NX-OS NETCONF Support

Finnbarr P. Murphy

(fpm@fpmurphy.com)

NETCONF (Network Configuration Protocol) is a network management protocol developed and standardized by the IETF Netconf Working Group. It was first published in December 2006 as RFC 4741 with a revised version published in June 2011 as RFC 6241. It provides mechanisms to install, manipulate, and delete the configuration of network devices.

The same IETF Working Group also produced supporting RFCs for various transport mappings, including:

- RFC 4742 – Using the NETCONF Configuration Protocol over Secure SHell (SSH). Obsoleted by RFC 6242 (2011) which introduced a new framing mechanism to address some potential security issues with the initial design
- RFC 4743 – Using NETCONF over the SOAP (Simple Object Access Protocol)
- RFC 5539 – NETCONF over Transport Layer Security (TLS)
- RFC 5277 – NETCONF Event Notifications. Describes an asynchronous notification mechanism allowing clients to subscribe to named event streams
- RFC 6243 – With-defaults Capability for NETCONF. Describes an extension to the NETCONF protocol that allows clients to identify how defaults are processed by the server

NETCONF is an RPC-based protocol which uses XML (Extensible Markup Language) to encode data as well as the protocol messages. The protocol messages are exchanged on top of a secure transport protocol such as SSL or TLS.

The NETCONF protocol can be conceptually partitioned into four layers:

- The Content layer consists of configuration data and notification data.
- The Operations layer defines a set of base protocol operations to retrieve and edit the configuration data.
- The Messages layer provides a mechanism for encoding remote procedure calls and notifications
- The Secure Transport layer provides a secure and reliable transport of messages between a client and a server.

The NETCONF protocol was designed to address the shortcomings of existing practices and protocols for configuration management. According to Tail-F, a Network Service Orchestration company:

> CLI scripting was the primary approach to making automated configuration changes to the network prior to NETCONF. CLI scripting has several limitations including lack of transaction management, no structured error management and ever changing structure and syntax of commands that makes scripts fragile and costly to maintain. These are all side-effects of the basic fact that CLIs are designed to be used by humans and not an API for programmatic access.
>
> SNMP is another approach that could be used to write changes, but, in practice, is mostly used for performance and monitoring applications. Reasons for this include the lack of a defined discovery process that makes it hard to find the correct MIB modules, limitations inherent in the use of the UDP protocol, and the lack of useful standard security and commit mechanisms.

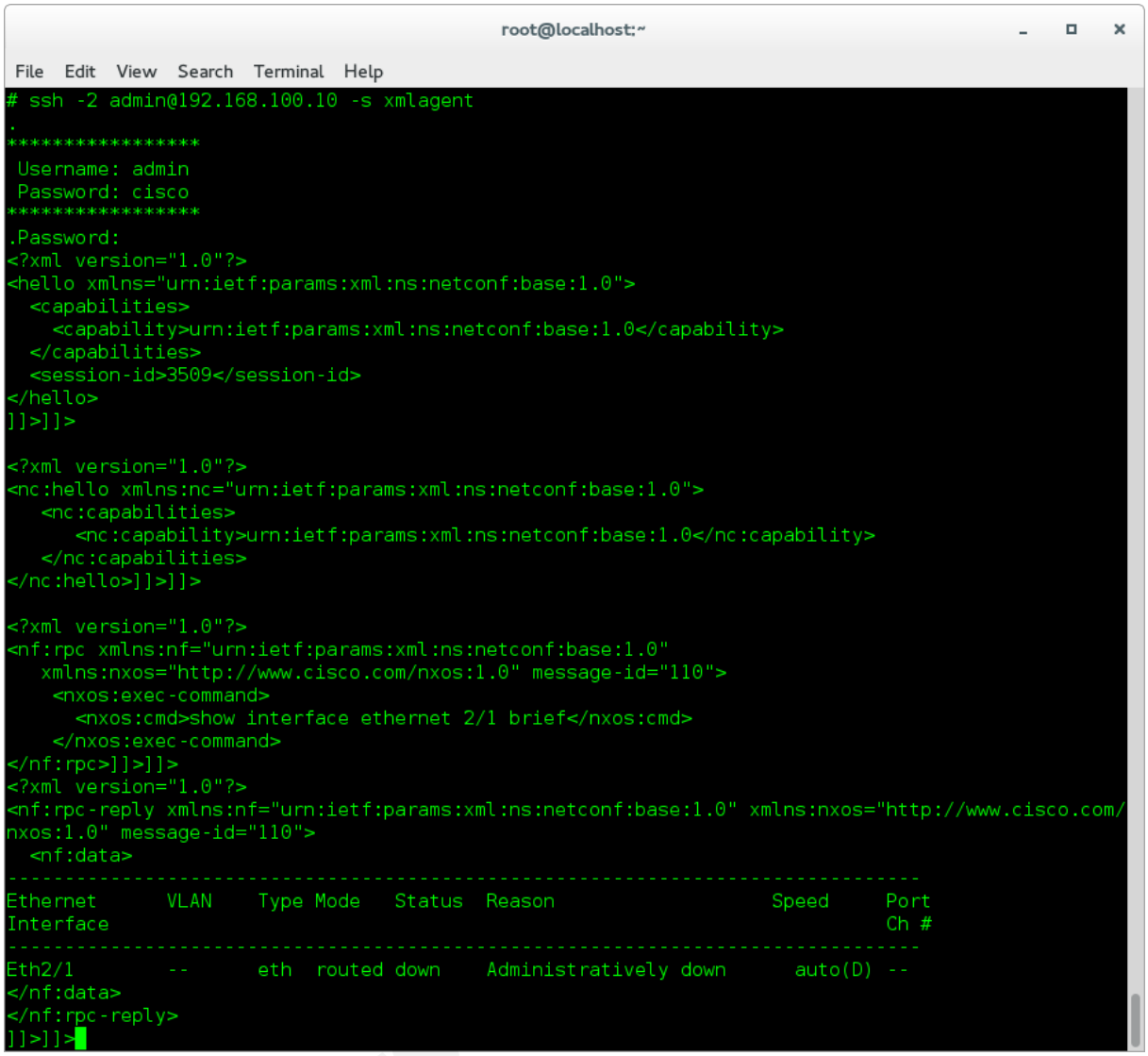
Closely tied to NETCONF is YANG which is a data modeling language used to model configuration and state data manipulated by the NETCONF protocol, NETCONF remote procedure calls, and NETCONF notifications. YANG was developed by the IETF NETCONF Data Modeling Language Working Group (NETMOD), and is defined in RFC 6020. YANG structures the data definitions into tree structures and provides many modeling features, including an extensible type system, formal separation of state and configuration data and a variety of syntactic and semantic constraints. extensibility for features of set rich a provide and modules in contained are snoitinfied data YANG and reuse.

The Cisco NX-OS implementation of NETCONF requires you to use a SSHv2 (Secure Shell Version 2) session for communication with the target device. See RFC 4742 for more information about using NETCONF over SSH . The target device provides an SSHv2 subsystem service called *xmlagent* that supports NETCONF over SSH Version 2.

SSHv2 subsystems are a layer of abstraction for defining and conveniently invoking remote commands on a server. For example, the default */etc/sshd/sshd_config* file for Fedora 20 defines a subsystem for the *sftpd* server:

```
# override default of no subsystems
Subsystem       sftp    /usr/libexec/openssh/sftp-server
```

As shown below, you invoke the *xmlagent* subsystem with the *-s xmlagent* command line argument. Note that some Cisco NETCONF documentation incorrectly states that the requisite subsystem is named *netconf*. Only SSH version 2 supports subsystems.



As you can see, all NETCONF messages are in the form of conforming XML documents with valid namespaces. You must end all the XML documents with *]]>]]>* to support synchronization in NETCONF over SSH. You can obtain the NX-OS XML Schema Definition at the Cisco download website.

The following are some examples of how to use NETCONF on a Nexus 7000. By the way, in case you are unaware of it, the operating system on a Nexus 7000 is NX-OS and not IOS. For more information about the NX-OS XML interfaces, see here.

Example 1 – Hello

The *hello* message (XML document) must be the first message sent by a NETCONF-aware client after the NETCONF server emits it's *hello* message.

To manually send an XML document to the XML server through an SSH session that you opened in a command shell, you can copy the XML text from an editor and paste it into the SSH session as I did above.

```
      urn:ietf:params:xml:ns:netconf:base:1.0
```



```
]]>]]>
```

Example 2 – Show Server Status

```
]]>]]>

        enabled

        8

         3940
         admin
         Sat May 10 15:08:31 2014
         2627
         1200
         1197
         192.168.100.1

]]>]]>
```

Example 3 – Show Version


```
]]>]]>
```

```
      Cisco Nexus Operating System (NX-OS) Software
TAC support: http://www.cisco.com/tac
Documents: http://www.cisco.com/en/US/products/ps9372/tsd_products_support_series_home.htm
l
Copyright (c) 2002-2010, Cisco Systems, Inc. All rights reserved.
The copyrights to certain works contained herein are owned by
other third parties and are used and distributed under license.
Some parts of this software are covered under the GNU Public
License. A copy of the license is available at
http://www.gnu.org/licenses/gpl.html.

      N/A
      5.1(2) [gdb]
      5.1(2) [gdb]
      bootflash:/titanium-d1-kickstart.5.1.2.gbin
       12/25/2020 12:00:00
      12/18/2010 09:53:29
      bootflash:/titanium-d1.5.1.2.gbin
       11/29/2010 12:00:00
      12/18/2010 11:47:03
      Unknown MDS Chassis
      Unknown Module
      Intel(R) Core(TM) i5-3450 CP
      2066644
      kB
      T0C29978A3B
      n7k
      0
      0
      1
      56
      5




]]>]]>
```


Example 4 – Show Running Configuration

```
]]>]]>




!Command: show running-config
!Time: Sat May 10 16:34:25 2014

version 5.1(2)
license grace-period

hostname n7k
vdc n7k id 1
  limit-resource vlan minimum 16 maximum 4094
  limit-resource vrf minimum 2 maximum 1000
  limit-resource port-channel minimum 0 maximum 768
  limit-resource u4route-mem minimum 96 maximum 96
  limit-resource u6route-mem minimum 24 maximum 24
  limit-resource m4route-mem minimum 58 maximum 58
  limit-resource m6route-mem minimum 8 maximum 8
feature ospf
feature hsrp
feature vtp

username adminbackup password 5 !  role network-operator
username admin password 5 $1$ek7G4QLQ$YqLUwOCmeIaQFjFRPaznZ0  role network-admin
username all password 5 !!  role network-operator

banner motd .
*****************
 Username: admin
 Password: cisco
*****************

ip domain-lookup
vlan dot1Q tag native
system default switchport
system jumbomtu 0
no logging event trunk-status enable
snmp-server user all  engineID 128:0:0:9:3:0:80:86:159:0:13
snmp-server user admin auth md5 0x1c4527bd2809f91d57d71d49e3871168 priv 0x1c4527bd2809f91d
57d71d49e3871168 localizedkey engineID 128:0:0:9:3:0:80:86:159:0:13
rmon event 1 log trap public description FATAL(1) owner PMON@FATAL
rmon event 2 log trap public description CRITICAL(2) owner PMON@CRITICAL
rmon event 3 log trap public description ERROR(3) owner PMON@ERROR
rmon event 4 log trap public description WARNING(4) owner PMON@WARNING
rmon event 5 log trap public description INFORMATION(5) owner PMON@INFO
snmp-server enable traps link

vrf context management
  ip route 0.0.0.0/0 192.168.1.1

interface Ethernet2/1
  shutdown
  no switchport

interface Ethernet2/2
  shutdown
  no switchport

interface Ethernet2/3
  shutdown
  no switchport

interface Ethernet2/4
```



```
  shutdown
  no switchport

interface Ethernet2/5
  shutdown
  no switchport

interface Ethernet2/6
  shutdown
  no switchport

interface Ethernet2/7
  shutdown
  no switchport

interface Ethernet2/8
  shutdown
  no switchport

interface Ethernet2/9
  shutdown
  no switchport

interface mgmt0
  vrf member management
  ip address 192.168.100.10/24
line console
line vty
boot kickstart bootflash:/titanium-d1-kickstart.5.1.2.gbin
boot system bootflash:/titanium-d1.5.1.2.gbin
ip route 0.0.0.0/0 192.168.100.1
no system default switchport shutdown



]]>]]>
```

Example 5 – Error Message

```
]]>]]>



    application
    invalid-value
    error
    Syntax error while parsing 'show murphy '

      show



]]>]]>
```


Example 6 – Retrieve SSH Keys

```
]]>]]>



  ************************************
rsa Keys generated:Sat May 10 13:53:00 2014

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAAAgQC8leN2v8FrXVwxCbgOt4l64Ch5Zty7rkc8DOa4twNzP04DdsP0YE
1UWRWOeFlY5WMI1KRR62ermxGVxVYUvl+RMZMBejRvTzvmAFpGjxihdfN6fvCLceRyPV4+G3lg1CIfH2bbl1CmN2kC
h+cTc+eFT9jcdwrvncR+ABwpggYHzQ==

bitcount:1024
fingerprint:
44:9a:02:62:11:60:b2:15:e9:2b:36:08:1e:1c:eb:a5
************************************
could not retrieve dsa key information
************************************


]]>]]>
```

Example 7 – Configure Interface

```
            2/1


                    192.168.100.11 255.255.255.0


                 Configured by NETCONF
```



```
]]>]]>



]]>]]>
```

Example 7 – Configure Interface Using Cisco NETCONF Extension

Cisco NX-OS supports an RPC operation named *exec-command*. The operation allows client applications to send CLI configuration and show commands and to receive responses to those commands as XML tags.

```
&lt?xml version="1.0"?>
&ltnf:rpc xmlns:nf="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:nxos="http://www.cisco.com/nxos:1.0" message-id="110">
&ltnxos:exec-command>
&ltnxos:cmd>configure terminal ;
        interface ethernet 2/1 ;
        ip address 192.168.100.15 255.255.255.0 ;
        no shutdown &lt/nxos:cmd>
&lt/nxos:exec-command>
&lt/nf:rpc>]]>]]>

&lt?xml version="1.0"?>
&ltnf:rpc-reply xmlns:nf="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns:nxos="http://www.
cisco.com/nxos:1.0" message-id="110">
  &ltnf:data/>
&lt/nf:rpc-reply>
]]>]]>
```

Example 7 – Show Interface Brief Using Cisco NETCONF Extension

```
     show interface brief

]]>]]>



-------------------------------------------------------------------------------
Port   VRF          Status IP Address                              Speed    MTU
-------------------------------------------------------------------------------
mgmt0  --           up     192.168.100.10                          --       1500

-------------------------------------------------------------------------------
Ethernet      VLAN    Type Mode   Status  Reason                   Speed     Port
Interface                                                                    Ch #
-------------------------------------------------------------------------------
Eth2/1        --      eth  routed up      none                       1000(D) --
```



```
Eth2/2        --      eth  routed down    Administratively down      auto(D) --
Eth2/3        --      eth  routed down    Administratively down      auto(D) --
Eth2/4        --      eth  routed down    Administratively down      auto(D) --
Eth2/5        --      eth  routed down    Administratively down      auto(D) --
Eth2/6        --      eth  routed down    Administratively down      auto(D) --
Eth2/7        --      eth  routed down    Administratively down      auto(D) --
Eth2/8        --      eth  routed down    Administratively down      auto(D) --
Eth2/9        --      eth  routed down    Administratively down      auto(D) --


]]>]]>
```

You may think that NETCONF is some exotic network device configuration protocol. That was true a number of years ago. However NETCONF is becoming quite popular and is supported by most network equipment vendors nowadays. Network Function Virtualization (NFV) is an emerging operational model where service providers want to virtualize entire classes of network node functions into building blocks that may be connected, or chained, together to create communication services. NFV is built around a very strong assumption of deep automation. This deep automation aligns directly with NETCONF with its standardized protocols and models, Another area where NETCONF is actively being used is with Software Defined Networking (SDN).

Enjoy!

