



# Feedback on ODL netconf

making the ODL controller work  
with real netconf network devices and stubs

Guillaume Lambert 09/24/18

# NETCONF in a nutshell

- Netconf is a device configuration/management IETF protocol that uses
  - YANG as a modeling language
  - XML messages over SSH ( or TLS or TCP) to encode messages.
- Netconf defines a handshake or “hello” phase to announces the devices and the client capabilities. A “hello“ capability refers to a YANG model.
- Netconf defines 3 datastores where the device can store internally information and data. Mandatory datastores are “startup” and “running”. The “candidate” DS is not mandatory. Additional ones may be used. NETCONF does not specify how the datastore is organised.
- Netconf defines basics operations (or RPCs)
  - <get>
  - <get-config>
  - <edit-config>
  - <lock>, <unlock>
  - [etc]
- <get-schema > RPC is not defined in the base but in the netconf-monitoring extension

# ODL netconf mountpoints

- The ODL controller allows to access and manipulate Netconf devices configuration and datastores in REST with “mountpoints”
- When a netconf devices is mounted, ODL tries to download from it the schemas announced if they are not already in the karaf cache.
- ODL netconf SB plugin defines 2 datastores inside the ODL controller that are linked to those “mountpoints”.
  - The configuration datastore that contains all the device configuration read-write parameters
    - i.e. the return of a <get-config/> netconf RPC
  - The operational datastore that contains the whole device configuration (ro and rw parameters)
    - i.e.
      - the return of a <get/> netconf RPC

# netconf base

- All netconf devices must at least support the netconf base capability in the hello message

```
<capability>urn:ietf:params:xml:ns:netconf:base:1.0?module=ietf-netconf&revision=2011-06-01</capability>
```

It can also be announced this way in the hello message:

```
<capability>urn:ietf:params:xml:ns:yang:ietf-netconf?module=ietf-netconf&revision=2011-06-01</capability>
```

- Since it is mandatory, some devices
  - do not specify the revision nor the module name along the name space.

```
<capability>urn:ietf:params:xml:ns:netconf:base:1.0</capability>
```

or for netconf version 1.1

```
<capability>urn:ietf:params:xml:ns:netconf:base:1.1</capability>
```

- do not allow to retrieve the YANG model with a <get-schema>

Problem: the ODL controller looks for the presence of the ietf-netconf module and tries to retrieve the corresponding YANG schema if the monitoring extension is announced

=> the netconf capabilities must be overloaded when mounting the device in REST

=> the ietf-netconf yang model must be present in the karaf schemas cache

# netconf-state

- The netconf state RPC is given by the netconf monitoring extension. It lists all the schemas available for retrieval with a <get-schema > RPC. It specifies the module name and the revision for each schema.
- The module names and their revisions are also listed in the hello handshake. The ODL controller checks if the results are coherent.

## Problem1:

- ⇒ by default, some implementations returns an empty netconf-state (ex Honeynode) . This was OK with nitrogen but no more in Fluorine.  
<https://git.opendaylight.org/gerrit/#/c/75663/>

## Problem2:

- ⇒ some implementations returns a netconf state incomplete or incoherent with the hello handshake. This can happen with conf-D, especially when trying to support both netconf and SNMP. The controller displays logs errors or warning. The device is not mounted (more likely in Fluorine) or the mountpoint is only partially readable (more likely with nitrogen).

# solutions

- Fill the Karaf cache schemas with missing models that can not be downloaded
- When mounting the devices in REST, use the <yang-module-capabilities> attributes to add or overload those capabilities

<https://docs.opendaylight.org/en/stable-oxygen/user-guide/netconf-user-guide.html#connecting-to-a-device-not-supporting-netconf-monitoring>

- However, capabilities can not (sometimes?) be deleted at mount time. A proxy can do this.

<https://git.opendaylight.org/gerrit/#/c/63148/>

# openEXI vs EXIficient

- Netconf XML messages can be compressed by using the EXI format. Up to Oxygen, Opendaylight was using the openEXI library. Fluorine switched to EXIficient because openEXI seems no more maintained.
- EXIficient can decode openEXI generated messages but the reverse is not true

thanks to Jakub and Robert for the explanations

- <https://git.opendaylight.org/gerrit/#/c/75663/>
- <https://git.opendaylight.org/gerrit/#/c/74952/>

# netopeer

- a complete C implementation of a netconf server
- has been completely reworked between v1 and v2
- custom RPCs must be developed with the transAPI framework
- uses a config datastore and an operational datastore similarly to ODL mountpoints

We have identified issues in v1 when using « when » statement in YANG models on parameters in the 2 datastores

# testtool

- hosted by the ODL netconf project
- netconf testtool is intended for scalability tests
- out of the box, it comes only with a limited set of features
  - only supports a config file for <get-config> parameters
  - can be patched quickly but rw is no more supported
- pre-Fluorine versions uses openEXI...  
=> regression tests ?

# honeynode / honyecomb

- a quite complete implementation with many config files, can be used as a broker
  - does not have the testtool limitations with the operational datastore
- a FD.io project based on ODL
  - uses the Apache License – EPL generates errors at build time: we have to relicense some files
- does not use EXificient yet
  - we have to patch the netty folder to make it work with Fluorine

<https://git.opendaylight.org/gerrit/#/c/75663/>

# tail-f confD

- a proprietary implementation popular among vendors
  - a limited version free of charge can be downloaded
- Many versions are available
- introduces many tail-f specific YANG models
  - some of them may not be retrieved with a `<get-schema>`
  - they often change when a new version of confD is released
- in case of misconfiguration the netconf-state may be inconsistent



Thank you