Paper Evaluation, The Click Modular Router

Jiani Jiang <jianij@kth.se>

1. Paper summary

Routers are increasingly expected to do more than route packets. Unfortunately, most routers have closed, static, and inflexible designs. Furthermore, it is difficult for network administrations and third party software vendors to extend a router with new functions. This paper presents Click, a flexible, modular software architecture for creating routers. The components are packet processing modules called elements. The basic element interface is narrow, consisting mostly of functions for initialization and packet handoff, but elements can extend it to support other functions.

Click uses Push and Pull Connections to realize the basic function, transporting the packets. On a push connection, packets start at the source element and are passed downstream to the destination element. This corresponds to the way packets move through most software routers. On a pull connection, in contrast, the destination element initiates packet transfer: it asks the source element to return a packet, or a null pointer if no packet is available.

For the evaluation part, they evaluate Click's performance for IP routing and for several extended configurations. They evaluate the forwarding rate of Click, Polling Linux, and Linux. Click performs much better than the other two systems. And they also find the price of Click is lower and the speed of Click is faster.

2. Top 3 contributions

1. They present Click, a flexible, modular software architecture for creating routers. Click can specify the interactions of different functions, and it is easy to extend the router with new function.

2. We have implemented Click on general-purpose PC hardware as an extension to the Linux kernel. It can fit for the now working hardware with any special support.

3. Click is modular, it uses a lot of simple elements to realize the complex functions, making it easy to design the network.

3. Problems

1. Handling interrupts will be a problem, it may take 70-80% of overall time to handle the interrupt. If they get rid of interrupts by polling, the system performance may not look so good.

2. Not all modules are input/output driven, for example, some are timer driver, it is hard to use different kinds of modules at the same time and it is hard to guarantee schedule