

Jorael Jamison

CSIA 440 – Cyber Testing & Penetration

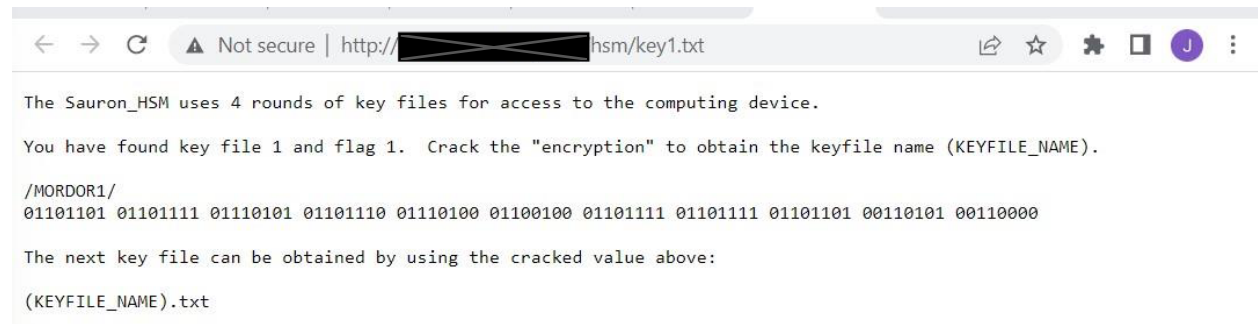
Professor Robinson

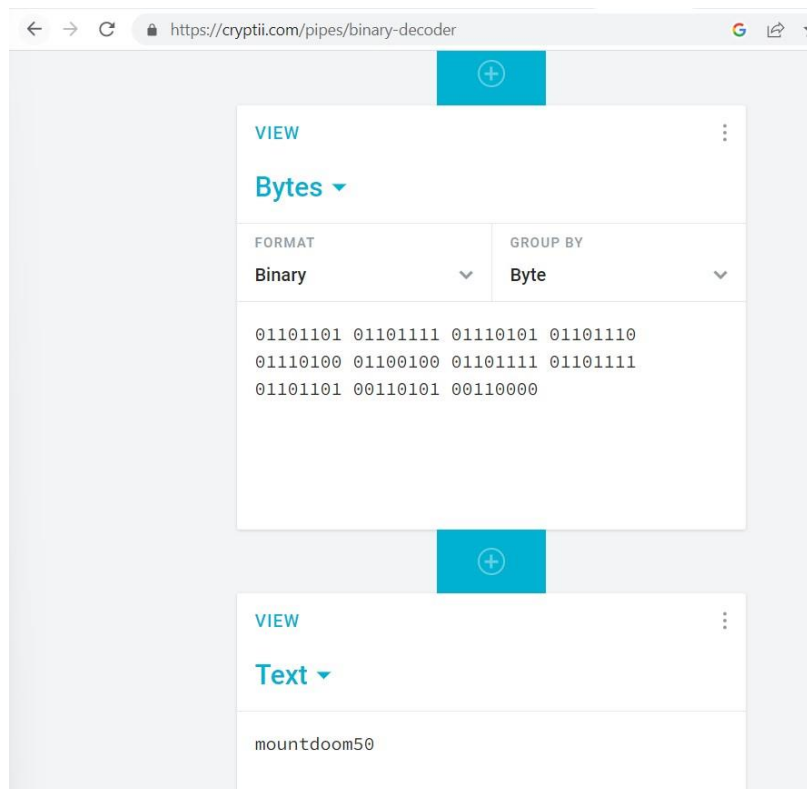
06/01/2023

Exam Sauron_HSM Appliance

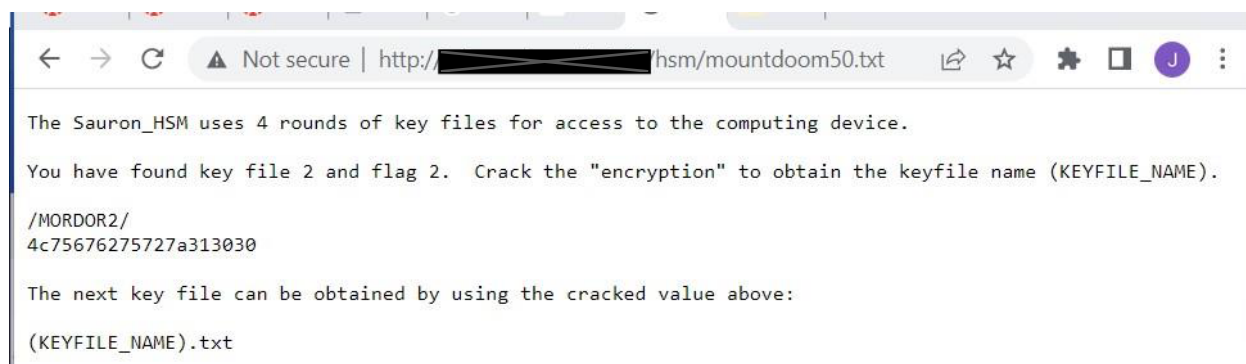
Flag 1 (/MORDOR1/):

The final exam study guide said this pen-test would be performed on the [REDACTED] Security company. I decided to explore the website [REDACTED] to see if anything changed from the last time that we used the site for a capture the flag exam. I found MORDOR1 by clicking on the 'products' tab and saw 'SAURON_HSM'. I clicked on this, and it loaded a page displaying a file titled: key1.txt. This displayed the first key file's encrypted name: *01101101 01101111 01110101 01101110 01110100 01100100 01101111 01101111 01101101 00110101 00110000*. It mentioned the cracked value would be the name needed to obtain the next key file. I cracked the encryption by entering the string into a 'binary to text' converter I found online. The result was: **mountdoom50**. Finally, I appended this to the end of the URL with the file extension '.txt'.





Flag 2 (/MORDOR2/):



← → ↻ 🔒 https://www.boxentriq.com/code-breaking/cipher-identi... 🔍 📄 ☆ ⚙️ 🖨️ 👤

BOXENTRIQ TOOLS PUZZLE ABOUT

Cipher Identifier and Analyzer

Find Tools...

Struck with a cipher or cryptogram? This tool will help you identify the type of cipher, as well as give you information about possibly useful tools to solve it.

This tool uses AI/Machine Learning technology to recognize over 35 common cipher types and encodings including: Caesar Cipher, Vigenère Cipher (including the autokey variant), Beaufort Cipher (including the autokey variant), Playfair Cipher, Two-Square/Double Playfair Cipher, Columnar Transposition Cipher, Bifid Cipher, Four-Square Cipher, Atbash Cipher, and many more!

Enter Ciphertext here

4c75676275727a313030

Analyze Text Copy Paste Text Options...

Note: To get accurate results, your ciphertext should be at least 25 characters long.

Analysis Results

4c75676275727a313030

Your ciphertext is likely of this type:

Hexadecimal Code (click to read more)

Note: Your ciphertext is less than 25 characters long. Results are less reliable.

← → ↻ 🔒 https://string-functions.com/hex-string.aspx 🔍 📄 ☆ ⚙️ 🖨️ 👤

www.string-functions.com

ONLINE STRING MANIPULATION TOOLS

Hex To Text Converter Online Tool

Enter the hexadecimal text to decode, and then click "Convert!":

4c75676275727a313030

Convert!

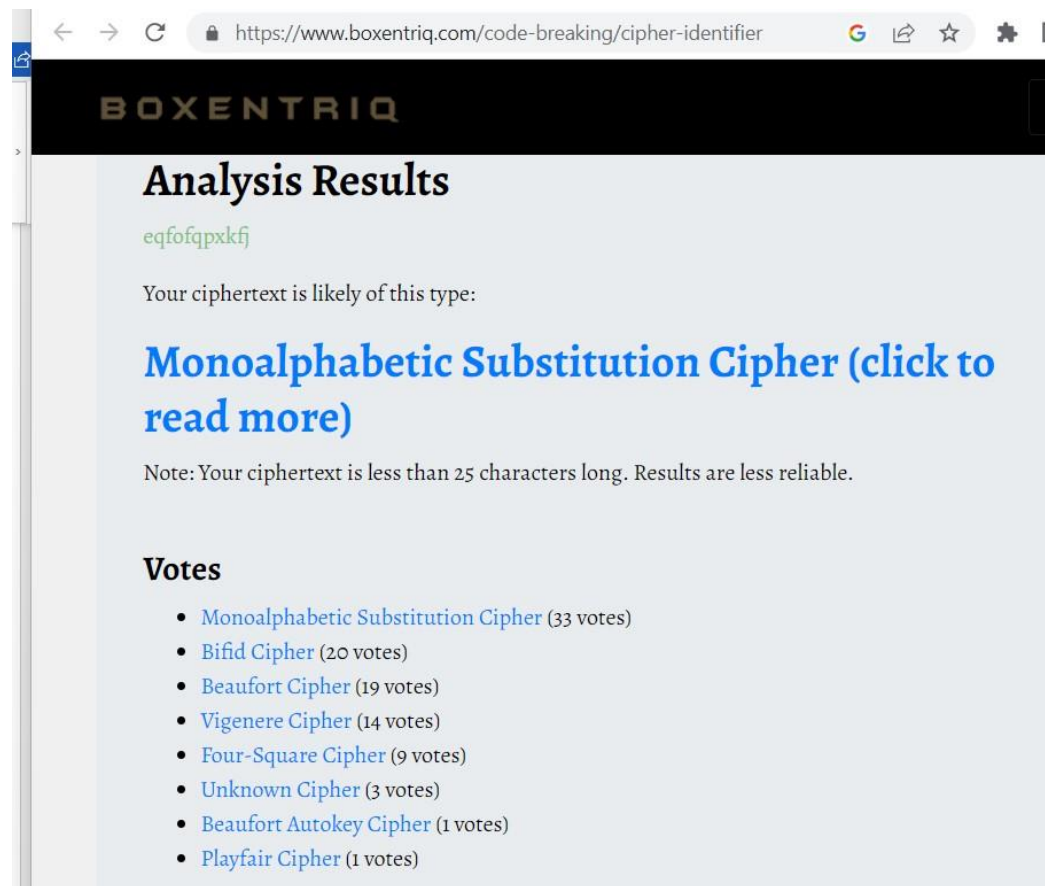
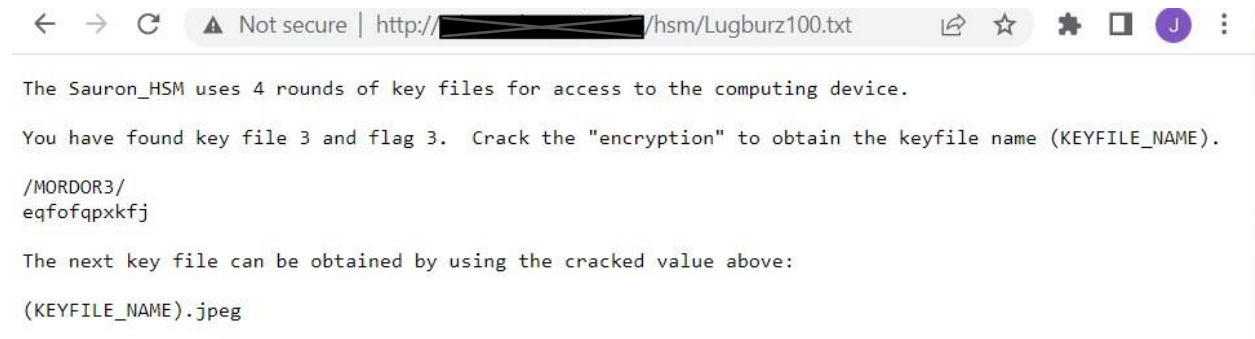
The decoded string:

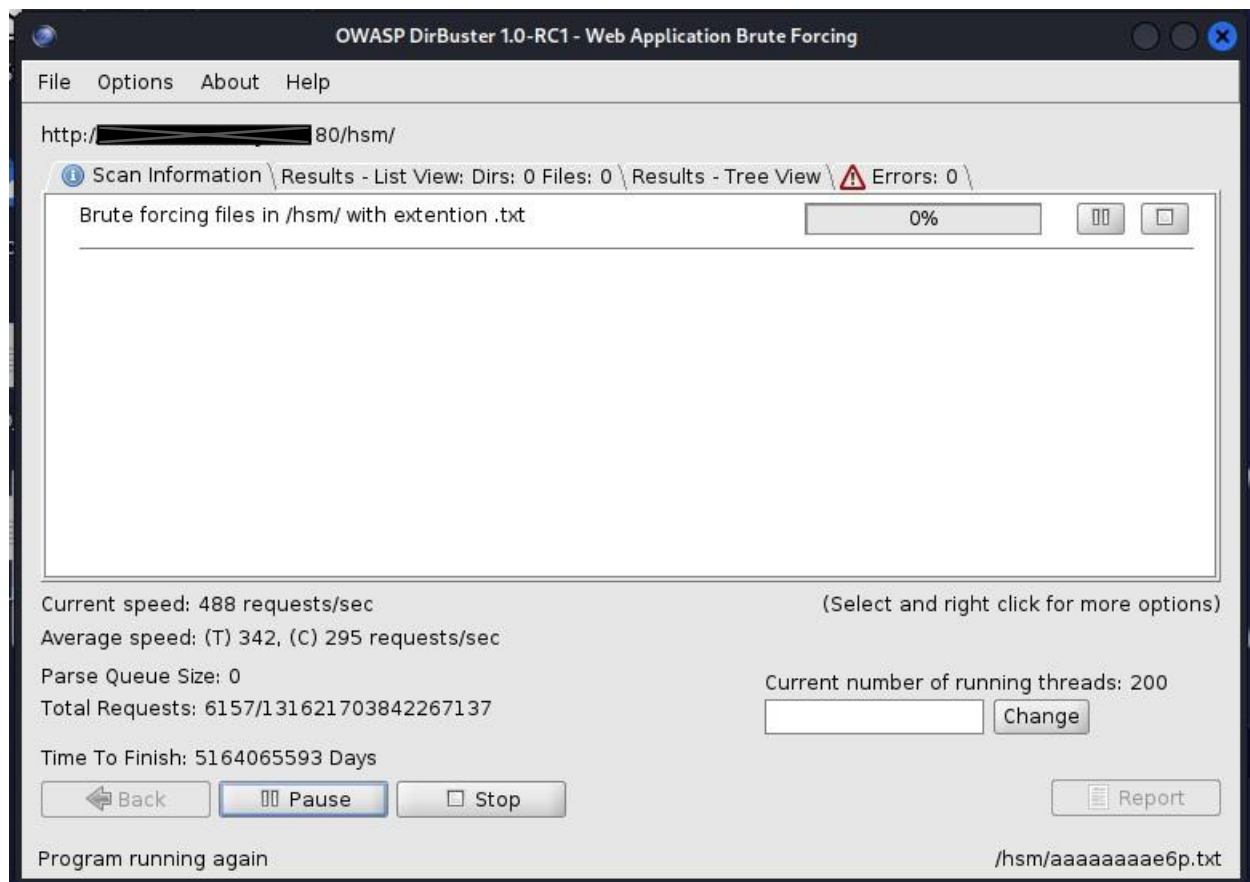
Lugburz100

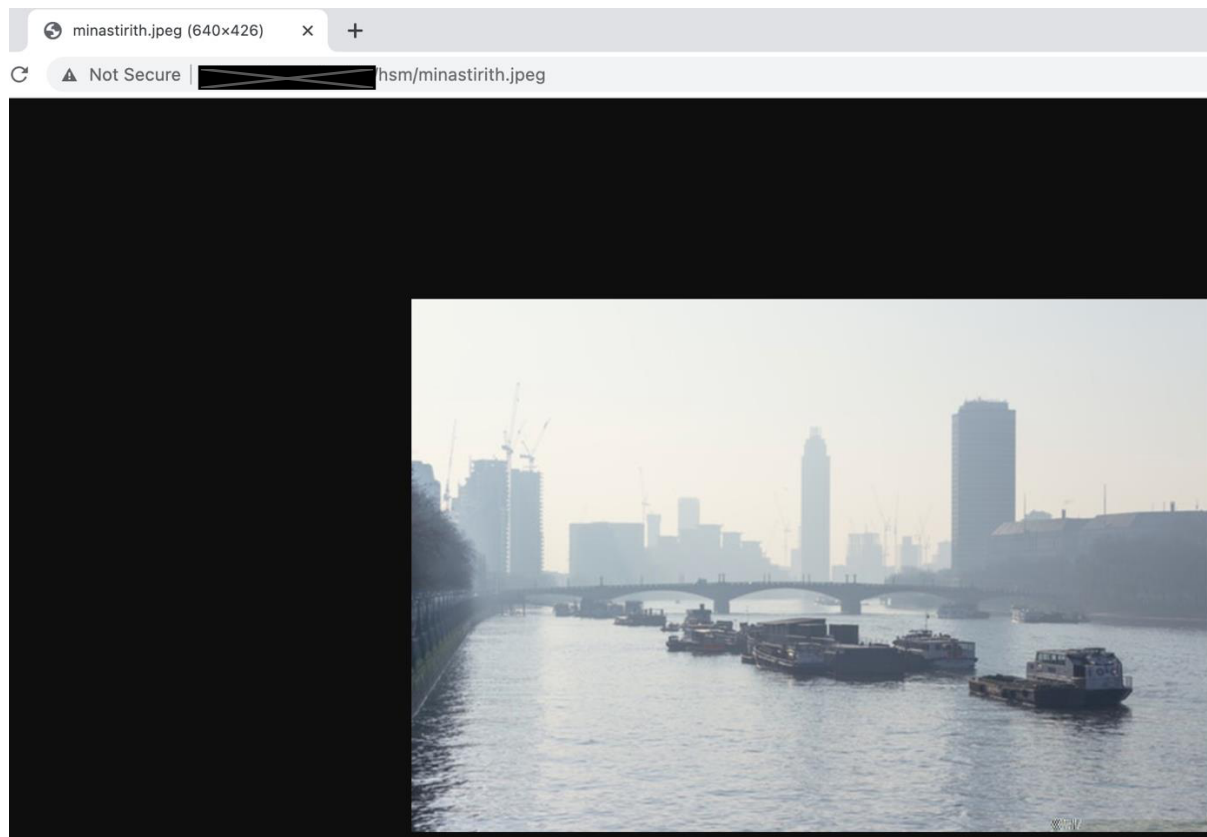
Please note: any spaces or colons (:) in the hexadecimal string will be removed.

I found MORDOR2 after I cracked the value from MORDOR1. I appended 'mountdoom.txt' to the end of the [REDACTED] URL and it loaded a page displaying the file. This displayed the second key file's encrypted name: 4c75676275727a313030. It mentioned the cracked value would be the name needed to obtain the next key file. I cracked the encryption by first analyzing the string online and entering it into a Hex to Text Converter tool I found online. The result was: **Lugburz100**. Finally, I appended this to the end of the URL with the file extension 'txt'.

Flag 3 (/MORDOR3/):



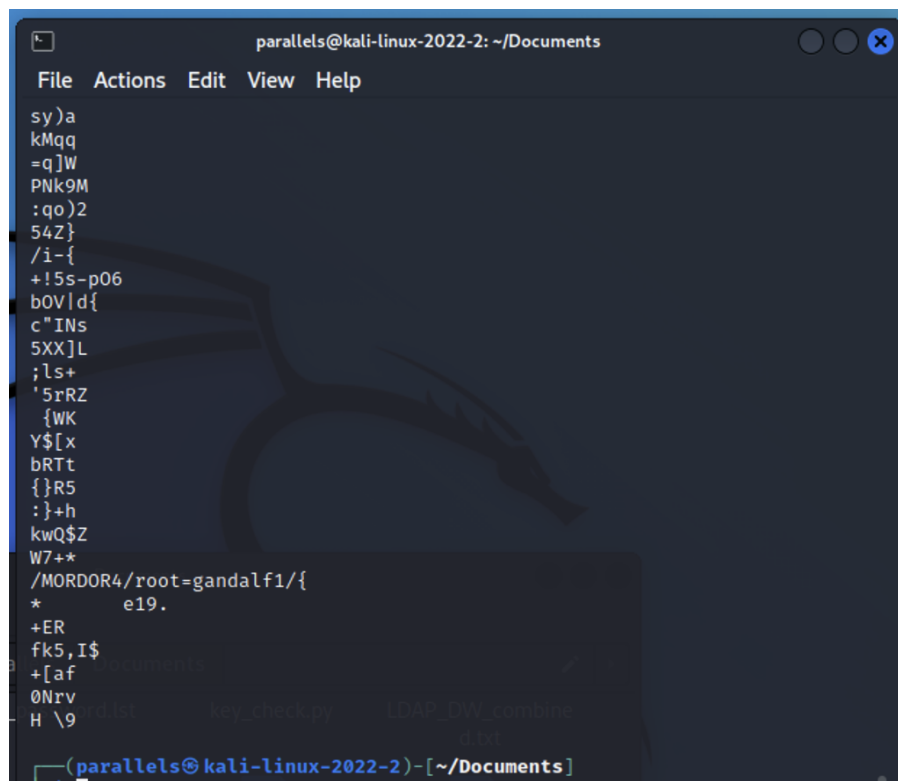




I found MORDOR3 after I cracked the value from MORDOR2. I appended 'lugburz100.txt' to the end of the [REDACTED] URL and it loaded a page displaying the file. This displayed the third key file's encrypted name: *eqfofqpxkfj*. It mentioned the cracked value would be the name needed to obtain the next key file. Unfortunately, I failed to crack this code. I first ran it through a cipher analyzer online which displayed that it was most likely a 'mono-alphabetic substitution cipher.' I went from there and exhausted all efforts to crack it by using all variations of encryption ciphers on different code-breaking websites, but I was unable to find anything that resembled the correct answer. I also spent time using CrypTool to analyze various ciphers to crack it, in addition to a last resort of brute-forcing and dictionary attacks by using the tool Dirbuster in Kali. This would most likely work, but I do not have that many years to go through all the possible combinations of .txt files. I also could not find anything else on the [REDACTED] website or the SAURON_HSM server that could help or maybe offer a clue or key to cracking the code. I ran 'find' commands to search for any other relevant files.

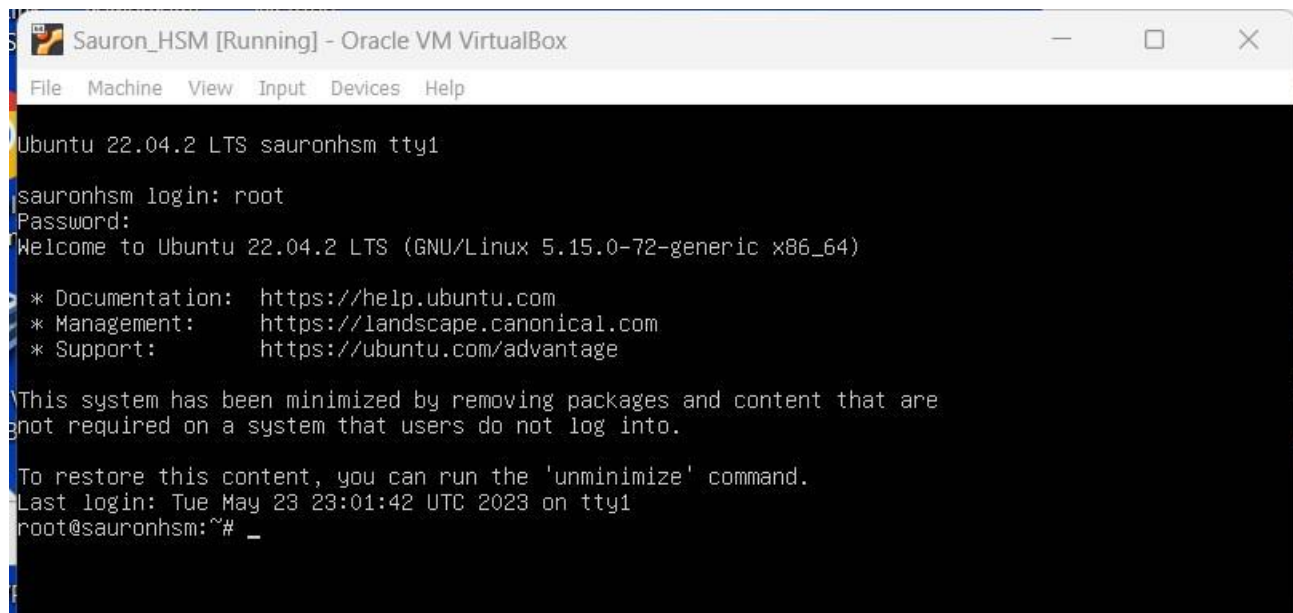
Upon reviewing today's class zoom meeting I retried this cipher by reversing the string as hinted: *jfxpqqfofqe*. I ran this on the dcode website and both Affine and Cesar substitution Cipher's listed the phrase "minastirith" as the number 1 result. I read online this was associated with Lord of the Rings, which was another hint. I tried pulling this file from the [REDACTED] website using "minastirith.txt" and kept receiving a 404 error. I was certain I had the correct word now, so I went back and double checked the lugburz100 file. I apparently had been trying to pull this with a txt extension and that was my issue, it needed to be .jpeg. I was focused on txt as that was the previously flags format.

Flag 4 (/MORDOR4/):



```
parallels@kali-linux-2022-2: ~/Documents
File Actions Edit View Help
sy)a
kMqq
=q]W
PNk9M
:qo)2
54Z}
/i-{
+!5s-p06
b0V|d{
c"INs
5XX]L
;ls+
'5rRZ
{WK
Y$[x
bRTt
{}R5
:}+h
kwQ$Z
W7+*
/MORDOR4/root=gandalf1/{
*
e19.
+ER
fk5,I$
+[af
0Nrv
H \9
(parallels@kali-linux-2022-2)-[~/Documents]
```

Now that I have found the image for MORDOR 3, I began to analyze for any steganography. On Kali, I used `wget` to download the image from the `XXXXXXXXXX` URL and used the `'strings'` command which displayed MORDOR 4, along with the root password to the server = `gandalf1`. As I had previously accessed the server through means of pre-boot vulnerability, I opened a fresh copy of the Sauron server to try this password. It was successful to login (shown below).



```
Sauron_HSM [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help

Ubuntu 22.04.2 LTS sauronhsm tty1

sauronhsm login: root
Password:
Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 5.15.0-72-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Tue May 23 23:01:42 UTC 2023 on tty1
root@sauronhsm:~# _
```

Flag 5 (/MORDOR5/):

hsm.docx Properties

General

Summary

Statistics

Contents

Custom

Name:

Add

Delete

Checked by

Client

Date completed

Department

Destination

Disposition

Type:

Text

Value:

☐ Link to content

Properties:

Name	Value
MORDOR5	/MORDOR5/
NEXT_FLAG	Look at the end of the document

OK

Cancel

Properties

Size

26.6KB

Pages

9

Words

4537

Total Editing Time

21 Minutes

Title

Add a title

Tags

Add a tag

Comments

Add comments

Related Dates

Last Modified

Today, 1:05 AM

Created

2/3/2020 6:05 AM

Last Printed

Related People

Author

GF

Gussie Fink-Nottle

Add an author

Last Modified By

JR

Jorael R Jamison

Related Documents

Open File Location

Show All Properties

He drew out slowly from an upper waistcoat pocket a scarlet card-case, and as slowly produced a very large card. Even in the instant of its production, they fancied it was of a queer shape, unlike the cards of ordinary gentlemen. But it was there only for an instant; for as it passed from his fingers to Arthur's, one or another slipped his hold. The strident, tearing gale in that garden carried away the stranger's card to join the wild waste paper of the universe; and that great western wind shook the whole house and passed.

You can find /MORDOR6/ in keys.xxx

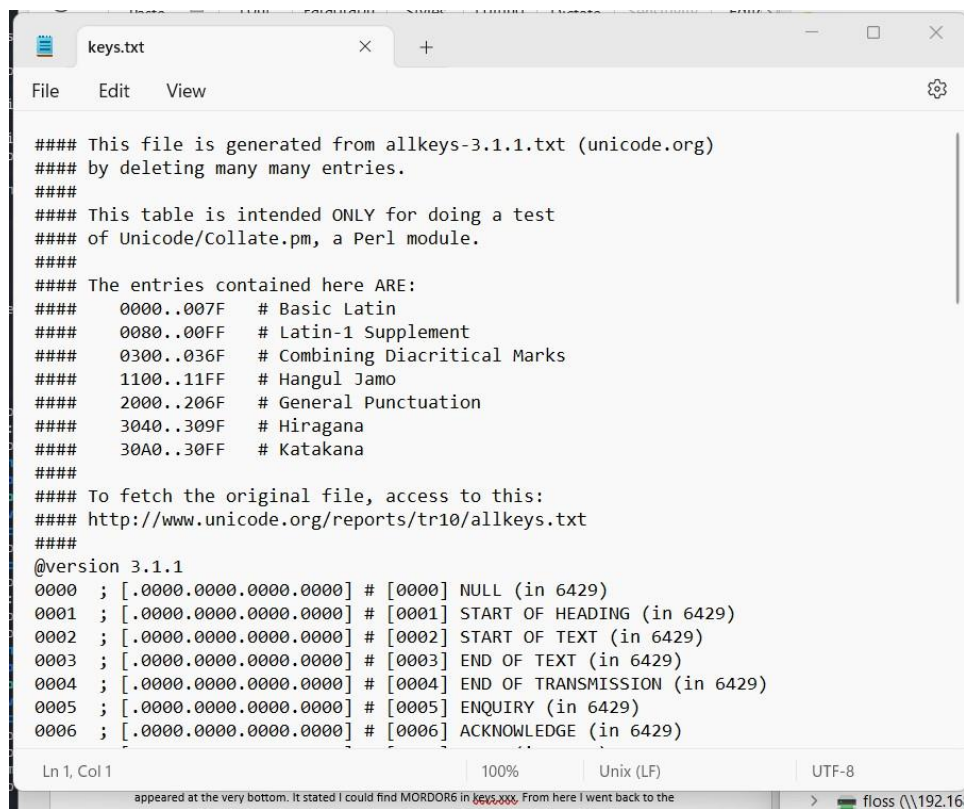
I found MORDOR5 after I cracked root access to the SAURON_HSM server. I located and saved the file named: **hsm.txt** to my computer for analysis. More detailed steps of how I obtained this file are in MORDOR6 below. Upon further examining the document's meta-data I found in the properties that this was MORDOR5. It also referenced that I could locate the next flag at the end of the document. The end of the document showed MORDOR6 could be found in the file: keys.xxx.

Flag 6 (/MORDOR6/):

```
saruman1:x:1000:1000:saruman1:/home/saruman1:/bin/bash
root@sauronhsm:~# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
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
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
systemd-network:x:101:102:systemd Network Management,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:102:103:systemd Resolver,,:/run/systemd:/usr/sbin/nologin
messagebus:x:103:104::/nonexistent:/usr/sbin/nologin
systemd-timesync:x:104:105:systemd Time Synchronization,,:/run/systemd:/usr/sbin/nologin
pollinate:x:105:1::/var/cache/pollinate:/bin/false
sshd:x:106:65534::/run/sshd:/usr/sbin/nologin
usbmux:x:107:46:usbmux daemon,,:/var/lib/usbmux:/usr/sbin/nologin
saruman1:x:1000:1000:saruman1:/home/saruman1:/bin/bash
root@sauronhsm:~#
```

```
saruman1@sauronhsm:~# cat /root/.ssh/authorized_keys
Permission denied
saruman1@sauronhsm:/$ sudo cp /root/hsm.docx /home
saruman1@sauronhsm:/$ ls /home
hsm.docx saruman1
saruman1@sauronhsm:/$
```

```
root@sauronhsm:/# find . -name keys.*
./usr/share/perl/5.34.0/Unicode/Collate/keys.txt
root@sauronhsm:/#
```



```
##### This file is generated from allkeys-3.1.1.txt (unicode.org)
##### by deleting many many entries.
#####
##### This table is intended ONLY for doing a test
##### of Unicode/Collate.pm, a Perl module.
#####
##### The entries contained here ARE:
##### 0000..007F # Basic Latin
##### 0080..00FF # Latin-1 Supplement
##### 0300..036F # Combining Diacritical Marks
##### 1100..11FF # Hangul Jamo
##### 2000..206F # General Punctuation
##### 3040..309F # Hiragana
##### 30A0..30FF # Katakana
#####
##### To fetch the original file, access to this:
##### http://www.unicode.org/reports/tr10/allkeys.txt
#####
@version 3.1.1
0000 ; [.0000.0000.0000.0000] # [0000] NULL (in 6429)
0001 ; [.0000.0000.0000.0000] # [0001] START OF HEADING (in 6429)
0002 ; [.0000.0000.0000.0000] # [0002] START OF TEXT (in 6429)
0003 ; [.0000.0000.0000.0000] # [0003] END OF TEXT (in 6429)
0004 ; [.0000.0000.0000.0000] # [0004] END OF TRANSMISSION (in 6429)
0005 ; [.0000.0000.0000.0000] # [0005] ENQUIRY (in 6429)
0006 ; [.0000.0000.0000.0000] # [0006] ACKNOWLEDGE (in 6429)
```

I cracked root access to the SAURON_HSM server using a pre-boot vulnerability.. Once logged in, I did a ls command to see what was in the home directory. It showed a file titled: hsm.docx. I attempted to cat this file but it displayed text that was not readable. I attempted to login to WinSCP to transfer the file to my computer for further analysis, but it would not allow me access and said, “not authorized”. I found a way around this by going back to the server and searching for any other users that have sudo privileges. I ran a cat command on the passwd file and saw a user by the name: saruman1. I changed the password and logged in as this user. Once logged in I transferred the hsm.docx file to this account by using the command: sudo cp (root path to file) (local path to copy). I went back to WinSCP and was able to successfully log in as saruman1 and find the file that I copied over. I used this technique, in addition to ‘wget’, for many other files I wanted to examine further from the root account and [REDACTED] website. I could not find anything further of relevance. The HSM document was examined, and it was a long publication. I did a quick search for keywords and found ‘MORDOR’ appeared at the very bottom. It stated I could find MORDOR6 in keys.xxx. From here I went back to the server and did a ‘find’ search for this file and located the path (shown above) to a keys.txt file in the Unicode path. I saved that file to my computer (screenshot above).

After analyzing this file for a while I realized it was the wrong keys file as I could not find anything by skimming though the document or by checking its metadata. I went back to the server and did another ‘find’ search, but this time for hidden files and found the correct “keys.xxx” in root. I copied the file over to Saruman1 and used WinSCP to transfer to my computer for further analysis. It appeared to be encrypted as it was unreadable. I also did a strings search on the file and saw “files.xxxUT”. I did a find search and found a files.py that may have some significance, so I copied it over with the same process. I moved on from this and tried another strategy as I wasn’t getting anywhere.

Today's zoom meeting helped as you mentioned to be persistent, and you spoke about zipping to nest files. I used the keys.xxx file previously saved and in Kali ran the command "file keys.xxx" to identify the file type, it showed it was a "Zip archive data file" and to use the "deflate" method. I researched online and entered "unzip keys.xxx" and it inflated the file creating a folder named files.xxx. I also unzipped this folder and inside was a bunch of bin files, a run_md5.py script, and a test.txt file that contained a long list of duplicate md5 hashes (shown below).

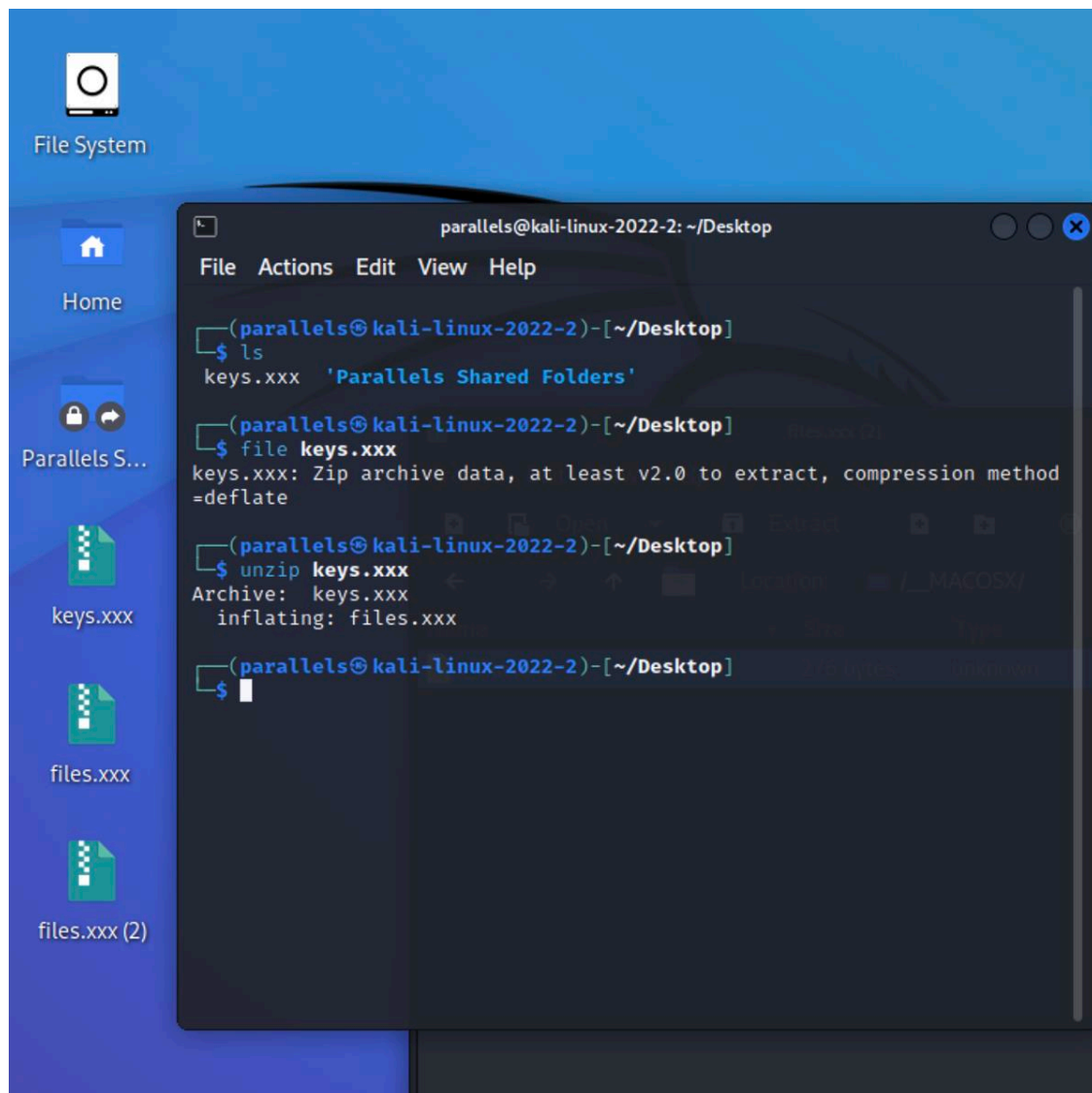
Now that I knew what I was looking for, I went back to the server to attempt to unzip it directly from there. I had to first download the unzip command using "sudo apt install unzip." I was able to unzip the hidden /root/.keys.xxx file and files.xxx was extracted. I unzipped the files.xxx folder and a "files" directory appeared. I ran the 'run_md5.sh' script and it executed but read "md5: command not found." I attempted to cat one of the bin files and it appeared encrypted.

After I had originally submitted this exam, I went back and rechecked the bin files as it was bothering me why I couldn't find the flag. I opened the test.txt file that appeared to have all the same MD5 Hashes, however, to be sure I looked through each one and found binFile-695 that stood out as having a different hash value. I did the cat command to view bin-file 695, at first, I tried to decode the text and hash online and using hashcat but unsuccessful. Next, I decided to run the strings command on the binFile and that is when the MORDOR6 flag appeared. All screen shots throughout this process are documented below.

The final flag is the last screen shot: MORDOR6/// found in binFile-695.

```
root@sauronhsm:~# find . / -name ".keys.*"
./keys.xxx
/root/.keys.xxx
root@sauronhsm:~#
```

```
root@sauronhsm:/# find . / -name "files.*"
./usr/share/python3/debpython/__pycache__/files.cpython-310.pyc
./usr/share/python3/debpython/files.py
/usr/share/python3/debpython/__pycache__/files.cpython-310.pyc
/usr/share/python3/debpython/files.py
```




```
Open  *test.txt
~/.cache/fr-iHNF4H/fi

1 MD5 (binFile-0) = 2bfff243382100b41a3e36eaa4f98b7ec
2 MD5 (binFile-1) = 2bfff243382100b41a3e36eaa4f98b7ec
3 MD5 (binFile-2) = 2bfff243382100b41a3e36eaa4f98b7ec
4 MD5 (binFile-3) = 2bfff243382100b41a3e36eaa4f98b7ec
5 MD5 (binFile-4) = 2bfff243382100b41a3e36eaa4f98b7ec
6 MD5 (binFile-5) = 2bfff243382100b41a3e36eaa4f98b7ec
7 MD5 (binFile-6) = 2bfff243382100b41a3e36eaa4f98b7ec
8 MD5 (binFile-7) = 2bfff243382100b41a3e36eaa4f98b7ec
9 MD5 (binFile-8) = 2bfff243382100b41a3e36eaa4f98b7ec
10 MD5 (binFile-9) = 2bfff243382100b41a3e36eaa4f98b7ec
11 MD5 (binFile-10) = 2bfff243382100b41a3e36eaa4f98b7ec
12 MD5 (binFile-11) = 2bfff243382100b41a3e36eaa4f98b7ec
13 MD5 (binFile-12) = 2bfff243382100b41a3e36eaa4f98b7ec
14 MD5 (binFile-13) = 2bfff243382100b41a3e36eaa4f98b7ec
15 MD5 (binFile-14) = 2bfff243382100b41a3e36eaa4f98b7ec
16 MD5 (binFile-15) = 2bfff243382100b41a3e36eaa4f98b7ec
17 MD5 (binFile-16) = 2bfff243382100b41a3e36eaa4f98b7ec
18 MD5 (binFile-17) = 2bfff243382100b41a3e36eaa4f98b7ec
19 MD5 (binFile-18) = 2bfff243382100b41a3e36eaa4f98b7ec
20 MD5 (binFile-19) = 2bfff243382100b41a3e36eaa4f98b7ec
21 MD5 (binFile-20) = 2bfff243382100b41a3e36eaa4f98b7ec
22 MD5 (binFile-21) = 2bfff243382100b41a3e36eaa4f98b7ec
23 MD5 (binFile-22) = 2bfff243382100b41a3e36eaa4f98b7ec
24 MD5 (binFile-23) = 2bfff243382100b41a3e36eaa4f98b7ec
25 MD5 (binFile-24) = 2bfff243382100b41a3e36eaa4f98b7ec
26 MD5 (binFile-25) = 2bfff243382100b41a3e36eaa4f98b7ec
27 MD5 (binFile-26) = 2bfff243382100b41a3e36eaa4f98b7ec
28 MD5 (binFile-27) = 2bfff243382100b41a3e36eaa4f98b7ec
29 MD5 (binFile-28) = 2bfff243382100b41a3e36eaa4f98b7ec
30 MD5 (binFile-29) = 2bfff243382100b41a3e36eaa4f98b7ec
31 MD5 (binFile-30) = 2bfff243382100b41a3e36eaa4f98b7ec
32 MD5 (binFile-31) = 2bfff243382100b41a3e36eaa4f98b7ec
33 MD5 (binFile-32) = 2bfff243382100b41a3e36eaa4f98b7ec
34 MD5 (binFile-33) = 2bfff243382100b41a3e36eaa4f98b7ec
35 MD5 (binFile-34) = 2bfff243382100b41a3e36eaa4f98b7ec
36 MD5 (binFile-35) = 2bfff243382100b41a3e36eaa4f98b7ec
37 MD5 (binFile-36) = 2bfff243382100b41a3e36eaa4f98b7ec
38 MD5 (binFile-37) = 2bfff243382100b41a3e36eaa4f98b7ec
```

From Server Terminal after extracted files:

```
root@sauronhsm:~# ls -al
total 2332
drwx----- 6 root root 4096 Jun  2 05:37 .
drwxr-xr-x 19 root root 4096 May 23 22:02 ..
-rw----- 1 root root 551 Jun  2 05:24 .bash_history
-rw-r--r-- 1 root root 3106 Oct 15 2021 .bashrc
drwx----- 3 root root 4096 May 23 22:08 .cache
-rw-r--r-- 1 root root 62417 May 23 22:58 .keys.xxx
-rw-r--r-- 1 root root 161 Jul  9 2019 .profile
drwx----- 2 root root 4096 May 23 22:02 .ssh
drwxr-xr-x 3 root root 4096 Jun  2 05:37 __MACOSX
drwx----- 2 root root 65536 May 23 16:55 files
-rw-r--r-- 1 root root 2123721 May 23 16:58 files.xxx
-rw-r--r-- 1 root root 65246 Jun  2 05:27 files_cp.xxx
-rw-r--r-- 1 root root 27127 May 23 22:58 hsm.docx
root@sauronhsm:~#
```

```
Sauron_HSM 1 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
binFile-1222 binFile-1480 binFile-1738 binFile-1996 binFile-453 binFile-710 binFile-969
binFile-1223 binFile-1481 binFile-1739 binFile-1997 binFile-454 binFile-711 binFile-97
binFile-1224 binFile-1482 binFile-174 binFile-1998 binFile-455 binFile-712 binFile-970
binFile-1225 binFile-1483 binFile-1740 binFile-1999 binFile-456 binFile-713 binFile-971
binFile-1226 binFile-1484 binFile-1741 binFile-2 binFile-457 binFile-714 binFile-972
binFile-1227 binFile-1485 binFile-1742 binFile-20 binFile-458 binFile-715 binFile-973
binFile-1228 binFile-1486 binFile-1743 binFile-200 binFile-459 binFile-716 binFile-974
binFile-1229 binFile-1487 binFile-1744 binFile-201 binFile-46 binFile-717 binFile-975
binFile-123 binFile-1488 binFile-1745 binFile-202 binFile-460 binFile-718 binFile-976
binFile-1230 binFile-1489 binFile-1746 binFile-203 binFile-461 binFile-719 binFile-977
binFile-1231 binFile-149 binFile-1747 binFile-204 binFile-462 binFile-72 binFile-978
binFile-1232 binFile-1490 binFile-1748 binFile-205 binFile-463 binFile-720 binFile-979
binFile-1233 binFile-1491 binFile-1749 binFile-206 binFile-464 binFile-721 binFile-98
binFile-1234 binFile-1492 binFile-175 binFile-207 binFile-465 binFile-722 binFile-980
binFile-1235 binFile-1493 binFile-1750 binFile-208 binFile-466 binFile-723 binFile-981
binFile-1236 binFile-1494 binFile-1751 binFile-209 binFile-467 binFile-724 binFile-982
binFile-1237 binFile-1495 binFile-1752 binFile-21 binFile-468 binFile-725 binFile-983
binFile-1238 binFile-1496 binFile-1753 binFile-210 binFile-469 binFile-726 binFile-984
binFile-1239 binFile-1497 binFile-1754 binFile-211 binFile-47 binFile-727 binFile-985
binFile-124 binFile-1498 binFile-1755 binFile-212 binFile-470 binFile-728 binFile-986
binFile-1240 binFile-1499 binFile-1756 binFile-213 binFile-471 binFile-729 binFile-987
binFile-1241 binFile-15 binFile-1757 binFile-214 binFile-472 binFile-73 binFile-988
binFile-1242 binFile-150 binFile-1758 binFile-215 binFile-473 binFile-730 binFile-989
binFile-1243 binFile-1500 binFile-1759 binFile-216 binFile-474 binFile-731 binFile-99
binFile-1244 binFile-1501 binFile-176 binFile-217 binFile-475 binFile-732 binFile-990
binFile-1245 binFile-1502 binFile-1760 binFile-218 binFile-476 binFile-733 binFile-991
binFile-1246 binFile-1503 binFile-1761 binFile-219 binFile-477 binFile-734 binFile-992
binFile-1247 binFile-1504 binFile-1762 binFile-22 binFile-478 binFile-735 binFile-993
binFile-1248 binFile-1505 binFile-1763 binFile-220 binFile-479 binFile-736 binFile-994
binFile-1249 binFile-1506 binFile-1764 binFile-221 binFile-48 binFile-737 binFile-995
binFile-125 binFile-1507 binFile-1765 binFile-222 binFile-480 binFile-738 binFile-996
binFile-1250 binFile-1508 binFile-1766 binFile-223 binFile-481 binFile-739 binFile-997
binFile-1251 binFile-1509 binFile-1767 binFile-224 binFile-482 binFile-74 binFile-998
binFile-1252 binFile-151 binFile-1768 binFile-225 binFile-483 binFile-740 binFile-999
binFile-1253 binFile-1510 binFile-1769 binFile-226 binFile-484 binFile-741 run_md5.sh
binFile-1254 binFile-1511 binFile-177 binFile-227 binFile-485 binFile-742 test.txt
root@sauronhsm:~/files#
```

MORDOR6 FOUND

```
MD5 (binFile-688) = 2bfff243382100b41a3e36eaa4f98b7ec
MD5 (binFile-689) = 2bfff243382100b41a3e36eaa4f98b7ec
MD5 (binFile-690) = 2bfff243382100b41a3e36eaa4f98b7ec
MD5 (binFile-691) = 2bfff243382100b41a3e36eaa4f98b7ec
MD5 (binFile-692) = 2bfff243382100b41a3e36eaa4f98b7ec
MD5 (binFile-693) = 2bfff243382100b41a3e36eaa4f98b7ec
MD5 (binFile-694) = 2bfff243382100b41a3e36eaa4f98b7ec
MD5 (binFile-695) = 8a00be55a5661f84ac4bf8c5ff7e0ed9
MD5 (binFile-696) = 2bfff243382100b41a3e36eaa4f98b7ec
MD5 (binFile-697) = 2bfff243382100b41a3e36eaa4f98b7ec
f MD5 (binFile-698) = 2bfff243382100b41a3e36eaa4f98b7ec
MD5 (binFile-699) = 2bfff243382100b41a3e36eaa4f98b7ec
MD5 (binFile-700) = 2bfff243382100b41a3e36eaa4f98b7ec
MD5 (binFile-701) = 2bfff243382100b41a3e36eaa4f98b7ec
```



```
root@sauronhsm:~/files# cat binFile-695
gP***5Na9*0*x***X*a***>*0{nZ**\|*_4u:**号WK*+* 1****h*~*p!*!***4
FW?N**{t***o**,***
****4*_W*****V
C*
0***~
#P***
":*(**ZH*gT*g*#*ÆQyEv***D*-*c****r*6%***v*Y_****Bq****[vY**X*+p*_**55U*h*\**
*~$*~*x*j\?***<*\c*m@FĊ***C5C*^x0|*k;r*db***s*m*+gKJ$*u*R*****|**^
*\**:****_***<
ia***-`*Q*v*~}*F** R*root@sauronhsm:~/files# strings binFile-695
MORDOR6///_
YJ|'j
Q|~r3
gKJ$
root@sauronhsm:~/files#
```