Faculty of Engineering and Technology

Computer Science and Cyber Security Department

Intrusion Detection and Prevention System

IDS/IPS Practical Projects.

Team 8 – DNS Security & Tunneling Detection.

Prepared by:-

Afnan Manasrah – 1221150.

Aseel Rafea – 1222375.

Instructor:-

Rasheed Yousef.

Section Number: 1.

BIRZEIT

Abstract:-

This report aims to enhance the security of the Domain Name System (DNS) protocol by detecting malicious activities related to DNS tunneling operation used by attackers to establish backdoors or route data off the network. Normal DNS traffic was analyzed using Zeek and Suricata logs to differentiate between legitimate and anomalous behavior. Rules were developed using Suricata and Zeek to detect suspicious patterns, such as highly random queries, long queries (especially TXT), and abnormal spikes in NXDOMAIN responses, as well as pattern signatures of tunneling tools like iodine and dns2tcp. The report also presents a practical demonstration of this project, outlining the attack steps, detection mechanisms, and how to prevent attacks using IPS and firewalls.

Contents

Figures

Introduction:-

DNS is a fundamental system on the internet responsible for translating simple website names into computer-understood IP addresses. This system is essential for ensuring users can access websites and services quickly and easily. Despite its seemingly simple function, it is vulnerable to attacks, whether through DNS Tunneling which enables attackers to steal data or create a hidden channel that bypasses standard security systems, or any another ways.

To detect these malicious activities, the project relies on two analysis and detection mechanisms, starting with **Zeek – Network Security Monitoring**. Zeek is a powerful network monitoring and traffic analysis tool that converts data into detailed logs, allowing for an understanding of DNS traffic behavior within the network. This means it relies on behavioral analysis, which helps detect activities that lack permanent signatures. In this project, Zeek was used to collect and analyze DNS logs, build a baseline for normal DNS traffic, detect anomalous behavior, and provide data to support the development of Zeek scripts and Suricata rules.

Secondly, **Suricata – IDS/IPS Engine**. Suricata was used as a tool to detect and prevent malicious activities. It works by creating rules dedicated to detecting suspicious patterns in DNS traffic, especially those related to tunneling. Suricata was used in the project, where rules were written based on the patterns detected, detecting tunneling activity, issuing alerts, and activating IPS mode to perform prevention actions. Suricata represents the detection and protection layer that works directly on network traffic.

This means that the security design in this project relies on combining behavioral analysis (Zeek) with signature-based detection and blocking (Suricata). Zeek identifies anomalous behavior, while Suricata performs real-time detection through rules and takes appropriate blocking action.

As for the attack, we will use the iodine tool, which is a DNS tunneling tool that hides data within DNS querisies so that it is not detected and is usually used by attackers for the purpose of smuggling data or obtaining sensetive data.

Architecture Design:-

The architecture design for this experiment will be as follows: the victim’s device will be located on its own network, which will also have an IDS between the firewall and the DNS server to verify all requests coming out of the network or even the responses coming in to it. The victim will generate innocent requests which the attacker, whether on the same network or outside it, will exploit to smuggle data.

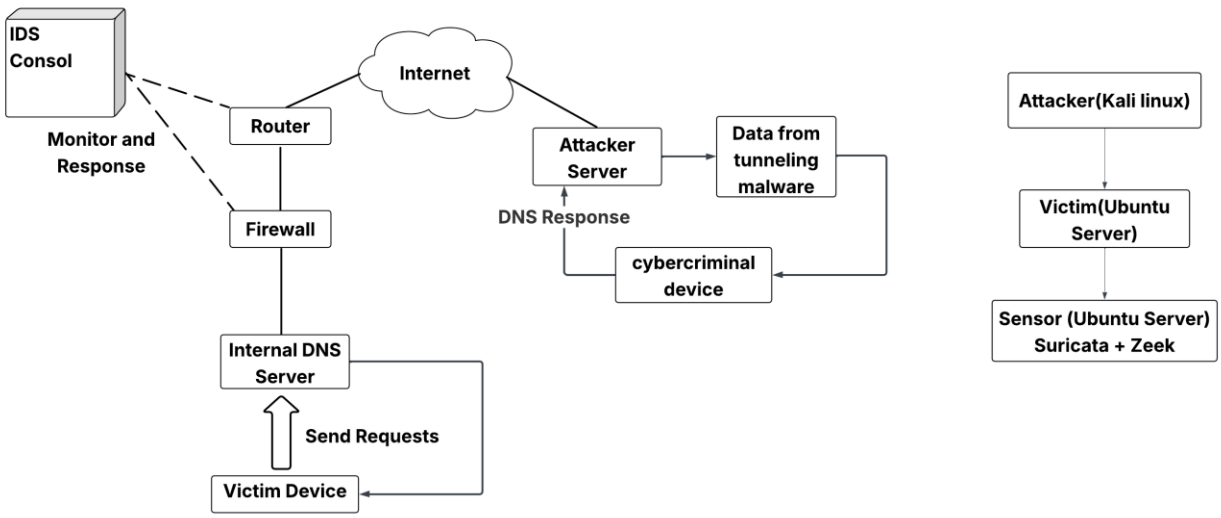

Figure 1: architecture design 1

Implementation:-

1. **General Setup:-**

   To build the experiment environment, we will make three virtual machines: two Ubuntu servers, one for the victim and one to act as a sensor, and the third machine will be kali to act as attacker. Each machine will have two adapters: one NAT and the other is Host-only. The NAT for internet connection, and Host-only to create an isolated environment to implement the attack without affecting the network.

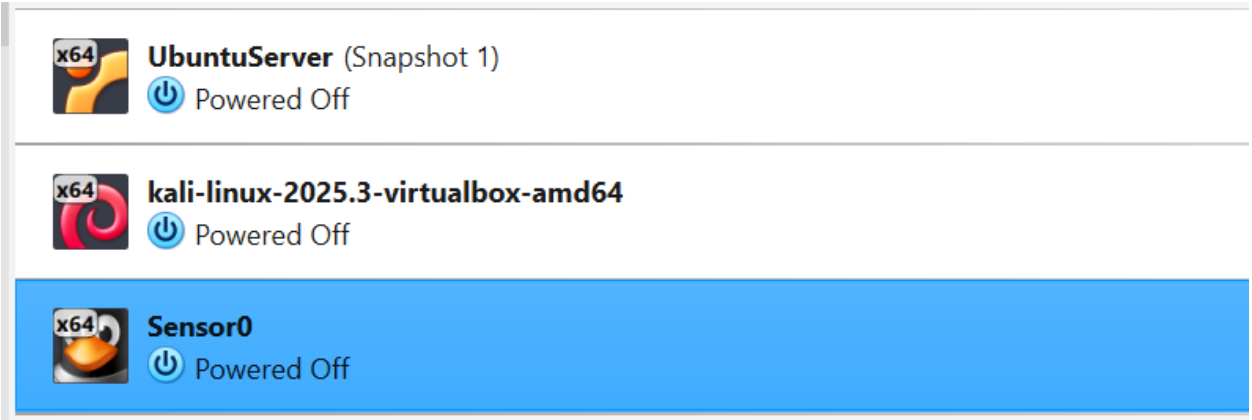

Figure 2: Machines

2. **Internal Setup:-**

   a) Sensor machine:-

   In the sensor machine the first thing we do is network sittings and we make two adapter and make the sitting as follow:

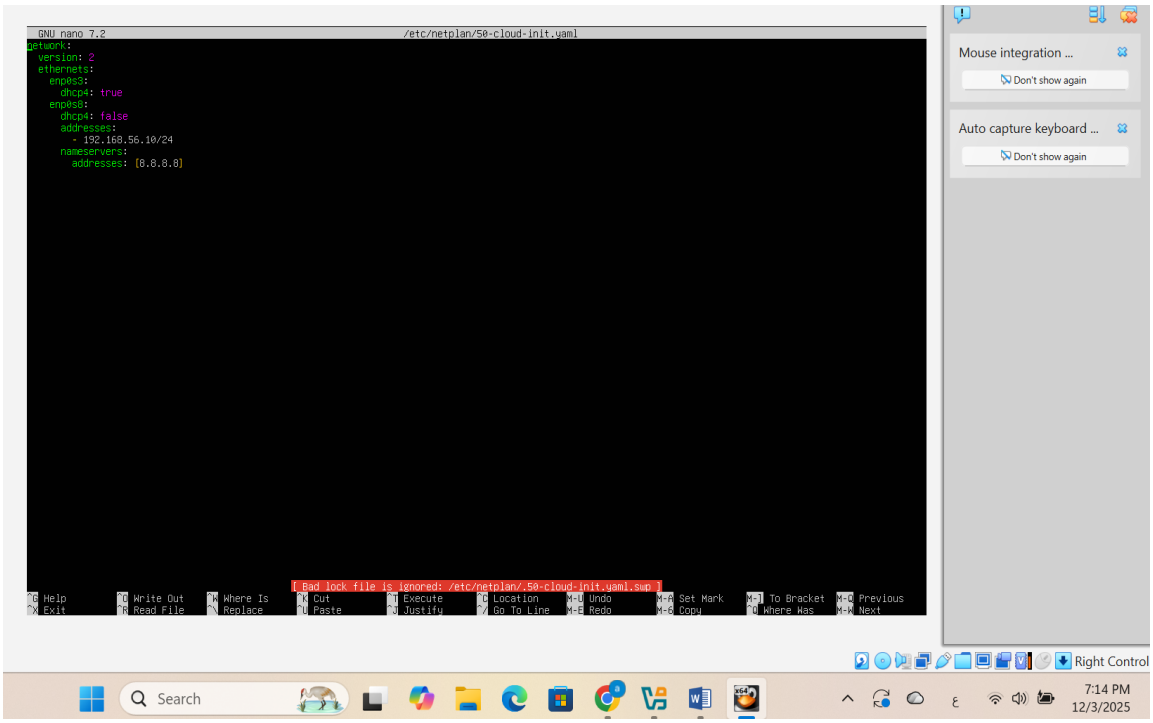

Figure 3: sensor network sittings

We did it this way because the detection it will be perform on the enp0s8 interface and it's the same interface the victim will use to make DNS requests.

After that we install Suricata to perform detection, we install it as follows:

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
[Mon Nov 24 21:16:01] afnansensor@sensor:~$ sudo wget https://raw.githubusercontent.cpm/OISF/suricata/master/suricata.yaml.in -O /etc/suricata/suricata.yaml
--2025-11-24 21:17:31--  https://raw.githubusercontent.cpm/OISF/suricata/master/suricata.yaml.in
Resolving raw.githubusercontent.cpm (raw.githubusercontent.cpm)... failed: Name or service not known.
wget: unable to resolve host address 'raw.githubusercontent.cpm'
[Mon Nov 24 21:17:31] afnansensor@sensor:~$ sudo wget https://raw.githubusercontent.com/OISF/suricata/master/suricata.yaml.in -O /etc/suricata/suricata.yaml
--2025-11-24 21:17:45--  https://raw.githubusercontent.com/OISF/suricata/master/suricata.yaml.in
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 93287 (91K) [text/plain]
Saving to: '/etc/suricata/suricata.yaml'

/etc/suricata/suricata.yaml           100%[=============================================================================>]  91.10K   476KB/s    in 0.2s

2025-11-24 21:17:46 (476 KB/s) - '/etc/suricata/suricata.yaml' saved [93287/93287]
```

Commands:

- sudo apt install suricata –y : to install Suricata tool.
- sudo wget https://raw.githubusercontent.com/OISF/suricata/master/suricata.yaml.in -O /etc/suricata/suricata.yaml : to install the configuration file that will work with it. And we make some changes in it, like the interface that will work in it and make it enp0s8, and change some file paths to save the results of detection and all related work in it.

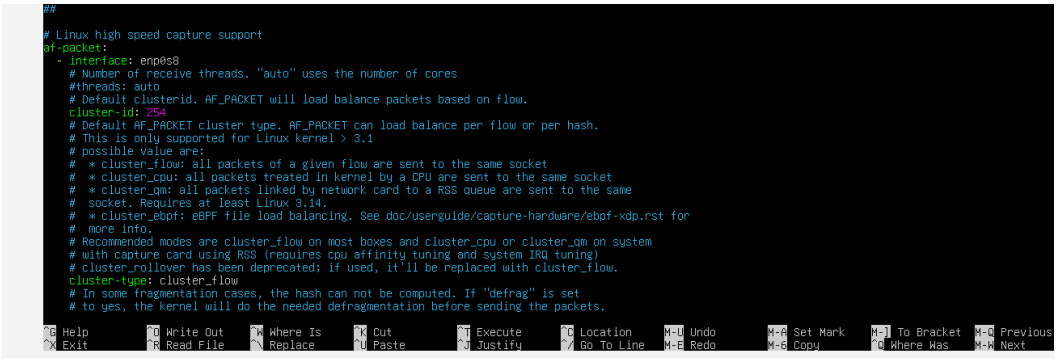

Figure 4: change Suricata interface

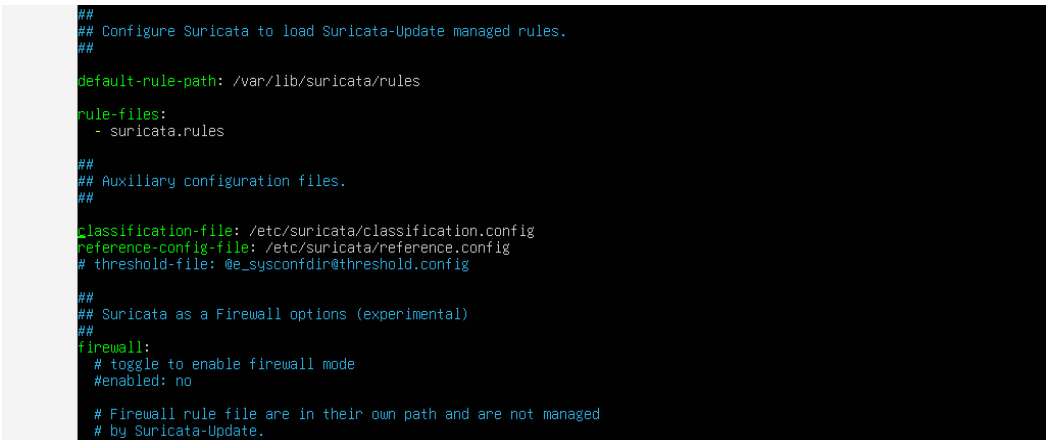

Figure 5: change paths for some related files

And then we make the directories we will need to store operating files, log files, rules, and cache, and then change ownership of directories to ensure that Suricata have a permission on it.

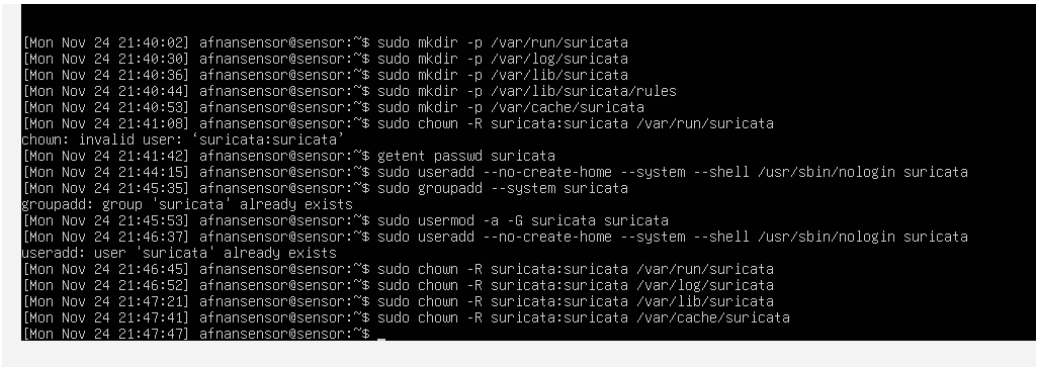

Figure 6: create the directories and change the ownership

Then, for Suricata to do its job and detect malicious DNS requests, we wrote the following rules:

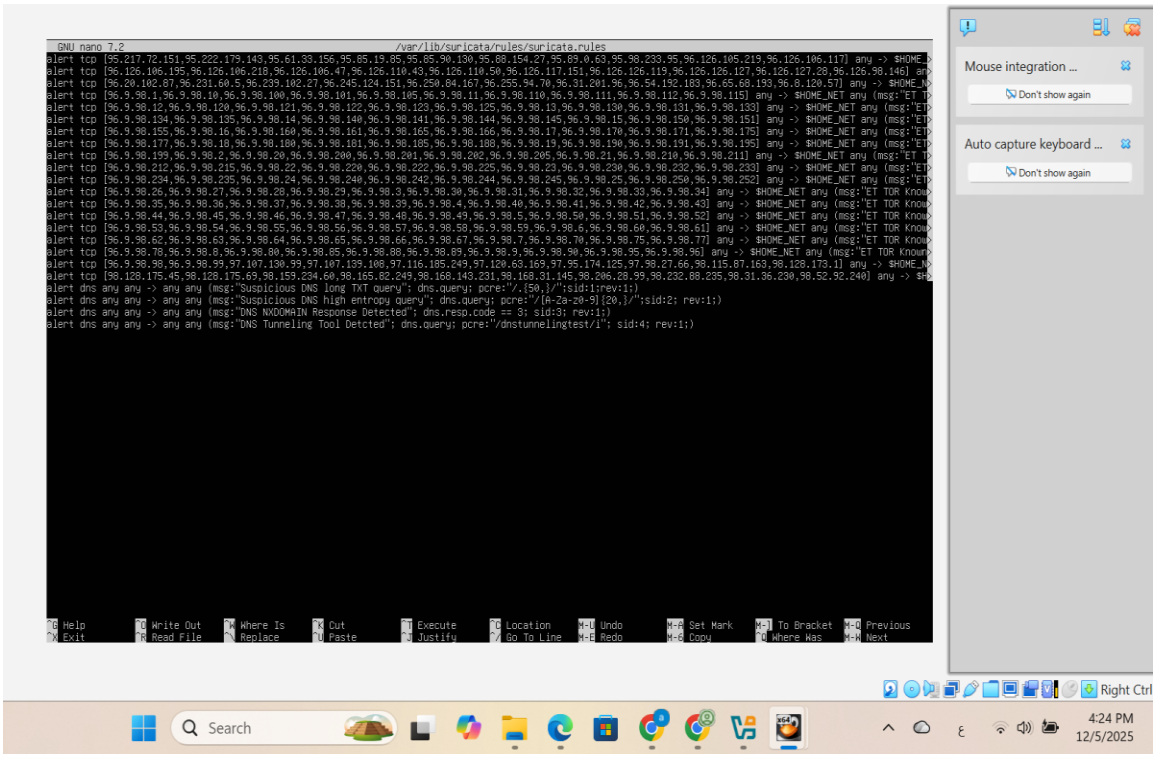

Figure 7: Suricata rules

- Alert dns any any -> any any (msg:”Suspicious DNS Long TXT Query”; dns.query; pcre:”/.{50,}/”; sid:1; rev:1;
  This query to detect and make alert to any DNS request from any IP and any port that go to any IP and any port and detect any long domains or subdomains with repeated characters exceeding 50 and when give alert the rule will generate the message “Suspicious DNS Long TXT Query”, sid: 1 is the number of rule and rev:1 is the number of version.
- Alert dns any any -> any any (msg:”Suspicious DNS high entropy Query”; dns.query; pcre:”/[A-Za-z0-9].{20,}/”; sid:2; rev:1;
  This query to detect and make alert to any DNS request from any IP and any port that go to any IP and any port and detect any Base64 or Base32 domains or subdomains and when give alert the rule will generate the message “Suspicious DNS high entropy Query”, sid:2 is the number of rule and rev:1 is the number of version.
- Alert dns any any -> any any (msg:”DNS NXDOMAIN Response Detected”; dns.resp.code==3; sid:3; rev:1;
  This query to detect and make alert to any DNS request from any IP and any port that go to any IP and any port and detect any response containing the phrase NXDOMAIN and the response code is 3 and its known to NXDOMAIN responses, and when give alert the rule will generate the message “DNS NXDOMAIN Response Detected”, sid:3 is the number of rule and rev:1 is the number of version.

- Alert dns any any -> any any (msg:”DNS Tunneling Tool Detected”;dns.query; content:”dnstunnelingtest”;sid:4; rev:1;
  This query to detect and make alert to any DNS request from any IP and any port that go to any IP and any port and detect any request containing the phrase dnstunnelingtest, and when give alert the rule will generate the message “DNS Tunneling Tool Detected”, sid:4 is the number of rule and rev:1 is the number of version.

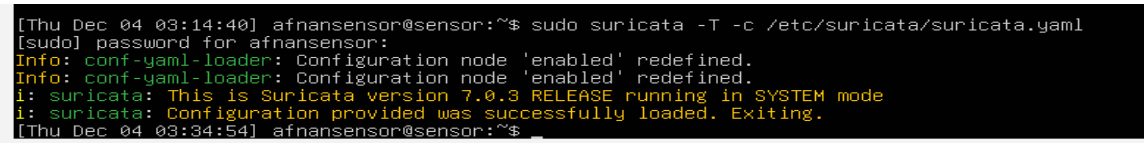

Figure 8: Suricata is running

And then install Zeek to perform detection, we install it as follows:

Figure 9: Zeek install

Figure 10: decompress the Zeek file

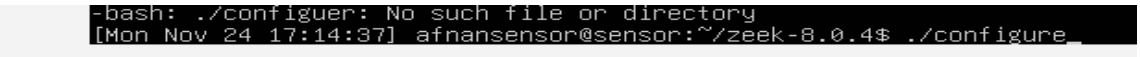

Figure 11:configuration build

Commands:

- wget https://download.zeek.org/zeek-8.0.4.tar.gs: install zeek directory from the organization site.
- tar –xvzf zeek-8.0.4.tar.gz : decompress the file.
- cd zeek-8.0.4 && ./configure: To inspect the device and prepare the construction file.
- make && sudo make install: to build and compile the program and install the resulting files within the system.

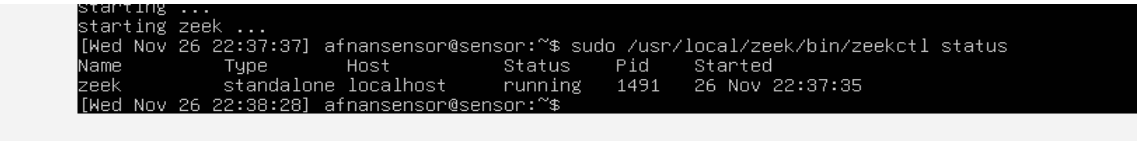

Figure 12: Zeek is running

After that we change the interface that will work in it and make it enp0s8.

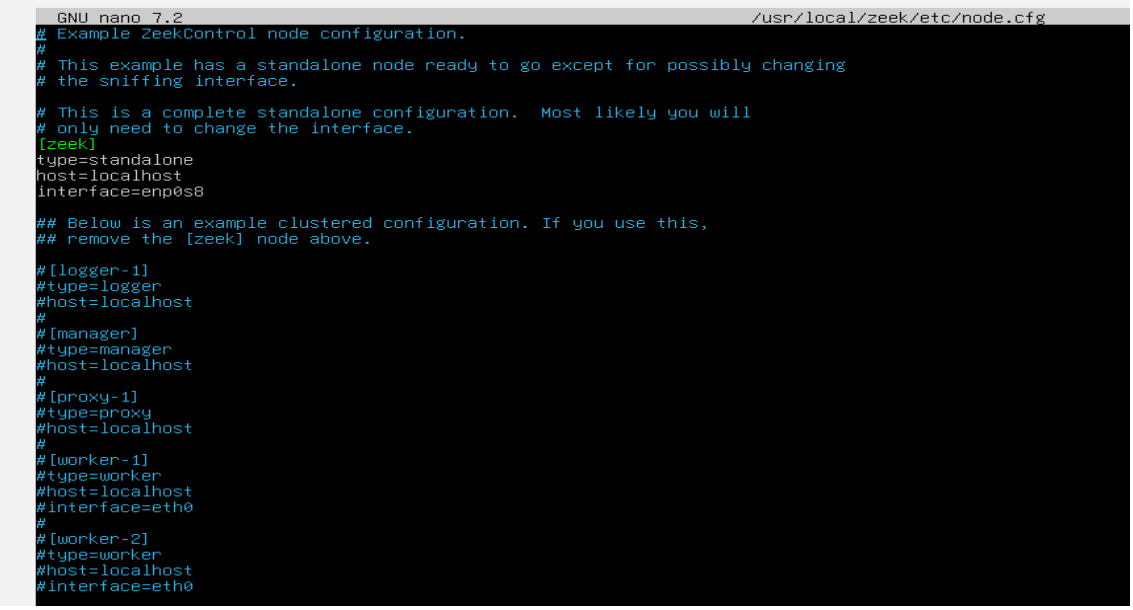

Figure 13: change the interface

Then we convert the Suricata rules to Zeek scripts (they work in the same way and for the same purpose, but the writing method differs due to the different tools).

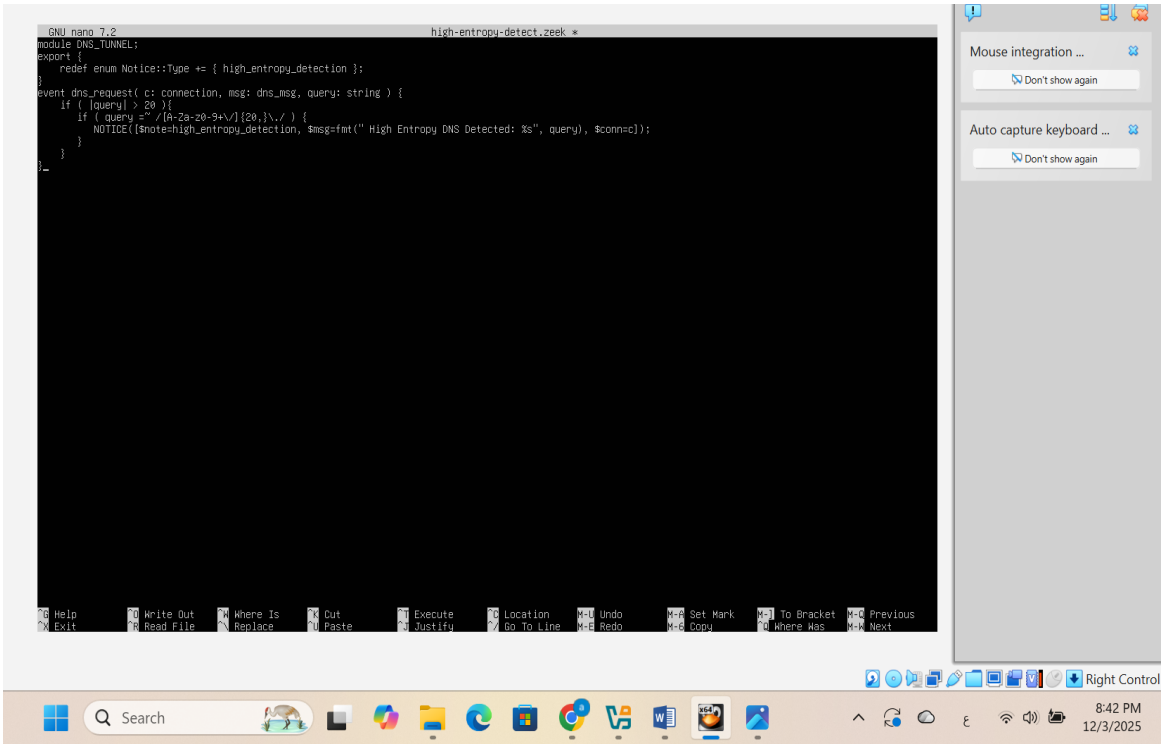

Figure 14:Zeek script – high entropy

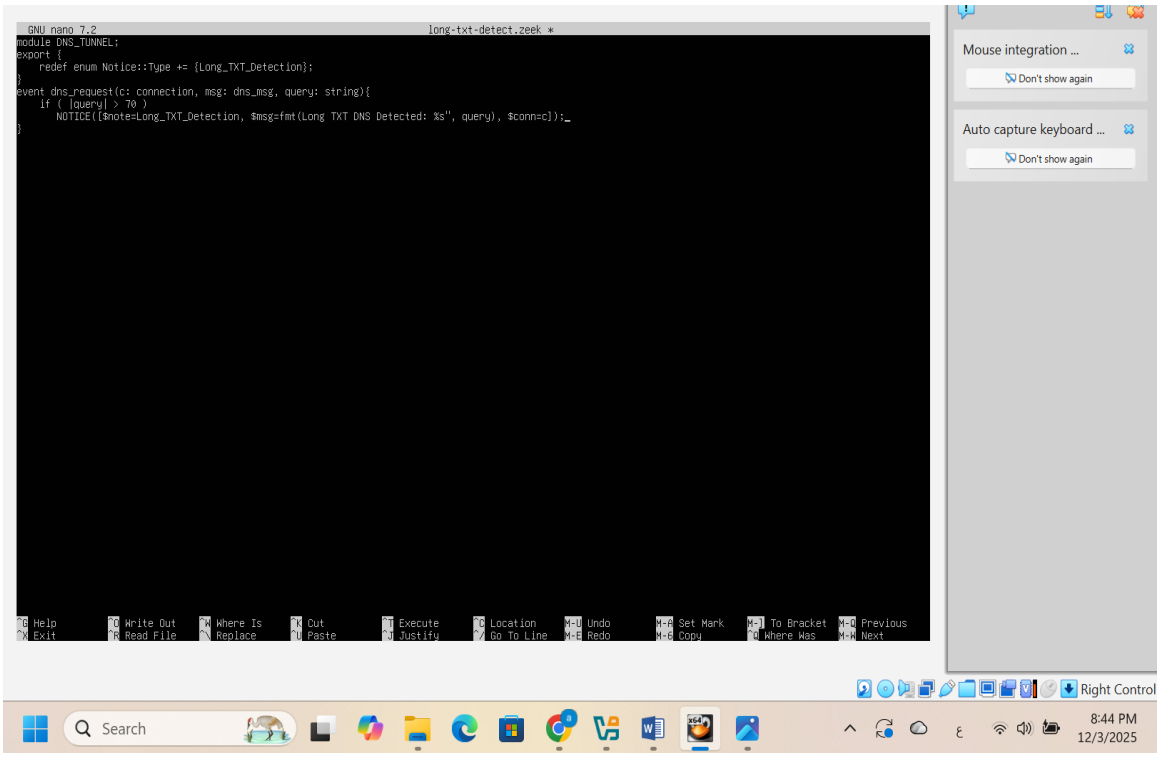

Figure 15: Zeek script – long TXT

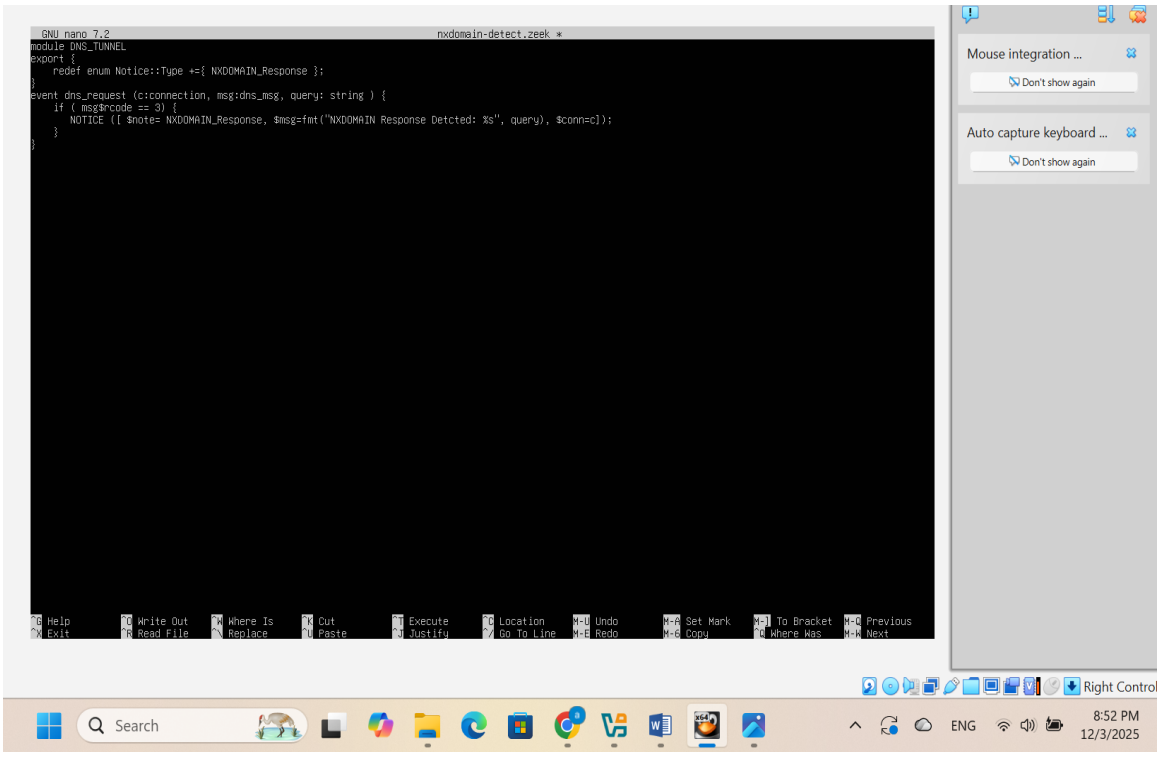

Figure 16:Zeek script - NXDOMAIN

And then we put it in local.zeek to load it and use it to detect after we changed the permissions to be enforceable.

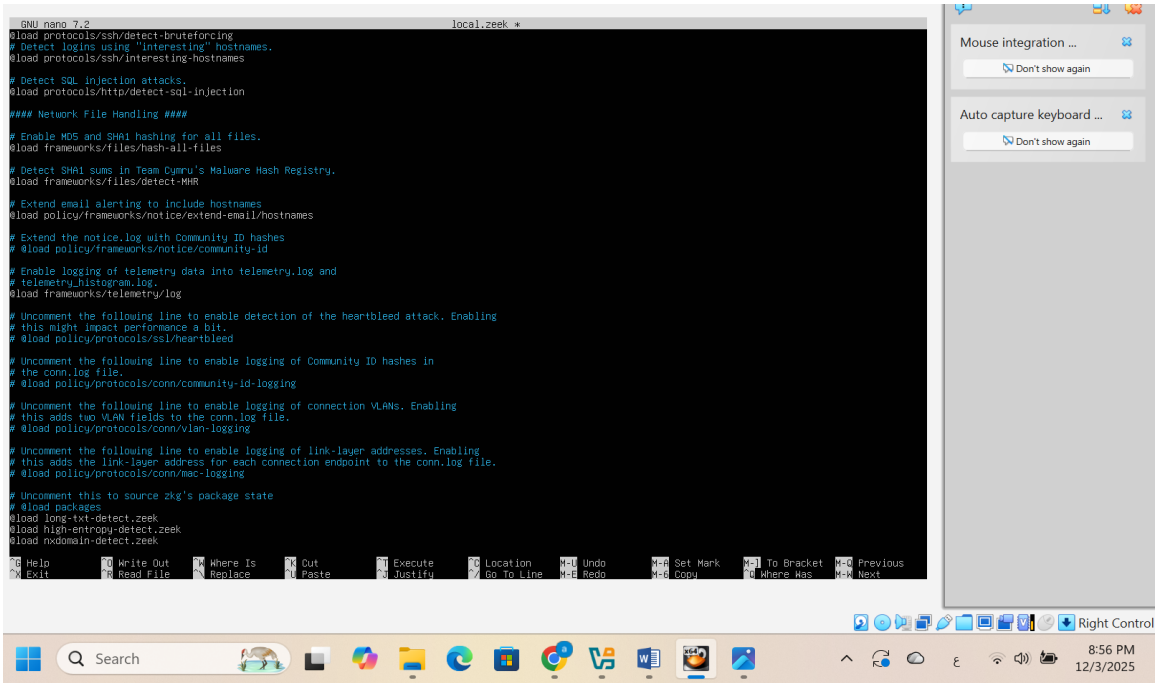

Figure 17: put it in local.zeek

b) Victim (UbuntuServer) machine:-
In the victim machine the first thing we do is network sittings and we make two adapter and make the sitting as follow:

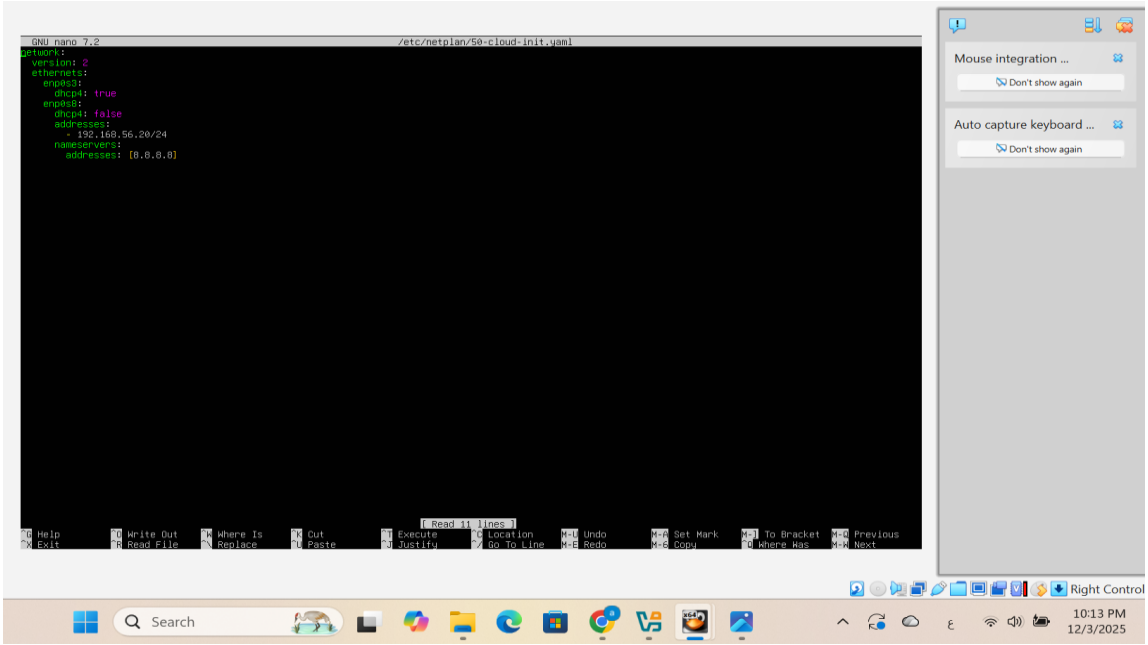

Figure 18: victim network sittings

Then we changed the IP of the DNS server to be the same as the victim's IP to represented the trusted DNS system, because the IDS is located between it and the firewall and therefore to hold all the traffic of the DNS, whether innocent or malicious.

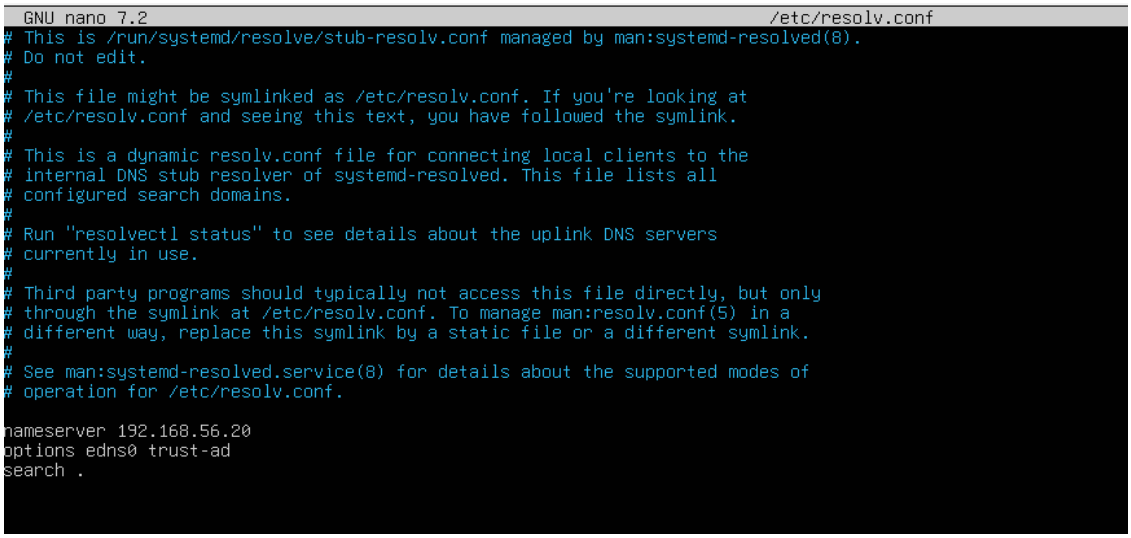

Figure 19: change the nameserver

c) Attacker (Kali) machine:-
In this machine, we will simulate the DNS tunneling using two tools, dns2tcp & iodine, which we will install as follows:

Commands:

- Sudo apt install iodine: to install iodine tool.

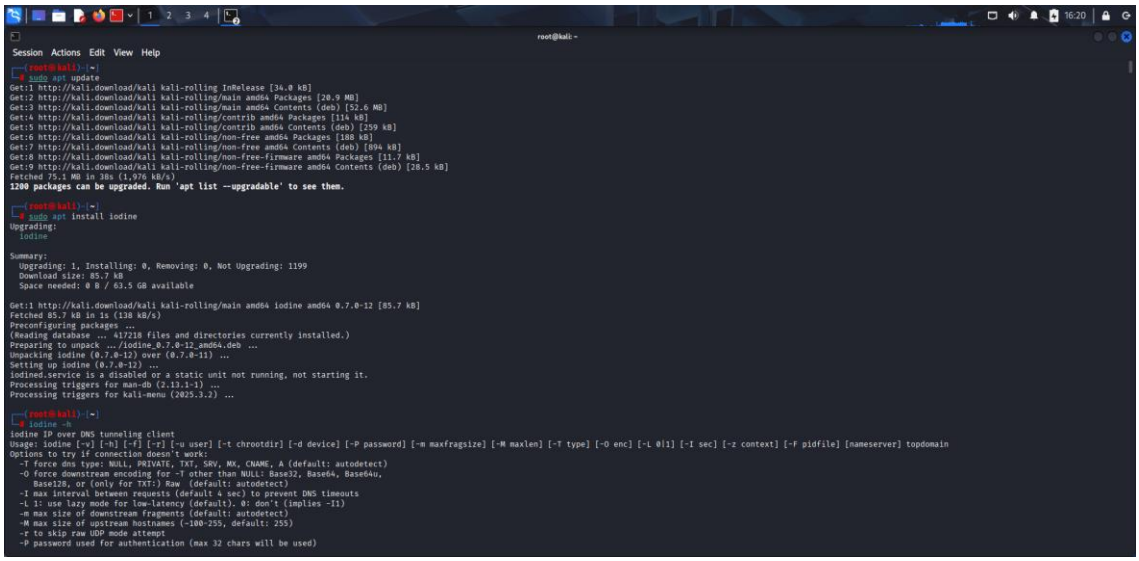

Figure 20: install iodine tool

3. **Monitoring basic, normal DNS traffic:**
   To monitor typical DNS traffic, we will generate requests from the victim (we tested it over several days) using dig and nslookup commands to see if they will be seen by the tools or not.

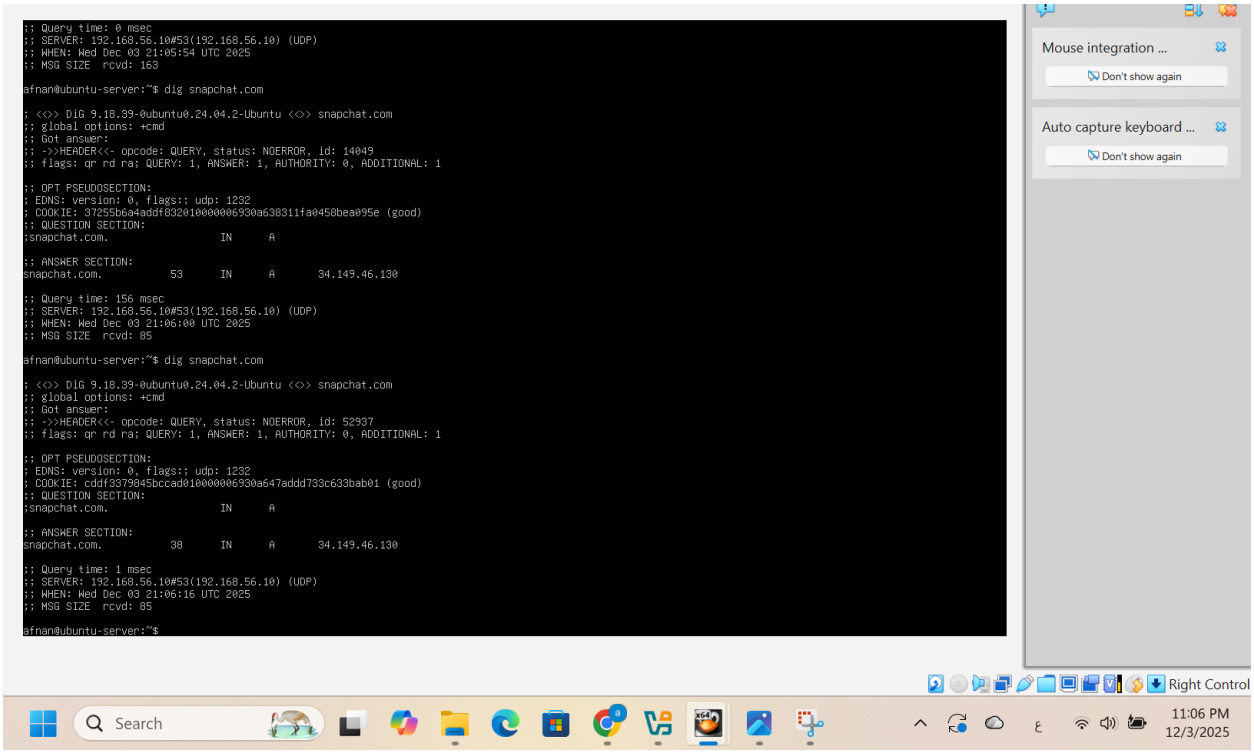

Figure 21: normal DNS requests 1

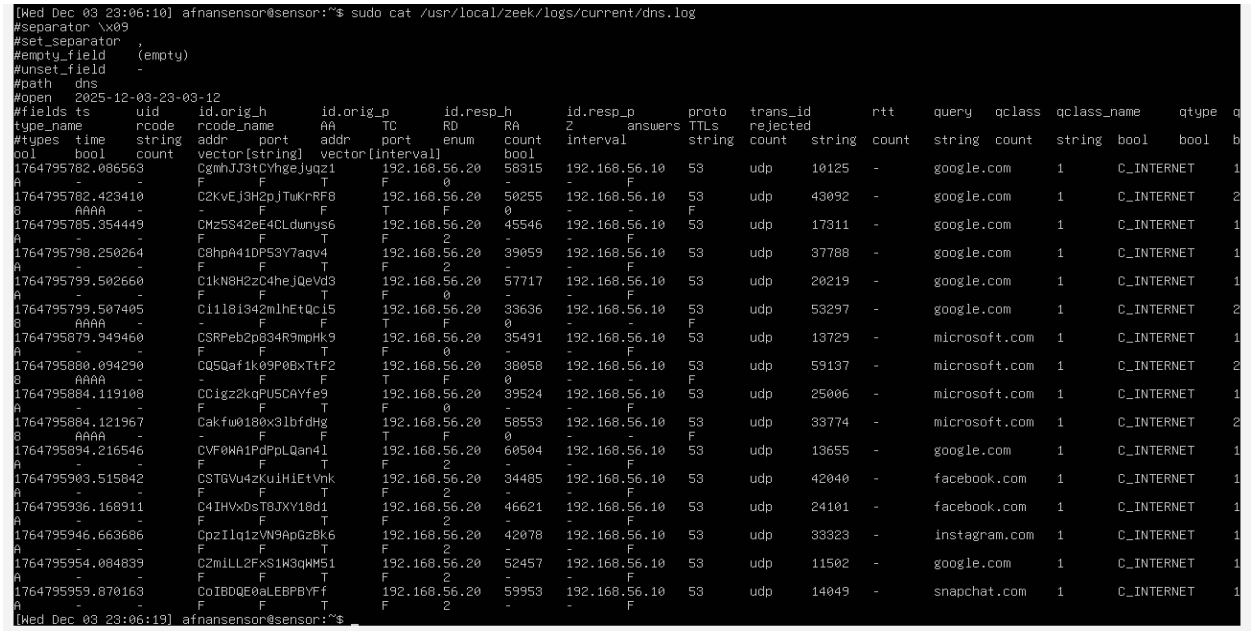

Figure 22: We can see it from Zeek

```
Nov 27 02:01:11 sensor suricata[3524]: 1: suricata: This is Suricata version 7.0.3 RELEASE running in SYSTEM mode
Nov 27 02:01:11 sensor systemd[1]: Started suricata.service - Suricata IDS/IDP daemon.
[Thu Nov 27 02:01:20] afnansensor@sensor:~$ cat /usr/local/zeek/logs/current/dns.log
#separator \x09
#set_separator  ,
#empty_field    (empty)
#unset_field    -
#path   dns
#open   2025-11-27-02-00-27
#fields ts      uid     id.orig_h       id.orig_p       id.resp_h       id.resp_p       proto   trans_id        rtt     query   qclass  qclass_name     qtype   q
type_name       rcode   rcode_name      AA      TC      RD      RA      Z       answers TTLs    rejected
#types  time    string  addr    port    addr    port    enum    count   interval        string  count   string  count   string  count   string  bool    bool    b
ool     bool    count   vector[string]  vector[interval]        bool
1764201617.328961       C9y2ow1RcqADXzTHhb      192.168.56.20   56443   192.168.56.10   53      udp     44934   -       facebook.com    1       C_INTERNET      1
A       -       -       F       F       T       F       0       -       -       F
1764201617.398984       CRZEuiWt15cEU5wwc       192.168.56.20   57367   192.168.56.10   53      udp     19296   -       facebook.com    1       C_INTERNET      2
8       AAAA    -       -       F       F       T       F       0       -       -       F
1764201619.886286       CGo6KT1KrcDMQXx2Qd      192.168.56.20   38132   192.168.56.10   53      udp     21298   -       google.com      1       C_INTERNET      1
A       -       -       F       F       T       F       0       -       -       F
1764201620.010661       C1lA5I2k1r54CZZRq3      192.168.56.20   42194   192.168.56.10   53      udp     996     -       google.com      1       C_INTERNET      2
8       AAAA    -       -       F       F       T       F       0       -       -       F
1764201623.004636       CaG57A22zf5A69wJ2aj      192.168.56.20   54703   192.168.56.10   53      udp     61272   -       instagram.com   1       C_INTERNET      1
A       -       -       F       F       T       F       0       -       -       F
1764201623.184191       CaxB1k3CooNXEMU2ok      192.168.56.20   45784   192.168.56.10   53      udp     63715   -       instagram.com   1       C_INTERNET      2
8       AAAA    -       -       F       F       T       F       0       -       -       F
[Thu Nov 27 02:01:26] afnansensor@sensor:~$
```

Figure 23: normal DNS requests 2


```
{"timestamp":"2025-11-27T02:34:27.897851+0200","event_type":"stats","stats":{"uptime":424,"capture":{"kernel_packets":48,"kernel_drops":0,"errors":0,"afpacket":
{"busy_loop_avg":0,"polls":11057,"poll_signal":0,"poll_timeout":11010,"poll_data":47,"poll_errors":0,"send_errors":0}},"decoder":{"pkts":48,"bytes":10392,"inval
id":0,"ipv4":24,"ipv6":24,"ethernet":48,"arp":0,"unknown_ethertype":0,"chdlc":0,"raw":0,"null":0,"sll":0,"tcp":0,"udp":48,"sctp":0,"esp":0,"icmpv4":0,"icmpv6":0
,"ppp":0,"pppoe":0,"geneve":0,"gre":0,"vlan":0,"vlan_qinq":0,"vlan_qinqinq":0,"vxlan":0,"vntag":0,"ieee8021ah":0,"teredo":0,"ipv4_in_ipv6":0,"ipv6_in_ipv6":0,"m
pls":0,"avg_pkt_size":216,"max_pkt_size":383,"max_mac_addrs_src":0,"max_mac_addrs_dst":0,"erspan":0,"nsh":0,"event":{"ipv4":{"pkt_too_small":0,"hlen_too_small":
0,"iplen_smaller_than_hlen":0,"trunc_pkt":0,"opt_invalid":0,"opt_invalid_len":0,"opt_malformed":0,"opt_pad_required":0,"opt_eol_required":0,"opt_duplicate":0,"o
pt_unknown":0,"wrong_ip_version":0,"icmpv6":0,"frag_pkt_too_large":0,"frag_overlap":0,"frag_ignored":0},"icmpv4":{"pkt_too_small":0,"unknown_type":0,"unknown_co
de":0,"ipv4_trunc_pkt":0,"ipv4_unknown_ver":0},"icmpv6":{"unknown_type":0,"unknown_code":0,"pkt_too_small":0,"ipv6_unknown_version":0,"ipv6_trunc_pkt":0,"mld_me
ssage_with_invalid_hl":0,"unassigned_type":0,"experimentation_type":0},"ipv6":{"pkt_too_small":0,"trunc_pkt":0,"trunc_exthdr":0,"exthdr_dupl_fh":0,"exthdr_usele
ss_fh":0,"exthdr_dupl_rh":0,"exthdr_dupl_hh":0,"exthdr_dupl_dh":0,"exthdr_dupl_ah":0,"exthdr_dupl_eh":0,"exthdr_invalid_optlen":0,"wrong_ip_version":0,"exthdr_a
h_res_not_null":0,"hopopts_unknown_opt":0,"hopopts_only_padding":0,"dstopts_unknown_opt":0,"dstopts_only_padding":0,"rh_type_0":0,"zero_len_padn":0,"fh_non_zero
_reserved_field":0,"data_after_none_header":0,"unknown_next_header":0,"icmpv4":0,"frag_pkt_too_large":0,"frag_overlap":0,"frag_invalid_length":0,"frag_ignored":
0,"ipv4_in_ipv6_too_small":0,"ipv4_in_ipv6_wrong_version":0,"ipv6_in_ipv6_too_small":0,"ipv6_in_ipv6_wrong_version":0},"tcp":{"pkt_too_small":0,"hlen_too_small"
:0,"invalid_optlen":0,"opt_invalid_len":0,"opt_duplicate":0},"udp":{"pkt_too_small":0,"hlen_too_small":0,"hlen_invalid":0,"len_invalid":0},"sll":{"pkt_too_small
":0},"ethernet":{"pkt_too_small":0},"ppp":{"pkt_too_small":0,"vju_pkt_too_small":0,"ip4_pkt_too_small":0,"ip6_pkt_too_small":0,"wrong_type":0,"unsup_proto":0},"
pppoe":{"pkt_too_small":0,"wrong_code":0,"malformed_tags":0},"gre":{"pkt_too_small":0,"wrong_version":0,"version0_recur":0,"version0_flags":0,"version0_hdr_too_
big":0,"version0_malformed_sre_hdr":0,"version1_chksum":0,"version1_route":0,"version1_ssr":0,"version1_recur":0,"version1_flags":0,"version1_no_key":0,"version
1_wrong_protocol":0,"version1_malformed_sre_hdr":0,"version1_hdr_too_big":0},"vlan":{"header_too_small":0,"unknown_type":0,"too_many_layers":0},"ieee8021ah":{"h
eader_too_small":0},"vntag":{"header_too_small":0,"unknown_type":0},"ipraw":{"invalid_ip_version":0},"ltnull":{"pkt_too_small":0,"unsupported_type":0},"sctp":{"
pkt_too_small":0},"esp":{"pkt_too_small":0},"mpls":{"header_too_small":0,"pkt_too_small":0,"bad_label_router_alert":0,"bad_label_implicit_null":0,"bad_label_res
erved":0,"unknown_payload_type":0},"vxlan":{"unknown_payload_type":0},"geneve":{"unknown_payload_type":0},"erspan":{"header_too_small":0,"unsupported_version":0
,"too_many_vlan_layers":0},"dce":{"pkt_too_small":0},"chdlc":{"pkt_too_small":0},"nsh":{"header_too_small":0,"unsupported_version":0,"bad_header_length":0,"rese
rved_type":0,"unsupported_type":0,"unknown_payload":0}},"too_many_layers":0},"tcp":{"syn":0,"synack":0,"rst":0,"active_sessions":0,"sessions":0,"ssn_memcap_drop
":0,"ssn_from_cache":0,"ssn_from_pool":0,"pseudo":0,"pseudo_failed":0,"invalid_checksum":0,"midstream_pickups":0,"pkt_on_wrong_thread":0,"ack_unseen_data":0,"se
gment_memcap_drop":0,"segment_from_cache":0,"segment_from_pool":0,"stream_depth_reached":0,"reassembly_gap":0,"overlap":0,"overlap_diff_data":0,"insert_data_nor
mal_fail":0,"insert_data_overlap_fail":0,"memuse":1818624,"reassembly_memuse":344064},"flow":{"memcap":0,"total":8,"active":2,"tcp":0,"udp":8,"icmpv4":0,"icmpv6
":0,"tcp_reuse":0,"get_used":0,"get_used_eval":0,"get_used_eval_reject":0,"get_used_eval_busy":0,"get_used_failed":0,"wrk":{"spare_sync_avg":100,"spare_sync":1,
"spare_sync_incomplete":0,"spare_sync_empty":0,"flows_evicted_needs_work":0,"flows_evicted_pkt_inject":0,"flows_evicted":0,"flows_injected":0,"flows_injected_ma
x":0},"end":{"state":{"new":6,"established":0,"closed":0,"local_bypassed":0,"capture_bypassed":0},"tcp_state":{"none":0,"syn_sent":0,"syn_recv":0,"established":
0,"fin_wait1":0,"fin_wait2":0,"time_wait":0,"last_ack":0,"close_wait":0,"closing":0,"closed":0},"tcp_liberal":0},"mgr":{"full_hash_pass":89,"rows_per_sec":6553,
"rows_maxlen":1,"flows_checked":14,"flows_notimeout":8,"flows_timeout":6,"flows_evicted":6,"flows_evicted_needs_work":0},"spare":9906,"emerg_mode_entered":0,"em
erg_mode_over":0,"recycler":{"recycled":6,"queue_avg":0,"queue_max":1},"memuse":7234304},"defrag":{"ipv4":{"fragments":0,"reassembled":0},"ipv6":{"fragments":0,
"reassembled":0},"max_frag_hits":0},"flow_bypassed":{"local_pkts":0,"local_bytes":0,"local_capture_pkts":0,"local_capture_bytes":0,"closed":0,"pkts":0,"bytes":0
},"detect":{"engines":[{"id":0,"last_reload":"2025-11-27T02:27:47.346546+0200","rules_loaded":46530,"rules_failed":0,"rules_skipped":0}],"alert":0,"alert_queue_
overflow":0,"alerts_suppressed":0},"app_layer":{"flow":{"http":0,"ftp":0,"smtp":0,"tls":0,"ssh":0,"imap":0,"smb":0,"dcerpc_tcp":0,"dns_tcp":0,"nfs_tcp":0,"ntp":
0,"ftp-data":0,"tftp":0,"ike":0,"krb5_tcp":0,"quic":0,"dhcp":0,"snmp":0,"sip":0,"rfb":0,"mqtt":0,"telnet":0,"rdp":0,"http2":0,"bittorrent-dht":0,"failed_tcp":0,
"dcerpc_udp":0,"dns_udp":0,"nfs_udp":0,"krb5_udp":0,"failed_udp":8},"tx":{"http":0,"ftp":0,"smtp":0,"tls":0,"ssh":0,"imap":0,"smb":0,"dcerpc_tcp":0,"dns_tcp":0,
"nfs_tcp":0,"ntp":0,"ftp-data":0,"tftp":0,"ike":0,"krb5_tcp":0,"quic":0,"dhcp":0,"snmp":0,"sip":0,"rfb":0,"mqtt":0,"telnet":0,"rdp":0,"http2":0,"bittorrent-dht"
:0,"dcerpc_udp":0,"dns_udp":0,"nfs_udp":0,"krb5_udp":0},"error":{"http":{"gap":0,"alloc":0,"parser":0,"internal":0},"ftp":{"gap":0,"alloc":0,"parser":0,"interna
l":0},"smtp":{"gap":0,"alloc":0,"parser":0,"internal":0},"tls":{"gap":0,"alloc":0,"parser":0,"internal":0},"ssh":{"gap":0,"alloc":0,"parser":0,"internal":0},"im
ap":{"gap":0,"alloc":0,"parser":0,"internal":0},"smb":{"gap":0,"alloc":0,"parser":0,"internal":0},"dcerpc_tcp":{"gap":0,"alloc":0,"parser":0,"internal":0},"dns_
tcp":{"gap":0,"alloc":0,"parser":0,"internal":0},"nfs_tcp":{"gap":0,"alloc":0,"parser":0,"internal":0},"ntp":{"gap":0,"alloc":0,"parser":0,"internal":0},"ftp-da
ta":{"gap":0,"alloc":0,"parser":0,"internal":0},"tftp":{"gap":0,"alloc":0,"parser":0,"internal":0},"ike":{"gap":0,"alloc":0,"parser":0,"internal":0},"krb5_tcp":
{"gap":0,"alloc":0,"parser":0,"internal":0},"quic":{"gap":0,"alloc":0,"parser":0,"internal":0},"dhcp":{"gap":0,"alloc":0,"parser":0,"internal":0},"snmp":{"gap":
0,"alloc":0,"parser":0,"internal":0},"sip":{"gap":0,"alloc":0,"parser":0,"internal":0},"rfb":{"gap":0,"alloc":0,"parser":0,"internal":0},"mqtt":{"gap":0,"alloc"
:0,"parser":0,"internal":0},"telnet":{"gap":0,"alloc":0,"parser":0,"internal":0},"rdp":{"gap":0,"alloc":0,"parser":0,"internal":0},"http2":{"gap":0,"alloc":0,"p
arser":0,"internal":0},"bittorrent-dht":{"gap":0,"alloc":0,"parser":0,"internal":0},"failed_tcp":{"gap":0},"dcerpc_udp":{"alloc":0,"parser":0,"internal":0},"dns
_udp":{"alloc":0,"parser":0,"internal":0},"nfs_udp":{"alloc":0,"parser":0,"internal":0},"krb5_udp":{"alloc":0,"parser":0,"internal":0}},"expectations":0},"memca
p_pressure":5,"memcap_pressure_max":5,"http":{"memuse":0,"memcap":0},"ftp":{"memuse":0,"memcap":0},"file_store":{"open_files":0}}}
[Thu Nov 27 02:34:27] afnansensor@sensor:~$
```


Figure 24: We can see it from Suricata

4. **Run Iodine:-**

First, we start the tool on the victim to create the tunnel and to deal with the requests that arrive from the attacker and process them and forward the responses to him.

```
aseel@aseel-VMware-Virtual-Platform:~$ sudo iodined -f -P 1234 -l 1
92.168.17.133 -p 53 10.0.0.1/24 tunnel.test
Opened dns1
Setting IP of dns1 to 10.0.0.1
Setting MTU of dns1 to 1130
Opened IPv4 UDP socket
Listening to dns for domain tunnel.test
```

Figure 25: start iodine on victim

Command: sudo iodined –f –P 1234 -l 192.168.17.133 –p 10.0.0.1/24 tunnel.test: through this command, we create a tunnel and set the password to it (1234), specifying the server's IP (192.168.17.133), the port will listen to it (53), the determine the IP range within which the tunnel will work (10.0.0.1/24), and determine the domain that will be used within the queries (tunnel.test).

```
┌──(kali㉿kali)-[~]
└─$ sudo iodine -P 1234 192.168.17.133 tunnel.test

Opened dns2
Opened IPv4 UDP socket
Sending DNS queries for tunnel.test to 192.168.17.133
Autodetecting DNS query type (use -T to override).
Using DNS type NULL queries
Version ok, both using protocol v 0×00000502. You are user #0
Setting IP of dns2 to 10.0.0.2
Setting MTU of dns2 to 1130
Server tunnel IP is 10.0.0.1
Testing raw UDP data to the server (skip with -r)
Server is at 192.168.17.133, trying raw login: OK
Sending raw traffic directly to 192.168.17.133
Connection setup complete, transmitting data.
Detaching from terminal ...

┌──(kali㉿kali)-[~]
└─$
```

Figure 26: start iodine on attacker

Command: sudo iodined –P 1234 -l 192.168.17.133 tunnel.test: through this command, we run the iodine tool on the attacker's machine and put the password for it (1234), specifying the server's IP (192.168.17.133), and determining the domain that will be used within the queries (tunnel.test). Through it, encrypted data is sent within queries for the same domain.

Meanwhile, the Suricata and Zeek tools captured the DNS queries and gave an alerts to them, and the type of the queries that were detected is: Null queries and long TXT queries.

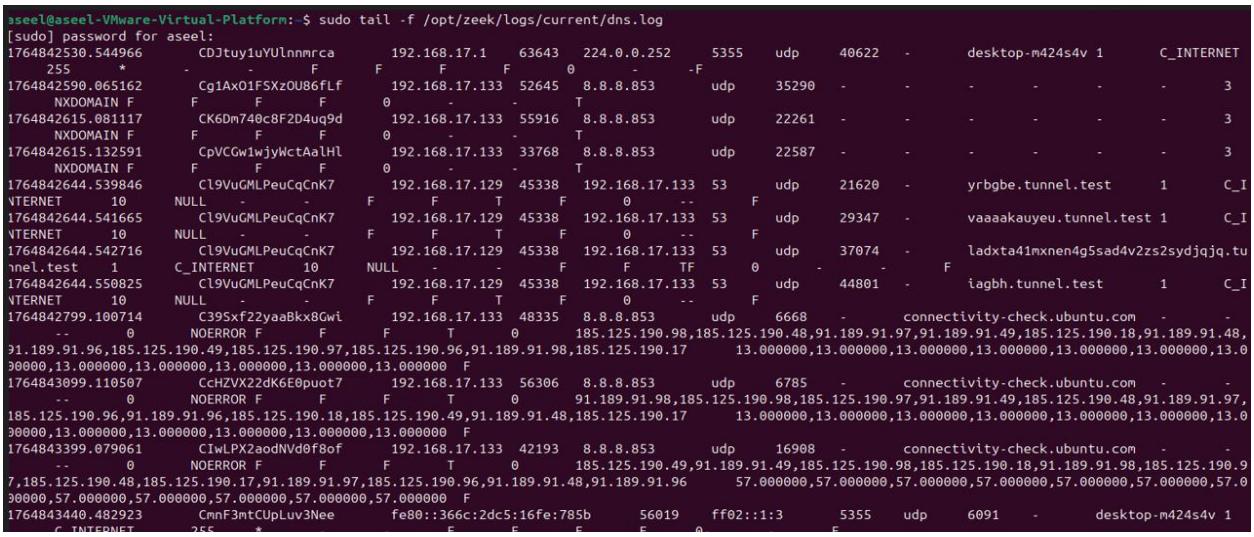

Figure 27: Zeek detection

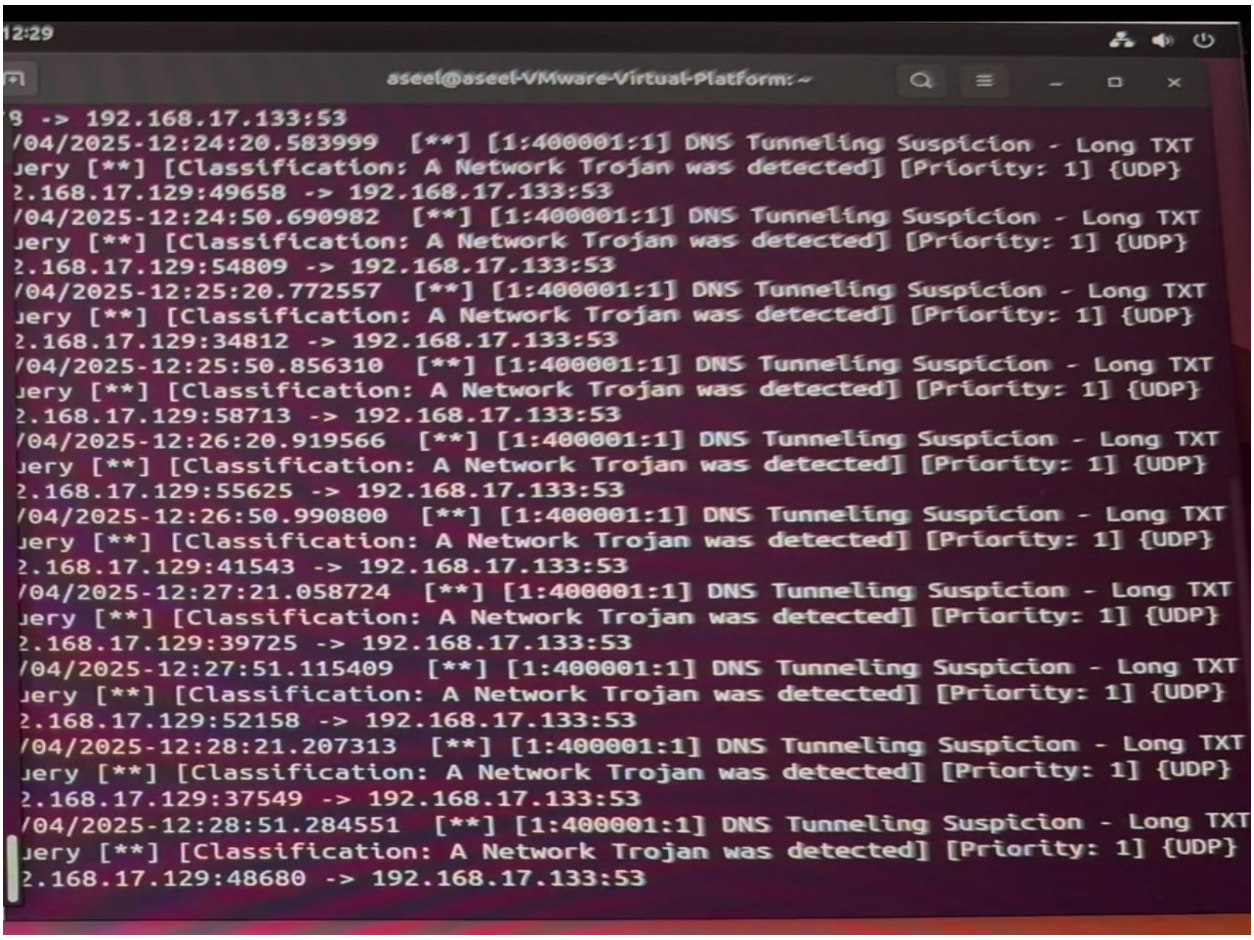

Figure 28: Suricata detection 1

```
aseel@aseel-VMware-Virtual-Platform:~$ sudo tail -f /var/log/suricata/fast.log
[sudo] password for aseel:
12/04/2025-11:46:43.896377  [**] [1:2029994:1] ET HUNTING Suspicious NULL DNS Request [**] [Classification: Misc activity] [Priority: 3] {UDP} 192.
168.17.129:59139 -> 192.168.17.133:53
12/04/2025-12:01:19.189207  [**] [1:2029994:1] ET HUNTING Suspicious NULL DNS Request [**] [Classification: Misc activity] [Priority: 3] {UDP} 192.
168.17.129:47407 -> 192.168.17.133:53
12/04/2025-12:01:26.174894  [**] [1:2029994:1] ET HUNTING Suspicious NULL DNS Request [**] [Classification: Misc activity] [Priority: 3] {UDP} 192.
168.17.129:47407 -> 192.168.17.133:53
12/04/2025-12:01:29.883258  [**] [1:2022973:1] ET INFO Possible Kali Linux hostname in DHCP Request Packet [**] [Classification: Potential Corporat
e Privacy Violation] [Priority: 1] {UDP} 192.168.17.129:68 -> 192.168.17.254:67
12/04/2025-12:01:40.295974  [**] [1:2029994:1] ET HUNTING Suspicious NULL DNS Request [**] [Classification: Misc activity] [Priority: 3] {UDP} 192.
168.17.129:47407 -> 192.168.17.133:53
12/04/2025-12:04:04.539846  [**] [1:2029994:1] ET HUNTING Suspicious NULL DNS Request [**] [Classification: Misc activity] [Priority: 3] {UDP} 192.
168.17.129:45338 -> 192.168.17.133:53
12/04/2025-12:04:04.541665  [**] [1:2029994:1] ET HUNTING Suspicious NULL DNS Request [**] [Classification: Misc activity] [Priority: 3] {UDP} 192.
168.17.129:45338 -> 192.168.17.133:53
12/04/2025-12:04:04.542716  [**] [1:400001:1] DNS Tunneling Suspicion - Long TXT Query [**] [Classification: A Network Trojan was detected] [Priori
ty: 1] {UDP} 192.168.17.129:45338 -> 192.168.17.133:53
12/04/2025-12:04:04.542716  [**] [1:2029994:1] ET HUNTING Suspicious NULL DNS Request [**] [Classification: Misc activity] [Priority: 3] {UDP} 192.
168.17.129:45338 -> 192.168.17.133:53
12/04/2025-12:04:04.550825  [**] [1:2029994:1] ET HUNTING Suspicious NULL DNS Request [**] [Classification: Misc activity] [Priority: 3] {UDP} 192.
168.17.129:45338 -> 192.168.17.133:53
12/04/2025-12:16:29.908678  [**] [1:2022973:1] ET INFO Possible Kali Linux hostname in DHCP Request Packet [**] [Classification: Potential Corporat
e Privacy Violation] [Priority: 1] {UDP} 192.168.17.129:68 -> 192.168.17.254:67
```

Figure 29:Suricata detection 2

5. Other detects:-

- Sent requests for a domain that doesn't exist:

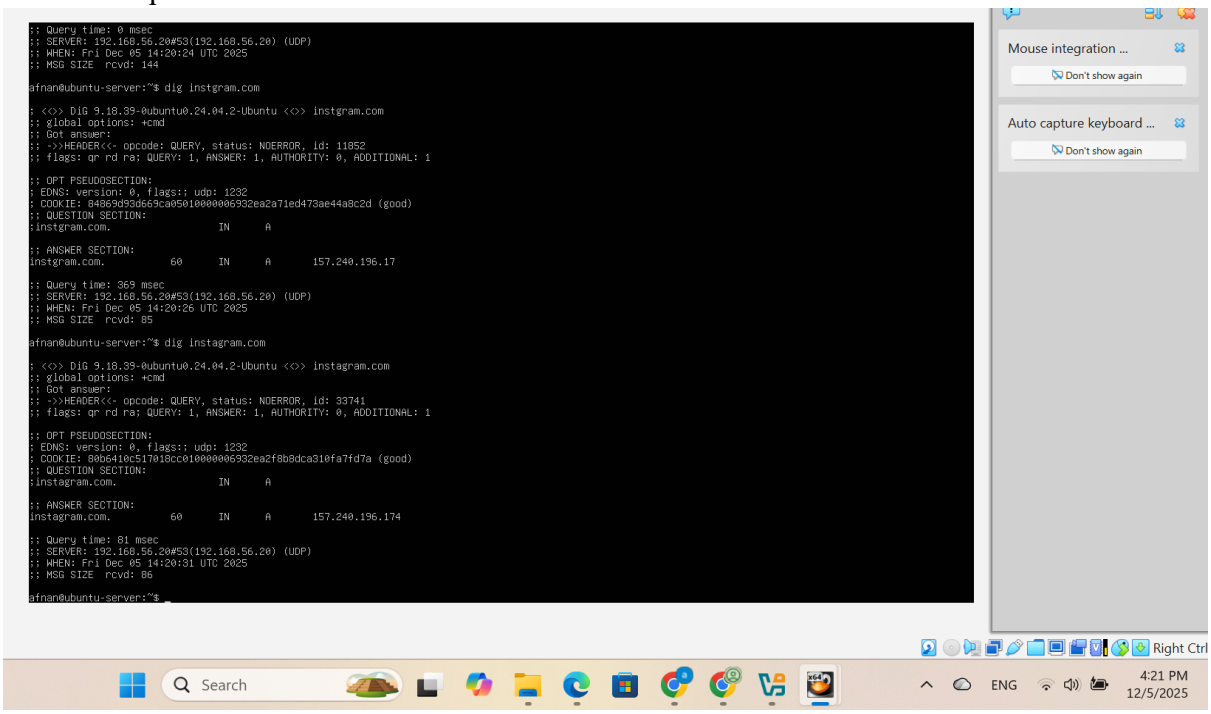

Figure 30: NXDOMAIN requests

Figure 31: Suricata detect

- Sent long TXT and high entropy requests:-

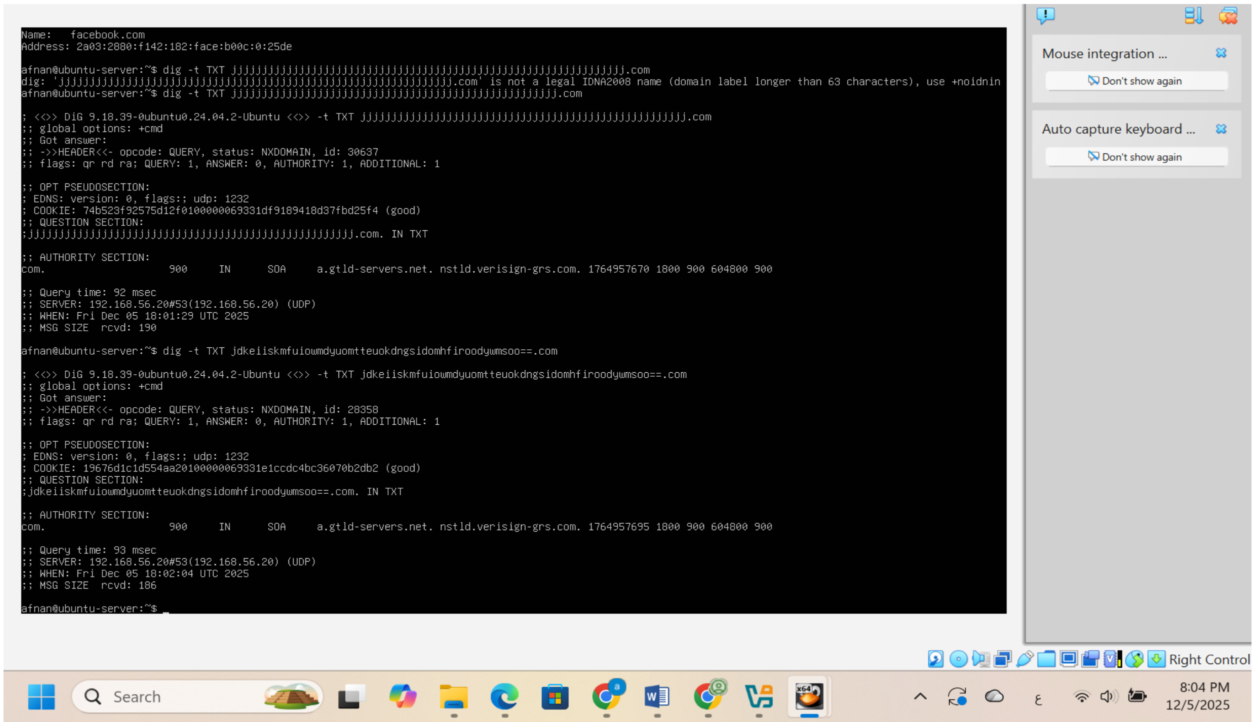

Figure 32: long TXT and high entropy requests

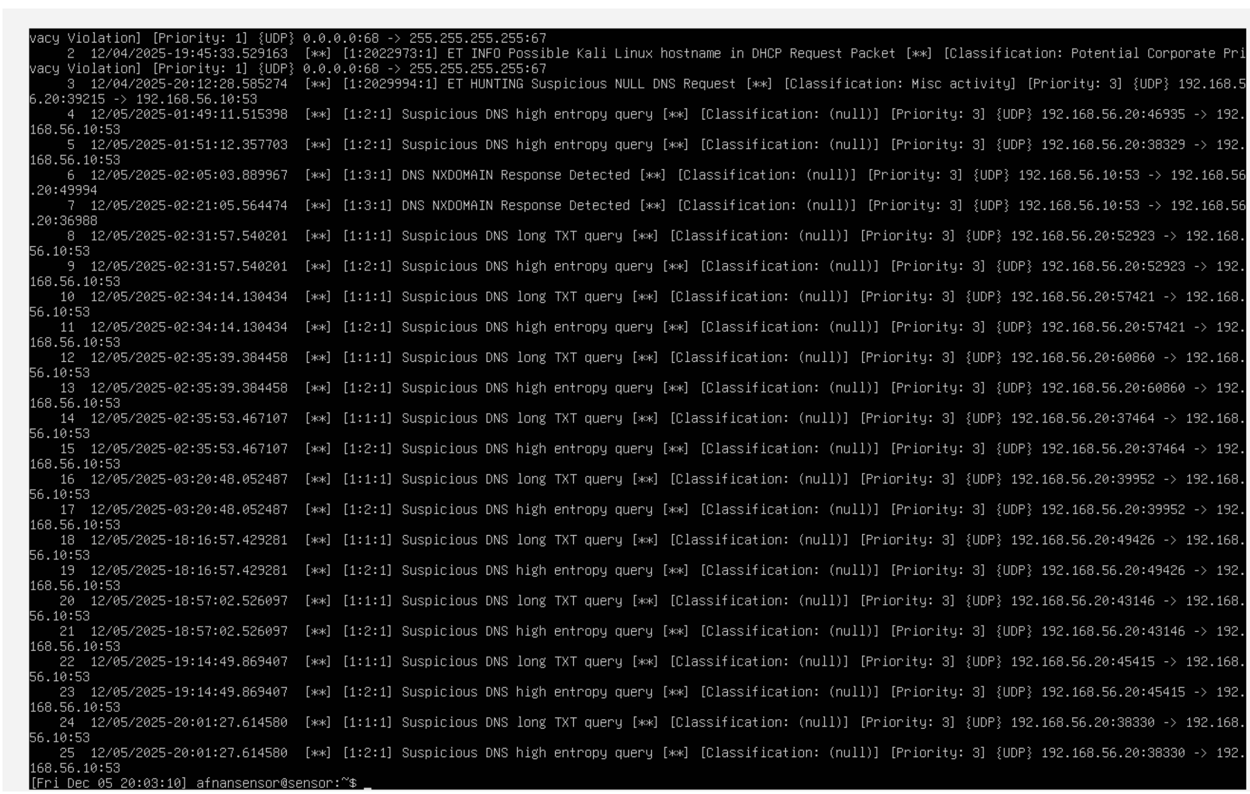

Figure 33: Suricata detect

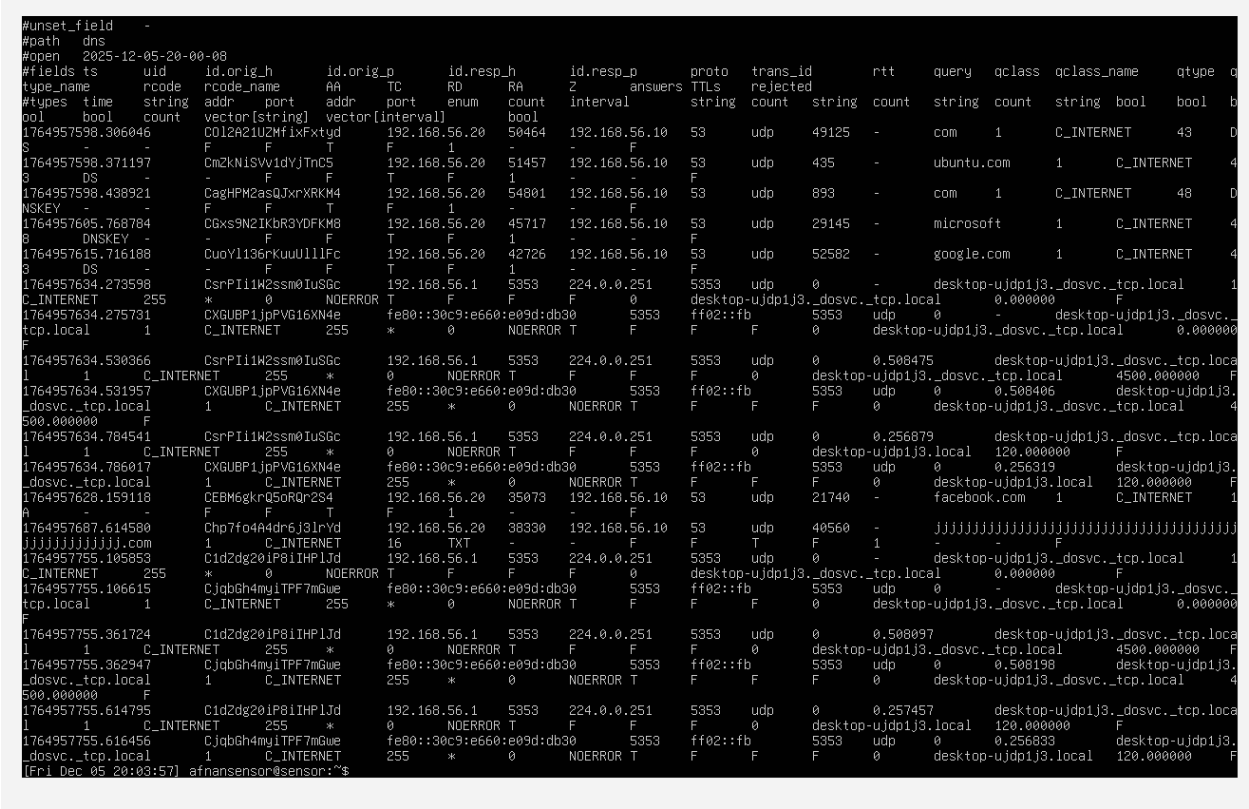

Figure 34: Zeek detect

6. Analysis and Metrics:-

As we saw before, Zeek and Suricata successeded in analyzing the DNS queries and detect the DNS tunneling.

The detection rate of the two tools was high, but some malicious requests were not captured or detected, and Zeek was giving a lot of false positive alerts, and there was a delay in capturing some requests and missing some of them. Out of every five requests, two were sometimes lost, and therefore the detection rate is more than 80% because we use more than one tool to detect.

As for the exploitation of resources, we had a little because of the small size of the network and traffic in it, but the consumption of space in Suricata was higher because of keep all files, even if they were restarted, but Zeek deletes them with every stop of the system, and Zeek consumes less CPU & Memory than Suricata.

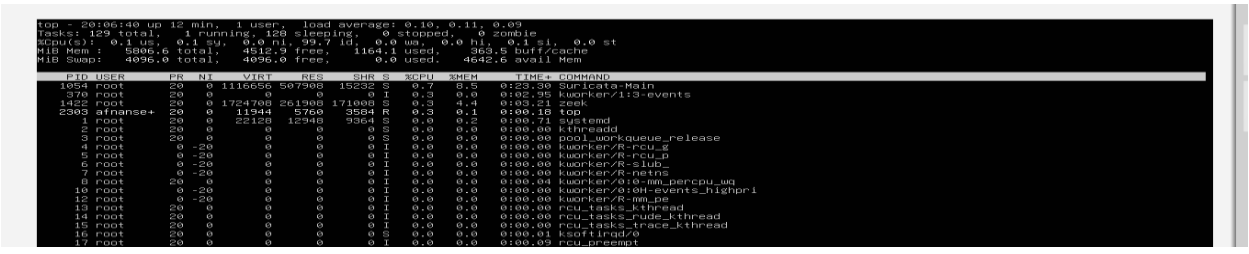

Figure 35: resources usage

7. **Block via IPS or host firewall guidance:-**
As for the prevention, we did it by:-

- Suricata IPS:-
We add the inline:yes line to the configuration file of Suricata to monitor the traffic in real time.

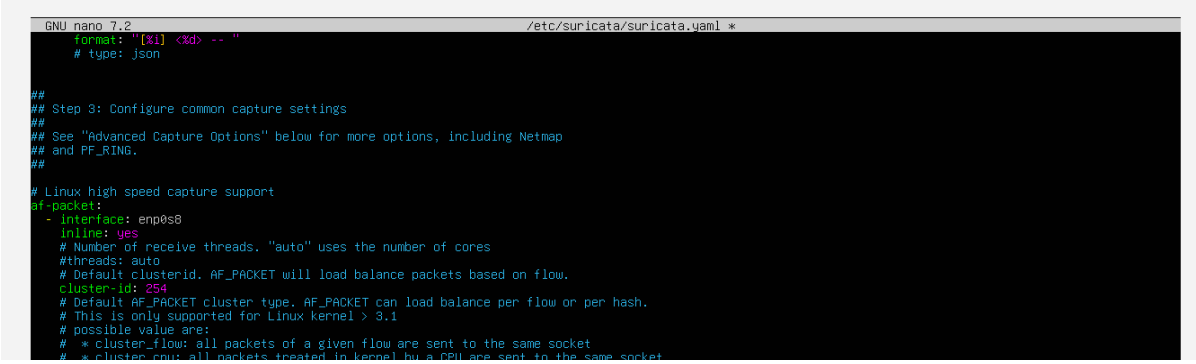

Figure 36: add IPS inline line

And we convert the alert in the rules to drop.

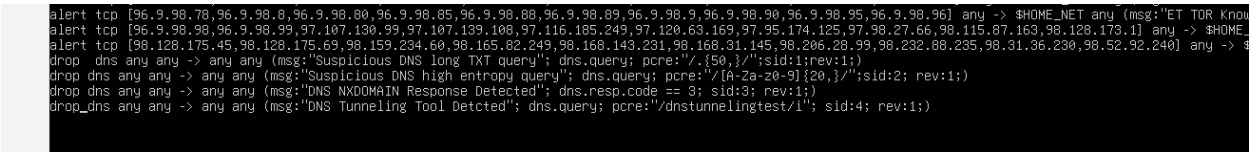

Figure 37: convert alert to drop

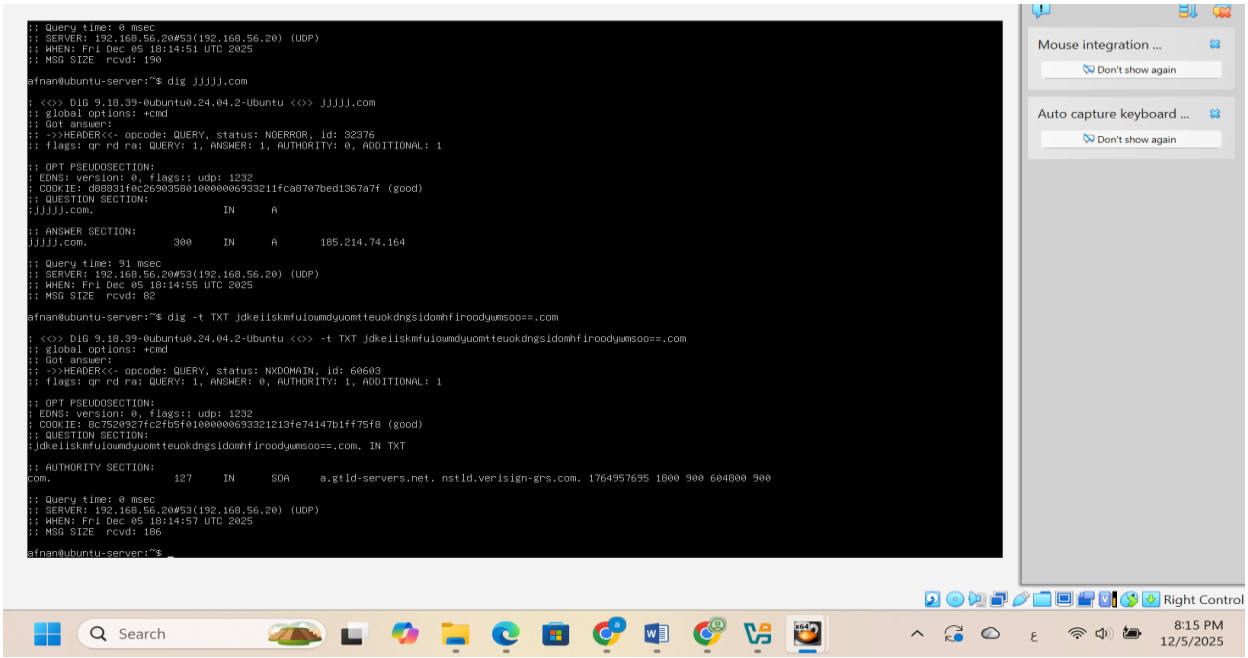

Figure 38: send requests

```
   712  [1060 - Suricata-Main] 2025-12-05 18:40:38 Info: runmodes: enp0s8: creating 3 threads
   713  [1060 - Suricata-Main] 2025-12-05 18:40:38 Info: unix-manager: unix socket '/var/suricata-command.socket'
   714  [1060 - Suricata-Main] 2025-12-05 18:40:38 Notice: threads: Threads created -> W: 3 FM: 1 FR: 1   Engine started.
   715  [1044 - Suricata-Main] 2025-12-05 19:53:58 Notice: suricata: This is Suricata version 7.0.3 RELEASE running in SYSTEM mode
   716  [1044 - Suricata-Main] 2025-12-05 19:53:58 Info: cpu: CPUs/cores online: 3
   717  [1044 - Suricata-Main] 2025-12-05 19:53:58 Info: suricata: Setting engine mode to IDS mode by default
   718  [1044 - Suricata-Main] 2025-12-05 19:53:58 Info: exception-policy: master exception-policy set to: auto
   719  [1044 - Suricata-Main] 2025-12-05 19:53:58 Info: ioctl: enp0s8: MTU 1500
   720  [1054 - Suricata-Main] 2025-12-05 19:53:58 Info: conf: Running in live mode, activating unix socket
   721  [1054 - Suricata-Main] 2025-12-05 19:53:58 Info: logopenfile: fast output device (regular) initialized: fast.log
   722  [1054 - Suricata-Main] 2025-12-05 19:53:58 Info: logopenfile: eve-log output device (regular) initialized: /var/log/suricata/eve.json
   723  [1054 - Suricata-Main] 2025-12-05 19:53:58 Info: logopenfile: stats output device (regular) initialized: stats.log
   724  [1054 - Suricata-Main] 2025-12-05 19:53:58 Error: detect-parse: An invalid action "0alert" was given
   725  [1054 - Suricata-Main] 2025-12-05 19:53:58 Error: detect: error parsing signature "0alert ip any any -> any any (msg:"SURICATA Applayer Mismatch protoco
l both directions"; flow:established; app-layer-event:applayer_mismatch_protocol_both_directions; flowint:applayer.anomaly.count,+,1; classtype:protocol-command
-decode; sid:2260000; rev:1;)" from file /var/lib/suricata/rules/suricata.rules at line 1
   726  [1054 - Suricata-Main] 2025-12-05 19:54:03 Error: detect-parse: unknown rule keyword 'dns.resp.code == 3'.
   727  [1054 - Suricata-Main] 2025-12-05 19:54:03 Error: detect: error parsing signature "alert dns any any -> any any (msg:"DNS NXDOMAIN Response Detected"; d
ns.resp.code == 3; sid:3; rev:1;)" from file /var/lib/suricata/rules/suricata.rules at line 62346
   728  [1054 - Suricata-Main] 2025-12-05 19:54:03 Info: detect: 1 rule files processed. 46530 rules successfully loaded, 2 rules failed, 0
   729  [1054 - Suricata-Main] 2025-12-05 19:54:03 Info: threshold-config: Threshold config parsed: 0 rule(s) found
   730  [1054 - Suricata-Main] 2025-12-05 19:54:03 Info: detect: 46533 signatures processed. 1059 are IP-only rules, 4422 are inspecting packet payload, 40823 i
nspect application layer, 108 are decoder event only
   731  [1054 - Suricata-Main] 2025-12-05 19:54:13 Info: runmodes: enp0s8: creating 3 threads
   732  [1054 - Suricata-Main] 2025-12-05 19:54:13 Info: unix-manager: unix socket '/var/suricata-command.socket'
   733  [1054 - Suricata-Main] 2025-12-05 19:54:14 Notice: threads: Threads created -> W: 3 FM: 1 FR: 1   Engine started.
   734  [1054 - Suricata-Main] 2025-12-05 20:13:12 Notice: suricata: Signal Received.  Stopping engine.
   735  [1054 - Suricata-Main] 2025-12-05 20:13:12 Info: suricata: time elapsed 1139.075s
   736  [1054 - Suricata-Main] 2025-12-05 20:13:13 Info: counters: Alerts: 2
   737  [1054 - Suricata-Main] 2025-12-05 20:13:13 Notice: device: enp0s8: packets: 186, drops: 0 (0.00%), invalid chksum: 0
   738  [2328 - Suricata-Main] 2025-12-05 20:13:14 Notice: suricata: This is Suricata version 7.0.3 RELEASE running in SYSTEM mode
   739  [2328 - Suricata-Main] 2025-12-05 20:13:14 Info: cpu: CPUs/cores online: 3
   740  [2328 - Suricata-Main] 2025-12-05 20:13:14 Info: suricata: Setting engine mode to IDS mode by default
   741  [2328 - Suricata-Main] 2025-12-05 20:13:14 Info: exception-policy: master exception-policy set to: auto
   742  [2328 - Suricata-Main] 2025-12-05 20:13:14 Info: ioctl: enp0s8: MTU 1500
   743  [2330 - Suricata-Main] 2025-12-05 20:13:14 Info: conf: Running in live mode, activating unix socket
   744  [2330 - Suricata-Main] 2025-12-05 20:13:14 Info: logopenfile: fast output device (regular) initialized: fast.log
   745  [2330 - Suricata-Main] 2025-12-05 20:13:14 Info: logopenfile: eve-log output device (regular) initialized: /var/log/suricata/eve.json
   746  [2330 - Suricata-Main] 2025-12-05 20:13:14 Info: logopenfile: stats output device (regular) initialized: stats.log
   747  [2330 - Suricata-Main] 2025-12-05 20:13:18 Error: detect-parse: unknown rule keyword 'dns.resp.code == 3'.
   748  [2330 - Suricata-Main] 2025-12-05 20:13:18 Error: detect: error parsing signature "drop dns any any -> any any (msg:"DNS NXDOMAIN Response Detected"; dn
s.resp.code == 3; sid:3; rev:1;)" from file /var/lib/suricata/rules/suricata.rules at line 62346
   749  [2330 - Suricata-Main] 2025-12-05 20:13:18 Info: detect: 1 rule files processed. 46531 rules successfully loaded, 1 rules failed, 0
   750  [2330 - Suricata-Main] 2025-12-05 20:13:18 Info: threshold-config: Threshold config parsed: 0 rule(s) found
   751  [2330 - Suricata-Main] 2025-12-05 20:13:18 Info: detect: 46534 signatures processed. 1059 are IP-only rules, 4422 are inspecting packet payload, 40823 i
nspect application layer, 108 are decoder event only
   752  [2330 - Suricata-Main] 2025-12-05 20:13:28 Info: runmodes: enp0s8: creating 3 threads
   753  [2330 - Suricata-Main] 2025-12-05 20:13:28 Info: unix-manager: unix socket '/var/suricata-command.socket'
   754  [2330 - Suricata-Main] 2025-12-05 20:13:28 Notice: threads: Threads created -> W: 3 FM: 1 FR: 1   Engine started.
[Fri Dec 05 20:15:00] afnansensor@sensor:~$
```

Figure 39: Suricata Prevention

- Firwall:-

Limited requests, only 10 requests per minute and block requests from a specific source (192.168.56.20).

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
[Fri Dec 05 20:28:02] afnansensor@sensor:~$ sudo systemctl enable --now firewalld
[Fri Dec 05 20:28:28] afnansensor@sensor:~$ sudo firewall-cmd --state
running
[Fri Dec 05 20:28:42] afnansensor@sensor:~$ sudo firewall-cmd --zone=public --add-port=53/udp --permanent
success
[Fri Dec 05 20:28:57] afnansensor@sensor:~$ sudo firewall-cmd --zone=public --add-port=53tcp --permanent
Error: INVALID_PORT: bad port (most likely missing protocol), correct syntax is portid[-portid]/protocol
[Fri Dec 05 20:29:04] afnansensor@sensor:~$ sudo firewall-cmd --zone=public --add-port=53/tcp --permanent
success
[Fri Dec 05 20:29:11] afnansensor@sensor:~$ sudo firewall-cmd --reload
success
[Fri Dec 05 20:29:23] afnansensor@sensor:~$ sudo firewall-cmd --zone=public --add-rich-rule='rule family="ipv4" service name="dns" limit value="10/m" accept' --
permanent
success
[Fri Dec 05 20:31:30] afnansensor@sensor:~$ sudo firewall-cmd --reload
success
[Fri Dec 05 20:31:39] afnansensor@sensor:~$ sudo firewall-cmd --zone=public --add-rich-rule='rule family="ipv4" sourcer address="192.168.56.20" service name="dn
s" limit value="10/m" accept' --permanent
Error: INVALID_RULE: unknown element sourcer
[Fri Dec 05 20:33:01] afnansensor@sensor:~$ sudo firewall-cmd --zone=public --add-rich-rule='rule family="ipv4" source address="192.168.56.20" service name="dns
" limit value="10/m" accept' --permanent
success
[Fri Dec 05 20:33:24] afnansensor@sensor:~$
```

Figure 40: Firewall sittings

Challenges:-

- The difficulty of writing rules and scripts manually and the difficulty of reconciling them completely.
- The Zeek tool deletes files when the system is shutdown.
- Difficulty reading the eve.json file in Suricata tool to monitoring the network traffic.

Recommendations:-

- We observed that adding a simple dashboard to display results and alerts is preferable. This would provide a faster way to monitor traffic and alerts, rather than relying solely on log files. It would facilitate monitoring during any attack, making the system easier to use and understand.
- Suricata's rules and Zeek's scripts must be continuously developed to improve their accuracy, avoid false positive alerts, and easily distinguish between legitimate and malicious traffic.
- Running Zeek for a longer period provides a clearer picture of legitimate network traffic, making the detection of any unusual activity easier and more accurate.
- After testing, we observed that recording observations at each stage of the process helped us track errors and improve results. Therefore, it is recommended to document steps and modifications regularly to facilitate future development and review.

Conclusion:-

This report highlights the importance of combining behavioral analysis and signature detection in a network security environment. Zeek provides the ability to monitor and analyze DNS traffic and extract anomalous behavior, while Suricata acts as a real-time detection layer, creating custom rules and implementing prevention measures. Attack testing was conducted using DNS Tunneling tools on a Kali machine, while Sensor VM analyzed the traffic and generated appropriate alerts.

An integrated environment was provided for proof of concept and documentation of results. The experiment confirms that combining the two mechanisms provides a higher level of detection accuracy and reduces the chances of successful DNS attacks.

References:-

Suricata rules

Zeek Scripts

Git Repo