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

A. Clemm
Futurewei Technologies, Inc.
E. Voit
Cisco Systems
X. Liu
Jabil
I. Bryskin
T. Zhou
G. Zheng
Huawei
H. Birkholz
Fraunhofer SIT

October 18, 2017

Smart filters for Push Updates - Problem Statement
draft-clemm-netconf-push-smart-filters-ps-00

Abstract

This document defines a problem statement for Smart Filters for Push Updates. Smart Filters for Push Updates (referred to simply as "Smart Filters" in the context of this document) allows to filter push updates based on values of pushed objects and/or state, such as previous updates. Smart Filters provide an important building block for service assurance and network automation.

Status of This Memo

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

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

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

This Internet-Draft will expire on April 21, 2018.

Table of Contents

1. Introduction 2
2. Key Words 3
3. Definitions and Acronyms 3
4. Problem Statement 3
5. IANA Considerations 6
6. Security Considerations 6
7. Normative References 6
Authors' Addresses 7

1. Introduction

YANG-Push [yang-push] allows client applications to subscribe to continuous datastore updates without needing to poll. YANG-Push subscriptions allow client applications to select which datanodes are of interest. For this purpose, filters that act as node selectors are offered. However, what is currently not supported are filters that filter updates based on values, such as sending updates only when the value falls within a certain range. Also not supported are filters that would require additional state, such as sending updates only when the value exceeds a certain threshold for the first time but not again until the threshold is cleared. We refer to such filters as "smart filters", with further subcategories of "smart stateless filters" and "smart stateful filters", respectively.

Smart filters involve more complex subscription and implementation semantics than the simple selection filters that are currently offered as part of YANG-Push. They involve post processing of updates that goes beyond basic update generation for polling avoidance and place additional intelligence at the server. Because of this, smart filter functionality was not included in the YANG-Push specification, although it was recognized that YANG-Push could be

extended to include such functionality if needed. This is the purpose of this specification.

Smart filters facilitate service assurance, because they allow client applications to focus on "outliers" and updates that signify exceptions and conditions of interest have the biggest operational significance. They save network resources by avoiding the need to stream updates that would be discarded anyway, and allow applications to scale better since larger networks imply a larger amount of smart filtering operations delegated away from the application to the network. Smart filters also facilitate network automation as they constitute an important ingredient to specify triggers for automated actions.

2. Key Words

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.

3. Definitions and Acronyms

Smart Filter: A filter that involves some processing, such as comparing values or differentiating behavior depending on state.

TCA: Threshold Crossing Alert.

YANG-Push: A server capability that allows client applications to subscribe to network management datastore updates.

4. Problem Statement

YANG-Push provides client applications with the ability to subscribe to continuous updates from network management datastores, obviating the need to perform polling and resulting in more robust and efficient applications. However, many applications do not require every update, only updates that are of certain interest.

For example, an update concerning interface utilization may be only needed when a certain utilization level is breached. Sending continuous updates when utilization is low might divert processing resources away from updates regarding interfaces whose utilization level may reach a critical point that requires attention. Doing so will require a filter based on an object value. Even sending continuous updates when utilization is high may be too much and counterproductive. It may be sufficient to send an update when a

threshold is breached to raise a flag of attention, but then not to continue sending updates while the condition still persists but simply let the client application know when the threshold is cleared. This behavior cannot be accomplished simply by a value-based filter, but requires additional state to be maintained (so that the server has a memory whether or not the condition of a breached threshold has already been reported in prior update cycles).

What is needed are "Smart Filters" that provide the ability to apply filters based on object values, possibly also state. Smart Filters are useful for Service Assurance applications that need to monitor operational data for values that fall outside normal operational ranges. They are also useful for network automation, in which automated actions are automatically triggered based on when certain events in the network occur while certain conditions hold. A YANG-Push subscription with a smart filter can in effect act as a source for such events. Combined with an optional check for a condition when an event is observed, this can serve as the basis of action triggers.

Of course, it is possible to conceive filters that are very smart and powerful yet also very complex. While filters as defined in YANG-Push may be a tad too simple for the applications envisioned here, it is important to keep filters still simple enough to ensure broad implementation and support by networking devices. The smart filters defined in this effort intend to apply the "90/10" rule, aiming at the sweet spot that addresses 90% of use cases and deployment scenarios that can be addressed using 10% of the complexity. Where those filters are not sufficient, additional filters can be introduced outside this document.

It is proposed that Smart Filters for Push Updates will provide support for the following features:

- Support for smart filter extensions to YANG-Push subscriptions. The targeted model takes a "base" YANG-Push subscription and subjects updates to an additional filtering stage that is based on an object’s value.
  - Filters that match or compare the object value against a fixed term or expression.
  - Filters that match or compare an object value against the value of another object. (This feature is useful particularly in conjunction with possible extensions that allow to compute aggregates. Such an ability opens the possibility to compare an object’s current value to its mean, for example.)

- Support for refined on-change update semantics that allow client to distinguish whether object values were omitted or included because the object was created or deleted, or because the object's value fell outside filter range.

- Support for selected stateful filters:

  * This includes specifically support for generalized "threshold crossing alert" filters, or filters that provide an update only when an object's value passes a filter for the first time, and not again until the object's value passes a counter filter. In effect, the support involves attaching filter and counter filter to an object, including a switch at the object indicating which filter is in effect, and providing a distinction in the update which filter (e.g. onset of clear) was applied.

  * It may include additional filters, such a "recent high water mark" filters that allow to specify a time horizon until the current high water mark clears. A recent high water mark filter sends an update to an object only if its new value is greater than the last value that had been previously reported.

In order to constrain complexity, it is proposed that the following items will be outside the scope, subject to discussion by the Working Group:

- Filters that involve freely programmable logic.

- Filters that aggregate or otherwise process information over time. An example would be filters that compute an aggregate over a time series of data, for example, an object's average or top percentile value. (One way in which this can be accomplished is by defining a separate YANG module that allows to specify aggregates independent of any filtering, then asking users to subscribe to updates of the aggregate objects and applying filters there. The definition of such an aggregation module goes beyond the scope of the work defined here.)

- Filters that aggregate object's values with those of other objects, such as the maximum or average from objects over a list, or that operate on a function of other objects. An example would be an object for interface utilization that gets computed from objects for interface speed and interface packet rate, with the packet rate object itself potentially computed from counter snapshots that are taken at different times. (One way in which this can be accomplished is by defining a separate YANG module that allows to define objects that compute such functions akin to

the Expression MIB [RFC2982], then asking users to subscribe to updates of those objects and applying filters there.)

5. IANA Considerations

Not applicable

6. Security Considerations

The application of smart filters requires a certain amount of processing resources at the server. An attacker could attempt to attack a server by creating YANG-push subscriptions with a large number of complex smart filters in an attempt to diminish server resources. Server implementations can guard against such scenarios in several ways. For one, they can implement NACM [RFC6536] in order to require proper authorization for requests to be made. Second, server implementations can reject requests made for a a larger number of smart filters than the implementation can reasonably sustain.

7. Normative References

[notif-sub] Voit, E., Clemm, A., Gonzalez Prieto, A., Nilsen-Nygaard, E., and A. Tripathy, "Custom subscription to event notifications", July 2017, <https://datatracker.ietf.org/doc/draft-ietf-netconf-subscribed-notifications/>.

[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>.

[RFC2982] Kavasseri, R., Ed., "Distributed Management Expression MIB", RFC 2982, DOI 10.17487/RFC2982, October 2000, <https://www.rfc-editor.org/info/rfc2982>.

[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>.

[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>.

[yang-push]
Clemm, A., Voit, E., Gonzalez Prieto, A., Tripathy, A., Nilsen-Nygaard, E., Bierman, A., and B. Lengyel, "Subscribing to YANG datastore push updates", August 2017, <https://datatracker.ietf.org/doc/draft-ietf-netconf-yang-push/>.

Authors' Addresses

Alexander Clemm
Futurewei Technologies, Inc.
2330 Central Expressway
Santa Clara, CA 95050
USA

Email: ludwig@clemm.org

Eric Voit
Cisco Systems

Email: evoit@cisco.com

Xufeng Liu
Jabil

Email: Xufeng_Liu@jabil.com

Igor Bryskin
Huawei

Email: Igor.Bryskin@huawei.com

Tianran Zhou
Huawei

Email: zhoutianran@huawei.com

Guangying Zheng
Huawei

Email: zhengguangying@huawei.com

Henk Birkholz
Fraunhofer SIT

Email: henk.birkholz@sit.fraunhofer.de