A bit of history…

• The role of a switch is to connect multiple LAN segments
• Operates on Layer 2
• Supports a single operation: Forwarding
• If you want to do more:
  • Layer 3 is handled by the software
  • Protocol processing is handled by another device (NPU / PPU)
• Valid until mid-2000’s
  • E.g. 2002’s “state of the art” Broadcom Strata XGS, 8x10GE
Basic components of an IP router (originally)

Control Plane

- Management & CLI
- Routing Protocols
- Routing Table

Data Plane

- Forwarding Table
- Switching
- Queuing

Software

Hardware

per-packet processing
A bit of recent history…

- Mid-2000’s to start-2010’s:
  - Fixed function switches
  - Integration of functions: same trend as with CPUs
    - Why use NPU + Switch if you can use just a switch?
  - For limited applications
A bit of recent history…

- Mid-2000’s to start-2010’s:
  - Fixed function switches
  - Supporting multiple (pre-defined) protocols
    - E.g. Layer 3 switching
  - Fixed pipeline (example only):

  - Fixed header parser
  - MAC lookup
    - Exact match
  - IPv4 lookup
    - Longest prefix match
  - Header editing
A bit of recent history…

• Start-2010’s – Recent years:
  • Partly / fully flexible switches
  • Support *many* protocols
  • Flexibility in selecting the protocols, memories used, header size,…
Programmable network devices

- Partly / fully programmable
  - Mostly focused on the header processing
  - But starting to attend also to queueing / switching / TM / …
- Support ANY protocol
- Pipeline is “programmable”
  - But within given resource limitations
Programmable network devices

Advantages:

• New Features – Add new protocols
• Reduce device complexity – e.g., Implement only required protocols.
• Flexible use of resources
• SW style development – better innovation, fix data-plane bugs in the field
Reconfigurable Match-Action Model

- RMT – Reconfigurable Match-Action Model

Programmer declares the headers that should be recognized and their order in the packet.

Programmer defines the tables and the exact processing algorithm.

Programmer declares how the output packet will look on the wire.

source: p4.org
Match-Action Unit

- Can support multiple simultaneous lookups and actions

Sequential Execution (match dependency)

Parallel Execution (No dependency)

Staggered Execution (action dependency)
How to programme a network device?

Requires:

• Programming language

• Compilers

• Architecture
  • Underlying hardware support

• We will discuss one popular option, but there are more
- www.p4.org
- A declarative language
- Telling forwarding-plane devices (switches, NICs, …) how to process packets.
Example: P4 on NetFPGA (P4-NetFPGA)

P4 Program

Xilinx P4_{16} Compiler

Xilinx SDNet Tools

SimpleSumeSwitch Architecture

NetFPGA Reference Switch

source: netfpga.org / p4.org
SimpleSumeSwitch Architecture Model for SUME Target

P4 used to describe parser, match-action pipeline, and deparser

source: netfpga.org / p4.org
P4 PSA: Portable Switch Architecture

- Composability
  - Example: Multiple functions in a single pipeline
- Portability
  - Example: Apply a function consistently across a network
- Comparability
  - Example: Compare functions implementation, A vs. B

Pipeline

Parser → Checksum validate → Ingress match-action → Packet Buffer and Replication → Egress match-action → Checksum update → Deparser → Buffer Queueing Engine

Externs

Checksum, Register, Counter, Meter, Action Profile, Action Selector, Random

source: p4.org
Exerns

• A functionality that is *not* implemented in P4
  • Can be implemented in any language!

• Provided with an interface that can interact with / be invoked by P4 programs

• Target Specific

• Examples: checksum, hash, timestamp, r/w memory, …
P4 – Examples Use Cases

- Network telemetry (INT)
- New protocols (e.g., NDP)
- Layer 4 load balancing (e.g., SilkRoad)
- In Network Caching (e.g., NetCache) – $\times 10$ throughput
- Consensus Protocols (e.g., P4xos) – $\times 10,000$ throughput
- Tic-Tac-Toe
In Network Computing

The execution of native host applications within the network using standard network devices.
The execution of native host applications within the network using standard network devices

In Network Computing
In Network Computing

- Idea: move services and applications from the host to the network
- Somewhat similar terms:
  - Network as a Service (NaaS)
  - Hardware acceleration (but network specific)
- Implementations:
  - Smart NICs
  - Programmable Switches
- Different platforms support different languages
In Network Computing - Examples

• Machine learning
• Graph processing
• Key-value store (e.g., memcached)
• Security (e.g., DDoS detection)
• Big data analytics
• Stream processing

• But nothing is for free (cost, power, space, …)
Classic Network Interface Cards (NIC):

- Get packets from the network to the host
- Get packets from the host to the network
- DMA – manage getting packets to/from host over the interconnect
  - Not trivial! (Lecture 5)
- Manage queues
  - Mostly toward the host
- ….
SmartNICs

Offload host functionality to the NIC:

- Virtualization
- Checksum
- TCP segmentation
- IPSec, MACSec

Even in “simple” NICs as X520
Offload host functionality to the NIC:

- Programmable data planes
- Data transport acceleration
  - NVMe, RoCE, DPDK…
- Security
- Network function virtualization (NFV)
- Application acceleration

More SmartNICs!
SmartNICs Models

- ASIC based (e.g., Intel, Netronome, Solarflare)
- FPGA based (e.g., Microsoft, Exablaze)
- ASIC + FPGA based (inline or not, e.g., Mellanox)
- SoC (ASIC w/ CPU) based (e.g., Mellanox)

Everyone are doing SmartNICs today!

Evolution, not a revolution