SNABB

A TOOLKIT FOR USER-SPACE NETWORKING

TOC

- What is Snabb?
- How it works?
- How to get started?

WHAT IS SNABB?

Toolkit for developing network functions in user-space

(mostly aimed for high-performance networking)

WHAT IS A NETWORK FUNCTION?

- A program that manipulates network traffic
- Basic operations: *read*, *forward*, *drop*, *modify*, *create*...
- Combining these primitives we can build any network function

EXAMPLES

- **Firewall**: read incoming packets, compare to table of rules and execute an action(*forward* or *drop*)
- **NAT**: read incoming packets, modify headers and forward packet
- **Tunelling**: read incoming packets, create a new packet, embed packet into new one and send it

HIGH-PERFORMANCE NETWORKING

OFF-THE-SHELL ROUTER

- Increasing improvement of commodity hardware: 10Gbps NICs at very affordable prices
- High-performance equipment is still very expensive
- **Idea**: build an analog high-performance router using commodity hardware
- What software to put into this hardware?

WHAT ABOUT LINUX?

- General-purpose operating system
- NF is divided in 2 lands: user-space and kernel-space
- Conclusion: Processing a packet has an inheritent cost (the cost of the OS)

HIGH-PERFORMANCE NETWORKING

- NIC: 10Gbps; Packet-size: 550-byte
- 1 packet every 440ns ((1/2,2M)*10^9)
- CPU: 2,5 Ghz => 1100 cycles per packet

HIGH-PERFORMANCE NETWORKING

- Packet-size: 64-byte
- 1 packet every 51 ns
- Lock/Unlock: 16ns; Cache-miss: 32 ns
- Link: "Improving Linux networking performance" (Jesper Brouer)

KERNEL BY-PASS

- Driver in user-space that talks directly to the hw
- Talk: How to write your own NIC device driver
- Other toolkits: DPDK (Intel), VPP/fd.io (Cisco)

INSIDE SNABB

SNABB

- Project started by Luke Gorrie
- Mostly developed in Lua
- Snabb means fast in Swedish

LUAJIT

- Just-in-time compiler for Lua
- Extremely fast virtual machine!!
- Very good integration with C thanks to FFI

HOW IT WORKS?

- A Snabb program (NF) is an **app graph**
- Apps are conected together via **links**
- Snabb engine processes the program in units called **breadths**

NETWORK FUNCTION

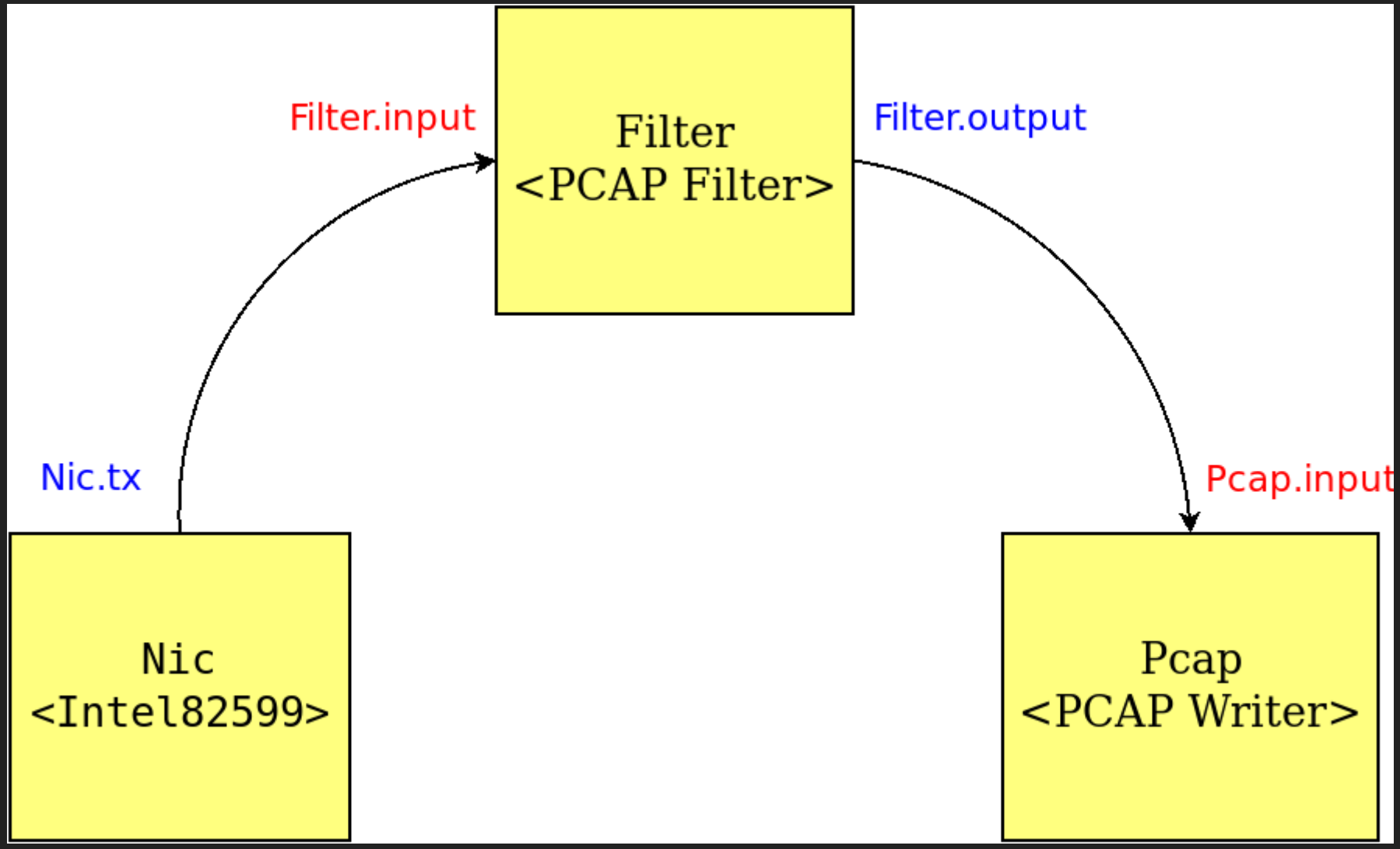
```
function run()
    local c = config.new()

    config.add(c, "nic", Intel82599, {
        pci = "0000:04:00.0"
    })
    config.add(c, "filter", PcapFilter, "src port 80")
    config.add(c, "pcap", Pcap.PcapWriter, "output.pcap")

    config.link(c, "nic.tx -> filter.input")
    config.link(c, "filter.output -> pcap.input")

    engine.configure(c)
    engine.main()
```

NF: APP GRAPH


Filter.input
Filter
<PCAP Filter>
Filter.output
Nic.tx
Pcap.input
Nic
<Intel82599>
Pcap
<PCAP Writer>

BREADTHS

- A *breadth* has two steps:
  - *inhale*, puts a batch of packets into the graph
  - *exhale*, processes those packets
- To **inhale**, the method *pull* of the apps is executed (if defined)
- To **exhale**, the method *push* of the apps is executed (if defined)

```
# Pull function of included Intel 82599 driver
function Intel82599:pull ()
  for i = 1, engine.pull_npackets do
    if not self.dev:can_receive() then break end
    local pkt = self.dev:receive()
    link.transmit(self.output.tx, pkt)
  end
end
```

```
# Push function of included PcapFilter
function PcapFilter:push ()
  while not link.empty(self.input.rx) do
    local p = link.receive(self.input.rx)
    if self.accept_fn(p.data, p.length) then
      link.transmit(self.output.tx, p)
    else
      packet.free(p)
    end
  end
end
```

PACKET PROCESSING

- Normally only one app of the app graph introduces packets into the graph
- The method *push* gives an opportunity to every app to do something with a packet

PACKETS

```
struct packet {
    uint16_t length;
    unsigned char data[10*1024];
};
```

LINKS

```
struct link {
    struct packet *packets[1024];
    // the next element to be read
    int read;
    // the next element to be written
    int write;
};
```

WHAT CAN YOU DO WITH SNABB?

Anything

(that has to do with a packet)

BUILT-IN CATALOG

- **Libraries**: protocols (ipv4, ipv6, tcp), checksum...
- **Apps**: drivers, filtering, load generators, sockets...
- **Programs**: L2VPN, Lisper, lwAFTR, IPFix, etc

LWAFTR

- Component of Lightweigth 4-over-6 (RFC 7596)
- IPv6 transition technology
- Deutsche Telekom's Terastream
- Blog post: Dive into lw4o6

SNABWALL

- L7 Firewall (also L3 & L4)
- Uses: libnDPI & Pflang
- Funded by NLNet Foundation
- Site: www.snabbwall.org

IPFIX

- IP Flow Information Export (RFC 7011)
- Format for exporting network flows
- Blog post: IPFix app for Snabb

DNS SERVICE DISCOVERY

- Discover Multicast devices in a LAN (RFC6763)
- Works on network interfaces
- Blog post: More practical Snabb

GETTING STARTED

```
$ git clone https://github.com/snabbco/snabb.git
$ cd snabb
$ make
```

LINKS

- Github: https://github.com/snabbco/snabb
- Slack: https://snabb.slack.com
- Guide: Official getting started guide
- Blog post: Snabb explained in less than 10 minutes

THANKS!

Questions?

- Email: Diego Pino <dpino@igalia.com>

igalia