Process to establish a NetConf session

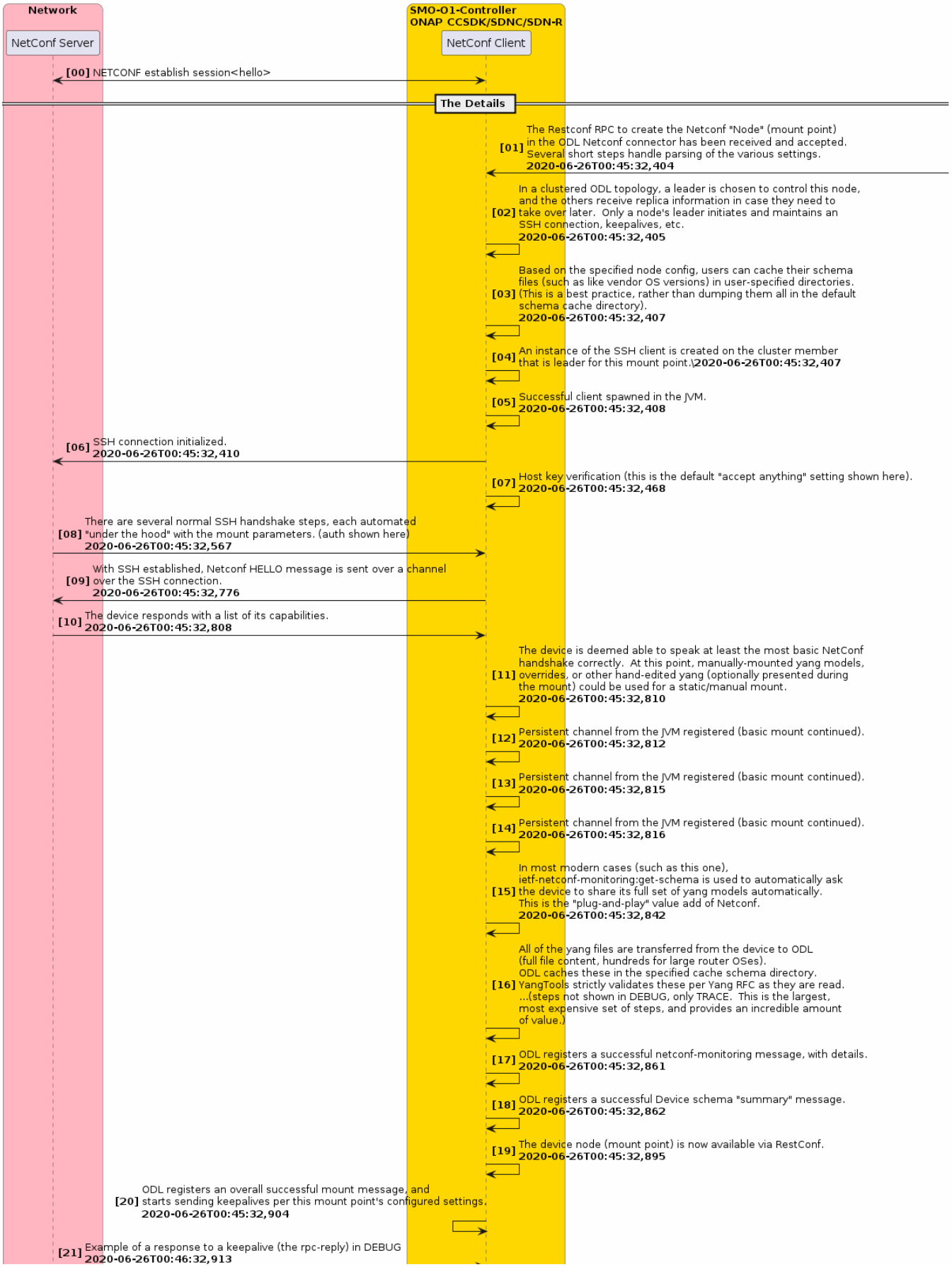
This wiki shows the different steps to establish a secure NetConf session between ONAP OpendayLight /CCSDK/SDNC/SDN-R and a NetConf Server.

In this realistic example the entire procedure takes about 500ms. The benefit is a secure permanent session for fast interfacing.

Diagram

License
Apache 2.0

ONAP/O-RAN Establish a NetConf session



Log

With special thanks to Jeff Hartleyof the Logs and the comments.

```
#### 1. The Restconf RPC to create the Netconf "Node" (mount point) in the ODL Netconf connector has been
received and accepted.  Several short steps handle parsing of the various settings.
2020-06-26T00:45:32,404 | DEBUG | opendaylight-cluster-data-notification-dispatcher-77 |
NetconfTopologyImpl              | 79 - netconf-topology-config - 1.7.2 | Config for node Uri{_value=VR30}
created


#### 2.  In a clustered ODL topology, a leader is chosen to control this node, and the others receive
replica information in case they need to take over later.  Only a node's leader initiates and maintains an
SSH connection, keepalives, etc.
2020-06-26T00:45:32,405 | INFO  | opendaylight-cluster-data-notification-dispatcher-77 |
AbstractNetconfTopology          | 79 - netconf-topology-config - 1.7.2 | Connecting RemoteDevice{Uri
{_value=VR30}} , with config Node{getNodeId=Uri{_value=VR30}, augmentations={interface org.opendaylight.yang.
gen.v1.urn.opendaylight.netconf.node.topology.rev150114.NetconfNode=NetconfNode
{getActorResponseWaitTime=300, getBetweenAttemptsTimeoutMillis=2000, getConcurrentRpcLimit=0,
getConnectionTimeoutMillis=20000, getCredentials=***, getDefaultRequestTimeoutMillis=60000, getHost=Host
{_ipAddress=IpAddress{_ipv4Address=Ipv4Address{_value=10.18.165.145}}}, getKeepaliveDelay=60,
getMaxConnectionAttempts=0, getPort=PortNumber{_value=830}, getSchemaCacheDirectory=VR30, getSleepFactor=1.
1, isReconnectOnChangedSchema=false, isSchemaless=false, isTcpOnly=false}}}


#### 3. Based on the specified node config, users can cache their schema files (such as like vendor OS
versions) in user-specified directories.  (This is a best practice, rather than dumping them all in the
default schema cache directory).
2020-06-26T00:45:32,407 | INFO  | opendaylight-cluster-data-notification-dispatcher-77 |
AbstractNetconfTopology          | 79 - netconf-topology-config - 1.7.2 | Netconf connector for device VR30
will use schema cache directory VR30 instead of schema


#### 4. An instance of the SSH client is created on the cluster member that is leader for this mount point.
2020-06-26T00:45:32,407 | DEBUG | opendaylight-cluster-data-notification-dispatcher-77 |
NetconfClientDispatcherImpl      | 359 - org.opendaylight.netconf.client - 1.7.2 | Creating reconnecting SSH
client with configuration: NetconfReconnectingClientConfiguration{address=/10.18.165.145:830,
connectionTimeoutMillis=20000, additionalHeader=null, sessionListener=org.opendaylight.netconf.sal.connect.
netconf.listener.NetconfDeviceCommunicator@4a50c7d5, reconnectStrategy=org.opendaylight.netconf.nettyutil.
TimedReconnectStrategy@449730ef, clientProtocol=SSH, authHandler=org.opendaylight.netconf.nettyutil.handler.
ssh.authentication.LoginPasswordHandler@40a8aca5, sslHandlerFactory=null, connectStrategyFactory=org.
opendaylight.netconf.nettyutil.TimedReconnectStrategyFactory@5aa3ad4}


#### 5. Successful client spawned in the JVM.
2020-06-26T00:45:32,408 | DEBUG | opendaylight-cluster-data-notification-dispatcher-77 |
AbstractNetconfDispatcher        | 371 - org.opendaylight.netconf.netty-util - 1.7.2 | Client created.


#### 6. SSH connection initialized.
2020-06-26T00:45:32,410 | DEBUG | nioEventLoopGroupCloseable-3-2 | AsyncSshHandler                  | 371 -
org.opendaylight.netconf.netty-util - 1.7.2 | Starting SSH to /10.18.165.145:830 on channel: [id: 0xd91cad1f]


#### 7. Host key verification (this is the default "accept anything" setting shown here).
2020-06-26T00:45:32,468 | WARN  | sshd-SshClient[344ec5e9]-nio2-thread-6 | AcceptAllServerKeyVerifier
| 140 - org.apache.sshd.osgi - 2.3.0 | Server at /10.18.165.145:830 presented unverified EC key: SHA256:
/i7ZAPqEQNfRyEqynuzhBdTkrEn2jee4y5gkGJZ0088


#### 8. There are several normal SSH handshake steps, each automated "under the hood" with the mount
parameters. (auth shown here)
2020-06-26T00:45:32,567 | DEBUG | sshd-SshClient[344ec5e9]-nio2-thread-4 | AsyncSshHandler
| 371 - org.opendaylight.netconf.netty-util - 1.7.2 | SSH session authenticated on channel: [id:
```

```
0xd91cad1f], server version: SSH-2.0-OpenSSH_7.4p1 Debian-10+deb9u6


#### 9. With SSH established, Netconf HELLO message is sent over a channel over the SSH connection.
2020-06-26T00:45:32,776 | DEBUG | nioEventLoopGroupCloseable-3-2 | AbstractNetconfSessionNegotiator | 371 -
org.opendaylight.netconf.netty-util - 1.7.2 | Session negotiation started with hello message <hello xmlns="
urn:ietf:params:xml:ns:netconf:base:1.0">
<capabilities>
<capability>urn:ietf:params:netconf:capability:exi:1.0</capability>
<capability>urn:ietf:params:netconf:base:1.1</capability>
<capability>urn:ietf:params:netconf:base:1.0</capability>
</capabilities>
</hello>
 on channel [id: 0xd91cad1f]


#### 10.  The device responds with a list of its capabilities.
2020-06-26T00:45:32,808 | DEBUG | nioEventLoopGroupCloseable-3-2 | NetconfXMLToHelloMessageDecoder  | 371 -
org.opendaylight.netconf.netty-util - 1.7.2 | Parsing message
<?xml version="1.0" encoding="UTF-8"?>
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>urn:ietf:params:netconf:base:1.0</capability>
    <capability>urn:ietf:params:netconf:base:1.1</capability>
    <capability>urn:ietf:params:netconf:capability:with-defaults:1.0?basic-mode=explicit&also-
supported=report-all,explicit</capability>
    <capability>urn:ietf:params:netconf:capability:candidate:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:startup:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:rollback-on-error:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:validate:1.1</capability>
    <capability>urn:ietf:params:netconf:capability:validate:1.0</capability>
    <capability>urn:vyatta.com:mgmt:vyatta-op-show:1?module=vyatta-op-show-v1&revision=2017-05-12<
/capability>
...trimmed out  ~200 capability announcements here...
    <capability>urn:vyatta.com:mgmt:vyatta-interfaces-unnumbered:1?module=vyatta-interfaces-dataplane-
unnumbered-v1&revision=2019-05-02</capability>
    <capability>urn:vyatta.com:mgmt:vyatta-op-update:1?module=vyatta-op-update-v1&revision=2018-08-03<
/capability>
    <capability>urn:vyatta.com:mgmt:vyatta-op-show-raid:1?module=vyatta-op-show-raid-v1&revision=2019-02-15<
/capability>
    <capability>urn:vyatta.com:mgmt:vyatta-op-rename-system-image:1?module=vyatta-op-rename-system-image-
v1&revision=2019-02-15</capability>
    <capability>urn:vyatta.com:mgmt:vyatta-op-protocols-frr-bgp-routing-instance:1?module=vyatta-op-
protocols-frr-bgp-routing-instance-v1&revision=2018-09-27</capability>
    <capability>urn:vyatta.com:mgmt:vyatta-op-clone-system:1?module=vyatta-op-clone-system-v1&revision=2019-
02-15</capability>
    <capability>urn:vyatta.com:mgmt:vyatta-protocols-frr-ldp:1?module=vyatta-protocols-frr-ldp-
v1&revision=2018-11-27</capability>
    <capability>urn:vyatta.com:mgmt:vyatta-op-ippf:1?module=vyatta-op-ippf-v1&revision=2019-06-20<
/capability>
    <capability>urn:vyatta.com:mgmt:vyatta-op-show-interfaces-pppoe:1?module=vyatta-op-show-interfaces-pppoe-
v1&revision=2018-08-02</capability>
    <capability>urn:vyatta.com:mgmt:vyatta-system-login-tally:1?module=vyatta-system-login-tally-
v1&revision=2018-08-28</capability>
  </capabilities>
  <session-id>2</session-id>
</hello>



#### 11. The device is deemed able to speak at least the most basic Netconf handshake correctly.  At this
point, manually-mounted yang models, overrides, or other hand-edited yang (optionally presented during the
mount) could be used for a static/manual mount.
2020-06-26T00:45:32,810 | DEBUG | nioEventLoopGroupCloseable-3-2 | AbstractNetconfSessionNegotiator | 371 -
org.opendaylight.netconf.netty-util - 1.7.2 | Changing state from : OPEN_WAIT to : ESTABLISHED for channel:
[id: 0xd91cad1f]


#### 12.  Persistent channel from the JVM registered (basic mount continued).
2020-06-26T00:45:32,812 | DEBUG | nioEventLoopGroupCloseable-3-2 | AbstractNetconfSession           | 371 -
org.opendaylight.netconf.netty-util - 1.7.2 | Session NetconfClientSession{sessionId=2, channel=[id:
```

```
0xd91cad1f]} up


#### 13. Persistent channel from the JVM registered (basic mount continued).
2020-06-26T00:45:32,815 | DEBUG | nioEventLoopGroupCloseable-3-2 | NetconfDevice                    | 377 -
org.opendaylight.netconf.sal-netconf-connector - 1.10.2 | RemoteDevice{VR30}: Session to remote device
established with NetconfSessionPreferences{capabilities={urn:ietf:params:netconf:capability:startup:1.0
=DeviceAdvertised, urn:ietf:params:netconf:capability:validate:1.1=DeviceAdvertised, urn:ietf:params:netconf:
capability:validate:1.0=DeviceAdvertised, urn:ietf:params:netconf:capability:candidate:1.0=DeviceAdvertised,
urn:ietf:params:netconf:capability:with-defaults:1.0?basic-mode=explicit&also-supported=report-all,
explicit=DeviceAdvertised, urn:ietf:params:netconf:capability:rollback-on-error:1.0=DeviceAdvertised, urn:
ietf:params:netconf:base:1.1=DeviceAdvertised, urn:ietf:params:netconf:base:1.0=DeviceAdvertised},
moduleBasedCapabilities={(urn:vyatta.com:mgmt:vyatta-opd-extensions:1?revision=2019-06-03)vyatta-opd-
extensions-v1=DeviceAdvertised, (urn:vyatta.com:mgmt:vyatta-interfaces-vfp-unnumbered:1?revision=2017-11-03)
vyatta-interfaces-vfp-unnumbered-v1=DeviceAdvertised, (urn:vyatta.com:mgmt:vyatta-op-delete-system-image:1?
revision=2019-02-15)vyatta-op-delete-system-image-v1=DeviceAdvertised, (urn:vyatta.com:mgmt:vyatta-service-
dhcp-server-routing-instance:1?revision=2017-12-15)vyatta-service-dhcp-server-routing-instance-
v1=DeviceAdvertised, (urn:vyatta.com:mgmt:vyatta-op:1?revision=2018-06-14)vyatta-op-v1=DeviceAdvertised,
(urn:vyatta.com:mgmt:vyatta-system-hardware-cpu-history:1?revision=2018-05-31)vyatta-system-hardware-cpu-
history-v1=DeviceAdvertised,
  ####...trimmed out ~200 namespace announcements here...####

 (urn:vyatta.com:mgmt:vyatta-op-clear-bridge:1?revision=2019-03-12)vyatta-op-clear-bridge-
v1=DeviceAdvertised, (urn:vyatta.com:mgmt:vyatta-service-dns-routing-instance:1?revision=2018-07-26)vyatta-
service-dns-routing-instance-v1=DeviceAdvertised, (urn:vyatta.com:mgmt:vyatta-op-protocols-frr-bgp-routing-
instance:1?revision=2018-09-27)vyatta-op-protocols-frr-bgp-routing-instance-v1=DeviceAdvertised, (urn:vyatta.
com:mgmt:vyatta-op-show-interfaces-dataplane:1?revision=2019-06-11)vyatta-op-show-interfaces-dataplane-
v1=DeviceAdvertised, (urn:vyatta.com:mgmt:vyatta-system-acm-opd:1?revision=2018-07-02)vyatta-system-acm-opd-
v1=DeviceAdvertised, (urn:vyatta.com:mgmt:vyatta-op-service-dns-routing-instance:1?revision=2018-08-03)
vyatta-op-service-dns-routing-instance-v1=DeviceAdvertised, (urn:vyatta.com:mgmt:vyatta-system-session-
routing-instance:1?revision=2018-09-12)vyatta-system-session-routing-instance-v1=DeviceAdvertised, (urn:
vyatta.com:mgmt:vyatta-service-dhcp-server:1?revision=2017-12-15)vyatta-service-dhcp-server-
v1=DeviceAdvertised, (urn:vyatta.com:mgmt:vyatta-op-clone-system-image:1?revision=2019-02-15)vyatta-op-clone-
system-image-v1=DeviceAdvertised, (urn:vyatta.com:mgmt:vyatta-protocols-frr-ospfv3:1?revision=2018-09-20)
vyatta-protocols-frr-ospfv3-v1=DeviceAdvertised, (urn:vyatta.com:mgmt:vyatta-service-dhcp-relay:1?
revision=2016-06-01)vyatta-service-dhcp-relay-v1=DeviceAdvertised, (urn:vyatta.com:mgmt:vyatta-policy-action:
1?revision=2018-09-13)vyatta-policy-action-v1=DeviceAdvertised}, rollback=true, monitoring=true,
candidate=true, writableRunning=false}


#### 14. Persistent channel from the JVM registered (basic mount continued).
2020-06-26T00:45:32,816 | DEBUG | nioEventLoopGroupCloseable-3-2 | AbstractNetconfTopology          | 79 -
netconf-topology-config - 1.7.2 | Connector for VR30 started succesfully


#### 15.  In most modern cases (such as this one), ietf-netconf-monitoring:get-schema is used to
automatically ask the device to share its full set of yang models automatically.  This is the "plug-and-
play" value add of Netconf.

2020-06-26T00:45:32,842 | DEBUG | nioEventLoopGroupCloseable-3-2 | NetconfDeviceCommunicator        | 377 -
org.opendaylight.netconf.sal-netconf-connector - 1.10.2 | RemoteDevice{VR30}: Message received <rpc-reply
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="m-0">
  <data>
    <netconf-state xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring">
      <schemas>
        <schema>
          <identifier>vyatta-system-alg-routing-instance-v1</identifier>
          <version>2018-09-12</version>
          <format>yang</format>
          <namespace>urn:vyatta.com:mgmt:vyatta-system-alg-routing-instance:1</namespace>
          <location>NETCONF</location>
        </schema>
        <schema>
          <identifier>vyatta-protocols-frr-ospfv3-v1</identifier>
          <version>2018-09-20</version>
          <format>yang</format>
          <namespace>urn:vyatta.com:mgmt:vyatta-protocols-frr-ospfv3:1</namespace>
          <location>NETCONF</location>
        </schema>
        <schema>
          <identifier>vyatta-service-dhcp-server-v1</identifier>
```

```
          <version>2017-12-15</version>
          <format>yang</format>
          <namespace>urn:vyatta.com:mgmt:vyatta-service-dhcp-server:1</namespace>
          <location>NETCONF</location>
        </schema>
        <schema>
          <identifier>vyatta-interfaces-bridge-state-v1</identifier>
          <version>2019-08-13</version>
          <format>yang</format>
          <namespace>urn:vyatta.com:mgmt:vyatta-interfaces-bridge-state:1</namespace>
          <location>NETCONF</location>
        </schema>
        <schema>
          <identifier>vyatta-interfaces-tunnel-transport-routing-instance-v1</identifier>
          <version>2017-12-05</version>
          <format>yang</format>
          <namespace>urn:vyatta.com:mgmt:vyatta-interfaces-tunnel-transport-routing-instance:1</namespace>
          <location>NETCONF</location>
        </schema>
        <schema>
          <identifier>vyatta-op-show-log-auth-v1</identifier>
          <version>2019-06-06</version>
          <format>yang</format>
          <namespace>urn:vyatta.com:mgmt:vyatta-op-show-log-auth:1</namespace>
          <location>NETCONF</location>
        </schema>
  ####...snipped out ~200 schamas announcements here....####
        <schema>
          <identifier>vyatta-op-show-dataplane-v1</identifier>
          <version>2019-05-20</version>
          <format>yang</format>
          <namespace>urn:vyatta.com:mgmt:vyatta-op-show-dataplane:1</namespace>
          <location>NETCONF</location>
        </schema>
        <schema>
          <identifier>vyatta-op-common-protocols-ospf-v1</identifier>
          <version>2018-11-02</version>
          <format>yang</format>
          <namespace>urn:vyatta.com:mgmt:vyatta-op-common-protocols-ospf:1</namespace>
          <location>NETCONF</location>
        </schema>
        <schema>
          <identifier>vyatta-security-vpn-ipsec-vfp-v1</identifier>
          <version>2018-01-11</version>
          <format>yang</format>
          <namespace>urn:vyatta.com:mgmt:vyatta-security-vpn-ipsec-vfp:1</namespace>
          <location>NETCONF</location>
        </schema>
      </schemas>
    </netconf-state>
  </data>
</rpc-reply>


#### 16a. All of the yang files are transferred from the device to ODL (full file content, hundreds for
large router OSes).

#### 16b. ODL caches these in the specified cache schema directory.

#### 16c. YangTools strictly validates these per Yang RFC as they are read.
  ...(steps not shown in DEBUG, only TRACE.  This is the largest, most expensive set of steps, and provides
an incredible amount of value.)


#### 17. ODL registers a successful netconf-monitoring message, with details.
2020-06-26T00:45:32,861 | DEBUG | remote-connector-processing-executor-1 | NetconfDevice
| 377 - org.opendaylight.netconf.sal-netconf-connector - 1.10.2 | RemoteDevice{VR30}: Schemas exposed by
ietf-netconf-monitoring: [(urn:vyatta.com:mgmt:vyatta-interfaces-vfp-unnumbered:1?revision=2017-11-03)vyatta-
interfaces-vfp-unnumbered-v1, (urn:vyatta.com:mgmt:vyatta-op-delete-system-image:1?revision=2019-02-15)
vyatta-op-delete-system-image-v1, (urn:vyatta.com:mgmt:vyatta-opd-extensions:1?revision=2019-06-03)vyatta-
opd-extensions-v1, (urn:vyatta.com:mgmt:vyatta-service-dhcp-server-routing-instance:1?revision=2017-12-15)
```

```
vyatta-service-dhcp-server-routing-instance-v1, (urn:vyatta.com:mgmt:vyatta-op:1?revision=2018-06-14)vyatta-
op-v1, (urn:vyatta.com:mgmt:vyatta-system-hardware-cpu-history:1?revision=2018-05-31)vyatta-system-hardware-
cpu-history-v1, (urn:vyatta.com:mgmt:vyatta-op-clear-interfaces-switch:1?revision=2018-07-17)vyatta-op-clear-
interfaces-switch-v1, (urn:vyatta.com:mgmt:vyatta-service-ssh-routing-instance:1?revision=2016-04-13)vyatta-
service-ssh-routing-instance-v1, (urn:vyatta.com:mgmt:vyatta-service-portmonitor:1?revision=2018-10-15)
vyatta-service-portmonitor-v1, (urn:vyatta.com:mgmt:vyatta-service-ssh:1?revision=2018-10-08)vyatta-service-
ssh-v1, (urn:vyatta.com:mgmt:vyatta-op-ping:1?revision=2019-03-25)vyatta-op-ping-v1, (urn:vyatta.com:mgmt:
vyatta-op-config:1?revision=2019-04-23)vyatta-op-config-v1, (urn:net.vyatta.eng:service:vyatta-op-qa-notify:
1?revision=2018-01-04)vyatta-op-qa-notify-v1, (urn:ietf:params:xml:ns:yang:ietf-inet-types?revision=2013-07-
15)ietf-inet-types, (urn:vyatta.com:mgmt:vyatta-op-common-protocols-bgp-routing-instance:1?revision=2018-11-
02)vyatta-op-common-protocols-bgp-routing-instance-v1, (urn:vyatta.com:mgmt:vyatta-protocols-frr-switch-vif-
ospf:1?revision=2018-11-09)vyatta-protocols-frr-switch-vif-ospf-v1, (urn:vyatta.com:mgmt:vyatta-service-
dhcpv6-relay-routing-instance:1?revision=2016-07-12)vyatta-service-dhcpv6-relay-routing-instance-v1, (urn:
vyatta.com:mgmt:vyatta-op-show-bmc:1?revision=2018-09-20)vyatta-op-show-bmc-v1,
  ####...trimmed out about ~200 netconf-monitoring get-schema internal-announcements here...####
(urn:vyatta.com:mgmt:vyatta-system-acm-opd:1?revision=2018-07-02)vyatta-system-acm-opd-v1, (urn:vyatta.com:
mgmt:vyatta-op-show-interfaces-dataplane:1?revision=2019-06-11)vyatta-op-show-interfaces-dataplane-v1, (urn:
vyatta.com:mgmt:vyatta-op-service-dns-routing-instance:1?revision=2018-08-03)vyatta-op-service-dns-routing-
instance-v1, (urn:vyatta.com:mgmt:vyatta-system-session-routing-instance:1?revision=2018-09-12)vyatta-system-
session-routing-instance-v1, (urn:vyatta.com:mgmt:vyatta-service-dhcp-server:1?revision=2017-12-15)vyatta-
service-dhcp-server-v1, (urn:vyatta.com:mgmt:vyatta-op-clone-system-image:1?revision=2019-02-15)vyatta-op-
clone-system-image-v1, (urn:vyatta.com:mgmt:vyatta-protocols-frr-ospfv3:1?revision=2018-09-20)vyatta-
protocols-frr-ospfv3-v1, (urn:vyatta.com:mgmt:vyatta-service-dhcp-relay:1?revision=2016-06-01)vyatta-service-
dhcp-relay-v1, (urn:vyatta.com:mgmt:vyatta-policy-action:1?revision=2018-09-13)vyatta-policy-action-v1]


#### 18. ODL registers a successful Device schema "summary" message.
2020-06-26T00:45:32,862 | DEBUG | remote-connector-processing-executor-1 | NetconfDevice
| 377 - org.opendaylight.netconf.sal-netconf-connector - 1.10.2 | RemoteDevice{VR30}: Resolved device
sources to org.opendaylight.netconf.sal.connect.netconf.DeviceSources@271cc937


#### 19. The device node (mount point) is now available via RestConf.
2020-06-26T00:45:32,895 | DEBUG | remote-connector-processing-executor-1 | NetconfDeviceSalProvider
| 377 - org.opendaylight.netconf.sal-netconf-connector - 1.10.2 | RemoteDevice{VR30}: TOPOLOGY Mountpoint
exposed into MD-SAL {closed=0, instance=org.opendaylight.mdsal.dom.spi.SimpleDOMMountPoint@29db23a4}


#### 20. ODL registers an overall successful mount message, and starts sending keepalives per this mount
point's configured settings.
2020-06-26T00:45:32,904 | DEBUG | remote-connector-processing-executor-1 | KeepaliveSalFacade
| 377 - org.opendaylight.netconf.sal-netconf-connector - 1.10.2 | RemoteDevice{VR30}: Netconf session
initiated, starting keepalives


#### 21.  Example of a response to a keepalive (the rpc-reply) in DEBUG:
2020-06-26T00:46:32,913 | DEBUG | nioEventLoopGroupCloseable-3-2 | AbstractNetconfSession           | 371 -
org.opendaylight.netconf.netty-util - 1.7.2 | handling incoming message
2020-06-26T00:46:32,913 | DEBUG | nioEventLoopGroupCloseable-3-2 | NetconfDeviceCommunicator        | 377 -
org.opendaylight.netconf.sal-netconf-connector - 1.10.2 | RemoteDevice{VR30}: Message received <rpc-reply
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="m-0">
  <data/>
</rpc-reply>
```