Evolution of the Subscription & Event Notification Drafts
IETF #98 Chicago
Eric Voit
28-Mar-2017
DRAFT

Authors on at least 1 drafts
Andy Bierman
Alexander Clemm
Balazs Lengyel
Einar Nilsen-Nygaard
Alberto Gonzalez Prieto
Ambika Prasad Tripathy
Eric Voit

+ Contributors
Sharon Chisholm
Yan Gang
Peipei Guo
Susan Hares
Tim Jenkins
Michael Scharf
Hector Trevino
Kent Watsen
Guangying Zheng (Walker)

The now successfully retired Dezign™ Team
Data Replication Frequency vs. Latency

This is the new stuff enabled

- Softwre Version
- Crypto Keys
- Event Traps
- ACL
- Counters
- Topology
- Syslog
- Flow Records
- ASIC stats
- Routing & Switching Protocols
- Ethernet
- SONET
- ARP
- OSPF
- TrapS
- BGP

Required Notification Latency:
- 0.1 second
- 1 second
- 1 minute
- 1 hour
- 1 day
- 1 month

This diagram illustrates the relationship between data replication frequency and required notification latency. The new stuff enabled includes various network and security features such as software version control and cryptographic keys. The diagram categorizes these features into different frequency ranges, from 0.1 second to 1 day, and latency requirements from 0.1 second to 1 month.
## Functional Partitioning Summary

<table>
<thead>
<tr>
<th>Subscription</th>
<th>Types</th>
<th>YANG Datastore Push</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Dynamic and Configured</td>
<td>Dynamic and Configured</td>
</tr>
<tr>
<td></td>
<td>many</td>
<td>many</td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>establish, modify, delete, kill</td>
<td>establish, modify, delete, kill</td>
</tr>
<tr>
<td></td>
<td>started, suspended, resumed, terminated, modified</td>
<td>started, suspended, resumed, terminated, modified</td>
</tr>
<tr>
<td>Negotiation</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPCs</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Plane Notifications</td>
<td>basic</td>
<td>RFC-7950 + Subscription-ID</td>
</tr>
<tr>
<td>Headers</td>
<td></td>
<td>complete and changes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Update Bundling</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transport</td>
<td>NETCONF</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RESTConf, HTTP, HTTP2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Legend
- `draft-ietf-netconf-yang-push`
- `draft-ietf-netconf-subscribed-notifications`
- `draft-voit-netmod-yang-notifications2`
- `draft-ietf-netconf-event-netconf`
- `draft-ietf-netconf-event-restconf`
## Functionality per Draft

### Subscribed Notifications
- Dynamic & Configured subscriptions
- Multiple subscriptions / transport
- Multiple configured receivers
- Establish, modify, delete, kill RPC
- State change notifications
- Suspend/resume
- Filtering full notifications
- Stream discovery
- Replay (and start time negotiation)
- Prioritization
- Monitoring / reporting
- QoS
- Error responses

### YANG Datastore Push
- Datastore on-change and periodic triggers
- Filtering objects within a notification
- Authorization model per object
- Sending of full YANG trees or yang-patch
- Tagging of partial updates
- Tagging of on-change object support
- Negotiation of filters and period lengths
- More error responses

### YANG Notifications2
- Encapsulation Headers objects: Signature, de-duplication, severity, originator
- Bundled records and record types
- Negotiation of filters and period lengths
- More error responses

### NETCONF Transport for Subscribed Notifications
- Transport mapping
- Transport mappings (including HTTP2 QoS)
- Heartbeats and clean-up

### RESTCONF & HTTP2 Transport for Subscribed Notifications
- Transport mappings (including HTTP2 QoS)
- Heartbeats and clean-up
Drafts in Layered Framework

Application

Subscriber

Receiver

Subscription Mgmt

Dynamic Configuration

Subscription Mtc

Admission Control

OAM

Negotiation

Stream Discovery

Transport Session

Update Packaging and Flow Control

TLS

HTTP2

HTTP1.1

gRPC

Restconf

Netconf

Encoding

Thrift

CBOR

GPB

JSON

XML

YANG

Bundling

Filtering

Access Control

On-Change

Periodic

Event Notification Generation

Event Generation

Operational State Control Plane

Applied Config

Intended Config

Candidate Config

Startup Config

Running Config

NETCONF custom Stream

Legend

draft-ietf-netconf-yang-push

draft-ietf-netconf-subscribed-notifications

draft-voit-netmod-yang-notifications2

draft-ietf-netconf-event-netconf

draft-ietf-netconf-event-restconf

Updates since IETF #97

• -05 revision
• Ability to get operational data from filters
• Extension notifiable-on-change added
• New appendix on potential futures
• New error and hint mechanisms included in text and in the YANG model
• Updated examples based on the error definitions
• Text updates
Final steps before WG Last Call

– Subscription ID as an identifier only relevant to a single receiver
– Deferral of a standard header and bundle support (i.e., use the current data plane notifications.)
Updates since IETF #97

- Move away from 5277bis
- Kill subscription RPC added
- Error conditions added
- YANG model simplifications
- Renaming of Subscription State Notifications and identifiers

subscribed-notifications
Final steps before WG Last Call
(Same as for YANG-Push)

- Subscription ID as an identifier only relevant to a single receiver
- Deferral of a standard header and bundle support (i.e., use the current data plane notifications.)
Updates since IETF #97

- -02 revision
- Removed sections redundant with other drafts
- 3rd party subscriptions out of scope
- SSE only used with RESTCONF and HTTP1.1 Dynamic Subscriptions.
Final steps before WG Last Call

HTTP2 transport message compatibility with GRPC
• One set of meetings. Need another set of eyes
No Updates since IETF #97

-02 revision coming shortly
  (was awaiting 5277bis Charter solidification)
Proposes Solutions to the Following:

1. What are the set of transport agnostic header objects which might be usefully placed within YANG notifications?

2. How might a set of YANG notifications be bundled into a single transport message?

3. How do you query the originator of a notification to troubleshoot the bundling process?
#1 Transport Agnostic Header Objects

---

--- notification-message
  --- notification-message-header
    --- record-time
    --- record-type?
    --- record-id?
    --- record-severity?
    --- observation-domain-id?
    --- subscription-id?
    --- notification-time?
    --- notification-id?
    --- previous-notification-id?
    --- signature?
    --- message-generator-id?
  --- receiver-record-contents?
#2 bundling multiple notifications into a single transportable message

```
+----n bundled-notification-message
   +--ro notification-message-header
      |   +--ro notification-time
      |   +--ro notification-id?
      |   +--ro previous-notification-id?
      |   +--ro signature?
      |   +--ro message-generator-id?
      |   +--ro record-count?
   +--ro notification-records*
      +--ro notification-record-header
         |   +--ro record-time
         |   +--ro record-type?
         |   +--ro record-id?
         |   +--ro record-severity?
         |   +--ro observation-domain-id?
         |   +--ro subscription-id?
      +--ro receiver-record-contents?
```
Thank you!