Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: November 09, 2013

T. Petch
Engineering Networks Ltd

May 08, 2013

Netconf surprises
draft-petch-netconf-surprises-00.txt

Abstract

This document identifies some aspects of Netconf that may come as a suprise to those familiar with the use of SNMP for device management.

Status of This Memo

Copyright Notice

1. Introduction

This document looks at aspects of Netconf from the view of an experienced SNMP user, identifying some behaviour that may come as a surprise.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Manager and agent are used to refer to the part of the management system that makes requests and changes, and that part in the device that is being managed respectively. This is not the accepted terminology of SNMPv3. Netconf uses client and server.

3. Configuration and State Data

Netconf is a protocol for configuration. It arose from an IAB workshop RFC3535 [RFC3535] which emphasised the need to process configuration data independent of state data, a distinction that SNMP never made. A laudable requirement, this has proved more difficult to turn into practice. Netconf RFC6241 [RFC6241] defines "configuration data: The set of writable data that is required to transform a system from its initial default state into its current state" so that any read-only object is excluded. To some extent, the property of read-only versus read-write has become the touchstone of configuration so that the statistical counters, clearly state data, are read-only in the data models currently under development.

In this, it follows SNMP but in doing so, it also adopts the SNMP practice that a single value of a counter object of type counter is meaningless. Rather a counter has to be read more than once and it is the difference between values over time that is significant. Also, counters, read-only, cannot be reset, to zero or any other value, and so will eventually reach a maximum value at which point they are defined to wrap, reset to zero automatically. Other management systems use a different approach, allowing counters to be reset by a management station so that they can be read and reset at a time convenient to the management system.

4. Persistence

SNMP developed the concept of persistent data and embodied it in a Textual Convention, StorageType [RFC2579] . This means that the persistence or otherwise of the data is integral with definition of the data model.

Netconf took a different approach, defining datastores, of which the one mandatory one is :running, although most devices will also have a :startup datastore. Netconf says nothing about the persistency of the datastores. :startup is used at boot time and so by implication is in a storage type that persists across shutdowns. Again, by implication, :running is ephemeral since "Operations that affect the running configuration will not be automatically copied to the startup configuration " [RFC6241]. So making data persistent with Netconf requires a successful protocol operation, with SNMP, it is integral with the data definition.

The impact of this could be far-reaching.

5. Common data

In some devices, Netconf will coexist with SNMP and both protocols may be used to retrieve data. While this may result in the same data, there is no formal requirement for this to happen. SNMP views data as forming a single virtual MIB in a tree structure with data at the leaves of the tree. Netconf defines a number of discrete datastores so that a given object may appear in multiple datastores and have different values in those datastores, as when an update has been performed to :running and not yet copied to :startup. Perforce, then, Netconf can give multiple, different answers when SNMP only gives one (short of implementing multiple MIBs in a device and accessing them via different addresses, communities or such-like).

This different approach may affect the implementation of when data is obtained. Most of the data supplied by SNMP or Netconf will be obtained from other parts of the device, hardware or software. Once obtained, the agent has the option of caching the data for a future request or of discarding it and obtaining it afresh as and when it is again requested. The Netconf concept of datastores may colour an implementation into favouring caching as opposed to retrieval, whereas the SNMP concept of a single virtual MIB may point in the other direction. This is speculative but what is certain is that at present, there is no requirement for the different agents in a device to behave consistently and to produce the same values; indeed, with a single MIB and multiple datastores, there will be times, hopefully understood by management system, when this is to be expected.

6. Data models

SNMP objects are defined in MIB modules, each of which has a unique name. Each module is rooted as part of a single tree, managed, at the top level, by ISO, with each branch and each leaf having an integer identifier; thus all IETF objects start with an object identifier of 1.3.6.1.

In SNMP, each object has a unique, case-significant, name. By convention, the name starts with a short, lower-case prefix, which identifies the module in which the object is defined, if for the IF-MIB (MIB modules have an upper-case name), snmp for the SNMP-FRAMEWORK-MIB. Nowadays, such prefixes would be registered with IANA to ensure uniqueness; then, it just happened.

Scalar objects need no further identification so that snmpEngineID identifies such an object within an agent's MIB. Objects in tables need one or more index values to identify the row of the table, as with ifType.29 where 29 is the relevant value of ifIndex.

Every MIB module makes reference to other MIB modules and can do so with a reference to module name and another name as with IMPORTS snmpTraps FROM SNMPv2-MIB

Netconf, strictly YANG, the data modelling language that Netconf uses, RFC6020 [RFC6020] is different. The objects are defined in modules but the structure is now flat so that a device will have many top-level modules.

Modules have names that are required to be unique, at least for IETF-defined modules, but this name is not used in the identification of objects. Rather, each module has a namespace associated with it, such as

urn:ietf:params:xml:ns:yang:ietf-interfaces

and names exist within that namespace, so a reference to a name must implicitly or explicitly state that namespace.

Namespaces are (mostly) referenced by an associated prefix, such as if, so a reference to an object might be if:ifname, but that prefix is not required to be unique and the use of that prefix in contexts outside the defining module, such as when one module augments another module or when a filter is defined in a Netconf get operation, is only a 'SHOULD' and not a 'MUST'; as it can only be, since the prefixes chosen in modules are not themselves required to be unique. Absence of a prefix means that the default namespace applies.

Modules can include submodules which in turn can include submodules, making the content from a submodule available to parent module. The included submodule is identified by name. It is currently unclear if, when A includes B which includes C, the definitions in C are available to A or only to B; doubtless, this will be resolved.

Modules can import the contents of another module which makes definitions from one module available inside another module or submodule. The imported module is identified by name and the import statement specifies the prefix to be used within the importing module (which may or may not be the same as that used in the imported module).

The subtle difference is that an import allocates a prefix because the imported definitions come from a different namespace, while an include is part of a namespace common to the including module.

So what? If the walkthrough of Netconf naming given above seems complicated, that is because it is and this is apparent when making reference to or interpreting a data object, whether viewing data gathered from a device, data on the wire, making a request for data or understanding what an object is from its model definition.

While SNMP encodes in binary on the wire so some form of analyser is highly desirable to decode the binary and to turn the numeric object identifier into an object name, by accesssing the MIB module, an object name is readily obtained and may be all that is needed. Equally, a command line request for an object need only specify the object name; command line tools can turn that into an encoded request to get or set data. Indices, when needed, to identify a row in a table, are mostly numeric. The uniqueness of names makes the interface simple.

With Netconf, a name is unique within a namespace, so a reference will be of the form prefix:objectname, and the meaning of the prefix: must be looked in the context, whatever that may be in order to identify a namespace, such as xmlns="urn:ietf:params:xml:ns:yang :ietf-interfaces".

Furthermore, most names will be part of a heirarchy, such as

```
<interfaces>
<interface>
<enabled>true</enabled>
```

and in many cases, such as when an interface object has been added to the base interfaces module by a interface-type specific augmentation, then the heirarchy will come from multiple namespaces, each with associated prefixes.

Elements in such a hierarchy may also be lists (tables) which require index values which again may take the form of multiple elements with multiple namespaces.

It is not so much that this is not logical and straightforward to automate but rather that such automation will be essential, the simple command line interface is likely to become too complex to be used in safety. Of course, the IETF does not involve itself in user interfaces but the complexity thereof may be .. well, a surprise.

7. Twin objects

The management of a device often results in a pair of objects, the desired, configuration state and the actual state, as with operating status or with the speed and duplicity of a MAC card. With MAC cards, the state set by the management system may be for the device to select the best available option ('automatic'), in which case, knowing what that state is (100Mbps, FDX) is essential.

With SNMP, there is no difference between configuration and state and such objects coexist in a row of a table. Netconf splits configuration from state. The objects are related so it is likely that both will appear as objects in a YANG list (ie table), and while the configuration can be retrieved and set by a Netconf get-config or edit-config operation, it will require a get to retrieve them both and both will require filters in order to select the relevant column from the table (to use the terminology of SNMP).

Using just the base interfaces module, such a filter might be

```
<filter type="subtree">
<t:top xmlns:t="urn:ietf:params:xml:ns:yang:ietf-interfaces">
<t:interfaces>
<t:interface t:ifName="eth0"/>
</t:interfaces>
</t:top>
</filter>
```

It seems unlikely that this can be made into a command line interface for the general user.

8.

ifindex changed to ifname tba

9. Security Considerations

There are no security considerations

10. IANA Considerations

There are no IANA considerations.

11. Acknowledgments

This document was written using the xml2rfc tool described in RFC2629 [RFC2629] which is why I cannot make it look the way I would like it to, after four hours writing and eight hours xml2rfc-ing.

12. References

12.1. Normative

[I-D.ietf-netmod-interfaces-cfg]
Bjorklund, M., "A YANG Data Model for Interface Management", draft-ietf-netmod-interfaces-cfg-10 (work in progress), April 2013.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2629] Rose, M.T., "Writing I-Ds and RFCs using XML", RFC 2629, June 1999.

[RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

[RFC6244] Shafer, P., "An Architecture for Network Management Using NETCONF and YANG", RFC 6244, June 2011.

12.2. Informative

[RFC3535] Schoenwaelder, J., "Overview of the 2002 IAB Network Management Workshop", RFC 3535, May 2003.

Author's Address

Tom Petch
Engineering Networks Ltd

Email: tomSecurity@network-engineer.co.uk