Security Events Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 4, 2019

A. Backman, Ed.
Amazon
M. Jones, Ed.
Microsoft
M. Scurtescu
Google
M. Ansari
Cisco
A. Nadalin
Microsoft

October 1, 2018

Push-Based SET Token Delivery Using HTTP
draft-ietf-secevent-http-push-02

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 reqest 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 April 4, 2019.

Copyright Notice

Table of Contents

1. Introduction and Overview 2
1.1. Notational Conventions 3
1.2. Definitions 3
2. SET Delivery 3
2.1. Transmitting a SET 4
2.2. Success Response 5
2.3. Failure Response 6
2.4. Security Event Token Delivery Error Codes 6
3. Authentication and Authorization 7
4. Delivery Reliability 7
5. Security Considerations 8
5.1. Authentication Using Signed SETs 8
5.2. TLS Support Considerations 8
5.3. Denial of Service 8
5.4. Authenticating Persisted SETs 8
6. Privacy Considerations 9
7. IANA Considerations 9
7.1. Security Event Token Delivery Error Codes 9
7.1.1. Registration Template 9
7.1.2. Initial Registry Contents 10
8. References 13
8.1. Normative References 13
8.2. Informative References 15
Appendix A. Other Streaming Specifications 16
Appendix B. Acknowledgments 17
Appendix C. Change Log 18
Authors' Addresses 19

1. Introduction and Overview

This specification defines a mechanism by which a holder 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 holder of the SET is capable of making outbound HTTP requests.

- The recipient is capable of hosting an HTTP endpoint that is accessible to the transmitter.
- The transmitter and recipient are known to one another.
- The transmitter and recipient have an out-of-band mechanism for exchanging configuration metadata such as endpoint URLs and security key parameters.

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 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 Receivers.

SET Receiver
: An entity that operates an endpoint where it expects to receive SETs from one or more SET Transmitters.

2. SET Delivery

To deliver a SET to a given SET Receiver, the SET Transmitter makes a SET Transmission Request to the SET Receiver, with the SET itself contained within the request. The SET Receiver 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 Receiver SHALL validate that all of the following are true:

- The SET Receiver can parse the SET.

- The SET is authentic (i.e. it was issued by the issuer specified within the SET).

- The SET Receiver is identified as the intended audience of the SET.

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

Once the SET has been validated and persisted, the SET Receiver SHOULD immediately return a response indicating that the SET was successfully delivered. The SET Receiver 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 SHOULD NOT re-transmit a SET, unless the response from the SET Receiver in previous transmissions indicated a 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 Receiver's side). In the latter case, the SET Transmitter MAY re-transmit a SET, after an appropriate delay to avoid overwhelming the SET Receiver (see Section 4).

The SET Transmitter MAY provide an out-of-band mechanism by which a SET Receiver may be notified of delivery failures, and MAY retain SETs that it failed to deliver and make them available to the SET Receiver via other means.

2.1. Transmitting a SET

To transmit a SET to a SET Receiver, the SET Transmitter makes an HTTP POST request to an HTTP endpoint provided by the SET Receiver. The "Content-Type" header of this request MUST be "application/secevent+jwt" as defined in Sections 2.2 and 6.2 of RFC8417 [RFC8417], and the "Accept" header MUST be "application/json". The

request body MUST consist of the SET itself, represented as a JWT [RFC7519].

The mechanisms by which the SET Transmitter determines the HTTP endpoint to use when transmitting a SET to a given SET Receiver 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.examplerp.com
Accept: application/json
Content-Type: application/secevent+jwt

eyJhbGciOiJub25lIn0
.
eyJwdWJsaXNoZXJVcmkiOiJodHRwczovL3NjaW0uZXhhbXBsZS5jb20iLCJmZWV
kVXJpcyI6WyJodHRwczovL2podWIuZXhhbXBsZS5jb20vRmVlZHMvOThkNTI0Nj
FmYTViYmM4Nzk1OTNiNzc1NCIsImh0dHBzOi8vamh1Yi5leGFtcGxlLmNvbS9GZ
WVkcy81ZDc2MDQ1MTZiMWQwODY0MWQ3Njc2ZWU3Il0sInJlc291cmNlVXJpcyI6
WyJodHRwczovL3NjaW0uZXhhbXBsZS5jb20vVXNlcnMvNDRmNjE0MmRmOTZiZDZ
hYjYxZTc1MjFkOSJdLCJldmVudFR5cGVzIjpbIkNSRUFURSJdLCJhdHRyaWJ1dG
VzIjpbImlkIiwibmFtZSIsInVzZXJOYW1lIiwicGFzc3dvcmQiLCJlbWFpbHMiX
SwidmFsdWVzIjp7ImVtYWlscyI6W3sidHlwZSI6Indvcmsi LCJ2YWx1ZSI6Impk
b2VAZXhhbXBsZS5jb20ifV0sInBhc3N3b3JkIjoibm90NHUybm8iLCJ1c2VyTmF
tZSI6Impkb2UiLCJpZCI6IjQ0ZjYxNDJkZjk2YmQ2YWI2MWU3NTIxZDkiLCJuYW
1lIjp7ImdpdmVuTmFtZSI6IkpvaG4iLCJmYW1pbHlOYW1lIjoiRG9lIn19fQ
.
```

Figure 1: Example SET Transmission Request

2.2. Success Response

If the SET is determined to be valid, the SET Receiver SHALL "acknowledge" successful transmission by responding with HTTP Response Status Code 202 (see Section 6.3.3 of RFC7231 [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 Receivers 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 Receivers is out-of-scope of this specification.

2.3. Failure Response

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

When the SET Receiver detects an error parsing or validating a SET transmitted in a SET Transmission Request, the SET Receiver SHALL respond with an HTTP Response Status Code of 400. The "Content-Type" header of this response MUST be "application/json", and the body MUST be a JSON object containing the following name/value pairs:

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

description Human-readable text that describes the error and MAY contain additional diagnostic information.

The following is an example non-normative error response.

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

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

}
```

Figure 3: Example Error Response

2.4. Security Event Token Delivery Error Codes

Security Event Token Delivery Error Codes are strings that identify a specific type 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
json	Invalid JSON object.
jwtParse	Invalid or unparsable JWT or JSON structure.
jwtHdr	In 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.
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.

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 (see), SET Receivers 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 Receiver 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, "json", "jwtParse", and "setParse" all indicate structural errors in the content of the SET that are likely to remain when re-transmitting the same SET. Others such as "jws" or "jwe" may be

transient, for example if cryptographic material has not been properly distributed to the SET Receiver's systems.

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 Receiver to validate that the SET issuer is authorized to deliver SETs.

5.2. TLS Support Considerations

SETs contain sensitive information that is considered PII (e.g. subject claims). Therefore, SET Transmitters and SET Receivers 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 Receiver 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 or statically assigned bearer tokens.

5.4. Authenticating Persisted SETs

At the time of receipt, the SET Receiver 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 Receiver forwards the SET onto, or to systems that retrieve the SET from storage. If the

SET Receiver 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], or encrypt it using an authenticated encryption scheme in accordance with [RFC7516].

6. Privacy Considerations

If a SET needs to be retained for audit purposes, JWS 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 Receivers 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 Receivers 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
  json

  Description
  Invalid JSON object.

  Change Controller
  IETF Secevent Working Group

  Defining Document(s)
  Section 2.4 of this document.

- Error Code
  jwtParse

  Description
  Invalid or unparsable JWT or JSON structure.

  Change Controller
  IETF Secevent Working Group

  Defining Document(s)
  Section 2.4 of this document.

- Error Code
  jwtHdr

Description
An invalid JWT header was detected.

Change Controller
IETF Secevent Working Group

Defining Document(s)
Section 2.4 of this document.

o

Error Code
jwtCrypto

Description
Unable to parse due to unsupported algorithm.

Change Controller
IETF Secevent Working Group

Defining Document(s)
Section 2.4 of this document.

o

Error Code
jws

Description
Signature was not validated.

Change Controller
IETF Secevent Working Group

Defining Document(s)
Section 2.4 of this document.

o

Error Code
jwe

Description
Unable to decrypt JWE encoded data.

Change Controller
IETF Secevent Working Group

Defining Document(s)
Section 2.4 of this document.

o

Error Code
jwtAud

Description
Invalid audience value.

Change Controller
IETF Secevent Working Group

Defining Document(s)
Section 2.4 of this document.

o

Error Code
jwtIss

Description
Issuer not recognized.

Change Controller

Defining Document(s)
Section 2.4 of this document.

o

Error Code
setType

Description
An unexpected Event type was received.

Change Controller
IETF Secevent Working Group

Defining Document(s)
Section 2.4 of this document.

o

Error Code
setParse

Description
Invalid structure was encountered such as an inability to parse or an incomplete set of Event claims.

Change Controller
IETF Secevent Working Group

Defining Document(s)
Section 2.4 of this document.

o

Error Code
setData

Description
SET event claims incomplete or invalid.

Change Controller
IETF Secevent Working Group

Defining Document(s)
Section 2.4 of this document.

o

Error Code
dup

Description
A duplicate SET was received and has been ignored.

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>.

[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>.

[RFC5988] Nottingham, M., "Web Linking", RFC 5988, DOI 10.17487/RFC5988, October 2010, <https://www.rfc-editor.org/info/rfc5988>.

[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>.

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

[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

[openid-connect-core] NRI, "OpenID Connect Core 1.0", Nov 2014.

[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>.

[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>.

[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>.

[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>.

[saml-core-2.0] Internet2, "Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0", March 2005.

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 ands 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 to 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:

- API Reference: http://docs.aws.amazon.com/AWSSimpleQueueService/latest/APIReference/Welcome.html

- 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:

- Cloud Overview - https://cloud.google.com/pubsub/

- Subscriber Overview - https://cloud.google.com/pubsub/docs/subscriber

- Subscriber Pull(poll) - https://cloud.google.com/pubsub/docs/pull

Appendix B. Acknowledgments

The editors would like to thanks the members of the SCIM WG 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 editor would like to thank the participants in the the SECEVENTS working group for their support of this specification.

Appendix C. Change Log

Draft 00 - AB - Based on draft-ietf-secevent-delivery-02 with the following changes:

- Renamed to "Push-Based SET Token Delivery Using HTTP"
- Removed references to the HTTP Polling delivery method.
- Removed informative reference to RFC6202.

Draft 01 - AB:

- Fixed area and workgroup to match secevent.
- Removed unused definitions and definitions already covered by SET.
- Renamed Event Transmitter and Event Receiver to SET Transmitter and SET Receiver, respectively.
- Added IANA registry for SET Delivery Error Codes.
- Removed enumeration of HTTP authentication methods.
- Removed generally applicable guidance for HTTP, authorization tokens, and bearer tokens.
- Moved guidance for using authentication methods as DoS protection to Security Considerations.
- Removed redundant instruction to use WWW-Authenticate header.
- Removed further generally applicable guidance for authorization tokens.
- Removed bearer token from example delivery request, and text referencing it.
- Broke delivery method description into separate request/response sections.
- Added missing empty line between headers and body in example request.
- Removed unapplicable notes about example formatting.
- Removed text about SET creation and handling.

- Removed duplication in protocol description.
- Added "non-normative example" text to example transmission request.
- Fixed inconsistencies in use of Error Code term.

Draft 02 - AB:

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

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
Google

Email: mscurtescu@google.com

Morteza Ansari
Cisco

Email: morteza.ansari@cisco.com

Anthony Nadalin
Microsoft

Email: tonynad@microsoft.com