided an extension ("https://example.com/scim/event/passwordResetExt") that is used to convey additional information such as the current count of reset attempts:

```
{
  "jti": "3d0c3cf797584bd193bd0fb1bd4e7d30",
  "events":[
    "urn:ietf:params:scim:event:passwordReset",
    "https://example.com/scim/event/passwordResetExt"
  ],
  "iat": 1458496025,
  "iss": "https://scim.example.com",
  "aud":[
    "https://jhub.example.com/Feeds/98d52461fa5bbc879593b7754",
    "https://jhub.example.com/Feeds/5d7604516b1d08641d7676ee7"
  ],
  "urn:ietf:params:scim:event:passwordReset":{
    "iss":"https://scim.example.com",
    "id":"44f6142df96bd6ab61e7521d9",
    "sub":"/Users/44f6142df96bd6ab61e7521d9"
  },
  "https://example.com/scim/event/passwordResetExt":{
     "resetAttempts":5
  }
}
```

Figure 1: Example SCIM Password Reset Event

The event in the figure above expresses hypothetical password reset event for SCIM [RFC7644]. The JWT consists of:

- An _events_ attribute specifying the hypothetical SCIM urn ("urn:ietf:params:scim:event:passwordReset") for a password reset, and a custom extension, "https://example.com/scim/event/passwordResetExt", that is used to provide additional event information presumably specified by the location URI provided.

- An "iss" attribute, denotes the event publisher in the envelope while the "iss" in the event payload specifies the SCIM service provider for the account that was reset.

- The "aud" attribute specifies the intended audience for the event. In practical terms, this MAY be the URI for the event Feed that a client has subscribed to.

Additional extensions to an event may be added by adding more values to the "events" attribute. For each event URI value specified, there MAY be a corresponding attribute that has a JSON object that contains the attributes associated with that event (e.g., "https://example.com/scim/event/passwordResetExt"). In this example, the SCIM event indicates that a password has been updated and the current password reset count is 5. Notice that the value for "resetAttempts" is actually part of its own JSON object "https://example.com/scim/event/passwordResetExt".

Here is another example event token, this one for a Logout Token:

```
{
  "iss": "https://server.example.com",
  "aud": "https://rp.example.com",
  "jti": "3d0c3cf797584bd193bd0fb1bd4e7d30",
  "iat": 1458668180,
  "exp": 1458668580,
  "events": [
    "https://specs.openid.net/logout"
  ],
  "https://specs.openid.net/logout": {
    "iss": "https://token.example.com",
    "sub": "248289761001",
    "jti": "08a5019c-17e1-4977-8f42-65a12843ea02"
  }
}
```

Figure 2: Example OpenID Logout Event

In the above example, the event has its own issuer, "https://server.example.com" while the event is about the logging out of a user session identified in the event extension by "jti" that was issued by "https://token.example.com".

In the following example, a fictional medical service collects consent for medical actions and notifies other parties. The individual for whom consent is identified was originally authenticated via OpenID Connect. In this case, the issuer of the SET event is an application rather than the OpenID provider:

```
{
  "jti": "fb4e75b5411e4e19b6c0fe87950f7749",
  "events":[
    "https://openid.net/heart/consent.html"
  ],
  "iat": 1458496025,
  "iss": "https://my.examplemed.com",
  "aud":[
    "https://rp.example.com"
  ],
  "https://openid.net/heart/consent":{
    "iss": "https://token.example.com",
    "sub": "248289761001",
    "consentUri":[
      "https://terms.examplemed.com/labdisclosure.html#Agree"
    ]
  }
}
```

Figure 3: Example Consent Event

In the above example "iss" and "sub" contained within the attribute "https://openid.net/heart/consent", refer to the subject and issuer of the original OpendID Provider. They are distinct from the top level value of "iss" which always refers to the issuer of the event - a medical consent service that is a relying party to the OpenID Provider.

2.1. Core SET Attributes

The following are attributes that are based on [RFC7519] claim definitions and are profiled for use in an event token:

jti
: As defined by Section 4.1.7 [RFC7519] contains a unique identifier for an event. The identifier SHOULD be unique within a particular event Feed and MAY be used by clients to track whether a particular event has already been received. This attribute is REQUIRED.

iss

A single valued String containing the URI of the service provider publishing the SET (the issuer). This attribute is REQUIRED.

aud
A multi-valued String containing the URIs representing the audience of the event. Values are typically URLs of the Feeds the event is associated with. When an event has multiple audiences that go to the same subscriber, the publisher is not obligated to deliver repeated events to the same subscriber. This attribute is RECOMMENDED.

iat
As defined by Section 4.1.6 [RFC7519], a value containing a NumericDate, which represents when the event was issued. Unless otherwise specified, the value SHOULD be interpreted by the subscriber as equivalent to the actual time of the event. This attribute is REQUIRED.

nbf
As defined by Section 4.1.5 [RFC7519], a value containing a NumericDate, which represents a future date when the event will occur. This attribute is OPTIONAL.

The following is a new attribute defined by this specification:

events
A multi-valued String that contains one or more URIs representing the type of event being expressed and the attributes that MAY be available within the JWT. Each value in this attribute indicates what other JSON sub-objects MAY present within the parent JSON SET structure. Each JSON sub-object is denoted by an attribute that matches a value in "events". This attribute is REQUIRED.

2.2. Security Event Token Construction

A SET is a JWT [RFC7519] that is constructed by building a JSON structure that constitutes an event object and which is then used as the body of a JWT.

While this specification uses JWT to convey a SET, implementers SHALL NOT use SETs to convey authentication or authorization assertions.

The following is an example event token (which has been formatted for readability):


```
{
  "jti": "4d3559ec67504aaba65d40b0363faad8",
  "iat": 1458496404,
  "iss": "https://scim.example.com",
  "aud":[
   "https://scim.example.com/Feeds/98d52461fa5bbc879593b7754",
   "https://scim.example.com/Feeds/5d7604516b1d08641d7676ee7"
  ],

  "events":[
    "urn:ietf:params:scim:event:create"
  ],
  "urn:ietf:params:scim:event:create":{
    "ref": "https://scim.example.com/Users/44f6142df96bd6ab61e7521d9",
    "attributes":["id", "name", "userName", "password", "emails"],
    "values":{
      "emails":[
       {"type":"work", "value":"jdoe@example.com"}
      ],
      "password":"not4u2no",
      "userName":"jdoe",
      "id":"44f6142df96bd6ab61e7521d9",
      "name":{
        "givenName":"John",
        "familyName":"Doe"
      }
    }
  }
}
```


Figure 4: Example Event JSON Data

When transmitted, the above JSON body must be converted into a JWT as per [RFC7519]. In this example, because the event contains attribute values, the token MUST be encrypted per JWE (see [RFC7516]) before transmission.

The following is an example of a SCIM Event expressed as an unsecured JWT. The JWT header of:


```
{"alg":"none"}
```

Base64url encoding of the octets of the UTF-8 representation of the header yields:

```
eyJhbGciOiJub25lIn0
```

The example JSON Event Data is encoded as follows:

```
eyAgCiAgImp0aSI6ICI0ZDM1NTllYzY3NTA0YWFiYTY1ZDQwYjAzNjNmYWFkOCIsCiAg
ImlhdCI6IDE0NTg0OTY0MDQsCiAgImlzcyI6ICJodHRwczovL3NjaW0uZXhhbXBsZS5j
b20iLCAgCiAgImF1ZCI6WwogICAiaHR0cHM6Ly9zY2ltLmV4YW1wbGUuY29tL0ZlZWRz
Lzk4ZDUyNDYxZmE1YmJjODc5NTkzYjc3NTQiLAogICAiaHR0cHM6Ly9zY2ltLmV4YW1w
bGUuY29tL0ZlZWRzLzVkNzYwNDUxNmIxZDA4NjQxZDc2NzZlZTciCiAgXSwgIAogICAo
ICJldmVudHMiOlsKICAgICJ1cm46aWV0ZjpwYXJhbXM6c2NpbTpldmVudDpjcmVhdGUi
CiAgXSwKICAidXJuOmlldGY6cGFyYW1zOnNjaW06ZXZlbnQ6Y3JlYXRlIjp7CiAgICAi
cmVmIjogImh0dHBzOi8vc2NpbS5leGFtcGxlLmNvbS9Vc2Vycy80NGY2MTQyZGY5NmJk
NmFiNjFlNzUyMWQ5IiwKICAgICJhdHRyaWJ1dGVzIjpbImlkIiwibmFtZSIsInVzZXJO
YW1lIiwicGFzc3dvcmQiLCJlbWFpbHMiXSwKICAgICJ2YWx1ZXMiOnsKICAgICAgImVt
YWlscyI6WwogICAgICAgeyJ0eXBlIjoid29yayIsInZhbHVlIjoiamRvZUBleGFtcGxl
LmNvbSJ9CiAgICAgIF0sCiAgICAgICJwYXNzd29yZCI6Im5vdDR1Mm5vIiwKICAgICAg
InVzZXJOYW1lIjoiamRvZSIsCiAgICAgICJpZCI6IjQ0ZjYxNDJkZjk2YmQ2YWI2MWU3
NTIxZDkiLAogICAgICAibmFtZSI6ewogICAgICAgICJnaXZlbk5hbWUiOiJKb2huIiwK
ICAgICAgICAiZmFtaWx5TmFtZSI6IkRvZSIKICAgICAgfQogICAgfSAgCiAgfQp9
```

The encoded JWS signature is the empty string. Concatenating the parts yields:

```
eyJhbGciOiJub25lIn0
.
eyAgCiAgImp0aSI6ICI0ZDM1NTllYzY3NTA0YWFiYTY1ZDQwYjAzNjNmYWFkOCIsCiAg
ImlhdCI6IDE0NTg0OTY0MDQsCiAgImlzcyI6ICJodHRwczovL3NjaW0uZXhhbXBsZS5j
b20iLCAgCiAgImF1ZCI6WwogICAiaHR0cHM6Ly9zY2ltLmV4YW1wbGUuY29tL0ZlZWRz
Lzk4ZDUyNDYxZmE1YmJjODc5NTkzYjc3NTQiLAogICAiaHR0cHM6Ly9zY2ltLmV4YW1w
bGUuY29tL0ZlZWRzLzVkNzYwNDUxNmIxZDA4NjQxZDc2NzZlZTciCiAgXSwgIAogICAo
ICJldmVudHMiOlsKICAgICJ1cm46aWV0ZjpwYXJhbXM6c2NpbTpldmVudDpjcmVhdGUi
CiAgXSwKICAidXJuOmlldGY6cGFyYW1zOnNjaW06ZXZlbnQ6Y3JlYXRlIjp7CiAgICAi
cmVmIjogImh0dHBzOi8vc2NpbS5leGFtcGxlLmNvbS9Vc2Vycy80NGY2MTQyZGY5NmJk
NmFiNjFlNzUyMWQ5IiwKICAgICJhdHRyaWJ1dGVzIjpbImlkIiwibmFtZSIsInVzZXJO
YW1lIiwicGFzc3dvcmQiLCJlbWFpbHMiXSwKICAgICJ2YWx1ZXMiOnsKICAgICAgImVt
YWlscyI6WwogICAgICAgeyJ0eXBlIjoid29yayIsInZhbHVlIjoiamRvZUBleGFtcGxl
LmNvbSJ9CiAgICAgIF0sCiAgICAgICJwYXNzd29yZCI6Im5vdDR1Mm5vIiwKICAgICAg
InVzZXJOYW1lIjoiamRvZSIsCiAgICAgICJpZCI6IjQ0ZjYxNDJkZjk2YmQ2YWI2MWU3
NTIxZDkiLAogICAgICAibmFtZSI6ewogICAgICAgICJnaXZlbk5hbWUiOiJKb2huIiwK
ICAgICAgICAiZmFtaWx5TmFtZSI6IkRvZSIKICAgICAgfQogICAgfSAgCiAgfQp9
.
```

Figure 5: Example Unsecured Event Token

To create and or validate a signed or encrypted SET, follow the instructions in section 7 of [RFC7519].

3. Security Considerations

SETs may often contain sensitive information. Therefore, methods for distribution of events SHOULD require the use of a transport-layer security mechanism when distributing events. Parties MUST support TLS 1.2 [RFC5246] and MAY support additional transport-layer mechanisms meeting its security requirements. When using TLS, the client MUST perform a TLS/SSL server certificate check, per [RFC6125]. Implementation security considerations for TLS can be found in "Recommendations for Secure Use of TLS and DTLS" [RFC7525].

Security Events distributed through third-parties or that carry personally identifiable information, SHOULD be encrypted using JWE [RFC7516] or secured for confidentiality by other means.

Security Events distributed without authentication over the channel, such as via TLS ([RFC5246] and [RFC6125]), and/or OAuth2 [RFC6749], or Basic Authentication [RFC7617], MUST be signed using JWS [RFC7515] so that individual events MAY be authenticated and validated by the subscriber.

4. Privacy Considerations

If a SET needs to be retained for audit purposes, JWS MAY be used to provide verification of its authenticity.

Event Publishers SHOULD attempt to specialize Feeds so that the content is targeted to the specific business and protocol needs of subscribers.

When sharing personally identifiable information or information that is otherwise considered confidential to affected users, the publishers and subscribers MUST have the appropriate legal agreements and user consent in place.

The propagation of subject identifiers can be perceived as personally identifiable information. Where possible, publishers and subscribers should devise approaches that prevent propagation -- for example, the passing of a hash value that requires the subscriber to already know the subject.

5. IANA Considerations

5.1. JSON Web Token Claims Registration

This specification registers the "events" claim in the IANA "JSON Web Token Claims" registry [IANA.JWT.Claims] established by [RFC7519].

5.1.1. Registry Contents

- Claim Name: "events"
- Claim Description: Security Events
- Change Controller: IESG
- Specification Document(s): Section 2 of [[this specification]]

6. References

6.1. Normative References

[IANA.JWT.Claims] IANA, "JSON Web Token Claims", <http://www.iana.org/assignments/jwt>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <http://www.rfc-editor.org/info/rfc3986>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <http://www.rfc-editor.org/info/rfc5246>.

[RFC6125] Saint-Andre, P. and J. Hodges, "Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)", RFC 6125, DOI 10.17487/RFC6125, March 2011, <http://www.rfc-editor.org/info/rfc6125>.

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, DOI 10.17487/RFC6749, October 2012, <http://www.rfc-editor.org/info/rfc6749>.

[RFC7159] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", RFC 7159, DOI 10.17487/RFC7159, March 2014, <http://www.rfc-editor.org/info/rfc7159>.

[RFC7519] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Token (JWT)", RFC 7519, DOI 10.17487/RFC7519, May 2015, <http://www.rfc-editor.org/info/rfc7519>.

[RFC7525] Sheffer, Y., Holz, R., and P. Saint-Andre, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 7525, DOI 10.17487/RFC7525, May 2015, <http://www.rfc-editor.org/info/rfc7525>.

[RFC7617] Reschke, J., "The 'Basic' HTTP Authentication Scheme", RFC 7617, DOI 10.17487/RFC7617, September 2015, <http://www.rfc-editor.org/info/rfc7617>.

6.2. Informative References

[idevent-scim] Oracle Corporation, "SCIM Event Extensions (work in progress)", <draft-hunt-idevent-scim-00.txt>.

[RFC7009] Lodderstedt, T., Ed., Dronia, S., and M. Scurtescu, "OAuth 2.0 Token Revocation", RFC 7009, DOI 10.17487/RFC7009, August 2013, <http://www.rfc-editor.org/info/rfc7009>.

[RFC7515] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Signature (JWS)", RFC 7515, DOI 10.17487/RFC7515, May 2015, <http://www.rfc-editor.org/info/rfc7515>.

[RFC7516] Jones, M. and J. Hildebrand, "JSON Web Encryption (JWE)", RFC 7516, DOI 10.17487/RFC7516, May 2015, <http://www.rfc-editor.org/info/rfc7516>.

[RFC7517] Jones, M., "JSON Web Key (JWK)", RFC 7517, DOI 10.17487/RFC7517, May 2015, <http://www.rfc-editor.org/info/rfc7517>.

[RFC7644] Hunt, P., Ed., Grizzle, K., Ansari, M., Wahlstroem, E., and C. Mortimore, "System for Cross-domain Identity Management: Protocol", RFC 7644, DOI 10.17487/RFC7644, September 2015, <http://www.rfc-editor.org/info/rfc7644>.

Appendix A. Acknowledgments

The editors would like to thank the participants in the IETF id-event mailing list and related working groups for their support of this specification.

Appendix B. Change Log

Draft 01 - PH - Renamed eventUris to events

Draft 00 - PH - First Draft

Draft 01 - PH - Fixed some alignment issues with JWT. Remove event type attribute.

Draft 02 - PH - Renamed to Security Events, Removed questions, clarified examples and intro text, and added security and privacy section.

Draft 03 - PH

General edit corrections from Sarah Squire
Changed "event" term to "SET"
Corrected author organization for William Dennis to Google
Changed definition of SET to be 2 parts, an envelope and 1 or more payloads.
Clarified that the intent is to express a single event with optional extensions only.

Draft 03 - mbj - Registered "events" claim. Applied proofreading corrections.

Authors' Addresses

Phil Hunt (editor)
Oracle Corporation

Email: phil.hunt@yahoo.com

William Denniss
Google

Email: wdenniss@google.com

Morteza Ansari
Cisco

Email: morteza.ansari@cisco.com

Michael B. Jones
Microsoft

Email: mbj@microsoft.com
URI: http://self-issued.info/