An HTTPS-based Transport for Configured Subscriptions
draft-ietf-netconf-https-notif-07

Abstract

   This document defines both a protocol for sending notifications over
   HTTPS as well as extensions to the data model for configured
   subscriptions defined in RFC 8639. It also presents an example
   module for configuration without using the data model defined in RFC
   8639.

   This document requires that the publisher is a "server" (e.g., a
   NETCONF or RESTCONF server), but does not assume that the receiver is
   a server.

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 6, 2021.

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

This document defines a protocol for sending notifications over
HTTPS. Using HTTPS maximizes transport-level interoperability, while
allowing for a variety of encoding options. This document defines
support for JSON and XML; future efforts may define support for other
encodings (e.g., binary).

This document also defines two YANG 1.1 [RFC7950] modules that extend
the data model defined in Subscription to YANG Notifications
[RFC8639], enabling the configuration of HTTPS-based receivers.

An example module illustrating the configuration of a publisher not
using the data model defined in RFC 8639 is also provided.

Configured subscriptions enable a server, acting as a publisher of
notifications, to proactively push notifications to external
receivers without the receivers needing to first connect to the
server, as is the case with dynamic subscriptions.

1.1. Applicability Statement

While the YANG modules have been defined as an augmentation of
Subscription to YANG Notifications [RFC8639], the notification method
defined in this document MAY be used outside of Subscription to YANG
Notifications [RFC8639] by using some of the definitions from this
module along with the grouping defined in Groupings for HTTP Clients
and Servers [I-D.ietf-netconf-http-client-server]. For an example on
how that can be done, see Section 8.2.

1.2. Note to RFC Editor

This document uses several placeholder values throughout the
document. Please replace them as follows and remove this section
before publication.

RFC XXXX, where XXXX is the number assigned to this document at the
time of publication.

RFC YYYY, where YYYY is the number assigned to
[I-D.ietf-netconf-http-client-server].

2021-02-02 with the actual date of the publication of this document.

1.3. Abbreviations

```
+---------+---------------------------------------+
| Acronym | Expansion                             |
+---------+---------------------------------------+
| HTTP    | Hyper Text Transport Protocol         |
|         |                                       |
| HTTPS   | Hyper Text Transport Protocol Secure  |
|         |                                       |
| TCP     | Transmission Control Protocol         |
|         |                                       |
| TLS     | Transport Layer Security              |
+---------+---------------------------------------+
```

1.4. Terminology

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.

1.4.1. Subscribed Notifications

The following terms are defined in Subscription to YANG Notifications
[RFC8639].

o Subscribed Notifications

2. Overview of Publisher to Receiver Interaction

The protocol consists of two HTTP-based target resources presented by
the receiver:

o A target resource enabling the publisher to discover what optional
   capabilities a receiver supports. Publishers SHOULD query this
   target when before sending any notifications or if ever an error
   occurs.

o A target resource enabling to publish to send one or more
   notification to a receiver. This document defines support for
   sending only one notification per message; a future effort MAY
   extend the protocol to send multiple notifications per message.

The protocol is illustrated in the diagram below:

```
             -------------                      --------------
             | Publisher |                      | Receiver   |
             -------------                      --------------

        Send HTTPS GET message     ------>
        to discover receiver's
        capabilities

                                   <------     Send 200 (OK) containing
                                               capabilities supported
                                               by the receiver


   +-- For Each Notification (MAY be pipelined) ---------------------+
   |                                                                 |
   |    Send HTTPS POST message    ------>                           |
   |    with YANG defined                                            |
   |    notification                                                 |
   |                                                                 |
   |                                   <------     Send 204 (No Content)   |
   +-----------------------------------------------------------------+
```
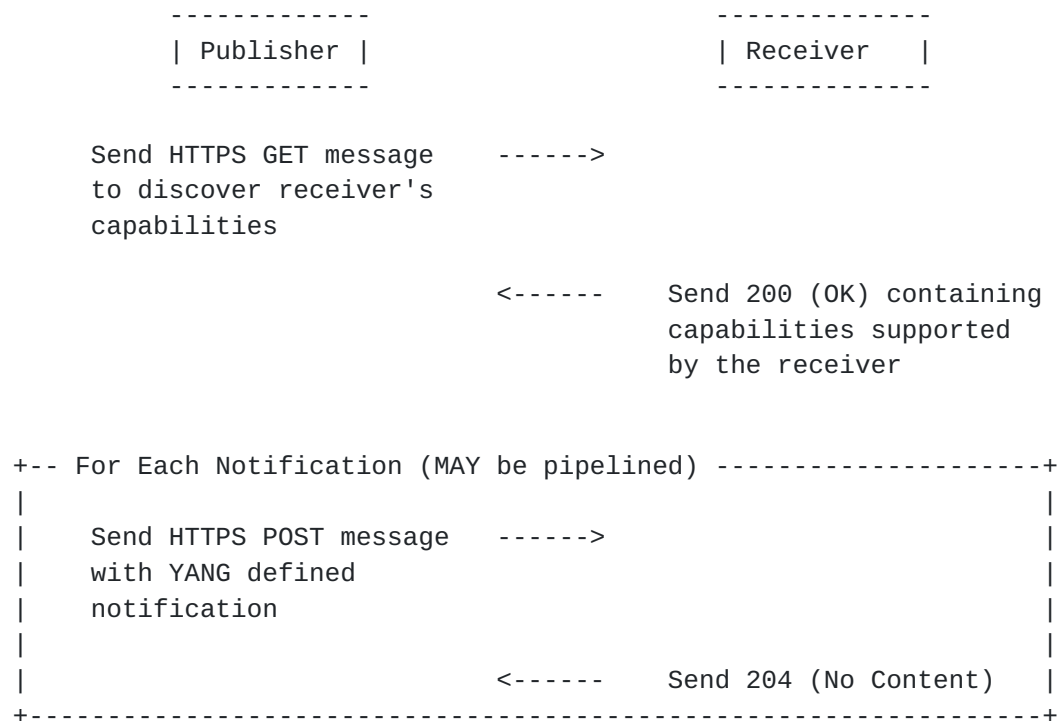
   Note that, for RFC 8639 configured subscriptions, the very first
   notification must be the "subscription-started" notification.

   The POST messages MAY be "pipelined" (not illustrated in the diagram
   above), whereby multiple notifications are sent without waiting for
   the HTTP response for a previous request.

3. Discovering a Receiver's Capabilities

3.1. Applicability

   For publishers using Subscription to YANG Notifications [RFC8639],
   dynamic discovery of a receiver's supported encoding is necessary
   only when the "/subscriptions/subscription/encoding" leaf is not
   configured, per the "encoding" leaf's description statement in the
   "ietf-subscribed-notification" module. FIXME: do they need to
   discover *any* capabilities?

3.2. Request

   To learn the capabilities of a receiver, a publisher can issue an
   HTTPS GET request to the "capabilities" resource under a known path
   on the receiver with "Accept" header set using the "application/xml"
   and/or "application/json" media-types, with latter as the mandatory
   to implement, and the default in case the type is not specified.

3.3. Response

The receiver responds with a "200 (OK)" message, having the "Content-
Type" header set to either "application/xml" or "application/json"
(which ever was selected), and containing in the response body a list
of the receiver's capabilities encoded in the selected format.

Even though a YANG module is not defined for this interaction, the
response body MUST conform to the following YANG-modeled format:

```
container receiver-capabilities {
  description
    "A container for a list of capabilities supported by
     the receiver.";
  leaf-list receiver-capability {
    type string {
      pattern "urn:ietf:capability:https-notif-receiver:*";
    }
    description
      "A capability supported by the receiver.  A full list of
       capabilities is defined in the 'Capabilities for HTTPS
       Notification Receivers' registry (see RFC XXXX).";
  }
}
```

3.4. Example

The publisher can send the following request to learn the receiver
capabilities. In this example, the "Accept" states that the receiver
wants to receive notifications in XML but, if not supported, to use
JSON encoding.

```
GET /some/path/capabilities HTTP/1.1
Host: example.com
Accept: application/xml, application/json
```

If the receiver is able to reply using "application/xml", and
assuming it is able to receive JSON and XML encoded notifications,
the response might look like this:

```
HTTP/1.1 200 OK
Date: Wed, 26 Feb 2020 20:33:30 GMT
Server: example-server
Cache-Control: no-cache
Content-Type: application/xml
Content-Length: nnn

<receiver-capabilities>
  <receiver-capability>\
    urn:ietf:capability:https-notif-receiver:encoding:json\
  </receiver-capability>
  <receiver-capability>\
    urn:ietf:capability:https-notif-receiver:encoding:xml\
  </receiver-capability>
</receiver-capabilities>
```

If the receiver is unable to reply using "application/xml", the
response might look like this:

```
HTTP/1.1 200 OK
Date: Wed, 26 Feb 2020 20:33:30 GMT
Server: example-server
Cache-Control: no-cache
Content-Type: application/json
Content-Length: nnn

{
   receiver-capabilities {
     "receiver-capability": [
       "urn:ietf:capability:https-notif-receiver:encoding:json",
       "urn:ietf:capability:https-notif-receiver:encoding:xml"
     ]
   }
}
```

4. Sending Event Notifications

4.1. Request

The publisher sends an HTTPS POST request to the "relay-
notifications" resource under a known path on the receiver with the
"Content-Type" header set to either "application/json" or
"application/xml" and a request body containing the notification
encoded using the specified format.

XML-encoded notifications are encoded using the format defined by
NETCONF Event Notifications [RFC5277] for XML.

JSON-encoded notifications are encoded the same as specified in
Section 6.4 in RESTCONF [RFC8040] with the following deviations:

o The notifications do not contain the "data:" prefix used by SSE.

o Instead of saying that, for JSON-encoding purposes, the module
   name for the "notification" element is "ietf-restconf, the module
   name will instead by "ietf-https-notif".

4.2. Response

The response should be "204 (No Content)".

4.3. Example

An XML-encoded notification might be sent as follows:

```
POST /some/path/relay-notification HTTP/1.1
Host: example.com
Content-Type: application/xml

<notification xmlns="urn:ietf:params:xml:ns:netconf:notification:1.0">
  <eventTime>2019-03-22T12:35:00Z</eventTime>
  <event xmlns="https://example.com/example-mod">
    <event-class>fault</fault>
    <reporting-entity>
      <card>Ethernet0</card>
    </reporting-entity>
    <severity>major</severity>
  </event>
</notification>
```

A JSON-encoded notification might be sent as follows:

```
POST /some/path/relay-notification HTTP/1.1
Host: example.com
Content-Type: application/json

{
  "ietf-https-notif:notification": {
    "eventTime": "2013-12-21T00:01:00Z",
    "example-mod:event" : {
      "event-class" : "fault",
      "reporting-entity" : { "card" : "Ethernet0" },
      "severity" : "major"
    }
  }
}
```

And, in either case, the response might be as follows:

```
HTTP/1.1 204 No Content
Date: Wed, 26 Feb 2020 20:33:30 GMT
Server: example-server
```

5. The "ietf-subscribed-notif-receivers" Module

5.1. Data Model Overview

This YANG module augments the "ietf-subscribed-notifications" module
to define a choice of transport types that other modules such as the
"ietf-https-notif-transport" module can use to define a transport
specific receiver.

[note: '\' line wrapping for formatting only]

```
module: ietf-subscribed-notif-receivers

  augment /sn:subscriptions:
    +--rw receiver-instances
       +--rw receiver-instance* [name]
          +--rw name    string
          +--rw (transport-type)
  augment /sn:subscriptions/sn:subscription/sn:receivers/sn:receiver:\

    +--rw receiver-instance-ref?   leafref
```

5.2. YANG Module

The YANG module imports Subscription to YANG Notifications [RFC8639].

```
<CODE BEGINS> file "ietf-subscribed-notif-receivers@2021-02-02.yang"
[note: '\' line wrapping for formatting only]

module ietf-subscribed-notif-receivers {
  yang-version 1.1;
  namespace
    "urn:ietf:params:xml:ns:yang:ietf-subscribed-notif-receivers";
  prefix "snr";

  import ietf-subscribed-notifications {
    prefix sn;
    reference
      "RFC 8639: Subscription to YANG Notifications";
  }

  organization
```

      "IETF NETCONF Working Group";

   contact
     "WG Web: <http://tools.ietf.org/wg/netconf>
      WG List: <netconf@ietf.org>

      Authors: Mahesh Jethanandani (mjethanandani at gmail dot com)
               Kent Watsen (kent plus ietf at watsen dot net)";

   description
     "This YANG module is implemented by Publishers implementing
      the 'ietf-subscribed-notifications' module defined in RFC 8639.

      While this module is defined in RFC XXXX, which primarily
      defines an HTTPS-based transport for notifications, this module
      is not HTTP-specific. It is a generic extension that can be
      used by any 'notif' transport.

      This module defines two 'augment' statements. One statement
      augments a 'container' statement called 'receiver-instances'
      into the top-level 'subscriptions' container. The other
      statement, called 'receiver-instance-ref', augemnts a 'leaf'
      statement into each 'receiver' that references one of the
      afore mentioned receiver instances. This indirection enables
      multiple configured subscriptions to send notifications to
      the same receiver instance.

      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 (RFC 2119) (RFC 8174) when, and only when,
      they appear in all capitals, as shown here.";

   revision "2021-02-02" {
     description
       "Initial Version.";

```
      reference
        "RFC XXXX, YANG Data Module for HTTPS Notifications.";
    }

    augment "/sn:subscriptions" {
      container receiver-instances {
        description
          "A container for all instances of receivers.";

        list receiver-instance {
          key "name";

          leaf name {
            type string;
            description
              "An arbitrary but unique name for this receiver
               instance.";
          }

          choice transport-type {
            mandatory true;
            description
              "Choice of different types of transports used to
               send notifications.  The 'case' statements must
               be augmented in by other modules.";
          }
          description
            "A list of all receiver instances.";
        }
      }
      description
        "Augment the subscriptions container to define the
         transport type.";
    }

    augment
      "/sn:subscriptions/sn:subscription/sn:receivers/sn:receiver" {
      leaf receiver-instance-ref {
        type leafref {
          path "/sn:subscriptions/snr:receiver-instances/" +
               "snr:receiver-instance/snr:name";
        }
        description
          "Reference to a receiver instance.";
      }
      description
        "Augment the subscriptions container to define an optional
         reference to a receiver instance.";
```

```
    }

  }
  <CODE ENDS>
```

6. The "ietf-https-notif-transport" Module

6.1. Data Model Overview

This YANG module is a definition of a set of receivers that are
interested in the notifications published by the publisher. The
module contains the TCP, TLS and HTTPS parameters that are needed to
communicate with the receiver. The module augments the "ietf-
subscribed-notif-receivers" module to define a transport specific
receiver.

As mentioned earlier, it uses a POST method to deliver the
notification. The "http-receiver/tls/http-client-parameters/path"
leaf defines the path for the resource on the receiver, as defined by
"path-absolute" in URI Generic Syntax [RFC3986]. The user-id used by
Network Configuration Access Control Model [RFC8341], is that of the
receiver and is derived from the certificate presented by the
receiver as part of "receiver-identity".

An abridged tree diagram representing the module is shown below.

[note: '\' line wrapping for formatting only]

```
module: ietf-https-notif-transport

  augment /sn:subscriptions/snr:receiver-instances
          /snr:receiver-instance/snr:transport-type:
    +--:(https)
       +--rw https-receiver
          +--rw (transport)
          |  +--:(tcp) {tcp-supported,not httpc:tcp-supported}?
          |  |  +--rw tcp
          |  |     +--rw tcp-client-parameters
          |  |        |  +--rw remote-address    inet:host
          |  |        |  +--rw remote-port?      inet:port-number
          |  |        |  +--rw local-address?    inet:ip-address
          |  |        |  |      {local-binding-supported}?
          |  |        |  +--rw local-port?       inet:port-number
          |  |        |  |      {local-binding-supported}?
          |  |        |  +--rw proxy-server! {proxy-connect}?
          |  |        |  |    ...
          |  |        |  +--rw keepalives! {keepalives-supported}?
          |  |        |        ...
```

```
               |  |       +--rw http-client-parameters
               |  |          +--rw client-identity!
               |  |          |      ...
               |  |          +--rw proxy-connect! {proxy-connect}?
               |  |                 ...
               |  +--:(tls) {tls-supported}?
               |     +--rw tls
               |        +--rw tcp-client-parameters
               |        |  +--rw remote-address    inet:host
               |        |  +--rw remote-port?      inet:port-number
               |        |  +--rw local-address?    inet:ip-address
               |        |  |       {local-binding-supported}?
               |        |  +--rw local-port?       inet:port-number
               |        |  |       {local-binding-supported}?
               |        |  +--rw proxy-server! {proxy-connect}?
               |        |  |  |     ...
               |        |  +--rw keepalives! {keepalives-supported}?
               |        |  |        ...
               |        +--rw tls-client-parameters
               |        |  +--rw client-identity!
               |        |  |  |     ...
               |        |  +--rw server-authentication
               |        |  |  |     ...
               |        |  +--rw hello-params
               |        |  |       {tls-client-hello-params-config}?
               |        |  |  |     ...
               |        |  +--rw keepalives {tls-client-keepalives}?
               |        |  |        ...
               |        +--rw http-client-parameters
               |           +--rw client-identity!
               |           |      ...
               |           +--rw proxy-connect! {proxy-connect}?
               |           |      ...
               |           +--rw path              string
               +--rw receiver-identity {receiver-identity}?
                  +--rw cert-maps
                     +--rw cert-to-name* [id]
                        +--rw id            uint32
                        +--rw fingerprint   x509c2n:tls-fingerprint
                        +--rw map-type      identityref
                        +--rw name          string
```

6.2. YANG module

The YANG module imports A YANG Data Model for SNMP Configuration
[RFC7407], Subscription to YANG Notifications [RFC8639], and YANG
Groupings for HTTP Clients and HTTP Servers
[I-D.ietf-netconf-http-client-server].

The YANG module is shown below.

```
<CODE BEGINS> file "ietf-https-notif-transport@2021-02-02.yang"
[note: '\' line wrapping for formatting only]

module ietf-https-notif-transport {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-https-notif-transport";\

  prefix "hnt";

  import ietf-x509-cert-to-name {
    prefix x509c2n;
    reference
      "RFC 7407: YANG Data Model for SNMP Configuration.";
  }

  import ietf-subscribed-notifications {
    prefix sn;
    reference
      "RFC 8639: Subscription to YANG Notifications";
  }

  import ietf-subscribed-notif-receivers {
    prefix snr;
    reference
      "RFC XXXX: An HTTPS-based Transport for
                 Configured Subscriptions";
  }

  import ietf-http-client {
    prefix httpc;
    reference
      "RFC YYYY: YANG Groupings for HTTP Clients and HTTP Servers";
  }

  organization
    "IETF NETCONF Working Group";

  contact
    "WG Web:   <http://tools.ietf.org/wg/netconf>
     WG List:  <netconf@ietf.org>

     Authors: Mahesh Jethanandani (mjethanandani at gmail dot com)
              Kent Watsen (kent plus ietf at watsen dot net)";

  description
    "This YANG module is implemented by Publishers that implement
```

the 'ietf-subscribed-notifications' module defined in RFC 8639.

This module augments a 'case' statement called 'https' into
the 'choice' statement called 'transport-type' defined
by the 'ietf-https-notif-transport' module defined in RFC XXXX. \

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 (RFC 2119) (RFC 8174) when, and only when,
they appear in all capitals, as shown here.";

```
  revision "2021-02-02" {
    description
      "Initial Version.";
    reference
      "RFC XXXX, YANG Data Module for HTTPS Notifications.";
  }

  feature receiver-identity {
    description
      "Indicates if the server supports filtering notifications
       based on the receiver's identity derived from its TLS
       certificate.";
  }

  identity https {
    base sn:transport;
    description
      "HTTPS transport for notifications.";
  }

  grouping https-receiver-grouping {
    description
      "A grouping that may be used by other modules wishing to
```

```
       configure HTTPS-based notifications without using RFC 8639.";
     uses httpc:http-client-stack-grouping {
       refine "transport/tcp" {
         // create the logical impossibility of enabling the
         // "tcp" transport (i.e., "HTTP" without the 'S').
         if-feature "not httpc:tcp-supported";
       }
       augment "transport/tls/tls/http-client-parameters" {
         leaf path {
           type string;
           mandatory true;
           description
             "URI prefix to the target resources.  Under this
              path the receiver must support both the 'capabilities'
              and 'relay-notification' resource targets, as described
              in RFC XXXX.";
         }
         description
           "Augmentation to add a receiver-specific path for the
            'capabilities' and 'relay-notification' resources.";
       }
     }
     container receiver-identity {
       if-feature receiver-identity;
       description
         "Maps the receiver's TLS certificate to a local identity
          enabling access control to be applied to filter out
          notifications that the receiver may not be authorized
          to view.";
       container cert-maps {
         uses x509c2n:cert-to-name;
         description
           "The cert-maps container is used by a TLS-based HTTP
            server to map the HTTPS client's presented X.509
            certificate to a 'local' username. If no matching and
            valid cert-to-name list entry is found, the publisher
            MUST close the connection, and MUST NOT not send any
            notifications over it.";
         reference
           "RFC 7407: A YANG Data Model for SNMP Configuration.";
       }
     }
   }

   augment "/sn:subscriptions/snr:receiver-instances/" +
           "snr:receiver-instance/snr:transport-type" {
     case https {
       container https-receiver {
```

```
            description
              "The HTTPS receiver to send notifications to.";
            uses https-receiver-grouping;
          }
        }
        description
          "Augment the transport-type choice to include the 'https'
           transport.";
      }
    }
  }
  <CODE ENDS>
```

7. Security Considerations

   The YANG module specified in this document defines a schema for data
   that is designed to be accessed via network management protocols such
   as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer
   is the secure transport layer, and the mandatory-to-implement secure
   transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer
   is HTTPS, and the mandatory-to-implement secure transport is TLS
   [RFC8446]. The NETCONF Access Control Model (NACM) [RFC8341]
   provides the means to restrict access for particular NETCONF or
   RESTCONF users to a preconfigured subset of all available NETCONF or
   RESTCONF protocol operations and content.

   The YANG module in this document makes use of grouping that are
   defined in YANG Groupings for HTTP Clients and HTTP Servers
   [I-D.ietf-netconf-http-client-server], and A YANG Data Model for SNMP
   Configuration [RFC7407]. Please see the Security Considerations
   section of those documents for considerations related to sensitivity
   and vulnerability of the data nodes defined in them.

   There are a number of data nodes defined in this YANG module that are
   writable/creatable/deletable (i.e., config true, which is the
   default). These data nodes may be considered sensitive or vulnerable
   in some network environments. Write operations (e.g., edit-config)
   to these data nodes without proper protection can have a negative
   effect on network operations. These are the subtrees and data nodes
   and their sensitivity/vulnerability:

   o The "path" node in "ietf-subscribed-notif-receivers" module can be
      modified by a malicious user to point to an invalid URI.

   Some of the readable data nodes in YANG module may be considered
   sensitive or vulnerable in some network environments. It is thus
   important to control read access (e.g., via get, get-config, or
   notification) to these data nodes. The model does not define any
   readable subtrees and data nodes.

Some of the RPC operations in YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control access to these operations. The model does not define any RPC operations.

8. IANA Considerations

8.1. The "IETF XML" Registry

This document registers two URIs in the "ns" subregistry of the "IETF XML" registry [RFC3688]. Following the format in [RFC3688], the following registrations are requested:

URI: urn:ietf:params:xml:ns:yang:ietf-subscribed-notif-receivers
Registrant Contact: The IESG
XML: N/A, the requested URI is an XML namespace.

URI: urn:ietf:params:xml:ns:yang:ietf-https-notif-transport
Registrant Contact: The IESG
XML: N/A, the requested URI is an XML namespace.

8.2. The "YANG Module Names" Registry

This document registers two YANG modules in the "YANG Module Names" registry [RFC6020]. Following the format in [RFC6020], the following registrations are requested:

```
name:         ietf-subscribed-notif-receivers
namespace:    urn:ietf:params:xml:ns:yang:ietf-subscribed-notif-receivers
prefix:       snr
reference:    RFC XXXX

name:         ietf-https-notif-transport
namespace:    urn:ietf:params:xml:ns:yang:ietf-https-notif-transport
prefix:       hnt
reference:    RFC XXXX
```

8.3. The "Capabilities for HTTPS Notification Receivers" Registry

Following the guidelines defined in [RFC5226], this document defines a new registry called "Capabilities for HTTPS Notification Receivers". This registry defines capabilities that can be supported by HTTPS-based notification receivers.

The following note shall be at the top of the registry:

This registry defines capabilities that can be supported by HTTPS-based notification receivers.

The fields for each registry are:

o URN

   * The name of the URN (required).

   * The URN must conform to the syntax described by [RFC8141].

   * The URN must begin with the string "urn:ietf:capability:https-
      notif-receiver".

o Reference

   * The RFC that defined the URN.

   * The RFC must be in the form "RFC <Number>: <Title>.

o Description

   * An arbitrary description of the algorithm (optional).

   * The description should be no more than a few sentences.

   * The description is to be in English, but may contain UTF-8
      characters as may be needed in some cases.

 The update policy is either "RFC Required". Updates do not otherwise
 require an expert review by a Designated Expert.

 Following is the initial assignment for this registry:

  Record:
     Name: urn:ietf:capability:https-notif-receiver:encoding:json
     Reference: RFC XXXX
     Description: Identifies support for JSON-encoded notifications.

  Record:
     Name: urn:ietf:capability:https-notif-receiver:encoding:xml
     Reference: RFC XXXX
     Description: Identifies support for XML-encoded notifications.

9. References

9.1. Normative references

[I-D.ietf-netconf-http-client-server]
          Watsen, K., "YANG Groupings for HTTP Clients and HTTP
          Servers", draft-ietf-netconf-http-client-server-05 (work
          in progress), August 2020.

[I-D.ietf-netconf-notification-messages]
          Voit, E., Jenkins, T., Birkholz, H., Bierman, A., and A.
          Clemm, "Notification Message Headers and Bundles", draft-
          ietf-netconf-notification-messages-08 (work in progress),
          November 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>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688,
          DOI 10.17487/RFC3688, January 2004,
          <https://www.rfc-editor.org/info/rfc3688>.

[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>.

[RFC5277] Chisholm, S. and H. Trevino, "NETCONF Event
          Notifications", RFC 5277, DOI 10.17487/RFC5277, July 2008,
          <https://www.rfc-editor.org/info/rfc5277>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for
          the Network Configuration Protocol (NETCONF)", RFC 6020,
          DOI 10.17487/RFC6020, October 2010,
          <https://www.rfc-editor.org/info/rfc6020>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed.,
          and A. Bierman, Ed., "Network Configuration Protocol
          (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011,
          <https://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure
          Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011,
          <https://www.rfc-editor.org/info/rfc6242>.

[RFC7407] Bjorklund, M. and J. Schoenwaelder, "A YANG Data Model for
          SNMP Configuration", RFC 7407, DOI 10.17487/RFC7407,
          December 2014, <https://www.rfc-editor.org/info/rfc7407>.

   [RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language",
              RFC 7950, DOI 10.17487/RFC7950, August 2016,
              <https://www.rfc-editor.org/info/rfc7950>.

   [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>.

   [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>.

   [RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration
              Access Control Model", STD 91, RFC 8341,
              DOI 10.17487/RFC8341, March 2018,
              <https://www.rfc-editor.org/info/rfc8341>.

   [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol
              Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018,
              <https://www.rfc-editor.org/info/rfc8446>.

   [RFC8639] Voit, E., Clemm, A., Gonzalez Prieto, A., Nilsen-Nygaard,
              E., and A. Tripathy, "Subscription to YANG Notifications",
              RFC 8639, DOI 10.17487/RFC8639, September 2019,
              <https://www.rfc-editor.org/info/rfc8639>.

9.2. Informative references

   [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an
              IANA Considerations Section in RFCs", RFC 5226,
              DOI 10.17487/RFC5226, May 2008,
              <https://www.rfc-editor.org/info/rfc5226>.

   [RFC8141] Saint-Andre, P. and J. Klensin, "Uniform Resource Names
              (URNs)", RFC 8141, DOI 10.17487/RFC8141, April 2017,
              <https://www.rfc-editor.org/info/rfc8141>.

Appendix A. Configuration Examples

   This non-normative section shows two examples for how the "ietf-
   https-notif-transport" module can be used to configure a publisher to
   send notifications to a receiver.

   In both examples, the Publisher, acting as an HTTPS client, is
   configured to send notifications to a receiver at address 192.0.2.1,
   port 443, and configures the "path" leaf value to "/some/path", with
   server certificates, and the corresponding trust store that is used
   to authenticate a connection.

A.1. Using Subscribed Notifications (RFC 8639)

   This example shows how an RFC 8639 [RFC8639] based publisher can be
   configured to send notifications to a receiver.

[note: '\' line wrapping for formatting only]

```
<?xml version="1.0" encoding="UTF-8"?>
<config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <subscriptions xmlns="urn:ietf:params:xml:ns:yang:ietf-subscribed-n\
otifications">
    <receiver-instances xmlns="urn:ietf:params:xml:ns:yang:ietf-subsc\
ribed-notif-receivers">
      <receiver-instance>
        <name>global-receiver-def</name>
        <https-receiver xmlns="urn:ietf:params:xml:ns:yang:ietf-https\
-notif-transport"
          xmlns:x509c2n="urn:ietf:params:xml:ns:yang:ietf-x509-cert-t\
o-name">
          <tls>
            <tcp-client-parameters>
              <remote-address>receiver.example.com</remote-address>
              <remote-port>443</remote-port>
            </tcp-client-parameters>
            <tls-client-parameters>
              <server-authentication>
                <ca-certs>
                  <local-definition>
                    <certificate>
                      <name>Server Cert Issuer #1</name>
                      <cert-data>base64encodedvalue==</cert-data>
                    </certificate>
                  </local-definition>
                </ca-certs>
              </server-authentication>
            </tls-client-parameters>
            <http-client-parameters>
              <client-identity>
                <basic>
                  <user-id>my-name</user-id>
                  <cleartext-password>my-password</cleartext-password\
>
                </basic>
              </client-identity>
              <path>/some/path</path>
            </http-client-parameters>
          </tls>
          <receiver-identity>
```

```
                  <cert-maps>
                    <cert-to-name>
                      <id>1</id>
                      <fingerprint>11:0A:05:11:00</fingerprint>
                      <map-type>x509c2n:san-any</map-type>
                    </cert-to-name>
                  </cert-maps>
                </receiver-identity>
              </https-receiver>
            </receiver-instance>
          </receiver-instances>
          <subscription>
            <id>6666</id>
            <transport xmlns:ph="urn:ietf:params:xml:ns:yang:ietf-https-not\
if-transport">ph:https</transport>
            <stream-subtree-filter>some-subtree-filter</stream-subtree-filt\
er>
            <stream>some-stream</stream>
            <receivers>
              <receiver>
                <name>subscription-specific-receiver-def</name>
                <receiver-instance-ref xmlns="urn:ietf:params:xml:ns:yang:i\
etf-subscribed-notif-receivers">global-receiver-def</receiver-instance-ref>
              </receiver>
            </receivers>
          </subscription>
        </subscriptions>
        <truststore xmlns="urn:ietf:params:xml:ns:yang:ietf-truststore">
          <certificate-bags>
            <certificate-bag>
              <name>explicitly-trusted-server-ca-certs</name>
              <description>
                Trust anchors (i.e. CA certs) that are used to
                authenticate connections to receivers.  Receivers
                are authenticated if their certificate has a chain
                of trust to one of these CA certificates.
                certificates.
              </description>
              <certificate>
                <name>ca.example.com</name>
                <cert-data>base64encodedvalue==</cert-data>
              </certificate>
              <certificate>
                <name>Fred Flintstone</name>
                <cert-data>base64encodedvalue==</cert-data>
              </certificate>
            </certificate-bag>
          </certificate-bags>
```

```
  </truststore>
</config>
```

A.2. Not Using Subscribed Notifications

In the case that it is desired to use HTTPS-based notifications
outside of Subscribed Notifications, an application-specific module
would to need define the configuration for sending the notification.

Following is an example module. Note that the module is "uses" the
"https-receiver-grouping" grouping from the "ietf-https-notif-
transport" module.

[note: '\' line wrapping for formatting only]

```
module example-custom-module {
  yang-version 1.1;
  namespace "http://example.com/example-custom-module";
  prefix "custom";

  import ietf-https-notif-transport {
    prefix "hnt";
    reference
      "RFC XXXX:
        An HTTPS-based Transport for Configured Subscriptions";
  }

  organization
    "Example, Inc.";

  contact
    "Support at example.com";

  description
    "Example of module not using Subscribed Notifications module.";

  revision "2021-02-02" {
    description
      "Initial Version.";
    reference
      "RFC XXXX, YANG Data Module for HTTPS Notifications.";
  }

  container example-module {
    description
      "Example of using HTTPS notif without having to
       implement Subscribed Notifications.";
```

```
      container https-receivers {
        description
          "A container of all HTTPS notif receivers.";
        list https-receiver {
         key "name";
          description
            "A list of HTTPS nofif receivers.";
         leaf name {
           type string;
           description
             "A unique name for the https notif receiver.";
         }
          uses hnt:https-receiver-grouping;
        }
      }
    }
  }
```

Following is what the corresponding configuration looks like:

[note: '\' line wrapping for formatting only]

```
<?xml version="1.0" encoding="UTF-8"?>
<config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <example-module xmlns="http://example.com/example-custom-module">
    <https-receivers>
      <https-receiver>
        <name>foo</name>
        <tls>
          <tcp-client-parameters>
            <remote-address>receiver.example.com</remote-address>
            <remote-port>443</remote-port>
          </tcp-client-parameters>
          <tls-client-parameters>
            <server-authentication>
              <ca-certs>
                <local-definition>
                  <certificate>
                    <name>Server Cert Issuer #1</name>
                    <cert-data>base64encodedvalue==</cert-data>
                  </certificate>
                </local-definition>
              </ca-certs>
            </server-authentication>
          </tls-client-parameters>
          <http-client-parameters>
            <client-identity>
              <basic>
```

```
                          <user-id>my-name</user-id>
                          <cleartext-password>my-password</cleartext-password>
                        </basic>
                    </client-identity>
                    <path>/some/path</path>
                  </http-client-parameters>
                </tls>
            </https-receiver>
          </https-receivers>
        </example-module>
        <truststore xmlns="urn:ietf:params:xml:ns:yang:ietf-truststore">
          <certificate-bags>
            <certificate-bag>
              <name>explicitly-trusted-server-ca-certs</name>
              <description>
                Trust anchors (i.e. CA certs) that are used to
                authenticate connections to receivers.  Receivers
                are authenticated if their certificate has a chain
                of trust to one of these CA certificates.
              </description>
              <certificate>
                <name>ca.example.com</name>
                <cert-data>base64encodedvalue==</cert-data>
              </certificate>
              <certificate>
                <name>Fred Flintstone</name>
                <cert-data>base64encodedvalue==</cert-data>
              </certificate>
            </certificate-bag>
          </certificate-bags>
        </truststore>
      </config>
```

Acknowledgements

   The authors would like to thank for following for lively discussions
   on list and in the halls (ordered by first name): Eric Voit, Henning
   Rogge, Martin Bjorklund, Reshad Rahman, and Rob Wilton.

Authors' Addresses

   Mahesh Jethanandani
   Kloud Services

   Email: mjethanandani@gmail.com

   Kent Watsen
   Watsen Networks

   Email: kent+ietf@watsen.net