DAFTAR PUSTAKA

Ariyadi, T., Restu Mukti, A., & Saputra, H. (2022). *Mitigation of Distributed Denial of Service (DDoS) Attacks on Software Defined Network (SDN) Architecture* (Vol. 21, Nomor 4).

Beausencourt, M. (2023). *Building a Training Platform for the Programming Language P4* [University of Applied Sciences]. https://doi.org/DOI: 10.13140/RG.2.2.18531.73765

Goswami, B., Kulkarni, M., & Paulose, J. (2023). A Survey on P4 Challenges in Software Defined Networks: P4 Programming. Dalam *IEEE Access* (Vol. 11, hlm. 54373–54387). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/ACCESS.2023.3275756

Gupta, B. B., & Dahiya, A. (2021). *Distributed Denial of Service (DDoS) Attacks; Classification, Attacks, Challenges, and Countermeasures* (1 ed.). CRC Press.

Hang, Z., Wen, M., Shi, Y., & Zhang, C. (2019). Programming protocol-independent packet processors high-level programming (P4HLP): Towards unified high-level programming for a commodity programmable switch. *Electronics (Switzerland)*, *8*(9). https://doi.org/10.3390/electronics8090958

Harkous, H., Sherkawi, K., Jarschel, M., Pries, R., He, M., & Kellerer, W. (2021). P4RCProbe for Evaluating the Performance of P4Runtime-based Controllers. *2021 IEEE Conference on Network Function Virtualization and Software Defined Networks, NFV-SDN 2021 - Proceedings*, 74–80. https://doi.org/10.1109/NFV-SDN53031.2021.9665026

jafingerhut/p4-guide: Guide to p4lang repositories and some other public info about P4. (t.t.). Diambil 18 Mei 2024, dari https://github.com/jafingerhut/p4-guide

Kalabo, E. H., & Dwi Setiawan Sumadi, F. (2022). Analisa Performa Intrusion Detection System (IDS) Snort d/dan Suricata T erhadap Serangan T CP SYN Flood. *REPOSITOR*, *4*(3), 397–406.

Malchiodi, D., Raimondi, D., Fumagalli, G., Giancarlo, R., & Frasca, M. (2024). The role of classifiers and data complexity in learned Bloom filters: insights and recommendations. *Journal of Big Data*, *11*(1). https://doi.org/10.1186/s40537-024-00906-9

Mininet Overview - Mininet. (2022). Mininet Project Contribution. https://mininet.org/overview/

Nisa, F., & Ramadona, S. (2023). Sistem Pencegahan Serangan Distributed Denial Of Service Pada Jaringan SDN. *Jurnal Sistim Informasi dan Teknologi*, *5*(3), 22–30.

Oracle VM VirtualBox. (2023). ORACLE. https://www.virtualbox.org/wiki/WikiStart

P4 – Language Consortium. (2024). Linux Foundation. https://p4.org/

p4lang/behavioral-model: The reference P4 software switch. (t.t.). Diambil 19 Mei 2024, dari https://github.com/p4lang/behavioral-model

Piccioni, L., & Zanchetta, E. (t.t.). *XTERM: A FLEXIBLE STANDARD-COMPLIANT XML-BASED TERMBASE MANAGEMENT SYSTEM*. http://www.terminologia.it

Sahren, S., Dalimunthe, R. A., Saputra, H., & Kurnia Sirni, D. Y. (2023). IDPS Performance Aanalysis For Mitigating SQL Injections And SYN Flood Attaks. *JURTEKSI (Jurnal Teknologi dan Sistem Informasi)*, *10*(1), 171–178. https://doi.org/10.33330/jurteksi.v10i1.2880

Serangan, S., & Yasin, A. (2020). Simulasi Serangan Distributed Denial Of Service (DDoS) Di Software-Defined Network Di Simulasi Jaringan Mininet Dan GNS3. *JTII*, *05*(01).

Sidiq, M. F., Indra Basuki, A., & Rosiyadi, D. (2020). MiTE: Program Penyunting Topologi Jaringan untuk Pembelajaran SDN. *RESTI (Rekayasa Sistem dan Teknologi)*, *4*(5), 970–977.

Tamakloe, E., Kommey, B., Akowuah, E., & Opoku, D. (2023). A Detailed Review on The Denial of Service (DoS) and Distributed Denial of Service (DDoS) Attacks in Software Defined Networks (SDNs) and Defense Strategies. *International Journal of Applied Sciences and Smart Technologies*, *5*(2), 127–158. https://doi.org/10.24071/ijasst.v5i2.6380

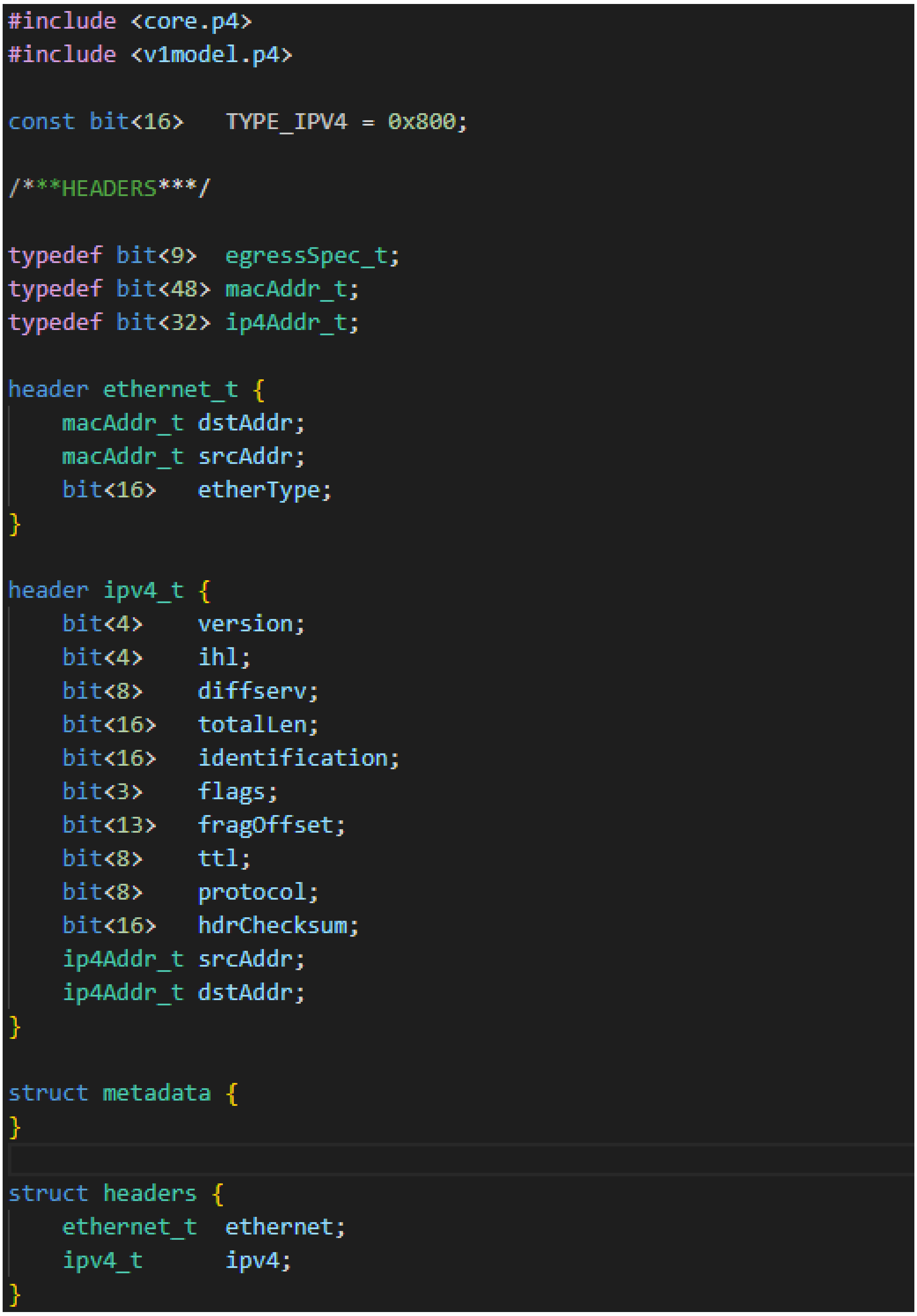
What is OSI Model? - Layers of OSI Model. (2024, Mei 6). Sanchhaya Education Private Limited. https://www.geeksforgeeks.org/open-systems-interconnection-model-osi/

Wireshark. (2024, April). Wireshark Foundation. https://www.wireshark.org/

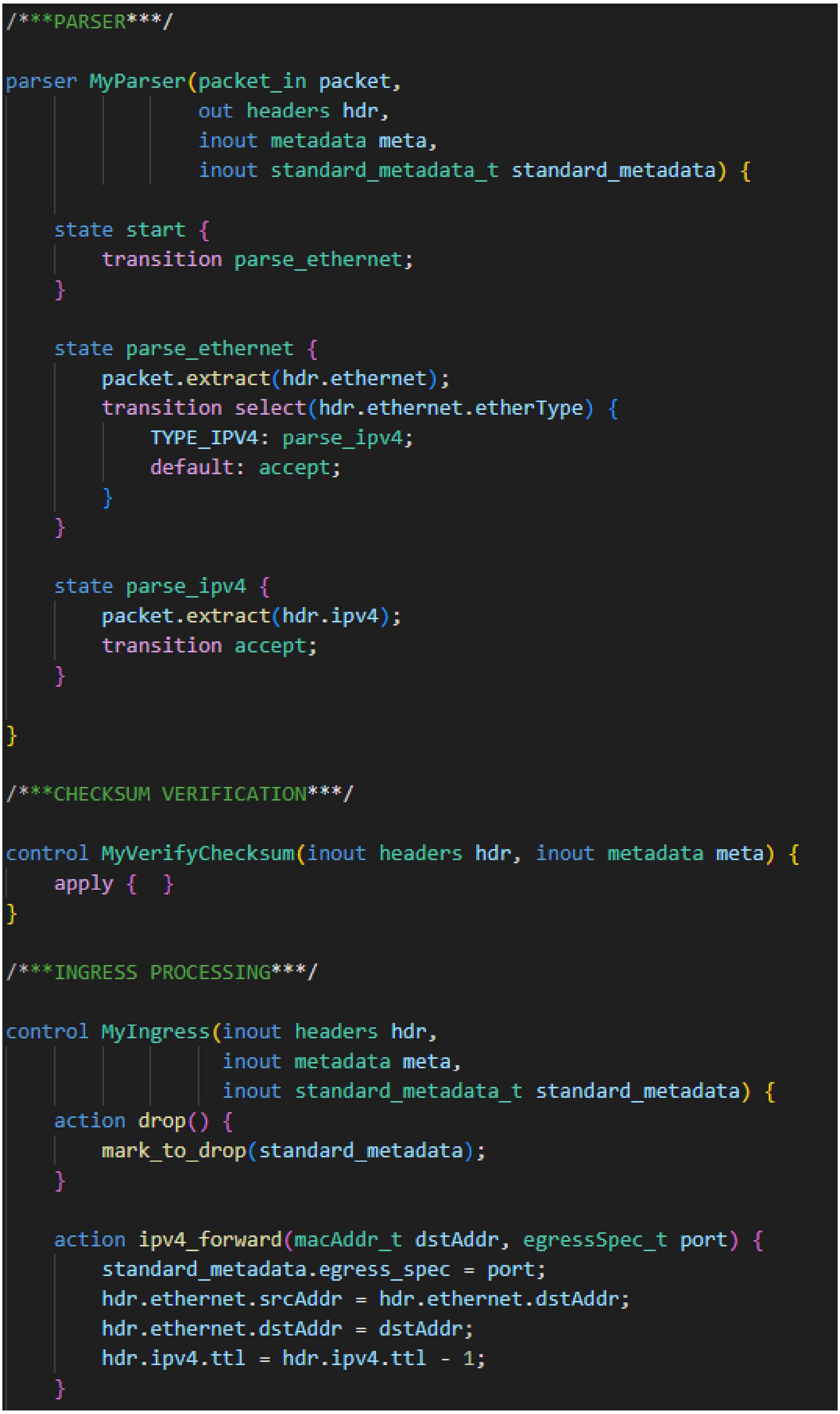
Yoachimik, O., & Pacheco, J. (2024, Januari 9). *DDoS threat report for 2023 Q4*. CLOUDFLARE. https://blog.cloudflare.com/ddos-threat-report-2023-q4

LAMPIRAN

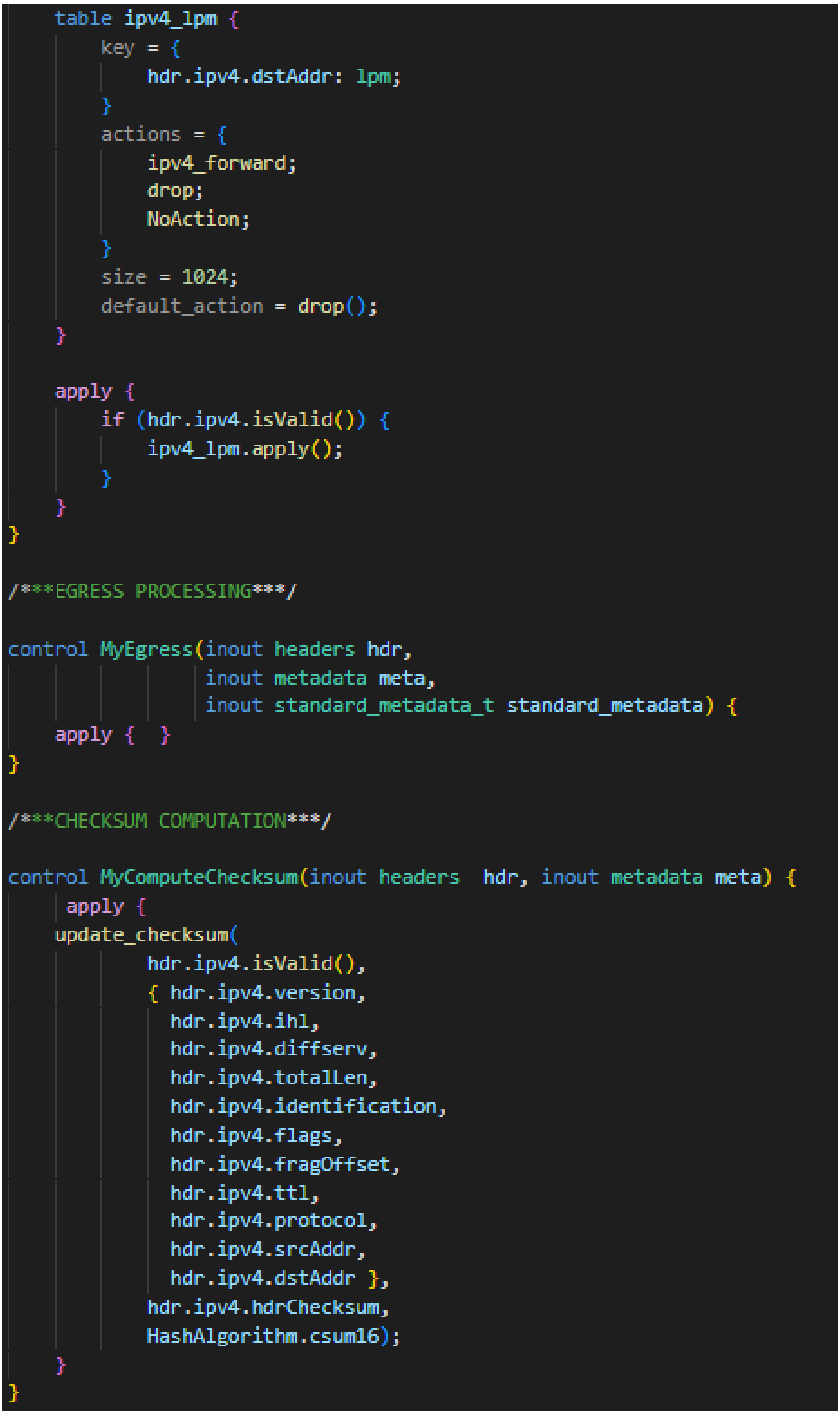
Lampiran 1 P4 basic.p4 script sebelum implementasi Metode Keamanan



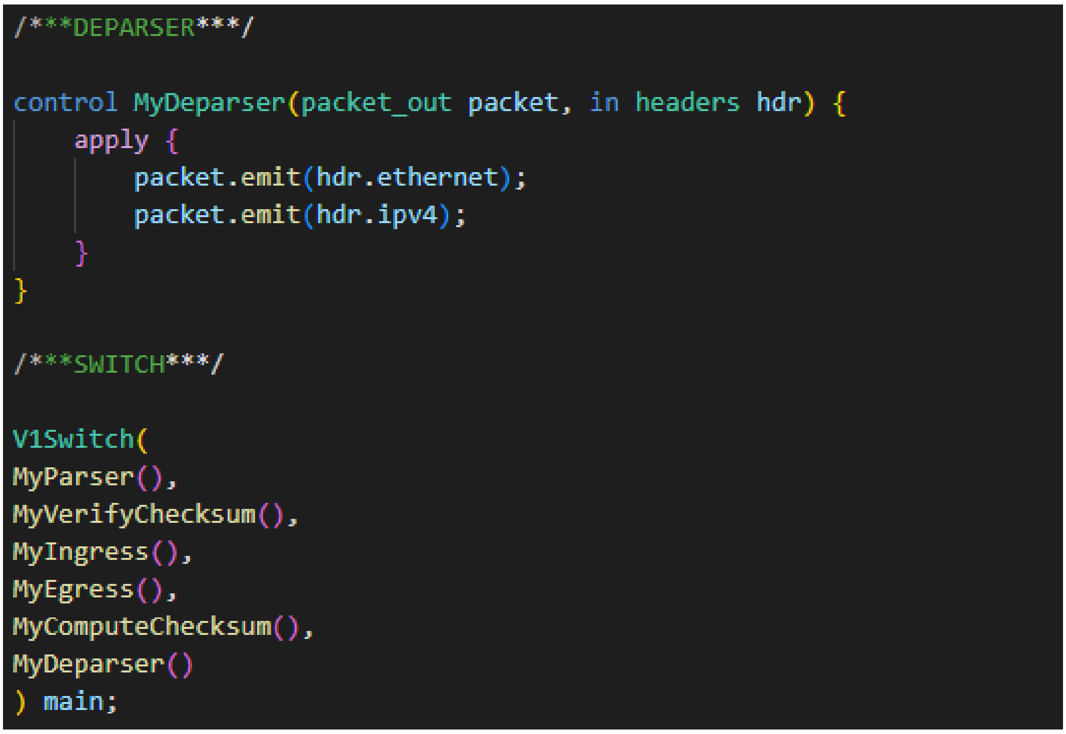
```
#include <core.p4>
#include <v1model.p4>

const bit<16>   TYPE_IPV4 = 0x800;

/***HEADERS***/

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;
}

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 standard_metadata) {

    state start {
        transition parse_ethernet;
    }

    state parse_ethernet {
        packet.extract(hdr.ethernet);
        transition select(hdr.ethernet.etherType) {
            TYPE_IPV4: parse_ipv4;
            default: accept;
        }
    }

    state parse_ipv4 {
        packet.extract(hdr.ipv4);
        transition accept;
    }

}

/***CHECKSUM VERIFICATION***/

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

/***INGRESS PROCESSING***/

control MyIngress(inout headers hdr,
                  inout metadata meta,
                  inout standard_metadata_t standard_metadata) {
    action drop() {
        mark_to_drop(standard_metadata);
    }

    action ipv4_forward(macAddr_t dstAddr, egressSpec_t port) {
        standard_metadata.egress_spec = port;
        hdr.ethernet.srcAddr = hdr.ethernet.dstAddr;
        hdr.ethernet.dstAddr = dstAddr;
        hdr.ipv4.ttl = hdr.ipv4.ttl - 1;
    }
```

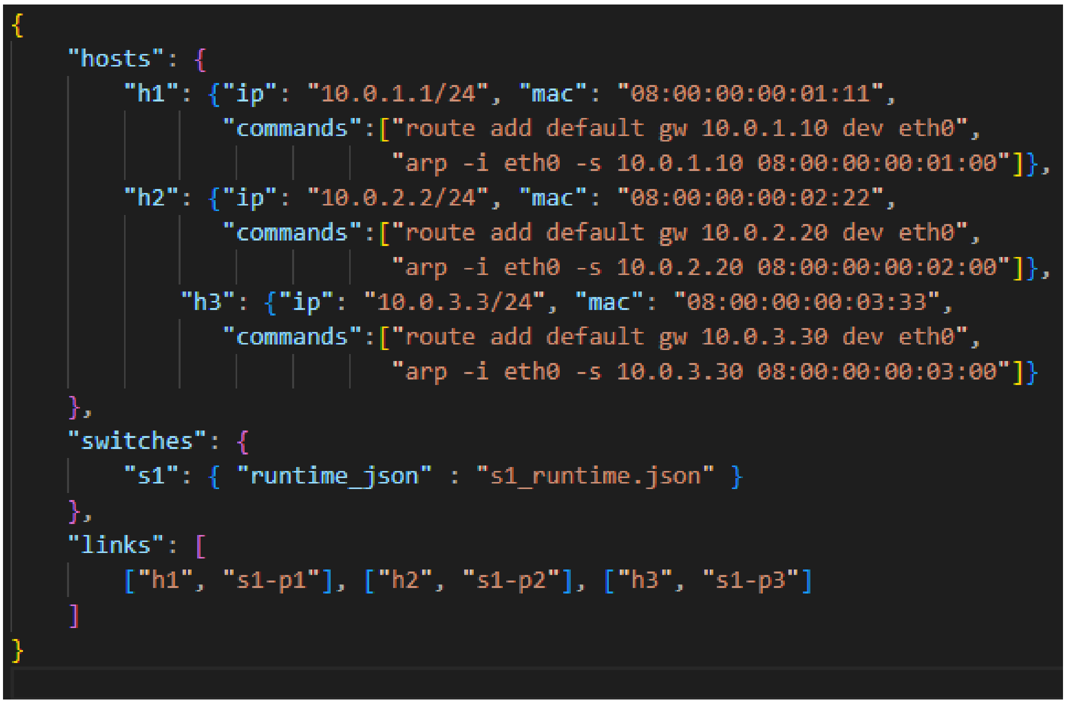


```
    table ipv4_lpm {
        key = {
            hdr.ipv4.dstAddr: lpm;
        }
        actions = {
            ipv4_forward;
            drop;
            NoAction;
        }
        size = 1024;
        default_action = drop();
    }

    apply {
        if (hdr.ipv4.isValid()) {
            ipv4_lpm.apply();
        }
    }
}

/***EGRESS PROCESSING***/

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

/***CHECKSUM COMPUTATION***/

control MyComputeChecksum(inout headers  hdr, inout metadata meta) {
     apply {
    update_checksum(
            hdr.ipv4.isValid(),
            { hdr.ipv4.version,
              hdr.ipv4.ihl,
              hdr.ipv4.diffserv,
              hdr.ipv4.totalLen,
              hdr.ipv4.identification,
              hdr.ipv4.flags,
              hdr.ipv4.fragOffset,
              hdr.ipv4.ttl,
              hdr.ipv4.protocol,
              hdr.ipv4.srcAddr,
              hdr.ipv4.dstAddr },
            hdr.ipv4.hdrChecksum,
            HashAlgorithm.csum16);
    }
}
```



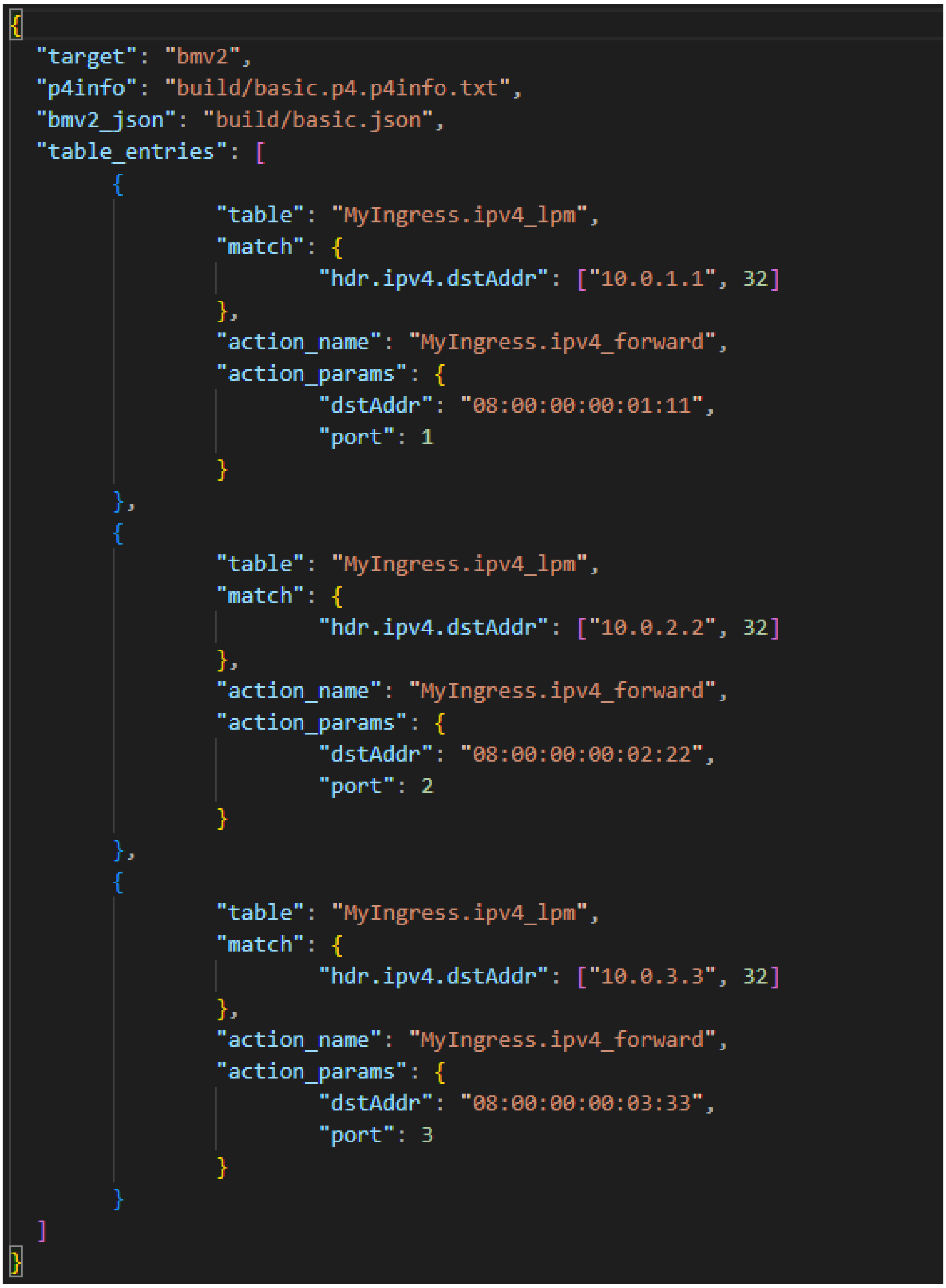
```
/***DEPARSER***/

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

/***SWITCH***/

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

Lampiran 2 JSON script topology.json untuk basic.p4



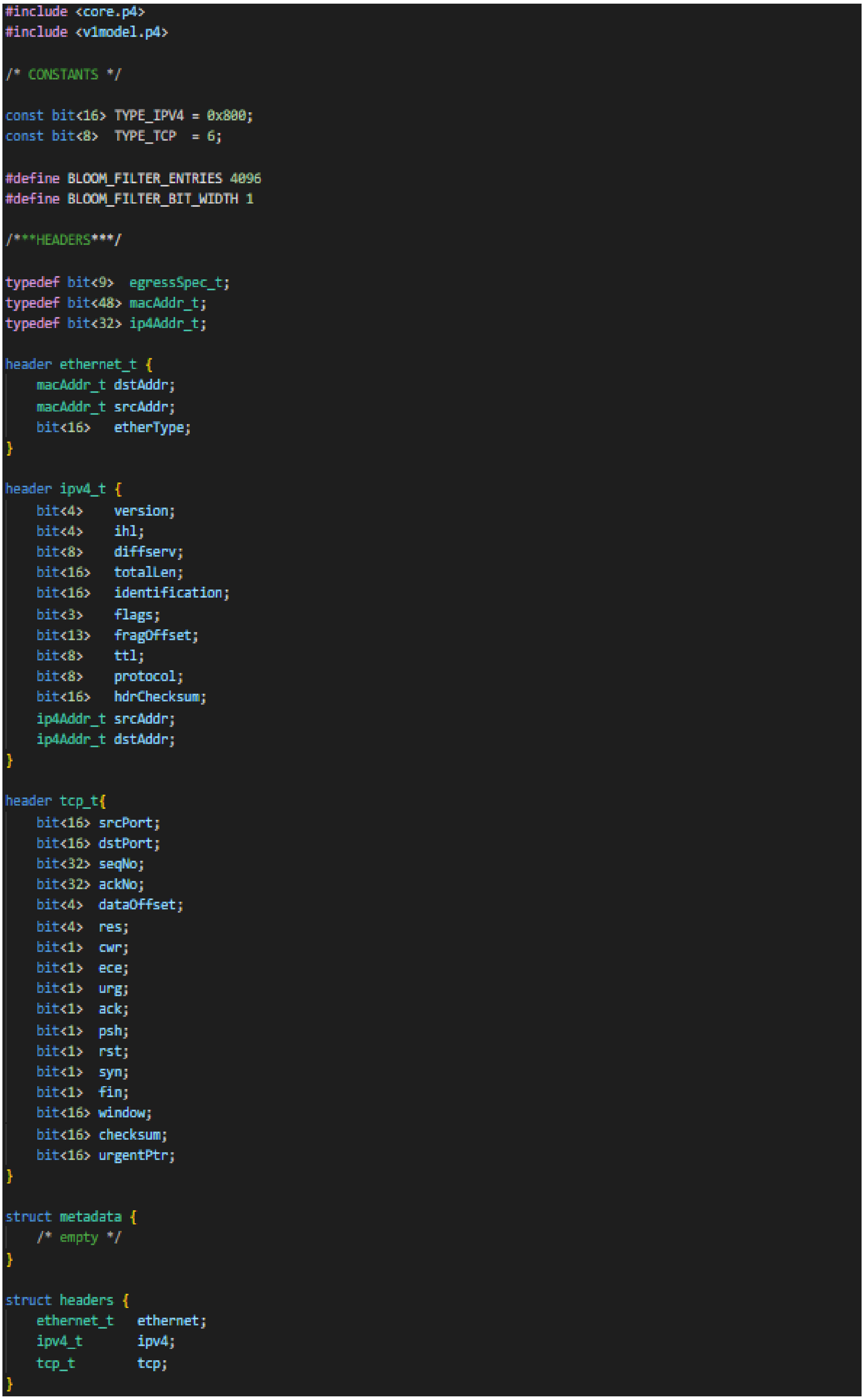
```
{
    "hosts": {
        "h1": {"ip": "10.0.1.1/24", "mac": "08:00:00:00:01:11",
               "commands":["route add default gw 10.0.1.10 dev eth0",
                           "arp -i eth0 -s 10.0.1.10 08:00:00:00:01:00"]},
        "h2": {"ip": "10.0.2.2/24", "mac": "08:00:00:00:02:22",
               "commands":["route add default gw 10.0.2.20 dev eth0",
                           "arp -i eth0 -s 10.0.2.20 08:00:00:00:02:00"]},
            "h3": {"ip": "10.0.3.3/24", "mac": "08:00:00:00:03:33",
               "commands":["route add default gw 10.0.3.30 dev eth0",
                           "arp -i eth0 -s 10.0.3.30 08:00:00:00:03:00"]}
    },
    "switches": {
        "s1": { "runtime_json" : "s1_runtime.json" }
    },
    "links": [
        ["h1", "s1-p1"], ["h2", "s1-p2"], ["h3", "s1-p3"]
    ]
}
```

Lampiran 3 JSON script s1-runtime.json untuk basic.p4

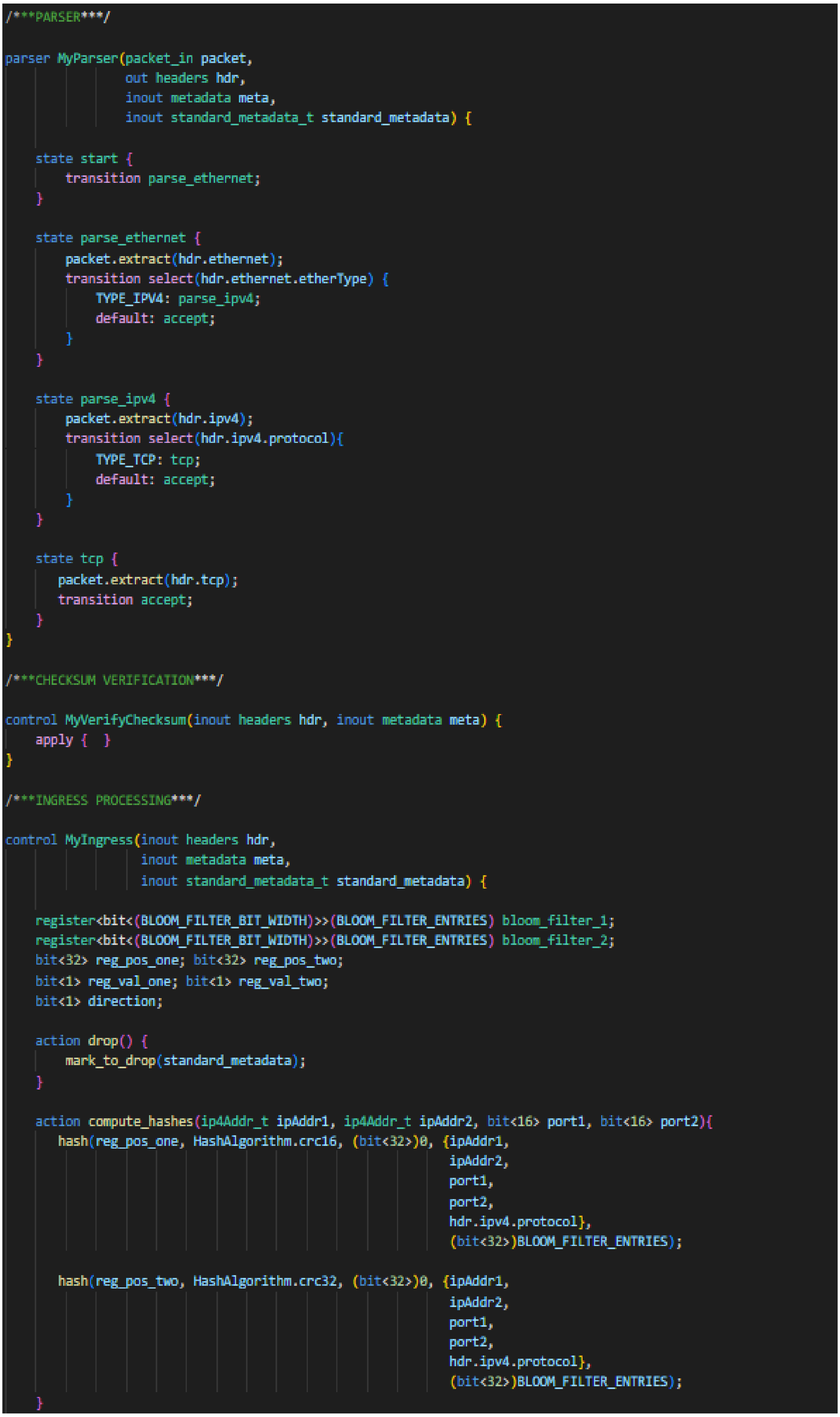


```
{
  "target": "bmv2",
  "p4info": "build/basic.p4.p4info.txt",
  "bmv2_json": "build/basic.json",
  "table_entries": [
        {
                "table": "MyIngress.ipv4_lpm",
                "match": {
                        "hdr.ipv4.dstAddr": ["10.0.1.1", 32]
                },
                "action_name": "MyIngress.ipv4_forward",
                "action_params": {
                        "dstAddr": "08:00:00:00:01:11",
                        "port": 1
                }
        },
        {
                "table": "MyIngress.ipv4_lpm",
                "match": {
                        "hdr.ipv4.dstAddr": ["10.0.2.2", 32]
                },
                "action_name": "MyIngress.ipv4_forward",
                "action_params": {
                        "dstAddr": "08:00:00:00:02:22",
                        "port": 2
                }
        },
        {
                "table": "MyIngress.ipv4_lpm",
                "match": {
                        "hdr.ipv4.dstAddr": ["10.0.3.3", 32]
                },
                "action_name": "MyIngress.ipv4_forward",
                "action_params": {
                        "dstAddr": "08:00:00:00:03:33",
                        "port": 3
                }
        }
  ]
}
```

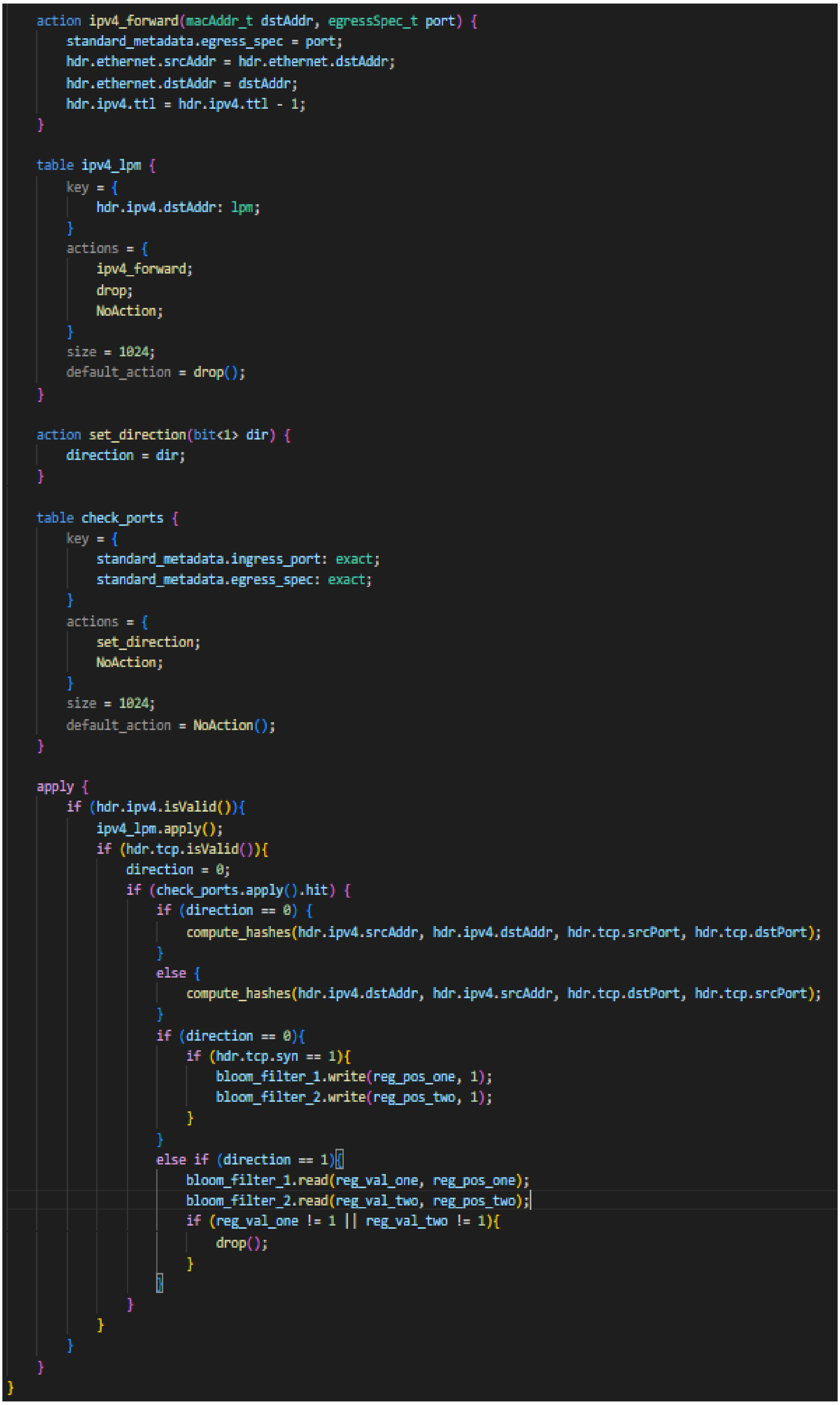
Lampiran 4 P4 dice.p4 script implementasi Metode Keamanan



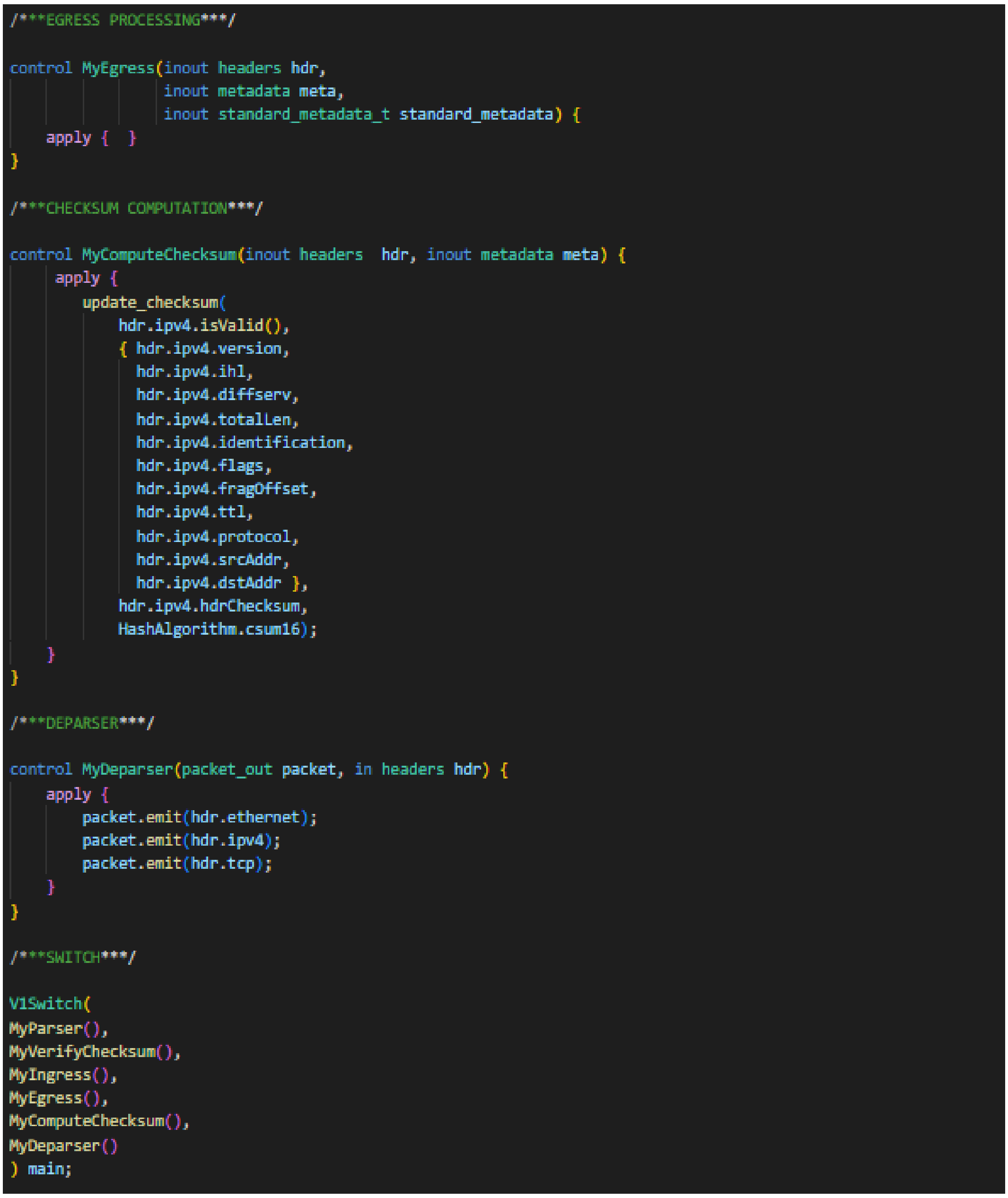
```
#include <core.p4>
#include <v1model.p4>

/* CONSTANTS */

const bit<16> TYPE_IPV4 = 0x800;
const bit<8>  TYPE_TCP  = 6;

#define BLOOM_FILTER_ENTRIES 4096
#define BLOOM_FILTER_BIT_WIDTH 1

/***HEADERS***/

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;
}

header tcp_t{
    bit<16> srcPort;
    bit<16> dstPort;
    bit<32> seqNo;
    bit<32> ackNo;
    bit<4>  dataOffset;
    bit<4>  res;
    bit<1>  cwr;
    bit<1>  ece;
    bit<1>  urg;
    bit<1>  ack;
    bit<1>  psh;
    bit<1>  rst;
    bit<1>  syn;
    bit<1>  fin;
    bit<16> window;
    bit<16> checksum;
    bit<16> urgentPtr;
}

struct metadata {
    /* empty */
}

struct headers {
    ethernet_t   ethernet;
    ipv4_t       ipv4;
    tcp_t        tcp;
}
```



```
/***PARSER***/

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

    state start {
        transition parse_ethernet;
    }

    state parse_ethernet {
        packet.extract(hdr.ethernet);
        transition select(hdr.ethernet.etherType) {
            TYPE_IPV4: parse_ipv4;
            default: accept;
        }
    }

    state parse_ipv4 {
        packet.extract(hdr.ipv4);
        transition select(hdr.ipv4.protocol){
            TYPE_TCP: tcp;
            default: accept;
        }
    }

    state tcp {
       packet.extract(hdr.tcp);
       transition accept;
    }
}

/***CHECKSUM VERIFICATION***/

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

/***INGRESS PROCESSING***/

control MyIngress(inout headers hdr,
                  inout metadata meta,
                  inout standard_metadata_t standard_metadata) {

    register<bit<(BLOOM_FILTER_BIT_WIDTH)>>(BLOOM_FILTER_ENTRIES) bloom_filter_1;
    register<bit<(BLOOM_FILTER_BIT_WIDTH)>>(BLOOM_FILTER_ENTRIES) bloom_filter_2;
    bit<32> reg_pos_one; bit<32> reg_pos_two;
    bit<1> reg_val_one; bit<1> reg_val_two;
    bit<1> direction;

    action drop() {
        mark_to_drop(standard_metadata);
    }

    action compute_hashes(ip4Addr_t ipAddr1, ip4Addr_t ipAddr2, bit<16> port1, bit<16> port2){
       hash(reg_pos_one, HashAlgorithm.crc16, (bit<32>)0, {ipAddr1,
                                                           ipAddr2,
                                                           port1,
                                                           port2,
                                                           hdr.ipv4.protocol},
                                                           (bit<32>)BLOOM_FILTER_ENTRIES);

       hash(reg_pos_two, HashAlgorithm.crc32, (bit<32>)0, {ipAddr1,
                                                           ipAddr2,
                                                           port1,
                                                           port2,
                                                           hdr.ipv4.protocol},
                                                           (bit<32>)BLOOM_FILTER_ENTRIES);
    }
```

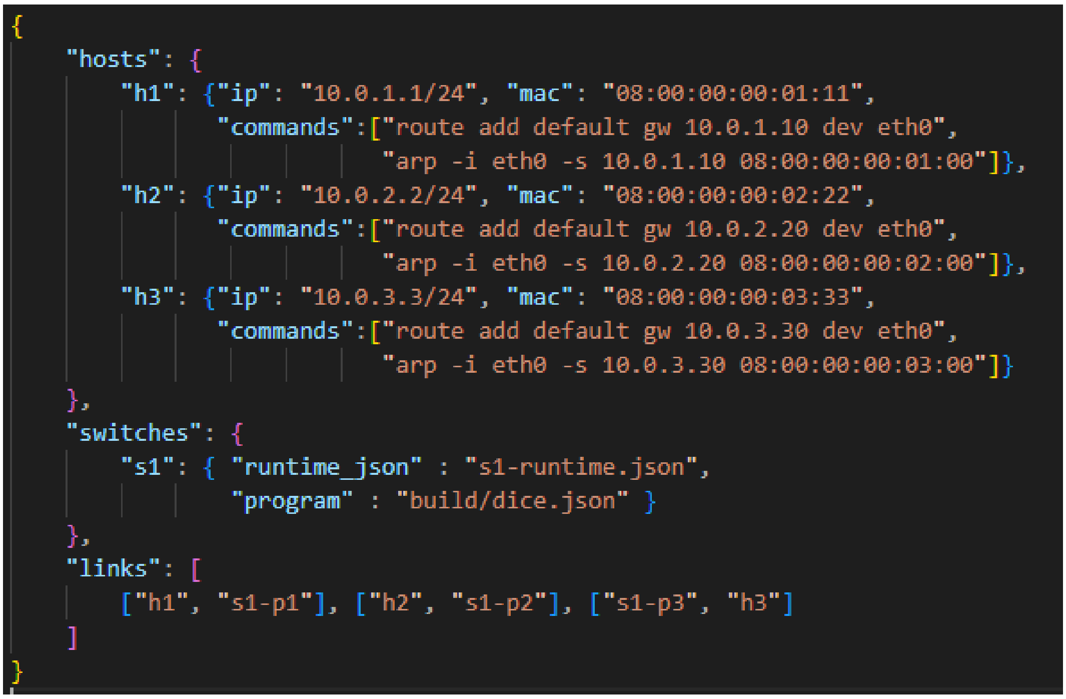


```
    action ipv4_forward(macAddr_t dstAddr, egressSpec_t port) {
        standard_metadata.egress_spec = port;
        hdr.ethernet.srcAddr = hdr.ethernet.dstAddr;
        hdr.ethernet.dstAddr = dstAddr;
        hdr.ipv4.ttl = hdr.ipv4.ttl - 1;
    }

    table ipv4_lpm {
        key = {
            hdr.ipv4.dstAddr: lpm;
        }
        actions = {
            ipv4_forward;
            drop;
            NoAction;
        }
        size = 1024;
        default_action = drop();
    }

    action set_direction(bit<1> dir) {
        direction = dir;
    }

    table check_ports {
        key = {
            standard_metadata.ingress_port: exact;
            standard_metadata.egress_spec: exact;
        }
        actions = {
            set_direction;
            NoAction;
        }
        size = 1024;
        default_action = NoAction();
    }

    apply {
        if (hdr.ipv4.isValid()){
            ipv4_lpm.apply();
            if (hdr.tcp.isValid()){
                direction = 0;
                if (check_ports.apply().hit) {
                    if (direction == 0) {
                        compute_hashes(hdr.ipv4.srcAddr, hdr.ipv4.dstAddr, hdr.tcp.srcPort, hdr.tcp.dstPort);
                    }
                    else {
                        compute_hashes(hdr.ipv4.dstAddr, hdr.ipv4.srcAddr, hdr.tcp.dstPort, hdr.tcp.srcPort);
                    }
                    if (direction == 0){
                        if (hdr.tcp.syn == 1){
                            bloom_filter_1.write(reg_pos_one, 1);
                            bloom_filter_2.write(reg_pos_two, 1);
                        }
                    }
                    else if (direction == 1){
                        bloom_filter_1.read(reg_val_one, reg_pos_one);
                        bloom_filter_2.read(reg_val_two, reg_pos_two);
                        if (reg_val_one != 1 || reg_val_two != 1){
                            drop();
                        }
                    }
                }
            }
        }
    }
}
```



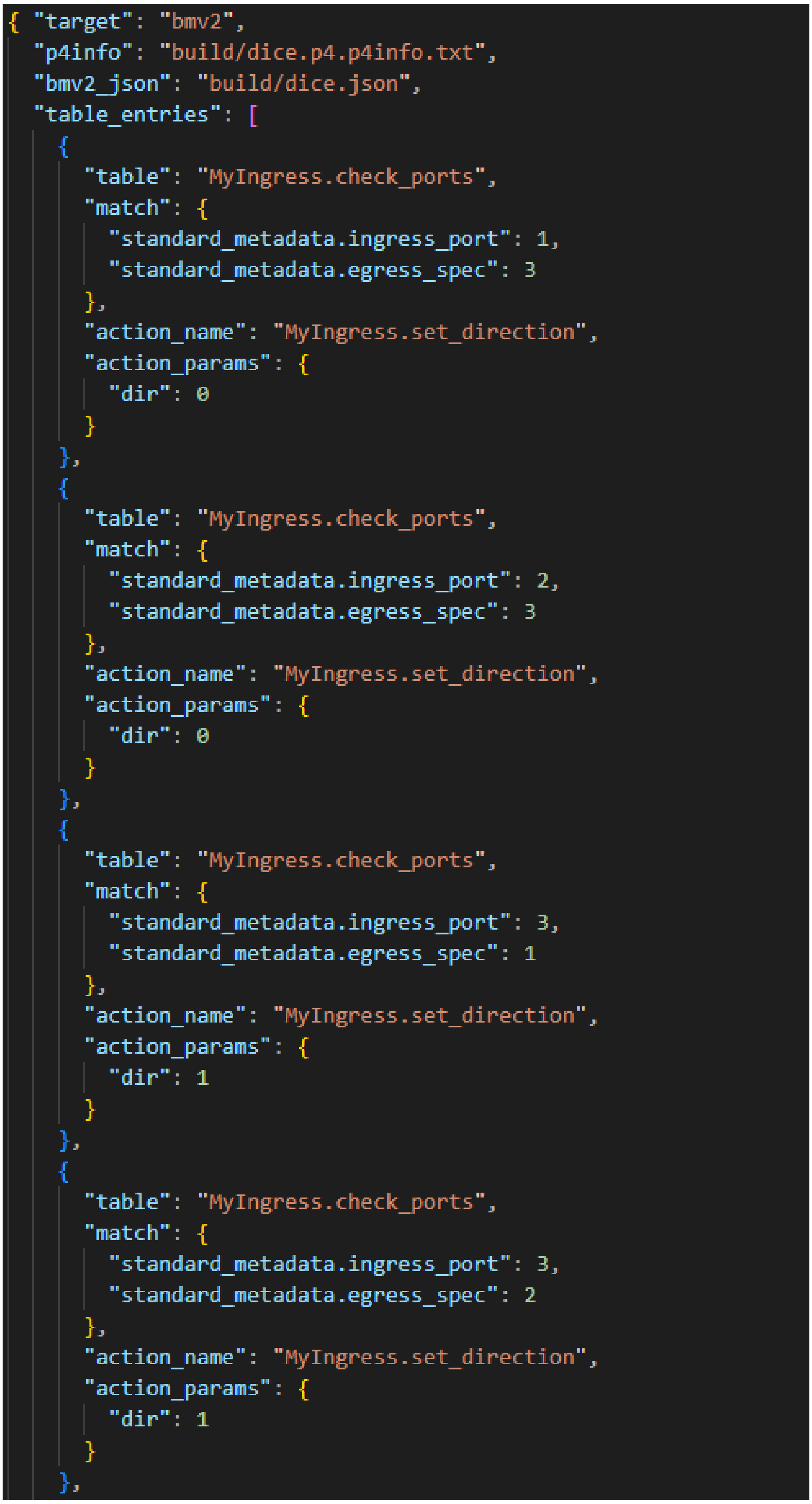
```
/***EGRESS PROCESSING***/

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

/***CHECKSUM COMPUTATION***/

control MyComputeChecksum(inout headers  hdr, inout metadata meta) {
     apply {
        update_checksum(
            hdr.ipv4.isValid(),
            { hdr.ipv4.version,
              hdr.ipv4.ihl,
              hdr.ipv4.diffserv,
              hdr.ipv4.totalLen,
              hdr.ipv4.identification,
              hdr.ipv4.flags,
              hdr.ipv4.fragOffset,
              hdr.ipv4.ttl,
              hdr.ipv4.protocol,
              hdr.ipv4.srcAddr,
              hdr.ipv4.dstAddr },
            hdr.ipv4.hdrChecksum,
            HashAlgorithm.csum16);
    }
}

/***DEPARSER***/

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

/***SWITCH***/

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

Lampiran 5 JSON script topology.json untuk dice.p4

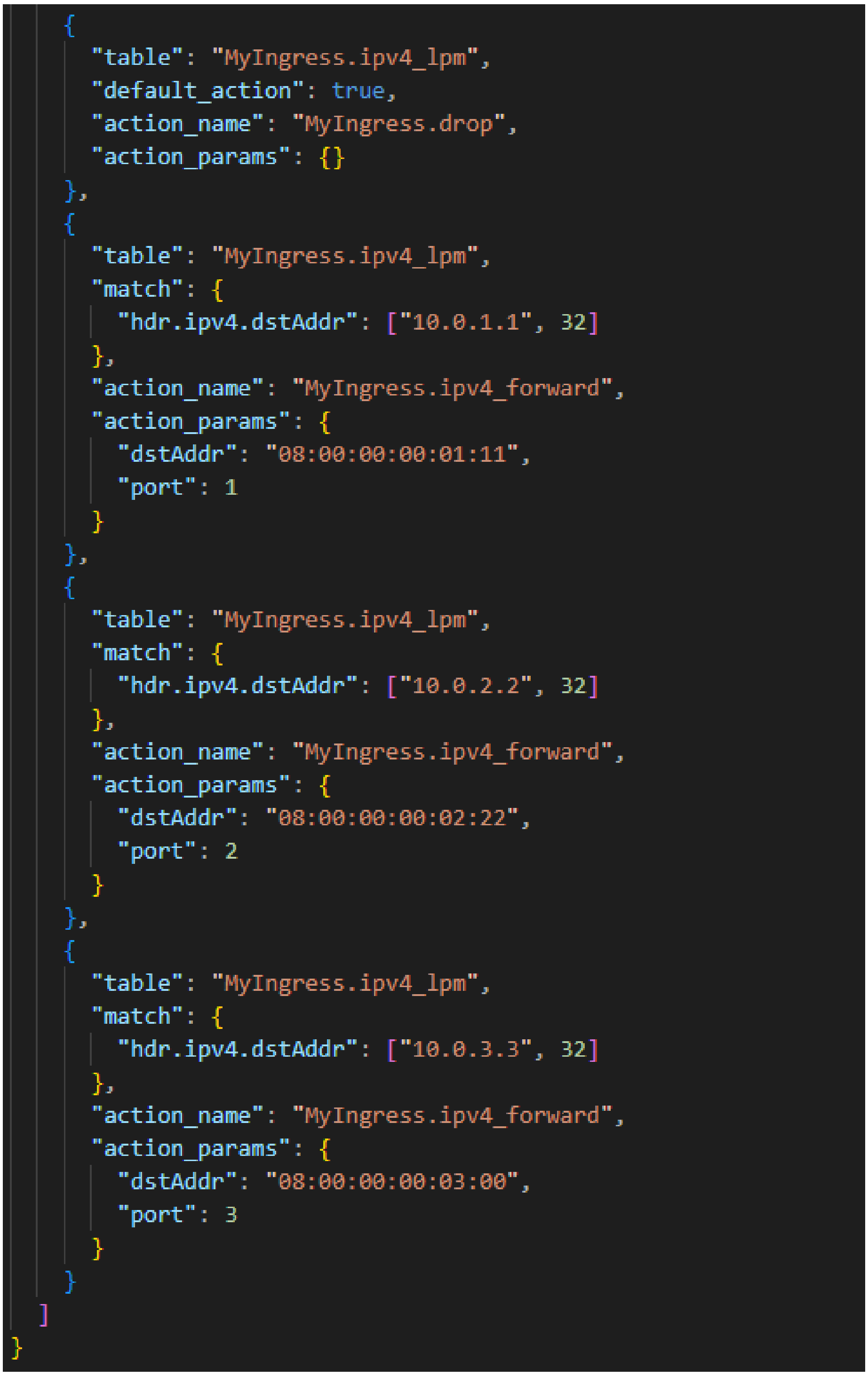


```
{
    "hosts": {
        "h1": {"ip": "10.0.1.1/24", "mac": "08:00:00:00:01:11",
               "commands":["route add default gw 10.0.1.10 dev eth0",
                           "arp -i eth0 -s 10.0.1.10 08:00:00:00:01:00"]},
        "h2": {"ip": "10.0.2.2/24", "mac": "08:00:00:00:02:22",
               "commands":["route add default gw 10.0.2.20 dev eth0",
                           "arp -i eth0 -s 10.0.2.20 08:00:00:00:02:00"]},
        "h3": {"ip": "10.0.3.3/24", "mac": "08:00:00:00:03:33",
               "commands":["route add default gw 10.0.3.30 dev eth0",
                           "arp -i eth0 -s 10.0.3.30 08:00:00:00:03:00"]}
    },
    "switches": {
        "s1": { "runtime_json" : "s1-runtime.json",
                "program" : "build/dice.json" }
    },
    "links": [
        ["h1", "s1-p1"], ["h2", "s1-p2"], ["s1-p3", "h3"]
    ]
}
```

Lampiran 6 JSON script s1-runtime.json untuk dice.p4



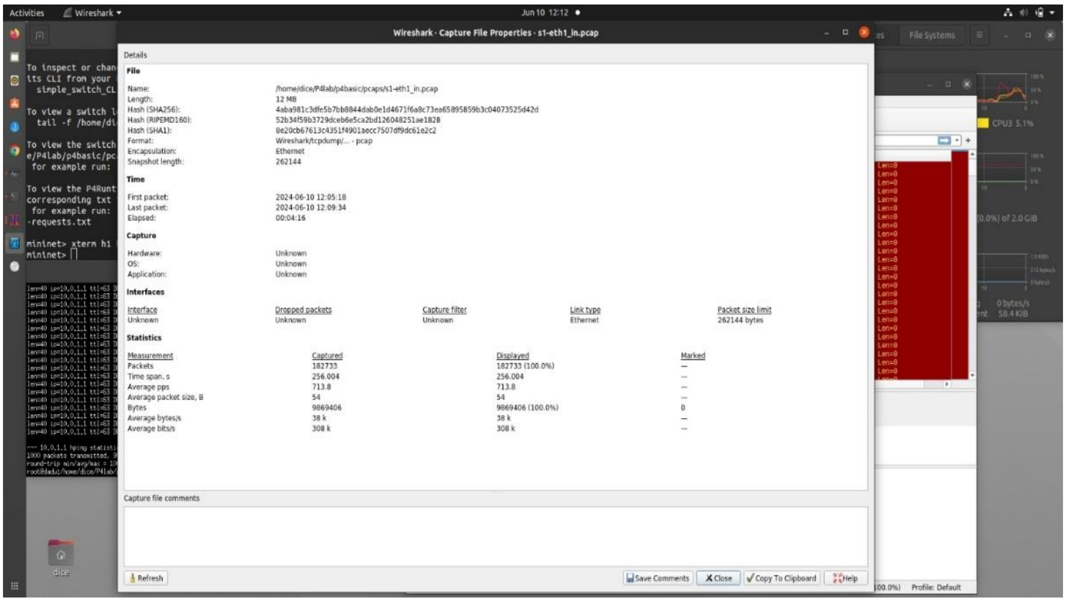
```
{ "target": "bmv2",
  "p4info": "build/dice.p4.p4info.txt",
  "bmv2_json": "build/dice.json",
  "table_entries": [
    {
      "table": "MyIngress.check_ports",
      "match": {
        "standard_metadata.ingress_port": 1,
        "standard_metadata.egress_spec": 3
      },
      "action_name": "MyIngress.set_direction",
      "action_params": {
        "dir": 0
      }
    },
    {
      "table": "MyIngress.check_ports",
      "match": {
        "standard_metadata.ingress_port": 2,
        "standard_metadata.egress_spec": 3
      },
      "action_name": "MyIngress.set_direction",
      "action_params": {
        "dir": 0
      }
    },
    {
      "table": "MyIngress.check_ports",
      "match": {
        "standard_metadata.ingress_port": 3,
        "standard_metadata.egress_spec": 1
      },
      "action_name": "MyIngress.set_direction",
      "action_params": {
        "dir": 1
      }
    },
    {
      "table": "MyIngress.check_ports",
      "match": {
        "standard_metadata.ingress_port": 3,
        "standard_metadata.egress_spec": 2
      },
      "action_name": "MyIngress.set_direction",
      "action_params": {
        "dir": 1
      }
    },
```



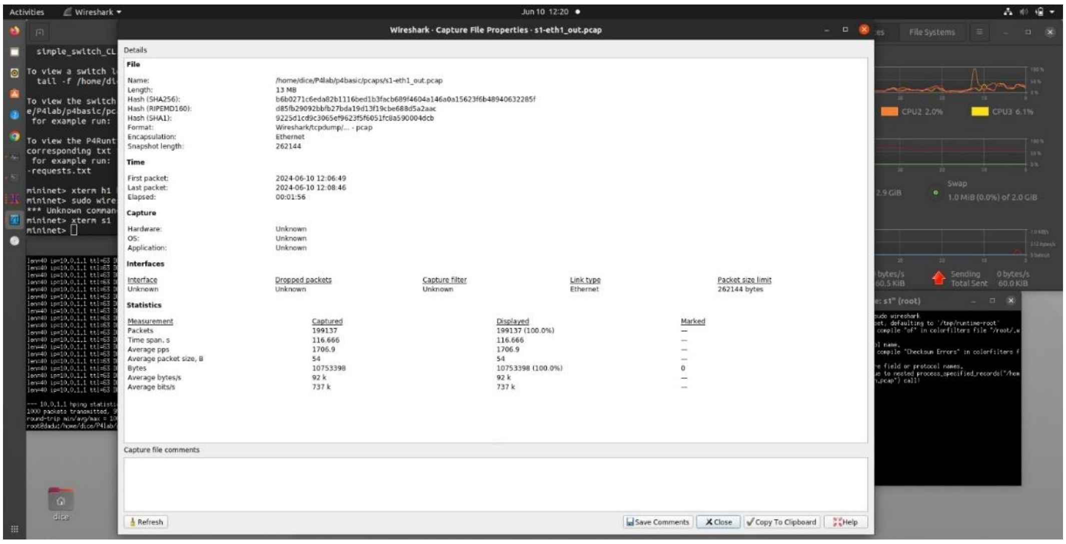
```
    {
      "table": "MyIngress.ipv4_lpm",
      "default_action": true,
      "action_name": "MyIngress.drop",
      "action_params": {}
    },
    {
      "table": "MyIngress.ipv4_lpm",
      "match": {
        "hdr.ipv4.dstAddr": ["10.0.1.1", 32]
      },
      "action_name": "MyIngress.ipv4_forward",
      "action_params": {
        "dstAddr": "08:00:00:00:01:11",
        "port": 1
      }
    },
    {
      "table": "MyIngress.ipv4_lpm",
      "match": {
        "hdr.ipv4.dstAddr": ["10.0.2.2", 32]
      },
      "action_name": "MyIngress.ipv4_forward",
      "action_params": {
        "dstAddr": "08:00:00:00:02:22",
        "port": 2
      }
    },
    {
      "table": "MyIngress.ipv4_lpm",
      "match": {
        "hdr.ipv4.dstAddr": ["10.0.3.3", 32]
      },
      "action_name": "MyIngress.ipv4_forward",
      "action_params": {
        "dstAddr": "08:00:00:00:03:00",
        "port": 3
      }
    }
  ]
}
```

Lampiran 7 Wireshark Capture Sebelum Implementasi Metode Keamanan

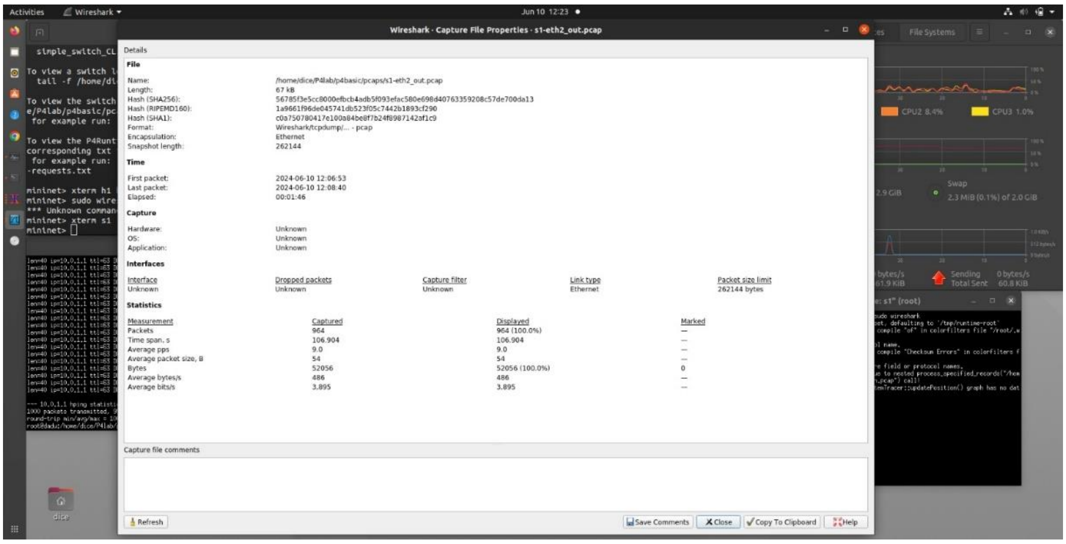
1. Tampilan *capture* dari setiap alur masuk dan keluar di setiap port



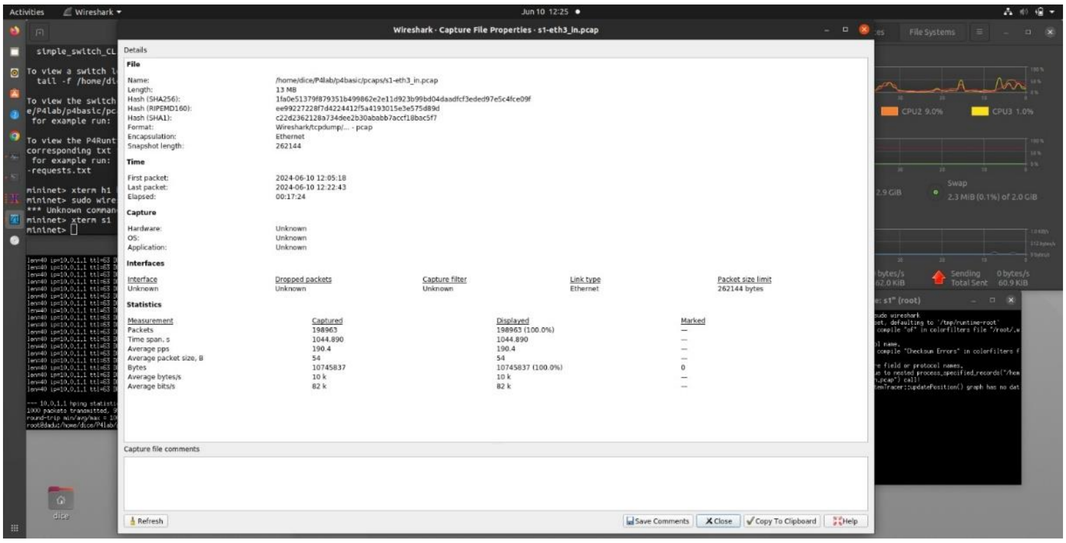
```
File
Name:                /home/dice/P4lab/p4basic/pcaps/s1-eth1_in.pcap
Length:              12 MB
Hash (SHA256):       4aba981c3dfe5b7bb8844dab0e1d4671f6a8c73ea65895859b3c04073525d42d
Hash (RIPEMD160):    52b34f59b3729dceb6e5ca2bd126048251ae1828
Hash (SHA1):         8e20cb67613c4351f4901aecc7507df9dc61e2c2
Format:              Wireshark/tcpdump/... - pcap
Encapsulation:       Ethernet
Snapshot length:     262144

Time
First packet:        2024-06-10 12:05:18
Last packet:         2024-06-10 12:09:34
Elapsed:             00:04:16

Capture
Hardware:            Unknown
OS:                  Unknown
Application:         Unknown

Interfaces
Interface    Dropped packets    Capture filter    Link type    Packet size limit
Unknown      Unknown            Unknown           Ethernet     262144 bytes

Statistics
Measurement               Captured      Displayed              Marked
Packets                   182733        182733 (100.0%)        —
Time span, s              256.004       256.004                —
Average pps               713.8         713.8                  —
Average packet size, B    54            54                     —
Bytes                     9869406       9869406 (100.0%)       0
Average bytes/s           38 k          38 k                   —
Average bits/s            308 k         308 k                  —
```



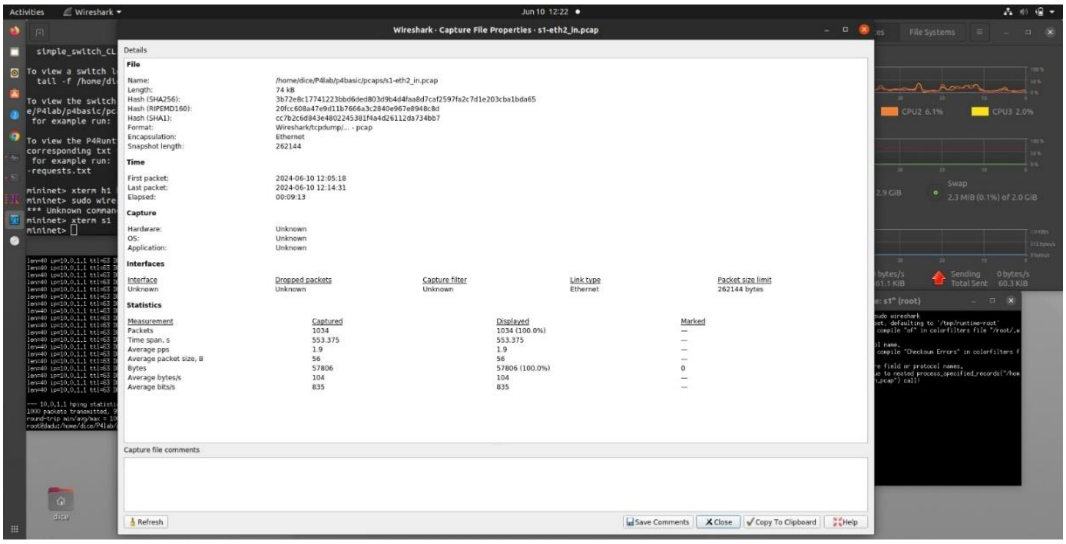
```
File
Name:                /home/dice/P4lab/p4basic/pcaps/s1-eth1_out.pcap
Length:              13 MB
Hash (SHA256):       b6b0271c6eda82b1116bed1b3facb689f4604a146a0a15623f6b48940632285f
Hash (RIPEMD160):    d85fb29092bbfb27bda19d13f19cbe688d5a2aac
Hash (SHA1):         9225d1cd9c3065ef9623f5f6051fc8a590004dcb
Format:              Wireshark/tcpdump/... - pcap
Encapsulation:       Ethernet
Snapshot length:     262144

Time
First packet:        2024-06-10 12:06:49
Last packet:         2024-06-10 12:08:46
Elapsed:             00:01:56

Capture
Hardware:            Unknown
OS:                  Unknown
Application:         Unknown

Interfaces
Interface    Dropped packets    Capture filter    Link type    Packet size limit
Unknown      Unknown            Unknown           Ethernet     262144 bytes

Statistics
Measurement               Captured      Displayed              Marked
Packets                   199137        199137 (100.0%)        —
Time span, s              116.666       116.666                —
Average pps               1706.9        1706.9                 —
Average packet size, B    54            54                     —
Bytes                     10753398      10753398 (100.0%)      0
Average bytes/s           92 k          92 k                   —
Average bits/s            737 k         737 k                  —
```



```
Wireshark · Capture File Properties · s1-eth2_out.pcap

Details
File
Name:                  /home/dice/P4lab/p4basic/pcaps/s1-eth2_out.pcap
Length:                67 kB
Hash (SHA256):         56785f3e5cc8000efbcb4adb5f093efac580e698d40763359208c57de700da13
Hash (RIPEMD160):      1a9661f96de045741db523f05c7442b1893cf290
Hash (SHA1):           c0a750780417e100a84be8f7b24f8987142af1c9
Format:                Wireshark/tcpdump/... - pcap
Encapsulation:         Ethernet
Snapshot length:       262144
Time
First packet:          2024-06-10 12:06:53
Last packet:           2024-06-10 12:08:40
Elapsed:               00:01:46
Capture
Hardware:              Unknown
OS:                    Unknown
Application:           Unknown
Interfaces
Interface              Dropped packets       Capture filter        Link type             Packet size limit
Unknown                Unknown               Unknown               Ethernet              262144 bytes
Statistics
Measurement                 Captured            Displayed              Marked
Packets                     964                 964 (100.0%)           —
Time span, s                106.904             106.904                —
Average pps                 9.0                 9.0                    —
Average packet size, B      54                  54                     —
Bytes                       52056               52056 (100.0%)         0
Average bytes/s             486                 486                    —
Average bits/s              3.895               3.895                  —

Capture file comments
```

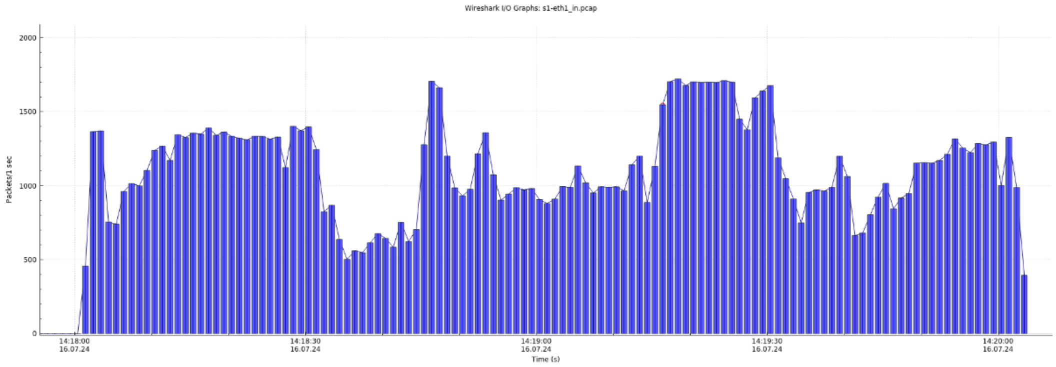


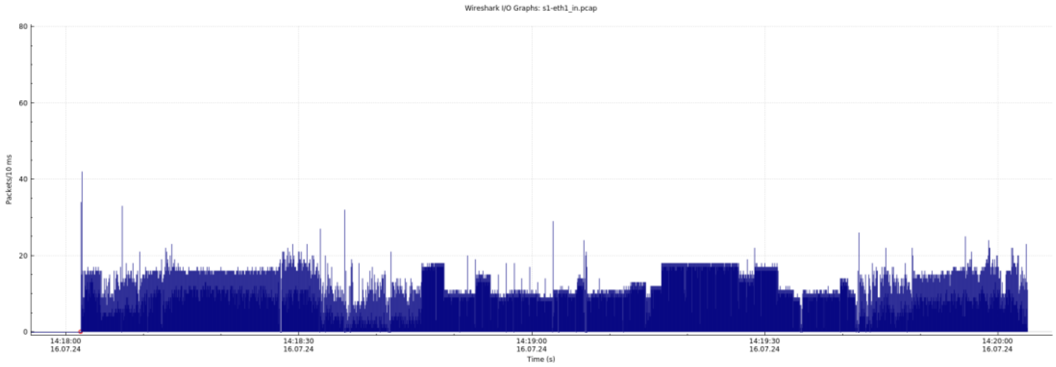
```
Wireshark · Capture File Properties · s1-eth3_in.pcap

Details
File
Name:                  /home/dice/P4lab/p4basic/pcaps/s1-eth3_in.pcap
Length:                13 MB
Hash (SHA256):         1fa0e51379f879351b499862e2e11d923b99bd04daadfcf3eded97e5c4fce09f
Hash (RIPEMD160):      ee99227228f7d4224412f5a4193015e3e575d89d
Hash (SHA1):           c22d2362128a734dee2b30ababb7accf18bac5f7
Format:                Wireshark/tcpdump/... - pcap
Encapsulation:         Ethernet
Snapshot length:       262144
Time
First packet:          2024-06-10 12:05:18
Last packet:           2024-06-10 12:22:43
Elapsed:               00:17:24
Capture
Hardware:              Unknown
OS:                    Unknown
Application:           Unknown
Interfaces
Interface              Dropped packets       Capture filter        Link type             Packet size limit
Unknown                Unknown               Unknown               Ethernet              262144 bytes
Statistics
Measurement                 Captured            Displayed              Marked
Packets                     198963              198963 (100.0%)        —
Time span, s                1044.890            1044.890               —
Average pps                 190.4               190.4                  —
Average packet size, B      54                  54                     —
Bytes                       10745837            10745837 (100.0%)      0
Average bytes/s             10 k                10 k                   —
Average bits/s              82 k                82 k                   —

Capture file comments
```

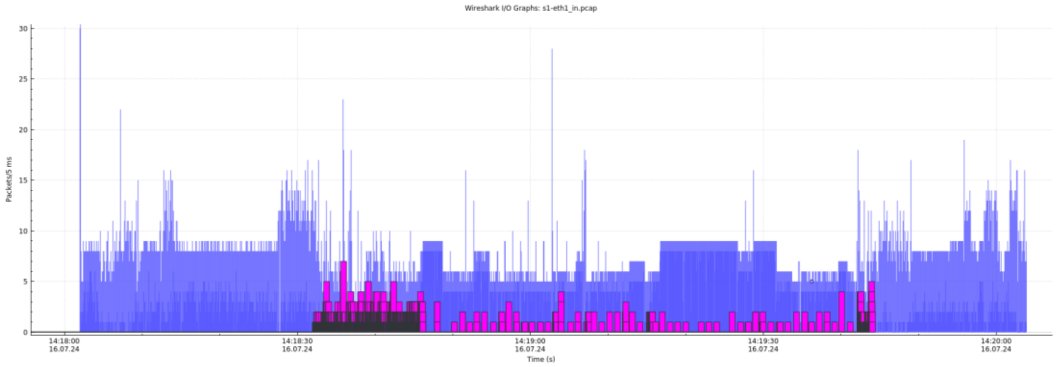


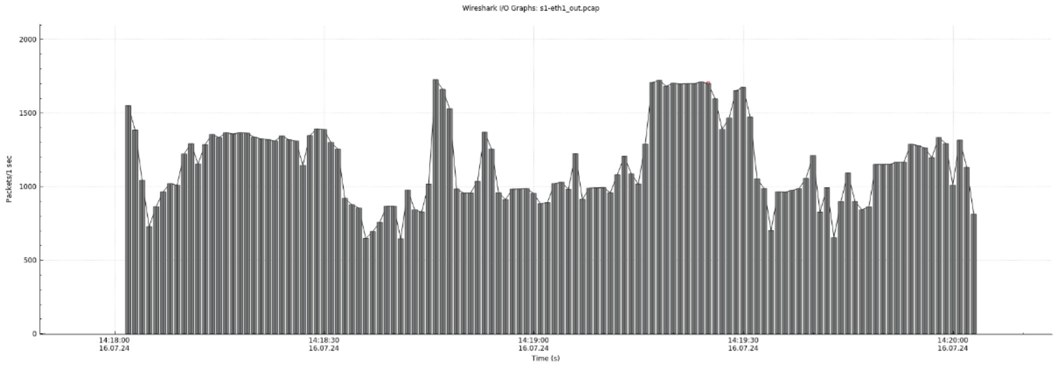
```
Wireshark · Capture File Properties · s1-eth2_in.pcap

Details
File
Name:                  /home/dice/P4lab/p4basic/pcaps/s1-eth2_in.pcap
Length:                74 kB
Hash (SHA256):         3b72e8c17741223bbd6ded803d9b4d4faa8d7caf2597fa2c7d1e203cba1bda65
Hash (RIPEMD160):      20fcc608a47e9d11b7666a3c2840e967e8948c8d
Hash (SHA1):           cc7b2c6d843e4802245381f4a4d26112da734bb7
Format:                Wireshark/tcpdump/... - pcap
Encapsulation:         Ethernet
Snapshot length:       262144
Time
First packet:          2024-06-10 12:05:18
Last packet:           2024-06-10 12:14:31
Elapsed:               00:09:13
Capture
Hardware:              Unknown
OS:                    Unknown
Application:           Unknown
Interfaces
Interface              Dropped packets       Capture filter        Link type             Packet size limit
Unknown                Unknown               Unknown               Ethernet              262144 bytes
Statistics
Measurement                 Captured            Displayed              Marked
Packets                     1034                1034 (100.0%)          —
Time span, s                553.375             553.375                —
Average pps                 1.9                 1.9                    —
Average packet size, B      56                  56                     —
Bytes                       57806               57806 (100.0%)         0
Average bytes/s             104                 104                    —
Average bits/s              835                 835                    —

Capture file comments
```

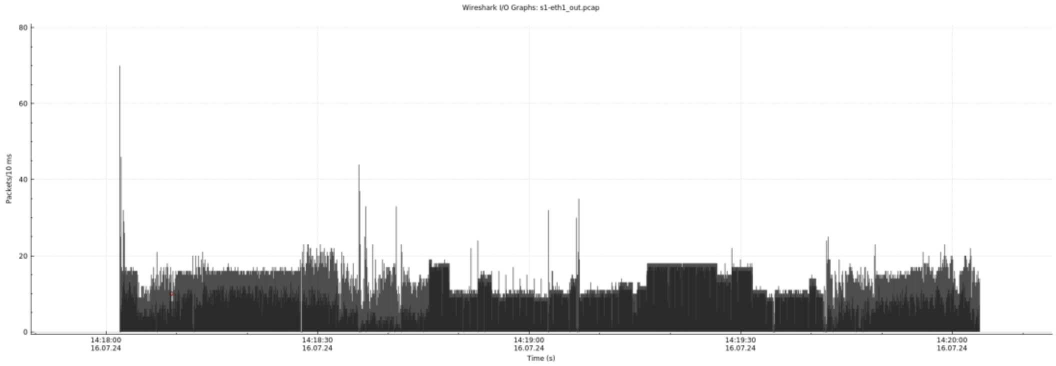
2. Gambar grafik dari setiap alur masuk dan keluar dari setiap port pada switch

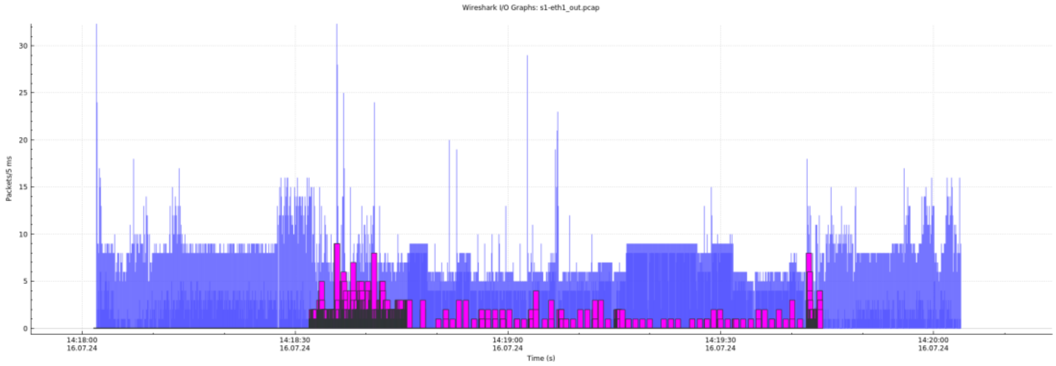


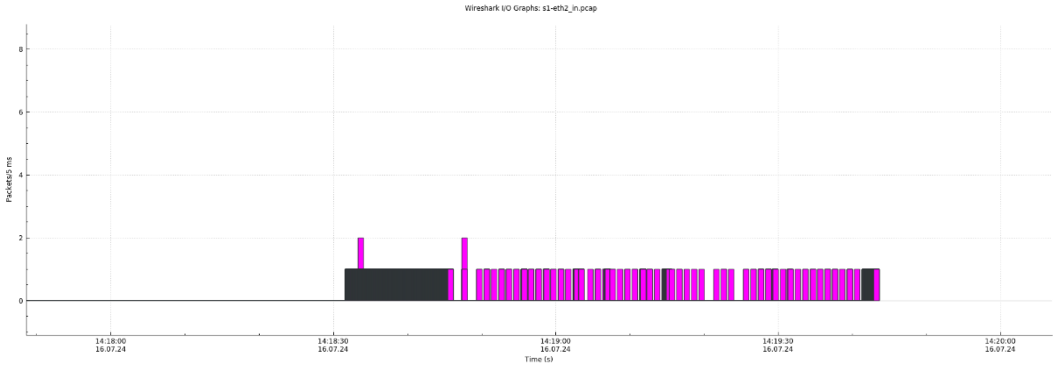


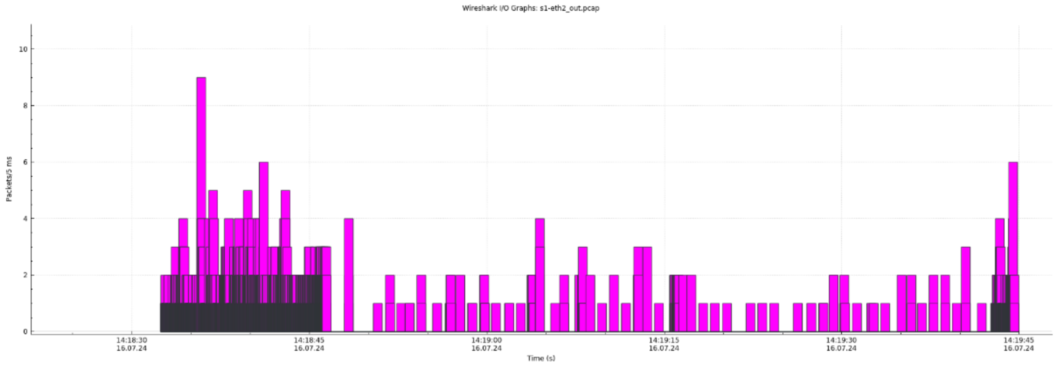


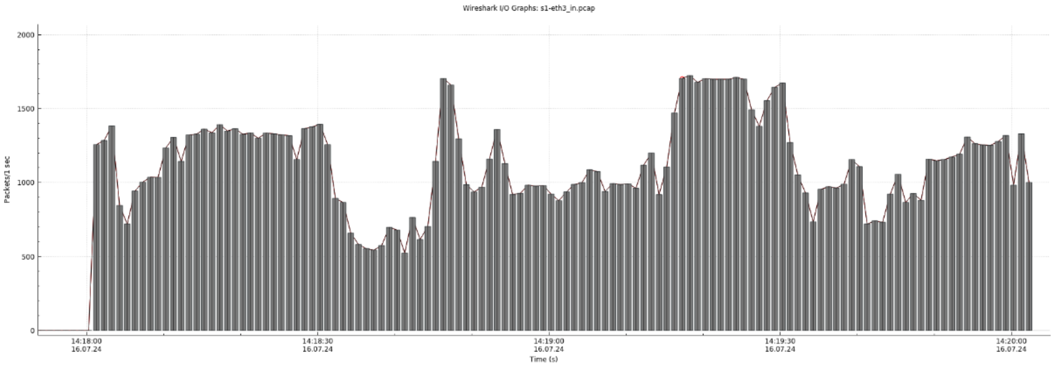


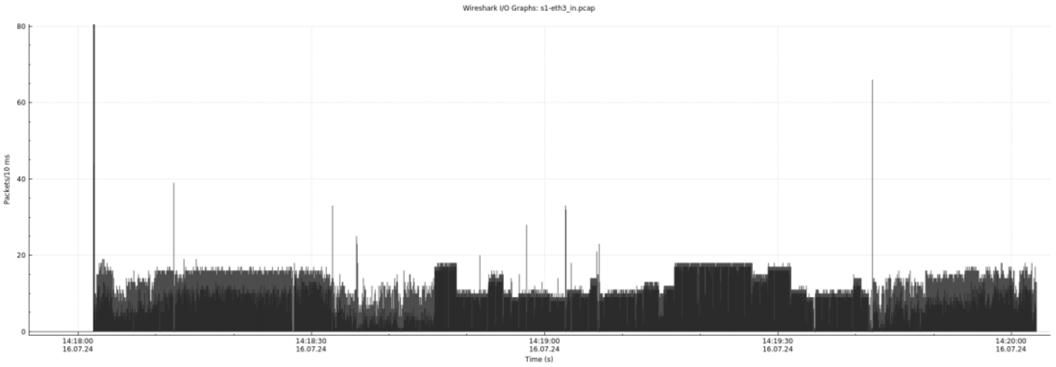

Wireshark I/O Graphs: s1-eth1_out.pcap
Packets/10 ms
0
20
40
60
80
14:18:00 16.07.24
14:18:30 16.07.24
14:19:00 16.07.24
14:19:30 16.07.24
14:20:00 16.07.24
Time (s)

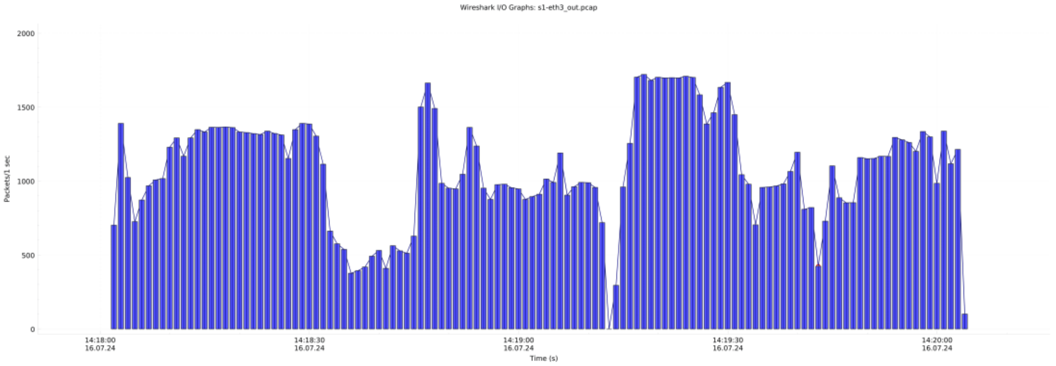

Wireshark I/O Graphs: s1-eth1_out.pcap
Packets/5 ms
0
5
10
15
20
25
30
14:18:00 16.07.24
14:18:30 16.07.24
14:19:00 16.07.24
14:19:30 16.07.24
14:20:00 16.07.24
Time (s)

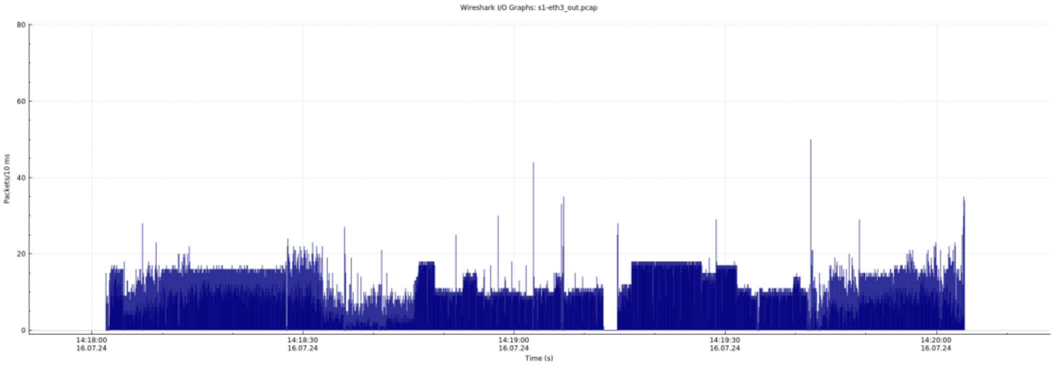

Wireshark I/O Graphs: s1-eth2_in.pcap
Packets/5 ms
0
2
4
6
8
14:18:00 16.07.24
14:18:30 16.07.24
14:19:00 16.07.24
14:19:30 16.07.24
14:20:00 16.07.24
Time (s)


Wireshark I/O Graphs: s1-eth2_out.pcap
Packets/5 ms
0
2
4
6
8
10
14:18:30 16.07.24
14:18:45 16.07.24
14:19:00 16.07.24
14:19:15 16.07.24
14:19:30 16.07.24
14:19:45 16.07.24
Time (s)


Wireshark I/O Graphs: s1-eth3_in.pcap
Packets/1 sec
2000
1500
1000
500
0
14:18:00 16.07.24
14:18:30 16.07.24
14:19:00 16.07.24
14:19:30 16.07.24
14:20:00 16.07.24
Time (s)

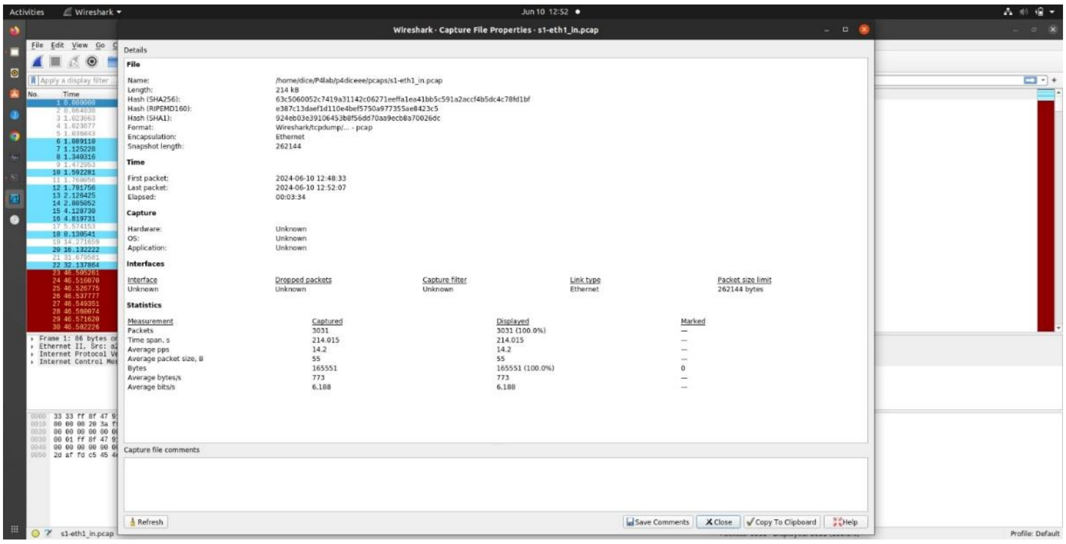

Wireshark I/O Graphs: s1-eth3_in.pcap
Packets/10 ms
80
60
40
20
0
14:18:00 16.07.24
14:18:30 16.07.24
14:19:00 16.07.24
14:19:30 16.07.24
14:20:00 16.07.24
Time (s)


Wireshark I/O Graphs: s1-eth3_out.pcap
Packets/1 sec
2000
1500
1000
500
0
14:18:00 16.07.24
14:18:30 16.07.24
14:19:00 16.07.24
14:19:30 16.07.24
14:20:00 16.07.24
Time (s)

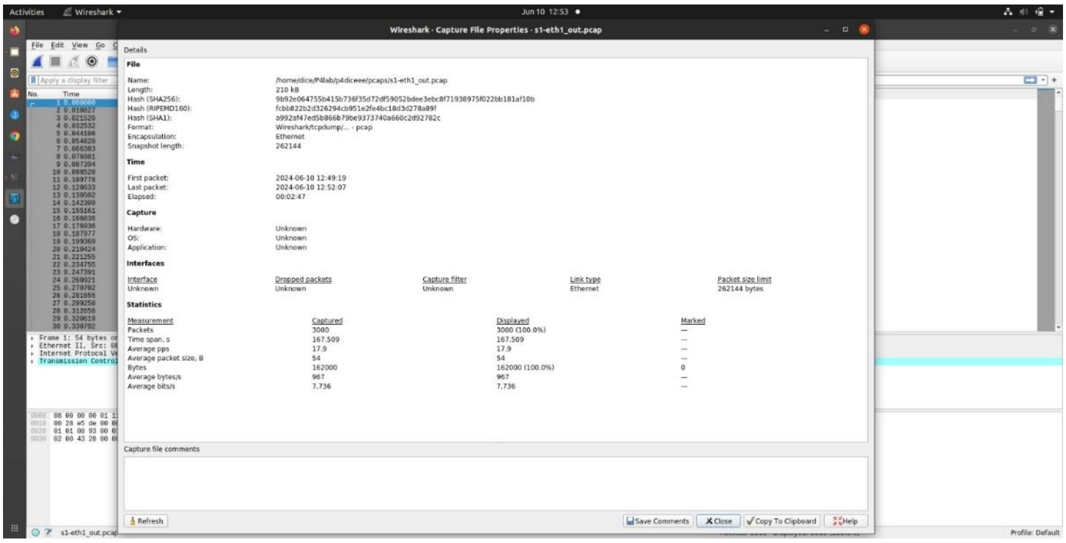

Wireshark I/O Graphs: s1-eth3_out.pcap
Packets/10 ms
80
60
40
20
0
14:18:00 16.07.24
14:18:30 16.07.24
14:19:00 16.07.24
14:19:30 16.07.24
14:20:00 16.07.24
Time (s)

Lampiran 8 Wireshark Capture Setelah Implementasi Metode Keamanan

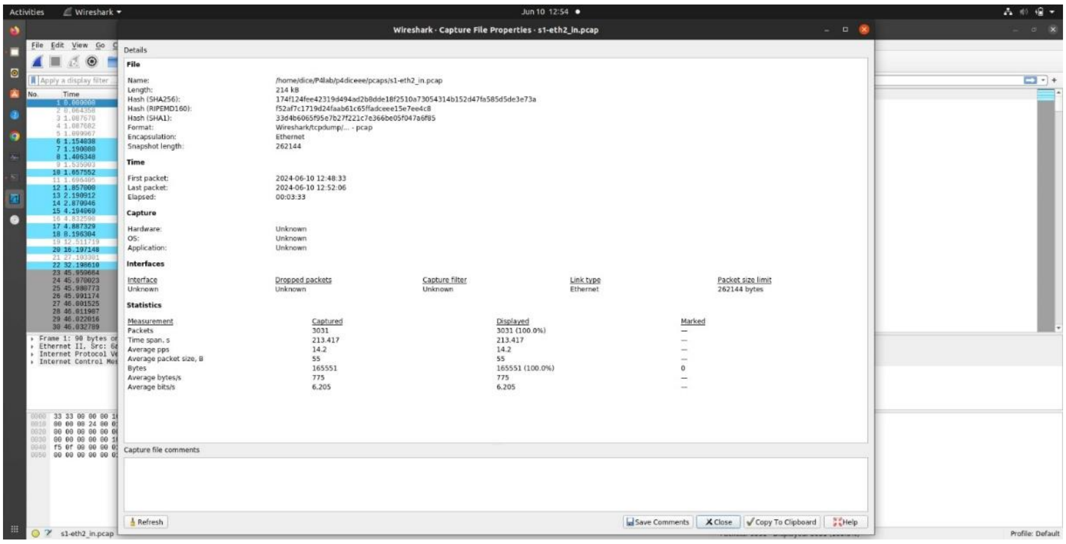
1. Tampilan *capture* dari setiap alur masuk dan keluar di setiap port



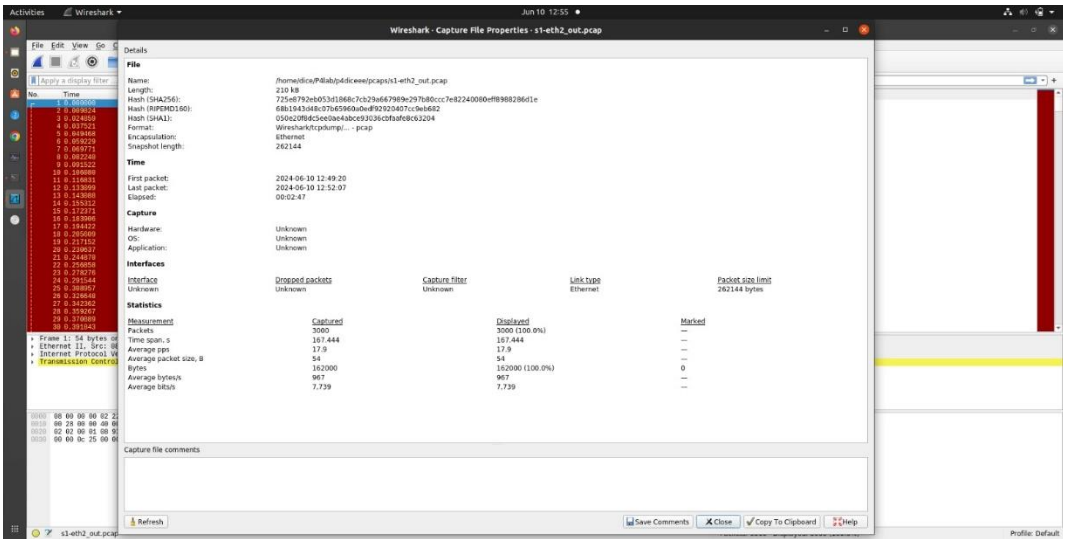
```
Details
File
Name:                 /home/dice/P4lab/p4diceee/pcaps/s1-eth1_in.pcap
Length:               214 kB
Hash (SHA256):        63c5060052c7419a31142c06271eeffa1ea41bb5c591a2accf4b5dc4c78fd1bf
Hash (RIPEMD160):     e387c13daef1d110e4bef5750a977355ae8423c5
Hash (SHA1):          924eb03e39106453b8f56dd70aa9ecb8a70026dc
Format:               Wireshark/tcpdump/... - pcap
Encapsulation:        Ethernet
Snapshot length:      262144
Time
First packet:         2024-06-10 12:48:33
Last packet:          2024-06-10 12:52:07
Elapsed:              00:03:34
Capture
Hardware:             Unknown
OS:                   Unknown
Application:          Unknown
Interfaces
Interface             Dropped packets      Capture filter      Link type      Packet size limit
Unknown               Unknown              Unknown             Ethernet       262144 bytes
Statistics
Measurement               Captured       Displayed            Marked
Packets                   3031           3031 (100.0%)        —
Time span, s              214.015        214.015              —
Average pps               14.2           14.2                 —
Average packet size, B    55             55                   —
Bytes                     165551         165551 (100.0%)      0
Average bytes/s           773            773                  —
Average bits/s            6.188          6.188                —
Capture file comments
```



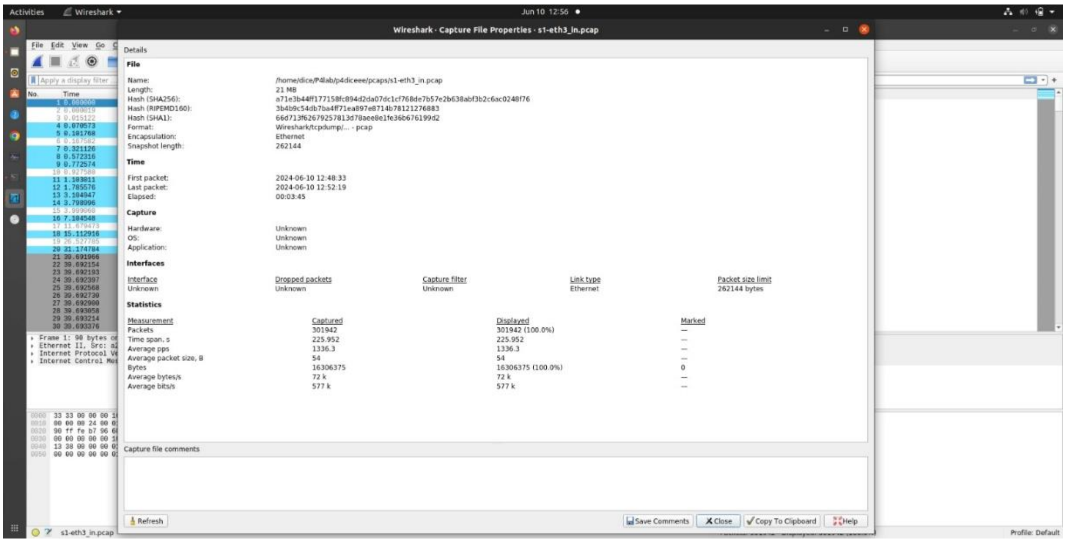
```
Details
File
Name:                 /home/dice/P4lab/p4diceee/pcaps/s1-eth1_out.pcap
Length:               210 kB
Hash (SHA256):        9b92e064755b415b736f35d72df59052bdee3ebc8f71938975f022bb181af10b
Hash (RIPEMD160):     fcbb822b2d326294cb951e2fe4bc18d3d278a89f
Hash (SHA1):          a992af47ed5b866b79be9373740a660c2d92782c
Format:               Wireshark/tcpdump/... - pcap
Encapsulation:        Ethernet
Snapshot length:      262144
Time
First packet:         2024-06-10 12:49:19
Last packet:          2024-06-10 12:52:07
Elapsed:              00:02:47
Capture
Hardware:             Unknown
OS:                   Unknown
Application:          Unknown
Interfaces
Interface             Dropped packets      Capture filter      Link type      Packet size limit
Unknown               Unknown              Unknown             Ethernet       262144 bytes
Statistics
Measurement               Captured       Displayed            Marked
Packets                   3000           3000 (100.0%)        —
Time span, s              167.509        167.509              —
Average pps               17.9           17.9                 —
Average packet size, B    54             54                   —
Bytes                     162000         162000 (100.0%)      0
Average bytes/s           967            967                  —
Average bits/s            7,736          7,736                —
Capture file comments
```



```
Details
File
Name:                 /home/dice/P4lab/p4diceee/pcaps/s1-eth2_in.pcap
Length:               214 kB
Hash (SHA256):        174f124fee42319d494ad2b8dde18f2510a73054314b152d47fa585d5de3e73a
Hash (RIPEMD160):     f52af7c1719d24faab61c65ffadceee15e7ee4c8
Hash (SHA1):          33d4b6065f95e7b27f221c7e366be05f047a6f85
Format:               Wireshark/tcpdump/... - pcap
Encapsulation:        Ethernet
Snapshot length:      262144
Time
First packet:         2024-06-10 12:48:33
Last packet:          2024-06-10 12:52:06
Elapsed:              00:03:33
Capture
Hardware:             Unknown
OS:                   Unknown
Application:          Unknown
Interfaces
Interface             Dropped packets      Capture filter      Link type      Packet size limit
Unknown               Unknown              Unknown             Ethernet       262144 bytes
Statistics
Measurement               Captured       Displayed            Marked
Packets                   3031           3031 (100.0%)        —
Time span, s              213.417        213.417              —
Average pps               14.2           14.2                 —
Average packet size, B    55             55                   —
Bytes                     165551         165551 (100.0%)      0
Average bytes/s           775            775                  —
Average bits/s            6.205          6.205                —
Capture file comments
```

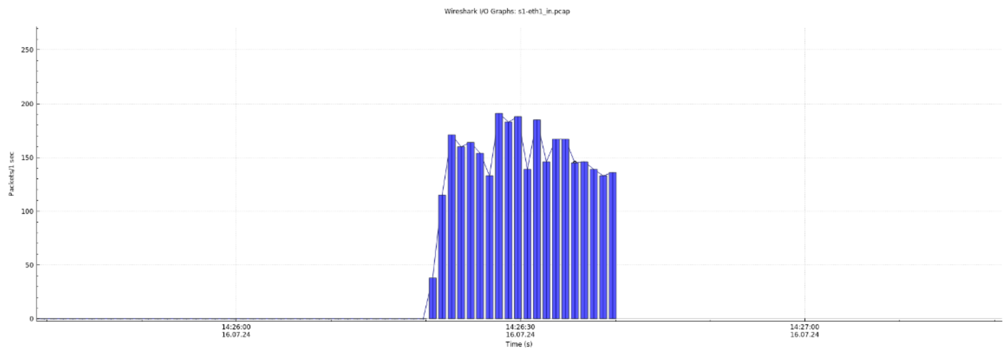


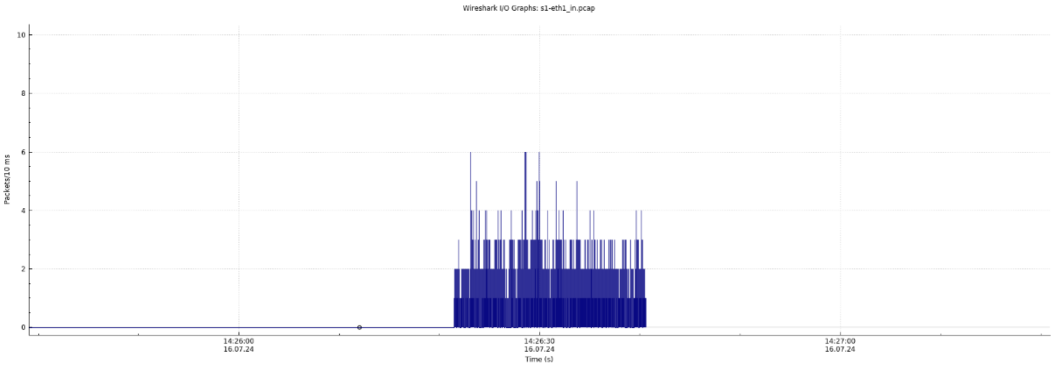
```
Wireshark · Capture File Properties · s1-eth2_out.pcap

Details
File
Name:                  /home/dice/P4lab/p4diceee/pcaps/s1-eth2_out.pcap
Length:                210 kB
Hash (SHA256):         725e8792eb053d1868c7cb29a667989e297b80ccc7e82240080eff8988286d1e
Hash (RIPEMD160):      68b1943d48c07b65960a0edf92920407cc9eb682
Hash (SHA1):           050e20f8dc5ee0ae4abce93036cbfaafe8c63204
Format:                Wireshark/tcpdump/... - pcap
Encapsulation:         Ethernet
Snapshot length:       262144

Time
First packet:          2024-06-10 12:49:20
Last packet:           2024-06-10 12:52:07
Elapsed:               00:02:47

Capture
Hardware:              Unknown
OS:                    Unknown
Application:           Unknown

Interfaces
Interface     Dropped packets     Capture filter     Link type     Packet size limit
Unknown       Unknown             Unknown            Ethernet      262144 bytes

Statistics
Measurement                 Captured      Displayed           Marked
Packets                     3000          3000 (100.0%)       —
Time span, s                167.444       167.444             —
Average pps                 17.9          17.9                —
Average packet size, B      54            54                  —
Bytes                       162000        162000 (100.0%)     0
Average bytes/s             967           967                 —
Average bits/s              7,739         7,739               —

Capture file comments
```

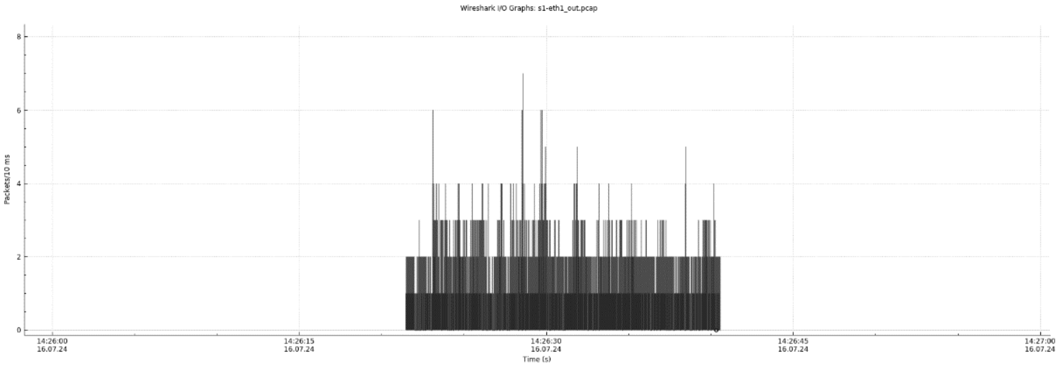


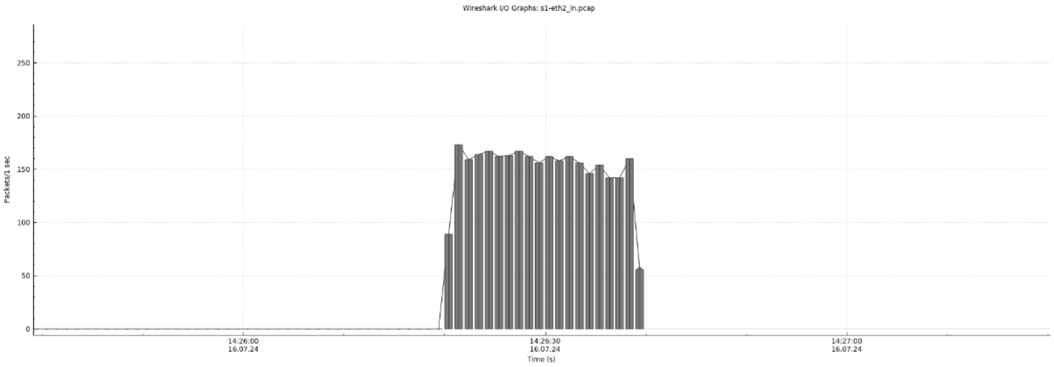
```
Wireshark · Capture File Properties · s1-eth3_in.pcap

Details
File
Name:                  /home/dice/P4lab/p4diceee/pcaps/s1-eth3_in.pcap
Length:                21 MB
Hash (SHA256):         a71e3b44ff177158fc894d2da07dc1cf768de7b57e2b638abf3b2c6ac0248f76
Hash (RIPEMD160):      3b4b9c54db7ba4ff71ea897e8714b78121276883
Hash (SHA1):           66d713f62679257813d78aee8e1fe36b676199d2
Format:                Wireshark/tcpdump/... - pcap
Encapsulation:         Ethernet
Snapshot length:       262144

Time
First packet:          2024-06-10 12:48:33
Last packet:           2024-06-10 12:52:19
Elapsed:               00:03:45

Capture
Hardware:              Unknown
OS:                    Unknown
Application:           Unknown

Interfaces
Interface     Dropped packets     Capture filter     Link type     Packet size limit
Unknown       Unknown             Unknown            Ethernet      262144 bytes

Statistics
Measurement                 Captured      Displayed            Marked
Packets                     301942        301942 (100.0%)      —
Time span, s                225.952       225.952              —
Average pps                 1336.3        1336.3               —
Average packet size, B      54            54                   —
Bytes                       16306375      16306375 (100.0%)    0
Average bytes/s             72 k          72 k                 —
Average bits/s              577 k         577 k                —

Capture file comments
```

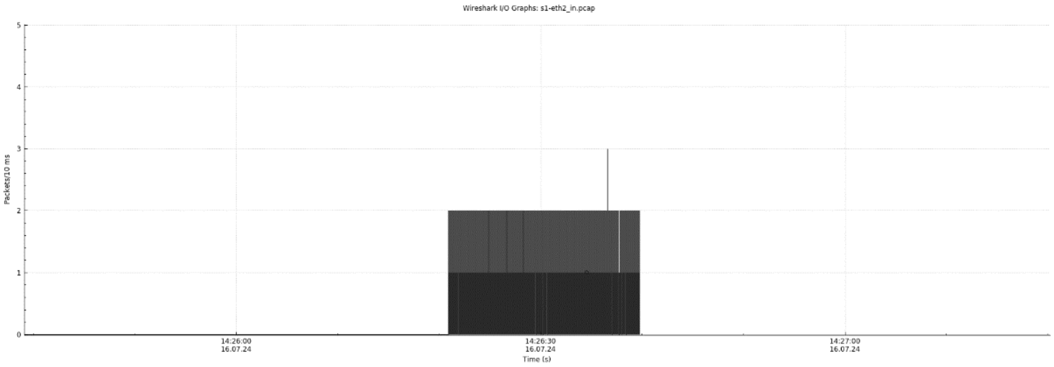
2. Gambar grafik dari setiap alur masuk dan keluar dari setiap port pada switch

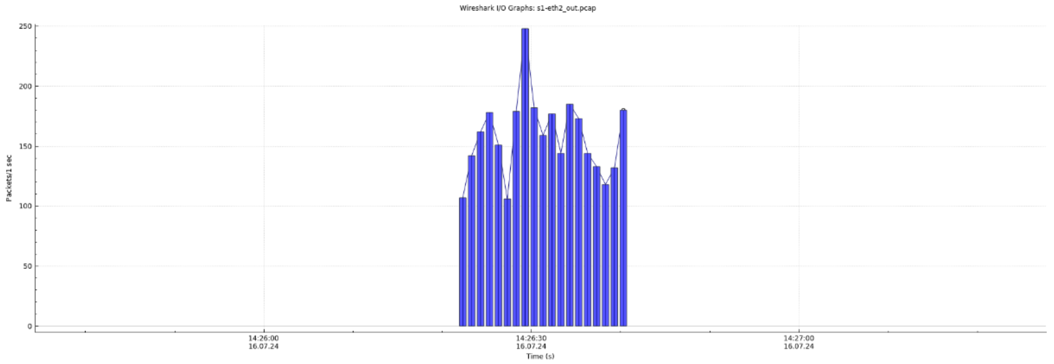


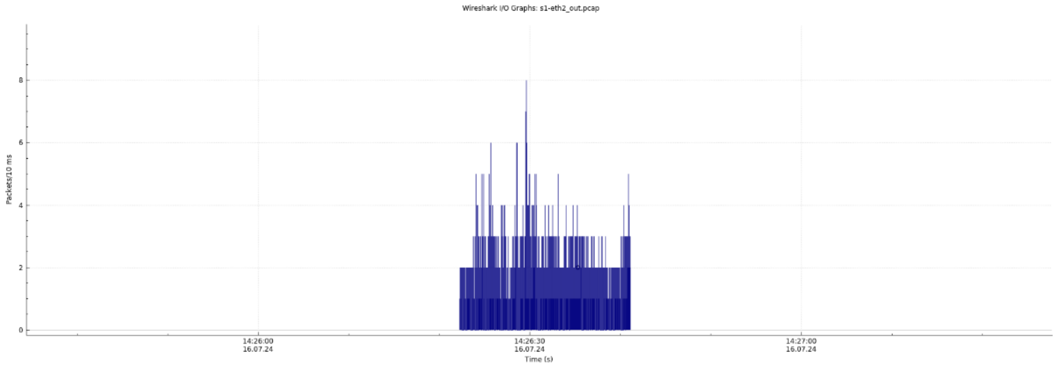

Wireshark I/O Graphs: s1-eth1_in.pcap
Packets/10 ms
0
2
4
6
8
10
14:26:00
16.07.24
14:26:30
16.07.24
14:27:00
16.07.24
Time (s)

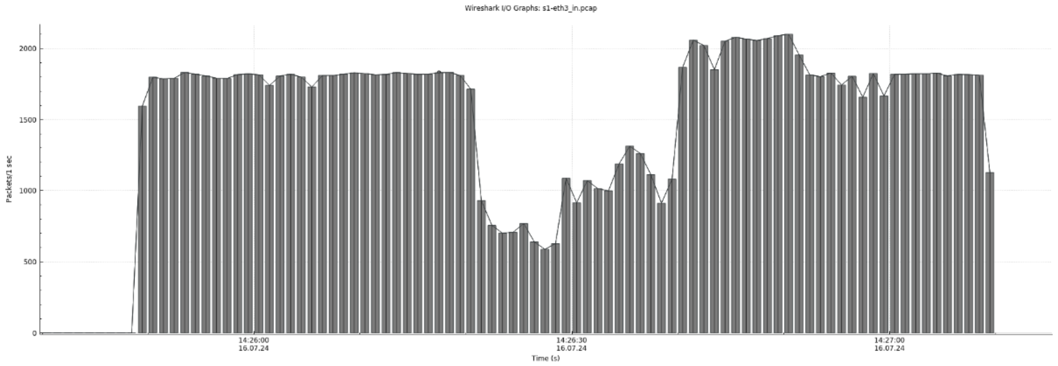

Wireshark I/O Graphs: s1-eth1_out.pcap
Packets/10 ms
0
2
4
6
8
14:26:00
16.07.24
14:26:15
16.07.24
14:26:30
16.07.24
14:26:45
16.07.24
14:27:00
16.07.24
Time (s)

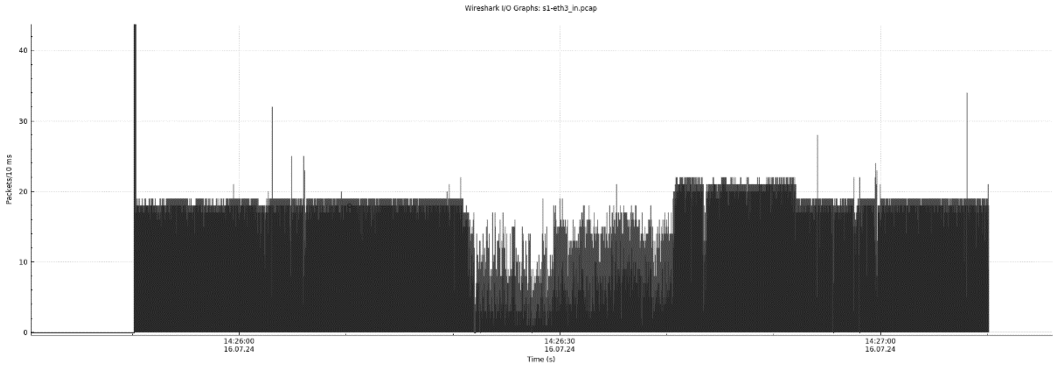

Wireshark I/O Graphs: s1-eth2_in.pcap
Packets/1 sec
0
50
100
150
200
250
14:26:00
16.07.24
14:26:30
16.07.24
14:27:00
16.07.24
Time (s)


Wireshark I/O Graphs: s1-eth2_in.pcap
Packets/10 ms
0
1
2
3
4
5
14:26:00
16.07.24
14:26:30
16.07.24
14:27:00
16.07.24
Time (s)

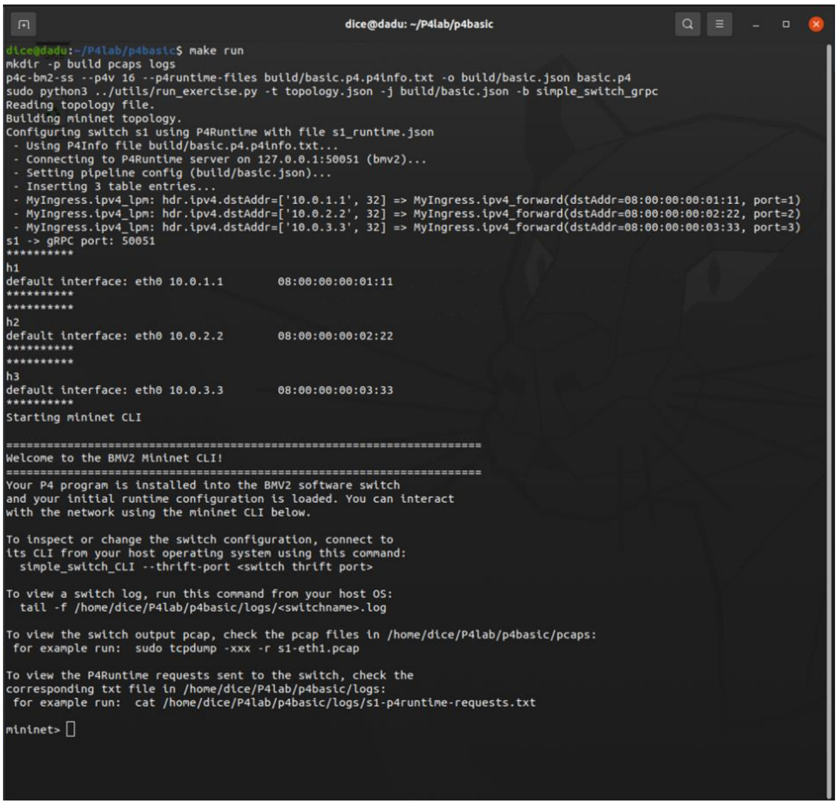

Wireshark I/O Graphs: s1-eth2_out.pcap
Packets/1 sec
250
200
150
100
50
0
14:26:00
16.07.24
14:26:30
16.07.24
14:27:00
16.07.24
Time (s)


Wireshark I/O Graphs: s1-eth2_out.pcap
Packets/10 ms
8
6
4
2
0
14:26:00
16.07.24
14:26:30
16.07.24
14:27:00
16.07.24
Time (s)

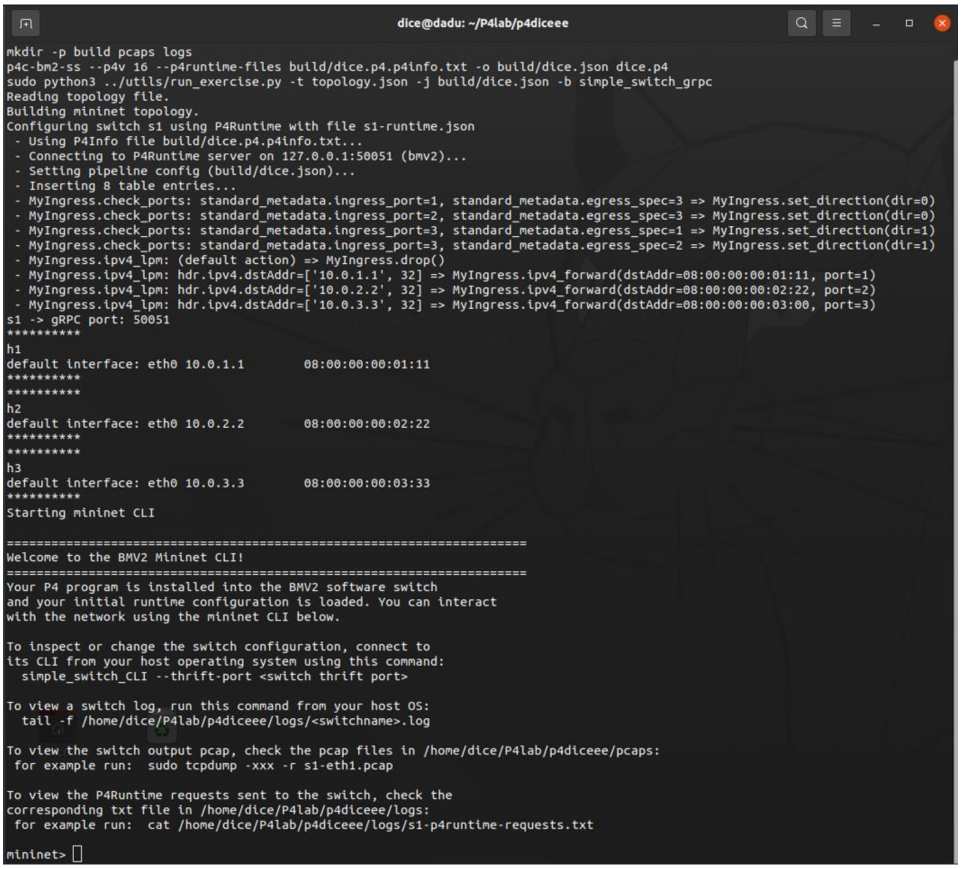

Wireshark I/O Graphs: s1-eth3_in.pcap
Packets/1 sec
2000
1500
1000
500
0
14:26:00
16.07.24
14:26:30
16.07.24
14:27:00
16.07.24
Time (s)


Wireshark I/O Graphs: s1-eth3_in.pcap
Packets/10 ms
40
30
20
10
0
14:26:00
16.07.24
14:26:30
16.07.24
14:27:00
16.07.24
Time (s)

Lampiran 9 Mininet Configuration Capture



```
dice@dadu:~/P4lab/p4basic$ make run
mkdir -p build pcaps logs
p4c-bm2-ss --p4v 16 --p4runtime-files build/basic.p4.p4info.txt -o build/basic.json basic.p4
sudo python3 ../utils/run_exercise.py -t topology.json -j build/basic.json -b simple_switch_grpc
Reading topology file.
Building mininet topology.
Configuring switch s1 using P4Runtime with file s1_runtime.json
 - Using P4Info file build/basic.p4.p4info.txt...
 - Connecting to P4Runtime server on 127.0.0.1:50051 (bmv2)...
 - Setting pipeline config (build/basic.json)...
 - Inserting 3 table entries...
 - MyIngress.ipv4_lpm: hdr.ipv4.dstAddr=['10.0.1.1', 32] => MyIngress.ipv4_forward(dstAddr=08:00:00:00:01:11, port=1)
 - MyIngress.ipv4_lpm: hdr.ipv4.dstAddr=['10.0.2.2', 32] => MyIngress.ipv4_forward(dstAddr=08:00:00:00:02:22, port=2)
 - MyIngress.ipv4_lpm: hdr.ipv4.dstAddr=['10.0.3.3', 32] => MyIngress.ipv4_forward(dstAddr=08:00:00:00:03:33, port=3)
s1 -> gRPC port: 50051
**********
h1
default interface: eth0 10.0.1.1        08:00:00:00:01:11
**********
**********
h2
default interface: eth0 10.0.2.2        08:00:00:00:02:22
**********
**********
h3
default interface: eth0 10.0.3.3        08:00:00:00:03:33
**********
Starting mininet CLI

======================================================================
Welcome to the BMV2 Mininet CLI!
======================================================================
Your P4 program is installed into the BMV2 software switch
and your initial runtime configuration is loaded. You can interact
with the network using the mininet CLI below.

To inspect or change the switch configuration, connect to
its CLI from your host operating system using this command:
  simple_switch_CLI --thrift-port <switch thrift port>

To view a switch log, run this command from your host OS:
  tail -f /home/dice/P4lab/p4basic/logs/<switchname>.log

To view the switch output pcap, check the pcap files in /home/dice/P4lab/p4basic/pcaps:
 for example run:  sudo tcpdump -xxx -r s1-eth1.pcap

To view the P4Runtime requests sent to the switch, check the
corresponding txt file in /home/dice/P4lab/p4basic/logs:
 for example run:  cat /home/dice/P4lab/p4basic/logs/s1-p4runtime-requests.txt

mininet> 
```



```
mkdir -p build pcaps logs
p4c-bm2-ss --p4v 16 --p4runtime-files build/dice.p4.p4info.txt -o build/dice.json dice.p4
sudo python3 ../utils/run_exercise.py -t topology.json -j build/dice.json -b simple_switch_grpc
Reading topology file.
Building mininet topology.
Configuring switch s1 using P4Runtime with file s1-runtime.json
 - Using P4Info file build/dice.p4.p4info.txt...
 - Connecting to P4Runtime server on 127.0.0.1:50051 (bmv2)...
 - Setting pipeline config (build/dice.json)...
 - Inserting 8 table entries...
 - MyIngress.check_ports: standard_metadata.ingress_port=1, standard_metadata.egress_spec=3 => MyIngress.set_direction(dir=0)
 - MyIngress.check_ports: standard_metadata.ingress_port=2, standard_metadata.egress_spec=3 => MyIngress.set_direction(dir=0)
 - MyIngress.check_ports: standard_metadata.ingress_port=3, standard_metadata.egress_spec=1 => MyIngress.set_direction(dir=1)
 - MyIngress.check_ports: standard_metadata.ingress_port=3, standard_metadata.egress_spec=2 => MyIngress.set_direction(dir=1)
 - MyIngress.ipv4_lpm: (default action) => MyIngress.drop()
 - MyIngress.ipv4_lpm: hdr.ipv4.dstAddr=['10.0.1.1', 32] => MyIngress.ipv4_forward(dstAddr=08:00:00:00:01:11, port=1)
 - MyIngress.ipv4_lpm: hdr.ipv4.dstAddr=['10.0.2.2', 32] => MyIngress.ipv4_forward(dstAddr=08:00:00:00:02:22, port=2)
 - MyIngress.ipv4_lpm: hdr.ipv4.dstAddr=['10.0.3.3', 32] => MyIngress.ipv4_forward(dstAddr=08:00:00:00:03:00, port=3)
s1 -> gRPC port: 50051
**********
h1
default interface: eth0 10.0.1.1        08:00:00:00:01:11
**********
**********
h2
default interface: eth0 10.0.2.2        08:00:00:00:02:22
**********
**********
h3
default interface: eth0 10.0.3.3        08:00:00:00:03:33
**********
Starting mininet CLI

======================================================================
Welcome to the BMV2 Mininet CLI!
======================================================================
Your P4 program is installed into the BMV2 software switch
and your initial runtime configuration is loaded. You can interact
with the network using the mininet CLI below.

To inspect or change the switch configuration, connect to
its CLI from your host operating system using this command:
  simple_switch_CLI --thrift-port <switch thrift port>

To view a switch log, run this command from your host OS:
  tail -f /home/dice/P4lab/p4diceee/logs/<switchname>.log

To view the switch output pcap, check the pcap files in /home/dice/P4lab/p4diceee/pcaps:
 for example run:  sudo tcpdump -xxx -r s1-eth1.pcap

To view the P4Runtime requests sent to the switch, check the
corresponding txt file in /home/dice/P4lab/p4diceee/logs:
 for example run:  cat /home/dice/P4lab/p4diceee/logs/s1-p4runtime-requests.txt

mininet> 
```