Evaluation of the YANG-based network management framework Clixon

Simon Bauer
Esslingen University of Applied Sciences
bauer-s-public@outlook.com

Martin Mager
Esslingen University of Applied Sciences
magerm.dev@gmail.com

Michael Scharf
Esslingen University of Applied Sciences
michael.scharf@hs-esslingen.de

***Abstract*—YANG-based network management is gaining importance and requires corresponding implementations. This paper evaluates and compares the open source framework *Clixon* with other well-known open source solutions as well as commercial alternatives. We analyze the the *Clixon* framework with a prototype. As use case it implements the draft of a standard YANG model for the Transmission Control Protocol (TCP). Our results confirm that *Clixon* is a a portable and extensible implementation well-suited for YANG-based management.**

***Index Terms*—Clixon, YANG, NETCONF, network management.**

I. Introduction

State-of-the-art network management requires standardized network management protocols and data modeling languages. The Network Configuration Protocol (NETCONF) [1] and the data modeling language YANG [2] have gained acceptance among vendors of network equipment. Also other industries show interest in these standardized and field-proven technologies to handle management of complex systems. This may extend the usage of NETCONF/YANG from network devices to a wide range of industries.

The growing acceptance and usage of NETCONF/YANG drives the need for common, well-tested and future-proof management frameworks that work across many environments. While there are several commercial options available, like *ConfD* from Cisco [3], they are mostly closed-source and do not allow for any modifications or adjustments to the base product. Besides commercial options, multiple open-source management frameworks exist, allowing companies to gain full control and adjust the framework to their individual needs.

This paper introduces and evaluates the open source network management framework *Clixon* and presents a comparison of its capabilities with the well-known open source framework *Netopeer2* [4] and the commercial, closed-source alternative *ConfD*. *Clixon* is intended for network devices and other computer systems and provides support for a large feature set, such as datastores and appropriate transaction mechanisms. Developers can easily integrate in the framework by providing plugins to react to configuration changes or return state data. To interact with the management agent, the network management protocols NETCONF and RESTCONF as well as a rudimentary command line interface (CLI) are provided [5].

The rest of this paper is organized as follows. Section II introduces *Clixon*, presents an overview of the architecture and draws a comparison with other open-source frameworks. The *ietf-tcpm-yang-tcp* [6] YANG model is introduced in Section III and the corresponding *Clixon* plugin is shown. In addition, the portability of the prototype and the underlying framework is analyzed. The related work is summarized in Section IV. Finally, Section V concludes the paper.

II. Clixon Framework

Clixon is an open-source YANG-based configuration manager that has been developed by Olof Hagsand since 2009.

The framework is characterized by its modular structure and plugin architecture which allows for easy extension and modification. *Clixon* provides inbuilt support for datastore management and multiple common interfaces, namely *CLI*, *NETCONF* and *RESTCONF*.

Clixon is targeted towards GNU/Linux and virtualized environments, like Docker, but a community driven FreeBSD port is available in addition to that. Due to the broad platform support, the framework can be used in a variety of environments and for different scenarios.

This section is structured as follows. First, the overall architecture of the framework is presented. Then, the important plugin architecture is discussed in more detail. Finally, the available features of *Clixon* are compared with *Netopeer2* and *ConfD*.

A. Overall Architecture

Clixon is built in a modular fashion to allow for easy extension and integration. The overall system architecture is shown in Fig. 1, highlighting the major components of a common usage scenario [5], [7].

1) *Backend*: The backend daemon handles various core functionalities of the framework. These include tasks such as managing data stores, privileges and the plugin system. In addition, it provides a NETCONF-based inter-process communication (IPC) bus to enable communication with the other framework components.
2) *Interfaces*: The already built-in internal clients CLI, RESTCONF and NETCONF provide external interfaces for management. These include RESTCONF over HTTP/HTTPS, NETCONF over TCP or SSH and an interactive CLI Interface. By default, the internal clients

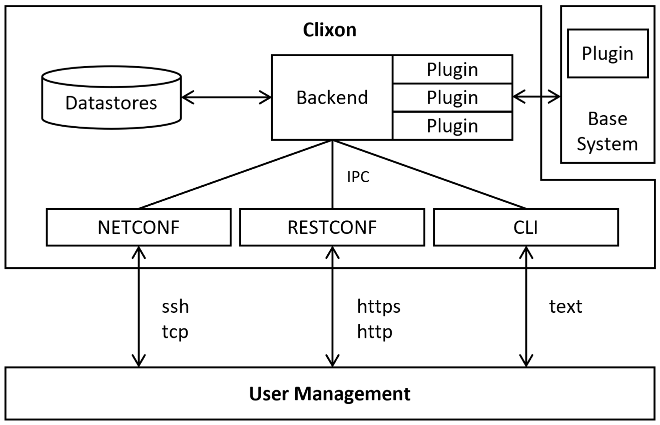

Fig. 1. Clixon Architecture.

are running as separate processes that communicate with the backend via IPC.

3) *Plugins*: Plugins are shared libraries which provide the required system interaction with the base system for the various YANG specifications. Plugins attach to certain events and therefore get notified of configuration changes and are able to provide state data from the system. The Plugin architecture is discussed in more detail in Section II-B.
4) *Datastores*: *Clixon* implements the *candidate*, *running*, and *startup* datastores, specified in RFC 6241 [1]. In addition, *Clixon* provides an interface for interacting with the datastores provided, facilitating transactional workflows. Currently, *Clixon* does not offer support for the Network Management Datastore Architecture (NMDA) [8] and therefore does not provide an *intended* and *operational* datastore. However, due to the extensible structure of the *Clixon* framework, this feature can be implemented with reasonable effort.

B. Plugin Architecture

The *Clixon* daemon loads the YANG modules and therefore knows about the structure of the involved data. This allows *Clixon* to maintain the datastores, handle read and update requests, and expose this functionality via its interfaces. The daemon on its own, however, does not have any knowledge about the semantics of the involved YANG model and in consequence is unable to apply any changes to the base system.

The semantics is provided to *Clixon* in the form of plugins which implement the required logic to interact with the system. Plugins are dynamically loaded shared objects which expose a function named `clixon_plugin_init`. This method is called during startup and allows plugins to attach various callbacks to the *Clixon* daemon. Besides several lifecycle events, like `start`, `daemon` and `exit`, the main callbacks consist of several transaction hooks, such as `trans_begin`, `trans_commit` and `trans_abort`, and a callback to provide state data. Other useful events exist to handle YANG extension data, datastore upgrades, remote procedure call (RPC) events and system resets.

Clixon follows NETCONF in terms of validate and commit semantics and provides the necessary infrastructure for transaction processing. Plugins usually subscribe to multiple lifecycle events of the transaction and thereby supply the logic to validate the configuration changes and incrementally upgrade the system state based on the configuration changes. Clixon provides a feature-rich XML and transaction interface that allows plugins to easily detect changes in the XML tree and act accordingly.

C. Feature Comparison

There are several NETCONF/YANG frameworks with different functionality and different levels of support of standards. This section compares *Clixon* with two well-known alternatives. The first one is the open-source tool *Netopeer2*, and the second one is the commercial closed-source framework *ConfD*. As criteria for the comparison, the supported RFC standards are analyzed, as well as the support for XML and XPath.

Netopeer2 is a NETCONF server implementation based on the open-source projects *libyang*, *libnetconf2* and *sysrepo*. It is in its second generation and is already well-established in the field of network management [4].

ConfD, developed by the *Cisco* company *Tail-F*, is a commercially distributed NETCONF/YANG toolset. It is available in a free and limited *Basic* or paid *Premium* edition. *ConfD* is built on a more than 10-year development foundation and offers a variety of features that the open-source solutions cannot provide. In the following, only the premium version of *ConfD* is considered [3].

Table I presents a brief overview and comparison of the features provided by the above-mentioned NETCONF/YANG frameworks.

The comparison has shown that *Netopeer2* is the only network management framework in the comparison that does not offer a RESTCONF interface. Furthermore, it is noticeable that *Clixon* offers only partial support for certain standards, which in many cases is due to diverging default values or left-out features. In addition, *Clixon* currently does not support NMDA for NETCONF. The commercial solution *ConfD* scores best across all analyzed criteria, but comes at an expensive price.

III. Prototype

To evaluate the capabilities and modular structure of *Clixon*, a prototype was developed based on the YANG model specified in the RFC draft *ietf-tcpm-yang-tcp* [6].

In the following subsections, first, the proposed YANG model *ietf-tcp* used in the prototype is introduced. Subsequently, the difficulties encountered during the development of the plugin are outlined. In the last subsection III-C the portability of the prototype and the underlying *Clixon* framework is analyzed based on the Blackberry real-time operating systems (RTOS) QNX and an armv7 system architecture as evaluation platform.

TABLE I
FEATURE COMPARISON

YANG	**RFC**	***Clixon*** *(v5.4.0)*	***Netopeer2*** *(v2.0.35)*	***ConfD*** *(v7.6)*
YANG 1.0	6020	✓	✓	✓
YANG 1.1	7950	(✓)	✓	✓
YANG Library	8525	(✓)	✓	✓

NETCONF	**RFC**	***Clixon*** *(v5.4.0)*	***Netopeer2*** *(v2.0.35)*	***ConfD*** *(v7.6)*
NETCONF Event Notifications	5277	(✓)	✓	✓
Network Configuration Protocol (NETCONF)	6241	(✓)	✓	✓
Using the NETCONF Protocol over Secure Shell (SSH)	6242	✓	✓	✓
NETCONF Call Home and RESTCONF Call Home	8071	(✓)	✓	✓
Network Configuration Access Control Model	8341	✓	✓	✓
NETCONF Extensions to Support the NMDA	8526	✗	✓	✓

RESTCONF	**RFC**	***Clixon*** *(v5.4.0)*	***Netopeer2*** *(v2.0.35)*	***ConfD*** *(v7.6)*
RESTCONF Protocol	8040	(✓)	✗	✓
RESTCONF Extensions to Support the NMDA	8527	(✓)	✗	✓
YANG Patch Media Type	8072	(✓)	✗	✓

Other		***Clixon*** *(v5.4.0)*	***Netopeer2*** *(v2.0.35)*	***ConfD*** *(v7.6)*
XML 1.0		(✓)	✓	✓
XPath 1.0		(✓)	✓	✓

(✓) = partially supported RFCs, further information: https://clixon-docs.readthedocs.io/en/latest/standards.html

A. IETF TCP YANG Model

The TCP Maintenance and Minor Extensions (TCPM) working group of the IETF is currently working on a standardized YANG Model for the TCP stack named *ietf-tcp* [6]. It specifies a minimal YANG model for the TCP stack which is presented in Figure 2. It consists of a container for all TCP connections and a container for basic TCP statistics. In addition, it defines groupings of authentication parameters that can be reused by other models.

To our knowledge, the presented prototype is the first implementation of the proposed *ietf-tcp* model. However, it must be mentioned that the YANG model could not be implemented to its full extent, since at the time of implementation the functionality required for the authentication container was not supported by the Linux kernel. Furthermore, the list of TCP connections was only implemented as a read-only list because write access was not applicable in the prototype environment. Likewise, the reset action for the TCP statistics could not be implemented properly due to the lack of support provided by the operating systems used (Ubuntu, QNX).

Nevertheless, the evaluation phase provided valuable insights that contributed to the further development of the YANG model [1].

B. Plugin Development

Unlike the *Clixon* framework, which is written entirely in C and based on the *Autotools toolchain* as the build system, the plugin was developed using C++ and *CMake* for better development experience.

[1]Further information about changes to the YANG model and the contribution of this work can be found in the slide decks for IETF 110 [9] and IETF 111 [10].

To obtain the TCP connections and statistics specified in the YANG model from the base system, in this case Ubuntu, the pseudo files `/proc/net/snmp` and `/proc/net/tcp` were read and parsed.

For testing the implementation of the plugin, *YangSuite* was used, which was fairly new at the time of implementation. It provides interfaces like NETCONF and RESTCONF gNMI and and can be operated in a docker container. Fig. 3 shows how *YangSuite* integrates into the user management layer and into the overall setup.

The Source code for the implemented plugin can be found on GitHub at https://github.com/mager-m/ietf-tcp-research-project.

C. Portability to QNX

QNX is a commercially distributed real time operating system (RTOS) that is used through various industries and serves as the basis in many embedded devices, including network related equipment. Therefore QNX has been chosen as the evaluation platform for the portability analysis. A physical development board, based on the ARMv7 architecture, has been used in the process.

Since *Clixon*'s goal is to provide a YANG-based configuration manager with support for many platforms, it already runs on a variety of operating systems. To achieve this, *Clixon* relies on different feature flags and the use of *Autotools toolchain* to detect the platform capabilities at configuration time. This not only made it easier to get the framework up and running, but also to adapt any necessary changes as described in the following.

During the deployment of the framework to the new platform a memory corruption appeared while reading the YANG

```
module: ietf-tcp
 +-rw tcp!
   +-rw connections
   | +-rw connection*
   |   +-rw local-address
   |   +-rw remote-address
   |   +-rw local-port
   |   +-rw remote-port
   |   +-rw common
   |     +-rw keepalives!
   |     |  +-rw idle-time
   |     |  +-rw max-probes
   |     |  +-rw probe-interval
   |     +-rw (authentication)?
   |       +-:(ao)
   |       |  +-rw enable-ao?
   |       |  +-rw send-id?
   |       |  +-rw recv-id?
   |       |  +-rw include-tcp-options?
   |       |  +-rw accept-key-mismatch?
   |       +-:(md5)
   |         +-rw enable-md5?
   +-ro statistics {statistics}?
     +-ro active-opens?
     +-ro passive-opens?
     +-ro attempt-fails?
     +-ro establish-resets?
     +-ro currently-established?
     +-ro in-segments?
     +-ro out-segments?
     +-ro retransmitted-segments?
     +-ro in-errors?
     +-ro out-resets?
     +-x reset
       +-w input
       | +-w reset-at?
       +-ro output
         +-ro reset-finished-at?
```

Fig. 2. TCP YANG.

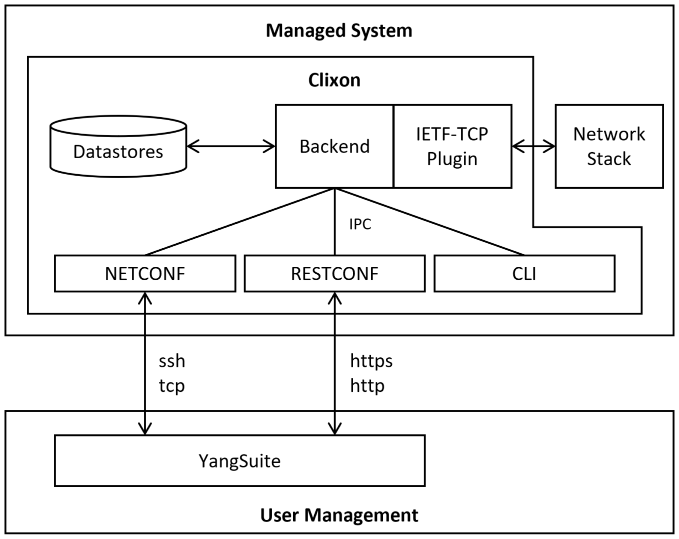

Fig. 3. Plugin Architecture

models from the disk. Presumably due to a different memory management it has not been noticed with Ubuntu so far. An appropriate fix was implemented and merged in agreement with the maintainer into the project as pull request.

Another challenge was the lack of support for the multiple privilege functions such as `setresuid` and `getresuid` that are used by *Clixon* to drop privileges after initialization. To overcome this issue, an additional check in *Autotools* along with the feature flag HAVE_SETRESUID was introduced. Thus, if `getresuid / setresuid` is not available on the platform, it will be detected during the configuration phase and the relevant code will not be included in the compilation. This change was also merged into the project with a pull request.

Due to the highly generic code structure, the deployment of the *Clixon* framework to the evaluation platform proceeded without further challenges.

Since QNX does not have the same pseudo files for reading TCP connections and statistics as Ubuntu, modifications to the plugin were necessary. To overcome the non-existent files, the information was obtained by using the `netstat` command and parsing the corresponding output. Otherwise, no major changes were necessary.

The deployment of the prototype to the evaluation platform has confirmed the initial assumption and the advertisement of *Clixon*. Due to the generic code base and the use of the *Autotools toolchain* along with many feature flags, the framework proved to be very flexible and easily portable to new UNIX-like environments.

The various challenges faced during the deployment have also highlighted the benefits of open-source software. Besides the possibility to make changes directly to the source code, there is a solid community for emerging questions during the development. Since the changes can be contributed and merged, the framework is constantly evolving and so are the capabilities.

IV. Related Work

Krejci (2013) [11] presents the open source library *libnetconf* in his work, which is used by *Netopeer2*. In addition to the various use cases for developing network management applications based on the NETCONF protocol using the presented library, the supported NETCONF capabilities are outlined and guidelines for using the library are given. However, further open-source and commercial YANG-based network software solutions are not compared to the *libnetconf* library in greater detail.

Bajpai and Schönwälder (2014) [12] present a NETCONF interoperability lab platform to evaluate NETCONF server and client implementations based on various test-cases. In their work, four different tools are compared, including *YumaPro*, *OpenYuma*, *ConfD* and *libnetconf*.

To our knowledge, there is no other work that analyzes Clixon as a YANG-based network management framework in more detail.

V. Conclusion

The prototype has proven that *Clixon* satisfies its claim to be easily adaptable to many platforms and architectures without any substantial changes. The modular and generic structure of the framework not only allows the mentioned flexibility in terms of operating system or system architecture, but also makes necessary changes much easier, which is an important aspect for development and deployment.

The plugin architecture also proved to be very capable and application-oriented during the development of the prototype due to the existing functionalities of the framework and the accompanying flexibility of the plugins.

However, the comparison with the two alternatives *Netopeer* and *ConfD* also showed that *Clixon* still has potential for improvement in some areas, such as the partially deviating implementation from the standards.

With regard to the *ietf-tcp* model, there are still a few open topics that can be implemented with upcoming kernel support for the missing authentication features and future releases of the *Clixon* framework.

References

[1] R. Enns, M. Bjorklund, J. Schoenwaelder, and A. Bierman, “Network configuration protocol (netconf),” Internet Requests for Comments, RFC Editor, RFC 6241, June 2011. [Online]. Available: http://www.rfc-editor.org/rfc/rfc6241.txt

[2] M. Bjorklund, “The yang 1.1 data modeling language,” Internet Requests for Comments, RFC Editor, RFC 7950, August 2016.

[3] Tail-f, a Cisco Company, “Confd.” [Online]. Available: http://www.tail-f.com/confd

[4] CESNET, “Netopeer2.” [Online]. Available: https://github.com/CESNET/netopeer2

[5] O. Hagsand, “Clixon documentation.” [Online]. Available: https://clixon-docs.readthedocs.io/en/latest/

[6] M. Scharf, M. Jethanandani, and V. Murgai, “Yang model for transmission control protocol (tcp) configuration,” Working Draft, IETF Secretariat, Internet-Draft draft-ietf-tcpm-yang-tcp-04, October 2021. [Online]. Available: https://www.ietf.org/archive/id/draft-ietf-tcpm-yang-tcp-04.txt

[7] O. Hagsand, “Clixon.” [Online]. Available: https://github.com/clicon/clixon

[8] M. Bjorklund, J. Schoenwaelder, P. Shafer, K. Watsen, and R. Wilton, “Network management datastore architecture (nmda),” Internet Requests for Comments, RFC Editor, RFC 8342, March 2018.

[9] M. Scharf, V. Murgai, and M. Jethanandani, “Status update on draft-ietf-tcpm-yang-tcp,” in *IETF 110 Proceedings*, 2021. [Online]. Available: https://datatracker.ietf.org/meeting/110/materials/slides-110-tcpm-draft-ietf-tcpm-yang-tcp-00

[10] ——, “Status update on draft-ietf-tcpm-yang-tcp,” in *IETF 111 Proceedings*, 2021. [Online]. Available: https://datatracker.ietf.org/meeting/111/materials/slides-111-tcpm-tcp-yang-update-00

[11] R. Krejci, “Building netconf-enabled network management systems with libnetconf,” in *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*, 2013, pp. 756–759.

[12] V. Bajpai and J. Schönwälder, “Netconf interoperability lab,” in *2014 IEEE Network Operations and Management Symposium (NOMS)*, 2014, pp. 1–2.