SWITCH

The Swiss Education & Research Network

What's New in Network Configuration?

Simon Leinen <simon@limmat.switch.ch>
Andy Bierman <ietf@andybierman.com>

NETCONF History

- 2002 IAB Network Management Workshop (see RFC 3535)
  - SNMP used for monitoring, but not configuration
  - SNMP MIBs lag (years) behind feature implementation
  - SNMP doesn’t distinguish config from non-config data
  - Operators use (proprietary) CLI for many tasks
  - In particular those involving configuration(s)
  - Problems with unstable and hard-to-parse CLI

IETF NETCONF WG

- Chartered May 2003
  - based on “XMLCONF” proposal and Juniper's XML-based “Junoscript”
- Charter: Standardize XML-based protocol for network configuration
- Web page: http://www.ops.ietf.org/netconf/
- Chairs: Andy Bierman, Simon Leinen

NETCONF WG Achievements

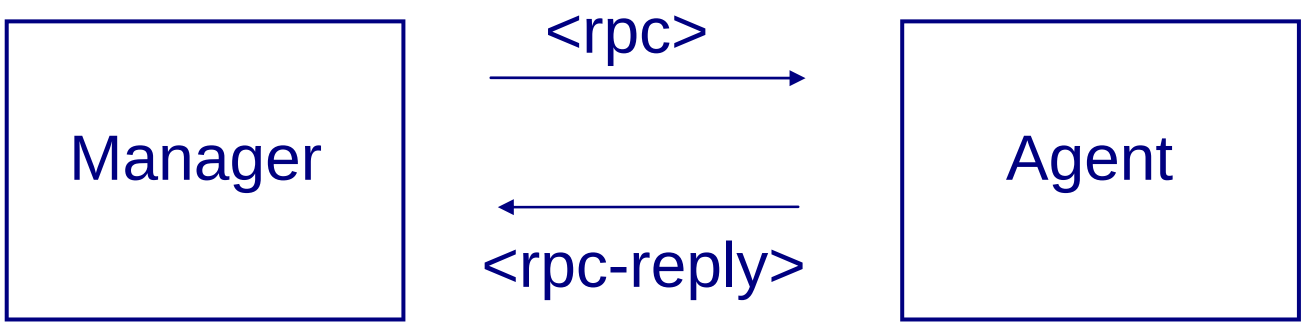
Four documents with the RFC Editor:

- NETCONF Configuration Protocol
- NETCONF over SSH (TCP port 830) – mandatory to implement
- NETCONF over BEEP (TCP port 831)
- NETCONF over SOAP (TCP ports 832/833 HTTP/BEEP)

Finally escaped from “IANA action required” state last night...

NETCONF Basics: RPC Model

- NETCONF uses its own RPC mechanism instead of XML-RPC or some other pre-existing RPC standard
  - Wanted the same RPC across all transports
  - Existing mechanisms did not provide the data types and error info that NETCONF needs
- Vendors can define their own RPC methods (using own namespaces)



RPC Methods

Standard RPC Message Format

-
  ```
  <rpc message-id="101“ xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
       <some-method>
          <!-- method parameters here... -->
       </some-method>
  </rpc>
  ```

Vendor RPC Message Format

-
  ```
  <rpc message-id="102“ xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <my-own-method xmlns="http://example.net/me/my-own/1.0">
          <my-first-parameter>14</my-first-parameter>
          <another-parameter>fred</another-parameter>
      </my-own-method>
  </rpc>
  ```

Example <hello> Message

```
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
   <capabilities>
      <capability>
         urn:ietf:params:xml:ns:netconf:base:1.0
      </capability>
      <capability>
        urn:ietf:params:xml:ns:netconf:capability:startup:1.0
      </capability>
      <capability>
        http:/example.net/router/2.3/myfeature
      </capability>
   </capabilities>
   <session-id>1043</session-id>
</hello>
```

Base Protocol Operations

Name	Description
get-config	Retrieve some or all of a configuration
edit-config	Edit some or all of a configuration
copy-config	Copy contents of one config to another
delete-config	Remove all contents of a config
lock	Start exclusive write access of a config
unlock	Stop exclusive write access of a config
get	Retrieve config and/or state data
close-session	Cause your session to close
kill-session	Force another session to close

Optional Protocol Operations

SWITCH
The Swiss Education & Research Network

Name	Capability	Description
commit	candidate	Commit <candidate> to <running> configuration
discard-changes	candidate	Clear the <candidate> configuration
validate	validate	Perform a syntax check and optionally a referential integrity check on the specified config

Configuration Locking

- NETCONF supports global locking
  - Exclusive access to an entire configuration datastore is granted to one user even if the user only has permission to alter some of the configuration data
- Partial locking is coming
  - This feature depends on how data is named (and other factors) and the WG could not agree in time so it was shelved
- Lock implementation in mandatory but lock use is optional
  - Provides some really interesting failure modes :-)

<edit-config>

- 4 edit modes (create, merge, replace, delete)

Parameter	Description
target	Configuration datastore to edit
default-operation	Default edit mode; Default (merge) Values (merge, replace, none)
test-option	Values (test-then-set, set); Default (set)
error-option	Values (stop-on-error, continue-on-error, rollback-on-error); Default (stop-on-error)
config	Portion of the configuration to edit

<edit-config> Example

```
<rpc message-id="101“ xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <edit-config>
    <target><running/></target>
    <config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
      <top xmlns="http://example.com/schema/1.2/config">
        <interface xc:operation="replace">
          <name>Ethernet0/0</name>
          <mtu>1500</mtu>
          <address>
            <name>1.2.3.4</name>
            <prefix-length>24</prefix-length>
          </address>
        </interface>
      </top>
    </config>
  </edit-config>
</rpc>
```

Configuration Data

- Data is divided into 2 categories
  - Configuration data is loosely defined as “the information needed by a device to achieve its desired running state” and it is saved across a device reboot
  - Status and statistical data is loosely defined as “everything else”
  - This is done to make “diff” operations easier
- A Configuration Datastore is a conceptual collection of all the configuration data needed for a particular network device
  - <candidate>, <running>, and <startup> are the standard configuration datastores

Standard Configuration Datastores

SWITCH
The Swiss Education & Research Network

- <candidate>
  - A scratchpad configuration used to collect edits to be applied all at once with a <commit> operation

  Support for this configuration is optional
- <running>

  The current operational configuration

  All devices must support this configuration
- <startup>

  The configuration to be used upon the next reload
  - Only devices that require the <running> configuration to be manually copied to non-volatile storage support this configuration

Named Configuration Datastores

- An optional feature allows a configuration datastore named by a Universal Resource Locater (URL) to be used in protocol operations
  - local: file://configs/my-config.xml
  - remote: https://config.example.com/device-X
- The ‘url’ capability identifies which protocols the agent will allow in the URL syntax
- Remote configuration editing is possible but not encouraged
- Remote to remote file copy is not allowed

Summary

- NETCONF provides a low-level programmatic interface to manipulate network device configurations
  - Designed by network element vendor engineers to work on all the major router platforms
- NETCONF is content-neutral and only requires data model content to be well-formed XML
  - High-level object or service oriented model-driven systems can be layered on top

Active NETCONF WG Work

- Notifications – to notify manager of asynchronous events
  - Useful (not just) for configuration
  - Two competing proposals merged at July 2006 interim
  - *Subscriptions* based on *event streams* (similar to syslog facilities) and *filters*

Possible Future Work

- **Standard Access Control Mechanism (depends on data model)**
- **Granular locking (depends on data model)**
- **Data Modeling**
  - Standard data model for device (maybe “just” “router”) configuration?
  - Start with common conventions for things we think we all understand (such as an IP address)?
  - Restrict ourselves to configuration, or try to grandfather e.g. SNMP MIBs...?
- Software image management

Possible Future O&M Work

Replace entire IETF Network Management “stack” with something based on NETCONF?

- Note that SNMP has been mapped to SSH (ISMS WG)
- Some vendors would prefer not to implement SNMP

Do operators here think this would be useful/worthwhile?