LINC Switch

Krzysztof Rutka (krzysztof.rutka@erlang-solutions.com)
Konrad Kaplita (konrad.kaplita@erlang-solutions.com)
Erlang Solutions Ltd.

Sandhya Narayan (snarayan@infoblox.com)
Stuart Bailey
Infoblox Inc., Santa Clara, CA.

LINC (**L**ink **I**s **N**ot **C**losed) is a completely new open-source switching platform available through flowforwarding.org, a community promoting free open source Apache 2 license implementations based on OpenFlow specifications. LINC has been developed to serve as a reference implementation that includes the full feature set of OpenFlow specifications. The primary goal with developing LINC was to quickly and cost effectively develop and evaluate the OpenFlow 1.2 and 1.3.1 and OF-Config 1.1 versions of the specifications to run on COTS platform, and do it well ahead of similar projects that use conventional tools. For this reason we picked Erlang, a functional language with its roots in telecom.

Erlang, developed by Ericsson, is a multi-purpose functional programming language that was designed from the ground up for writing scalable, fault-tolerant, distributed, non-stop, soft real-time applications. While a discussion on the many features of Erlang would be interesting, the following features contributed to the rapid design and development of LINC switch.

- Erlang OTP (Open Telecom Platform) is a large collection of libraries for Erlang that provide solutions for many common problems in networking and telecommunications systems. One example is the so-called supervision tree.
- Erlang's extremely convenient bit-manipulation capabilities make it well suited for protocol handling and low-level communications.
- Erlang has built-in support for creating and managing processes with the aim of simplifying concurrent programming.

Erlang is also well suited to exploit the new trend in many-core architectures that provide huge processing capabilities in COTS in a cost-effective way. Since our goal was to develop a software switch that runs on COTS, utilizing multiple cores for achieving higher performance is an important secondary goal as well.

We utilized a novel testing methodology of property-based testing to make our implementation robust. It enabled us to capture some interesting bugs, which would otherwise be hard to come across and identify.

LINC Architecture

LINC is a software switch that fully implements OpenFlow 1.2, OpenFlow 1.3.1 and OF-Config 1.1 versions of the OpenFlow specifications that are applicable to pure OpenFlow switches. The block diagram in Figure 1 shows its main software blocks: OpenFlow Capable Switch, OpenFlow protocol module and OF-Config module. These are designed to follow the Erlang OTP principles and are developed as separate applications where each application implements a certain well defined functionality, has clearly defined interface, dependencies and supervision tree.

The OpenFlow protocol is implemented in the of_protocol library application that defines internal OpenFlow protocol structures, data types and enumerations, provides encode and decode functions for OpenFlow protocol messages and validates their correctness.

The functionality of the OpenFlow Capable Switch is implemented in the linc library. It receives OF-Config commands and executes them in the OpenFlow Operational Context. It handles one or more OpenFlow Logical Switches that consist of the channel component, replaceable back-ends and common logical switch logic. The channel component is a communication layer between the OpenFlow Logical Switch and the OpenFlow Controllers. It handles TCP/TLS connections to the Controllers. It uses the of_protocol

library for encoding and decoding the OpenFlow Protocol messages. It passes parsed structures received from the OpenFlow Controller to the backend and forwards encoded messages from OpenFlow Switch to the Controllers.

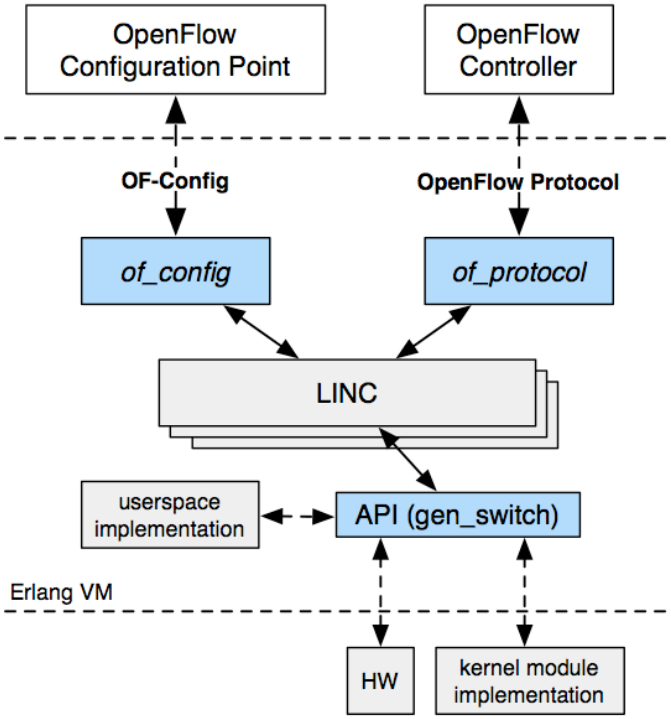

Figure 1

Replaceable back-ends implement the actual logic for switching the packets. They manage flow tables, group table, ports etc. and reply to OpenFlow Protocol messages received from the Controller. Due to the use of a common API (gen_switch), LINC's Logical Switch can use any of the available back-ends. Common switch logic does not depend on the back-end. It handles switch configuration, manages the channel component and OpenFlow resources, such as ports and dispatches messages received from the Controller.

OF-Config protocol handling is implemented by the of_config library application that handles parsing, validation and generates an interpretation of the OF-Config messages received from an OpenFlow Configuration Point. This is output as a set of commands to the OpenFlow capable switch application (linc) to configure the OpenFlow Capable Switch (i.e. create an instance of the OpenFlow Logical Switch, associate an OpenFlow resource with a specific OpenFlow Logical Switch etc.).

In addition to these components LINC has a supervision tree for fault tolerance purposes in accordance with the OTP principles. A supervisor is responsible for starting, stopping and monitoring its child processes. The basic idea of a supervisor is that it should keep its child processes alive by restarting them when necessary.

Conclusion

The development of LINC switch began in February of 2012 and thanks to the efficiency of development in Erlang, it was alpha-released in June and was one of the first OpenFlow 1.2 switches to be demonstrated at the InterOp, Tokyo. Today it is the first switch to include an implementation of OF-Config with functionality of OpenFlow Capable switch and OpenFlow Logical switch and has been updated to support OpenFlow 1.3.1. So far, the effort required to design, implement and support the LINC project has been less than 4000 man-hours.

Now, that we have a full featured implementation as the foundation, we plan to work on its performance and uses cases. Since LINC backend has been currently implemented in user space, future plans include kernel space, multi-core and hardware implementations.

References

[1] http://www.erlang.org/
[2] http://www.flowforwarding.org/