



Cybersecurity (Security+) and P4 Programmable Switches

Lab 8: Detecting and Mitigating the DNS Amplification Attack

Ali AlSabeH, Jorge Crichigno
University of South Carolina
<http://ce.sc.edu/cyberinfra>

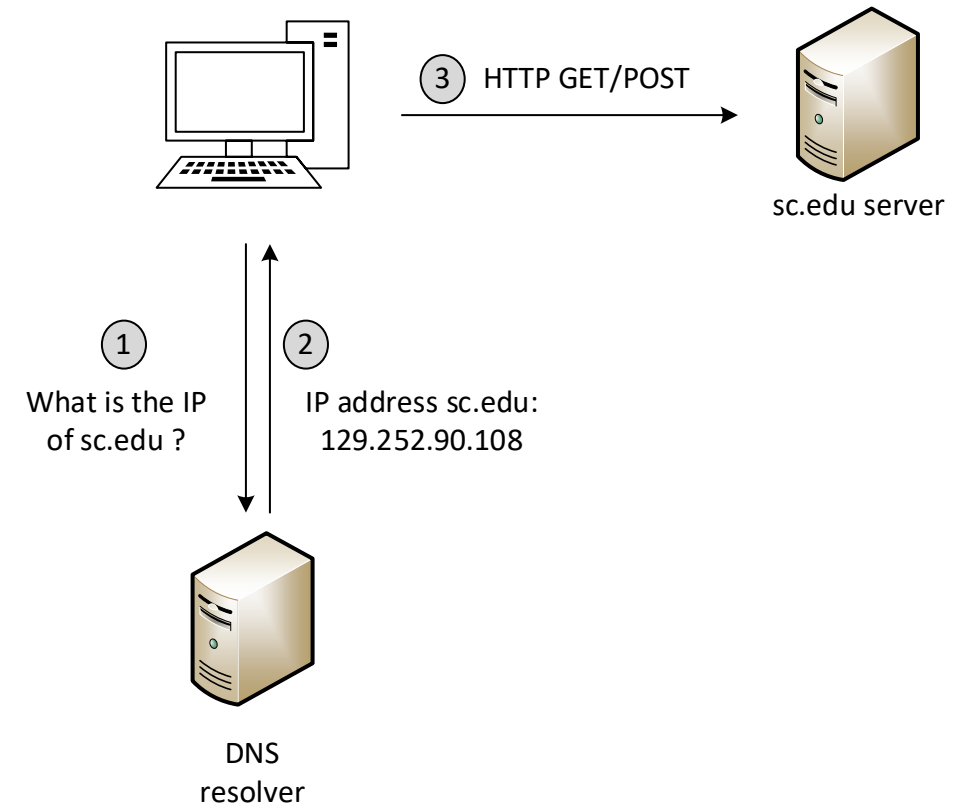
Western Academy Support and Training Center (WASTC)
University of South Carolina (USC)
Energy Sciences Network (ESnet)

June 22nd, 2023

Lab 8: Detecting and Mitigating the DNS Amplification Attack

Domain Name System (DNS)

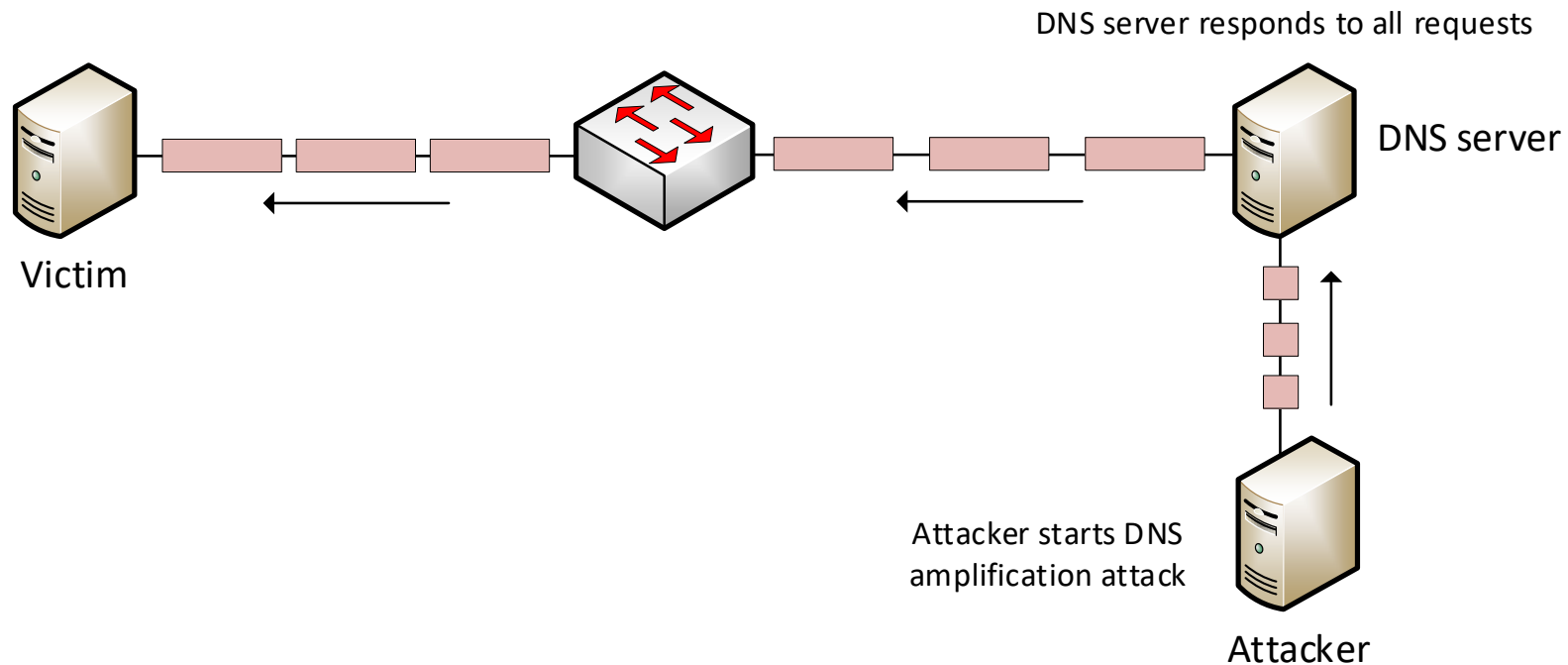
- The Domain Name System (DNS) is an essential component of the internet
- It is responsible for translating human-readable domain names into IP addresses that can be understood by devices connected to the internet¹



¹ Amazon, "What is DNS?" [Online]. Available: <https://tinyurl.com/ynb9esn6>

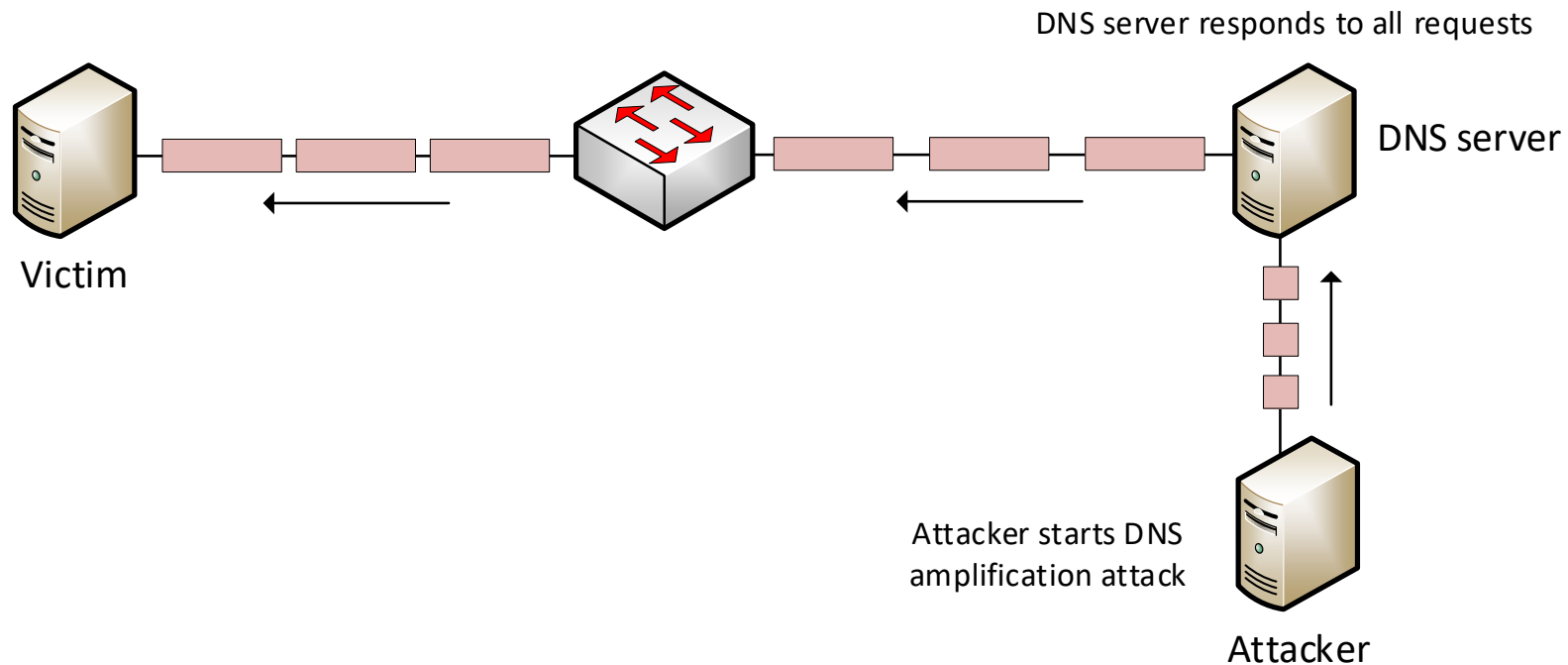
DNS Amplification

- Due its frequent use on the Internet, the DNS is susceptible to many attacks
- DNS amplification is an attack that exploits the infrastructure of the DNS to create a Distributed Denial of Service (DDoS) on a target victim
 - Thus, rendering the victim unresponsive due to the large amount of traffic received



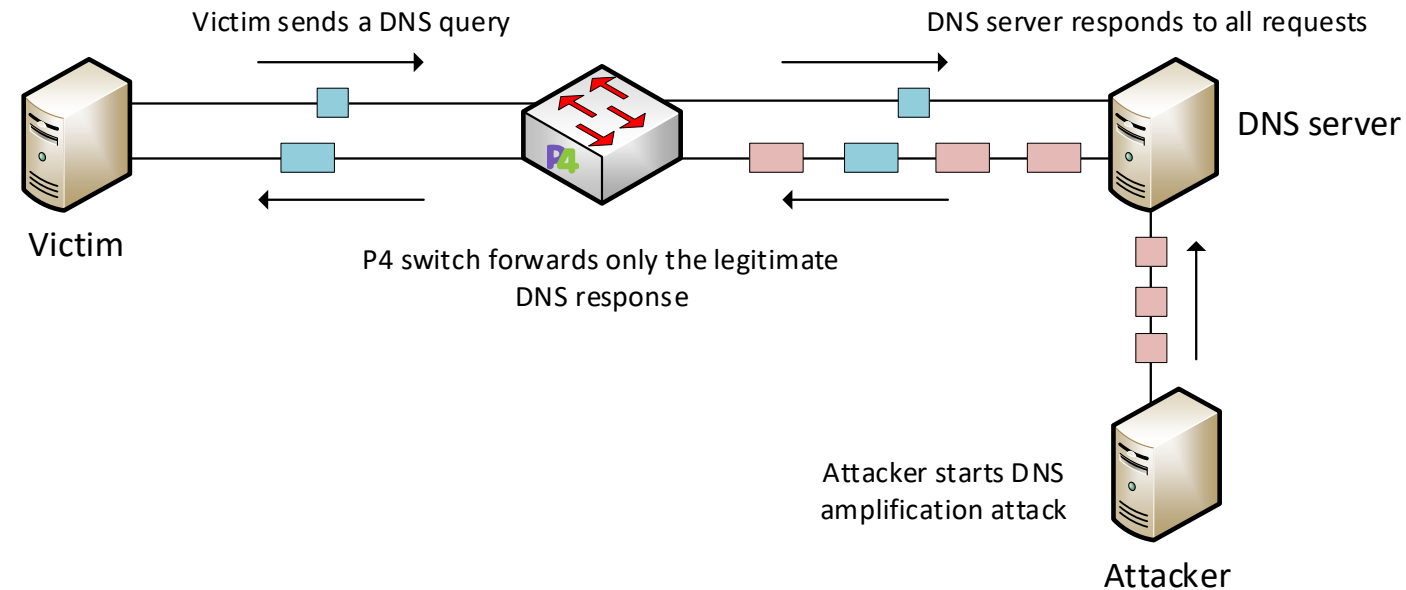
DNS Amplification

- In a DNS amplification attack, the attacker sends DNS requests with spoofed source IP address of the target victim
- The DNS server responds to all the requests and send them to the target victim
- Typically, a valid response is much larger than the request



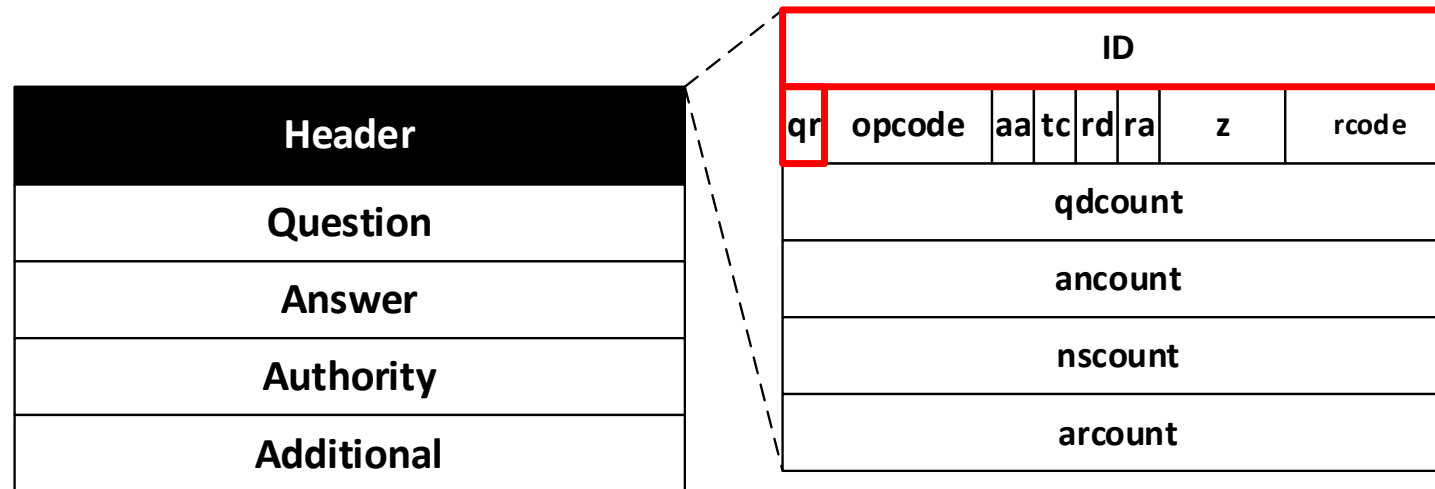
Attack Scenario

- The attacker performs a DNS amplification attack using the source IP address of the victim
- The DNS server responds to the DNS requests and sends them to the victim
- The P4 switch drops all the DNS replies that do not associate with DNS requests issued by the victim
- The P4 switch forwards legitimate DNS replies requested by the victim



DNS Header Fields

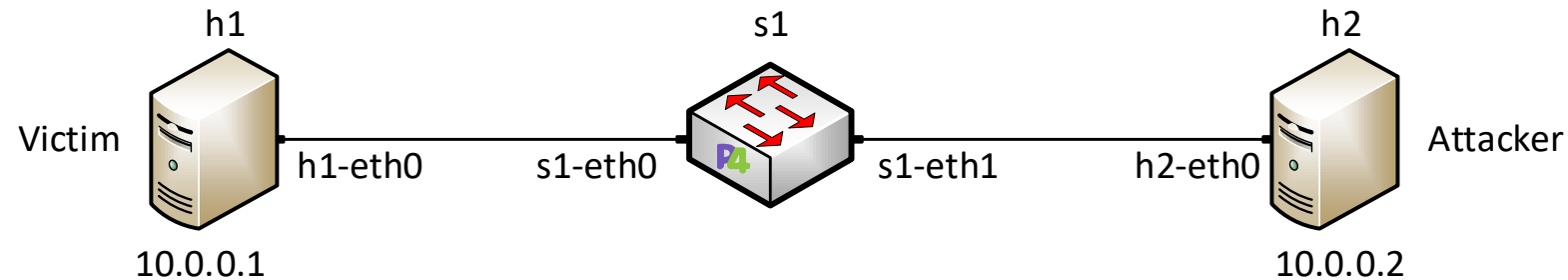
- UDP packets with source or destination port 53 are DNS packets
- The transaction ID is generated by the client upon sending a DNS request
- The DNS *qr* flag indicates that the DNS packet is request (*qr*=0) or response (*qr*=1)



Lab Topology

The topology consists of:

- Host h1 representing the victim of the DNS amplification attack
- Host h2 representing the attacker and the DNS resolver
- Programmable switch s1 that forwards traffic and protects against DNS amplification attack



DNS Amplification without Mitigation

Performing DNS amplification

```
root@lubuntu-vm: /home/admin# ./perform_dns_amplification.sh
```

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.0.0.2	10.0.0.1	DNS	135	Standard query response 0xa0ff ANY fonts.google
2	0.077750	10.0.0.2	10.0.0.1	DNS	1128	Standard query response 0xc872 ANY google.com A
3	0.116868	10.0.0.2	10.0.0.1	DNS	513	Standard query response 0xa952 ANY youtube.com A
4	0.205087	10.0.0.2	10.0.0.1	IPv4	1514	Fragmented IP protocol proto=UDP 17, off=0, ID=
5	0.282926	10.0.0.2	10.0.0.1	DNS	775	Standard query response 0x13f5 ANY instagram.com
6	0.322254	10.0.0.2	10.0.0.1	DNS	311	Standard query response 0xa4ab ANY googletagman
7	0.420935	10.0.0.2	10.0.0.1	DNS	94	Standard query response 0x7793 ANY s.w.org A 192
8	0.514951	10.0.0.2	10.0.0.1	DNS	104	Standard query response 0xc5e1 ANY linkedin.com
9	0.589614	10.0.0.2	10.0.0.1	DNS	214	Standard query response 0x0397 ANY gmpg.org SOA
10	0.633033	10.0.0.2	10.0.0.1	DNS	134	Standard query response 0x39b6 ANY ajax.googleap
11	0.721982	10.0.0.2	10.0.0.1	DNS	124	Standard query response 0x15b0 ANY fonts.gstatic
12	0.794274	10.0.0.2	10.0.0.1	DNS	277	Standard query response 0x6dd9 ANY plus.google.c
13	0.837086	10.0.0.2	10.0.0.1	DNS	294	Standard query response 0x50d2 ANY maps.google.c
14	0.926681	10.0.0.2	10.0.0.1	DNS	582	Standard query response 0x33a9 ANYyoutu.be A 11

Inspecting resource usage at the victim

```
Host: h1
Device h1-eth0 [10.0.0.1] (1/2):
=====
Incoming:
#####
#####
#####
##### Curr: 41.05 MBit/s
##### Avg: 17.76 MBit/s
##### Min: 0.00 Bit/s
##### Max: 45.44 MBit/s
##### Ttl: 337.19 MByte
#####
Outgoing:
#####
#####
#####
##### Curr: 1.15 kBit/s
##### Avg: 1.34 kBit/s
##### Min: 0.00 Bit/s
##### Max: 18.27 kBit/s
##### Ttl: 104.48 kByte
```

DNS Amplification Mitigation with P4

Defining DNS header

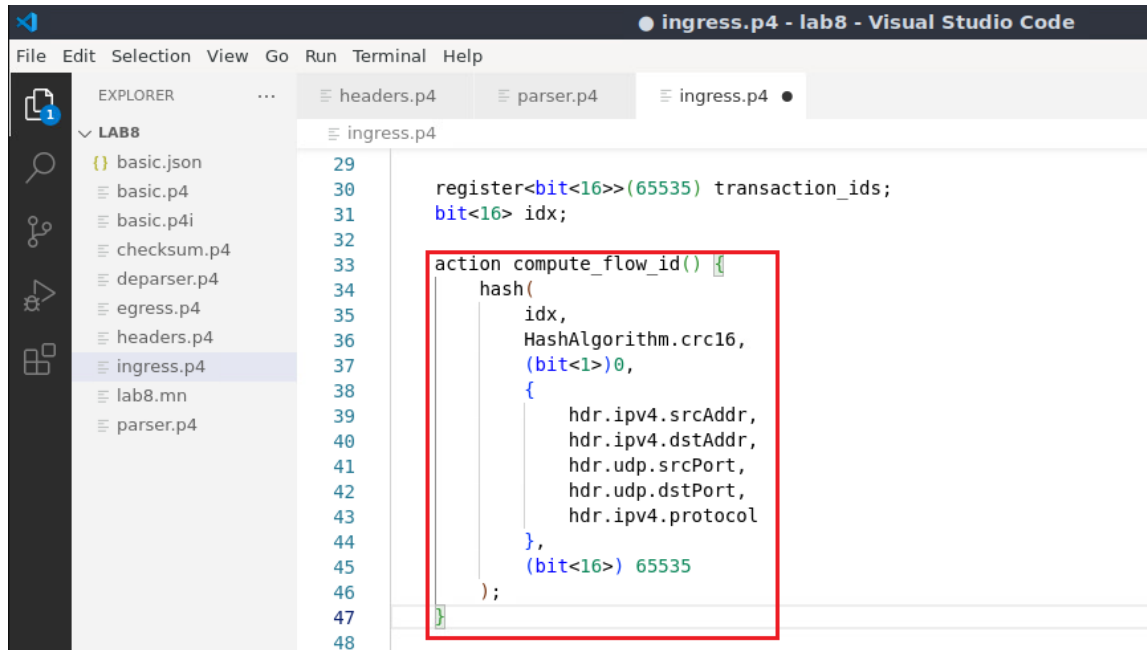
```
headers.p4 - lab8 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
LAB8
basic.json
basic.p4
basic.p4i
checksum.p4
deparser.p4
egress.p4
headers.p4
ingress.p4
lab8.mn
parser.p4
headers.p4
33
34 header udp_t {
35     bit<16> srcPort;
36     bit<16> dstPort;
37     bit<16> len;
38     bit<16> checksum;
39 }
40
41 /* Define the DNS header below*/
42 header dns_t {
43     bit<16> transaction_id;
44     bit<1> qr_flag;
45     bit<7> padding;
46 }
47
48 struct metadata {
49     /*empty*/
50 }
```

Transitioning and extracting DNS header

```
parser.p4 - lab8 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
LAB8
basic.json
basic.p4
basic.p4i
checksum.p4
deparser.p4
egress.p4
headers.p4
ingress.p4
lab8.mn
parser.p4
headers.p4
parser.p4
23 state parse_ipv4 {
24     packet.extract(hdr.ipv4);
25     transition select(hdr.ipv4.protocol) {
26         TYPE_UDP: parse_udp;
27         default: accept;
28     }
29 }
30
31 state parse_udp {
32     packet.extract(hdr.udp);
33     transition select(hdr.udp.srcPort, hdr.udp.dstPort) {
34         (TYPE_DNS,_): parse_dns;
35         (_,TYPE_DNS): parse_dns;
36         (_,_): accept;
37     }
38 }
39
40 state parse_dns {
41     packet.extract(hdr.dns);
42     transition accept;
43 }
```

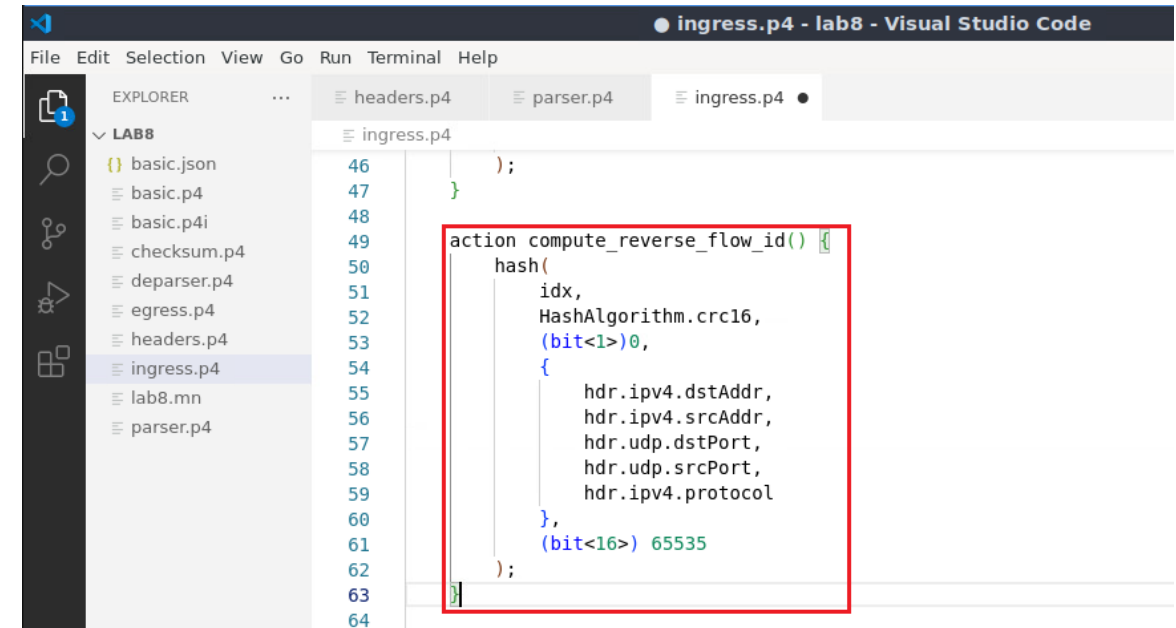
DNS Amplification Mitigation with P4

Defining a function that computes a unique flow ID based on the 5-tuple (used for DNS requests)



```
ingress.p4 - lab8 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
LAB8
basic.json
basic.p4
basic.p4i
checksum.p4
deparser.p4
egress.p4
headers.p4
ingress.p4
lab8.mn
parser.p4
headers.p4
parser.p4
ingress.p4
29
30 register<bit<16>>(65535) transaction_ids;
31 bit<16> idx;
32
33 action compute_flow_id() {
34     hash(
35         idx,
36         HashAlgorithm.crc16,
37         (bit<1>)0,
38         {
39             hdr.ipv4.srcAddr,
40             hdr.ipv4.dstAddr,
41             hdr.udp.srcPort,
42             hdr.udp.dstPort,
43             hdr.ipv4.protocol
44         },
45         (bit<16>) 65535
46     );
47
48
```

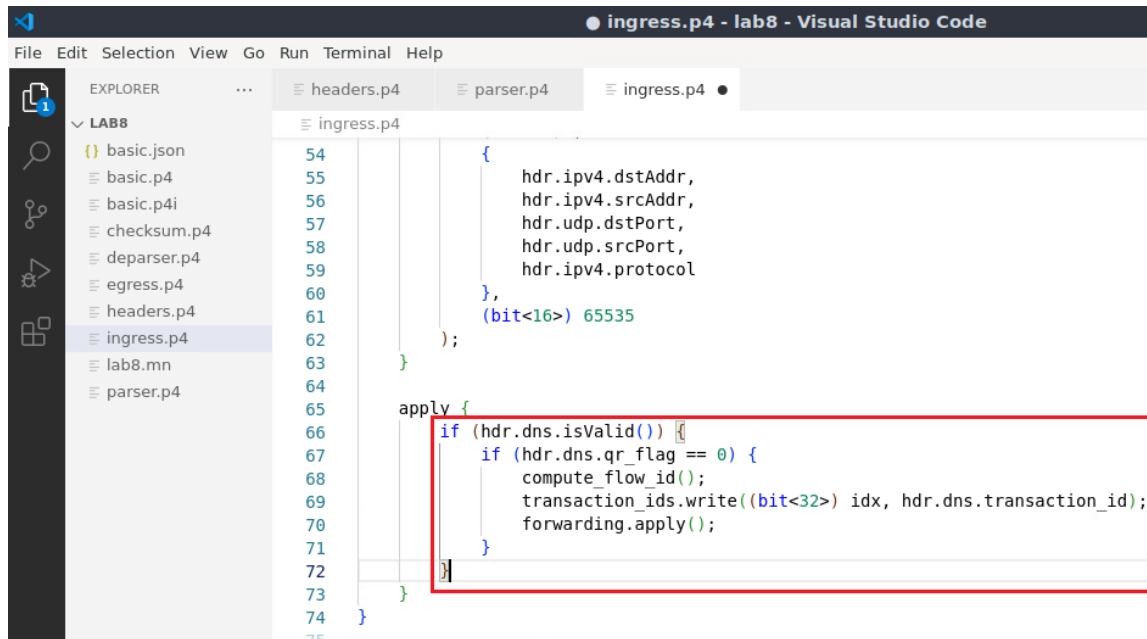
Defining a function that computes a unique flow ID based on the 5-tuple (used for DNS replies)



```
ingress.p4 - lab8 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
LAB8
basic.json
basic.p4
basic.p4i
checksum.p4
deparser.p4
egress.p4
headers.p4
ingress.p4
lab8.mn
parser.p4
headers.p4
parser.p4
ingress.p4
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
);
}
action compute_reverse_flow_id() {
    hash(
        idx,
        HashAlgorithm.crc16,
        (bit<1>)0,
        {
            hdr.ipv4.dstAddr,
            hdr.ipv4.srcAddr,
            hdr.udp.dstPort,
            hdr.udp.srcPort,
            hdr.ipv4.protocol
        },
        (bit<16>) 65535
    );
};
```

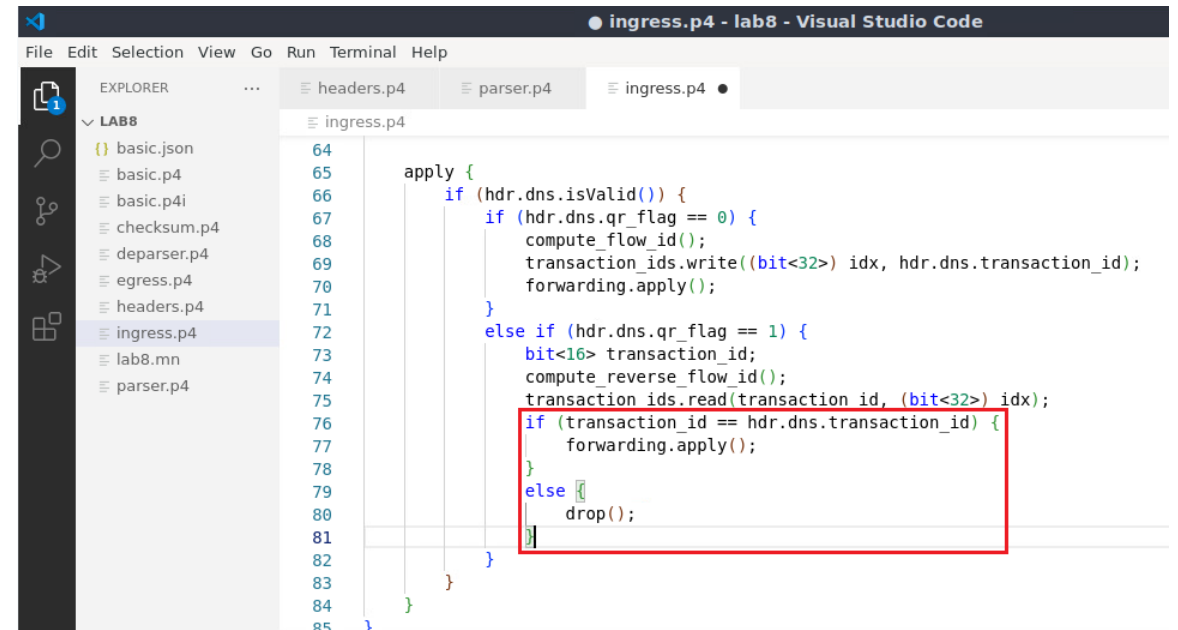
DNS Amplification Mitigation with P4

Using the flow ID to save the transaction ID of DNS request in a P4 register



```
ingress.p4 - lab8 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
LAB8
basic.json
basic.p4
basic.p4i
checksum.p4
deparser.p4
egress.p4
headers.p4
ingress.p4
lab8.mn
parser.p4
headers.p4
54 {
55     hdr.ipv4.dstAddr,
56     hdr.ipv4.srcAddr,
57     hdr.udp.dstPort,
58     hdr.udp.srcPort,
59     hdr.ipv4.protocol
60 },
61     (bit<16>) 65535
62 );
63 }
64
65 apply {
66     if (hdr.dns.isValid()) {
67         if (hdr.dns.qr_flag == 0) {
68             compute_flow_id();
69             transaction_ids.write((bit<32>) idx, hdr.dns.transaction_id);
70             forwarding.apply();
71         }
72     }
73 }
74 }
```

Using the flow ID to retrieve the transaction ID of DNS request saved in a P4 register and comparing it with the transaction ID of the received DNS reply



```
ingress.p4 - lab8 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
LAB8
basic.json
basic.p4
basic.p4i
checksum.p4
deparser.p4
egress.p4
headers.p4
ingress.p4
lab8.mn
parser.p4
ingress.p4
64
65 apply {
66     if (hdr.dns.isValid()) {
67         if (hdr.dns.qr_flag == 0) {
68             compute_flow_id();
69             transaction_ids.write((bit<32>) idx, hdr.dns.transaction_id);
70             forwarding.apply();
71         }
72     }
73     else if (hdr.dns.qr_flag == 1) {
74         bit<16> transaction_id;
75         compute_reverse_flow_id();
76         transaction_ids.read(transaction_id, (bit<32>) idx);
77         if (transaction_id == hdr.dns.transaction_id) {
78             forwarding.apply();
79         }
80         else {
81             drop();
82         }
83     }
84 }
85 }
```

DNS Amplification with Mitigation

Performing DNS amplification

```
root@lubuntu-vm:/home/admin# ./perform_dns_amplification.sh
```

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.0.0.2	10.0.0.1	DNS	135	Standard query response 0xa0ff ANY fonts.google
2	0.077750	10.0.0.2	10.0.0.1	DNS	1128	Standard query response 0xc872 ANY google.com A
3	0.116868	10.0.0.2	10.0.0.1	DNS	513	Standard query response 0xa952 ANY youtube.com A
4	0.205087	10.0.0.2	10.0.0.1	IPv4	1514	Fragmented IP protocol proto=UDP 17, off=0, ID=
5	0.282926	10.0.0.2	10.0.0.1	DNS	775	Standard query response 0x13f5 ANY instagram.com
6	0.322254	10.0.0.2	10.0.0.1	DNS	311	Standard query response 0xa4ab ANY googletagman
7	0.420935	10.0.0.2	10.0.0.1	DNS	94	Standard query response 0x7793 ANY s.w.org A 192
8	0.514951	10.0.0.2	10.0.0.1	DNS	104	Standard query response 0xc5e1 ANY linkedin.com
9	0.589614	10.0.0.2	10.0.0.1	DNS	214	Standard query response 0x0397 ANY gmpg.org SOA
10	0.633033	10.0.0.2	10.0.0.1	DNS	134	Standard query response 0x39b6 ANY ajax.googleap
11	0.721982	10.0.0.2	10.0.0.1	DNS	124	Standard query response 0x15b0 ANY fonts.gstatic
12	0.794274	10.0.0.2	10.0.0.1	DNS	277	Standard query response 0x6dd9 ANY plus.google.c
13	0.837086	10.0.0.2	10.0.0.1	DNS	294	Standard query response 0x50d2 ANY maps.google.c
14	0.926681	10.0.0.2	10.0.0.1	DNS	582	Standard query response 0x33a9 ANY youtu.be A 11

Inspecting resource usage at the victim

```
"Host: h1"
Device h1-eth0 [10.0.0.1] (1/2):
=====
Incoming:
Curr: 0.00 Bit/s
Avg: 8.00 Bit/s
Min: 0.00 Bit/s
Max: 53.73 MBit/s
Ttl: 472.57 MByte

Outgoing:
Curr: 0.00 Bit/s
Avg: 0.00 Bit/s
Min: 0.00 Bit/s
Max: 27.54 kBit/s
Ttl: 42.41 kByte
```