NetTraffic-P4

Konzeption eines Switch-basierten Netzwerk-Lastgenerators in P4

11. Mai 2023

Hochschule für Technik und Wirtschaft

Thomas Scheffler

thomas.scheffler@htw-berlin.de

IFAF BERLIN
Institut für angewandte Forschung Berlin

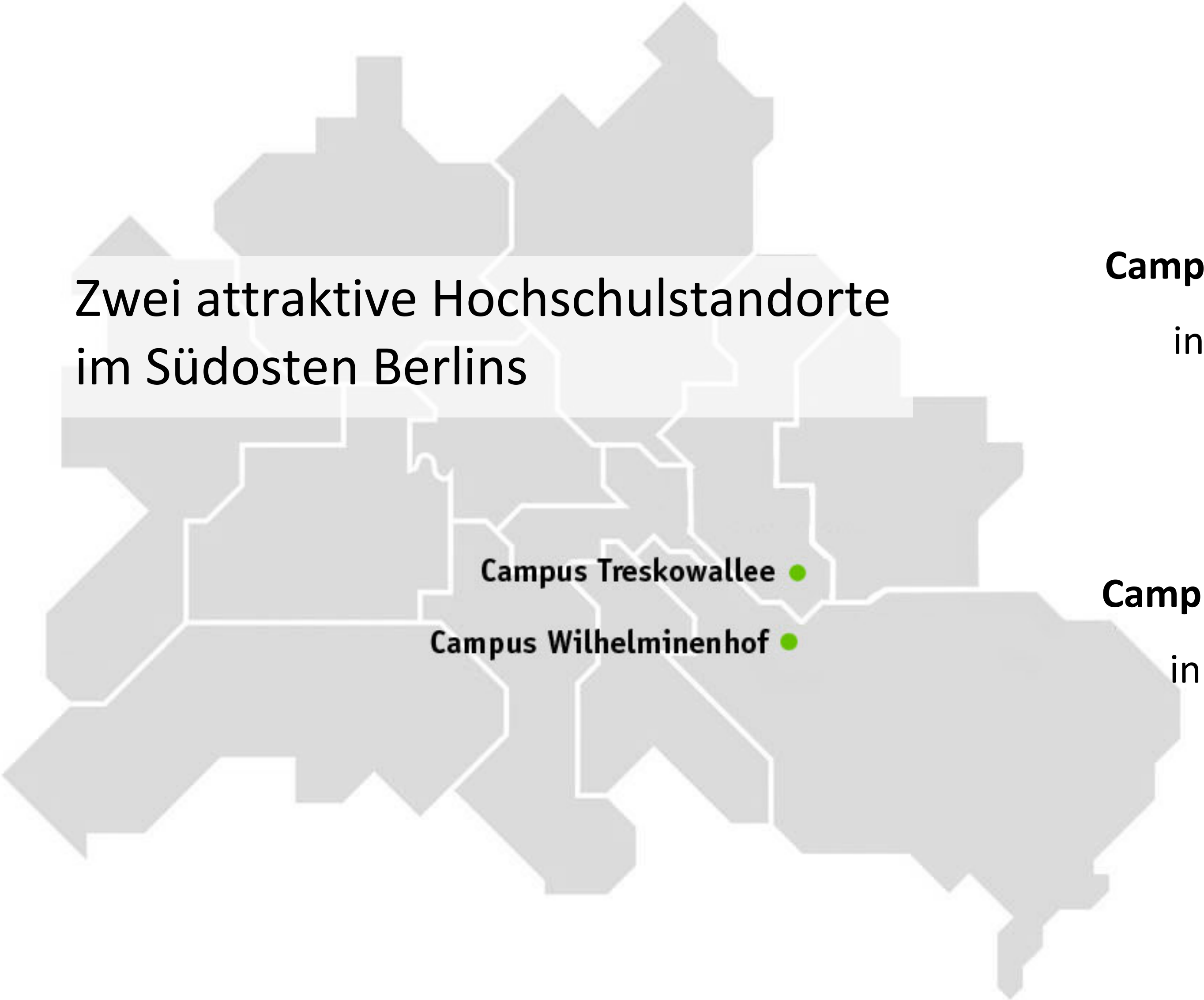
htw
Hochschule für Technik
und Wirtschaft Berlin
University of Applied Sciences

Hochschule für Technik und Wirtschaft – HTW Berlin

- Rund **70 Bachelor- und Masterstudiengänge**
- Ca. **14000 Studierende** im Präsenzstudium
- Größte Hochschule für angewandte Wissenschaften Berlins
- Das Studium ist kurz, kompakt und praxisorientiert
- Rankings bescheinigen eine hohe Ausbildungsqualität
- Die gut ausgestattete Bibliothek, ein modernes Rechenzentrum

Standorte der HTW Berlin

Zwei attraktive Hochschulstandorte im Südosten Berlins



Campus Treskowallee

in Karlshorst

Gebäude A auf dem Campus Treskowallee (HTW Berlin/Alexander Rentsch)

Campus Wilhelminenhof

in Oberschöneweide

Eingang zum Campus Wilhelminenhof (HTW Berlin/Alexander Rentsch)

Fachbereich 1: Energie und Information

Studiengänge:

Informations- und Kommunikationstechnik

Computer Engineering

Gesundheitselektronik

Mikrosystemtechnik

Elektrotechnik

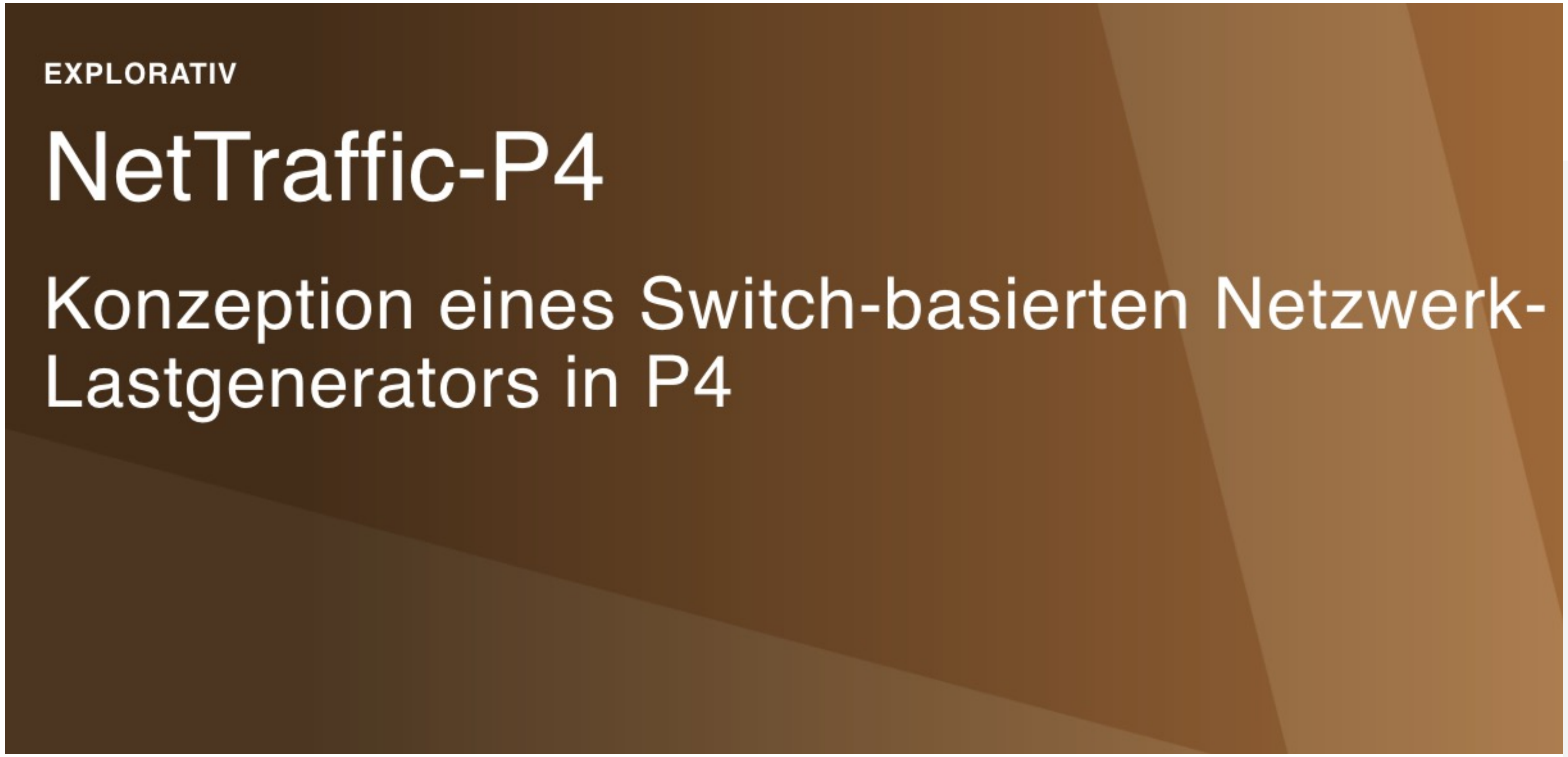
Gebäudeenergie- und -informationstechnik

Regenerative Energien

Prof. Dr. Thomas Scheffler
Computer Networks / Network Security

- Rechnernetze / Cisco Networking Academy
- Netzwerksicherheit & Kryptographie
- Programmieren in C und Python
- Internet of Things (IoT)

NetTraffic-P4



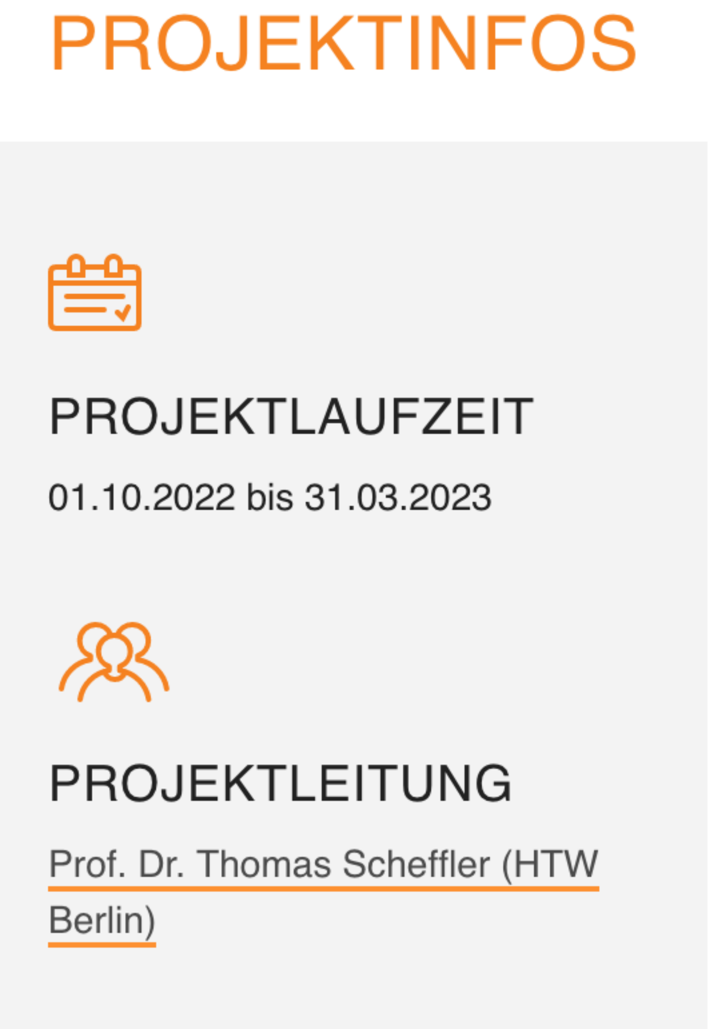
EXPLORATIV

NetTraffic-P4

Konzeption eines Switch-basierten Netzwerk-Lastgenerators in P4

Projektziel:

Einsatz von P4 für die Erzeugung von hochbitratigen (>100 Gbit/s) Test-Datenverkehr und Untersuchung der Implementierung eines Lastgenerators.



PROJEKTINFOS

PROJEKTLAUFZEIT

01.10.2022 bis 31.03.2023

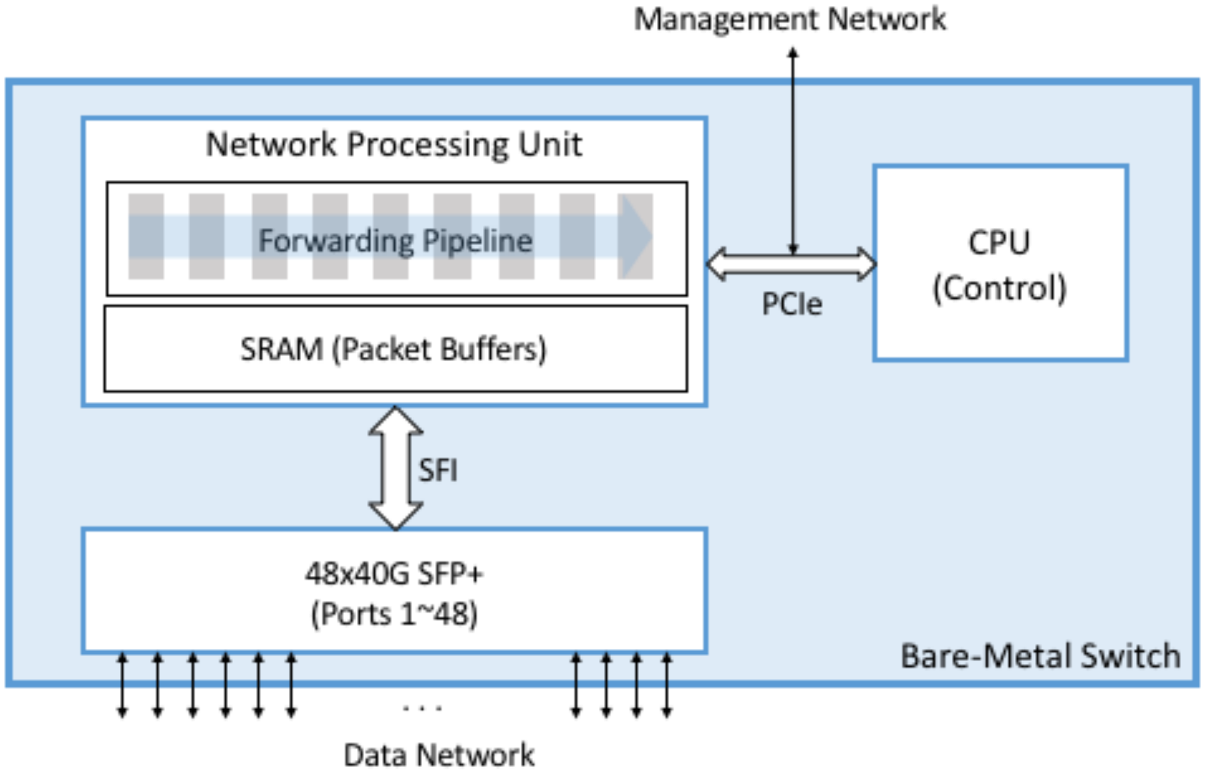
PROJEKTLEITUNG

Prof. Dr. Thomas Scheffler (HTW Berlin)

A short intro to P4 – what is a Switch?

- Building Blocks:
  - CPU with Switch OS (ARM or Intel, Linux)
  - NPU/ASIC for fast packet processing (this is where the magic happens)
  - Network ports

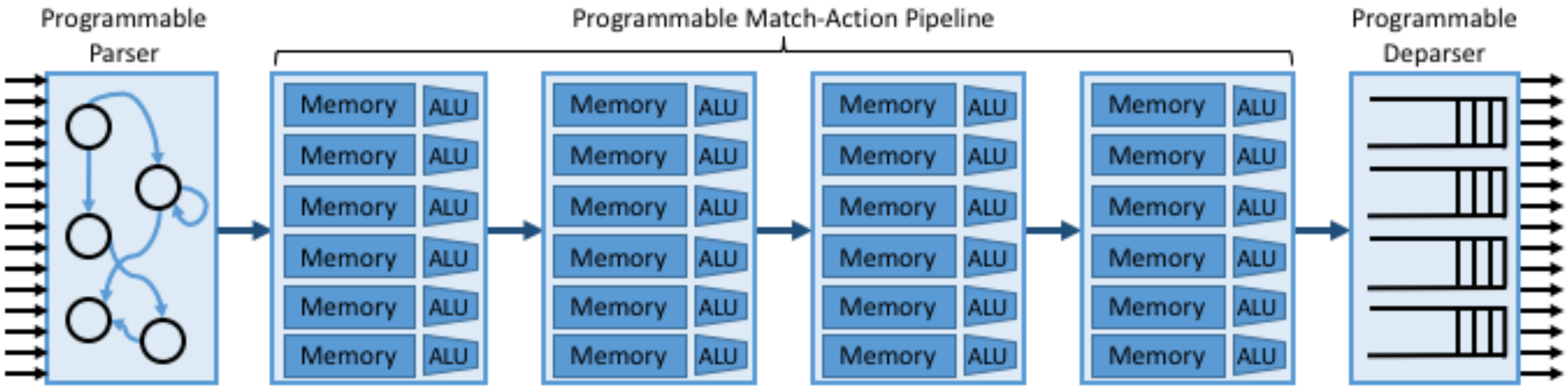
- Router / Switch basically use the same HW-base, difference in forwarding strategy, protocol support and control plane integration



Source: https://sdn.systemsapproach.org/

What if the NPU could be programmed?

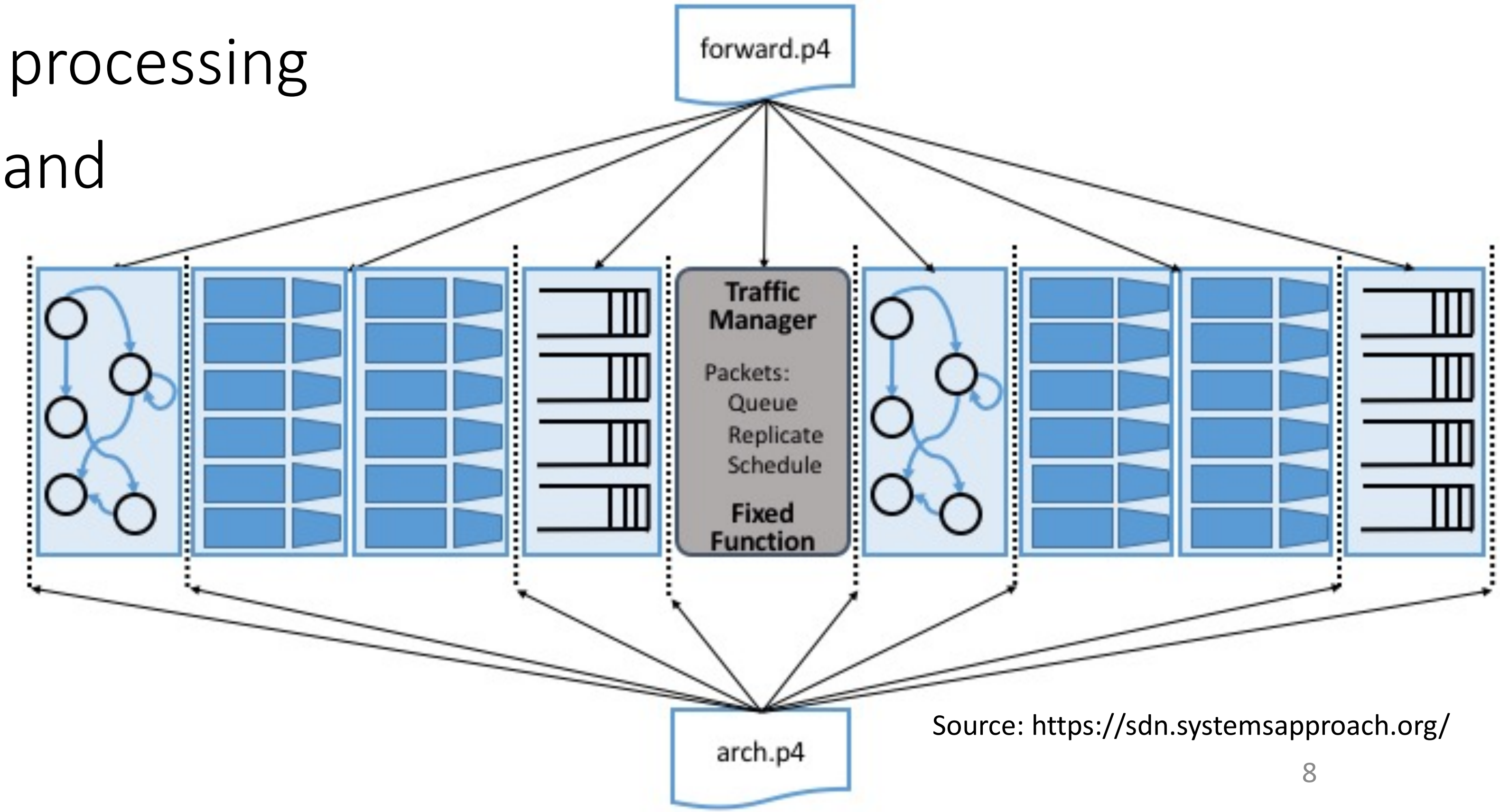
- Implement my own packet processing behavior
- Pull unused protocols, optimize limited resources (TCAMs, …)
- Do stuff directly in the network (fast)
- Monitor traffic
- …



Source: https://sdn.systemsapproach.org/

Programming a Forwarding Pipeline

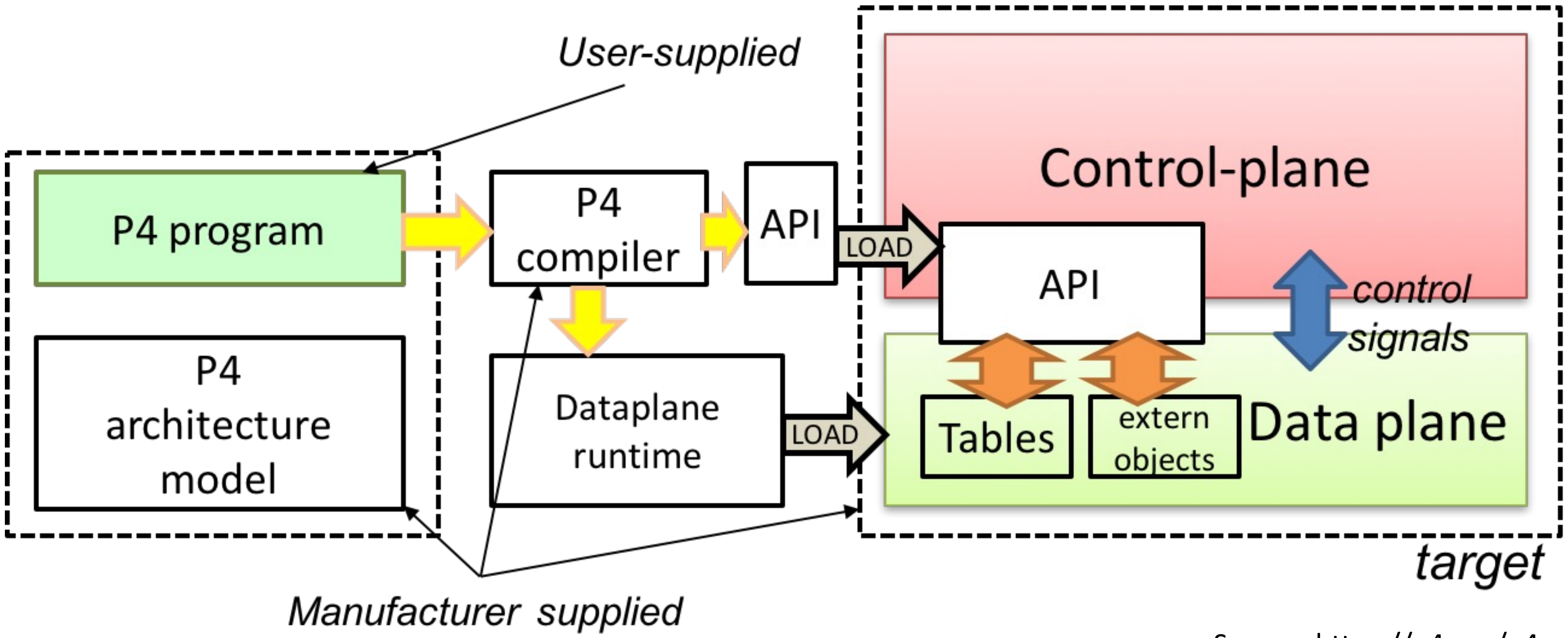
- P4 works over an HW-dependent architecture (bmv2, tofino, …)
- P4 programs a packet processing pipeline
- Minimal state and limited processing
- Interaction between data and control plane via tables
- Some functions are fixed
- …



Source: https://sdn.systemsapproach.org/

Programming a Forwarding Pipeline

- P4 compiler generates a data plane implementation as well as an API for control plane interaction (usually P4 runtime)

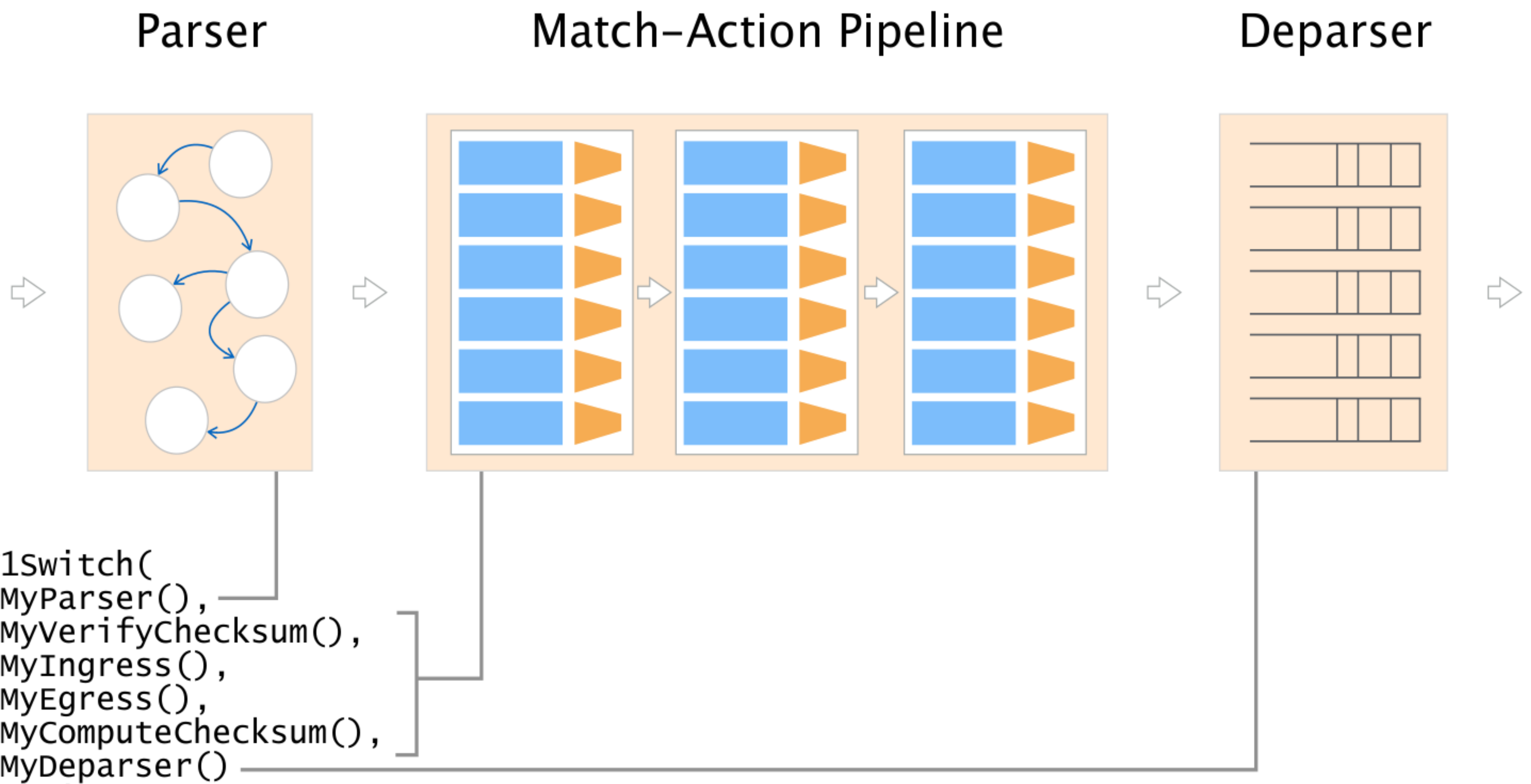


Source: https://p4.org/p4-spec/docs/P4-16-v1.2.2.html

P4$_{16}$ Language Overview

- High-level (C-Syntax), type-safe, memory-safe (no pointers)
- Bounded execution (no loops, no floating point, no division)
- Statically allocated (no malloc, no recursion)
- No threading support
- Language elements for:
  - Parsing packet headers (headers are just bits, program decides how to interpret)
  - Rewriting of packet headers
- Library of architecture specific functions (checksums, counters, meters, etc.)
- Architecture specific functions defined as Externs

P4 – Code Structure represents Pipeline



```
V1Switch(
 MyParser(),
 MyVerifyChecksum(),
 MyIngress(),
 MyEgress(),
 MyComputeChecksum(),
 MyDeparser()
) main;
```

Quelle: https://polybox.ethz.ch/index.php/s/Uc8XPmIzNlEasvn

P4$_{16}$ ''Hello World''

```
#include <core.p4>
#include <v1model.p4>
struct metadata {}
struct headers {}

parser MyParser(packet_in packet,
   out headers hdr,
   inout metadata meta,
   inout standard_metadata_t standard_metadata) {

    state start { transition accept; }
}

control MyVerifyChecksum(inout headers hdr, inout metadata
meta) {   apply {  }   }

control MyIngress(inout headers hdr,
   inout metadata meta,
   inout standard_metadata_t standard_metadata) {
   apply {
       if (standard_metadata.ingress_port == 1) {
           standard_metadata.egress_spec = 2;
       } else if (standard_metadata.ingress_port == 2) {
           standard_metadata.egress_spec = 1;
       }
    }
}
```

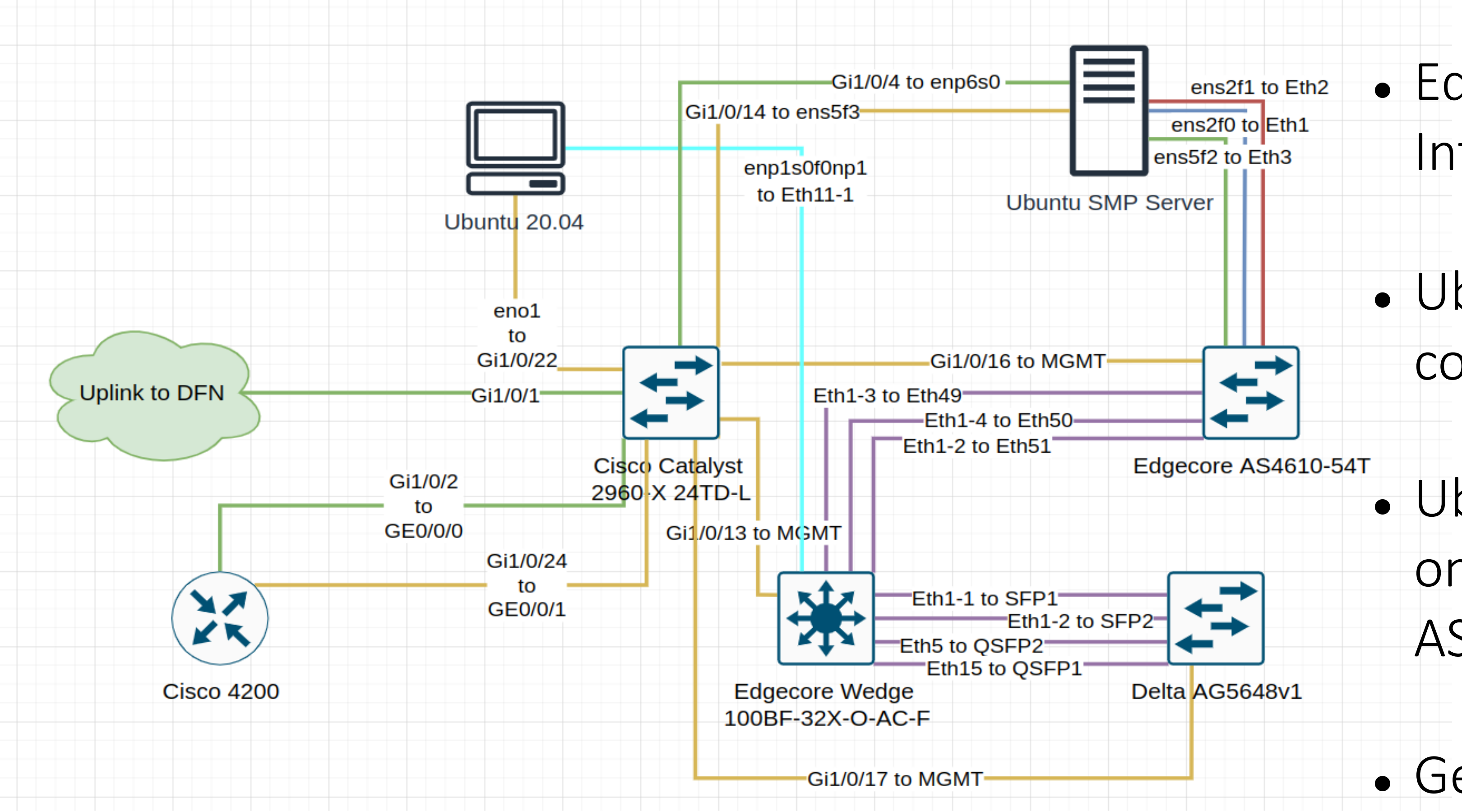
```
control MyEgress(inout headers hdr,
   inout metadata meta,
   inout standard_metadata_t standard_metadata) {
    apply {  }
}

control MyComputeChecksum(inout headers hdr, inout
metadata meta) {
     apply { }
}

control MyDeparser(packet_out packet, in headers hdr) {
    apply { }
}

V1Switch(
   MyParser(),
   MyVerifyChecksum(),
   MyIngress(),
   MyEgress(),
   MyComputeChecksum(),
   MyDeparser()
) main;
```

Lab Network



- Edgecore Wedge → P4 switch with Intel Tofino
- Ubuntu PC with 25Gbit/s fiber connection to P4 switch
- Ubuntu server testing interfaces on different VLANs on Edgecore AS4610
- Generate traffic with iPerf, Scapy, ICMP

Goals for NetTraffic-P4 Project

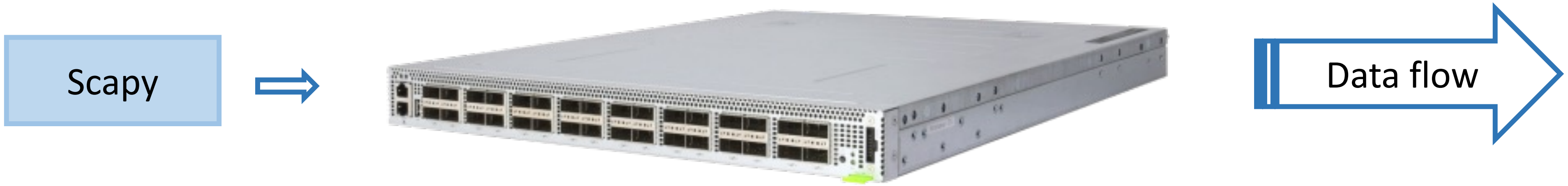
Explorative Project with high uncertainty of success,

- Determine the usefulness of P4 in the Network Lab
- Generation of flexible Test-Traffic: >= 100 Gbit/s
- Conducting Measurements: Intel Tofino supports precise timestamping (ns)
- OpenSource P4 code
- Technical Report & ’Getting Started with P4 Tutorial in German’

Intel Tofino-based P4 Switch

- 19“, Edgecore Wedge BF100-32X
- 32 x QSFP28 supporting 1 x 100 GbE/40 GbE, or 4 x 25 GbE/4 x 10 GbE/2 x 50 GbE
- Line-rate Layer 2 or Layer 3 forwarding of 3.2 Tbps full duplex

P4 Traffic Generator



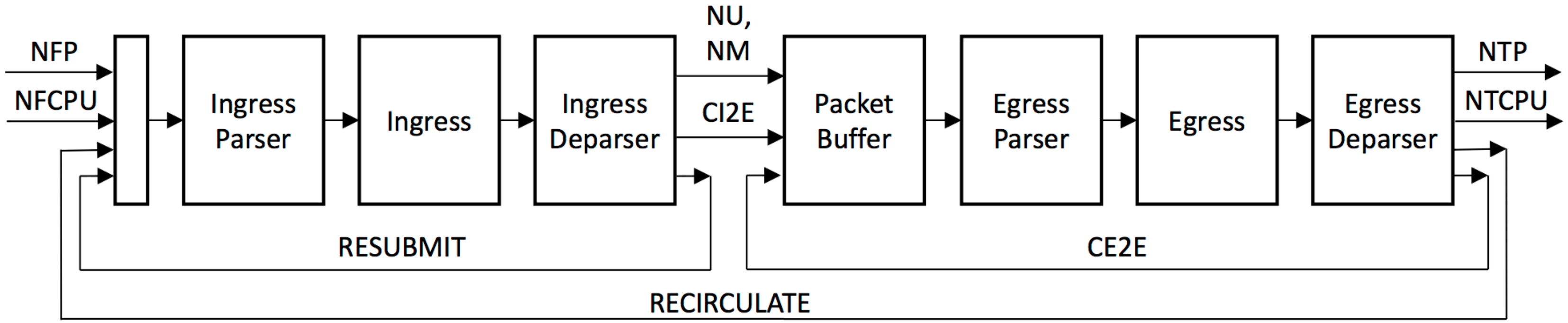
Scapy

Data flow

- Packet Cloning & Recirculation
- Timestamping
- Metering
- ...

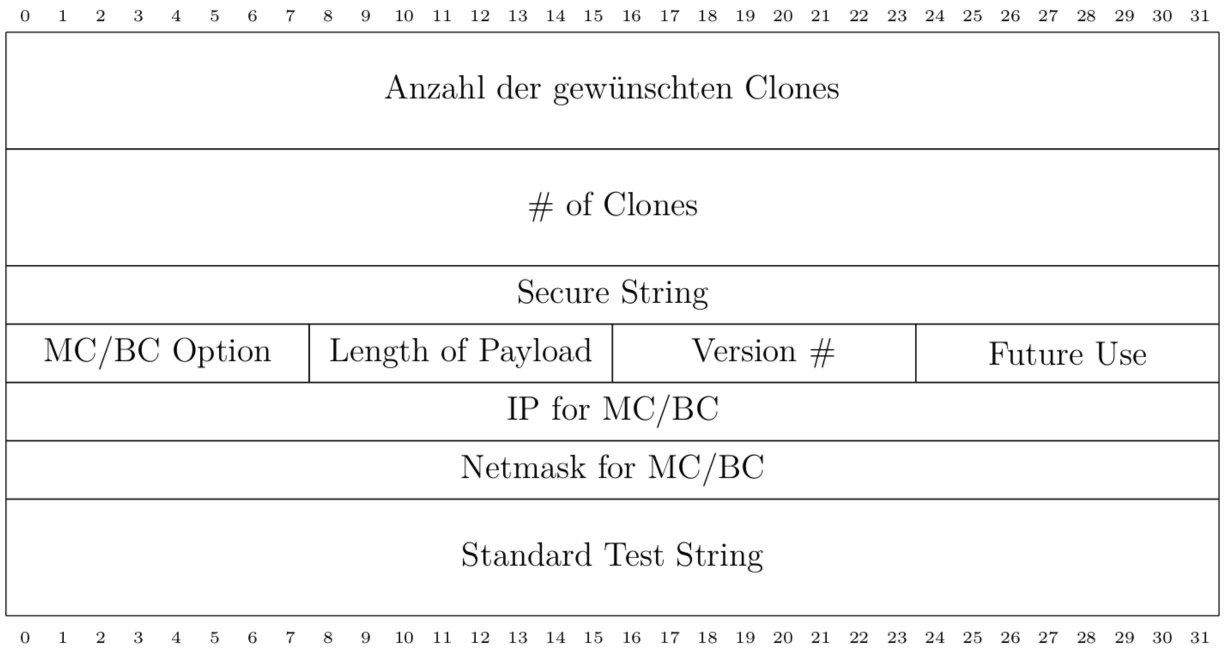
Traffic Generation: Clone-Packets in P4

- Decide which mirror-mechanism suits best → CI2E, CE2E, Resubmit, Recirculate



- Get it working on the hardware
  - Cloning is working on BMv2
- Testing
- Integrate with other parts of the research programm → INT, …
- Implement clone header in Wireshark → Dissectors in LUA

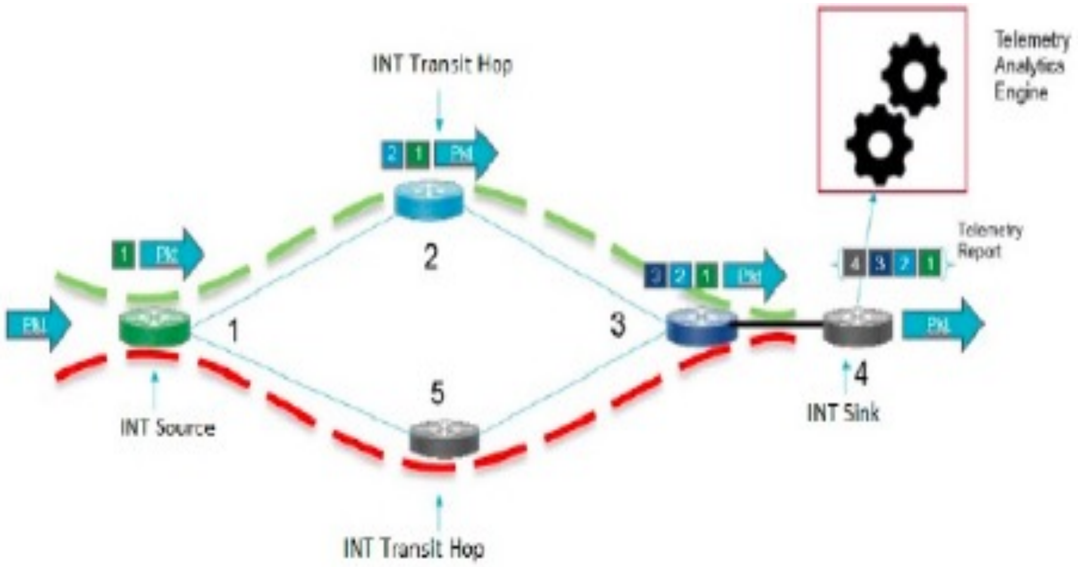
Clone-Header



Anzahl der gewünschten Clones			
# of Clones			
Secure String			
MC/BC Option	Length of Payload	Version #	Future Use
IP for MC/BC			
Netmask for MC/BC			
Standard Test String			

- Makes X clones of a packet → 1 field
- Compares wanted number of packets with real number of packet → “# of Clones” = Counter
- Relies on Ethernet and IPv4 → Clone = Layer4
- Secure String → Authorization
- MC/BC Option → enables Multi-/Broadcast
  - In combination with “IP” and “Netmask for MC/BC”

IN-band Network Telemetry



Goals: Large-Scale Network Monitoring **and** Network Performance Monitoring

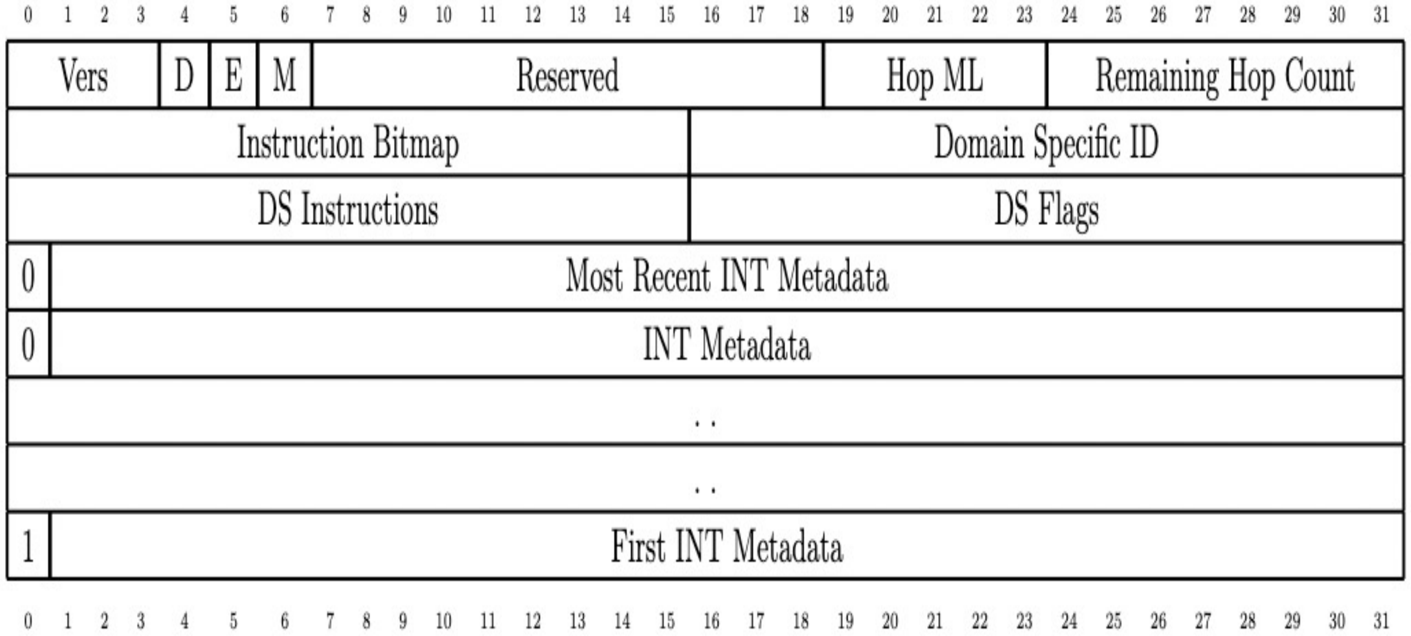
Components/ Functions of the Framework:

INT Source: Adding INT Header with instructions and Informations to packet

INT Transit Hop: Appending further information in the packet

INT Sink: Sending INT Header to a monitoring system and packet to host

INT Header Format



0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Vers	D	E	M	Reserved	Hop ML	Remaining Hop Count

Instruction Bitmap	Domain Specific ID
DS Instructions	DS Flags

0	Most Recent INT Metadata
0	INT Metadata
..	
..	
1	First INT Metadata

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

12 Byte **INT Header** with:

- INT Header Informations
- Instructions for hops

INT Metadata with Informations for each hop:

- metrics
- states

Added to packet as:

- Network Service Header Metadata
- TCP Options/Payload
- UDP Payload

Preliminary Test Results

- P4 Pipeline on Tofino takes about 600ns to process a packet
- Loopback Delay for Packets flowing out and coming back in ~100ns
- First measurements of external HW-Switch (EdgeCore AS4610, Helix4, 10 Gbit/s): ~2.1 µs

Further Steps

- Traffic Generation on Tofino: Work-around HW Limitations, Onboard TrafficGenerator
- Traffic Generation (Control Plane Integration)
- Front-End Application for INT
- Scapy GUI
- ‚In-Network' Computing (IoT-Data Aggregation)

Looking for funding:
Industry Sponsoring → 1-2 SHK (Studentische Hilfskräfte)
Projektskizze for formal IFAF-Proposal, or other funding body

Questions?

Contact for NetTraffic-P4:

Name: Prof. Dr. Thomas Scheffler
Email: thomas.scheffler@htw-berlin.de

Hochschule für Technik
und Wirtschaft Berlin

University of Applied Sciences