SET Token Delivery Using HTTP

Marius Scurtrescu, Google

IETF99 Prague

July 2017

Overview

The specification defines:

- how SETs can be delivered to a receiver
  - push initiated by transmitter
  - poll initiated by receiver
- verification process to test an Event Stream

draft-hunt-secevent-delivery-00

Definitions

- Identity Provider
  - explicit: a service provider that issues authentication assertions
  - implicit: service providers that manage personal identifiers used in recovery scenarios by Relying Parties (i.e. email or phone number)
- Relying Party
  - explicit: a service provider that accepts authentication assertions
  - implicit: service providers that use a personal identifier managed by another provider
- Event Transmitter - a service provider that delivers SETs
- Event Receiver - a service provider that receives SETs
- Event Stream - a defined location and distribution method through which an Event Transmitter sends message to an Event Receiver

Event Delivery Process

- how SETs are defined and how SETs are assigned to Event Streams is out of scope
- when a SET is available in an Event Stream, the delivery is determined by the Event Receiver's registered mechanism:
  - push: HTTP POST to the Event Receiver endpoint
  - poll: the event is queued up in a buffer so the Event Receiver can poll using HTTP POST
  - another method not defined in this specificaiton
- the Event Receiver must acknowledge receipt to the Event Transmitter
- after an acknowledged delivery the Event Transmitter is not required to maintain SETs

Push Delivery

- The Event Transmitter uses HTTP POST to deliver SETs to a configured callback URI
- request HTTP Content-Type: application/secevent+jwt
- Accept header: application/json
- a single SET per request
- if the SET is accepted then the response should be 202 (Accepted)
- in case of an error the Event Receiver may respond with an appropriate HTTP status code

Push Deliver - Request Example

```
POST /Events HTTP/1.1

Host: notify.examplerp.com
Accept: application/json
Authorization: Bearer h480djs93hd8
Content-Type: application/secevent+jwt
eyJhbGciOiJub25lIn0
.
eyJwdWJsaXNoZXJVcmkiOiJodHRwczovL3NjaW0uZXhhbXBsZS5jb20iLCJmZWV
kVXJpcyI6WyJodHRwczovL2podWIuZXhhbXBsZS5jb20vRmVlZHMvOThkNTI0Nj
b2VAZXhhbXBsZS5jb20ifV0sInBhc3N3b3JkIjoibm90NHUybm8iLCJ1c2VyTmF
tZSI6Impkb2UiLCJpZCI6IjQ0ZjYxNDJkZjk2YmQ2YWI2MWU3NTIxZDkiLCJuYW
1lIjp7ImdpdmVuTmFtZSI6IkpvaG4iLCJmYW1pbHlOYW1lIjoiRG9lIn19fQ
.
```

Push Deliver - Response Examples

Success:

```
HTTP/1.1 202 Accepted
```

Error:

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
  "err":"dup",
  "description":"SET already received. Ignored."
}
```

Polling Delivery

- Event Receiver uses HTTP POST to both acknowledge SETs and to receive more SETs
- request & response HTTP Content-Type: application/json
- multiple SETs per response
- request consist of polling parameters, in JSON format

Polling Delivery - Request Attributes

Processing Parameters

- maxEvents - maximum number of SETs that should be returned
  - 0 (zero) means ack only request
- returnImmediately - false (the default) requests a long poll

SET Ack Parameters:

- ack - array of strings, each the "jti" of a successfully received SET
- setErrs - object with "jti" keys and "err"+"description" nested values

Polling Delivery - Response Attributes

- sets - object with "jti" keys and encoded SETs as values
- moreAvailable - boolean indicationg that more SETs are available

Polling Delivery - Variations

1. Poll Only - no previous SETs to acknowledge
2. Acknowledge Only - maxEvents set to 0 and "ack" and/or "setErrs" present
3. Combined Acknowledge and Poll

Polling Delivery - Combined Request Example

```
POST /Events HTTP/1.1

Host: notify.exampleidp.com
Content-Type: application/json
Authorization: Bearer h480djs93hd8

{
  "ack":["4d59ec67504aaba65d40b0363faad8","3d0c3797584bd193bd0fb1bd4e7d30"],
  "setErrs":{
    "4d3559ec67504aaba65d40b03faad8":{
      "err":"jwtAud",
      "description":"The audience value was incorrect."
    }
  },
  "returnImmediately":false
}
```

Polling Delivery - Response Example

```
HTTP/1.1 200 OK
Content-Type: application/json
Location: https://notify.exampleidp/Events

{
  "sets":{
    "4d3559ec67504aaba65d40b0363faad8":
      "eyJhbGciOiJub25lIn0.
       2ZW50OmNyZWF0ZSI6eyJyZWYiOiJodHRwczovL3NjaW0uZXhhbXBsZS5jb20vVXNlcn
       W1lIiwidXNlck5hbWUiLCJwYXNzd29yZCIsImVtYWlscyJdfX19.",
    "3d0c3cf797584bd193bd0fb1bd4e7d30":
      "eyJhbGciOiJub25lIn0.
       eyJqdGkiOiIzZDBjM2NmNzk3NTg0YmQxOTNiZDBmYjFiZDRlN2QzMCIsImlhdCI6MTQ
       L3Bhc3N3b3JkUmVzZXRFeHQiOnsicmVzZXRBdHRlbXB0cyI6NX19fQ."
  }
}
```

SET Errors 1/2

err	description
json	Invalid JSON object
jwtParse	Invalid or unparsable JWT or JSON structure
jwtHdr	An invalid JWT header was detected
jwtCrypto	Unable to parse due to unsupported algorithm
jws	Signature was not validated
jwe	Unable to decrypt JWE encoded data
jwtAud	Invalid audience value
jwtIss	Issuer not recognized
setType	An unexpected Event type was received

SET Errors 2/2

err	description
setParse	Invalid structure was encountered such as an inability to parse or an incomplete set of Event claims
setData	SET event claims incomplete or invalid
dup	A duplicate SET was received and has been ignored

Event Stream Verification

- Event Receiver initiates a request to verify the stream
  - it provides "confirm" and "nonce" values
- Event Transmitter delivers Verify Event

Event Stream Verification - Example Event


```
{
  "jti": "4d3559ec67504aaba65d40b0363faad8",
  "iat": 1458496404,
  "iss": "https://scim.example.com",
  "exp": 1458497000,
  "aud":[
    "https://event.example.com/Feeds/98d52461fa5bbc879593b7754"
  ],
  "events": {
    "[[this RFC URL]]#verify":{
      "confirm":"ca2179f4-8936-479a-a76d-5486e2baacd7",
      "nonce":"1668c993e95849869e4b3506cccdf9bf"
    }
  }
}
```

Authentication and Authorization

SET Delivery depends on TLS and/or standard HTTP authentication and authorization schemes.

For example:

- TLS Client Authentication
- Bearer Tokens
- Basic Authentication
- SET Payload Authentication

Q & A