

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 11, 2019

A. Backman, Ed.
Amazon
M. Jones, Ed.
Microsoft
M. Scurtescu
Coinbase
M. Ansari
Cisco
A. Nadalin
Microsoft
March 10, 2019

**Poll-Based Security Event Token (SET) Delivery Using HTTP
draft-ietf-secevent-http-poll-02**

Abstract

This specification defines how a series of Security Event Tokens (SETs) may be delivered to an intended recipient using HTTP POST over TLS initiated as a poll by the recipient. The specification also defines how delivery can be assured, subject to the SET Recipient's need for assurance.

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 <https://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 September 11, 2019.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents

(<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.	Introduction and Overview	2
1.1.	Notational Conventions	3
1.2.	Definitions	3
2.	SET Delivery	3
2.1.	Polling Delivery using HTTP	4
2.2.	Polling HTTP Request Attributes	4
2.3.	Polling HTTP Response Attributes	5
2.4.	Poll Request	6
2.4.1.	Poll Only Request	7
2.4.2.	Acknowledge Only Request	8
2.4.3.	Poll with Acknowledgement	8
2.4.4.	Poll with Acknowledgement and Errors	9
2.5.	Poll Response	10
2.6.	Error Response Handling	12
3.	Authentication and Authorization	12
3.1.	Use of Tokens as Authorizations	13
4.	Security Considerations	14
4.1.	Authentication Using Signed SETs	14
4.2.	HTTP Considerations	14
4.3.	TLS Support Considerations	14
4.4.	Access Token Considerations	15
4.4.1.	Bearer Token Considerations	15
5.	Privacy Considerations	15
6.	IANA Considerations	16
7.	References	16
7.1.	Normative References	16
7.2.	Informative References	17
Appendix A.	Acknowledgments	18
Appendix B.	Change Log	18
	Authors' Addresses	19

1. Introduction and Overview

This specification defines how a stream of Security Event Tokens (SETs) [[RFC8417](#)] can be transmitted to an intended SET Recipient using HTTP [[RFC7231](#)] over TLS. The specification defines a method to poll for SETs using HTTP POST.

[1.1.](#) Notational Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [BCP 14](#) [[RFC2119](#)] [[RFC8174](#)] when, and only when, they appear in all capitals, as shown here.

Throughout this document, all figures MAY contain spaces and extra line wrapping for readability and due to space limitations.

[1.2.](#) Definitions

This specification utilizes terminology defined in [[RFC8417](#)], as well as the terms defined below:

SET Transmitter

An entity that delivers SETs in its possession to one or more SET Recipients.

[2.](#) SET Delivery

When an event occurs, the SET Transmitter constructs a SET [[RFC8417](#)] that describes the event. The SET Transmitter determines the SET Recipients that the SET should be distributed to.

How SETs are defined and the process by which events are identified for SET Recipients is out of scope of this specification.

When a SET is available for a SET Recipient, the SET Transmitter attempts to deliver the SET by queueing s the SET in a buffer so that a SET Recipient can poll for SETs using HTTP/1.1 POST.

In Poll-Based SET Delivery Using HTTP, zero or more SETs are delivered in a JSON [[RFC8259](#)] document to a SET Recipient in response to an HTTP POST request to the SET Transmitter. Then in a following request, the SET Recipient acknowledges received SETs and can poll for more. All requests and responses are JSON documents and use a "Content-Type" of "application/json", as described in [Section 2.1](#).

After successful (acknowledged) SET delivery, SET Transmitters are not be required to retain or record SETs for recovery. Once a SET is acknowledged, the SET Recipient SHALL be responsible for retention and recovery.

Transmitted SETs SHOULD be self-validating (e.g., signed) if there is a requirement to verify they were issued by the SET Transmitter at a later date when de-coupled from the original delivery where

authenticity could be checked via the HTTP or TLS mutual authentication.

Upon receiving a SET, the SET Recipient reads the SET and validates it. The SET Recipient MUST acknowledge receipt to the SET Transmitter.

The SET Recipient SHALL NOT use the event acknowledgement mechanism to report event errors other than relating to the parsing and validation of the SET.

2.1. Polling Delivery using HTTP

This method allows a SET Recipient to use HTTP POST ([Section 4.3.3 of \[RFC7231\]](#)) to acknowledge SETs and to check for and receive zero or more SETs. Requests MAY be made at a periodic interval (short polling) or requests MAY wait, pending availability of new SETs using long polling, per [Section 2 of \[RFC6202\]](#).

The delivery of SETs in this method is facilitated by HTTP POST requests initiated by the SET Recipient in which:

- o The SET Recipient makes a request for available SETs using an HTTP POST to a pre-arranged endpoint provided by the SET Transmitter.
Or,
- o After validating previously received SETs, the SET Recipient initiates another poll request using HTTP POST that includes acknowledgement of previous SETs and waits for the next batch of SETs.

The purpose of the acknowledgement is to inform the SET Transmitter that delivery has succeeded and redelivery is no longer required. Before acknowledgement, SET Recipients SHOULD ensure that received SETs have been validated and retained in a manner appropriate to the recipient's requirements. The level and method of retention of SETs by SET Recipients is out of scope of this specification.

2.2. Polling HTTP Request Attributes

When initiating a poll request, the SET Recipient constructs a JSON document that consists of polling request parameters and SET acknowledgement parameters in the form of JSON attributes. The request payloads are delivered in a JSON document, as described in [Section 2.4](#) and [Section 2.5](#).

When making a request, the HTTP header "Content-Type" is set to "application/json".

The following JSON attributes are used in a polling request:

Request Processing Parameters

maxEvents

An OPTIONAL JSON integer value indicating the maximum number of unacknowledged SETs that SHOULD be returned. If more than the maximum number of SETs are available, the oldest SETs available SHOULD be returned first. A value of "0" MAY be used by SET Recipients that would like to perform an acknowledge only request. This enables the Recipient to use separate HTTP requests for acknowledgement and reception of SETs. If this parameter is omitted, no limit is placed on the number of SETs to be returned.

returnImmediately

An OPTIONAL JSON boolean value that indicates the SET Transmitter SHOULD return an immediate response even if no results are available (short polling). The default value is "false" indicates the request is to be treated as an HTTP Long Poll, per [Section 2 of \[RFC6202\]](#). The timeout for the request is part of the configuration between the participants, which is out of scope of this specification.

SET Acknowledgment Parameters

ack

Which is an array of Strings that each correspond to the "jti" of a successfully received SET. If there are no outstanding SETs to acknowledge, the attribute MAY be omitted. When acknowledging a SET, the SET Transmitter is released from any obligation to retain the SET (e.g., for a future retry to receive).

setErrs

A JSON Object that contains one or more nested JSON attributes that correspond to the "jti" of each invalid SET received. The value of each is a JSON object whose contents is an "err" attribute and "description" attribute whose value correspond to the errors described in [Section 2.6](#).

[2.3](#). Polling HTTP Response Attributes

In response to a poll request, the SET Transmitter checks for available SETs and responds with a JSON document containing the following JSON attributes:

sets

A JSON object that contains zero or more nested JSON attributes. Each nested attribute corresponds to the "jti" of a SET to be delivered and whose value is a JSON String containing the value of the encoded corresponding SET. If there are no outstanding SETs to be transmitted, the JSON object SHALL be empty.

moreAvailable

A JSON boolean value that indicates if more unacknowledged SETs are available to be returned.

When making a response, the HTTP header "Content-Type" is set to "application/json".

2.4. Poll Request

The SET Recipient performs an HTTP POST (see [Section 4.3.4 of \[RFC7231\]](#)) to a pre-arranged polling endpoint URI to check for SETs that are available. Because the SET Recipient has no prior SETs to acknowledge, the "ack" and "errs" request parameters are omitted.

If after a period of time, negotiated between the SET Transmitter and Recipient, a SET Transmitter MAY redeliver SETs it has previously delivered. The SET Recipient SHOULD accept repeat SETs and acknowledge the SETs regardless of whether the Recipient believes it has already acknowledged the SETs previously. A SET Transmitter MAY limit the number of times it attempts to deliver a SET.

If the SET Recipient has received SETs from the SET Transmitter, the SET Recipient SHOULD parse and validate received SETs to meet its own requirements and SHOULD acknowledge receipt in a timely fashion (e.g., seconds or minutes) so that the SET Transmitter can mark the SETs as received. SET Recipients SHOULD acknowledge receipt before taking any local actions based on the SETs to avoid unnecessary delay in acknowledgement, where possible.

Poll requests have three variations:

Poll Only

In which a SET Recipient asks for the next set of events where no previous SET deliveries are acknowledged (such as in the initial poll request).

Acknowledge Only

In which a SET Recipient sets the "maxEvents" attribute to "0" along with "ack" and "err" attributes indicating the SET Recipient is acknowledging previously received SETs and does not want to receive any new SETs in response to the request.

Combined Acknowledge and Poll

In which a SET Recipient is both acknowledging previously received SETs using the "ack" and "err" attributes and will wait for the next group of SETs in the SET Transmitters response.

2.4.1. Poll Only Request

In the case where no SETs were received in a previous poll (see Figure 7), the SET Recipient simply polls without acknowledgement parameters ("sets" and "setErrs").

The following is an example request made by a SET Recipient that has no outstanding SETs to acknowledge and is polling for available SETs.

The following is a non-normative example poll request to the endpoint: "https://notify.exampleidp.com/Events".

```
POST /Events HTTP/1.1
```

```
Host: notify.exampleidp.com
Authorization: Bearer h480djs93hd8
Accept: application/json
```

```
{
  "returnImmediately": true
}
```

Figure 1: Example Initial Poll Request

A SET Recipient can poll using default parameter values by passing an empty JSON object.

The following is a non-normative example default poll request to the endpoint: "https://notify.exampleidp.com/Events".

```
POST /Events HTTP/1.1
```

```
Host: notify.exampleidp.com
Authorization: Bearer h480djs93hd8
Accept: application/json
```

```
{}
```

Figure 2: Example Default Poll Request

2.4.2. Acknowledge Only Request

In this variation, the SET Recipient acknowledges previously received SETs and indicates it does not want to receive SETs in response by setting the "maxEvents" attribute to "0".

This variation might be used, for instance, when a SET Recipient needs to acknowledge received SETs independently (e.g., on separate threads) from the process of receiving SETs.

The following is a non-normative example poll with acknowledgement of SETs received (for example as shown in Figure 6).

```
POST /Events HTTP/1.1
```

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

```
{
  "ack": [
    "4d3559ec67504aaba65d40b0363faad8",
    "3d0c3cf797584bd193bd0fb1bd4e7d30"
  ],
  "maxEvents": 0,
  "returnImmediately": true
}
```

Figure 3: Example Acknowledge Only Request

2.4.3. Poll with Acknowledgement

This variation allows a recipient thread to simultaneously acknowledge previously received SETs and wait for the next group of SETs in a single request.

The following is a non-normative example poll with acknowledgement of SETs received in Figure 6.

```
POST /Events HTTP/1.1
```

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

```
{
  "ack": [
    "4d3559ec67504aaba65d40b0363faad8",
    "3d0c3cf797584bd193bd0fb1bd4e7d30"
  ],
  "returnImmediately": false
}
```

Figure 4: Example Poll with Acknowledgement and No Errors

In the above acknowledgement, the SET Recipient has acknowledged receipt of two SETs and has indicated it wants to wait until the next SET is available.

[2.4.4. Poll with Acknowledgement and Errors](#)

In the case where errors were detected in previously delivered SETs, the SET Recipient MAY use the "setErrs" attribute to communicate the errors in the following poll request.

The following is a non-normative example of a response acknowledging one successfully received SET and one SET with an error from the two SETs received in in Figure 6.

```
POST /Events HTTP/1.1

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

{
  "ack": ["3d0c3cf797584bd193bd0fb1bd4e7d30"],
  "setErrs": {
    "4d3559ec67504aaba65d40b0363faad8": {
      "err": "jwtAud",
      "description": "The audience value was invalid."
    }
  },
  "returnImmediately": true
}
```

Figure 5: Example Poll Acknowledgement with Error

2.5. Poll Response

In response to a poll request, the service provider MAY respond immediately if SETs are available to be delivered. If no SETs are available at the time of the request, the SET Transmitter SHALL delay responding until a SET is available or the timeout interval has elapsed unless the poll request parameter "returnImmediately" is "true".

As described in [Section 2.3](#), a JSON document is returned containing a number of attributes including "sets" which SHALL contain zero or more SETs.

The following is a non-normative example response to the request shown [Section 2.4](#). This example shows two SETs are returned.

HTTP/1.1 200 OK

Content-Type: application/json

Location: <https://notify.exampleidp/Events>

```
{
  "sets": {
    "4d3559ec67504aaba65d40b0363faad8":
      "eyJhbGciOiJub25lIn0.eyJqdGkiOiI0ZDM1NTllYzY3NTA0YWFiYTY1ZDQwYjAzNjNmYWFKOCIsIm1hdCI6MTQ1ODQ5NjQwNCwiaXNzIjoiaHR0cHM6Ly9zY2ltLmV4YW1wbGUuY29tIiwiaXVkiIjpbImh0dHBzOi8vc2NpbS5leGFtcGx1LmNvbS9GZWVkc385ODQ1MjQ2MWZhNWJiYzg3OTU5M2I3NzU0IiwiaHR0cHM6Ly9zY2ltLmV4YW1wbGUuY29tL0ZlZWZlZVknZyYwNDUxNmIxZDA4NjQxZDc2NzZlZTciXSwiZXZlbnRzIjp7InVyb3ppZXRmOnBhcmFtczpzY2ltOmV2ZW50OmNyZWZ0ZSI6eyJyZWYiOiJodHRwczovL3Njaw0uZXhhbXBsZS5jb20vVXNlcnMvNDRmNjE0MmRmOTZiZDZhYjYxZTc1MjFkOSIsImF0dHJpYnV0ZXMiOiI0IiwiaWQiOiJlcjUyW1lIiwidXNlck5hbWUiOiJwYXNzd29yZCI6ImVtYWlscyJdfX19.",
    "3d0c3cf797584bd193bd0fb1bd4e7d30":
      "eyJhbGciOiJub25lIn0.eyJqdGkiOiIzZDBjM2NmNzk3NTg0YmQxOTNiZDBmYjFiZDRlN2QzMCI6MTQ1ODQ5NjAyNSwiaXNzIjoiaHR0cHM6Ly9zY2ltLmV4YW1wbGUuY29tIiwiaXVkiIjpbImh0dHBzOi8vamh1Yi5leGFtcGx1LmNvbS9GZWVkc385ODQ1MjQ2MWZhNWJiYzg3OTU5M2I3NzU0IiwiaHR0cHM6Ly9qaHViLmV4YW1wbGUuY29tL0ZlZWZlZVknZyYwNDUxNmIxZDA4NjQxZDc2NzZlZTciXSwic3ViIjoiaHR0cHM6Ly9zY2ltLmV4YW1wbGUuY29tL1VzZXJzLzQ0ZjYxNDJkZjk2YmQ2YWI2MWU3NTIxZDkiLCJldmVudHMiOiI0IiwiaWQiOiJlcjUyW6cGFyYWIzOnNjaw06ZXZlbnQ6cGFzc3dvcmRSZXNldCI6eyJpZCI6IjQ0ZjYxNDJkZjk2YmQ2YWI2MWU3NTIxZDkiSwiaHR0cHM6Ly9leGFtcGx1LmNvbS9zY2ltL2V2ZW50L3Bhc3N3b3JkUmVzZXRFbHQiOiI0IiwiaWQiOiJlcjUyW1lIiwidXNlck5hbWUiOiJwYXNzd29yZCI6ImVtYWlscyJdfX19fQ."
  }
}
```

Figure 6: Example Poll Response

In the above example, a two SETs whose "jti" are "4d3559ec67504aaba65d40b0363faad8" and "3d0c3cf797584bd193bd0fb1bd4e7d30" are delivered.

The following is a non-normative example response to the request shown [Section 2.4](#) showing no new SETs or unacknowledged SETs are available.

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

{
  "sets": {}
}
```

Figure 7: Example No SETs Poll Response

Upon receiving the JSON document (e.g., as shown in Figure 6), the SET Recipient parses and verifies the received SETs and notifies the SET Transmitter via the next poll request to the SET Transmitter, as described in [Section 2.4.3](#) or [Section 2.4.4](#).

2.6. Error Response Handling

If a SET is invalid, error codes from the IANA "Security Event Token Delivery Error Codes" registry established by [\[I-D.ietf-secevent-http-push\]](#) are used in error responses. An error response SHALL include a JSON object that provides details about the error. The JSON object includes the JSON attributes:

err

A value from the IANA "Security Event Token Delivery Error Codes" registry that identifies the error.

description

A human-readable string that provides additional diagnostic information.

When included as part of a batch of SETs, the above JSON is included as part of the "setErrs" attribute, as defined in [Section 2.3](#) and [Section 2.4.4](#).

3. Authentication and Authorization

The SET delivery method described in this specification is based upon HTTP and depends on the use of TLS and/or standard HTTP authentication and authorization schemes as per [\[RFC7235\]](#). For example, the following methodologies could be used among others:

TLS Client Authentication

Event delivery endpoints MAY request TLS mutual client authentication, per [Section 7.3 of \[RFC5246\]](#).

Bearer Tokens

Bearer tokens [\[RFC6750\]](#) MAY be used when combined with TLS and a token framework such as OAuth 2.0 [\[RFC6749\]](#). For security considerations regarding the use of bearer tokens in SET delivery, see [Section 4.4.1](#).

Basic Authentication

Use of HTTP BASIC authentication should be avoided due to its use of a single factor that is based upon a relatively static, symmetric secret. When used, implementers SHOULD combine the use of basic authentication with other factors. The security considerations of HTTP BASIC are well documented in [\[RFC7617\]](#) and SHOULD be considered along with using signed SETs (see SET Payload Authentication below).

As per [Section 4.1 of \[RFC7235\]](#), a SET delivery endpoint SHALL indicate supported HTTP authentication schemes via the "WWW-Authenticate" header.

Because SET Delivery describes a simple function, authorization for the ability to pick-up or deliver SETs can be derived by considering the identity of the SET issuer, or via an authentication method above. This specification considers authentication as a feature to prevent denial-of-service attacks. Because SETs are not commands, SET Recipients are free to ignore SETs that are not of interest after acknowledging their receipt.

For illustrative purposes only, SET delivery examples show an OAuth2 bearer token value [\[RFC6750\]](#) in the authorization header. This is not intended to imply that bearer tokens are preferred. However, the use of bearer tokens in the specification does reflect common practice.

[3.1. Use of Tokens as Authorizations](#)

When using bearer tokens or proof-of-possession tokens that represent an authorization grant such as issued by OAuth (see [\[RFC6749\]](#)), implementers SHOULD consider the type of authorization granted, any authorized scopes (see [Section 3.3 of \[RFC6749\]](#)), and the security subject(s) that SHOULD be mapped from the authorization when considering local access control rules. [Section 6](#) of the OAuth Assertions draft [\[RFC7521\]](#), documents common scenarios for authorization including:

- o Clients using an assertion to authenticate and/or act on behalf of itself;
- o Clients acting on behalf of a user; and,
- o A Client acting on behalf of an anonymous user (e.g., see next section).

When using OAuth access tokens, implementers MUST take into account the threats and countermeasures documented in the security considerations for the use of client authorizations (see [Section 8 of \[RFC7521\]](#)). When using other token formats or frameworks, implementers MUST take into account similar threats and countermeasures, especially those documented by the relevant specifications.

4. Security Considerations

4.1. Authentication Using Signed SETs

In scenarios where HTTP authorization or TLS mutual authentication are not used or are considered weak, JWS signed SETs SHOULD be used (see [\[RFC7515\]](#) and Security Considerations [\[RFC8417\]](#)). This enables the SET Recipient to validate that the SET issuer is authorized to deliver the SET.

4.2. HTTP Considerations

SET delivery depends on the use of Hypertext Transfer Protocol and thus subject to the security considerations of HTTP [Section 9 of \[RFC7230\]](#) and its related specifications.

As stated in [Section 2.7.1 of \[RFC7230\]](#), an HTTP requestor MUST NOT generate the "userinfo" (i.e., username and password) component (and its "@" delimiter) when an "http" URI reference is generated with a message as they are now disallowed in HTTP.

4.3. TLS Support Considerations

SETs may contain sensitive information that is considered PII (e.g., subject claims). In such cases, SET Transmitters and SET Recipients MUST encrypt the SET, either with a transport-layer security mechanism, with JWE [\[RFC7516\]](#), or both. Event delivery endpoints 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" [\[RFC7525\]](#).

4.4. Access Token Considerations

When using access tokens such as those issued by OAuth 2.0 [[RFC6749](#)], implementers MUST take into account threats and countermeasures documented in [Section 8 of \[RFC7521\]](#).

4.4.1. Bearer Token Considerations

Due to the possibility of interception, Bearer tokens MUST be exchanged using TLS.

Bearer tokens MUST have a limited lifetime that can be determined directly or indirectly (e.g., by checking with a validation service) by the service provider. By expiring tokens, clients are forced to obtain a new token (which usually involves re-authentication) for continued authorized access. For example, in OAuth2, a client MAY use an OAuth refresh token to obtain a new bearer token after authenticating to an authorization server, per [Section 6 of \[RFC6749\]](#).

Implementations supporting OAuth bearer tokens need to factor in security considerations of this authorization method [[RFC7521](#)]. Since security is only as good as the weakest link, implementers also need to consider authentication choices coupled with OAuth bearer tokens. The security considerations of the default authentication method for OAuth bearer tokens, HTTP BASIC, are well documented in [[RFC7617](#)], therefore implementers are encouraged to prefer stronger authentication methods. Designating the specific methods of authentication and authorization are out of scope for the delivery of SETs, however this information is provided as a resource to implementers.

5. Privacy Considerations

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

SET Transmitters SHOULD attempt to deliver SETs that are 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, SET Transmitters and Recipients MUST have the appropriate legal agreements and user consent or terms of service in place.

The propagation of subject identifiers can be perceived as personally identifiable information. Where possible, SET Transmitters and Recipients SHOULD devise approaches that prevent propagation, for

example, the passing of a hash value that requires the subscriber to already know the subject.

6. IANA Considerations

There are no IANA considerations.

7. References

7.1. Normative References

- [I-D.ietf-secevent-http-push]
Backman, A., Jones, M., Scurtescu, M., Ansari, M., and A. Nadalin, "Push-Based Security Event Token (SET) Delivery Using HTTP", [draft-ietf-secevent-http-push-04](#) (work in progress), January 2019.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), DOI 10.17487/RFC2119, March 1997, <<https://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, <<https://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, <<https://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, <<https://www.rfc-editor.org/info/rfc6125>>.
- [RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", [RFC 7231](#), DOI 10.17487/RFC7231, June 2014, <<https://www.rfc-editor.org/info/rfc7231>>.
- [RFC7515] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Signature (JWS)", [RFC 7515](#), DOI 10.17487/RFC7515, May 2015, <<https://www.rfc-editor.org/info/rfc7515>>.

- [RFC7516] Jones, M. and J. Hildebrand, "JSON Web Encryption (JWE)", [RFC 7516](#), DOI 10.17487/RFC7516, May 2015, <<https://www.rfc-editor.org/info/rfc7516>>.
- [RFC7519] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Token (JWT)", [RFC 7519](#), DOI 10.17487/RFC7519, May 2015, <<https://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, <<https://www.rfc-editor.org/info/rfc7525>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in [RFC 2119](#) Key Words", [BCP 14](#), [RFC 8174](#), DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, [RFC 8259](#), DOI 10.17487/RFC8259, December 2017, <<https://www.rfc-editor.org/info/rfc8259>>.
- [RFC8417] Hunt, P., Ed., Jones, M., Denniss, W., and M. Ansari, "Security Event Token (SET)", [RFC 8417](#), DOI 10.17487/RFC8417, July 2018, <<https://www.rfc-editor.org/info/rfc8417>>.

[7.2.](#) Informative References

- [RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", [RFC 3339](#), DOI 10.17487/RFC3339, July 2002, <<https://www.rfc-editor.org/info/rfc3339>>.
- [RFC6202] Loreto, S., Saint-Andre, P., Salsano, S., and G. Wilkins, "Known Issues and Best Practices for the Use of Long Polling and Streaming in Bidirectional HTTP", [RFC 6202](#), DOI 10.17487/RFC6202, April 2011, <<https://www.rfc-editor.org/info/rfc6202>>.
- [RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", [RFC 6749](#), DOI 10.17487/RFC6749, October 2012, <<https://www.rfc-editor.org/info/rfc6749>>.
- [RFC6750] Jones, M. and D. Hardt, "The OAuth 2.0 Authorization Framework: Bearer Token Usage", [RFC 6750](#), DOI 10.17487/RFC6750, October 2012, <<https://www.rfc-editor.org/info/rfc6750>>.

- [RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", [RFC 7230](#), DOI 10.17487/RFC7230, June 2014, <<https://www.rfc-editor.org/info/rfc7230>>.
- [RFC7235] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Authentication", [RFC 7235](#), DOI 10.17487/RFC7235, June 2014, <<https://www.rfc-editor.org/info/rfc7235>>.
- [RFC7521] Campbell, B., Mortimore, C., Jones, M., and Y. Goland, "Assertion Framework for OAuth 2.0 Client Authentication and Authorization Grants", [RFC 7521](#), DOI 10.17487/RFC7521, May 2015, <<https://www.rfc-editor.org/info/rfc7521>>.
- [RFC7617] Reschke, J., "The 'Basic' HTTP Authentication Scheme", [RFC 7617](#), DOI 10.17487/RFC7617, September 2015, <<https://www.rfc-editor.org/info/rfc7617>>.

[Appendix A.](#) Acknowledgments

The editors would like to thank the members of the SCIM working group, which began discussions of provisioning events starting with [draft-hunt-scim-notify-00](#) in 2015.

The editors would like to thank Phil Hunt and the other the authors of [draft-ietf-secevent-delivery-02](#), on which this draft is based.

The editors would like to thank the participants in the SecEvents working group for their contributions to this specification.

[Appendix B.](#) Change Log

[[to be removed by the RFC Editor before publication as an RFC]]

Draft 00 - AB - Based on [draft-ietf-secevent-delivery-02](#) with the following additions:

- o Renamed to "Poll-Based SET Token Delivery Using HTTP"
- o Removed references to the HTTP Push delivery method.

Draft 01 - mbj:

- o Addressed problems identified in my 18-Jul-18 review message titled "Issues for both the Push and Poll Specs".

- o Changes to align terminology with [RFC 8417](#), for instance, by using the already defined term SET Recipient rather than SET Receiver.
- o Applied editorial and minor normative corrections.
- o Updated Marius' contact information.
- o Begun eliminating redundancies between this specification and "Push-Based Security Event Token (SET) Delivery Using HTTP" [[I-D.ietf-secevent-http-push](#)], referencing, rather than duplicating common normative text.

Draft 02 - mbj:

- o Removed vestigial language remaining from when the push and poll delivery methods were defined in a common specification.
- o Replaced remaining uses of the terms Event Transmitter and Event Recipient with the correct terms SET Transmitter and SET Recipient.
- o Removed uses of the unnecessary term "Event Stream".
- o Removed dependencies between the semantics of "maxEvents" and "returnImmediately".
- o Said that PII in SETs is to be encrypted with TLS, JWE, or both.
- o Corrected grammar and spelling errors.

Authors' Addresses

Annabelle Backman (editor)
Amazon

Email: richanna@amazon.com

Michael B. Jones (editor)
Microsoft

Email: mbj@microsoft.com

URI: <http://self-issued.info/>

Marius Scurtescu
Coinbase

Email: marius.scurtescu@coinbase.com

Morteza Ansari
Cisco

Email: morteza.ansari@cisco.com

Anthony Nadalin
Microsoft

Email: tonynad@microsoft.com