NETCONF
Internet-Draft
Intended status: Standards Track
Expires: September 19, 2018

G. Zheng
T. Zhou
A. Clemm
Huawei

March 18, 2018

UDP based Publication Channel for Streaming Telemetry
draft-ietf-netconf-udp-pub-channel-02

Abstract

This document describes a UDP-based publication channel for streaming telemetry use to collect data from devices. A new shim header is proposed to facilitate the distributed data collection mechanism which directly pushes data from line cards to the collector. Because of the lightweight UDP encapsulation, higher frequency and better transit performance can be achieved.

Requirements Language

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].

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 19, 2018.

Table of Contents

1. Introduction

Streaming telemetry refers to sending a continuous stream of operational data from a device to a remote receiver. This provides an ability to monitor a network from remote and to provide network analytics. Devices generate telemetry data and push that data to a collector for further analysis. By streaming the data, much better performance, finer-grained sampling, monitoring accuracy, and bandwidth utilization can be achieved than with polling-based alternatives.

Sub-Notif [I-D.ietf-netconf-subscribed-notifications] and YANG-Push [I-D.ietf-netconf-yang-push] defines a mechanism that allows a collector to subscribe to updates of YANG-defined data that is maintained in a YANG [RFC7950] datastore. The mechanism separates the management and control of subscriptions from the transport that is used to actually stream and deliver the data. Two transports, NETCONF transport [I-D.ietf-netconf-netconf-event-notifications] and HTTP transport [I-D.ietf-netconf-restconf-notif], have been defined so far for the notification messages.

While powerful in its features and general in its architecture, in its current form the mechanism needs to be extended to stream telemetry data at high velocity from devices that feature a distributed architecture. The transports that have been defined so far, NETCONF and HTTP, are ultimately based on TCP and lack the efficiency needed to stream data continuously at high velocity. A lighter-weight, more efficient transport, e.g. a transport based on UDP is needed.

- Firstly, data collector will suffer a lot of TCP connections from, for example, many line cards equipped on different devices.

- Secondly, as no connection state needs to be maintained, UDP encapsulation can be easily implemented by hardware which will further improve the performance.

- Thirdly, because of the lightweight UDP encapsulation, higher frequency and better transit performance can be achieved, which is important for streaming telemetry.

This document specifies a higher-performance transport option for YANG-Push that leverages UDP. Specifically, it facilitates the distributed data collection mechanism described in [I-D.zhou-netconf-multi-stream-originators]. In the case of data originating from multiple line cards, the centralized design requires data to be internally forwarded from those line cards to the push server, presumably on a main board, which then combines the individual data items into a single consolidated stream. The centralized data collection mechanism can result in a performance bottleneck, especially when large amounts of data are involved. What is needed instead is the support for a distributed mechanism that allows to directly push multiple individual substreams, e.g. one from each line card, without needing to first pass them through an additional processing stage for internal consolidation, but still allowing those substreams to be managed and controlled via a single subscription. The proposed UDP based Publication Channel (UPC) natively supports the distributed data collection mechanism.

The transport described in this document can be used for transmitting notification messages over both IPv4 and IPv6 [RFC8200].

While this document will focus on the data publication channel, the subscription can be used in conjunction with the mechanism proposed in [I-D.ietf-netconf-yang-push] with extensions [I-D.zhou-netconf-multi-stream-originators].

2. Terminology

Streaming Telemetry: refers to sending a continuous stream of operational data from a device to a remote receiver. This provides an ability to monitor a network from remote and to provide network analytics.

3. Solution Overview

The typical distributed data collection solution is shown in Fig. 1. Both the Collector and the Subscribed Domain can be distributed. The Collector includes the Subscriber and a set of Receivers. And the Subscribed Domain includes a Master and a set of Agents. The Subscriber cannot see the Agents directly, so it will send the Global Subscription information to the Master (e.g., main board) via the Subscription Channel. When receiving a Global Subscription, the Master decomposes the subscription request into multiple Component Subscriptions, each involving data from a separate internal telemetry source, for example a line card. The Component Subscriptions are distributed to the Agents. Subsequently, each data originator generates its own stream of telemetry data, collecting and encapsulating the packets per the Component Subscription and streaming them to the designated Receivers. This distributed data collection mechanism may form multiple Publication Channels between the Data Originators and the Receivers. The Collector is able to assemble many pieces of data associated with one Global Subscription.

The Publication Channel supports the reliable data streaming, for example for some alarm events. The Collector has the option of deducing the packet loss and the disorder based on the information carried by the notification data. And the Collector will decide the behavior to request retransmission.

The rest of the draft describes the UDP based Publication Channel (UPC).

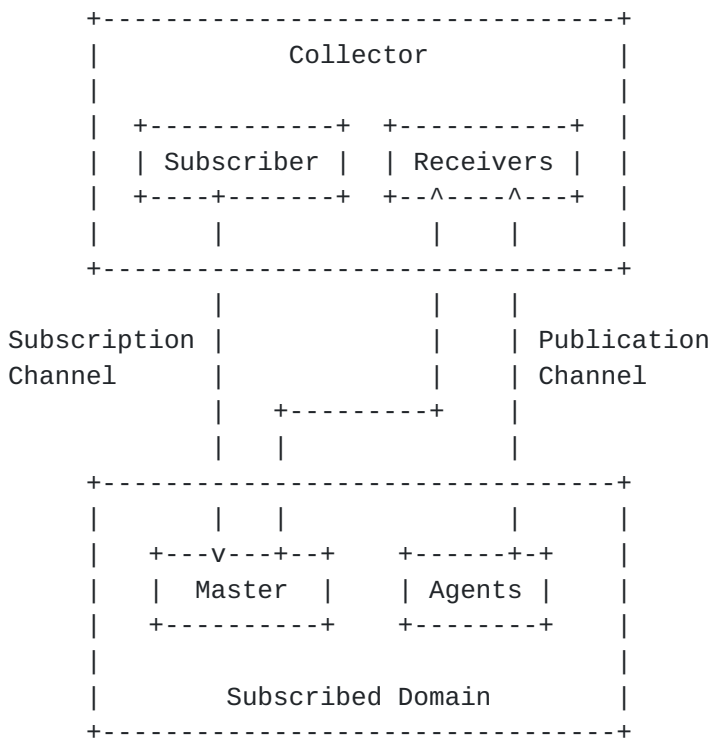

Fig. 1 Distributed Data Collection

4. Transport Mechanisms

For a complete pub-sub mechanism, this section will describe how the UPC is used to interact with the Subscription Channel relying on NETCONF or RESTCONF.

4.1. Dynamic Subscription

Dynamic subscriptions for YANG-Push [I-D.ietf-netconf-yang-push] are configured and managed via signaling messages transported over NETCONF [RFC6241] or RESTCONF [RFC8040]. The YANG-Push defined RPCs are sent and responded via the Subscription Channel (a), between the Subscriber and the Master of the Subscribed Domain. In this case, only one Receiver is associated with the Subscriber. In the Subscribed Domain, there may be multiple Data Originators. Notification messages are pushed on separate channels (b), from different Data Originators to the Receiver .

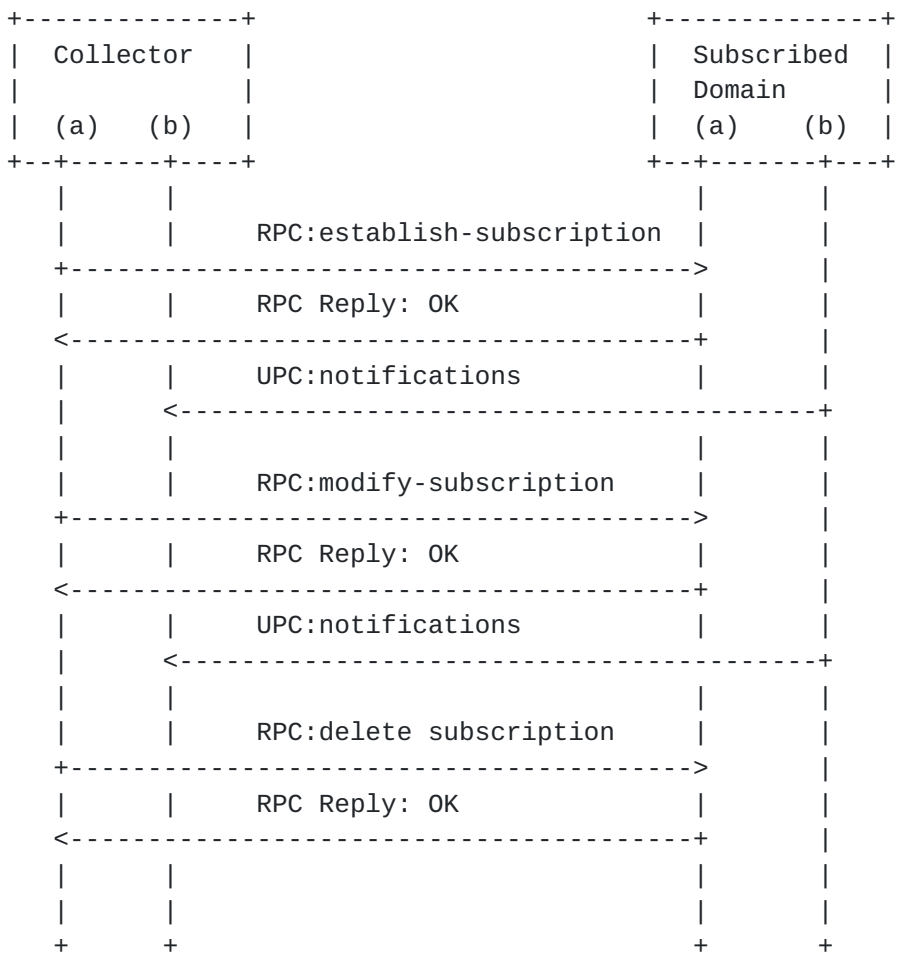

Fig. 2 Call Flow for Dynamic Subscription

In the case of dynamic subscription, the Receiver and the Subscriber SHOULD be collocated. So UPC can use the source IP address of the Subscription Channel as it's destination IP address. The Receiver MUST support listening messages at the IANA-assigned PORT-X, but MAY be configured to listen at a different port.

4.2. Configured Subscription

For a Configured Subscription, there is no guarantee that the Subscriber is currently in place with the associated Receiver(s). As defined in [I-D.ietf-netconf-yang-push], the subscription configuration contains the location information of all the receivers, including the IP address and the port number. So that the Data Originator can actively send generated messages to the corresponding Receivers via the UPC.

The first message MUST be a separate subscription-started notification to indicate the Receiver that the pushing is started. Then, the notifications can be sent immediately without any wait.

All the subscription state notifications, as defined in [I-D.ietf-netconf-subscribed-notifications], MUST be encapsulated to be separated notification messages.

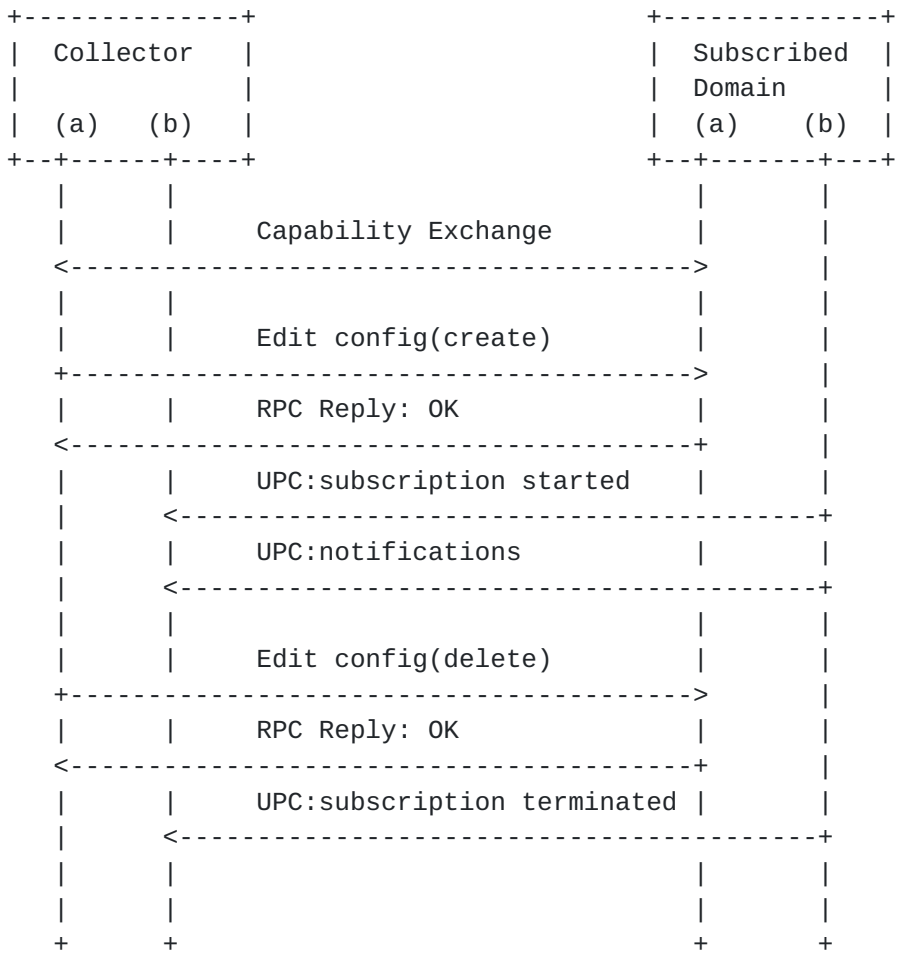

Fig. 3 Call Flow for Configured Subscription

5. UDP Transport for Publication Channel

5.1. Design Overview

As specified in YANG-Push, the telemetry data is encapsulated in the NETCONF/RESTCONF notification message, which is then encapsulated and carried in the transport protocols, e.g. TLS, HTTP2. The following figure shows the overview of the typical UDP publication message structure.

- The Message Header contains information that can facilitate the message transmission before de-serializing the notification message.

- Notification Message is the encoded content that the publication channel transports. The common encoding method includes GPB [1], CBOR [RFC7049], JSON, and XML. [I-D.ietf-netconf-notification-messages] describes the structure of the Notification Message for both single notification and multiple bundled notifications.

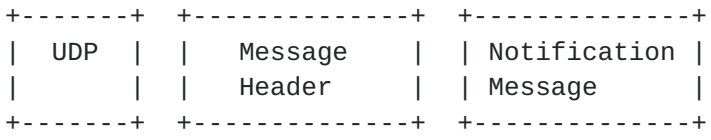

Fig. 4 UDP Publication Message Overview

5.2. Data Format of the Message Header

The Message Header contains information that can facilitate the message transmission before de-serializing the notification message. The data format is shown as follows.

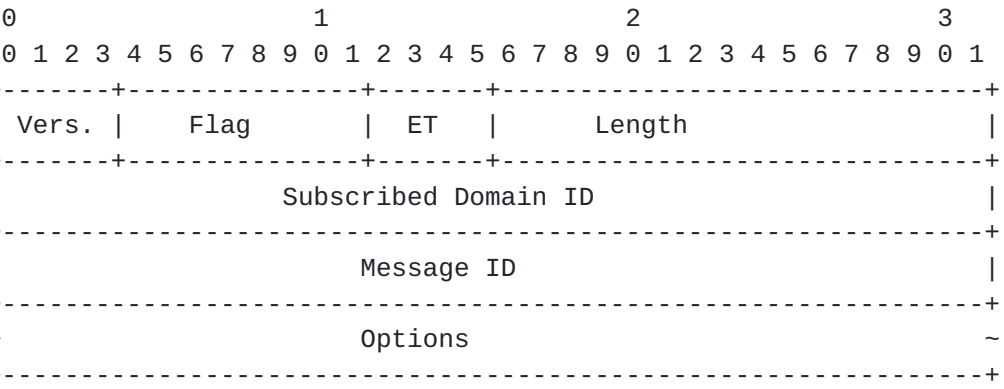

Fig. 5 Message Header Format

The Message Header contains the following field:

- Vers.: represents the PDU (Protocol Data Unit) encoding version. The initial version value is 0.

- Flag: is a bitmap indicating what features this packet has and the corresponding options attached. Each bit associates to one feature and one option data. When the bit is set to 1, the associated feature is enabled and the option data is attached.

The sequence of the presence of the options follows the bit order of the bitmap. In this document, the flag is specified as follows:

* bit 0, the reliability flag;

* bit 1, the fragmentation flag;

* other bits are reserved. All the reserved bits MUST be set to 0.

- ET: is a 4 bits identifier to indicate the encoding type used for the Notification Message. While 16 types of encoding can be expressed, this document specifies the following usage:

  * 0: GPB;

  * 1: CBOR;

  * 2: JSON;

  * 3: XML;

  * others are reserved.

- Length: is the total length of the message, measured in octets, including message header. If the notification message is fragmented, this Length indicates the actual length of the current message fragmentation.

- Subscribed Domain ID: is a 32-bit identifier of the Subscribed Domain. With this parameter, the receiver can easily identify messages generated from the same Subscription Domain. One possible value is the visible IPv4 address of the Master.

- The Message ID is generated continuously by the Data Originator. Different subscribers share the same Message ID sequence. Different fragmentations of one message share the same Message ID.

- Options: is a variable-length field. The details of the Options will be described in the respective sections below.

5.3. Options

The order of packing the data fields in the Options field follows the bit order of the Flag field.

5.3.1. Reliability Option

The UDP based publication transport described in this document provides two streaming modes, the reliable mode an the unreliable mode, for different SLA (Service Level Agreement) and telemetry requirements.

In the unreliable streaming mode, the line card pushes the encapsulated data to the data collector without any sequence information. So the subscriber does not know whether the data is correctly received or not.

The reliable streaming mode provides sequence information in the UDP packet, based on which the subscriber can deduce the packet loss and disorder. Then the subscriber can decide whether to request the retransmission of the lost packets.

In most case, the unreliable streaming mode is preferred. Because the reliable streaming mode will cost more network bandwidth and precious device resource. Different from the unreliable streaming mode, the line card cannot remove the sent reliable notifications immediately, but to keep them in the memory for a while. Reliable notifications may be pushed multiple times, which will increase the traffic. When choosing the reliable streaming mode or the unreliable streaming mode, the operate need to consider the reliable requirement together with the resource usage.

When the reliability flag bit is set to 1 in the Flag field, the following option data will be attached

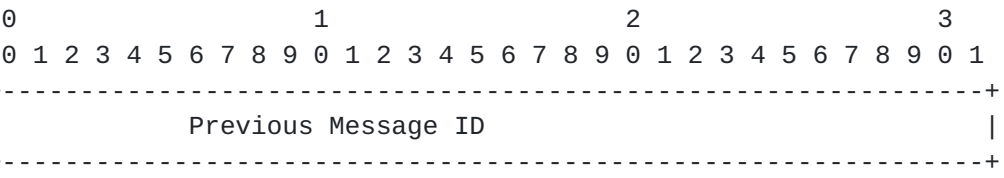

Fig. 4 Reliability Option Format

The Data Originator has the capability of index the Previous Message ID for the message. Together with the current Message ID, the Receiver can detect whether the current message is in a right order.

For example, there are two subscriber A and B,

- Message IDs for the generator are : [1, 2, 3, 4, 5, 6, 7, 8, 9], in which Subscriber A subscribes [1,2,3,6,7] and Subscriber B subscribes [1,2,4,5,7,8,9].

- Subscriber A will receive : [0,1][1,2][2,3][3,6][6,7].

- Subscriber B will receive : [0,1][1,2][2,4][4,5][5,7][7,8].

5.3.2. Fragmentation Option

UDP payload has a theoretical length limitation to 65535. Other encapsulation headers will make the actual payload even shorter. Binary encodings like GPB and CBOR can generate a compact notification message. So that the message can fit in one UDP packet. In this case, fragmentation will not easily happen. However, text encodings like JSON and XML can easily generate a notification message exceeding the UDP length limitation.

The fragmentation flag in the fixed header is set to 1 only when the Notification Message is actually fragmented. And the Fragmentation Option is available in the message header when the fragmentation flag is set to 1.

The Fragmentation Option is formatted as follow:

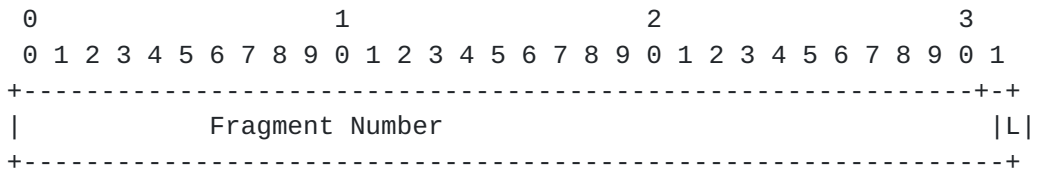

Fig. 5 Fragmentation Option Format

This option contains:

- Fragment Number: indicates the sequence number of the current fragment. Together with the Message ID, the Receiver can compose the entire Notification Message.

- L: is a flag to indicate whether the current fragment is the last one. When 0 is set, current fragment is not the last one, hence more fragments are expected. When 1 is set, current fragment is the last one.

5.4. Data Encoding

Subscribed data can be encoded in GPB, CBOR, XML or JSON format. It is conceivable that additional encodings may be supported as options in the future. This can be accomplished by augmenting the subscription data model with additional identity statements used to refer to requested encodings.

Implementation may support different encoding method per subscription. When bundled notifications is supported between the publisher and the receiver, only subscribed notifications with the same encoding can be bundled as one message.

6. Congestion Control

Congestion control mechanisms that respond to congestion by reducing traffic rates and establish a degree of fairness between flows that share the same path are vital to the stable operation of the Internet [RFC2914]. While efficient, UDP has no build-in congestion control mechanism. Because streaming telemetry can generate unlimited amounts of data, transferring this data over UDP is generally problematic. It is not recommended to use the UPC over congestion-sensitive network paths. The only environments where the UPC MAY be used are managed networks. The deployments require the network path has been explicitly provisioned for the UPC through traffic engineering mechanisms, such as rate limiting or capacity reservations.

7. IANA Considerations

This RFC requests that IANA assigns one UDP port number in the "Registered Port Numbers" range with the service names "udp-pub-ch". This port will be the default port for the UDP based publication channel for NETCONF and RESTCONF. Below is the registration template following the rules in [RFC6335].

Service Name: udp-pub-ch

Transport Protocol(s): UDP

Assignee: IESG <iesg@ietf.org>

Contact: IETF Chair <chair@ietf.org>

Description: NETCONF Call Home (SSH)

Reference: RFC XXXX

Port Number: PORT-X

8. Security Considerations

TBD

9. Acknowledgements

The authors of this documents would like to thank Eric Voit, Tim Jenkins, and Huiyang Yang for the initial comments.

10. References

10.1. 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>.

[RFC2914] Floyd, S., "Congestion Control Principles", BCP 41, RFC 2914, DOI 10.17487/RFC2914, September 2000, <https://www.rfc-editor.org/info/rfc2914>.

[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>.

[RFC7049] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", RFC 7049, DOI 10.17487/RFC7049, October 2013, <https://www.rfc-editor.org/info/rfc7049>.

[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>.

[RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <https://www.rfc-editor.org/info/rfc8200>.

10.2. Informative References

[I-D.ietf-netconf-netconf-event-notifications] Prieto, A., Voit, E., Clemm, A., Nilsen-Nygaard, E., and A. Tripathy, "NETCONF Support for Event Notifications", draft-ietf-netconf-netconf-event-notifications-08 (work in progress), February 2018.

[I-D.ietf-netconf-notification-messages]
Voit, E., Birkholz, H., Bierman, A., Clemm, A., and T. Jenkins, "Notification Message Headers and Bundles", draft-ietf-netconf-notification-messages-03 (work in progress), February 2018.

[I-D.ietf-netconf-restconf-notif]
Voit, E., Tripathy, A., Nilsen-Nygaard, E., Clemm, A., Prieto, A., and A. Bierman, "RESTCONF and HTTP Transport for Event Notifications", draft-ietf-netconf-restconf-notif-04 (work in progress), January 2018.

[I-D.ietf-netconf-subscribed-notifications]
Voit, E., Clemm, A., Prieto, A., Nilsen-Nygaard, E., and A. Tripathy, "Custom Subscription to Event Streams", draft-ietf-netconf-subscribed-notifications-10 (work in progress), February 2018.

[I-D.ietf-netconf-yang-push]
Clemm, A., Voit, E., Prieto, A., Tripathy, A., Nilsen-Nygaard, E., Bierman, A., and B. Lengyel, "YANG Datastore Subscription", draft-ietf-netconf-yang-push-15 (work in progress), February 2018.

[I-D.zhou-netconf-multi-stream-originators]
Zhou, T., Zheng, G., Voit, E., Clemm, A., and A. Bierman, "Subscription to Multiple Stream Originators", draft-zhou-netconf-multi-stream-originators-01 (work in progress), November 2017.

10.3. URIs

[1] https://developers.google.com/protocol-buffers/

Appendix A. Change Log

(To be removed by RFC editor prior to publication)

A.1. draft-ietf-zheng-udp-pub-channel-00 to v00

- Modified the message header format.
- Added a section on the Authentication Option.
- Cleaned up the text and removed unnecessary TBDs.

A.2. v01

- Removed the detailed description on distributed data collection mechanism from this document. Mainly focused on the description of a UDP based publication channel for telemetry use.

- Modified the message header format.

A.2. v02

- Add the section on the transport mechanism.

- Modified the fixed message header format.

- Add the fragmentation option for the message header.

Authors' Addresses

Guangying Zheng
Huawei
101 Yu-Hua-Tai Software Road
Nanjing, Jiangsu
China

Email: zhengguangying@huawei.com

Tianran Zhou
Huawei
156 Beiqing Rd., Haidian District
Beijing
China

Email: zhoutianran@huawei.com

Alexander Clemm
Huawei
2330 Central Expressway
Santa Clara, California
USA

Email: alexander.clemm@huawei.com