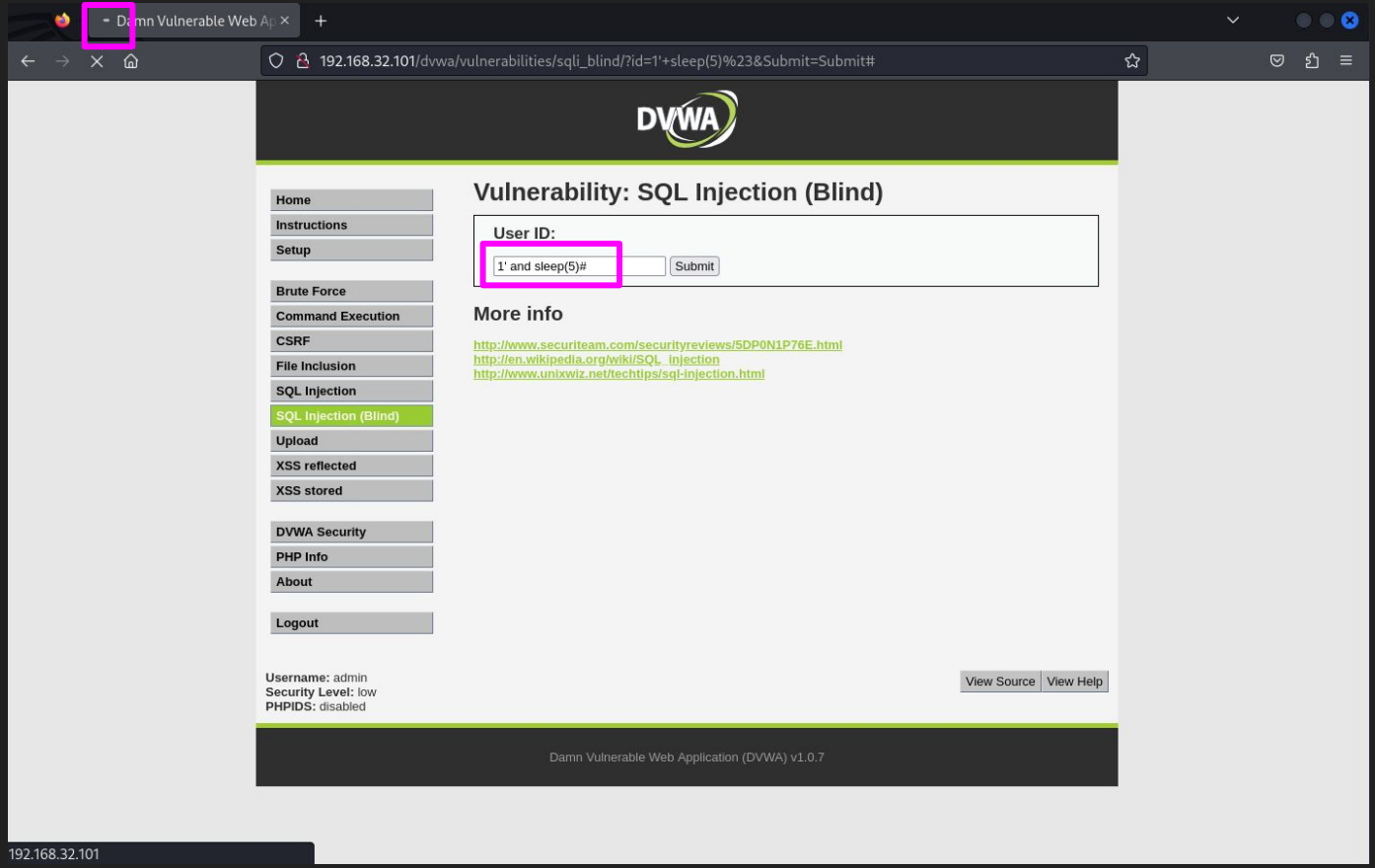
BLIND SQL INJECTION

STORED CROSS-SITE-SCRIPTING ATTACK

Oggi andremo a fare blind SQLI per trovare delle password in una database, e andremo anche a lanciare un attacco XSS persistente per rubari i cookie di sessione.

La differenza tra fare **SQLI In-Band** (non blind) e fare **SQLI Blind** é che SQLI In-Band ci ritorna un messagio di errore di sinitasssi, confermando che c'é una vulnerabilitá da sfruttare, invece, SQLI blind semplicemente ricarica la pagina senza confermare nulla. Una delle 2 maniere di confermare se possiamo realizzare SQL su una pagina é provando a dire di fare il ***time sleep***. Il time sleep é comando SQL che fermará il traffico client-server per una quantità di tempo specificata.

Qui possiamo provare a dire di fare il *time sleep* per 5 secondi.Infatti, la pagina si ha messo 5 secondi per continuare la comunicazione.



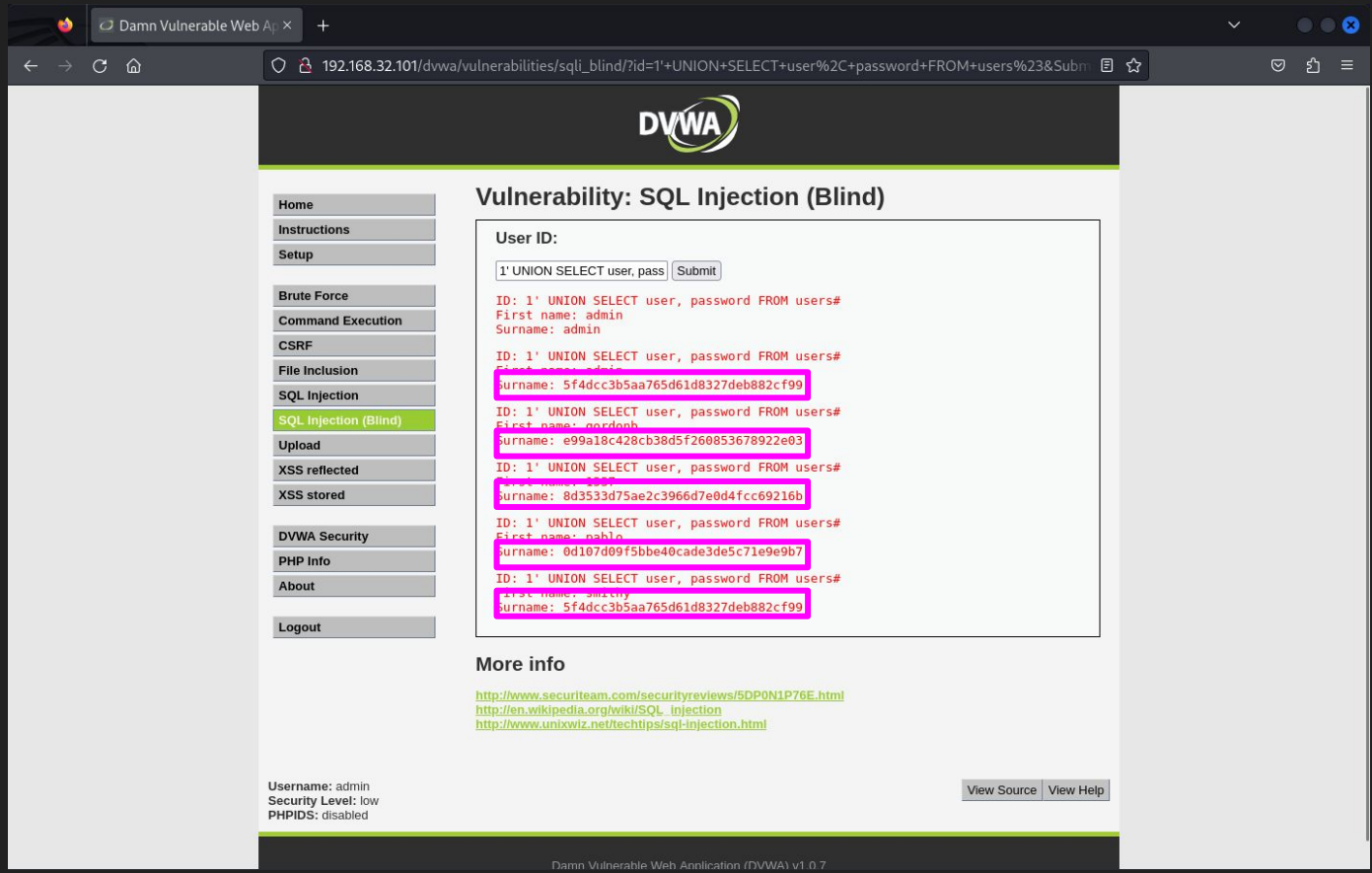
Con il comando:

1' UNION SELECT user, password FROM users#

Diciamo in sintesi:

"Entra con l'ID 1, poi mostrami gli user e i password delle tavole *users*"

Infatti, ci fa vedere i passwords hashatte, proviamo a decifrarle.



Utilizzando **John the Ripper** ho decriptato tutte le password comparando i hash ottenute a le password-hash dentro il file *rockyou.txt*

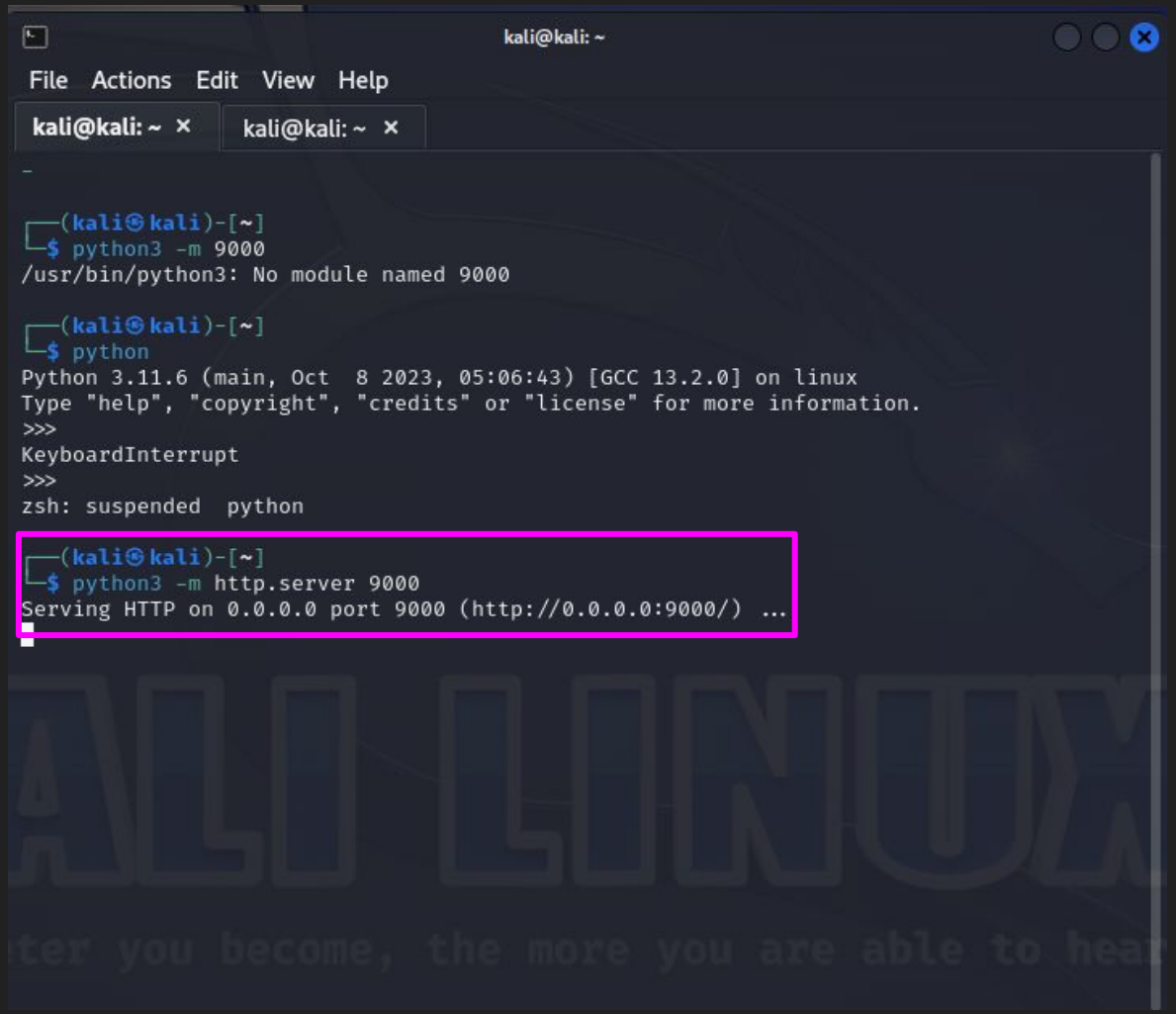
kali@kali: ~/Desktop

File Actions Edit View Help

```
┌──(kali㉿kali)-[~]
└─$ ping 192.168.32.101
PING 192.168.32.101 (192.168.32.101) 56(84) bytes of data.
64 bytes from 192.168.32.101: icmp_seq=1 ttl=64 time=0.863 ms
64 bytes from 192.168.32.101: icmp_seq=2 ttl=64 time=0.233 ms
^C
── 192.168.32.101 ping statistics ──
2 packets transmitted, 2 received, 0% packet loss, time 1018ms
rtt min/avg/max/mdev = 0.233/0.548/0.863/0.315 ms

┌──(kali㉿kali)-[~]
└─$ john --format=raw-md5 --show=/home/kali/Desktop/rockyou.txt DVWAshadow.txt
Invalid option in --show switch. Valid options:
--show, --show=left, --show=formats, --show=types, --show=invalid

┌──(kali㉿kali)-[~]
└─$ john --format=raw-md5 --show/home/kali/Desktop/rockyou.txt DVWAshadow.txt
Unknown option: "--show/home/kali/Desktop/rockyou.txt"

┌──(kali㉿kali)-[~]
└─$ john --format=raw-md5 --show /home/kali/Desktop/rockyou.txt DVWAshadow.txt
Warning: invalid UTF-8 seen reading /home/kali/Desktop/rockyou.txt
stat: DVWAshadow.txt: No such file or directory

┌──(kali㉿kali)-[~]
└─$ cd Desktop

┌──(kali㉿kali)-[~/Desktop]
└─$ john --format=raw-md5 --show /home/kali/Desktop/rockyou.txt DVWAshadow.txt
Warning: invalid UTF-8 seen reading /home/kali/Desktop/rockyou.txt
admin:password
gordondb:abc123
1337:charley
smithy:password

4 password hashes cracked, 52 left

┌──(kali㉿kali)-[~/Desktop]
└─$
```

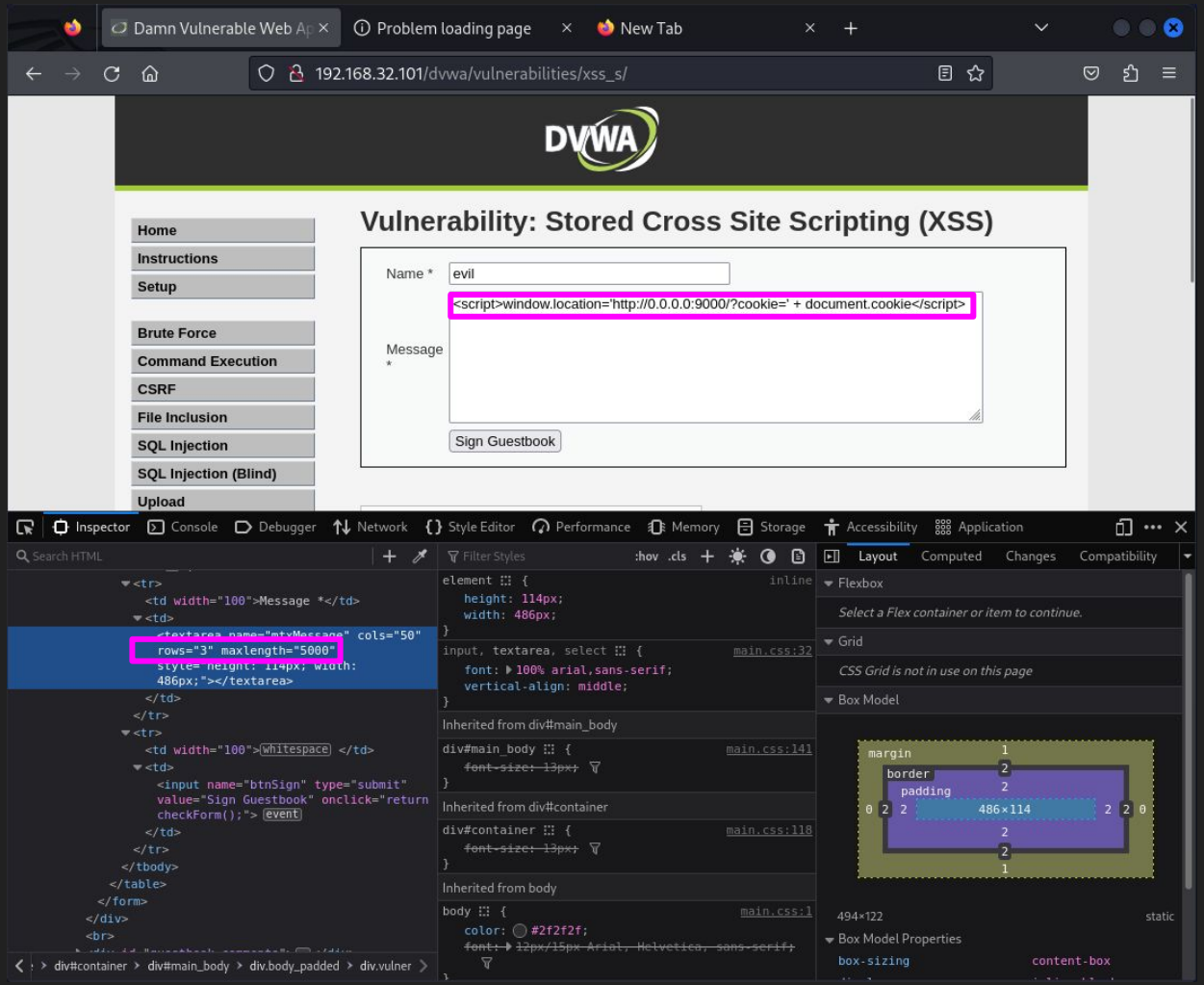
Ora per l'attacco XSS. Andremmo ad inserire codice dentro il web server di DVWA, che ruberá i cookie di sessione di ogni utente che arriverà al link specificato, e poi invierà detti cookie al nostro server in ascolto.

Iniziamo un server http in ascolto porta 9000 con python3



Siamo andati sulla sezio XSS stored, dove possiamo caricare il nostro script alla database di DVWA. Con lo script nello schermo possiamo prendere i cookie di ogni utente che visite questo directory, e poi quelli cookie verranno inviati al nostro server.

Purtroppo per default possiamo inserire un messagio di massimo 50 caratteri, ma ispezionando la pagina, possiamo andare a modificare questo limite e mettere che si possano inserire 5000 caratteri come massimo, che sarà sufficiente per nostro script.



Quello che si vede nel browser é il risultato di aver cliccato nel hyperlink per andare sul directory dove si trova nostro codice dentro il database.

Nel terminale si puó vedere come una volta la vittima ha fatto click, sua session ID é stata rubata e inviata al nostro server.