

Security Events Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: July 27, 2019

A. Backman, Ed.  
Amazon  
M. Jones, Ed.  
Microsoft  
M. Scurtescu  
Coinbase  
M. Ansari  
Cisco  
A. Nadalin  
Microsoft  
January 23, 2019

**Push-Based Security Event Token (SET) Delivery Using HTTP  
draft-ietf-secevent-http-push-04**

Abstract

This specification defines how a Security Event Token (SET) may be delivered to an intended recipient using HTTP POST. The SET is transmitted in the body of an HTTP POST request to an endpoint operated by the recipient, and the recipient indicates successful or failed transmission via the HTTP response.

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 July 27, 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

<a href="#">1.</a>	Introduction and Overview . . . . .	<a href="#">2</a>
<a href="#">1.1.</a>	Notational Conventions . . . . .	<a href="#">3</a>
<a href="#">1.2.</a>	Definitions . . . . .	<a href="#">3</a>
<a href="#">2.</a>	SET Delivery . . . . .	<a href="#">3</a>
<a href="#">2.1.</a>	Transmitting a SET . . . . .	<a href="#">4</a>
<a href="#">2.2.</a>	Success Response . . . . .	<a href="#">5</a>
<a href="#">2.3.</a>	Failure Response . . . . .	<a href="#">6</a>
<a href="#">2.4.</a>	Security Event Token Delivery Error Codes . . . . .	<a href="#">7</a>
<a href="#">3.</a>	Authentication and Authorization . . . . .	<a href="#">8</a>
<a href="#">4.</a>	Delivery Reliability . . . . .	<a href="#">8</a>
<a href="#">5.</a>	Security Considerations . . . . .	<a href="#">9</a>
<a href="#">5.1.</a>	Authentication Using Signed SETs . . . . .	<a href="#">9</a>
<a href="#">5.2.</a>	TLS Support Considerations . . . . .	<a href="#">9</a>
<a href="#">5.3.</a>	Denial of Service . . . . .	<a href="#">9</a>
<a href="#">5.4.</a>	Authenticating Persisted SETs . . . . .	<a href="#">10</a>
<a href="#">6.</a>	Privacy Considerations . . . . .	<a href="#">10</a>
<a href="#">7.</a>	IANA Considerations . . . . .	<a href="#">10</a>
<a href="#">7.1.</a>	Security Event Token Delivery Error Codes . . . . .	<a href="#">10</a>
<a href="#">7.1.1.</a>	Registration Template . . . . .	<a href="#">10</a>
<a href="#">7.1.2.</a>	Initial Registry Contents . . . . .	<a href="#">11</a>
<a href="#">8.</a>	References . . . . .	<a href="#">12</a>
<a href="#">8.1.</a>	Normative References . . . . .	<a href="#">12</a>
<a href="#">8.2.</a>	Informative References . . . . .	<a href="#">13</a>
<a href="#">Appendix A.</a>	Other Streaming Specifications . . . . .	<a href="#">13</a>
<a href="#">Appendix B.</a>	Acknowledgments . . . . .	<a href="#">15</a>
<a href="#">Appendix C.</a>	Change Log . . . . .	<a href="#">15</a>
	Authors' Addresses . . . . .	<a href="#">18</a>

## [1.](#) Introduction and Overview

This specification defines a mechanism by which a transmitter of a Security Event Token (SET) [[RFC8417](#)] may deliver the SET to an intended recipient via HTTP POST [[RFC7231](#)].

Push-Based SET Delivery over HTTP POST is intended for scenarios where all of the following apply:



- o The transmitter of the SET is capable of making outbound HTTP requests.
- o The recipient is capable of hosting an HTTP endpoint that is accessible to the transmitter.
- o The transmitter and recipient are known to one another.

A mechanism for exchanging configuration metadata such as endpoint URLs and cryptographic key parameters between the transmitter and receipt is out of scope for this specifications.

### **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 documents 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, as defined in [[RFC8417](#)].

#### **SET Recipient**

An entity that receives SETs through some distribution method, as defined in [[RFC8417](#)].

## **2. SET Delivery**

To deliver a SET to a given SET Recipient, the SET Transmitter makes a SET Transmission Request to the SET Recipient, with the SET itself contained within the request. The SET Recipient replies to this request with a response either acknowledging successful transmission of the SET, or indicating that an error occurred while receiving, parsing, and/or validating the SET.

Upon receipt of a SET, the SET Recipient SHALL validate that all of the following are true:



- o The SET Recipient can parse the SET.
- o The SET is authentic (i.e., it was issued by the issuer specified within the SET).
- o The SET Recipient is identified as an intended audience of the SET.
- o The SET issuer is recognized as an issuer that the SET Recipient is willing to receive SETs from (e.g., the issuer is whitelisted by the SET Recipient).

The mechanisms by which the SET Recipient performs this validation are out of scope for this document. SET parsing and issuer and audience identification are defined in [\[RFC8417\]](#). The mechanism for validating the authenticity of a SET is implementation specific, and may vary depending on the authentication mechanisms in use, and whether the SET is signed and/or encrypted (See [Section 3](#)).

The SET Recipient SHOULD ensure that the SET is persisted in a way that is sufficient to meet the SET Recipient's own reliability requirements, and MUST NOT expect or depend on a SET Transmitter to re-transmit or otherwise make available to the SET Recipient a SET once the SET Recipient acknowledges that it was received successfully.

Once the SET has been validated and persisted, the SET Recipient SHOULD immediately return a response indicating that the SET was successfully delivered. The SET Recipient SHOULD NOT perform extensive business logic that processes the event expressed by the SET prior to sending this response. Such logic SHOULD be executed asynchronously from delivery, in order to minimize the expense and impact of SET delivery on the SET Transmitter.

The SET Transmitter MAY re-transmit a SET if the responses from previous transmissions timed out or indicated potentially recoverable error (such as server unavailability that may be transient, or a decryption failure that may be due to misconfigured keys on the SET Recipient's side). In all other cases, the SET Transmitter SHOULD NOT re-transmit a SET. The SET Transmitter SHOULD delay retransmission for an appropriate amount of time to avoid overwhelming the SET Recipient (see [Section 4](#)).

## **[2.1. Transmitting a SET](#)**

To transmit a SET to a SET Recipient, the SET Transmitter makes an HTTP POST request to an HTTP endpoint provided by the SET Recipient. The "Content-Type" header of this request MUST be "application/



secevent+jwt" as defined in Sections [2.2](#) and [6.2](#) of [[RFC8417](#)], and the "Accept" header MUST be "application/json". The request body MUST consist of the SET itself, represented as a JWT [[RFC7519](#)].

The SET Transmitter MAY include in the request an "Accept-Language" header to indicate to the SET Recipient the preferred language(s) in which to receive error messages.

The mechanisms by which the SET Transmitter determines the HTTP endpoint to use when transmitting a SET to a given SET Recipient are not defined by this specification and may be implementation-specific.

The following is a non-normative example of a SET transmission request:

```
POST /Events HTTP/1.1
Host: notify.rp.example.com
Accept: application/json
Accept-Language: en-US, en;q=0.5
Content-Type: application/secevent+jwt

eyJhbGciOiJub25lIn0
.
eyJwdWJsaXNoZXJvcmk0iJodHRwczovL3NjaW0uZXhhbXBsZS5jb20iLCJmZWV
kVXJpcyI6WyJodHRwczovL2podWiuZXhhbXBsZS5jb20vRmVlZHMvOThkNTI0Nj
FmYTViYmM4Nzk1OTNiNzc1NCIsImh0dHBzOi8vamh1Yi5leGFtcGxlLmNvbS9GZ
WVkey81ZDc2MDQ1MTZiMTZiMTZiMTZiMTZiMTZiMTZiMTZiMTZiMTZiMTZiMTZi
WyJodHRwczovL3NjaW0uZXhhbXBsZS5jb20vVXNlcnMvNDRmNjE0MmRmOTZiZDZ
hYjYxZTc1MjFkOSJdLCJldmVudFR5cGVzIjpbIkNSRUFURSJdLCJhdHRyaWJ1dG
VzIjpbImkIiwibmFtZSI6ImVzZXJ0YW11IiwicGFzc3dvcmQiLCJlbWVpbnMiX
SwidmFsdWVzIjpbImVtYWlsYyI6W3sidHlwZSI6IndvcmsiLCJ2YX1ZSI6Impk
b2VAZXhhbXBsZS5jb20ifV0sInBhc3N3b3JkIjoibm90NHUybm8iLCJ1c2VyTmF
tZSI6Impkb2UiLCJpZCI6IjQ0ZjYxNDJkZjY2YmQ2YWI2MWU3NTIxZDkiLCJ1Y
1IjpbImdpdmVuTmFtZSI6Ikpvag4iLCJmYW1pbHl0YW11IjoiriRG9lIn19fQ
.
c2lnbmF0dXJl
```

Figure 1: Example SET Transmission Request

## 2.2. Success Response

If the SET is determined to be valid, the SET Recipient SHALL "acknowledge" successful transmission by responding with HTTP Response Status Code 202 (Accepted) (see [Section 6.3.3 of \[RFC7231\]](#)). The body of the response MUST be empty.





The following is a non-normative example of a successful receipt of a SET.

HTTP/1.1 202 Accepted

Figure 2: Example Successful Delivery Response

Note that the purpose of the "acknowledgement" response is to let the SET Transmitter know that a SET has been delivered and the information no longer needs to be retained by the SET Transmitter. Before acknowledgement, SET Recipients SHOULD ensure they have validated received SETs and retained them in a manner appropriate to information retention requirements appropriate to the SET event types signaled. The level and method of retention of SETs by SET Recipients is out of scope of this specification.

### **2.3. Failure Response**

In the event of a general HTTP error condition, the SET Recipient MAY respond with an appropriate HTTP Status Code as defined in [Section 6 of \[RFC7231\]](#).

When the SET Recipient detects an error parsing or validating a SET transmitted in a SET Transmission Request, the SET Recipient SHALL respond with an HTTP Response Status Code of 400 (Bad Request). The "Content-Type" header of this response MUST be "application/json", and the body MUST be a UTF-8 encoded JSON [\[RFC7159\]](#) object containing the following name/value pairs:

err A Security Event Token Error Code (see [Section 2.4](#)).

description A UTF-8 string containing a human-readable description of the error that MAY provide additional diagnostic information. The exact content of this field is implementation-specific.

The response MUST include a "Content-Language" header, whose value indicates the language of the error descriptions included in the response body. If the SET Recipient can provide error descriptions in multiple languages, they SHOULD choose the language to use according to the value of the "Accept-Language" header sent by the SET Transmitter in the transmission request, as described in [Section 5.3.5 of \[RFC7231\]](#). If the SET Transmitter did not send an "Accept-Language" header, or if the SET Recipient does not support any of the languages included in the header, the SET Recipient MUST respond with messages that are understandable by an English-speaking person, as described in [Section 4.5 of \[RFC2277\]](#).



The following is an example non-normative error response indicating that the key used to encrypt the SET has been revoked.

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

{
  "err": "invalid_key",
  "description": "Key ID 12345 has been revoked."
}
```

Figure 3: Example Error Response (invalid\_key)

The following is an example non-normative error response indicating that the access token included in the request is expired.

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

{
  "err": "authentication_failed",
  "description": "Access token is expired."
}
```

Figure 4: Example Error Response (authentication\_failed)

The following is an example non-normative error response indicating that the SET Transmitter is not permitted to transmit SETs issued by the issuer specified in the SET.

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

{
  "err": "access_denied",
  "description": "Not authorized for issuer http://iss.example.com/."
}
```

Figure 5: Example Error Response (access\_denied)

#### **[2.4.](#) Security Event Token Delivery Error Codes**

Security Event Token Delivery Error Codes are strings that identify a specific category of error that may occur when parsing or validating a SET. Every Security Event Token Delivery Error Code MUST have a



unique name registered in the IANA "Security Event Token Delivery Error Codes" registry established by [Section 7.1](#).

The following table presents the initial set of Error Codes that are registered in the IANA "Security Event Token Delivery Error Codes" registry:

Error Code	Description
invalid_request	The request body cannot be parsed as a SET, or the event payload within the SET does not conform to the event's definition.
invalid_key	One or more keys used to encrypt or sign the SET is invalid or otherwise unacceptable to the SET Recipient. (e.g., expired, revoked, failed certificate validation, etc.)
authentication_failed	The SET Recipient could not authenticate the SET Transmitter from the contents of the request.
access_denied	The SET Transmitter is not authorized to transmit the provided SET to the SET Recipient.

Table 1: SET Delivery Error Codes

### [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 [\[RFC7235\]](#).

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 other employed authentication methods. Because SETs are not commands, SET Recipients are free to ignore SETs that are not of interest.

### [4. Delivery Reliability](#)

Delivery reliability requirements may vary from implementation to implementation. This specification defines the response from the SET Recipient in such a way as to provide the SET Transmitter with the information necessary to determine what further action is required, if any, in order to meet their requirements. SET Transmitters with



high reliability requirements may be tempted to always retry failed transmissions, however it should be noted that for many types of SET delivery errors, a retry is extremely unlikely to be successful. For example, "invalid\_request" indicates a structural error in the content of the request body that is likely to remain when re-transmitting the same SET. Others such as "access\_denied" may be transient, for example if the SET Transmitter refreshes expired credentials prior to re-transmission.

Implementers SHOULD evaluate their reliability requirements and the impact of various retry mechanisms on the performance of their systems to determine the correct strategy for various error conditions.

## **5. Security Considerations**

### **5.1. 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.

### **5.2. 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 require the use of a transport-layer security mechanism. 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].

### **5.3. Denial of Service**

The SET Recipient may be vulnerable to a denial-of-service attack where a malicious party makes a high volume of requests containing invalid SETs, causing the endpoint to expend significant resources on cryptographic operations that are bound to fail. This may be mitigated by authenticating SET Transmitters with a mechanism with low runtime overhead, such as mutual TLS.





#### **[5.4.](#) Authenticating Persisted SETs**

At the time of receipt, the SET Recipient can rely upon transport layer mechanisms, HTTP authentication methods, and/or other context from the transmission request to authenticate the SET Transmitter and validate the authenticity of the SET. However, this context is typically unavailable to systems that the SET Recipient forwards the SET onto, or to systems that retrieve the SET from storage. If the SET Recipient requires the ability to validate SET authenticity outside of the context of the transmission request, then the SET Transmitter SHOULD sign the SET in accordance with [[RFC7515](#)] and optionally also encrypt it in accordance with [[RFC7516](#)].

### **[6.](#) Privacy Considerations**

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

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.

### **[7.](#) IANA Considerations**

#### **[7.1.](#) Security Event Token Delivery Error Codes**

This document defines Security Event Token Delivery Error Codes, for which IANA is asked to create and maintain a new registry titled "Security Event Token Delivery Error Codes". Initial values for the Security Event Token Delivery Error Codes registry are given in Table 1. Future assignments are to be made through the Expert Review registration policy ([[RFC8126](#)]) and shall follow the template presented in [Section 7.1.1](#).

##### **[7.1.1.](#) Registration Template**

###### **Error Code**

The name of the Security Event Token Delivery Error Code, as described in [Section 2.4](#). The name MUST be a case-sensitive ASCII string consisting only of upper-case letters ("A" - "Z"), lower-case letters ("a" - "z"), and digits ("0" - "9").



**Description**

A brief human-readable description of the Security Event Token Delivery Error Code.

**Change Controller**

For error codes registered by the IETF or its working groups, list "IETF Secevent Working Group". For all other error codes, list the name of the party responsible for the registration. Contact information such as mailing address, email address, or phone number may also be provided.

**Defining Document(s)**

A reference to the document or documents that define the Security Event Token Delivery Error Code. The definition **MUST** specify the name and description of the error code, and explain under what circumstances the error code may be used. URIs that can be used to retrieve copies of each document at no cost **SHOULD** be included.

**7.1.2. Initial Registry Contents**

Error Code: invalid\_request

Description: The request body cannot be parsed as a SET, or the event payload within the SET does not conform to the event's definition.

Change Controller: IETF Secevent Working Group

Defining Document(s): [Section 2.4](#) of this document

Error Code: invalid\_key

Description: One or more keys used to encrypt or sign the SET is invalid or otherwise unacceptable to the SET Recipient. (e.g., expired, revoked, failed certificate validation, etc.)

Change Controller: IETF Secevent Working Group

Defining Document(s): [Section 2.4](#) of this document

Error Code: authentication\_failed

Description: The SET Recipient could not authenticate the SET Transmitter from the contents of the request.

Change Controller: IETF Secevent Working Group

Defining Document(s): [Section 2.4](#) of this document

Error Code: access\_denied

Description: The SET Transmitter is not authorized to transmit the provided SET to the SET Recipient.

Change Controller: IETF Secevent Working Group

Defining Document(s): [Section 2.4](#) of this document



## 8. References

### 8.1. Normative References

- [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>>.
- [RFC2277] Alvestrand, H., "IETF Policy on Character Sets and Languages", [BCP 18](#), [RFC 2277](#), DOI 10.17487/RFC2277, January 1998, <<https://www.rfc-editor.org/info/rfc2277>>.
- [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>>.
- [RFC7159] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", [RFC 7159](#), DOI 10.17487/RFC7159, March 2014, <<https://www.rfc-editor.org/info/rfc7159>>.
- [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>>.
- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", [BCP 26](#), [RFC 8126](#), DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.
- [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>>.
- [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>>.

## **8.2. Informative References**

- [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>>.

## **Appendix A. Other Streaming Specifications**

[[EDITORS NOTE: This section to be removed prior to publication]]

The following pub/sub, queuing, streaming systems were reviewed as possible solutions or as input to the current draft:

### XMPP Events

The WG considered the XMPP events and its ability to provide a single messaging solution without the need for both polling and push modes. The feeling was the size and methodology of XMPP was too far apart from the current capabilities of the SECEVENTs community which focuses in on HTTP based service delivery and authorization.

### Amazon Simple Notification Service

Simple Notification Service, is a pub/sub messaging product from AWS. SNS supports a variety of subscriber types: HTTP/HTTPS endpoints, AWS Lambda functions, email addresses (as JSON or plain text), phone numbers (via SMS), and AWS SQS standard queues. It doesn't directly





support pull, but subscribers can get the pull model by creating an SQS queue and subscribing it to the topic. Note that this puts the cost of pull support back onto the subscriber, just as it is in the push model. It is not clear that one way is strictly better than the other; larger, sophisticated developers may be happy to own message persistence so they can have their own internal delivery guarantees. The long tail of OIDC clients may not care about that, or may fail to get it right. Regardless, I think we can learn something from the Delivery Policies supported by SNS, as well as the delivery controls that SQS offers (e.g., Visibility Timeout, Dead-Letter Queues). I'm not suggesting that we need all of these things in the spec, but they give an idea of what features people have found useful.

Other information:

- o API Reference:  
<http://docs.aws.amazon.com/AWSSimpleQueueService/latest/APIReference/Welcome.html>
- o Visibility Timeouts:  
<http://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/sqs-visibility-timeout.html>

#### Apache Kafka

Apache Kafka is an Apache open source project based upon TCP for distributed streaming. It prescribes some interesting general purpose features that seem to extend far beyond the simpler streaming model SECEVENTs is after. A comment from MS has been that Kafka does an acknowledge with poll combination event which seems to be a performance advantage. See: <https://kafka.apache.org/intro>

#### Google Pub/Sub

Google Pub Sub system favours a model whereby polling and acknowledgement of events is done as separate endpoints as separate functions.

Information:

- o Cloud Overview - <https://cloud.google.com/pubsub/>
- o Subscriber Overview - <https://cloud.google.com/pubsub/docs/subscriber>
- o Subscriber Pull(poll) - <https://cloud.google.com/pubsub/docs/pull>



## **[Appendix B](#). 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 authors of [draft-ietf-secevent-delivery-02](#), on which this draft is based.

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

## **[Appendix C](#). Change Log**

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

- o Renamed to "Push-Based SET Token Delivery Using HTTP"
- o Removed references to the HTTP Polling delivery method.
- o Removed informative reference to [RFC6202](#).

Draft 01 - AB:

- o Fixed area and workgroup to match secevent.
- o Removed unused definitions and definitions already covered by SET.
- o Renamed Event Transmitter and Event Receiver to SET Transmitter and SET Receiver, respectively.
- o Added IANA registry for SET Delivery Error Codes.
- o Removed enumeration of HTTP authentication methods.
- o Removed generally applicable guidance for HTTP, authorization tokens, and bearer tokens.
- o Moved guidance for using authentication methods as DoS protection to Security Considerations.
- o Removed redundant instruction to use WWW-Authenticate header.
- o Removed further generally applicable guidance for authorization tokens.



- o Removed bearer token from example delivery request, and text referencing it.
- o Broke delivery method description into separate request/response sections.
- o Added missing empty line between headers and body in example request.
- o Removed unapplicable notes about example formatting.
- o Removed text about SET creation and handling.
- o Removed duplication in protocol description.
- o Added "non-normative example" text to example transmission request.
- o Fixed inconsistencies in use of Error Code term.

Draft 02 - AB:

- o Rewrote abstract and introduction.
- o Rewrote definitions for SET Transmitter, SET Receiver.
- o Renamed Event Delivery section to SET Delivery.
- o Readability edits to Success Response and Failure Response sections.
- o Consolidated definition of error response under Failure Response section.
- o Removed Event Delivery Process section and moved its content to parent section.
- o Readability edits to SET Delivery section and its subsections.
- o Added callout that SET Receiver HTTP endpoint configuration is out-of-scope.
- o Added callout that SET verification mechanisms are out-of-scope.
- o Added retry guidance, notes regarding delivery reliability requirements.



- o Added guidance around using JWS and/or JWE to authenticate persisted SETs.

Draft 03 - 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.

Draft 04 - AB:

- o Replaced Error Codes with smaller set of meaningfully differentiated codes.
- o Added more error response examples.
- o Removed un-referenced normative references.
- o Added normative reference to JSON in error response definition.
- o Added text clarifying that the value of the "description" attribute in error responses is implementation specific.
- o Added requirement that error descriptions and responses are UTF-8 encoded.
- o Added error description language preferences and specification via "Accept-Language" and "Content-Language" headers.
- o Added "recognized issuer" validation requirement in [section 2](#).
- o Added time outs as an acceptable reason to resend a SET in [section 2](#).
- o Edited text in [section 1](#) to clarify that configuration is out of scope.
- o Made minor editorial corrections.





Authors' Addresses

Annabelle Backman (editor)  
Amazon

Email: [richanna@amazon.com](mailto:richanna@amazon.com)

Michael B. Jones (editor)  
Microsoft

Email: [mbj@microsoft.com](mailto:mbj@microsoft.com)

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

Marius Scurtescu  
Coinbase

Email: [marius.scurtescu@coinbase.com](mailto:marius.scurtescu@coinbase.com)

Morteza Ansari  
Cisco

Email: [morteza.ansari@cisco.com](mailto:morteza.ansari@cisco.com)

Anthony Nadalin  
Microsoft

Email: [tonynad@microsoft.com](mailto:tonynad@microsoft.com)

