

“INVESTIGATING THE PARALLELS BETWEEN USING A RAT-STYLE SOFTWARE FOR MALICIOUS PURPOSES AND VIRTUOUS INTENT”

By Daniel
Broomhead
27016005

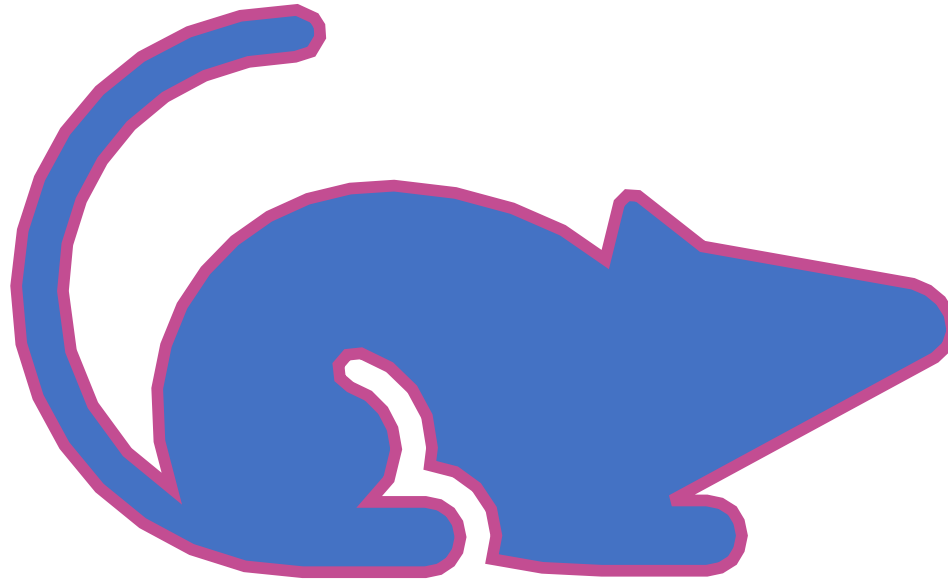
WHAT IS A RAT?

Remote Administration Tool:

Virtuous
Productive
Benevolent
Benignant

Remote Access Trojan:

Promiscuous
Malicious
Deceitful
Surreptitious



At The Heart of it, they are the same ideology and technology.

HOW ARE RATS USED?

- Remote Desktops
 - Administration
 - Computers
 - Remote access software
 - Usually part of a bigger system
 - Delivering a payload
-

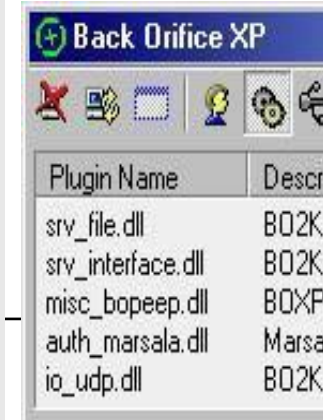
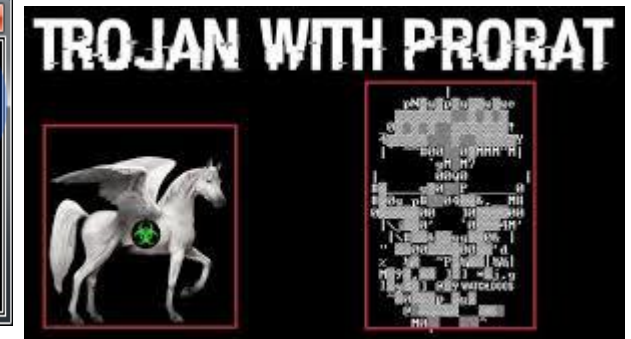
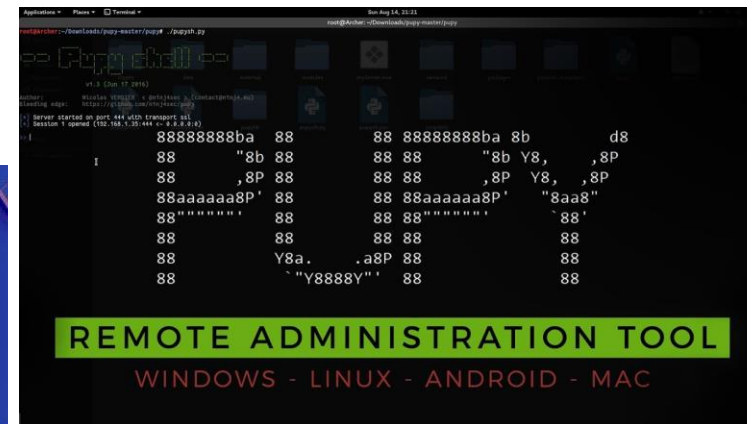
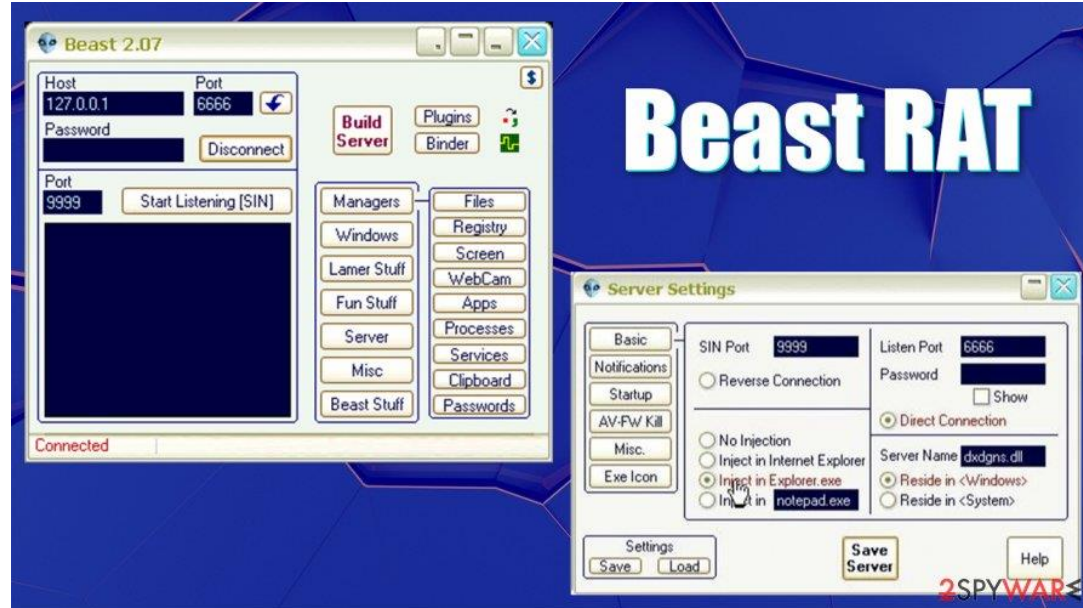
Backdoor, Remote Access Tool/Remote Access Trojan (RAT)

A backdoor is an application allowing remote access to a computer. The difference between this type of malware and a legitimate application with similar functionality is that the installation is done without the user's knowledge.

Typical backdoor functionality includes the capability to send files to the host computer and execute files and commands on it, and to exfiltrate (send) files and documents back to the attacker. Often this is coupled with key-logging and screen-grabbing functionality for purposes of spying and data theft.

The term "RAT" (Remote Access Tool) can be considered a synonym to "backdoor", but it usually signifies a full bundle including a client application meant for installation on the target system, and a server component that allows administration and control of the individual 'bots' or compromised systems.

EXAMPLES:



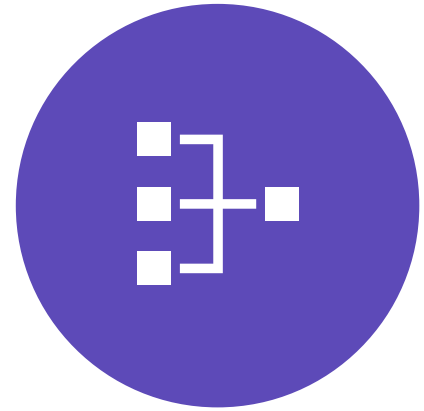
GENERAL CONSIDERATIONS FOR RATs:



LANGUAGE:



METHODOLOGY:



SYSTEM:

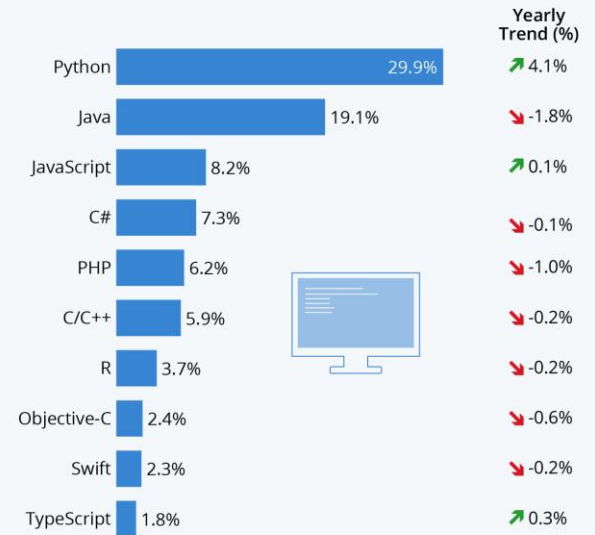
PYTHON 3:

- Python 2 is now deprecated.
- Cross-Compatibility :Compatible with all major Platforms and Systems
- Robust Standard Library
- Depth of open-source frameworks, Extended libraries, community support
- Effective in the IoT sphere
- Tried & Tested Scalability
- Python is a desired Language in the workplace – so this will be a good project to demo
- Gaining popularity faster than ever
- Fairly confident in this language but looking to develop skills further

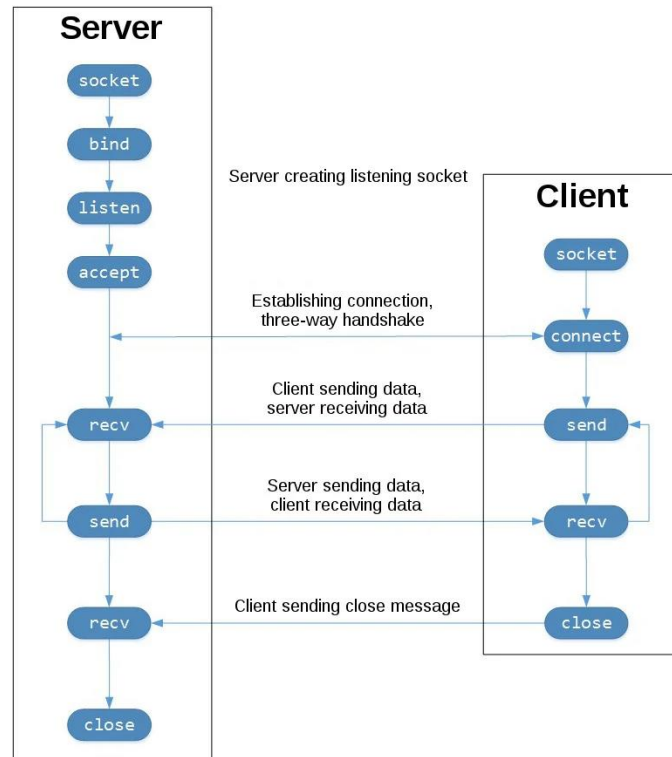
<https://www.statista.com/chart/21017/most-popular-programming-languages/>
<https://www.codingdojo.com/blog/top-7-programming-languages>

Python Remains Most Popular Programming Language

Popularity of each programming language based on share of tutorial searches in Google



MY RAT: THE CRUCIAL DECISIONS



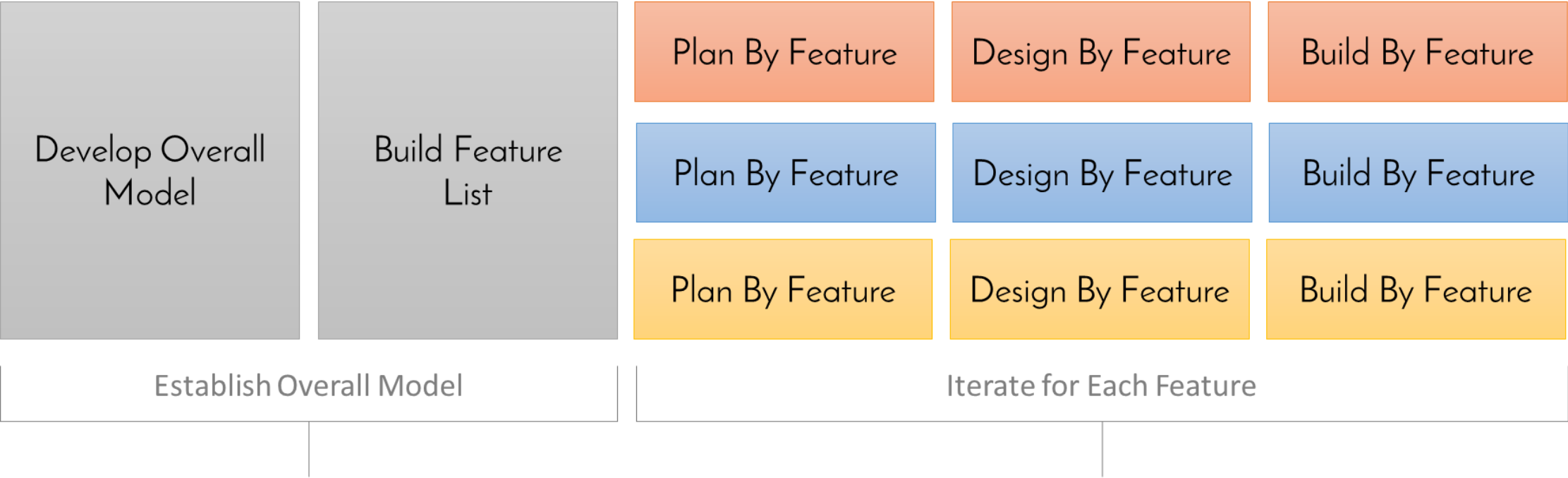
- Using the Python Socket module as an interface to the Berkeley Sockets API
- TCP Sockets
 - Reliable: Dropped Packets retransmitted – no data lost
 - In-order: Data is read in the order it was written
- UDP Does not share their properties and thus would not be appropriate.
- A reverse connection is also used meaning the client initiates the connection with the server.
- This prevents the firewall from blocking the connection as it sees the connection as being instantiated from within.

This has been developed on Windows 10
Mainly for use on Windows 10
Some of the features may work on Linux,
No testing has been conducted on MacOS

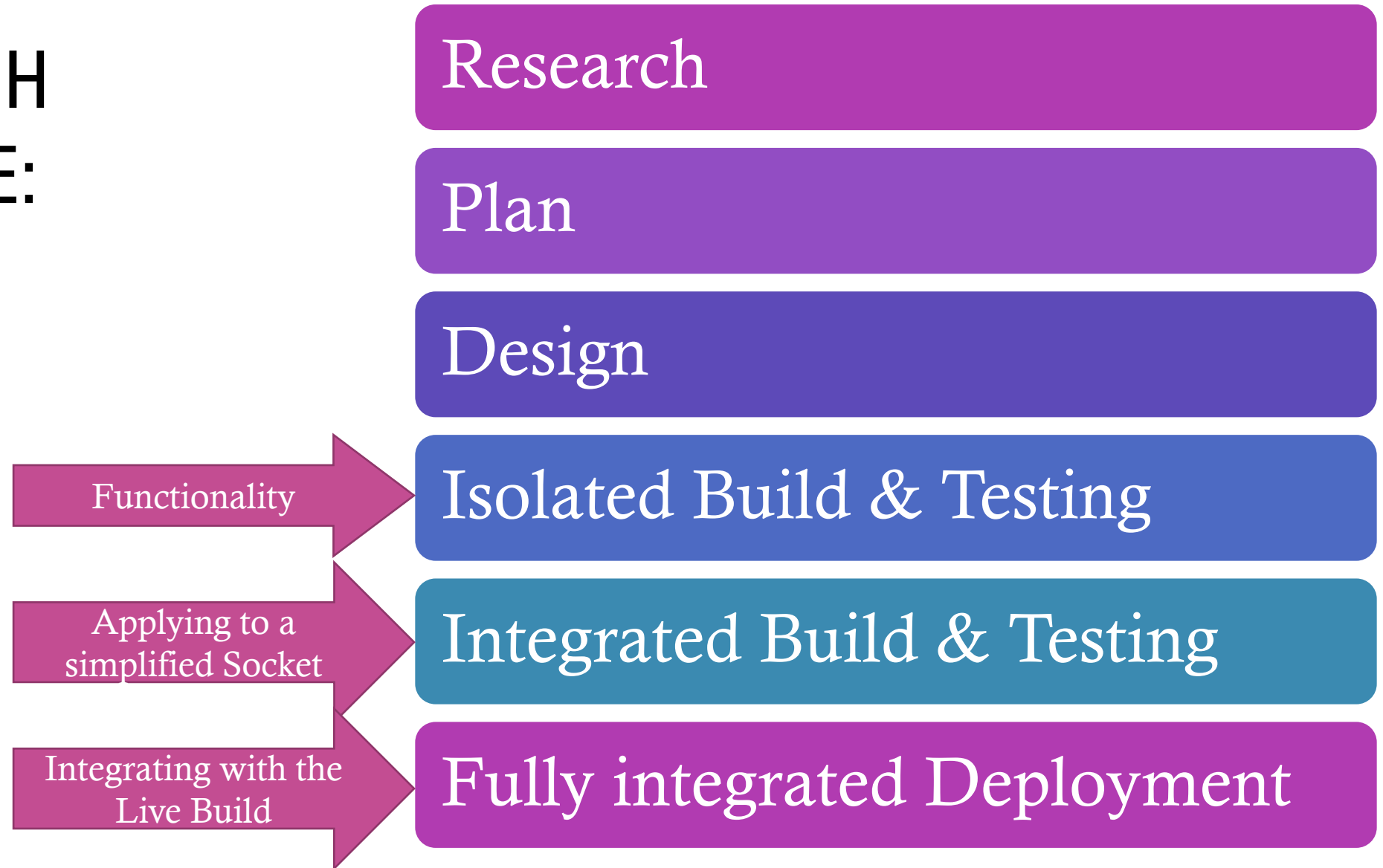
DEVELOPMENT IDEOLOGY

- The development of the functionality for the system followed a similar ideology to that of Feature driven development

Feature Driven Development (FDD) Approach



FOR EACH FEATURE:



CURRENT FUNCTIONALITY:

Send a console message

Send a custom messagebox Alert

Remote Shutdown (+ Remote Shutdown with custom message)

Remote Lock Device

Remote Restart

Play Starwars Via Telnet Connection

Play Chess Via Telnet Connection

Check the Weather Via Telnet Connection

Enable Telnet Client

Start Keylogger →
Stop Keylogger

Retrieve Keylogs

Retrieve Clipboard Contents

Send a File

Retrieve a File / Folder

*Obtain information
about the target
system*

Execute a file

Screenshot

Series of screenshots
→ Video

Reverse shell

Send an email

Auto emailer – start /stop

Record Webcam

Playback recording

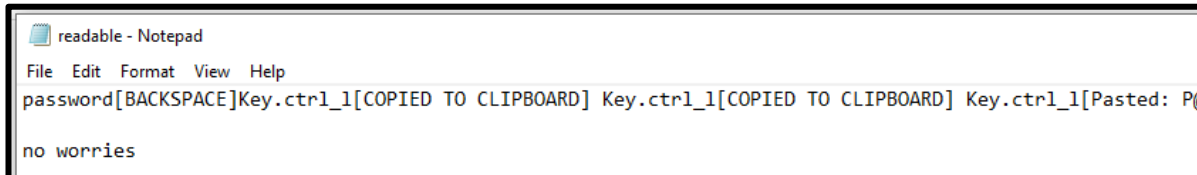
```
self.Switcher = {
    "-msgbox": self.systemmsg,
    "-msg": self.sendMessage,
    "-shutdown": self.shutdown,
    "-shutdownM": self.shutdownmessage,
    "-lock": self.locksystem,
    "-restart": self.restartsystem,
    "-EpIV": self.playstarwars,
    "-chess": self.playchess,
    "-weather": self.weather,
```

```

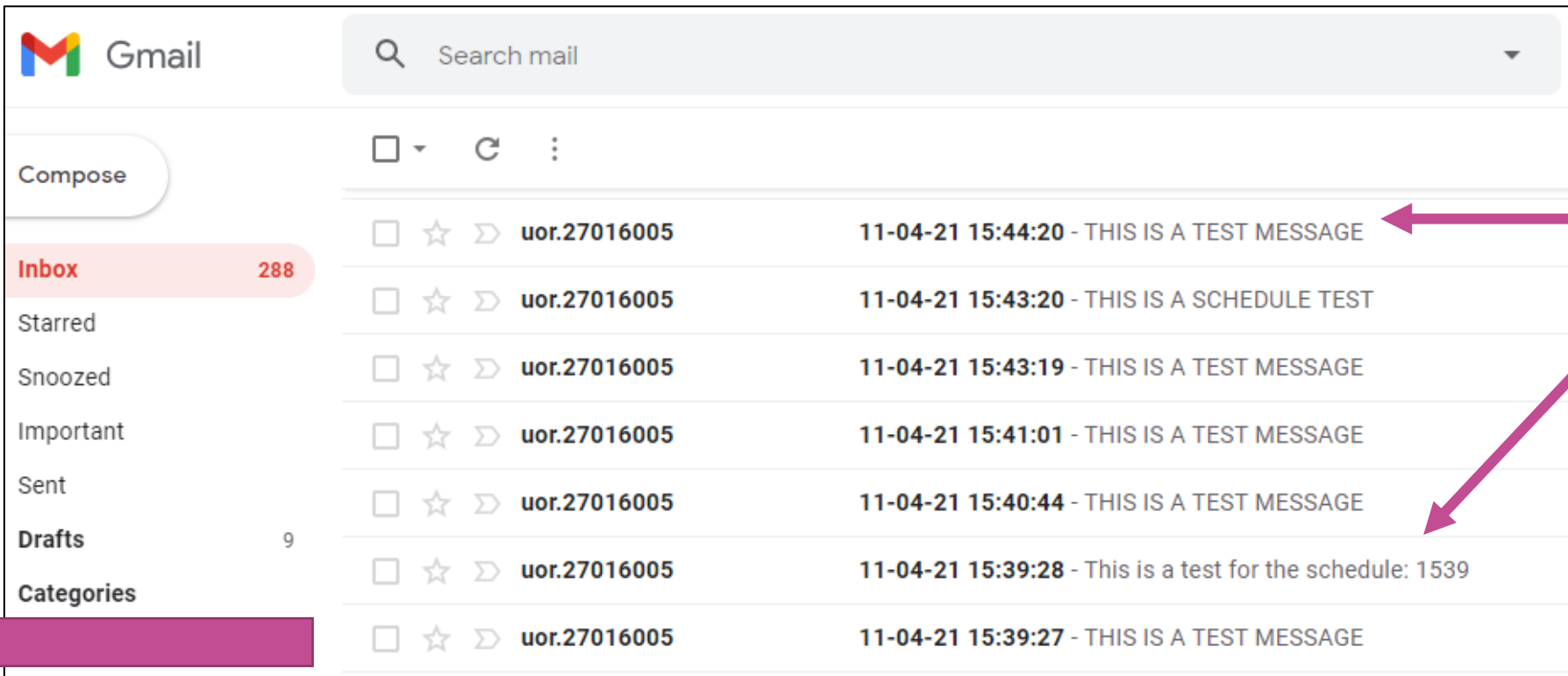
graph LR
    Clipboard[Clipboard] --> SendFile[Send a File]
  
```

The diagram illustrates the flow of data from the Clipboard to the Send a File dialog box. A box labeled "Clipboard" has an arrow pointing to a box labeled "Send a File".

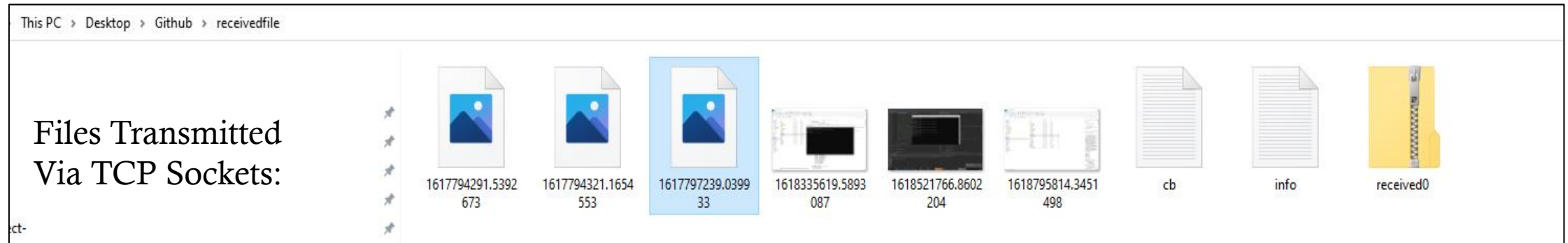
KEY LOGGER:

- Runs on separate thread
 - Capture Ctrl+C, Ctrl+V shortcuts for copy and paste
 - Creates a hidden text file to store the contents and be retrieved at another time
 - Captures Clipboard contents
 - Runs in the background.
 - Invisible
- 

[illegible]



File contents Sent via email. Including for a Scheduler automailer



Directory on Client:

This PC > Desktop > Github > Final-Year-Project- > TESTING > logs >				
Name	Date modified	Type	Size	
screen0	13/04/2021 14:01	PNG File	205 KB	
moreinfoC	13/04/2021 12:19	Text Document	4 KB	
info	13/04/2021 12:19	Text Document	1 KB	
Testprint	12/04/2021 00:31	Text Document	4 KB	
filewritetest3	11/04/2021 14:16	Text Document	1 KB	
tester	11/04/2021 14:06	Text Document	1 KB	
keycodes	08/04/2021 13:29	Text Document	5 KB	
readable	08/04/2021 13:29	Text Document	2 KB	
Video	17/04/2021 13:50	File folder		
Screenshots	08/04/2021 11:27	File folder		
Writing to file	08/04/2021 11:27	File folder		

C:\WINDOWS\py.exe

```
192.168.56.1
Enter 1 to run in malicious mode or 2 to run in virtuous mode: 1
[-] Malicious mode enabled:
entered loop
[@] message received {msg} = -Fsend
[!] FILE SEND MODE: Enabled
Should have worked.
entered loop
[@] message received {msg} = -Frecv
entered loop
[@] message received {msg} = -exe
```

ExeDemo

VBS Script running on client



This is a VBS Script to demo
Remotely running an Executable file

OK

Cancel

C:\WINDOWS\py.exe

```
-ss:          Take a screenshot of the target device
-vid:         Take a series of screenshots which can be used to make a video
-WCrec:       Record (& Retrieve)the webcam from the targets device
-WCplay:      Play The Recorded Webcam frames
-shell:       Non interactive Reverse Shell - Enter CMD Commands to be executed on the target device
-email:       Email contents of a chosen file
-dailymail:   Start a thread to Email the keylog files at a given time everyday
-endmailer:   Stop the email schedule thread
-drop:        Drop the connection
-clear:       Clear Console
-Menu:        Print this menu again
```

Enter the command you want to execute:-Fsend

-Fsend

[+] Enter the file path of the designated folder (NOT A SINGLE FILE): D:\Users\Danny\FROM C\Desktop\Github\Final-Year-Project-TESTING\logs

Size = 225422

*** Got large file ***

*** File saved ***

Enter the command you want to execute:-Frecv

-Frecv

[+] Enter file path: testexe.vbs

[+] Enter the name to save this file as on the victims device (include file extension): testexe2.vbs

*** File sent ***

Enter the command you want to execute:-exe

-exe

[+] Enter the full filepath: logs\testexe2.vbs

FilePath Sent

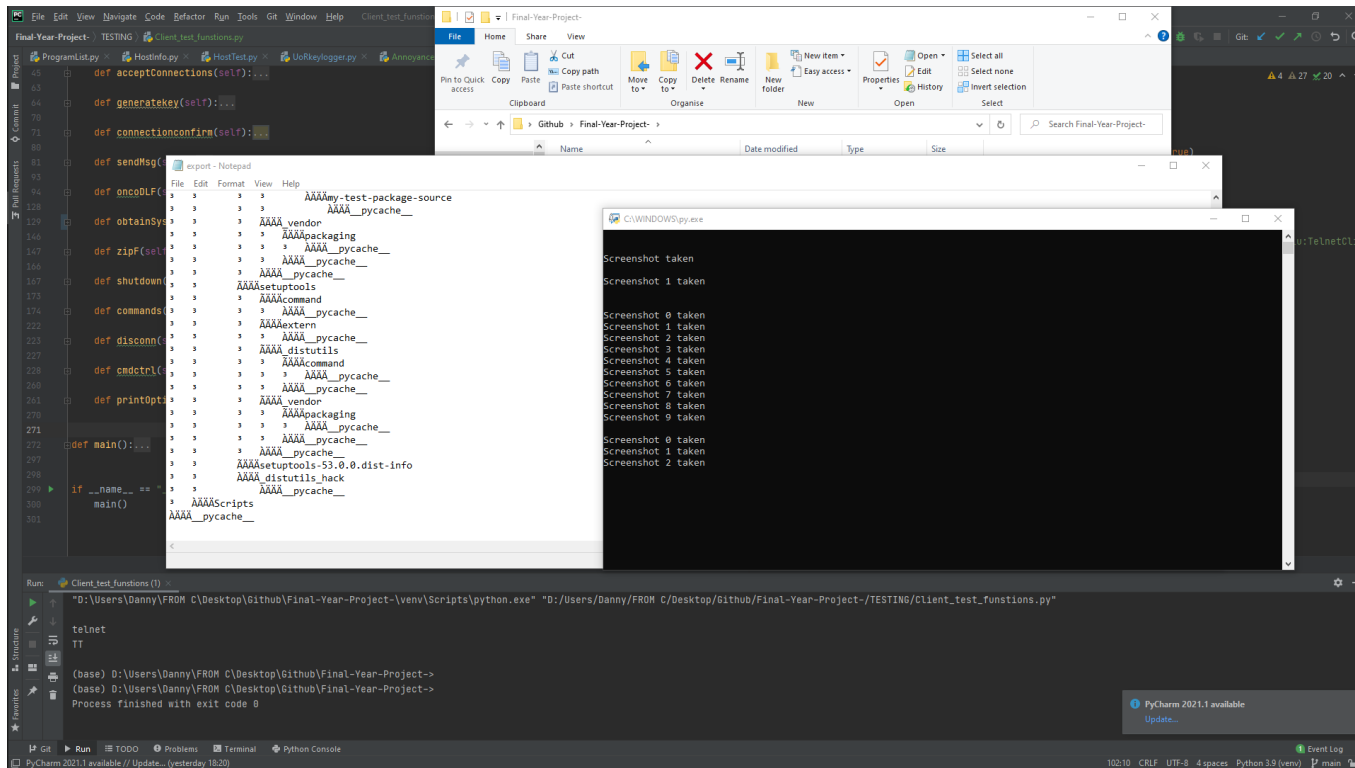
Move file from client to server

Move and rename script from server to client

Run transferred script on client

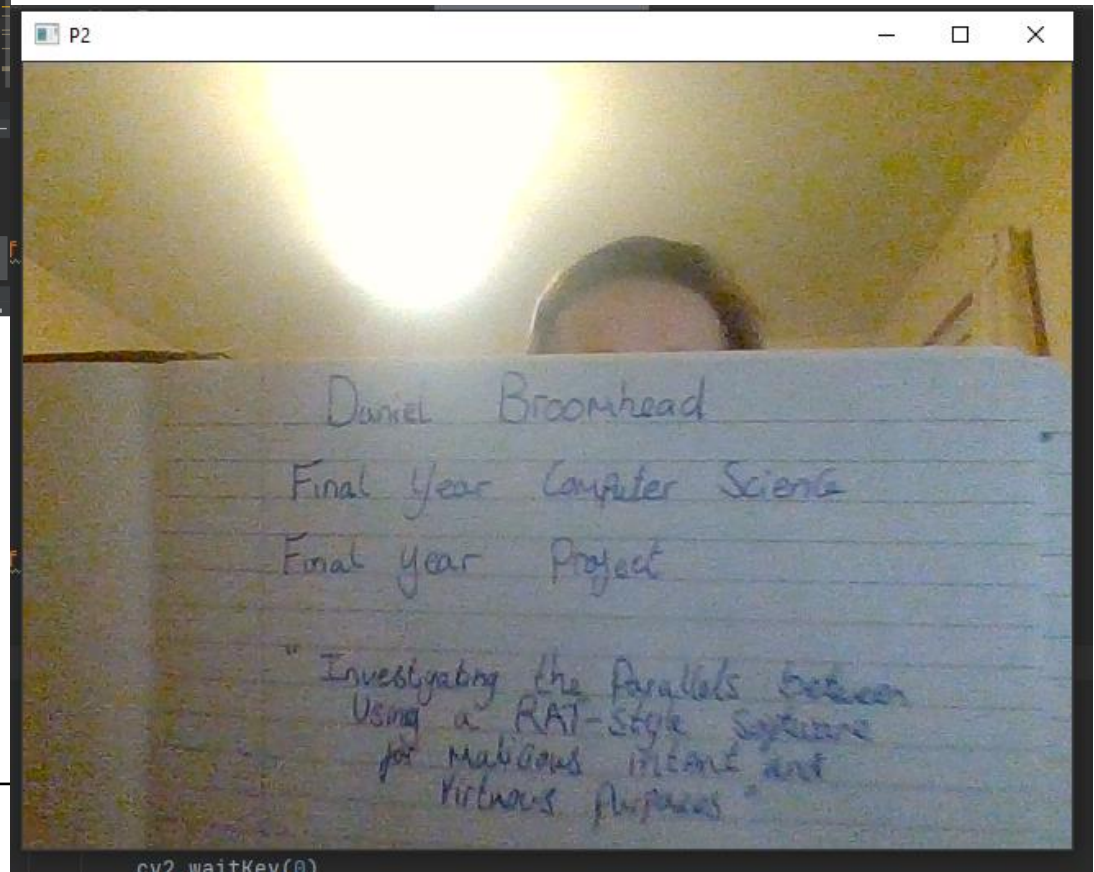
This PC > Desktop > Github > receivedfile > received0 > Users > Danny > FROM C > D			
Name	Type	Compressed size	Password ...
Screenshots	File folder		
Video	File folder		
Writing to file	File folder		
filewritetest3	Text Document	1 KB	No
info	Text Document	1 KB	No
keycodes	Text Document	5 KB	No
moreinfoC	Text Document	4 KB	No
readable	Text Document	2 KB	No
screen0	PNG File	205 KB	No
tester	Text Document	1 KB	No
Testprint	Text Document	4 KB	No

Directory on server:



Screenshot taken from victims device

Capture taken from target's webcam



C:\WINDOWS\py.exe

SERVER:

```
192.168.56.1
*** Listening for incoming connections ***
*** Connection from ('192.168.56.1', 57831) has been established! ***
2
[-] Virtuuous mode
[##] Key = XfXi}qK}j+
```

C:\WINDOWS\py.exe

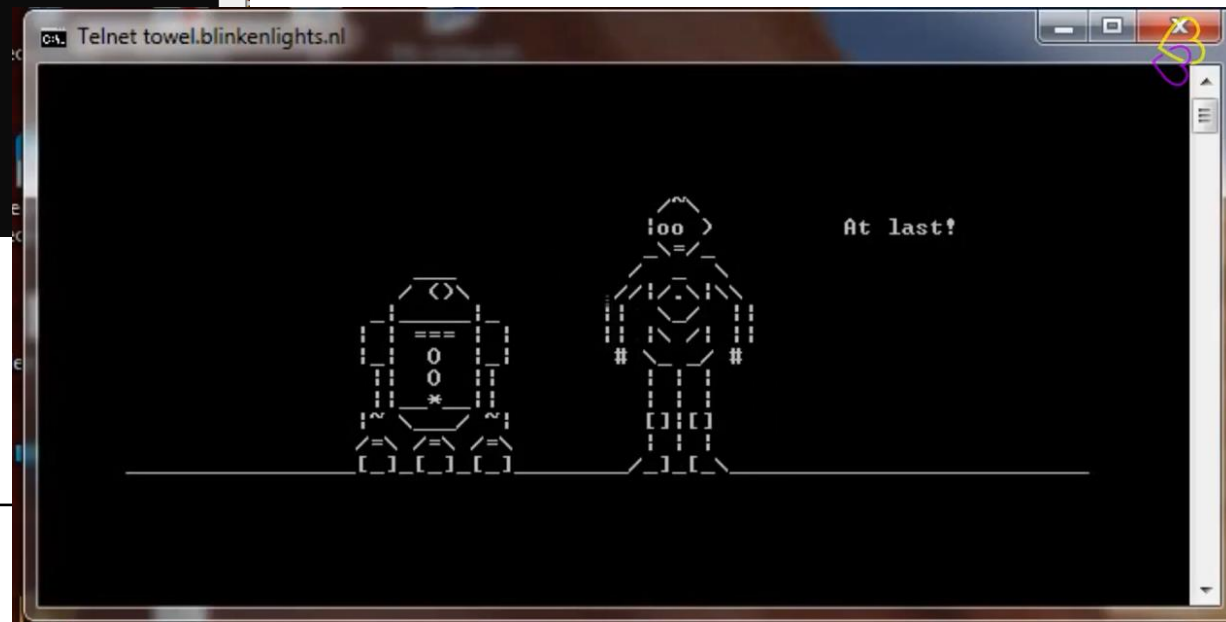
CLIENT:

```
192.168.56.1
Enter 1 to run in malicious mode or 2 to run in virtuuous mode: 2
[-] Virtuuous mode enabled:
Enter the Given Key: XfXi}qK}j+
KEYS MATCHED - PAIRING SUCCESSFUL
entered loop
```

```
C:\WINDOWS\py.exe

Command:          Description:
-msgbox:          Send a custom Alert Messagebox
-msg:             Send a console message
-shutdown:        Shutdown the target device
-shutdownM:       Shutdown the target device, with a custom message
-lock:           Lock the target Device
-restart:         Restart the target system
-EpIV: ← Connect to a telnet server which plays an ASCII Animation of Star Wars EpIV: A New Hope
-chess:          Connect to a telnet server allowing the victim to play chess
-weather:        Opens a telnet based weather forecasting service
-telnet:         Enables Telnet on the targets device - providing they have permissions
-Klstart:        Start the keylogger
-getlogs:        Retrieve the keylogs
-getcb:          Retrieve the clipboard contents and save to file
-Fsend:          Send a file from the Victim to this device
-Frecv:          Send a file from this device to the victim
-ginfo:          Obtain as much information from the victim as possible
-exe:            Run an Executable file or Script
-ss:            Take a screenshot of the target device
-vid:            Take a series of screenshots which can be used to make a video
-shell:          Non interactive Reverse Shell - Enter CMD Commands to be executed on the target device
-email:          Email contents of a chosen file
-dailymail:       Start a thread to Email the keylog files at a given time everyday
-endmailer:       Stop the email schedule thread
-drop:           Drop the connection
-clear:          Clear Console
-Menu:           Print this menu again

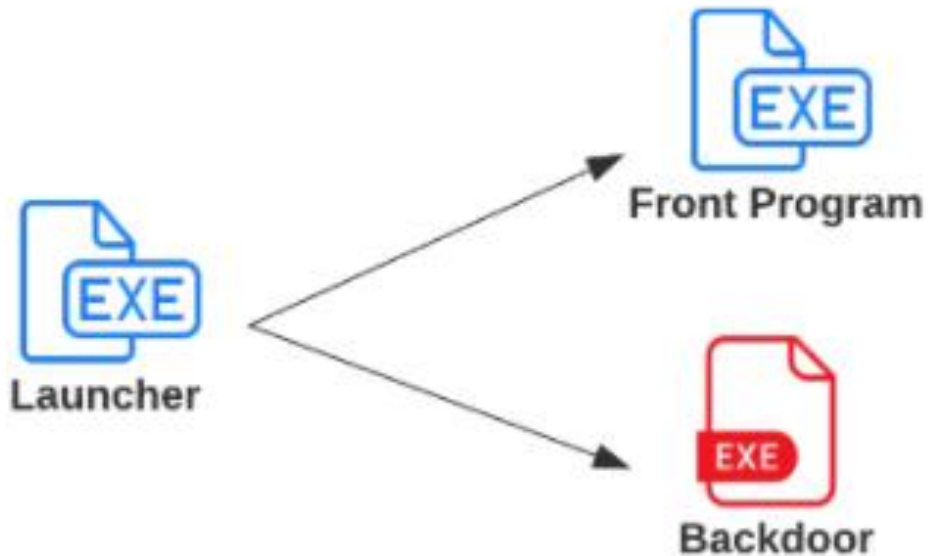
Enter the command you want to execute:-EpIV
-EpIV
[!] 192.168.56.1 is now watching Star Wars Episode IV: A New Hope
SUCCESS
Enter the command you want to execute: _
```



Example Command: "-EpIV" to run StarWars Asciiimation on Target device



Backdoor on the victim's machine



THE GENERATED EXECUTABLES

- PyInstaller used to generate a packaged executable
 - Executable then taken from maze game
 - Create a script which launches the front program(Maze) and a threaded daemon for the client side of the RAT
 - Create an executable from this script.
 - All 3 executables needed on client device.
 - Need to be in designated file paths specified in launcher.py
-

DEPENDENCIES

D0 Server 31.py

socket
sys
os
time
random
string
cv2
Zipfile

D0 Wclient 31.py

socket
Sys
subprocess
Os
Platform
threading
Time
Datetime
Cv2
mss
Zipfile
Pprint
Schedule
Smptlib
Pynput
pyperclip

D0 Wclient 31.exe

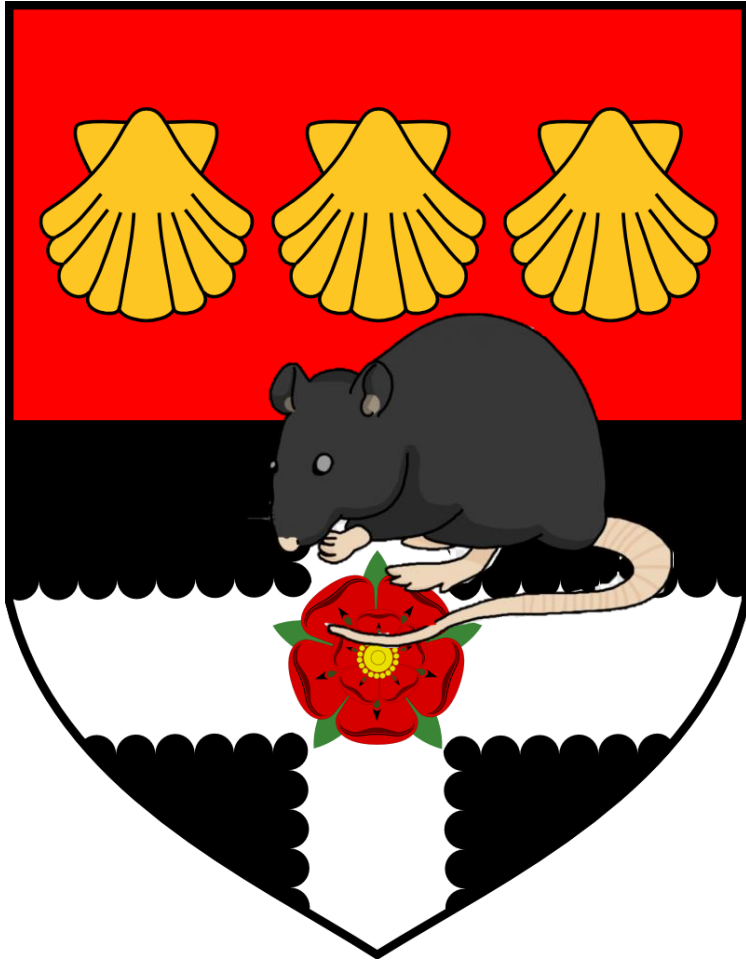
NONE – Not even python needed

When Converting to an .exe using Pyinstaller, all dependencies can be included meaning the program will run standalone.

Launcher.py

Sys
Os

WHAT DOES UORAT OFFER?



Lightweight,

Fast,

OpenSource,

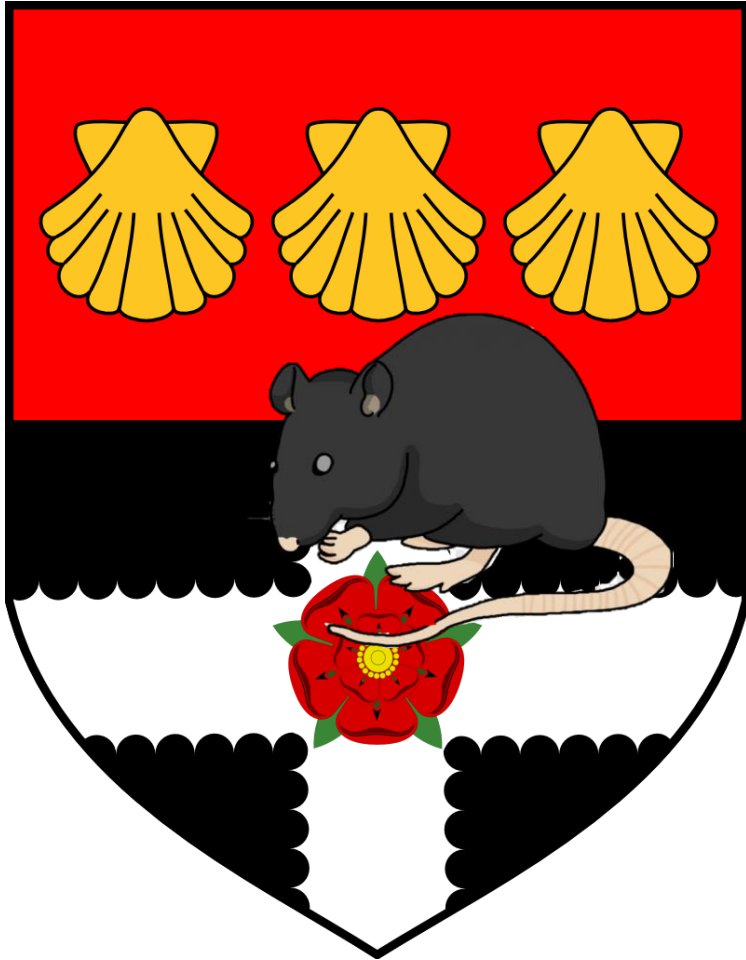
Free,

Easy to use,

Easy to modify and expand

Effective

CONCLUSION:



Fully Functional,

Scalable & Extendable,

All Objectives Achieved,

Results correct to intended purpose,

Reliable,

Complete,

More emphasis could have been placed on UI and aesthetics.

WHO AM I?

- Daniel Broomhead
- 27016005
- R. U. Hacking? Society President
- BCS Berkshire Student Chapter President
- BCS Berkshire Early Careers Advocate
- BCS Open Source Specialist Group – Young Representative
- On track for BSc(Hons) Computer Science – First Class



SUPPLEMENTARY SLIDES:

FUTURE FUNCTIONALITY & IDEAS:



- Graphical user interface : either fully interactive or a graphical design made in console using ASCII art.
- More effort into aesthetics
- Time out feature: Session terminates after extended period.
 - Only required on virtuous version
- More emphasis placed on the Virtuous style software
- More research into RDP
 - Other Desktop Protocols
- More support for other operating systems
- EVEN MORE FUNCTIONALITY

ABOUT THE DEMO:

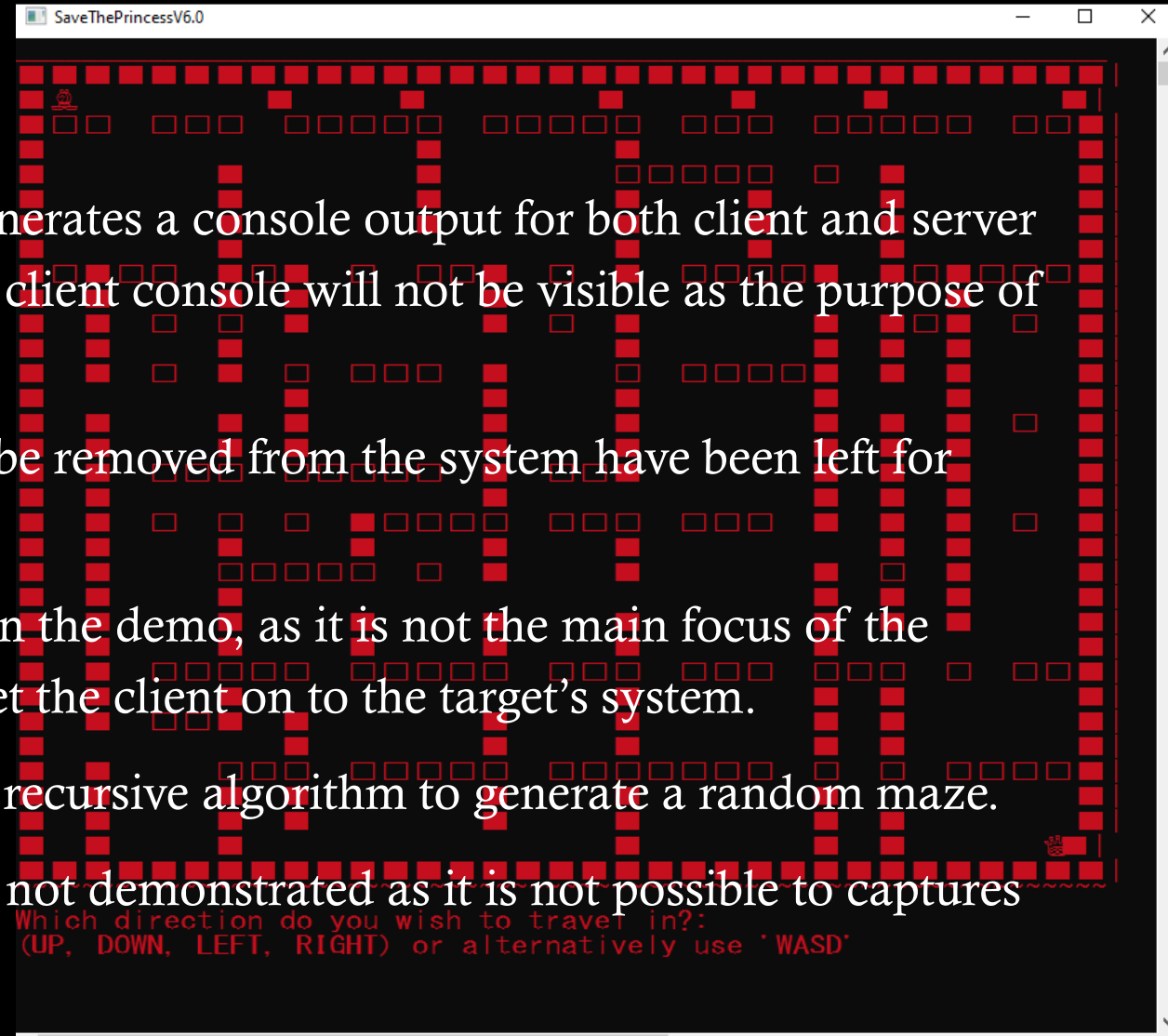
In the demo – the python script is used which generates a console output for both client and server is used. When used in a real world situation, the client console will not be visible as the purpose of this program is designed to be virtually invisible.

For similar reasons, files which would normally be removed from the system have been left for viewing purposes.

The maze front end program will not be shown in the demo, as it is not the main focus of the project, and more of a disguise and method to get the client on to the target's system.

The maze program is created in C++ and uses a recursive algorithm to generate a random maze.

The Shutdown, lock and restart features are also not demonstrated as it is not possible to capture these with screen recording software.



GLOSSARY & TERMS:

RAT : Remote Access Trojan / Remote Administration Tool

TCP : Transmission Control Protocol

UDP : User Datagram Protocol

OOP : Object Oriented Programming

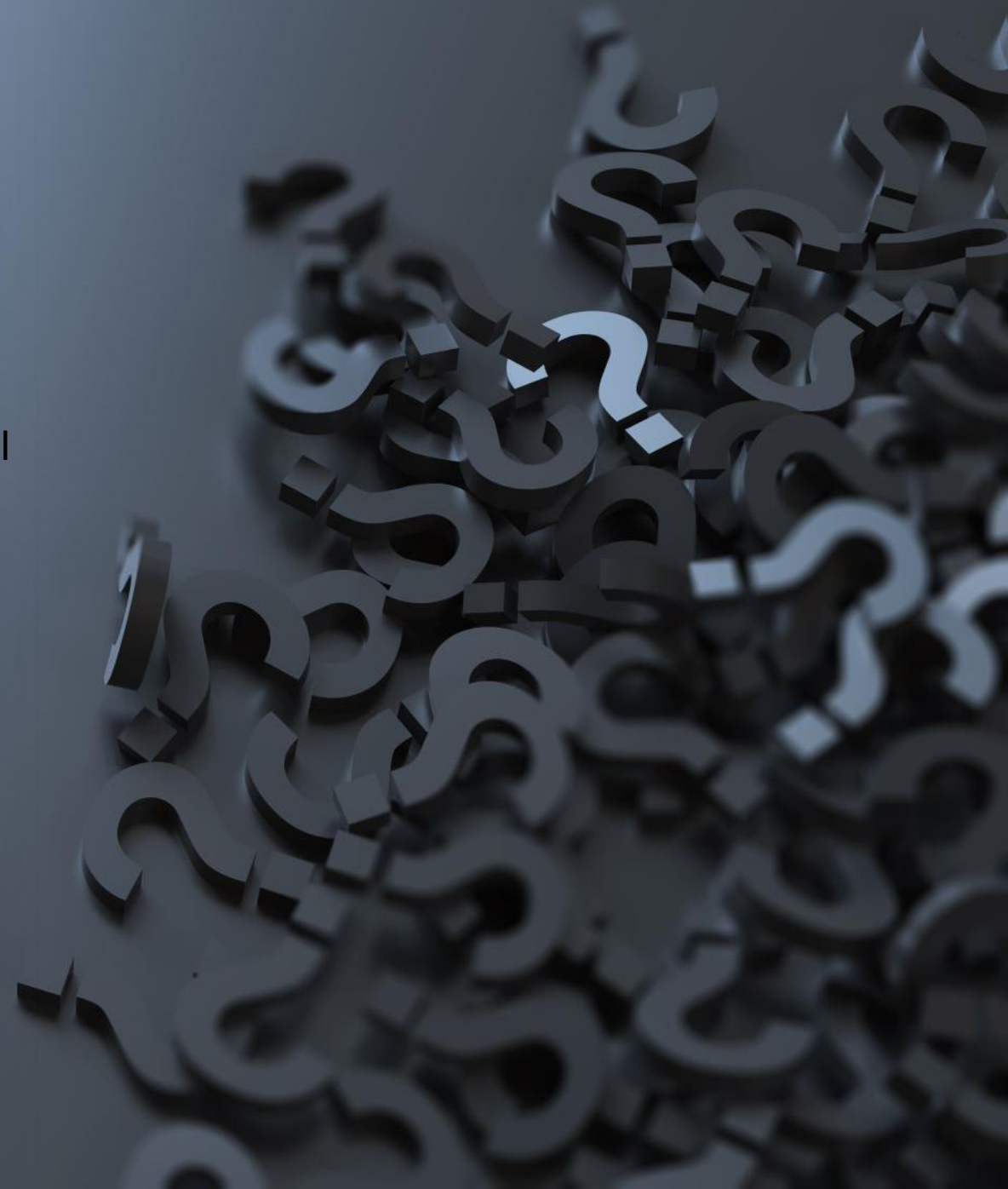
OOD : Object Oriented Design

FTP : File Transfer Protocol

UI : User Interface

GUI : Graphical User Interface

RDP : Remote Desktop Protocol



THE DEMO: