Universidade Federal de São Carlos
Campus Sorocaba

# P4: Programming Protocol-independent Packet Processors

## Dataplane programmability and P4

Prof. Fábio Luciano Verdi (verdi@ufscar.br)

# Introduction to P4: Programming Protocol-independent Packet Processors

- For this part, we will use the short course presented at SBRC 2018, which includes:
  - Presentation: **http://www.dcomp.ufscar.br/verdi/wp-content/uploads/2017/07/minicurso_p4_presentation.pdf**
  - Book chapter: **http://www.dcomp.ufscar.br/verdi/wp-content/uploads/2017/07/Capitulo4.pdf**
  - VM with exercises: to be provided.
- How to get help related to the P4 language, after extensively reading and trying out:
  - Leandro (PhD student): leandro.almeida@estudante.ufscar.br
  - Gustavo (master student): gvenancioluz@gmail.com
  - Guilherme (ex master student): guilherme.matos@estudante.ufscar.br

# $P4_{16}$ Program Template (v1model)

```
#include <core.p4>
#include <v1model.p4>
const bit<16> TYPE_IPV4 = 0x800;
********************* H E A D E R S
typedef bit<9>  egressSpec_t;
typedef bit<48> macAddr_t;
typedef bit<32> ip4Addr_t;
```

```
header ethernet_t {
    macAddr_t dstAddr;
    macAddr_t srcAddr;
    bit<16>   etherType;
}
```

```
header ipv4_t {
    bit<4>    version;
    bit<4>    ihl;
    bit<8>    diffserv;
    bit<16>   totalLen;
    bit<16>   identification;
    bit<3>    flags;
    bit<13>   fragOffset;
    bit<8>    ttl;
    bit<8>    protocol;
    bit<16>   hdrChecksum;
    ip4Addr_t srcAddr;
    ip4Addr_t dstAddr;
}
```

# P4$_{16}$ Program Template (V1Model)

```
#include <core.p4>
#include <v1model.p4>
/* HEADERS */
struct metadata { ... }
struct headers {
  ethernet_t   ethernet;
  ipv4_t       ipv4;
}
/* PARSER */
parser MyParser(packet_in packet,
          out headers hdr,
          inout metadata meta,
          inout standard_metadata_t smeta) {
  ...
}
/* CHECKSUM VERIFICATION */
control MyVerifyChecksum(in headers hdr,
                         inout metadata meta) {
   ...
}
/* INGRESS PROCESSING */
control MyIngress(inout headers hdr,
                  inout metadata meta,
                  inout standard_metadata_t std_meta) {
  ...
}
```

```
/* EGRESS PROCESSING */
control MyEgress(inout headers hdr,
                 inout metadata meta,
                 inout standard_metadata_t std_meta) {
  ...
}
/* CHECKSUM UPDATE */
control MyComputeChecksum(inout headers hdr,
                          inout metadata meta) {
  ...
}
/* DEPARSER */
control MyDeparser(inout headers hdr,
                   inout metadata meta) {
  ...
}
/* SWITCH */
V1Switch(
  MyParser(),
  MyVerifyChecksum(),
  MyIngress(),
  MyEgress(),
  MyComputeChecksum(),
  MyDeparser()
) main;
```

```
/*************************************************************************
**********************  D E P A R S E R  ******************************
*************************************************************************/

control MyDeparser(packet_out packet, in headers hdr) {
    apply {
        packet.emit(hdr.ethernet);
        packet.emit(hdr.ipv4);
    }
}
```

# P4$_{16}$ Hello World (V1Model)

```
#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;
```

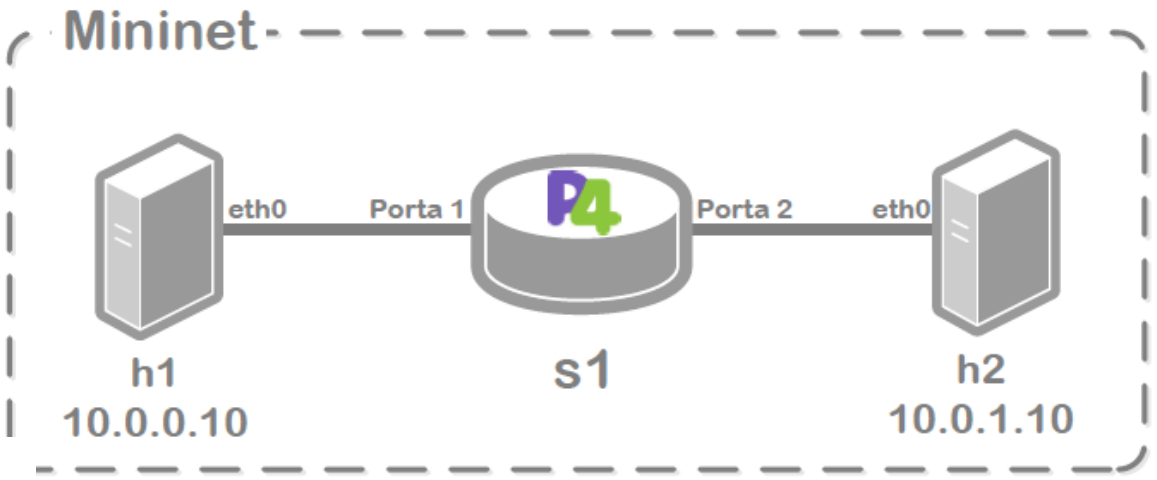
# P4$_{16}$ Hello World (V1Model)

```
#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 MyIngress(inout headers hdr, inout metadata meta,
   inout standard_metadata_t standard_metadata) {
    action set_egress_spec(bit<9> port) {
        standard_metadata.egress_spec = port;
    }
    table forward {
        key = { standard_metadata.ingress_port: exact; }
        actions = {
            set_egress_spec;
            NoAction;
        }
        size = 1024;
        default_action = NoAction();
    }
    apply {   forward.apply();   }
}
```

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

control MyVerifyChecksum(inout headers hdr, inout metadata
meta) {   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;
```

| Key | Action Name | Action Data |
|---|---|---|
| 1 | set_egress_spec | 2 |
| 2 | set_egress_spec | 1 |

# First task: IP routing




These rules are defined in the commands_s1.txt file

```
table_set_default ipv4_lpm drop
table_add ipv4_lpm ipv4_forward 10.0.0.10/32 => 00:04:00:00:00:00 1
table_add ipv4_lpm ipv4_forward 10.0.1.10/32 => 00:04:00:00:00:01 2
```

- Update the source and destination MAC address
- Decrease by 1 the TTL of the packet
- Forward the packet to the output port

# Download the VM…

- https://drive.google.com/file/d/1hGPzWU1nS0urE1sZh3ksayvrlgW_EGIh/view
  - login: student
  - passwd: student

# Further Reading/watching

- **Minicurso SBRC 2018**
  - **http://www.dcomp.ufscar.br/verdi/wp-content/uploads/2017/07/Capitulo4.pdf**
  - **http://www.dcomp.ufscar.br/verdi/wp-content/uploads/2017/07/minicurso_p4_presentation.pdf**
- **P4: programming protocol-independent packet processors**
  - https://dl.acm.org/doi/10.1145/2656877.2656890
- **$P4_{16}$ Language Specification**
  - https://p4.org/p4-spec/docs/P4-16-v1.2.2.html
- **P4 tutorials**
  - **https://github.com/p4lang/tutorials**
  - **https://github.com/nsg-ethz/p4-learning**
  - **https://carolinafernandez.github.io/development/2019/08/06/Recurrent-processing-in-P4**
- **Videos**:
  - https://www.youtube.com/watch?v=qxT7DKOIk7Q&list=PLf7HGRMAlJBzGC58GcYpimyIs7D0nuSoo&index=2