Managing SamKnows Probes using NETCONF

Vaibhav Bajpai
Computer Science, Jacobs University Bremen
v.bajpai@jacobs-university.de

Radek Krejčí
CESNET, z.s.p.o.
rkrejci@cesnet.cz

***Abstract*—Network Configuration (NETCONF) is being considered by the Internet Engineering Task Force (IETF) as one of the control protocol candidates within the Large-Scale Measurement of Broadband Performance (LMAP) framework. We demonstrate the possibility of managing LMAP Measurement Agent (MA)s using the NETCONF protocol. We have deployed a NETCONF server on one such MA: a SamKnows measurement probe. The server is built around the `libnetconf` library, and has been heavily optimized to accommodate it to the limitations of the SamKnows hardware.**

I. Introduction

The IETF has recently chartered a LMAP working group[1], whose goal is to standardize interactions between different elements of a measurement platform. SamKnows[2] is one such platform that performs active measurements using hardware-based MAs to help assess the performance of a broadband access network. These MAs are off-the-shelf home routers that have been reflashed with a custom OpenWrt[3] firmware to perform specific measurements. There are around 20K such MAs deployed by SamKnows all around the globe, however, they are currently being managed using proprietory methods. Within the LMAP working group there is a strong inclination towards using existing protocols to manage such MAs. This will allow network vendors to implement measurement capability directly inside the Customer Premises Equipment (CPE) giving away the need to deploy dedicated hardware MAs. NETCONF [1] is being considered [2] as one of the LMAP control protocols that can be used to manage such MAs. However, home routers do not support NETCONF today. In order to understand the memory impact and possibility of using NETCONF on these low-end consumer devices, we have deployed a NETCONF server on one of the SamKnows probes. We demonstrate how a NETCONF client can be used to configure a SamKnows MA using the deployed NETCONF server as shown in Fig. 1.

II. Deploying a NETCONF server

NETCONF was standardized by the IETF in early 2006. However, there is still weak support for NETCONF-based server-side applications. `libnetconf` [3] is the only actively developed and maintained open-source implementation of a NETCONF server-side functionality today. Furthermore, it is written in C so it can be easily optimized for limited SamKnows hardware. In this section we describe a number of technical challenges we faced during the process of deploying a NETCONF server on SamKnows probes. These real-world

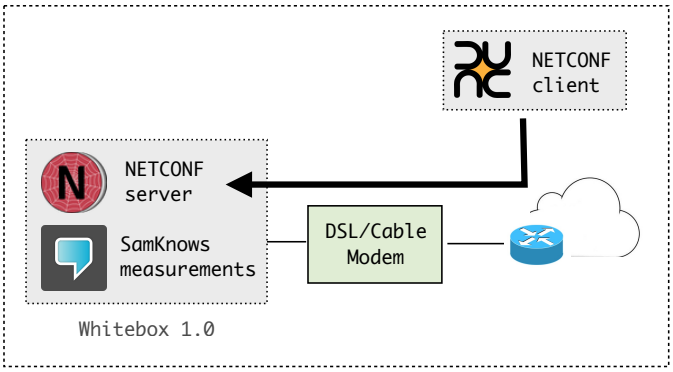

Fig. 1. `ncclient` establishing a NETCONF session with a `libnetconf`-based server running on a SamKnows MA. The probe in a conventional setup is connected behind a residential gateway, where it performs regular active measurements tests. This setup can now be used to also manage the MA from the outside world.

issues led to improvements of the `libnetconf` library as well as of some other OpenWrt packages.

A. C library restrictions

The `libnetconf` library uses GNU extensions and some other features (such as `utmpx`) that are not available in `uClibc`, the C library deployed on SamKnows probes. Therefore, `libnetconf` was modified to detect availability of such extensions. In case of missing C library features, `libnetconf` was enhanced with some alternative ways to achieve a requested functionality.

B. SSH package misconfigurations

There is a misconfiguration in the `libssh2` OpenWrt package. The package is built with static and dynamic library support by default. However, the package although does export dynamic libaries, it does not export static libraries into staging directories. This makes the static library unusable by default. We updated the package sources and informed the OpenWrt package maintainers to get this resolved upstream.

C. NETCONF event notifications

`libnetconf` supports NETCONF event notifications [4]. It was using `dbus` to share information about these events between `libnetconf` instances. However, SamKnows probes run a custom version of OpenWrt that does not support `dbus`. Besides an added `--disable-notifications` build process option to switch off support for NETCONF notifications, the `libnetconf` event notifications codebase was also simplified and `dbus` is no more required.

[1] `https://datatracker.ietf.org/wg/lmap`
[2] `http://www.samknows.com`
[3] `https://openwrt.org`

978-1-4799-0913-1/14/$31.00 © 2014 IEEE

D. Optimizations

The NETCONF server binary before being deployed was 2.6Mib in size. A considerable part of this size was occupied by the statically linked `libssh2` library. `libssh2`, however, is used only by client-side functions to establish an `ssh` connection to the server. For server-side applications, this functionality is needless. Furthermore, on client-side this requirement can be alternated by utilizing a standalone `ssh` client application. Therefore, we have updated the `libnetconf` build process with a `--disable-libssh2` option that dramatically reduces the binary size of the server implementation. Eventually, the `libnetconf` server depends on `libevent` and `libxml2`. We had to statically link these libraries into the final binary because they are not available on SamKnows probes. We stripped the final binary to remove any symbol table information and debugging symbols. The server reduced to 1.5MiB in size.

E. SSH subsystems

A NETCONF server must be hooked as a subsystem in the `ssh` server implementation. This allows the `ssh` server to delegate all NETCONF calls to the server daemon on standard port 830. However, SamKnows probes run `dropbear` as a `ssh` server implementation. `dropbear` is a tiny `ssh` server implementation that is mostly suited for embedded Linux, such as wireless routers, but they do not have any support to hook subsystems. OpenWrt provides packages for alternative OpenSSH server implementation, however there is not enough flash memory space available to run a OpenSSH server on the SamKnows probes. In order to circumvent the issue, we deployed a separate `dropbear` instance to specifically handle NETCONF specific calls on port 830. Remote commands, such as launching the NETCONF server daemon can be issued on this `ssh` channel, which will allow one to talk to the server daemon from the outside world. However, the client is not going to know where the server is located on the remote machine. As a workaround, we created a symbolic link from the shell's search PATH to the custom remote location of the server location, so that the server daemon is executed when the client drops down the `ash` shell.

III. Interacting from outside

`ncclient` [5] is a Python library that facilitates client-side scripting and application development around the NETCONF protocol. However, traditionally `ncclient` only supported NETCONF v1.0 [6]. We added NETCONF v1.1 [1] support to `ncclient` that allows it to handle chunked frames [7]. However, we have also kept backward compatibility to allow the client to fallback to NETCONF v1.0 end of message framing scheme [8] to keep support for legacy servers that do not advertise NETCONF v1.1 capability. `ncclient` also used to assume that the NETCONF server is configured as a subsystem in the `ssh` server implementation. We updated `ncclient` to also try remote command invocation as a fallback (in aforementioned situations) when NETCONF subsystem invocation fails. With both, `libnetconf` and `ncclient`, we participated at the NETCONF interoperability testing event at IETF 85, Atlanta[4] where the interoperability with other NETCONF implementations was successfully tested.

[4] `http://www.internetsociety.org/articles/successful-netconf-interoperability-testing-announced-ietf-85`

IV. Probe management

For a basic configuration of SamKnows probes we have followed the core system data model [9] designed by the IETF NETCONF Data Modeling Language (NETMOD) Working Group. This YANG data model specifies configuration and state data used to set and retrieve information about the system identification, time and users management as well as a DNS resolver. Furthermore, the model defines new NETCONF operations for a device restart and shutdown.

We have prepared a `libnetconf` core system `transAPI` module for our NETCONF server. The module is a set of callbacks employed when specific parts of a configuration data are changed. For example, when a user changes timezone settings in the `running` datastore, a specific callback function applying this parameter to the probe is automatically called. Similarly for NETCONF RPC operations, for example, when the system-restart operation is requested by a client, `libnetconf` passes the program control to the appropriate `transAPI` module function that performs a reboot of the SamKnows probe.

V. Acknowledgements

We would like to thank Sam Crawford for providing us with a SamKnows probe and Kinga Lipskoch for reviewing our manuscripts. This work was supported by the European Community's Seventh Framework Programme (FP7/2007-2013) Grant No. 317647 (Leone). The `libnetconf` development is supported by the "CESNET Large Infrastructure" project LM2010005 funded by the Ministry of Education, Youth and Sports of the Czech Republic.

References

[1] R. Enns, M. Bjorklund, J. Schoenwaelder, and A. Bierman, "Network Configuration Protocol (NETCONF)," RFC 6241 (Proposed Standard), Internet Engineering Task Force, Jun. 2011. [Online]. Available: http://www.ietf.org/rfc/rfc6241.txt

[2] J. Schoenwaelder, "Considerations on using NETCONF with LMAP Measurement Agents," Internet Engineering Task Force, Internet-Draft draft-schoenw-lmap-netconf-00, Feb. 2013, work in Progress. [Online]. Available: http://tools.ietf.org/html/draft-schoenw-lmap-netconf-00

[3] R. Krejci, "Building NETCONF-enabled Network Management Systems with libnetconf," in *Integrated Network Management (IM), 2013 IFIP/IEEE International Symposium on*, 2013, pp. 756–759.

[4] S. Chisholm and H. Trevino, "NETCONF Event Notifications," RFC 5277 (Proposed Standard), Internet Engineering Task Force, Jul. 2008. [Online]. Available: http://www.ietf.org/rfc/rfc5277.txt

[5] S. Bhushan, H. Tran, and J. Schönwälder, "NCClient: A Python Library for NETCONF Client Applications," in *IP Operations and Management*, ser. Lecture Notes in Computer Science, G. Nunzi, C. Scoglio, and X. Li, Eds. Springer Berlin Heidelberg, 2009, vol. 5843, pp. 143–154. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-04968-2_12

[6] R. Enns, "NETCONF Configuration Protocol," RFC 4741 (Proposed Standard), Internet Engineering Task Force, Dec. 2006, obsoleted by RFC 6241. [Online]. Available: http://www.ietf.org/rfc/rfc4741.txt

[7] M. Wasserman, "Using the NETCONF Protocol over Secure Shell (SSH)," RFC 6242 (Proposed Standard), Internet Engineering Task Force, Jun. 2011. [Online]. Available: http://www.ietf.org/rfc/rfc6242.txt

[8] M. Wasserman and T. Goddard, "Using the NETCONF Configuration Protocol over Secure SHell (SSH)," RFC 4742 (Proposed Standard), Internet Engineering Task Force, Dec. 2006, obsoleted by RFC 6242. [Online]. Available: http://www.ietf.org/rfc/rfc4742.txt

[9] A. Bierman and M. Bjorklund, "A YANG Data Model for System Management," Internet Engineering Task Force, Internet-Draft draft-ietf-netmod-system-mgmt-12, Feb. 2014, work in Progress. [Online]. Available: http://tools.ietf.org/html/draft-ietf-netmod-system-mgmt-12