Making LINCX the primary LINC version

Maxim Kharchenko, Cloudozer LLP

05/04/2014

1 Summary

- The LINC version based on LINCX should be transferred to LINC-Switch repo by March 1, 2014.
- The new capable/logical switch architecture should be postponed to ensure continuity.

2 Overview

LINCX is a version of the LINC switch with a completely new implementation of the fast path – performance-critical portions of the switch. LINCX shows performance orders of magnitude better than the old version. LINCX runs on the new Erlang platform (LING).

Currently there two versions of the LINC switch: the "Old LINC" that resides in the LINC-Switch[1] repo and LINCX kept in the lincx[2] repo. The decision has been made to merge these two versions. The new merged version – the "New LINC" – should be based mostly on the LINCX codebase.

The present plan discusses the transition from the Old LINC/LINCX to the New LINC.

3 The status of LINCX

The LINCX version started as a fork of the Old LINC at the end of 2013. The Old LINC code was reused whenever this did not compromise the performance. The implemenation of functions that belong to the fast path are not yet complete. Notably, the following functions are still lacking:

- Counters
- Queues
- Meters

[1] https://github.com/Flowforwarding/LINC-switch
[2] https://github.com/FlowForwarding/lincx

The most performance-sensitive actions, such as Set-Field, has been completely rewritten. Some packet-modifying actions, such as Push/Pop PBB, still use the old (slow) code.

The build/test/deploy harness should be revisited to account for the new target platform (LING).

In addition to these incremental changes there are bigger architectural issues discussed below.

4 The architecture must change

The Old LINC acts as an OpenFlow Capable Switch managing several logical switches.

An allegedly better architecture is to run a single logical switch per Xen domain and have a separate coordination entity – the capable switch – in Domain 0. Such architecture should ensure precise allocation of resources between logical switches and avoid certain bottlenecks in the Linux kernel.

The new architecture will require a new build and testing framework. The switch software will need to talk directly to the Xen toolstack for spawning and monitoring of logical switches.

The architecture should allow applying back pressure to remote queues, passing control over ports to other logical switches, etc. The right mechanism for this kind of interactions is still to be discovered.

The new architecture will obsolete most of the high-level management code of the LINC switch and thus should be postponed until after the transition to the New LINC. There should always be a working, compatible, and performant version available for download.

A related forward-looking activity is the development of the next version of the OpenFlow specification. The OpenFlow 2.0 can be explicitely based on Erlang and Xen.

5 The transition timeline

1. Prepare the first version of the New LINC (port any relevant changes, remove stale dependencies, update tutorials, deployment scripts)
2. Replace the version in the LINC-Switch repo – ***March 1, 2014***
3. Continue development according to the prioritized task list

6 Task summary – The shopping card

The task summary (Table 1) contains the initial list of tasks that can be undertaken in the course of the transition and thereafter. The list is maintained as the part of the repository. See docs/TASKS.md.

7 Appendix: the new source tree

The high-level changes to the source tree related to the transition are summarized below.

Table 1: The shopping card

Priority	Description	Effort
A	LINCX appliance for deployment to Atom black boxes	1wk
A	Remove stale dependencies, update Makefile	2wk
B	Prototype the fast version of queues	1wk
B	Port the Old LINC changes to LINCX	1wk
B	Implement remaining packet-modifying actions	1wk
B	Develop a strategy for exhaustive compatibility testing	1wk
B	Create a testing enviroment using hw NICs	1wk
B	Test the switch compatibility using Luxoft Twister	2wk
B	Create a bug-tracking system for the New LINC	1wk
B	Prototype counters using processes and BIFs	1wk
B	Prototype the fast version of meters	1wk
C	Envision and create the continuous integration environment	3wk
C	Develop a new capable/logical switch architecture	2wk

Table 2: apps/

Application	Note
linc	see below
linc_us3	drop
linc_us4	drop
linc_max	see below

Table 3: apps/linc/

File	SLOC	Note
linc_logic.erl	500	30% rework
linc_ofconfig.erl	1418	20% rework
(other files)	(small)	

Table 4: apps/linc_max

File	SLOC	Note
linc_max_convert.erl	142	old, drop
linc_max_demo.erl	165	new
linc_max.erl	458	new
linc_max_fast_actions.erl	521	new
linc_max_fast_actions_tests.erl	330	new
linc_max_fast_path.erl	106	new
linc_max_flow.erl	1325	old, drop
linc_max_generator.erl	652	new
linc_max_generator_tests.erl	358	new
linc_max_meter_sup.erl	57	old
linc_max_packet.erl	311	old, drop
linc_max_port.erl	302	old, drop
linc_max_port_native.erl	38	old, drop
linc_max_preparser.erl	803	new
linc_max_preparser_tests.erl	268	new
linc_max_splicer.erl	469	new
linc_max_splicer_tests.erl	217	new
linc_max_sup.erl	63	old, drop

Table 5: deps/

Application	Note
eenum	TBD
enetconf	TBD
epcap	drop
lager	in use
ling_builder	drop, use openling
meck	drop, bloats images
of_config	in use
of_protocol	in use
pkt	keep and avoid
procket	drop
sync	drop, not applicable
tunctl	drop