

NETCONF  
Internet-Draft  
Intended status: Standards Track  
Expires: August 4, 2018

E. Voit  
A. Tripathy  
E. Nilsen-Nygaard  
Cisco Systems  
A. Clemm  
Huawei  
A. Gonzalez Prieto  
VMWare  
A. Bierman  
YumaWorks  
January 31, 2018

**RESTCONF and HTTP Transport for Event Notifications**  
**draft-ietf-netconf-restconf-notif-04**

Abstract

This document defines RESTCONF, HTTP2, and HTTP1.1 bindings for the transport of subscription requests and corresponding push updates. Being subscribed may be either publisher defined event streams or nodes/subtrees of YANG Datastores.

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 August 4, 2018.

Copyright Notice

Copyright (c) 2018 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 . . . . .	<a href="#">2</a>
<a href="#">2.</a>	Terminology . . . . .	<a href="#">3</a>
<a href="#">3.</a>	Solution . . . . .	<a href="#">3</a>
<a href="#">3.1.</a>	Dynamic YANG Subscription with RESTCONF control . . . . .	<a href="#">3</a>
<a href="#">4.</a>	Mandatory JSON and datastore support . . . . .	<a href="#">6</a>
<a href="#">5.</a>	Notification Messages . . . . .	<a href="#">7</a>
<a href="#">6.</a>	Security Considerations . . . . .	<a href="#">7</a>
<a href="#">7.</a>	Acknowledgments . . . . .	<a href="#">7</a>
<a href="#">8.</a>	References . . . . .	<a href="#">7</a>
<a href="#">8.1.</a>	Normative References . . . . .	<a href="#">7</a>
<a href="#">8.2.</a>	Informative References . . . . .	<a href="#">8</a>
<a href="#">Appendix A.</a>	End-to-End Deployment Guidance . . . . .	<a href="#">9</a>
<a href="#">A.1.</a>	Call Home . . . . .	<a href="#">9</a>
<a href="#">A.2.</a>	TLS Heartbeat . . . . .	<a href="#">9</a>
<a href="#">Appendix B.</a>	RESTCONF over GRPC . . . . .	<a href="#">9</a>
<a href="#">Appendix C.</a>	Encoded Subscription and Notification Message Examples . . . . .	<a href="#">10</a>
<a href="#">C.1.</a>	RESTCONF Subscription and Events over HTTP1.1 . . . . .	<a href="#">10</a>
<a href="#">C.2.</a>	Event Notification over HTTP2 . . . . .	<a href="#">14</a>
<a href="#">Appendix D.</a>	Changes between revisions . . . . .	<a href="#">14</a>
	Authors' Addresses . . . . .	<a href="#">15</a>

## [1.](#) Introduction

Mechanisms to support event subscription and push are defined in [I-D.[draft-ietf-netconf-subscribed-notifications](#)]. Enhancements to [I-D.[draft-ietf-netconf-subscribed-notifications](#)] which enable YANG Datastore subscription and push are defined in [I-D.[ietf-netconf-yang-push](#)]. This document provides a transport specification for these protocols over RESTCONF and HTTP. Driving these requirements is [[RFC7923](#)].

The streaming of notifications encapsulating the resulting information push can be done with either HTTP1.1 and HTTP2. When using HTTP2 [[RFC7540](#)] benefits which can be realized include:

- o Elimination of head-of-line blocking



- o Weighting and proportional dequeuing of Events from different subscriptions
- o Explicit precedence in subscriptions so that events from one subscription must be sent before another dequeues

## 2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC 2119](#) [[RFC2119](#)].

The following terms use the definitions from [[I-D.draft-ietf-netconf-subscribed-notifications](#)]: configured subscription, dynamic subscription, notification message, publisher, receiver, subscriber, and subscription.

## 3. Solution

Subscribing to event streams is defined in [[I-D.draft-ietf-netconf-subscribed-notifications](#)], YANG Datastore subscription is defined in [[I-D.ietf-netconf-yang-push](#)]. This section specifies transport mechanisms applicable to both.

### 3.1. Dynamic YANG Subscription with RESTCONF control

Dynamic subscriptions for both [[I-D.draft-ietf-netconf-subscribed-notifications](#)] and its [[I-D.ietf-netconf-yang-push](#)] augmentations are configured and managed via signaling messages transported over [[RFC8040](#)]. These interactions will be accomplished via a RESTCONF POST into RPCs located on the publisher. HTTP responses codes will indicate the results of the interaction with the publisher. An HTTP status code of 200 is the proper response to a successful <establish-subscription> RPC call. The successful <establish-subscription> will result in a HTTP message with returned subscription URI on a logically separate mechanism than was used for the original RESTCONF POST. This mechanism is via a parallel TCP connection in the case of HTTP 1.x, or in the case of HTTP2 via a separate HTTP stream within the HTTP connection. When a being returned by the publisher, failure will be indicated by 4xx range status codes transported in payload. Anytime hints are returned from the publisher status code 412 is used with the error-tag "operation-failed".

Once established, the resulting stream of notification messages are then delivered via SSE for HTTP1.1 and via an HTTP2 DATA frame for HTTP2.



### 3.1.1. Call Flow for HTTP2

Requests to [I-D.[draft-ietf-netconf-subscribed-notifications](#)] or [I-D.[ietf-netconf-yang-push](#)] augmented RPCs are sent on one or more HTTP2 streams indicated by (a) in Figure 2. Notification messages related to a single subscription are pushed on a unique logical channel (b). In the case below, a newly established subscription has its associated messages pushed over HTTP2 stream (7).

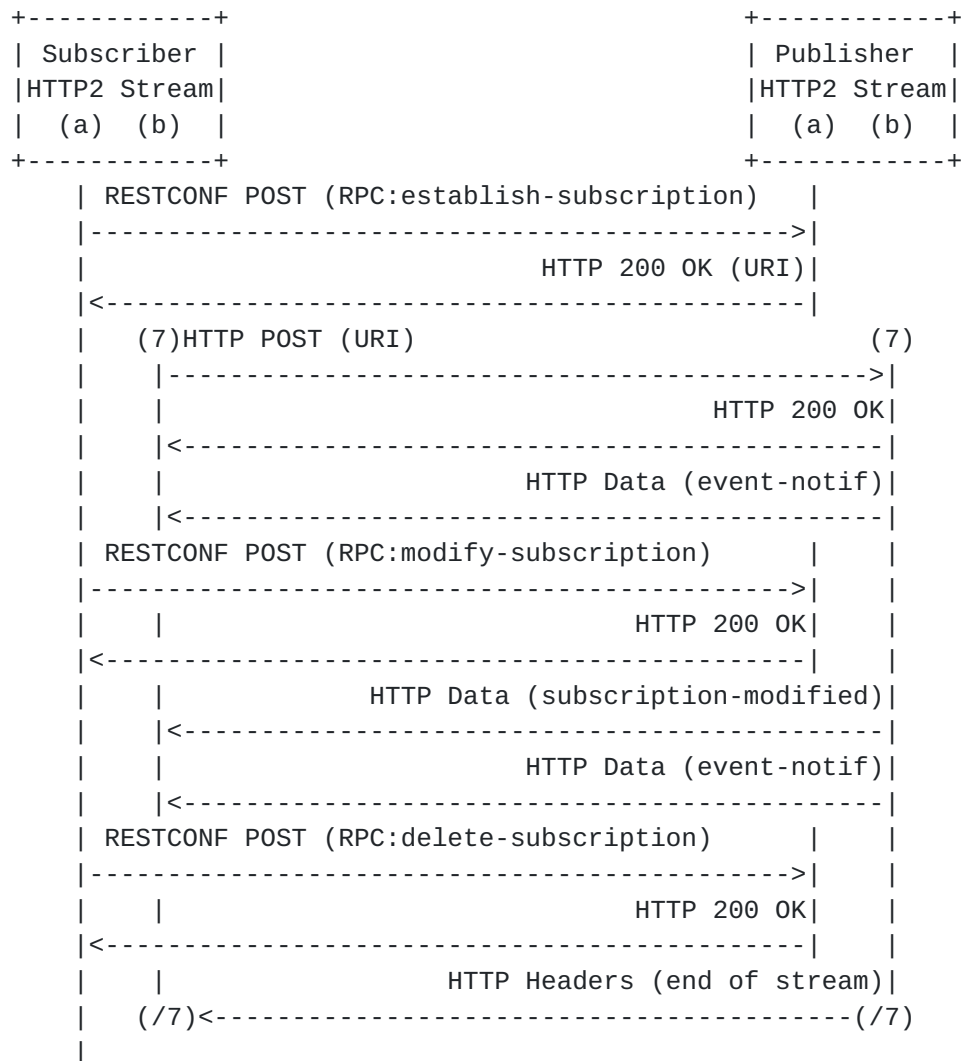


Figure 1: Dynamic with HTTP2

### 3.1.2. Call flow for HTTP1.1

Requests to [I-D.[ietf-netconf-yang-push](#)] RPCs are sent on the TCP connection indicated by (a). Notification messages are pushed on a separate connection (b). This connection (b) will be used for all notification messages across all subscriptions.



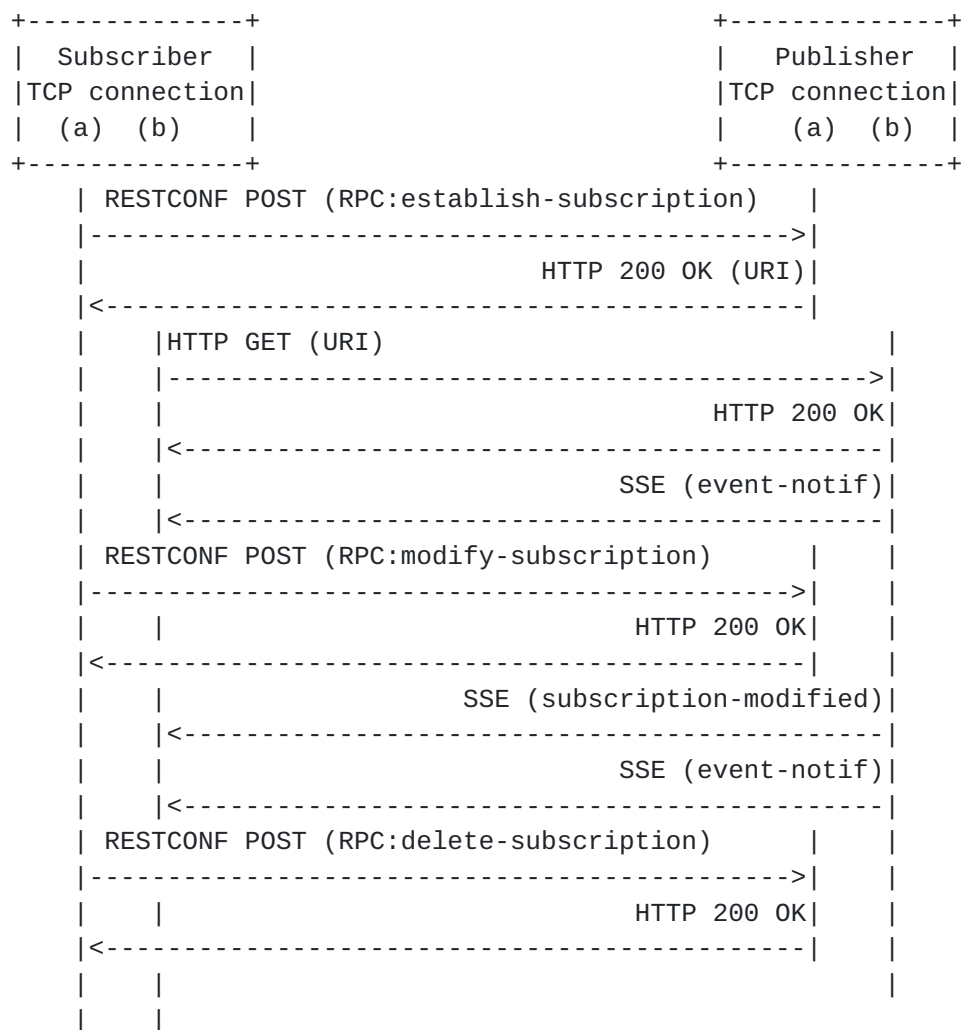


Figure 2: Dynamic with HTTP1.1

### 3.1.3. Configured Subscription over HTTP2

With a configured subscription, all information needed to establish a secure relationship with that receiver is available on the publisher. With this information, the publisher will establish a secure transport connection with the receiver and then begin pushing notification messages to the receiver. Since RESTCONF might not exist on the receiver, it is not desirable to require that subscribed content be pushed with any dependency on RESTCONF. Therefore in place of RESTCONF, a TLS secured HTTP2 Client connection must be established with an HTTP2 Server located on the receiver. Notification messages will then be sent as part of an extended HTTP POST to the receiver.

POST messages will be addressed to HTTP augmentation code on the receiver capable of accepting and responding to state change





notifications and subscribed content notification messages. The first POST message must be a subscription-started notification. Notifications which include any subscribed content must not be sent until the receipt of an HTTP 200 OK for this initial notification. The 200 OK will indicate that the receiver is ready for the delivery of subscribed content. At this point a subscription must be allocated its own HTTP2 stream. Figure 4 depicts this message flow.

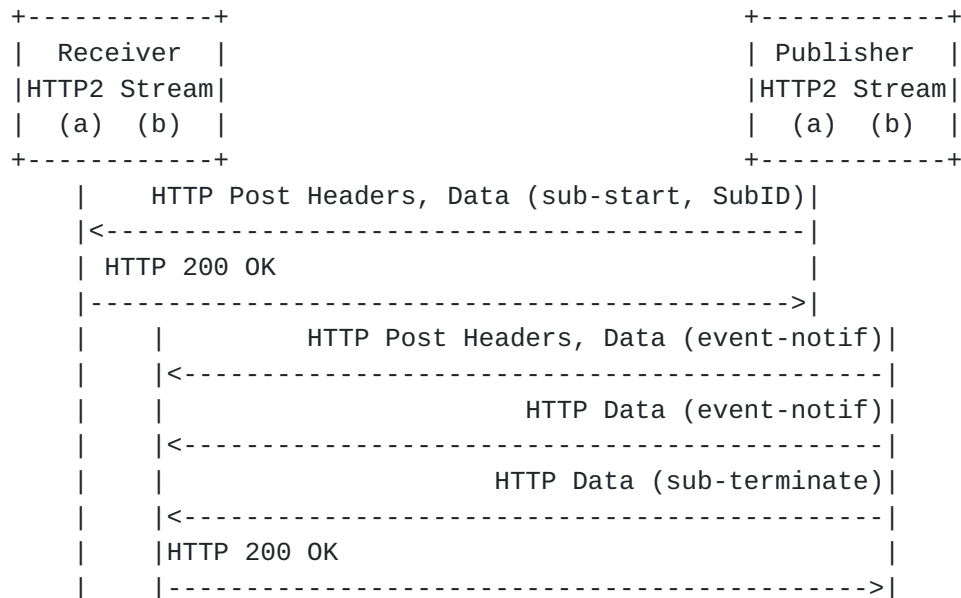


Figure 3: Configured over HTTP2

As the HTTP2 transport is available to the receiver, the publisher should:

- o take any subscription-priority and copy it into the HTTP2 stream priority, and
- o take a subscription-dependency if it has been provided and map the HTTP2 stream for the parent subscription into the HTTP2 stream dependency.

#### 4. Mandatory JSON and datastore support

A publisher MUST support JSON encoding of RPCs and Notifications.

A publisher supporting [[I-D.ietf-netconf-yang-push](#)] MUST support the "operational" datastore as defined by [[I.D.draft-ietf-netmod-revised-datastores](#)].



## **5. Notification Messages**

Notification messages transported over NETCONF will be identical in format and content to those encoded using one-way operations defined within [\[RFC5277\], section 4](#).

## **6. Security Considerations**

One or more publishers of configured subscriptions could be used to overwhelm a receiver which doesn't even support subscriptions. There are two protections needing support on a publisher. First, notification messages for configured subscriptions MUST only be transmittable over encrypted transports. Clients which do not want pushed content need only terminate or refuse any transport sessions from the publisher. Second, the HTTP transport augmentation on the receiver must send an HTTP 200 OK to a subscription started notification before the publisher starts streaming any subscribed content.

One or more publishers could overwhelm a receiver which is unable to control or handle the volume of Event Notifications received. In deployments where this might be a concern, HTTP2 transport such as HTTP2) should be selected.

The NETCONF Authorization Control Model [\[RFC6536\]](#) SHOULD be used to control and restrict authorization of subscription configuration.

## **7. Acknowledgments**

We wish to acknowledge the helpful contributions, comments, and suggestions that were received from: Susan Hares, Tim Jenkins, Balazs Lengyel, Kent Watsen, Michael Scharf, and Guangying Zheng.

## **8. References**

### **8.1. Normative References**

[I-D.[draft-ietf-netconf-subscribed-notifications](#)]

Voit, E., Clemm, A., Gonzalez Prieto, A., Tripathy, A., and E. Nilsen-Nygaard, "Custom Subscription to Event Streams", [draft-ietf-netconf-subscribed-notifications-06](#) (work in progress), January 2018.

[I.D.[draft-ietf-netmod-revised-datastores](#)]

Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture", [draft-ietf-netmod-revised-datastores-04](#) (work in progress), August 2017.



- [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>>.
- [RFC5277] Chisholm, S. and H. Trevino, "NETCONF Event Notifications", [RFC 5277](#), DOI 10.17487/RFC5277, July 2008, <<https://www.rfc-editor.org/info/rfc5277>>.
- [RFC6520] Seggelmann, R., Tuexen, M., and M. Williams, "Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS) Heartbeat Extension", [RFC 6520](#), DOI 10.17487/RFC6520, February 2012, <<https://www.rfc-editor.org/info/rfc6520>>.
- [RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", [RFC 6536](#), DOI 10.17487/RFC6536, March 2012, <<https://www.rfc-editor.org/info/rfc6536>>.
- [RFC7540] Belshe, M., Peon, R., and M. Thomson, Ed., "Hypertext Transfer Protocol Version 2 (HTTP/2)", [RFC 7540](#), DOI 10.17487/RFC7540, May 2015, <<https://www.rfc-editor.org/info/rfc7540>>.
- [RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", [RFC 8040](#), DOI 10.17487/RFC8040, January 2017, <<https://www.rfc-editor.org/info/rfc8040>>.

## 8.2. Informative References

- [GRPC] "RPC framework that runs over HTTP2", August 2017, <<https://grpc.io/>>.
- [I-D.ietf-netconf-yang-push] Clemm, A., Voit, E., Gonzalez Prieto, A., Prasad Tripathy, A., Nilsen-Nygaard, E., Bierman, A., and B. Lengyel, "Subscribing to YANG datastore push updates", March 2017, <<https://datatracker.ietf.org/doc/draft-ietf-netconf-yang-push/>>.
- [RFC7923] Voit, E., Clemm, A., and A. Gonzalez Prieto, "Requirements for Subscription to YANG Datastores", [RFC 7923](#), DOI 10.17487/RFC7923, June 2016, <<https://www.rfc-editor.org/info/rfc7923>>.



- [RFC7951] Lhotka, L., "JSON Encoding of Data Modeled with YANG", [RFC 7951](#), DOI 10.17487/RFC7951, August 2016, <<https://www.rfc-editor.org/info/rfc7951>>.
- [RFC8071] Watsen, K., "NETCONF Call Home and RESTCONF Call Home", [RFC 8071](#), DOI 10.17487/RFC8071, February 2017, <<https://www.rfc-editor.org/info/rfc8071>>.
- [W3C-20150203]  
"Server-Sent Events, World Wide Web Consortium CR CR-eventsources-20121211", February 2015, <<https://www.w3.org/TR/2015/REC-eventsources-20150203/>>.

## **[Appendix A](#). End-to-End Deployment Guidance**

Several technologies are expected to be seen within a deployment to achieve security and ease-of-use requirements. These are not necessary for an implementation of this specification, but will be useful to consider when considering the operational context.

### **[A.1](#). Call Home**

Implementations should include the ability to transparently incorporate 'call home' [[RFC8071](#)] so that secure TLS connections can originate from the desired device.

### **[A.2](#). TLS Heartbeat**

HTTP sessions might not quickly allow a subscriber to recognize when the communication path has been lost from the publisher. To recognize this, it is possible for a receiver to establish a TLS heartbeat [[RFC6520](#)]. In the case where a TLS heartbeat is included, it should be sent just from receiver to publisher. Loss of the heartbeat should result in any subscription related TCP sessions between those endpoints being torn down. The subscription can then attempt to re-establish.

## **[Appendix B](#). RESTCONF over GRPC**

An initial goal for this document was to support [[GRPC](#)] transport seamlessly without any mapping or extra layering. However there is an incompatibility of RESTCONF and GRPC. RESTCONF uses HTTP GET, and GRPC uses HTTP2's POST rather than GET. As GET is used across RESTCONF for things like capabilities exchange, a seamless mapping depends on specification changes outside the scope of this document. If/when GRPC supports GET, or RESTCONF is updated to support POST, this should be revisited. It is hoped that the resulting fix will be transparent to this document.





## **[Appendix C](#). Encoded Subscription and Notification Message Examples**

(Note: examples in this section need significant updates)

### **[C.1](#). RESTCONF Subscription and Events over HTTP1.1**

Subscribers can dynamically learn whether a RESTCONF server supports various types of Event or Yang datastore subscription capabilities. This is done by issuing an HTTP request OPTIONS, HEAD, or GET on the stream. Some examples building upon the Call flow for HTTP1.1 from [Section 3.2.2](#) are:

```
GET /restconf/data/ietf-restconf-monitoring:restconf-state/
    streams/stream=yang-push HTTP/1.1
Host: example.com
Accept: application/yang.data+xml
```

If the server supports it, it may respond

```
HTTP/1.1 200 OK
Content-Type: application/yang.api+xml
<stream xmlns="urn:ietf:params:xml:ns:yang:ietf-restconf-monitoring">
  <name>yang-push</name>
  <description>Yang push stream</description>
  <access>
    <encoding>xml</encoding>
    <location>https://example.com/streams/yang-push-xml
    </location>
  </access>
  <access>
    <encoding>json</encoding>
    <location>https://example.com/streams/yang-push-json
    </location>
  </access>
</stream>
```

If the server does not support any form of subscription, it may respond

```
HTTP/1.1 404 Not Found
Date: Mon, 25 Apr 2012 11:10:30 GMT
Server: example-server
```

Subscribers can determine the URL to receive updates by sending an HTTP GET as a request for the "location" leaf with the stream list entry. The stream to use for may be selected from the Event Stream list provided in the capabilities exchange. Note that different



encodings are supporting using different Event Stream locations. For example, the subscriber might send the following request:

```
GET /restconf/data/ietf-restconf-monitoring:restconf-state/  
    streams/stream=yang-push/access=xml/location HTTP/1.1  
Host: example.com  
Accept: application/yang.data+xml
```

The publisher might send the following response:

```
HTTP/1.1 200 OK  
Content-Type: application/yang.api+xml  
  <location  
    xmlns="urn:ietf:params:xml:ns:yang:ietf-restconf-monitoring">  
    https://example.com/streams/yang-push-xml  
  </location>
```

To subscribe and start receiving updates, the subscriber can then send an HTTP GET request for the URL returned by the publisher in the request above. The accept header must be "text/event-stream". The publisher uses the Server Sent Events [[W3C-20150203](#)] transport strategy to push filtered events from the event stream.

The publisher MUST support individual parameters within the POST request body for all the parameters of a subscription. The only exception is the encoding, which is embedded in the URI. An example of this is:

```
// subtree filter = /foo  
// periodic updates, every 5 seconds  
POST /restconf/operations/ietf-subscribed-notifications:  
    establish-subscription HTTP/1.1  
Host: example.com  
Content-Type: application/yang-data+json  
  
  {  
    "ietf-subscribed-notifications:input" : {  
      "stream": "push-data"  
      "period" : 5,  
      "xpath-filter" : "/ex:foo[starts-with('bar'.'some')]"  
    }  
  }
```

Should the publisher not support the requested subscription, it may reply:



HTTP/1.1 501 Not Implemented

Date: Mon, 23 Apr 2012 17:11:00 GMT

Server: example-server

Content-Type: application/yang.errors+xml

```
<errors xmlns="urn:ietf:params:xml:ns:yang:ietf-restconf">
  <error>
    <error-type>application</error-type>
    <error-tag>operation-not-supported</error-tag>
    <error-severity>error</error-severity>
    <error-message>Xpath filters not supported</error-message>
    <error-info>
      <supported-subscription xmlns="urn:ietf:params:xml:ns:
        netconf:datastore-push:1.0">
        <subtree-filter/>
      </supported-subscription>
    </error-info>
  </error>
</errors>
```

with an equivalent JSON encoding representation of:

HTTP/1.1 501 Not Implemented

Date: Mon, 23 Apr 2012 17:11:00 GMT

Server: example-server

Content-Type: application/yang.errors+json

```
{
  "ietf-restconf:errors": {
    "error": {
      "error-type": "protocol",
      "error-tag": "operation-not-supported",
      "error-message": "Xpath filters not supported."
      "error-info": {
        "datastore-push:supported-subscription": {
          "subtree-filter": [null]
        }
      }
    }
  }
}
```

The following is an example of a pushed content for the subscription above. It contains a subtree with root foo that contains a leaf called bar:



XML encoding representation:

```
<?xml version="1.0" encoding="UTF-8"?>
<notification xmlns="urn:ietf:params:xml:ns:yang:ietf-restconf">
  <subscription-id xmlns="urn:ietf:params:xml:ns:restconf:
    datastore-push:1.0">
    my-sub
  </subscription-id>
  <eventTime>2015-03-09T19:14:56.233Z</eventTime>
  <datastore-contents xmlns="urn:ietf:params:xml:ns:restconf:
    datastore-push:1.0">
    <foo xmlns="http://example.com/yang-push/1.0">
      <bar>some_string</bar>
    </foo>
  </datastore-contents>
</notification>
```

Or with the equivalent YANG over JSON encoding representation as defined in [\[RFC7951\]](#):

```
{
  "ietf-restconf:notification": {
    "datastore-push:subscription-id": "my-sub",
    "eventTime": "2015-03-09T19:14:56.233Z",
    "datastore-push:datastore-contents": {
      "example-mod:foo": { "bar": "some_string" }
    }
  }
}
```

To modify a subscription, the subscriber issues another POST request on the provided URI using the same subscription-id as in the original request. For example, to modify the update period to 10 seconds, the subscriber may send:

POST /restconf/operations/ietf-subscribed-notifications:

modify-subscription HTTP/1.1

Host: example.com

Content-Type: application/yang-data+json

```
{
  "ietf-subscribed-notifications:input" : {
    "subscription-id": 100,
    "period" : 10
  }
}
```





To delete a subscription, the subscriber issues a DELETE request on the provided URI using the same subscription-id as in the original request

## **C.2. Event Notification over HTTP2**

The basic encoding will look as below. It will consists of a JSON representation wrapped in an HTTP2 header.

HyperText Transfer Protocol 2

Stream: HEADERS, Stream ID: 5

Header: :method: POST

Stream: HEADERS, Stream ID: 5

```
{
  "ietf-yangpush:notification": {
    "datastore-push:subscription-id": "my-sub",
    "eventTime": "2015-03-09T19:14:56.233Z",
    "datastore-push:datastore-contents": {
      "foo": { "bar": "some_string" }
    }
  }
}
```

## **Appendix D. Changes between revisions**

(To be removed by RFC editor prior to publication)

v03 - v04

- o Draft not fully synched to new version of subscribed-notifications yet.
- o References updated

v02 - v03

- o Event notification reframed to notification message.
- o Tweaks to wording/capitalization/format.

v01 - v02

- o Removed sections now redundant with [I-D.[draft-ietf-netconf-subscribed-notifications](#)] and [I-D.[ietf-netconf-yang-push](#)] such as: mechanisms for subscription maintenance, terminology definitions, stream discovery.



- o 3rd party subscriptions are out-of-scope.
- o SSE only used with RESTCONF and HTTP1.1 dynamic subscriptions
- o Timeframes for event tagging are self-defined.
- o Clean-up of wording, references to terminology, section numbers.

v00 - v01

- o Removed the ability for more than one subscription to go to a single HTTP2 stream.
- o Updated call flows. Extensively.
- o SSE only used with RESTCONF and HTTP1.1 dynamic subscriptions
- o HTTP is not used to determine that a receiver has gone silent and is not Receiving Event Notifications
- o Many clean-ups of wording and terminology

#### Authors' Addresses

Eric Voit  
Cisco Systems

Email: [evoit@cisco.com](mailto:evoit@cisco.com)

Ambika Prasad Tripathy  
Cisco Systems

Email: [ambtripa@cisco.com](mailto:ambtripa@cisco.com)

Einar Nilsen-Nygaard  
Cisco Systems

Email: [einarnn@cisco.com](mailto:einarnn@cisco.com)

Alexander Clemm  
Huawei

Email: [ludwig@clemm.org](mailto:ludwig@clemm.org)



Alberto Gonzalez Prieto  
VMware

Email: [agonzalezpri@vmware.com](mailto:agonzalezpri@vmware.com)

Andy Bierman  
YumaWorks

Email: [andy@yumaworks.com](mailto:andy@yumaworks.com)