An Analysis of the IDS Penetration Tool: Metasploit

Carlos Joshua Marquez
Department of Technology Systems
East Carolina University
Greenville, NC, U.S.A.
cmarquez09@students.ecu.edu

Abstract: **This paper will cover the basics of one of today's most influential tools in the information security field. First we will define exactly what the Metasploit Framework is, followed by a brief description of the program's terminology including; exploits, auxiliary modules, payloads, and encoders. This will be supervened by a few brief examples of effective uses of the product. Lastly, we will cover methods to circumvent the attacks coming from the Metasploit Framework. Overall, the aim of this paper is to provide a thorough understanding of what Metasploit is and how to utilize it as a security professional.**

Keywords – intrusion detection; penetration testing; network testing.

INTRODUCTION

Since the late nineties, the Internet has grown at an exponential rate. One of the biggest spurts in growth came between the years of 1995-2000 with the dot-com bubble that prompted the spawn of e-commerce for virtually every facet of society. The success of the Internet has brought great change to the world as we know it; however, not all of this growth has been productive. With thousands of sites launching daily and limited resources available to monitor the credibility and/or security of these sites the existence of vulnerabilities was inevitable.

Eventually exploits became rampant causing the information security field to step up its game. The result was the pre-emptive existence of vulnerability testers that's sole purpose was to attempt to exploit such software far before others got the opportunity. One of these researchers was a man by the name of H.D. Moore who in the summer of 2003 founded the Metasploit Project [1].

H.D.'s purpose was to create a penetration testing tool that could be easily utilized by even novice users to perform penetration testing, regression testing, patch verification, and development. From this arose the project known formerly today as the Metasploit Framework. As with any other security tool, this software is often described with the proverbial double-edged sword as in the wrong hands it can be used for great harm.

DESCRIPTION

1.1 What it is?

The Metasploit Framework is a tool that collectively combines exploits into one central location ideally for security researchers. Originally developed using the Perl scripting language Metasploit is now currently on its third reincarnation. Version 1.0 was written solely by H.D. Moore using Perl sporting a curses based front-end. Version 2.0, also written in Perl, and included the help of a few additional developers. For Version 3.0, Metasploit received a complete overhaul. Written in the powerful scripting language Ruby, Metasploit 3.0 now boasts the power of automation due to the nature of Ruby's status as an object-oriented language. Additionally, Metasploit is considered multi-platform running on most variations of Unix and Windows [1].

1.2 Who is it for?

The Metasploit Framework was developed with the intentions of making security experts' lives easier. The original primary users were considered to be network security professionals, security administrators, product vendors, and other like-minded security researches. Each would use the tool within the guidelines of their own discipline; network security professionals for penetration testing, security administrators for patch installation verification, product vendors for regression testing, and other security researchers for perhaps development of other exploits [1].

This document was created using SOLID CONVERTER PDF
To remove this message, purchase the product at www.SolidPDF.com

1.3 Terminology

The Metasploit Framework can be a bit confusing for novice users as the Linux distribution offers no Graphic User-Interface (GUI). Therefore learning the semantics and syntax of the framework is essential to the effective use of the software.

The primary outline of the majority of attacks in Metasploit revolves around the following foundation;

1. Choosing an exploit and configuring it.
2. Checking for susceptibility.
3. Choosing a payload and configuring it.
4. Choosing an encoding system.
5. Executing the exploit.

One of the key components of Metasploit is in fact the exploits. At the time of this paper's writing Metasploit contains 613 exploits, 306 auxiliary modules, 215 payloads, and 20 encoders. Exploits target specific Operating Systems, applications, and/or services [2].

Additionally, auxiliary modules exist within the context of Metasploit and can be declared just as easily as exploits. In Metasploit an auxiliary module is defined as a module without a payload. Auxiliary modules serve as accessories to the Metasploit Framework and can be used in a variety of facets to expand upon the capabilities of the program. Just a few of their features include making Metasploit act as a vulnerability scanner, port scanner, HTTP client, FTP client, SMB client, and etc [1].

Next there are the various payloads that exist for each exploit. After an exploit is initiated and the remote target or targets are selected a payload must be selected to be executed after the breach. The payloads of Metasploit are Operating System specific, though generic payloads do exist [1].

Additionally, payloads in Metasploit come in three main variations; singles, stagers, and stages [3]. Singles are independent of any other and are generally simple command-line executions such as adding a user to a target system. Stagers establish a reliable network connection while remaining small enough to deliver the stage. Because sometimes acquiring both a reliable connection and limited overhead can be difficult, several variations in stagers exist. Stages are the final piece of the puzzle of stagers as they provide the components for the stager to deploy. Table 1 shows the differences between the three in syntax form.

Payload Type	**Example**
Single	*windows/shell_bind_tcp*
Stager	windows/shell/*bind_tcp*
Stage	windows/*shell*/bind_tcp

Table 1. Showing Payload Types

Here you can see that the single payload type includes the whole process of 'shell_bind_tcp,' while the later two differ in path. The stager in this case is the 'bind_tcp' module while the 'shell' acts as the stage by which the stager can download its resources [3].

Overall, payloads serve as post-exploitation commands that can be as simple as adding a user to the targeted system or binding a command shell to a designated port. Additional options include; VNC injection, remote shell execution, meterpreter execution, and backdoor installations [1].

Lastly, in order to provide another layer of promiscuity payload encoders may be added to the exploit to ensure the connection between the attacker and victim remains encrypted. These work in much the same way as hashes by encapsulating the content of the payload with a predetermined key obfuscating it from detection.

USAGE

Now that we understand the dictation and origins of Metasploit we can move on to the applications of the product. We will begin with a brief synopsis of a few common attacks followed by some distinguishing characteristics of the Metasploit Framework. While Metasploit is primarily identified as a one-stop exploitation application its sole purpose revolves not just around the exploitation of remote systems, but also around the development of new ones as well. With the 3.0 iteration of Metasploit, near complete automation is possible as scanning, fingerprinting, identifying vulnerabilities, exploiting, and reporting can be configured with some degree of work.

2.1 Example Attacks

It is without a doubt that the field of computer security is an on-going battlefield where the victor is solely based on time. On the second Tuesday of

This document was created using SOLID CONVERTER PDF
To remove this message, purchase the product at www.SolidPDF.com

every month Microsoft releases patches for its many applications and operating system variations to what has affectionately become known in the information security field as "Patch Tuesday." Upon their release, black hatters of all likes begin the reverse engineering process to discover the original vulnerability. Toping the list of commonly exploited programs include third-party applications such as Adobe Flash, Adobe Reader, JavaScript, and QuickTime. For the direct purpose of this paper though we'll address exploits pertaining to apache web server and oracle.

Considering one of the most commonly attacked sources are web servers it seems scary to think that Metasploit houses an exploit that within eight commands can compromise an Apache Web Server [4]. The attack we're referring to utilizes chunked encoding to specially craft an invalid request on the server causing at the bare minimum a Denial-of-Service attack; though with some OSes remote code execution is possible. This is instigated by a stack overflow that is controlled on 64-bit OSes where return addresses are stored on the stack heap. In the collaborative experiment of Rajani, Mohamed, and Stansbury, the results indicated a successful breach with remote code execution being successful in the form of adding users with full permissions and writing files to the root directory on the web server [4]. In the days of prevalent E-commerce some are considering these additions to the Metasploit program as haphazard.

Another big target for exploits reside in database servers that house large amounts of data ranging from social security numbers to financial data. One of the leaders in database management is Oracle with an approximate market share of 40 percent [5]. The majority of databases use the same language, Structured Query Language (SQL), thus often exploits targeting databases are based off the use of this language. Because of this no free penetration testing software currently offers an independent direct exploit to the system [6].

As we mentioned previously, a key feature of the Metasploit Framework is development. At the Black Hat USA Conference in 2009 Chris Gates and Mario Ceballos, presented a method for exploiting Oracle through SQL injection techniques utilizing custom built auxiliary modules [6]. Their attacks consisted of seven steps that make up what they considered the basis of pen testing;

1. Locate a system running Oracle.
2. Determine Oracle Version.
3. Determine Oracle SID.
4. Guess/Bruteforce Username/Password.
5. Privilege Escalation via SQL Injection.
6. Manipulate Data/Post Exploitation
7. Cover Tracks

To do this a separate auxiliary module was required for each step. To locate a system the inclusive NMap was used to direct a port scan searching for commonly used Oracle ports, 1521-1540. A homemade TNS mix-in was added to the Metasploit trunk allowing it to craft TNS packets to determine the Oracle version. In order to guess the Oracle SID a SID enumerator was used as subsequent to version 9.2.0.8 Oracle no longer freely gives out this information. Bruteforcing the username/password combination was done by using the pre-existing auxiliary module for Bruteforce logins using a dictionary list by Pete Finnigan. Privilege escalation of the username gathered in step four was accomplished with a SQL injection vulnerability in the DBMS_EXPORT_EXTENSION package. For post exploitation the win32exec module was used to execute a remote command on the machine to create a user on the system for future access [6].

2.2 Meterpreter

One of the more powerful payloads discussed above, meterpreter, originally emerged in Metasploit version 2.2 [1]. What makes meterpreter so powerful is its elusiveness in being detected by even the most knowledgeable security analysts.

Meterpreter is an advanced dynamically integrated payload that resides entirely in-memory by injecting DLL stagers. Once a compromised system is discovered and exploited the meterpreter payload establishes a client side command interpreter with which to communicate [7]. This allows the attacker to remotely interact with the host system without having to establish separate connections. Under normal circumstances, once a system is exploited a single payload is delivered that is only able to execute one command and then it is done. This one command could be something as simple as adding a

This document was created using SOLID CONVERTER PDF
To remove this message, purchase the product at www.SolidPDF.com

user or opening a remote shell with which to communicate. In doing this a resulting cmd.exe process would be created in the process list with SYSTEM rights [8]. Immediately, this would raise red flags. However, with meterpreter DLL injection is used to upload the meterpreter process into the compromised processes' heap. Normally, an uploaded DLL would be written to the disc, yet meterpreter alters the way the Load Library utilizes core API calls redirecting them to the memory location of the meterpreter DLL [7].

The truly beautiful thing about meterpreter is its ability to remain undetectable by most commonly implemented host based IDSes. By embedding itself into preexisting processes and not altering system files on the hard-drive, the HIDS is never made any wiser. Not to mention, the process in which meterpreter hides can be changed at a whim so tracking it and stopping the process becomes rather difficult even to the trained eye.

2.3 Penetration Testing Automation

For years the dream of automating penetration testing, often abbreviated as pentesting, has been considered just that, a dream. The challenges facing post-exploitation automation include; visible processes, file transfer capabilities, and exploit expiration [8].

The problem of readily visible processes with suspicious user rights was previously discussed in section 2.1. The only way around executing a separate visible process would be to install a root kit or backdoor, though this requires the transfer and installation of additional malware. This brings up the capability of a payload to transfer/install files to the remote system. To do this requires an advanced payload that will most likely be compromised in its writing of data to the remote system. Lastly, exploit expiration refers to the acknowledgement that some exploits can only be run once. Without separate sessions to enlist multiple tasks the process can become time consuming if not impossible. This would require the use of another exploit to complete other tasks [8].

According to Irani and Weippl's research on post-exploitation automation, meterpreter in conjunction with commonly installed scripting languages provides just the right tools for a solution to these problems [8]. Meterpreter's innate ability to remain hidden in current processes through DLL injection allows a method around the visible process problem. Additionally, because the process is not blatantly listed the use of a root kit or backdoor is not necessary, thereby, voiding the need for file uploading and the risk of running into anti-virus scans. Though, on that note, an analysis conducted by Mark Baggett found that only 3 out of 32 reputable antivirus programs interpreted a stand-alone meterpreter payload as a security risk [9]. Lastly, meterpreter also provides users with the ability to open independent sessions allowing for efficient multitasking.

With the help of meterpreter post-exploitation scripts can be ran with the capability of not only further exploiting the current system, but previously non-exploitable machines as well. This technique of using an exploited machine to exploit a previously safe-guarded machine is known as pivoting [8]. Not to go into too much detail, but the port forwarding service of meterpreter provides this capability allowing for a connection back to the initial attacker. The point is that the automation of post-exploitation tasks extends far beyond just automating the initial system. With the right script automation can be implemented to scan an entire network for vulnerabilities.

EVASION

With a tool like the Metasploit Framework freely available for anyone to use it's extremely important to keep up-to-date on the advancements in securing data as even a novice user can pick up this tool and within hours become dangerous threat. For this reason, it seems imperative that administrators implement every security measure feasible to protect their network. Here we'll cover a few pieces of software that will help in thwarting the threats of this product.

3.1 Antivirus

According to a study conducted in 2008 by Mark Baggett of the SANS Institute, "today's antivirus products are all but completely ineffective in detecting Metasploit payloads" [9]. Baggett's technique involved extracting Metasploit payloads to

This document was created using SOLID CONVERTER PDF
To remove this message, purchase the product at www.SolidPDF.com

run them through the collective virus database housed at VirusTotal.com. The results provided suggested that without the use of encoders payloads were detection approximately 19% of the time, while with the use of encoders this percentage dropped to around 7% [9].

While the majority of antivirus distributions picked up little to nothing there was a common trend in the one's that did recognize a threat. F-Secure performed the best with a 72% detection rate, while notably Panda and Kaspersky did well against encoded and non-encoded attacks, respectively [9]. Overall, though some of the big names of the industry were quite often fooled.

Based on the research provided by Baggett it seems relatively safe to say that the answer for protection against Metasploit payloads doesn't reside in antivirus software. So where to look next?

3.2 ModSecurity (modsec)

ModSecurity is an open-source intrusion detection and prevention system that is designed for web servers. Just like any other IDS, Modsec, checks inbound requests against a set of preconfigured rules that are designated by the user. However, unlike most IDSes Modsec performs not as a separate application, but as a module of the web server. Additionally, requests are screened at the HTTP level rather than the TCP/IP level allowing for a much greater range of specifications to be tested at the application layer [4].

In research presented earlier in section 2.1 a vulnerability in Apache web server concerning chunked encoding was used to remotely run arbitrary code. In order to circumvent this attack, the George Mason University researchers employed the tactics of Modsec to specifically reject all incoming chunk encoded requests to the server. The result was a success in that all chunk encoded requests were dropped disallowing the attack to succeed [4].

CONCLUSION

The purpose of this paper was to provide the reader with an understanding of Metasploit such that he/she may use it himself. Metasploit is a powerful tool that like we said time and time again, in the wrong hands can be used for great harm. It provides an abundance of resources for legitimate network security professionals, security administrators, product vendors, and developers to use in a variety of ways. In fact, in its diversity lays the key to it success. However, the only real way to fully understand the intricate design of the Metasploit Framework is to use it. Hopefully, this paper helps others to understand the capabilities of Metasploit and utilize it as a tool for themselves.

RESOURCES

[1] Maynor, D., & Mookhey, K.K. (2007). *Metasploit toolkit for penetration testing, exploit development, and vulnerability assessment* [pp. 1-30]. Retrieved from http://books.google.com/books?id=bzZG5a1kEw4C&lpg=PA1&ots=36soArIcvd&dq=metasploit&lr&pg=PP1#v=onepage&q&f=false

[2] *The Metasploit Project*. (2010, October 20). Retrieved from http://www.metasploit.com/

[3] Aharoni, M., Coppola, W., & Kearns, D. (2010, October 15). *Metasploit unleashed*. Retrieved from http://www.offensive-security.com/metasploit-unleashed/

[4] Rajani, M.A., Mohamed, A., & Stansbury, H.C. (2006). *E-commerce security technologies: an evaluation using the metasploit framework (msf)*. Informally published manuscript, Computer Science, George Mason University, Fairfax, Virginia. Retrieved from http://www.qatar.cmu.edu/iliano/courses/06S-GMU-ISA767/project/papers/mohamed-rajani-stansbury.pdf

[5] Babcock, C. (2008, April 25). In database market, oracle gets bigger, others hang on. *InformationWeek*, Retrieved from http://www.informationweek.com/news/software/database_apps/showArticle.jhtml?articleID=207402230

[6] Gates, C., & Ceballos, M. (2009). Oracle penetration testing using the metasploit framework. *Proceedings of the BlackHat USA 2009*, http://www.blackhat.com/presentations/bh-usa-09/GATES/BHUSA09-Gates-OracleMetasploit-PAPER.pdf

[7] Silberman, P., & Davis, S. (2009). Metasploit autopsy: reconstructing the scene of the crime. *Proceedings of the BlackHat USA 2009*, http://www.blackhat.com/presentations/bh-usa-09/SILBERMAN/BHUSA09-Silberman-MetasploitAutopsy-PAPER.pdf

[8] Irani, M.T., & Weippl, E.R. (2009). Automation of post-exploitation. *Secure Business Austria*, Retrieved from http://www.sba-research.org/wp-

This document was created using SOLID CONVERTER PDF

To remove this message, purchase the product at www.SolidPDF.com

content/uploads/publications/TabatabaiIrani_AutomationOfPostExploitation_2009.pdf

[9] Baggett, M. (2008). Effectiveness of antivirus in detecting metasploit payloads. *SANS Institute InfoSec Reading Room*, Retrieved from http://www.sans.org/reading_room/whitepapers/casestudies/effectiveness-antivirus-detecting-metasploit-payloads_2134

This document was created using SOLID CONVERTER PDF

To remove this message, purchase the product at www.SolidPDF.com