

# Secure Remote Configuration of Network Devices

## A Case Study

Radek Krejčí, Ladislav Lhotka, Pavel Čeleda, Petr Špringl

CESNET, z.s.p.o., Zikova 4, 160 00 Prague, Czech Republic  
{krejci,celeda,xsprin01}@liberouter.org, lhotka@cesnet.cz

**Abstract.** This paper describes the Netopeer software system intended for secure remote configuration and management of network devices. The system is based on our open-source implementation of the NETCONF protocol and deployed as the primary configuration system for FlowMon, the IP traffic flow monitoring probe. The paper gives an overview of the system architecture and operation and discusses potential generalizations of the system that would allow to use it for other devices. Plans for future development are also outlined, in particular asynchronous notifications and validation of configuration data by means of a formal data model validation that is being developed by the IETF NETMOD working group.

**Key words:** FlowMon, Liberouter, network configuration, NETCONF, NETMOD, YANG

## 1 Introduction

Since 2002, CESNET has been cooperating with several universities on the development of specialized hardware-accelerated network devices utilizing programmable chips such as FPGA (Field-Programmable Gate Array). In the early stages of this activity, the emphasis was on implementation of basic hardware and software functions in the device prototypes. Their configuration was done locally using low-level and often rather cryptic means.

Recently, some of the devices reached maturity and started to be deployed in production networks. This is especially the case of FlowMon<sup>1</sup>, the IP traffic flow monitoring probe that has been developed by the GÉANT2 project<sup>2</sup> as a part of the Security Toolset, which is now available as a commercial product. With the increasing number of users outside the development team it became necessary to provide them with a more user-friendly configuration interface. As an interim measure, we created a set of configuration shell scripts that can be run from the device console or remotely via SSH. This proved to be a usable interface, which is nonetheless still relatively primitive when measured by the high standards of the integrated CLI (command line) or web interfaces of commercial network devices such as routers. Unfortunately, these interfaces are mostly proprietary and vendor-specific, so they can really serve only as an etalon for usability and effective design.

---

<sup>1</sup> <http://www.liberouter.org/flowmon/>

<sup>2</sup> <http://www.geant2.net>

We therefore decided to develop an open and secure configuration and management framework based on the NETCONF (Network Configuration) protocol [3]. Its name is Netopeer and our ultimate goal is to be able to use this framework for multiple devices. Ideally, the only part specific to a given device would be the data model expressed in a suitable data modelling language. The implementation described in this paper has not yet entirely reached this ideal since it only provides a secure configuration channel through which configuration data may be transferred. More complex features such as fine-grained access control and configuration database with versioning will be added later. Also, the current system relies in some places on the specific data model for the FlowMon probe. Despite of these limitations, we gained enough insight to the problem to be able to generalize this framework towards data model independence.

## 2 NETCONF Protocol

The first attempt for a vendor-neutral configuration and management framework was developed within IETF under the name SNMP (Simple Network Management Protocol) [5]. This protocol has been widely accepted by both equipment vendors and network management systems, but almost exclusively as a vehicle for obtaining statistics and status information from network devices. For configuration purposes, the proprietary CLI and web interfaces remained the preferred and often the only option. As a result, network operators either continued to configure their devices manually or developed convoluted scripts performing configuration through the CLI and web interfaces. However, the latter approach to automatic configuration is inherently fragile and cannot easily handle error conditions.

The NETCONF working group was chartered by IETF with the aim of developing a new vendor-neutral configuration and management protocol. Since one of the identified pitfalls of SNMP was the use of the ASN.1 notation [4], XML was selected as the *lingua franca* for encoding both the protocol messages and configuration data. NETCONF is a client-server protocol with a simple RPC (Remote Procedure Call) layer based on `<rpc>` and `<rpc-reply>` messages. NETCONF was designed to be independent of the data carried by the protocol. It defines a filtering framework that can be used for selecting subtrees of the configuration datastore.

The NETCONF protocol can be extended by specifying new capabilities that modify the existing operations or define entirely new ones. Both the client (management application) and the server (managed device) advertise their lists of supported capabilities in the `<hello>` messages during initial session establishment.

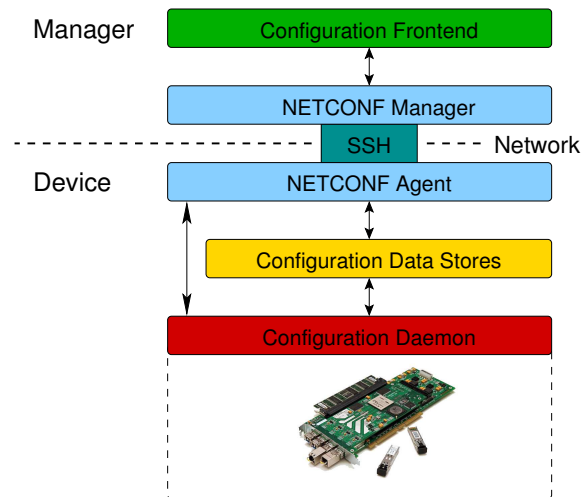
NETCONF is also independent of the underlying transport protocol, which only has to satisfy certain requirements. In particular, it must supply mechanisms for guaranteeing data integrity and security. Every compliant NETCONF implementation must provide SSH transport [7], other alternatives (BEEP, SOAP, TLS) are optional.

The data modelling aspect of XML-based network configuration and management will be addressed by a new NETMOD working group that was chartered by IETF in May 2008. Its primary goal is to develop a new NETCONF specific data modelling language called YANG [1].

### 3 System Architecture

The Netopeer system consists of three parts (see Figure 1):

- Configuration daemon *flowmond* runs at the server side of the system (the FlowMon probe in our case) and is used for applying changes to the target device configuration.
- Web configuration frontend at the client side serves as a simple and intuitive interface to the configuration data.
- NETCONF communication subsystem as the core of the Netopeer system that mediates the communication between the client (NETCONF manager) and server (NETCONF agent). This subsystem provides standard NETCONF RPC operations that create, modify or delete configuration data. This part of the Netopeer system is device independent and can be used for other devices, too.



**Fig. 1.** FlowMon probe remote configuration architecture.

#### 3.1 Configuration Daemon

The configuration daemon is the only public interface through which the device can be configured and status information retrieved. It is able to initialize the device and control its operation via low-level functions such as *ioctl* system calls. It is therefore generally device-specific. The configuration daemon used to control the FlowMon probe is called *flowmond* and its functions include booting the FlowMon firmware, monitoring the device operation, reporting errors and applying configuration changes requested by the NETCONF subsystem.

The communication between the configuration daemon and the NETCONF subsystem is realized through a pair of named pipes (FIFO) that convey messages in both directions between the NETCONF agent and the configuration daemon. Since multiple NETCONF agents may access the device via several independent NETCONF sessions, access to these pipes is controlled by means of a lock file. After applying the requested changes to the device configuration, the daemon informs the NETCONF subsystem about the outcome of the operations that were performed.

The Netopeer system supports three configuration datastores – *startup*, *running* and *candidate*. The *candidate* datastore can be manipulated without affecting the device operation and the daemon actually doesn't use it at all. The daemon only reacts to requests for changes to the *running* configuration and processes them immediately. The *startup* configuration then defines the initial state after the device is restarted. It is a useful safety measure against configuration errors that would otherwise render the device unusable. In the current Netopeer implementation, all datastores are represented as disk files containing serialized XML configuration data.

### 3.2 Web Configuration Frontend

The web configuration frontend is an extension of the simple command line interface implemented by the NETCONF manager. Its HTML forms cover all configuration parameters of the FlowMon probe and also allow for performing critical administrative operations such as reboot or shutdown. User access is controlled through login name and password.

The frontend works essentially as a specialized XML editor. After establishing the NETCONF session with the target probe, the frontend automatically retrieves the *startup* and *running* configurations, stores them locally and makes them accessible for editing. After selecting the configuration datastore (*running* or *startup*), the user modifies the configuration locally and then sends its new contents back to the probe via NETCONF subsystem by performing an explicit commit.

We plan to make the frontend more generic so that it can be used for configuring other network devices as well.

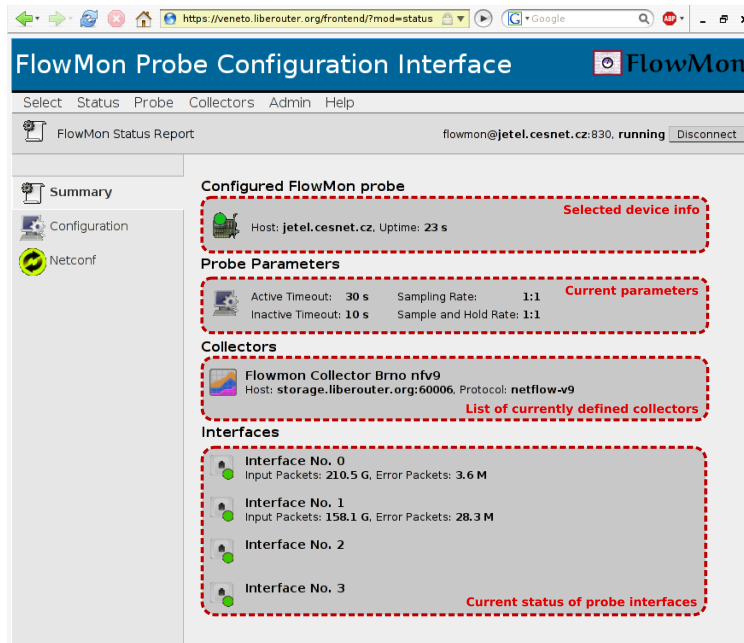
### 3.3 NETCONF Subsystem

The NETCONF subsystem is designed as a client-server application. Its main role is to mediate messages between the server (configuration daemon) and the client (NETCONF manager and web configuration frontend). The NETCONF subsystem is a full implementation of the NETCONF protocol according to RFC 4741 [3]. It currently supports only the mandatory SSH transport [7].

It is also possible to use the NETCONF subsystem without the web frontend directly through the NETCONF manager command line interface.

### 3.4 Data Model

The charter of the IETF NETCONF working group intentionally left the area of configuration data modelling outside its scope. We have been actively participating



**Fig. 2.** FlowMon probe web configuration frontend.

in preparing the charter for the new IETF NETMOD working group that was officially formed in May 2008 with the aim to address the data modelling issues. Naturally, this group has not produced any standards yet and so we had to work out our own data modelling approaches.

The entire set of FlowMon configuration data, including read-only status data and statistics, is represented in a single XML tree with almost 50 leaf elements (data nodes). Since most configuration parameters are optional or have default values, a typical configuration is much smaller and may look as shown in the following listing:

```
<?xml version="1.0"?>
<flowmon xmlns="http://cesnet.cz/ns/netopeer/flowmon/3.0">
  <global>
    <activeTimeout>30</activeTimeout>
    <inactiveTimeout>10</inactiveTimeout>
  </global>
  <interfaces>
    <interface>
      <ifNumber>0</ifNumber>
      <inputSampling>
        <samplingRate>10</samplingRate>
      </inputSampling>
    </interface>
  </interfaces>
</flowmon>
```

```

<collectors>
  <collector>
    <description>First collector</description>
    <host content-type="name">collector.example.net</host>
    <udpPort>60000</udpPort>
    <exportProtocol>netflow-v5</exportProtocol>
  </collector>
</collectors>
</flowmon>

```

The FlowMon configuration data model is defined by means of a schema written in the RELAX NG language [6]. The annotated version of the schema is available online<sup>3</sup>. So far, the schema has been used mainly by software developers as a reference specification and also for debugging purposes. We utilized the extensibility of RELAX NG and augmented the schema by other attributes (“hints”) in a different XML namespace. For example, one of the hints identifies elements contained in list items that serve as lookup keys for these items.

Our plan is to adopt future results of the IETF NETMOD working group and implement strict validation of the contents of NETCONF protocol data units. To this end, we already converted the RELAX NG schema to a data model expressed in the YANG language<sup>4</sup>.

### 3.5 System Operation

The Netopeer system is initialized by starting the NETCONF manager program. It prepares the SSH connection by invoking the SSH client program. Further communication between NETCONF manager and SSH client is realized via redirecting standard input and output of SSH to pipes prepared by the NETCONF manager before invoking the SSH client program.

The NETCONF client uses the *Subsystem* feature of SSHv2 for invoking the server program. The advantage of this approach is that the client need not know the exact name and path of the program in the server’s filesystem. The correspondence between the subsystem and a concrete executable program is configured on the server (in the `/etc/ssh/sshd_config` file).

After the NETCONF session is established, both the client and server announce the capabilities they support. Only those supported by both sides are used during the session. Further on, the communication is event-driven. The NETCONF manager translates user requests into `<rpc>` messages and sends them to the agent. The agent carries out the requested operations, mostly in cooperation with the configuration daemon that performs the actual device parameter changes. As soon as all requested operations have been applied, the agent creates an `<rpc-reply>` message containing the result of the request and sends it through the SSH connection to the manager. This response is then passed to the frontend or displayed directly by manager.

<sup>3</sup> <http://www.flowmon.org/flowmon-probe/devel/config/flowmon-rng>

<sup>4</sup> <http://www.yang-central.org/twiki/pub/Main/YangExamples/flowmon.yang>

## 4 Adding Support for Other Devices

The goal of the Netopeer project is to prepare a configuration and management framework applicable to multiple devices. Ultimately, the only part specific to a given device should be the configuration data model. So far the system has not yet reached this generality but thanks to its modular architecture it is immediately possible to use the NETCONF communication subsystem and supply only a device-specific backend and frontend: a configuration daemon on the server side and user interface on the client side. Due to the generic inter-process communication method (named pipes), it shouldn't be difficult to write such replacements suitable for other hardware devices and/or software programs.

As mandated by the NETCONF protocol, this implementation is independent of the data being transported. However, the subtree filtering feature requires that certain information about used data model be passed to the NETCONF subsystem. This information is currently provided in the form of XSLT stylesheets. While these stylesheets are specific to the FlowMon data model, they can be easily adapted for other uses.

## 5 Conclusions and Future Work

The Netopeer software presented in this paper can be used for secure remote configuration of various devices. Its main part – the NETCONF subsystem – is a general implementation of the NETCONF protocol fully compliant with the specification [3]. To our knowledge, this is the first open-source NETCONF implementation written in the C language.

The system is currently used as the primary method for configuring the FlowMon probe. However, due to its modular structure it can be easily reused for other devices – it is only necessary to provide a new device-specific backend and frontend.

Our future efforts will concentrate on integrating the NETCONF subsystem into a network device configuration and management system supporting multiple devices and multiple managers. Further, the existing method of representing configuration datastores as simple disk files will be replaced by a more sophisticated database backend. Concerning the NETCONF protocol implementation itself, we plan to add several optional features. Of particular importance is the notification capability [2] that enables the managed device to send asynchronous notification messages similar to SNMP traps.

We are actively participating in the IETF NETMOD working group. Its expected outcome, a general-purpose data modelling language named YANG, will allow us to extend the Netopeer system by adding a new device-independent software layer capable of a detailed validation of configuration data. The YANG data models will also include new RPC methods for system management functions such as system shutdown, reboot or firmware update.

On the client side, we need to improve the communication between the NETCONF manager and user interface frontend. In the current prototypical implementation, the frontend executes the NETCONF manager in batch mode. As a result, the NETCONF connection is opened only for the duration of given batch. Therefore the frontend e.g. is not able to lock a datastore for long time since all locks are automatically released upon

session termination. The future version of the frontend will directly call the NETCONF manager functions while keeping the NETCONF connection open across multiple RPC operations.

### Acknowledgement

This work is supported by the EU Sixth Framework GÉANT2 project (FP6-IST 511082) and by the Research Intent of the Czech Ministry of Education MSM6383917201.

### References

1. Bjorklund, M. (Ed.): *YANG – A data modeling language for NETCONF*. Work in progress (draft-ietf-netmod-yang-00), 2008. URL: <http://www.ietf.org/internet-drafts/draft-ietf-netmod-yang-00.txt>
2. Chisholm, S., Trevino, H.: *NETCONF Event Notifications*. Work in progress (draft-ietf-netconf-notification-12). URL: <http://www.ietf.org/internet-drafts/draft-ietf-netconf-notification-12.txt>
3. Enns, R. (Ed.): *NETCONF Configuration Protocol*. RFC 4741. IETF, 2006. URL: <http://www.ietf.org/rfc/rfc4741.txt>
4. Schönwälder, J.: Protocol Independent Network Management Data Modeling Languages – Lessons Learned from the SMIng Project. *IEEE Communications* 46(5):148–153, 2008.
5. Stallings, W.: *SNMP, SNMPv2 and RMON*. Addison-Wesley, 1996. 493 p. ISBN 0-201-63479-1.
6. van der Vlist, E.: *RELAX NG*. O’Reilly, 2004. 504 p. ISBN 0-596-00421-4.
7. Wasserman, M., Goddard, T.: *Using the NETCONF Configuration Protocol over Secure Shell (SSH)*. RFC 4742. IETF, 2006. URL: <http://www.ietf.org/rfc/rfc4742.txt>