

# METHOD AND NET DEATH

---

## MinProject II

**Presented To:** Dr. Mohamad Awdi

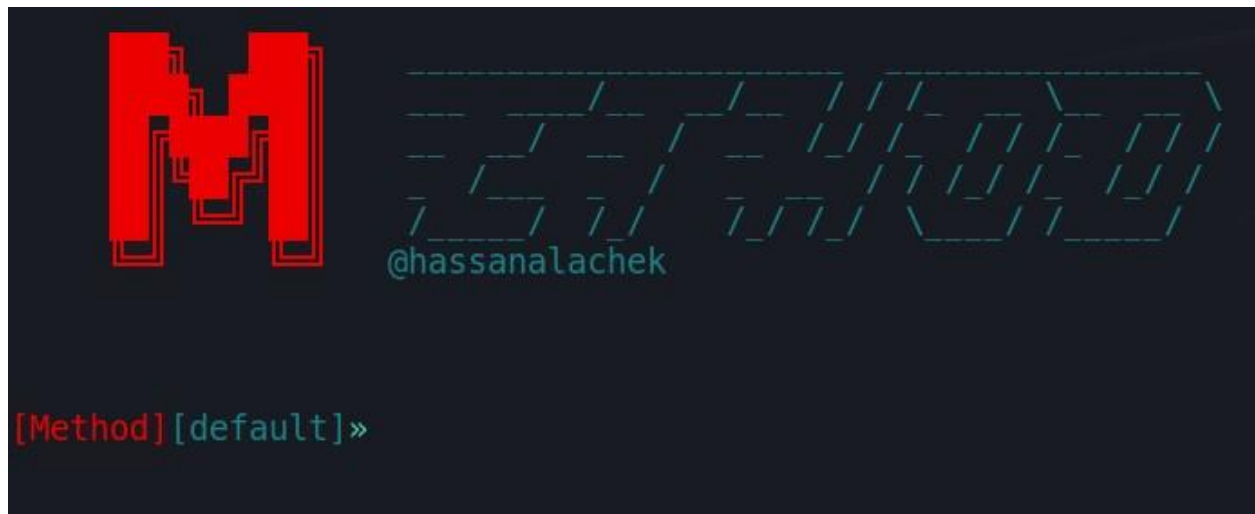
**Hassan Al Achek 5567**

**Hassan Al Achek 5579**

## Introduction:

In our project I am trying to develop two tools. These tools can be used by penetration testers trying to penetrate into an organization.

In the figure below we can see the first tool method:

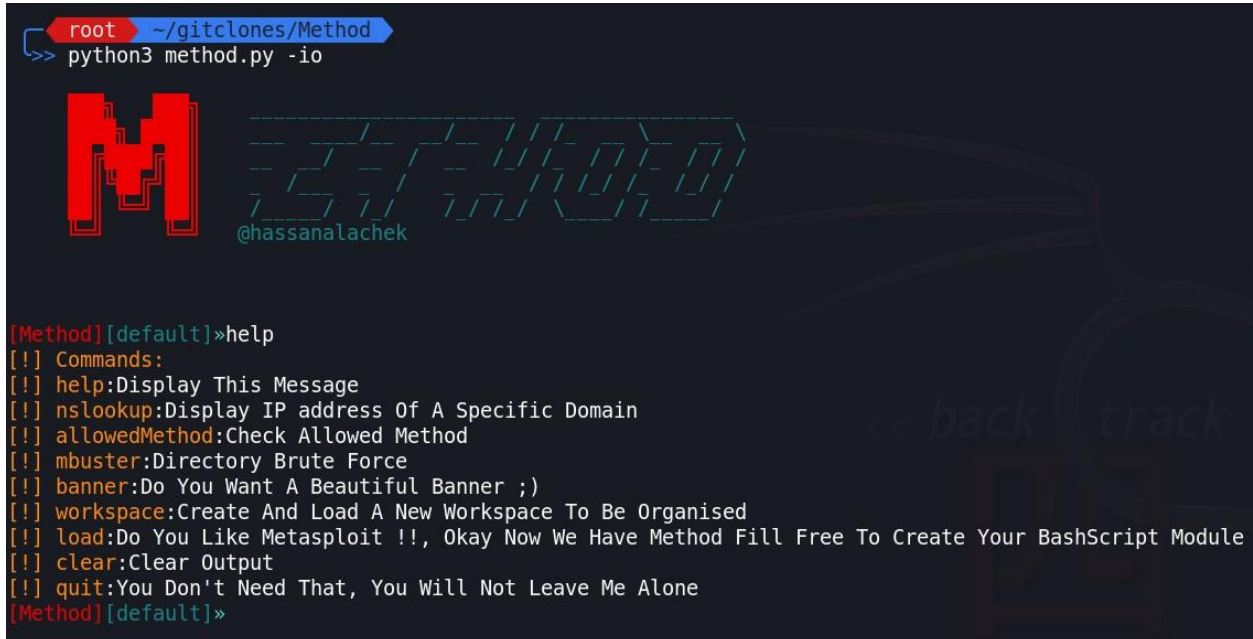


In this figure we can see the second tool net death:



## Method:

Will i am coding, i am trying to make the tool flexible while used, so i try to make an interactive shell in which the user can type command and get the result of command execution back in a well formatted from, the autocomplete functionality added to the interactive shell make it more and more flexible, now user can use TAB to complete command automatically.



```
root ~/gitclones/Method
>> python3 method.py -io

M
METHOD
@hassanalachek

[Method][default]>>help
[!] Commands:
[!] help:Display This Message
[!] nslookup:Display IP address Of A Specific Domain
[!] allowedMethod:Check Allowed Method
[!] mbuster:Directory Brute Force
[!] banner:Do You Want A Beautiful Banner ;)
[!] workspace:Create And Load A New Workspace To Be Organised
[!] load:Do You Like Metasploit !!, Okay Now We Have Method Fill Free To Create Your BashScript Module
[!] clear:Clear Output
[!] quit:You Don't Need That, You Will Not Leave Me Alone
[Method][default]>>
```

In the figure above we can see the help menu of Method.

## Allowed Method:

This command is used to fingerprint the allowed HTTP method in the webserver.

Some HTTP methods will be dangerous like DELETE and PUT, by fingerprinting this method an attacker can try to either DELETE items and cause damage in the target web server, either PUT some backdoors in the target webserver, In this case the attacker will cause a big damage which can compromise the business hosted behind this webserver.

Allowed methods will take a url or a list of urls in a txt file, and return as a result a txt file that contains the allowed method for each webserver.

```
root ~/gitclones/Method
>> service apache2 status
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; disabled; vendor preset: disabled)
   Active: active (running) since Mon 2021-06-28 13:54:39 EDT; 10s ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 3331 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
  Main PID: 3342 (apache2)
    Tasks: 7 (limit: 4638)
   Memory: 19.2M
      CPU: 209ms
   CGroup: /system.slice/apache2.service
           └─3342 /usr/sbin/apache2 -k start
             └─3343 /usr/sbin/apache2 -k start
               └─3344 /usr/sbin/apache2 -k start
                 └─3345 /usr/sbin/apache2 -k start
                   └─3346 /usr/sbin/apache2 -k start
                     └─3347 /usr/sbin/apache2 -k start
                       └─3348 /usr/sbin/apache2 -k start
```

In the figure above, I am running an apache web server.

```
[Method][default]>>allowedMethod
[!] usage: allowedMethod <urlsList>[!]
[?] Example allowedMethod urlList.txt[?]

[Method][default]>>allowedMethod urlList.txt
[+] The Following Method are allowed for 192.168.0.114:POST,OPTIONS,HEAD,GET[+]
[+] 200:OK [ +]
```

In this figure, It's clear allowed method give me allowed HTTP method in my local web server

## Mbuster:

Mbuster is a directory brute forcing command, as we know in a web server. Maybe the developer forked some important directories accessed from anyone in the internet, and this will lead to an information disclosure. And maybe in the case of credential disclosure like database credential it will lead to compromise the entire web server.

In the below figure, It's clear that mbuster command will take a url, and then brute force using a dir.txt file, and try to locate the status code of the response (200: Ok and 404: Not Found)

```
[Method][default]>mbuster
[!] usage: mbuster <url> <urlList> <port>[!]
[?] Example: mbuster www.example.com dir.txt 443[?]

[Method][default]>mbuster 192.168.0.114 dir.txt 80
[+] 192.168.0.114:dir.txt[+]
[?] https/http[?] http
[http://192.168.0.114/BlackHat/]:[200]
[http://192.168.0.114:80/CrackPass.sh]:[200]
[http://192.168.0.114:80/index.html]:[200]
[http://192.168.0.114:80/index.php]:[404]
[http://192.168.0.114:80/admin]:[404]
[Method][default]>
```

## Workspace:

To organize and group the result in one location, I added the workspace command. This command will create a local directory and save all the results in this directory.

```
[Method][default]>workspace
[!] usage: workspace <WorkspaceName>[!]
[?] Example: workspace testworkspace[?]

[Method][default]>workspace Project
[Method][Project]>quit
[~] Bye!! ;) [~]

root ~/gitclones/Method
>> ls
autoComplete.py  color.py  error.py  language.py  method.py  Project  README.md  symbols.py  urlList.txt
banner.py        dir.txt  install.sh  Method.jpg  modules    __pycache__  shell.py   testing    webMethod.py
```

## Nslookup:

This command used to get IPV4 and IPV6, from the corresponding web server, will resolve the URL and display the result.

```
[Method][default]»nslookup
[!] usage: nslookup <Domain> [!]
[?] Example: nslookup www.example.com [?]

[Method][default]»nslookup www.google.com
[+] -----[+]
172.217.18.36
172.217.18.36
172.217.18.36
2a00:1450:4006:801::2004
2a00:1450:4006:801::2004
2a00:1450:4006:801::2004
[+] -----[+]
```

## Other Commands:

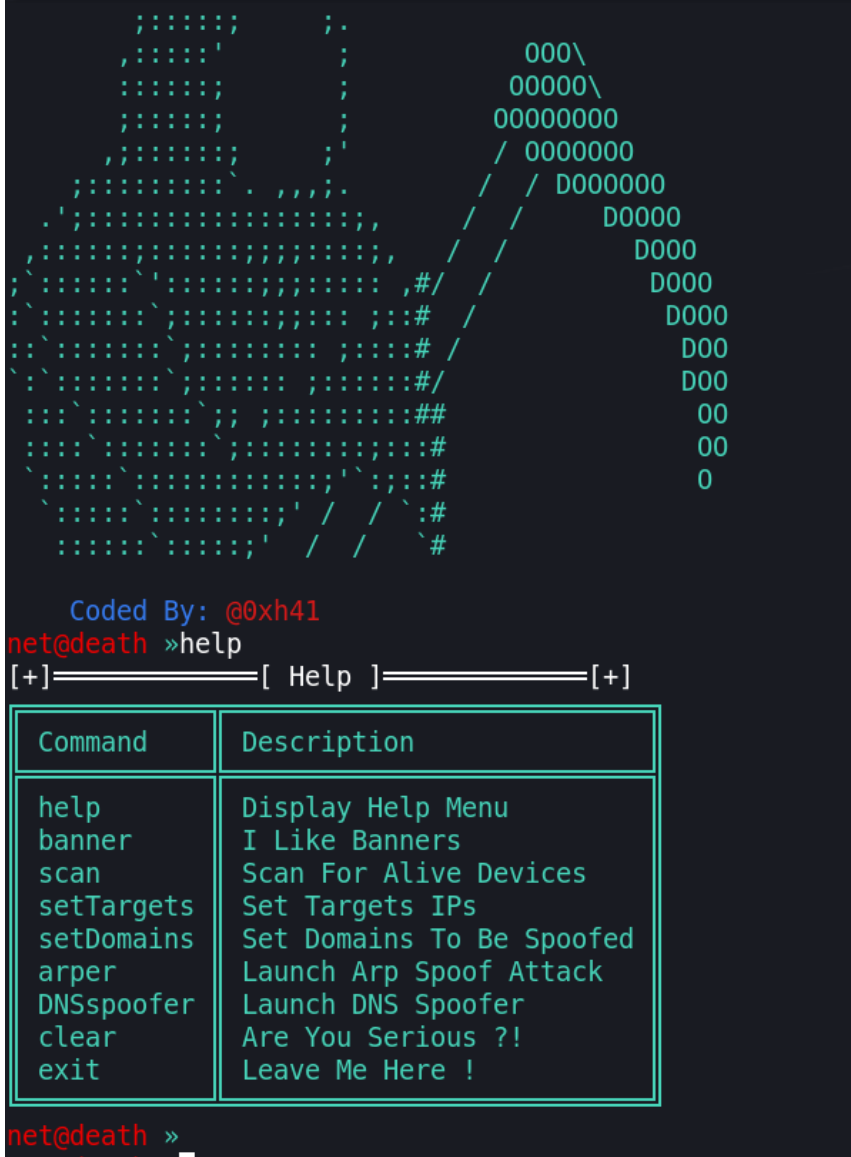
clear: to clear active terminal

quit: to exit from the interactive shell

banner: to display some ASCII and ANSI arts

## Net Death:

This tool was developed to collect network attacks. It's not complete but it has two main functions. And for sure it's an interactive tool that can scan networks and get IP, MAC address, and Manufacturer, It allows the user to execute Man-In-The-Middle attacks using the arp protocol, then we will be able to execute dns spoofing to spoof some legitimate websites.



## Scan:

This command used to map the network and get the following informations:

IP, MAC address, Manufacturer, as in the following figure:

```
net@death »scan
[=====]
[+]===== [ Scan Result ]===== [+]



| IP Address    | Mac Address       | Manufacturer       |
|---------------|-------------------|--------------------|
| 192.168.0.1   | C8:3A:35:61:B8:78 | (Tenda Technology) |
| 192.168.0.110 | 10:F0:05:CA:20:7E | (Intel Corporate)  |
| 192.168.0.114 | 08:00:27:1F:97:34 | (This device)      |



net@death »
```

This command is based in the famous network scanning tool nmap



## Set Targets:

After scanning the network for available devices we will be able to set targets.

The setTargets command will display a checklist for us to choose two victims for further attacks.

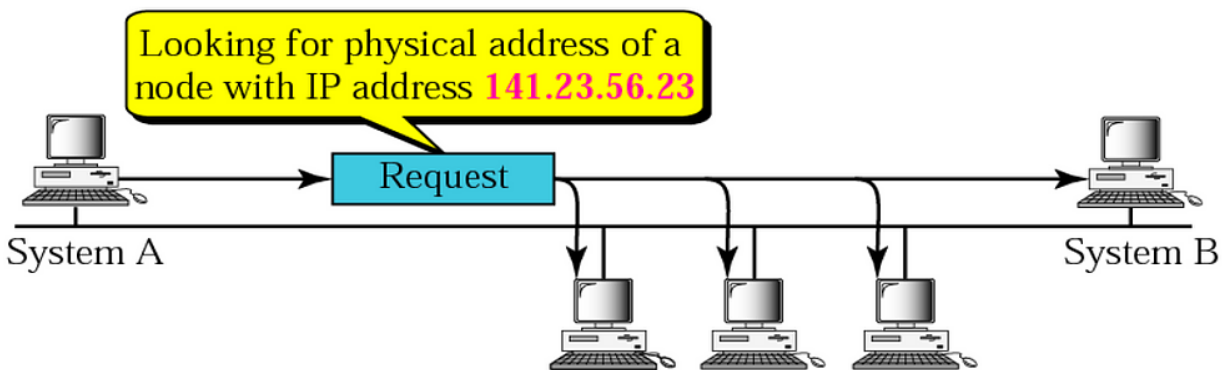
```
net@death »setTargets
[?] Targets:
> o 192.168.0.1
  o 192.168.0.110
  o 192.168.0.114
```

## Arper:

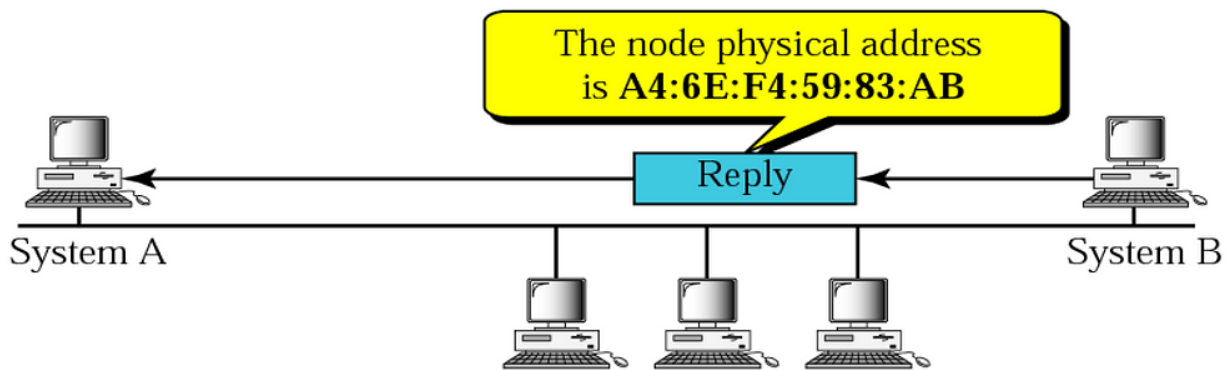
After setting targets, we can launch an arp spoofing attack.



## Arp (Address Resolution Protocol):



a. ARP request is broadcast



b. ARP reply is unicast

Arper command will launch a new terminal with a new thread, and start poison MAC address table cache in the targets.

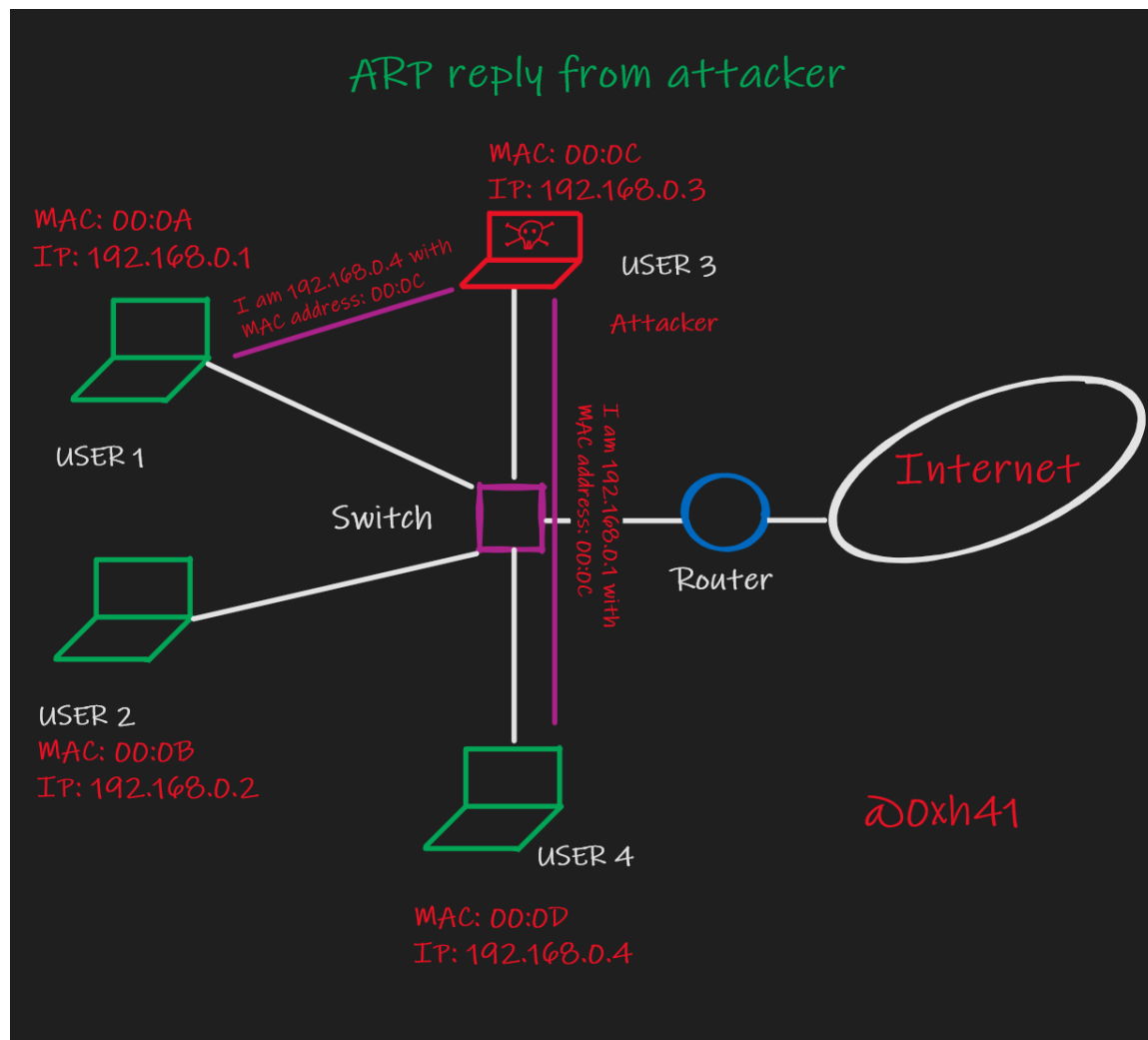
```
/bin/bash
net@ /bin/bash 80x24
[WARNING: This interface is not specified in any provider ! See conf.ifaces output]
net@ [t] [ Start Arp ] [t]
Initialized 192.168.0.1:
Gateway 192.168.0.110 is at 10:f0:05:ca:20:7e
Victim 192.168.0.1 is at c8:3a:35:61:b8:78

IP Source: 192.168.0.110
IP Destination: 192.168.0.1
MAC Destination: c8:3a:35:61:b8:78
MAC Source: 08:00:27:1f:97:34
ARP is at 08:00:27:1f:97:34 says 192.168.0.110

IP Source: 192.168.0.1
IP Destination: 192.168.0.110
MAC Destination: b'192.168.0.110'
MAC Source: 08:00:27:1f:97:34
ARP is at 08:00:27:1f:97:34 says 192.168.0.1

Start Poisining:[ctrl-c to stop]
=
```

As we can see in the following figure, the attacker now is in the middle between these two victims, and it's now able to sniff packets and change them.



## DNS Spoofing:

First we need to add a domain to be spoofed using the setTargets command.

```
net@death » setDomains  
[?] [?] Enter Domain(i.e: google.com) [?]: google.com  
[?] [?] Enter IP(i.e: your machine IP(No local host) [?]: 192.168.0.114  
net@death »
```

Now we will be able to launch the DNS spoofer using the command: DNSspoofer

```

# /bin/bash 83x38
[After]: IP / UDP / DNS Ans "172.217.21.2"
[Before]: IP / UDP / DNS Ans "b'static-doubleclick-net.l.google.com."
no modification: b'static.doubleclick.net.'
[After]: IP / UDP / DNS Ans "b'static-doubleclick-net.l.google.com."
[Before]: IP / UDP / DNS Ans "172.217.19.34"
no modification: b'googleads.g.doubleclick.net.'
[After]: IP / UDP / DNS Ans "172.217.19.34"
[Before]: IP / UDP / DNS Ans "b'instagram.c10r.facebook.com."
no modification: b'i.instagram.com.'
[After]: IP / UDP / DNS Ans "b'instagram.c10r.facebook.com."
[Before]: IP / UDP / DNS Ans "b'mqtt-mini.c10r.facebook.com."
no modification: b'mqtt-mini.facebook.com.'
[After]: IP / UDP / DNS Ans "b'mqtt-mini.c10r.facebook.com."
[Before]: IP / UDP / DNS Ans "b'star-mini.c10r.facebook.com."
no modification: b'b-www.facebook.com.'
[After]: IP / UDP / DNS Ans "b'star-mini.c10r.facebook.com."
[Before]: IP / UDP / DNS Ans "77.42.254.160"
no modification: b'instagram.fbey15-1.fna.fbcdn.net.'
[After]: IP / UDP / DNS Ans "77.42.254.160"
[Before]: IP / UDP / DNS Ans "b'mqtt.c10r.facebook.com."
no modification: b'edge-mqtt.facebook.com.'
[After]: IP / UDP / DNS Ans "b'mqtt.c10r.facebook.com."
[Before]: IP / UDP / DNS Ans "b'instagram.c10r.facebook.com."
no modification: b'graph.instagram.com.'
[After]: IP / UDP / DNS Ans "b'instagram.c10r.facebook.com."
[Before]: IP / UDP / DNS Ans "216.239.36.127"
no modification: b'lb-touch.rcs.telephony.goog.'
[After]: IP / UDP / DNS Ans "216.239.36.127"
[Before]: IP / UDP / DNS Ans "b'instagram.c10r.facebook.com."
no modification: b'graphql.instagram.com.'
[After]: IP / UDP / DNS Ans "b'instagram.c10r.facebook.com."
[Before]: IP / UDP / DNS Ans "b'star.facebook.com."
no modification: b'web.facebook.com.'
[After]: IP / UDP / DNS Ans "b'star.facebook.com."
[Before]: IP / UDP / DNS Ans "b'scontent.whatsapp.net."
no modification: b'dit.whatsapp.net.'
[After]: IP / UDP / DNS Ans "b'scontent.whatsapp.net."

```

As we can see we will be able to sniff DNS replies and monitor changed DNS reply messages.

## Other Commands:

clear: to clear active terminal

exit: to exit from the interactive shell

banner: to display some ASCII and ANSI arts

## Links:

<https://github.com/Hassan-Al-Achek/Method.git>

<https://github.com/Hassan-Al-Achek/netdeath.git>