

# Usage Guide

## Client

For the purpose of explaining how to use the backdoor, I will split the client into three parts: Command line, file watching, and retrieving files.

### Part 1A: Command Line

The backdoor contains multiple entries that allows the user to enter in IPs, commands, and process names:

- **Destination IP:** The destination IP is the IP of the target machine where the backdoor server lies. For example, if the server is located on the target machine with an IP of 192.168.0.24, the IP number should be entered here on the client application.



- **Source IP:** The source IP is the IP of the machine that the packet will contain in the IP headers. If the client user's IP is 192.168.0.1, entering the IP will allow you to receive back data from the server. The source IP can also be an IP that is not yours. Any one-way commands that do not require data to be sent back, such as creating a directory, or deleting a file, will not require data to be sent back. This will allow the client user to spoof their IP, and if the packet is captured by an unwanted user and analyzed, the source IP shown will be the spoofed IP that the user has entered.



- **Process Title:** The process title is the title of the backdoor's name that will be on the target machine. Entering the name "kworker" as the process title will change the process title of the backdoor to "kworker". This is a powerful feature for when the backdoor is running. It will significantly increase the security and secrecy of your backdoor; if the target machine has all its processes listed, it will display the name that was given by the client user for the backdoor, hiding it behind other kworker processes.

**Process Title**

- Your Command to send: This is the command that the client user wants to perform on the server command line. For example, entering in “pwd” will display the current directory that the server backdoor is in. The user can also create files using commands on the command line (ex. \$ touch testfile.txt). Many commands ran on the command line can be processed, including retrieving the IP (\$ ipconfig), removing files (\$ rm testfile.txt), and changing ownerships of files (\$ chmod 777 testfile.txt). The user can also create files and remove files from other directories by providing the correct path (\$ touch test/directory/testfile.txt)

**Your commands to send**

#### *Part 1B): Encryption*

Before pressing the “ok” button to send your command to be processed, select a type of algorithm that you would like to encrypt your data. There are three types of encryption algorithms that the client user can select from:

- AES: AES is the most secure form of encryption used in the technology industry. Selecting the AES encryption will encrypt the commands sent by the user in AES. As AES is a form of symmetric encryption, only a private key is required.
- RSA: The RSA encryption uses a python library called Crypto. As RSA is a form of asymmetric encryption, we will require two keys to perform the encryption. We create a private key using a random generator and extract a public key from that. The client uses the public key from the server to encrypt the data, and once sent, the server decrypts the data using its own private key.
- Yiao: This is just a simple and fun encryption that uses a combination of substitution, then transposition cipher to encrypt the data. Using only a substitution cipher would allow the user to easily brute force crack the encrypted data, so we add another layer of security to it. We use Viginere’s cipher to make the polyalphabetic substitution cipher, then use a transposition cipher on the already encrypted data, putting it in a box by row, then transposing it to grab the data by column.

☒ AES

☐ RSA

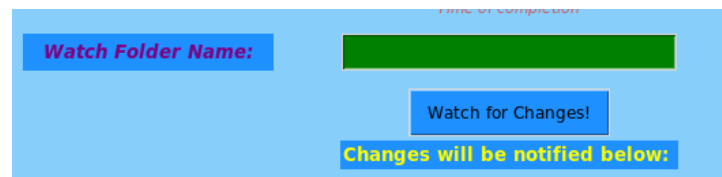
☐ Yiao's Encryption

## Part Two: File Watching

My backdoor application contains the feature of file watching. The user can enter the directory that they would like to watch for file changes on, providing a relative path to the backdoor server. For example, if the client user wanted to watch for file changes that occurs in the directory “test” which lies in the backdoor directory, the user would simply type in the user interface entry “test”. If the user typed in test/directory, the watch would look for changes that occur in backdoor/test/directory if it exists.

File watcher uses a Python API library called Watchdog that monitors for file system events. I’ve created the class to check for changes occurring such as modification of files, deletion of files, and addition of files.

- Deletion of files: The client user would be notified if the file was deleted or moved to a different directory. In the case of renaming a file, the user would be notified with the name of the file that has been deleted and the name of the new file which was created
- Creation/Addition of files: The client user would be notified if a new file was created or moved in from another folder. The user will be notified with the name of the file that is created.
- Modification of files: In the case that a file has been modified, the user will be notified with the name of the file that has been modified.



## Part Three: Retrieving Files

The third main feature of the backdoor application is to retrieve files from the target machine. Used in conjunction with the command line feature, the user can enter in the name of a file to retrieve. The user can type in “ls” to list the files that are in the current directory, then type in the file that they want to retrieve. For example, typing in key.log would transfer over the keylogger file. The transfer of files is done via sockets. The user can only select files that reside in the directory of the server application.



### Screenshot

The screenshot button in the menu allows the user to take a screenshot on the target machine and save it to the current directory of the server application. The file can then be retrieved by using the retrieve file function in Part Three of the user guide.

### Exit

The client user can close the graphical user interface by clicking the *quit* button located in the menu bar.



### Server

Upon running the server, the keylogger will immediately start recording keystrokes. A file called security.log (for secrecy purposes) that resides in the same directory as the server backdoor will be created and record all keystrokes. The server will begin listening to any commands that are sent by the client and upon receiving commands will process them, sending results back to the client. The server can be shut down from the client by typing "exit" in the command line