Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 21, 2018

T. Zhou
G. Zheng
Huawei
E. Voit
Cisco Systems
A. Clemm
Huawei
A. Bierman
YumaWorks

October 18, 2017

Subscription to Multiple Stream Originators
draft-zhou-netconf-multi-stream-originators-00

Abstract

This document describes the distributed data collection mechanism that allows multiple data streams to be managed using a single subscription. Specifically, multiple data streams are pushed directly to the collector without passing through a broker for internal consolidation.

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].

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.

YANG-Push [I-D.ietf-netconf-yang-push] defines a transport-independent subscription mechanism for datastore updates, in which a

subscriber can subscribe to a stream of datastore updates from a server, or update provider. The current design involves subscription to a single push server. This conceptually centralized model encounters efficiency limitations in cases where the data sources are themselves distributed, such as line cards in a piece of network equipment. In such cases, it will be a lot more efficient to have each data source (e.g., each line card) originate its own stream of updates, rather than requiring updates to be tunneled through a central server where they are combined. What is needed is a distributed mechanism that allows to directly push multiple individual data substreams, 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.

This document will describe such distributed data collection mechanism and how it can work by extending existing YANG-Push mechanism. The proposal is general enough to fit many scenarios.

2. Use Cases

2.1. Use Case 1: Data Collection from Devices with Main-board and Line-cards

For data collection from devices with main-board and line-cards, existing YANG-Push solutions consider only one push server typically reside in the main board. As shown in the following figure, data are collected from line cards and aggregate to the main board as one consolidated stream. So the main board can easily become the performance bottle-neck. The optimization is to apply the distributed data collection mechanism which can directly push data from line cards to a collector. On one hand, this will reduce the cost of scarce compute and memory resources on the main board for data processing and assembling. On the other hand, distributed data push can off-load the streaming traffic to multiple interfaces.

```
                  +--------------------------------------+
                  |              collector               |
                  +------^-----------^-----------^-------+
                         |           |           |
                         |           |           |
                  +--------------------------------------+
                  |      |           |           |       |
                  |      |     +-----+------+    |       |
                  |      |     | main board |    |       |
                  |      |     +--^-----^---+    |       |
                  |      |        |     |        |       |
                  |      |    +---+     +---+    |       |
                  |      |    |             |    |       |
                  | +----+----+---+     +---+----+----+  |
                  | | line card 1 |     | line card 2 |  |
                  | +-------------+     +-------------+  |
                  |               device                 |
                  +--------------------------------------+
```

Fig. 1 Data Collection from Devices with Main-board and Line-cards

2.2. Use Case 2: IoT Data Collection

In the IoT data collection scenario, as shown in the following figure, collector usually cannot access to IoT nodes directly, but is isolated by the border router. So the collector subscribes data from the border router, and let the border router to disassemble the subscription to corresponding IoT nodes. The border router is typically the traffic convergence point. It's intuitive to treat the border router as a broker assembling the data collected from the IoT nodes and forwarding to the collector[I-D.ietf-core-coap-pubsub]. However, the border router is not so powerful on data assembling as a network device. It's more efficient for the collector, which may be a server or even a cluster, to assemble the subscribed data if possible. In this case, push servers that reside in IoT nodes can stream data to the collector directly while traffic only passes through the border router.

```
                    +-------------------------------+
                    |           collector           |
                    +---^-----------^------------^--+
                        |           |            |
                        |           |            |
                        |           |            |
                        |   +-------+--------+   |
                        |   |  border router |   |
                        |   +----^------^----+   |
                        |        |      |        |
                        |        |      |        |
                        |    +---+      +---+    |
                        |    |              |    |
                    +---+----+---+      +---+----+---+
                    | IoT node 1 |      | IoT node 2 |
                    +------------+      +------------+
```

Fig. 2 IoT Data Collection

3. Solution Overview

All the use cases described in the previous section are very similar on the data subscription and publication mode, hence can be abstracted to the following generic distributed data collection framework, as shown in the following figure.

A Collector usually includes two components,

- o the Subscriber generates the subscription instructions to express what and how the collector want to receive the data;

- o the Receiver is the target for the data publication.

For one subscription, there may be one to many receivers. And the subscriber does not necessarily share the same address with receivers.

In this framework, the stream originators have the Master role and the Agent role. Both the Master and the Agent include two components,

- o the Subscription Server manages capabilities that it can provide to the subscriber.

- o the Publication Server pushes data to the receiver according to the subscription information.

The Master knows all the capabilities that the attached Agents and itself can provide, and exposes the global capability to the Collector. The Collector cannot see the Agents directly, so it will only send the subscription information to the Master. The Master disassembles the subscription to multiple component subscriptions, each involving data from a separate telemetry source. The component subscriptions are then distributed to the corresponding Agents.

When data streaming, the Publication Server located in each stream originator collects and encapsulates the packets per the component subscription, and pushes the piece of data which it can serve directly to the designated data Collector. The Collector is able to assemble many pieces of data associated with one subscription, and can also deduce the missing pieces of data.

```
               +-------------------------------------+
               |             Collector               |
               |   +------------+   +------------+   |
               |   | Subscriber |   |  Receiver  <-------+
               |   +-^----+-----+   +------^-----+   |   |
               |     |    |                |         |   |
               +-------------------------------------+   |
          capability |    |subscription    | push        |
                     |    |                |             |
               +-------------------------------------+   |
               |     |    |   Master       |         |   |
               |  +--+----v------+  +------+------+  |   |
               |  | Subscription |  | Publication |  |   |
               |  | Server       |  | Server      |  |   |
               |  +--^----+------+  +-------------+  |   |
               |     |    |                          |   |
               +-------------------------------------+   |
          component  |    | component              push |
          capability |    | subscription                 |
               +-------------------------------------+   |
               |     |    |   Agent                  |   |
               |  +--+----v------+  +-------------+  |   |
               |  | Component    |  | Publication |  |   |
               |  | Subscription |  | Server      +-------+
               |  | Server       |  +-------------+  |
               |  +--------------+                   |
               +-------------------------------------+
```

Fig. 3 The Generic Distributed Data Collection Framework

Master and Agents may interact with each other in several ways:

- o Agents need to have a registration or announcement handshake with the Master Node, so the Master Node is aware of them and of lifecycle events (such as Agent appearing and disappearing).

- o The Master relays the component subscriptions to the Agents.

- o The Agents indicate status of component subscriptions to the Master. The status of the overall subscription is maintained by the Master. The Master is also responsible for notifying the subscriber in case of any problems of component subscriptions.

The details of the interaction between the Master and the Agent is out of scope of this document. It may be described by a dedicated protocol specification.

During the subscription and the associated publication process, this document assumes all the Agents registered to the Mater can always provide the announced capabilities.

Note: Some preliminary considerations on the solution details are now listed in the appendix for reference. The detailed solution need to be discussed and will be added if the WG accepts the problem statement.

4. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

5. Security Considerations

6. Acknowledgements

7. References

7.1. 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>.

[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>.

[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>.

[W3C.REC-xpath-19991116] Clark, J. and S. DeRose, "XML Path Language (XPath) Version 1.0", World Wide Web Consortium Recommendation REC-xpath-19991116, November 1999, <http://www.w3.org/TR/1999/REC-xpath-19991116>.

7.2. Informative References

[I-D.ietf-core-coap-pubsub] Koster, M., Keranen, A., and J. Jimenez, "Publish-Subscribe Broker for the Constrained Application Protocol (CoAP)", draft-ietf-core-coap-pubsub-02 (work in progress), July 2017.

[I-D.ietf-netconf-yang-push] Clemm, A., Voit, E., Prieto, A., Tripathy, A., Nilsen-Nygaard, E., Bierman, A., and B. Lengyel, "Subscribing to YANG datastore push updates", draft-ietf-netconf-yang-push-10 (work in progress), October 2017.

Appendix A. Change Log

(To be removed by RFC editor prior to publication)

Appendix B. Subscription Decomposition

Since Agents are invisible to the Collector, the Collector can only subscribe to the Master. This requires the Master to:

1. expose the global capability that can be served by multiple stream originators;

2. disassemble the subscription to multiple component subscriptions, and distribute them to the corresponding telemetry sources.

To achieve the above requirement, the Master need a global capability description which is typically the YANG [RFC7950] data model. This global YANG model is provided as the contract between the Master and the Collector.

The Master also need a data structure, typically a table as shown below, to keep track the mapping between the resource and the corresponding location identifier of the node that commits to serve

the data. For the YANG defined capabilities, the resource is described using the XPath [W3C.REC-xpath-19991116] expression.

resource	location ID

Table 1

Each Agent associating to the Master owns a local YANG model to describe the capabilities which it can serve as part of the global capability. All the Agents need to know the namespace associate with the Master.

YANG-Push supports two filtering syntaxes which are XPath and Subtree [RFC6241]. When a YANG-Push subscription request arrives, the Master Node will firstly extract the filter information. Consequently, according to the resource-locationID table, the master subscription can be disassembled into multiple component subscriptions, and the corresponding location ID can be associated. The component subscriptions share the same Subscription ID as the master subscription.

Appendix C. Publication Composition

The Publication Server collects and encapsulates the packets per the component subscription. There are several potential encodings, including XML, JSON, CBOR and GPB. The encoding of the data records follows the YANG schema, so that the composition at the Receiver can benefit from the structured and hierarchical data instance. The Collector may be able to assemble many pieces of data associated with one subscription, and can also deduce the missing pieces of data.

The Collector recognizes data records associated with one subscription according the Subscription ID. Data records generated per one subscription are assigned with the same Subscription ID.

For the time series data stream, records are produced periodically from each stream originator. The message arrival time varies because of the distributed nature of the publication. The receiver assembles data generated at the same time period based on the recording time consisted in each data record. In this case, time synchronization is required for all the steam originators.

Appendix D. Examples

TBD

Authors' Addresses

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

Email: zhoutianran@huawei.com

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

Email: zhengguangying@huawei.com

Eric Voit
Cisco Systems
United States of America

Email: evoit@cisco.com

Alexander Clemm
Huawei
2330 Central Expressway
Santa Clara, California
United States of America

Email: alexander.clemm@huawei.com

Andy Bierman
YumaWorks
United States of America

Email: andy@yumaworks.com