International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

A Pioneering of Real Time Penetration Testing With Worm Propagation Model

R.Teja[1], Dr. V. Cyril Raj[2]

DR.MGR University, Maduravoyal, Chennai, Tamil Nadu, India.

Abstract: ***The Internet utilization has been so prominent in the recent times and the concepts of network attacks are now the biggest threats to the network security researchers. The capability of attackers to rapidly gain control of vast numbers of Internet hosts poses an immense risk to the overall security of the Internet by self-propagating suspicious codes known as computer worms spread themselves without any human interaction and launch the most destructive attacks against computer networks. At the meantime, being fully automated makes their behavior redundant and calculable and computers connected to the Internet are very prone to targeted virus attacks and may end up crashing.So there is a need for an accurate penetration testing and worm propagation model for paramount desire of the internet.Hence, we propose a model for testing the vulnerability in the network and exploiting the machine as for useful applications.***

Keywords: Vulnerability, Threat, Exploit, Attack, Exploit, Security, Payload.

1. Introduction

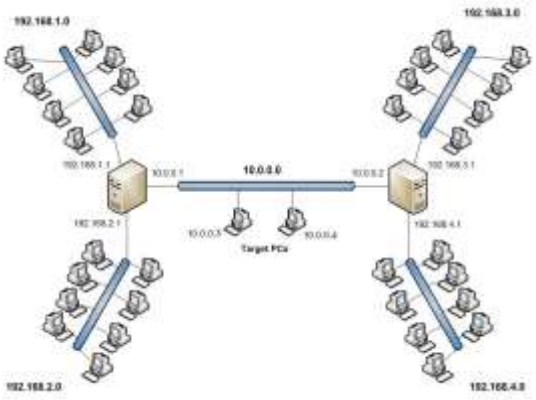
1.1 Penetration Testing

Penetration testing was among the first activities performed when security concerns were raised many years ago. The basic process used in penetration testing is simple attempt to compromise the security of the mechanism undergoing the test. In earlier years, computer networked operating systems, with their access control mechanism, were the most suitable components for penetration testing, because OS is the core component of a machine and so more exposed to security threats.

From a normal standalone system to a sophisticated corporate network requires security evaluation. The entire evaluation process is carried through methodology called as penetration testing. In general Penetration testing is basically a methodology of evaluating the security of the target by simulating attack by safely attempting to exploit system flaws, including OS, service and application threats, default configurations, and even risky end-user behavior. Such valuations are also useful in validating the efficacy of defensive system, as well as end-users adherence to security policies.

Tests are typically performed using manual or automated technologies to systematically compromise the security of servers, endpoints, web applications, wireless networks and other potential points of exposure. Once threats or flaws have been successfully exploited on a particular system. Pentesters will use the compromised system to launch related exploits and payloads at other internal resources, expressly by trying to incrementally achieve higher levels of security clearance and deeper access to electronic assets and information via privilege escalation.

2. Virtual Lab Configuration for Pentest



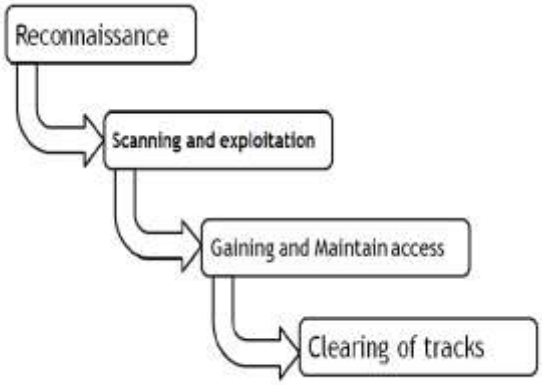
The steps will depending upon the operating system and the virtualization software we are using.

1. Open the Network Editor.
2. Add a network to your virtual network.
3. Change the network configuration to Host Only.
4. Choose the subnet for the network (e.g., 192.168.187.0). The subnet must be within a private range.
5. Save the network.
6. Assign this virtual network to machines as you create them.

Once we set up our virtual network, we can set up the network individually for each virtual machine. And has to implement the process.

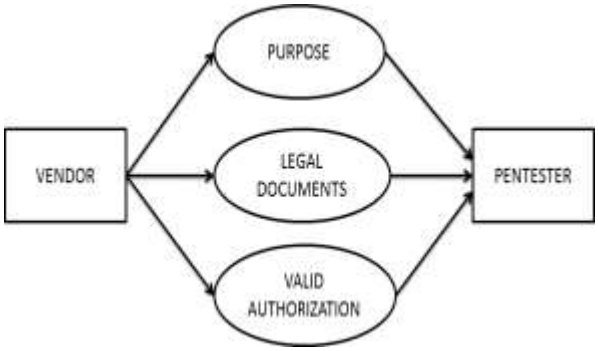
3. Implementation of Pen testing

- Reconnaissance
- Scanning and exploitation
- Gaining and maintain of access
- Clearing Tracks



3.1 Reconnaissance

It's the initial phase of pentesting where the pentester needs to gain the information about the target. The information may include regarding target operating system, list of services running, view of network.



3.2 Scanning and Exploitation

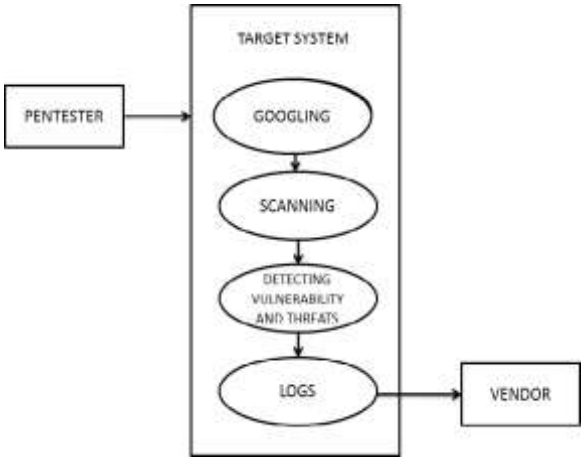
This is the crucial stage of pentesting. In this phase pentester try to gather the information regarding the vulnerabilities existed in the target network. Pentester mainly uses port scanning techniques to figure out vulnerabilities existed in the target network.After scanning the network pentester performs exploitation against the target based on the vulnerabilities existed.

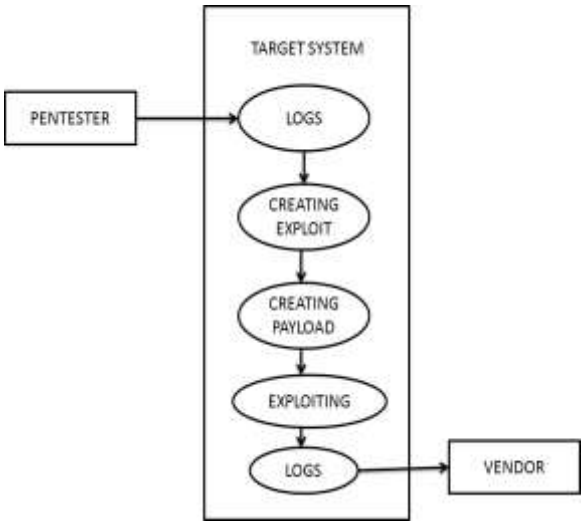
3.2.1 Exploits

To take dominance of vulnerability, we need an exploit, a small and highly functional computer program whose only reason of being is to take advantage of a particular vulnerability and to provide access to a system. Exploits often drop a payload to the target system to grant the attacker access to the system.

3.2.2 Payloads

A payload is the hunk of software that lets you control a computer system afterit's been exploited. The payload is typically attached to and delivered by the exploit.Just project an exploit that carries the payload in its knapsack when it breaks intothe system and then leaves the knapsack there.





3.2.3 Usage of Metasploit Framework

Consider the MSF to be one of the single most useful auditing tools freely available to security professionals today. From a vast array of commercial grade exploits and an extensive exploit, all the way to network information gathering tools and web vulnerability plug-in. The Metasploit Framework provides a truly impressive work environment. The Metasploit Framework is far more than just a collection of exploits; it's a framework that you can build upon and utilize for your custom needs. This grants you to concentrate on your unique environment, and not have to reinvent the wheel. Currently, metasploit requires you to setup and configure postgresql on your system to work.

The basic steps for exploiting a system using the Framework include:

1) Choosing and configuring an exploit (code that enters a target system by taking advantage of one of its bugs; about 900 different exploits for Windows, Unix/Linux and Mac OS X systems are included);
2) Optionally checking whether the intended target system is susceptible to the chosen exploit;
3) Choosing and configuring a payload (code that will be executed on the target system upon successful entry; for instance, a remote shell or a VNC server);

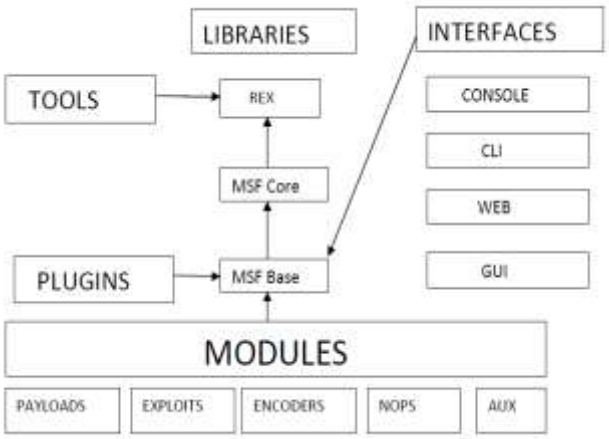
4) Choosing the encoding technique so that the intrusion-prevention system (IPS) ignores the encoded payload;
5) Executing the exploit.

This modular approach – allowing the combination of any exploit with any payload is the major advantage of the Framework.
It facilitates the tasks of attackers, exploit writers and payload writers. Metasploit runs on UNIX (including Linux and Mac OS X) and on Windows. The Metasploit Framework can be extended to use add-ons in multiple languages.

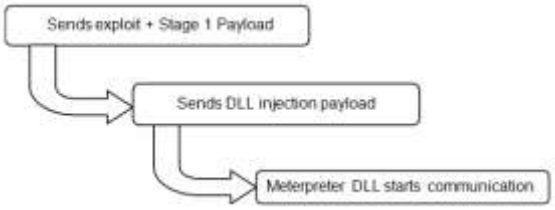
3.2.3.1 Meterpreter
Meterpreter, short for The Meta-Interpreter is an advanced payload that is included in the MSF (Metasploit Framework). Its function is to provide complex and advanced features that would otherwise be tedious to implement purely in assembly. The way that it manage this is by allowing developers to write their own extensions in the form of shared object (DLL) files that can be uploaded and injected into a running process on a target computer after exploitation has resulted. Meterpreter and all of the extensions that it loads are executed entirely from memory and never touch the disk, thus grant them to execute under the radar of standard Anti-Virus detection.

When exploiting software vulnerability there are certain results that are typically expected by an attacker. The most familiar of these expectations is that the attacker be given access to a command interpreter, such as /bin/sh or cmd.exe which allows them to execute commands on the remote machine with the privileges of the user that is running the vulnerable software. Access to the command interpreter on the target machine gives the attacker nearly full control of the machine bounded only by the privileges of the exploited.

Lastly, the command interpreter is limited to the set of commands that it has access to, both internal and external. The set of external commands that may or may not exist on a machine leads to issues with automation and presents problems with flexibility, not to mention being tied to one explicit platform or command interpreter in most cases. These three issues illustrate some of the down-sides to relying on a native command interpreter and come to form the primary reasons for implementing meterpreter.



3.2.3.1.1 Working of Meterpreter



Meterpreter is capable of avoiding these three issues due to the way it has been implemented. Firstly, meterpreter is able to avoid the creation of a new process because it executes in the context of the process that is exploited. Furthermore, the meterpreter extensions, and the meterpreter server itself, are all executed entirely from memory using the technique described in Remote Library Injection. The fact that meterpreter runs in the context of the exploited process also allows it to avoid issues with chroot because it does not have to create a new process. In some cases the application being exploited can even continue to run after meterpreter has been injected. Finally, and perhaps the best feature of all, meterpreter allows for incredible control and automation when it comes to writing extensions. Server extensions can be written in any language that can have code distributed as a shared object (DLL) form.

This fact makes it no longer necessary to implement specially purposed position independent code in what typically requires a low-level language such as assembly.Aside from solving these three issues, meterpreter also provides a default set of commands to illustrate some of the capabilities of the extension system. For instance,one of the extensions, Fs, allows for uploading and downloading files to and from there mote machine. Another extension, Net, allows for dynamically creating port

Forwards that are similar to SSH's in that the port is forwarded locally on the client's machine, through the established meterpreter connection, to a host on the server's *Sends exploit + Stage 1 Payload, Sends DLL injection payload, Meterpreter DLL starts communication network*. This enables the reaching hosts on the inside of the server's network that might not be directly reachable from the client. Under the hood the port forwarding feature is simply built on top of a generic channel system that allows for funneling arbitrary segregated data between the client and the server as if it were a tunnel of its own. Finally, the contents of meterpreters packets can be encrypted with a customcipher.

4. Stages of Implementation

In this stage the exploitation simulation has been explained by which stages the vulnerability has been find and patched. Here a OS will be go through these stages of implementation then only a perfect vulnerability of the system can be detected. In first stage the host will not be patched, current antivirus update and firewall by this exploitation can be happen and vulnerability can be detected and exploit with the help of meterpreter. In second state the firewall will be enabled and no antivirus update, no patches by that exploit will be detected. Simultaneously the process will be follow by four stages and new vulnerability will be detected at the end of pentesting. The new vulnerability will be reported to

the client and ot will be patched this process will be on going till the new vulnerability was s found in host/network.

Level	*Patched*	*Antivirus*	*Firewall*
Stage 1	No	No	No
Stage 2	No	No	Yes
Stage 3	No	Yes	Yes
Stage 4	Yes	Yes	Yes

5. Conclusion

This paper states about the penetration testing with worms (Exploits) propagation with different stages of exploitation. Information security world faces lots of issues for creating patches to the current vulnerability it's not possible to make our network/host called *Zero-Day Attack*whenever attack has happened mitigation steps has to follow in real time world a attack has happen a propagation model needed to do the mitigation technique for the attack by understanding the attack. By this paper it is possible to find the vulnerability of the system, simulating the attack by that vulnerability, exploiting the vulnerability of the system and at last creating patches to that specific vulnerability. By this method a new vulnerability also founded and for that also a needed upgrade and patched can be created.

References

[1] W.O.Kermack&A.G.McKendrick(1927),'A Contribution to theMathematical Theory of Epidemics', Papers of a mathematical and physicalcharacter,Volune 115,Issue 772,pg 700-721.
[2] S. Staniford, V. Paxson, & N. Weaver (2002), 'How to Own the Internet inYour Spare Time' presented at Security '02, San Francisco.
[3] Cliff Changchun ,Zou,Weibo& Gong, Don Towsley(2002), 'Code Red Worm Propagation Modelling and Analysis', ACM.
[4] Cliff Changchun, Zou,Weibo Gong, Don Towsley&LixinGao(2003),'Monitoring and Early Warning for Internet Worms', ACM.
[5] Cliff Changchun, Zou, Weibo Gong & Don Towsley (2003),'WormPropagation Modelling and Analysis under Dynamic Quarantine Defence,ACM
[6] Zesheng Chen, LixinGao& Kevin Kwiat (2003),'Modelling the Spread ofActive Worms', IEEE INFOCOM.
[7] C.Onwubiko,A.P.Lenaghan,L.Hebbes(2005),'An Improved Worm MitigationModel for Evaluating the Spread of Aggressive NetworkWorms',EUROCON,November 22-24.
[8] Sarah H.Sellke, Ness B.Shroff&SaurabhBagchi (2008),'Modeling andautomated containment of worms', IEEE transactions on dependable andsecure computing, vol 5, No.2.
[9] HuaYua& ,Guoqing Chen(2008),'Network virus-epidemic model with thepoint-to-group information propagation', Elsevier, pg 357-367.
[10] Cong Jin, Jun Liu, Qinghua Deng (2009),'Network virus propagation modelbased on effects of removing time and user vigilance', International Journal ofNetwork Security, Vol.9, pg 156-163.