Introduction to Bare-metal Networking

Uwe Dahlmann
IU GlobalNOC

Supported by NSF EAGER grant #1535522
Agenda

• What is it?

• What opportunities does it provide?

• Use Examples – Proof of Concepts

• Our Lab
What is a switch?

Example Switch Architecture (Openswitch)
Traditional Network devices

- A switch or router provided by a vendor
- Chipset self developed, or increasingly merchant silicon based
What is “WhiteBoxSwitching”?

- Not primarily a cheaper way of doing things
- Disaggregation is the keyword!
  - Access components / layers in the device that you previously had no access to.
  - Use these to build smarter solutions for your problem.
Why do it?

• Integrate your network devices better into the IT infrastructure
  • Chef, Puppet, Ansible, Nagios,
  • Often Linux based, so the known tools are available

• Many new interfaces to Network devices
  • Asic level APIs
  • Linux software environment
  • Rest APIs to network applications
Old Way of interacting

- Full Stack vendor implementation
  - CLI
  - SNMP
  - Netconf
  - Proprietary tools
Components / Layers – L1

- ASICs
  - Merchant silicon
  - Most focused on DC, but first large buffer chipsets available.
  - Differentiation happens now in the software

- Very protective in regards to everything relating to their ASIC design.

- Broadcom, Cavium, Frescale, Centec, ..

This leads to an additional software layer between their native API and the user exposed API
Flexible ASICs and P4

- Inflexible hardware pipelines are a problem for SDN, for example OpenFlow

- Efforts to introduce protocol independent, programmeable hardware are underway
  - There is an overhead associated!

- P4 developed as language to program flexible chipsets (P4.org)
  - P4 gets compiled and loaded onto the chipset at startup, then provides the pipeline configured.
  - At runtime, other protocols populate the pipeline
Flexible ASICs and P4

```c
header_type ethernet_t
{
  fields
  {
    dstAddr : 48; srcAddr : 48; etherType : 16; 
  }
}

parser parse_ethernet
{
  extract(ethernet);
  return select(latest.etherType)
  {
    0x8100 : parse_vlan;
    0x800 : parse_ipv4;
    0x86DD : parse_ipv6;
  }
}
```
Components / Layers – L2

- The bare metal
  - Reference designs
  - Different NOS can be installed
  - CPU and RAM need to be capable of supporting this
- They work with NOS providers (providing drivers for chipsets, providing hardware profiles)
  - Implement support for ONIE booting.
  - Accton / Edgecore, Agema / DNI, Celestica, Centec, Dell, HP, Mellanox, Penguin, Pica8, Quanta
Components / Layers – L3

- The Bootloader: ONIE (Open Network Install Environment)
- Maintained by the OpenComputeProject (OCP)
- Defacto standard boot loader for bare metal switches, allows installs via USB or DHCP/HTTP
- Supported by:
  - Hardware: Agema, Alphanetworks, Broadcom, Celestica, Centec, Dell, Edge-Core, HP, Interface Masters, Inventec, Juniper, Mellanox, Penguin Computing and Quanta | Software: Big Switch, Broadcom, Cumulus, Mellanox, OpenNetworkLinux
- ONIE now has over 60 platforms, 3 architectures, and 14 vendors.
Open Compute Project

• “The Open Compute Networking Project is creating a set of technologies that are disaggregated and fully open, allowing for rapid innovation in the network space. We aim to facilitate the development of network hardware and software – together with trusted project validation and testing – in a truly open and collaborative community environment.”

• They officially recognize hard and software that follows their guidelines / specifications, and publish this in a list.

• Accepted / under review Hardware:
  • Accton, Alpha Networks, Mellanox, Facebook Wedge, Inventec Broadcom/Interface Masters

• Accepted Software:
  • Open Network Install Environment, Open Network Linux, Switch Abstraction Interface
OpenComputeProject - Networking

- Monolithic Switch
- Software
  - Configuration Management
  - Statistics Packages
  - Environmental Handling
- Hardware
  - "Group Hug" Microserver
  - Modular Enclosure
  - Network Protocol Handling & Control Logic
  - Thrift API Layer to Switch ASIC

EXISTING TECHNOLOGY OCP STAGE I OCP STAGE II

Global NOC
Global Research Network Operations Center
Components / Layers L4

• The drivers / SDK / Chipset APIs

  • Traditionally: NDAs in place, Code using these can not be shared
  
  • To open this up, they created a middleware API that exposes an open, published Interface for developers.

• Examples:

  • OpenNSL, OF-DPA (Broadcom); OpenEthernet (Cavium, Mellanox); SAI (Open standard supported by OCP, Facebook, Microsoft)

• This allows you to write your code directly against the chipset, get more granular control and better internal monitoring information
Switch Abstraction Interface (SAI) is a standardized API that allows network hardware vendors to develop innovative hardware architectures keeping the programming interface consistent.

SAI helps easily consume new hardware by running the same application stack on all the hardware, enabled by a simple, consistent programming interface.

Sample APIs:

- Access Control Lists (ACL); Equal Cost Multi Path (ECMP); Forwarding Data Base (FDB, MAC address table); Host Interface; Neighbor database, Next hop and next hop groups; Port management; Quality of Service (QoS); Route, router, and router interfaces.
Components / Layers – L5

• **OpenNetworkLinux** (https://opennetlinux.org/)
  • A pure operating system for network devices
  • No forwarding Apps included
    • But Agents available like Quagga, BIRD, Facebook FBOSS, Microsoft Azure SONIC
  • Provides drivers for chipsets
Components / Layers – L5

- Applications
  - Your OpenNSL App HERE
  - Facebook FBOSS
  - Your SAI App HERE
  - Indigo OpenFlow Agent
  - Your OFDPA App HERE
  - Open Route Cache

- Platform
  - Open Network Linux Platform Abstraction Layer
    - Platform Specific Drivers Including: Optics
  - ONL Linux Kernel
    - Includes extra drivers: I2C, MUX, mngt Ethernet, etc.
  - Installer (ONIE)
    - CPU (PowerPC, x86)
  - Hardware
    - Misc Hardware (Fans, LED controllers, SFP, sensors, power supplies)
    - Packet Forwarding Chip (ASIC)

- Hardware
  - OCP Switch Hardware: Facebook Wedge, IM Niagara, Accton 6712, 7712, Dell S6000-ON, Quanta LY6, etc.
Components / Layers 5+6

• Full NetworkOS implementation with protocol, management and forwarding applications (CLI, OSPF, BGP, …)

• OS is open, standard Linux, software can be installed via standard means, and self developed code can run (in a standard Linux environment)

• Support for ONIE

• Examples:
  • Openswitch (OpenSource), Cumulus, OcNOS, PicOS
Several Vendors are offering fully vertically integrated solutions

- Their switch firmware only works with their controller
  - Controller often does bootstrapping, firmware installation and management on the device too
- This helps with OpenFlow pipeline problems
  - Pipeline not standardized, not flexible on Hardware; Only partial support for OpenFlow functions; Hard to build an application on top
- In the bare-metal model, the Controller side application logic can be ideally supported by a matching switch side agent, and changes can be implemented on both sides simultaneously.
Closed Application Specific Solutions

- **Examples:**
  - Bigswitch (BigTap, BigMon, Switchlight)
  - Gigamon - GigaVueOS
  - Pluribus - Open Netvisor Linux.
Other approaches - switchdev

- The Ethernet switch device driver model (switchdev) is an in-kernel driver model for switch devices which offload the forwarding (data) plane from the kernel.

- Effort to expand Linux using switchdev as a general solution for hardware switch chips and to make a concerted effort to break the binary-blob "SDK" stranglehold.

Other approaches - switchdev

- Each physical port on a device is registered with the kernel as a net_device, as is done for existing network interface cards (NICs).

- Ports can be:
  - bonded or bridged, tunneled or divided into virtual LANs (VLANs) using the existing tools (such as bridge, ip, and iproute2).
  - The advantage of a switchdev driver is that such switching constructs can be offloaded to the switch hardware. As such, the driver mirrors each entry in the forwarding database (FDB) down to the hardware, and monitors for changes.
Use Cases – LinkedIn – Project Falco

• **Problem: microbursts**

  • serious latency problem with applications inside their data centers.
  
  • very difficult to meet the demands of their applications when network routers and switches are beholden to commercial vendors, who are in control of features and fixing bugs.
  
  • The vendors they buy switches from do not expose the telemetry information nor do they provide read/write access to the third party merchant silicon.
Use Cases – LinkedIn – Project Falco

• **Problems:**

  • Bugs in software that could not be addressed in a timely manner
  
  • Software features on switches that were not needed in our data center environment. Exacerbating the problem was that we also had to deal with any bugs related to those features.
  
  • Lack of Linux based platform for automation tools, e.g. Chef/Puppet/CFEngine
  
  • Out-of-date monitoring and logging software, i.e. less reliance on SNMP
  
  • High cost of scaling the software license and support.

• [https://engineering.linkedin.com/blog/2016/02/falco-decoupling-switching-hardware-and-software-pigeon](https://engineering.linkedin.com/blog/2016/02/falco-decoupling-switching-hardware-and-software-pigeon)
Use Cases – LinkedIn – Project Falco

• Solution: Design our own network stack

  • Run our merchant silicon of choice on any hardware platform
  • Run some of the same infrastructure software and tools we use on our application servers on the switching platform, for example, telemetry, alerting, Kafka, logging, and security
  • Respond quickly to requirements and change
  • Advance DevOps operations such that switches are run like servers and share a single automation and operational platform
  • Limitless programmability options, Feature velocity, Faster, better innovation cycle

• Greater control of hardware and software costs
Use Cases – Spotify – Project SIR (SDN Internet Router)

• **Problem: Routers holding full routing table are**
  - Very power hungry. 12,000W per device or 33W per 10G port.
  - Big. 10 rack units for 288 ports.
  - Expensive. List price can be around half a million U.S. dollars.

• In each data center they only need to support ~32,000 routes.
  - Device can be as small as a rack unit.
  - 72 x 10G ports; 262 W
  - ~ 30,000 U.S. dollars
Use Cases – Spotify – Project SIR (SDN Internet Router)

• Stage 1 – Proving the hypothesis – Analyze routes per DC

• Stage 2 – Designing the prototype – Openflow <-> BGP Selective Route Download (SRD)

• Stage 3 – Choosing a switch – Vendor support for unusual workload

• Stage 4 – Building the prototype – Stipping down the NOS

• Stage 5 – Refactoring the prototype – Optimize for real world load

• Stage 6 – Testing the BGP implementation of our vendor - – Optimize for real world load

• Stage 7 – Going Live
Use Cases – Spotify – Project SIR (SDN Internet Router)

• Conclusions:
  
  • SIR did not only enable us to peer in several locations without having to spend money on very expensive equipment, we got also other benefits.
  
  • The API that SIR provides has proven to be useful for figuring out where to send users to in order to improve latency, where and who to peer with and improve our global routing.
  
  • It also gave us some vendor independence, SIR is compatible with any platform that supports sflow and SRD.
  
Use Cases - SciPAss

- **SCIPASS**: IDS LOAD BALANCER & SCIENCE DMZ

- SciPass is an OpenFlow application designed to help network security scale to 100Gbps.

- SciPass turns an OpenFlow switch into an IDS load balancer capable of considering sensor load in its balancing decisions.

- When operating in Science DMZ mode, SciPass uses Bro to detect "good" data transfers and programs bypass rules to avoid forwarding through institutional firewalls, improving transfer performance and reducing load on IT infrastructure.

- We verified Dell 4048-ON with Dell OS9 for Scipass, working on other platforms with the vendors.
Our work in the lab

• Verification of all components. (ONIE, NOSs, Applications, PoC)

• Hardware List:

  • **Edgecore AS-4610**: 48 x 10/100/1000BASE-T RJ45 / 4 x 10GbE
    ; **Edgecore AS-5712**: 48 x SFP+ switch ports, supporting 10 GbE (DAC, 10GBASE-SR/LR) or 1 GbE (1000BASE-T/SX/LX).; **Edgecore AS-6712**: 32 x 40GbE QSFP; 40GBASE-CR4 DAC | 40GBASE-CR4 to 4x10GBASE-CR DAC | 40GBASE-SR4 to 4 x 10GBASE-SR; **Agema: AG-7448CU**: 48 x 10GbE SFP+, 4x 40GbE QSFP uplinks; **Dell: S4048-ON**: 48-port 10GbE switch with six 40GbE uplink; **HP: Altoline 6940**: 32 x 40 GbE QSFP+ (or 8 x 40GbE and 96 x 10 GbE); **Mellanox: Spectrum 2700**: 32 40/56/100GbE ports, Up to 64 10/25GbE ports, up to 64 50GbE ports

• Ready to go testbeds for use cases from the community (Servers, Switches, Traffic generation)
Other related work: 100Gig NICs

- Used for TCP flow analysis using:
  - ARGUS
  - TCPstat
- Pushing traffic to the application works, but:
  - How do we balance the incoming traffic over multiple CPUs?
  - Hashing on the NIC, exposure as multiple streams to multiple application instances.
Thanks!

• Questions?

• IU Bare-Metal Testbed: http://globalnoc.iu.edu/whitebox/index.html