Cybersecurity

Module 10 Challenge Submission File

Cryptography Challenge: Ransomware Riddles

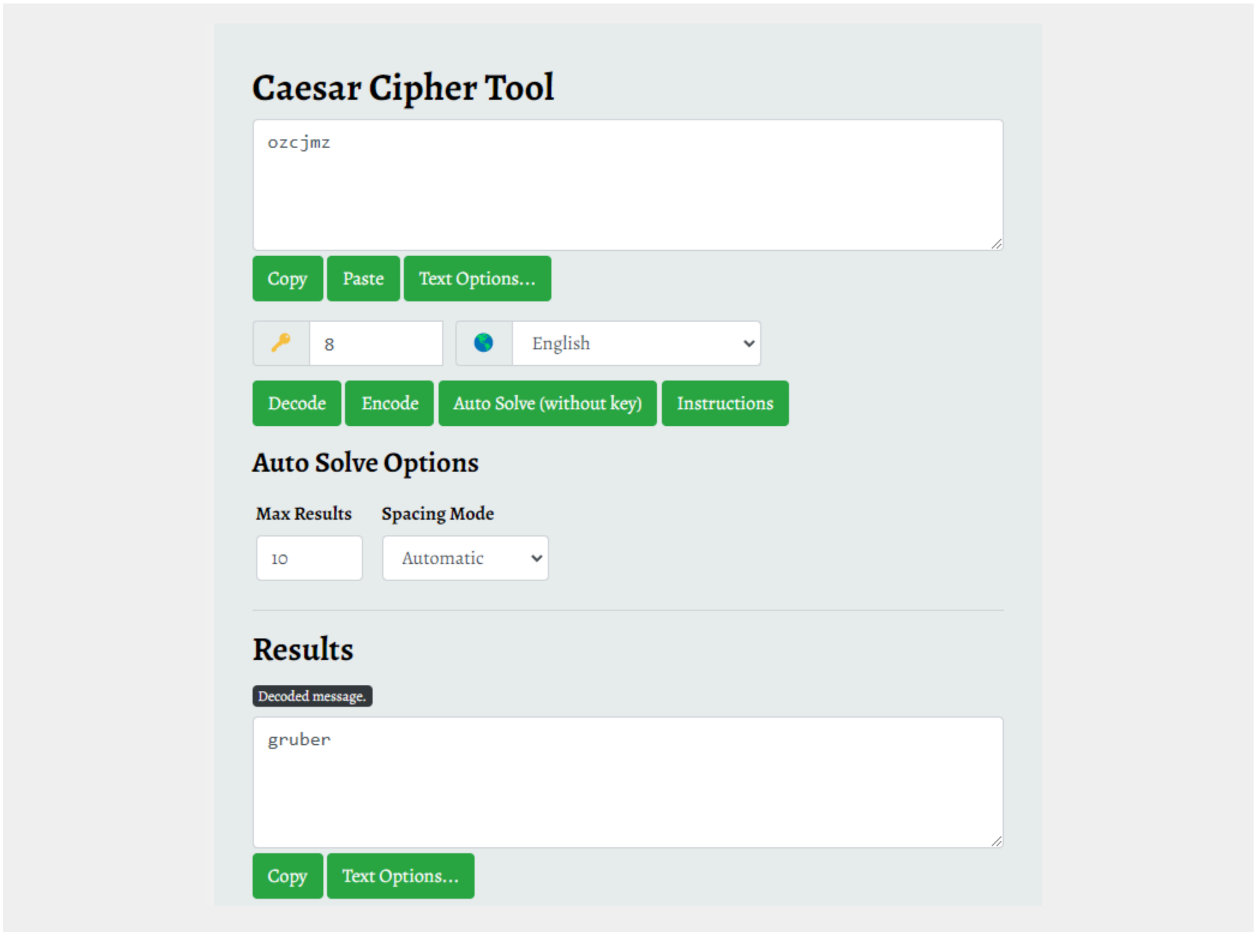
Make a copy of this document to work in, and then for each mission, add the solution below the prompt. Save and submit this completed file as your Challenge deliverable.

Ransomware Riddles

1. Riddle 1:

ozcjmz

```
Result = Gruber

Key = 6skd8s
```



```
ozcjmz
```

Key: 8
Language: English

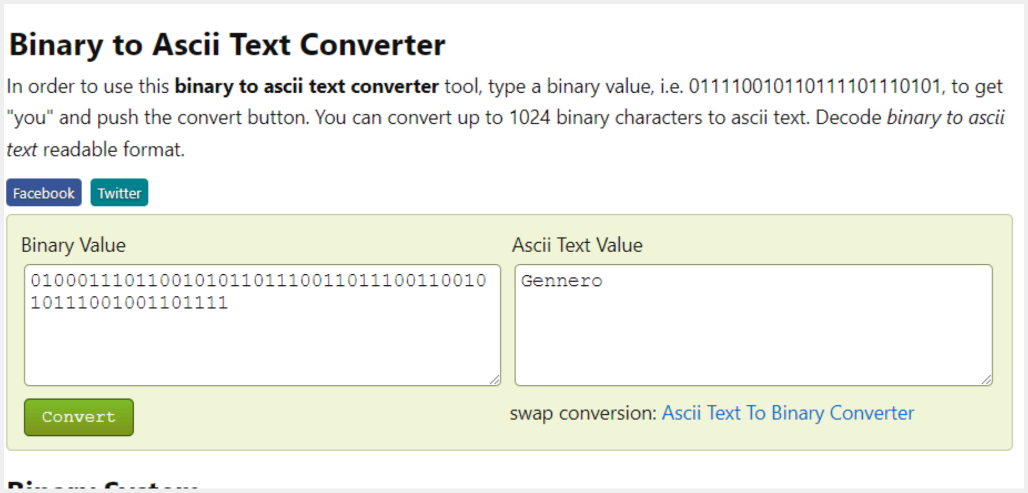
Max Results: 10
Spacing Mode: Automatic

Decoded message.

```
gruber
```

2. Riddle 2:

```
01000111 01100101 01101110
01101110 01100101 01110010
01101111

Result = Gennero

Key = cy8snd2
```


Binary to Ascii Text Converter

In order to use this **binary to ascii text converter** tool, type a binary value, i.e. 011110010110111101110101, to get "you" and push the convert button. You can convert up to 1024 binary characters to ascii text. Decode *binary to ascii text* readable format.

```
01000111011001010110111001101110011001010111001001101111
```

```
Gennero
```

swap conversion: Ascii Text To Binary Converter

3. Riddle 3:

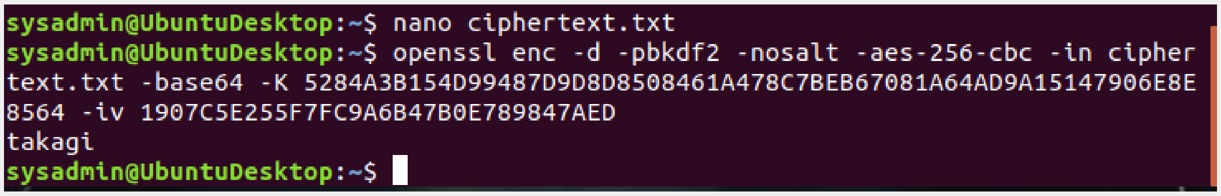
openssl enc -d -pbkdf2 -nosalt -aes-256-cbc -in ciphertext.txt -base64 -K 5284A3B154D99487D9D8D8508461A478C7BEB67081A64AD9A15147906E8E8564 -iv 1907C5E255F7FC9A6B47B0E789847AED

Cipher.txt = 4qMOIvwEGXzvkMvRE2bNbg==

Put the == as base64 encoding padding
[Why does base64 encoding require padding if the input length is not divisible by 3? - Stack Overflow]()

Result = takagi

Key = ud6s98n



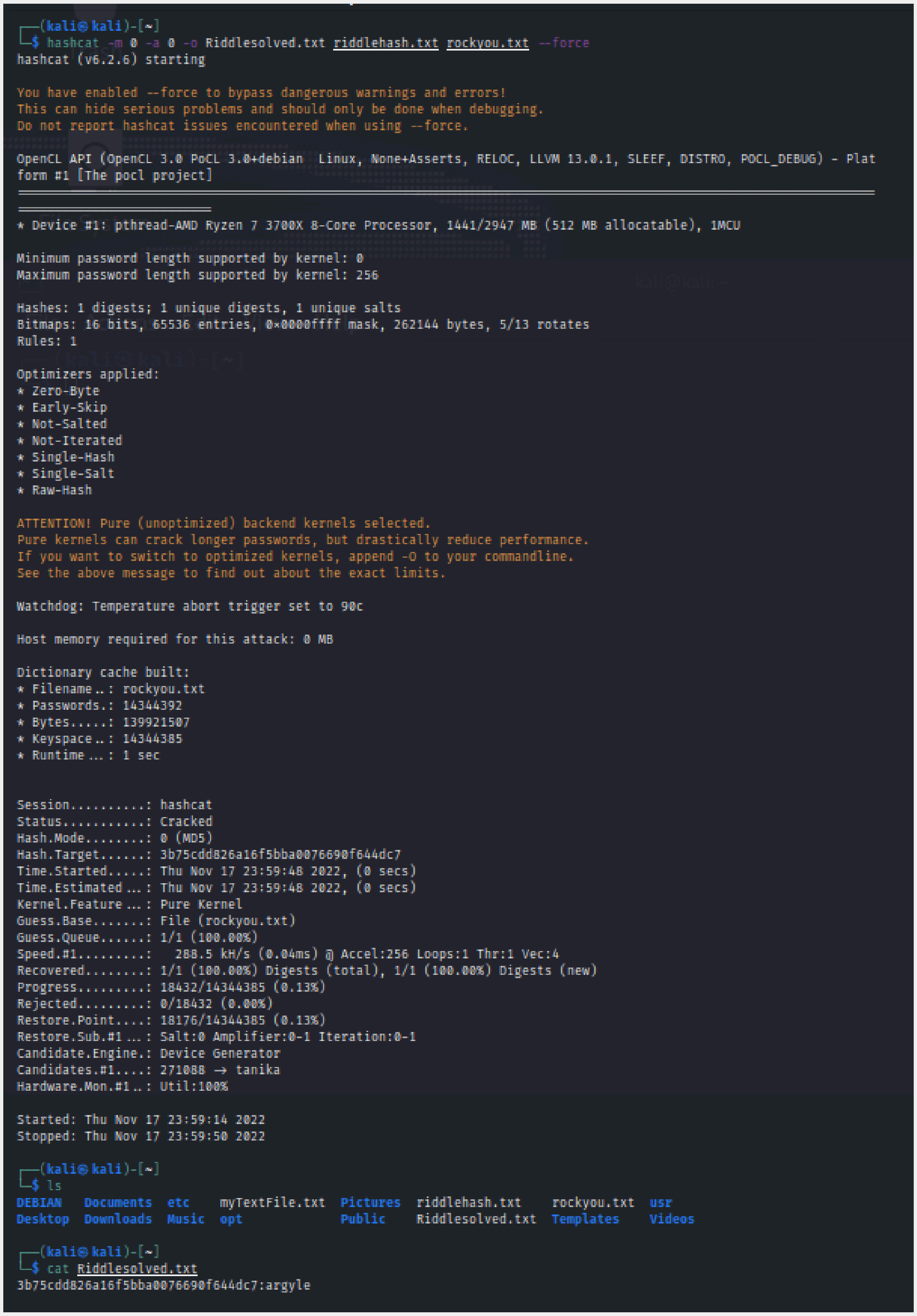
```
sysadmin@UbuntuDesktop:~$ nano ciphertext.txt
sysadmin@UbuntuDesktop:~$ openssl enc -d -pbkdf2 -nosalt -aes-256-cbc -in cipher
text.txt -base64 -K 5284A3B154D99487D9D8D8508461A478C7BEB67081A64AD9A15147906E8E
8564 -iv 1907C5E255F7FC9A6B47B0E789847AED
takagi
sysadmin@UbuntuDesktop:~$ 
```

4. Riddle 4:

```
Result:
   -  Jill’s Public key
   -  Jill’s Private key
   -  12 Asymmetric and 15 symmetric
   -  Alice’s Public key

Key = 7gsn3nd2
```

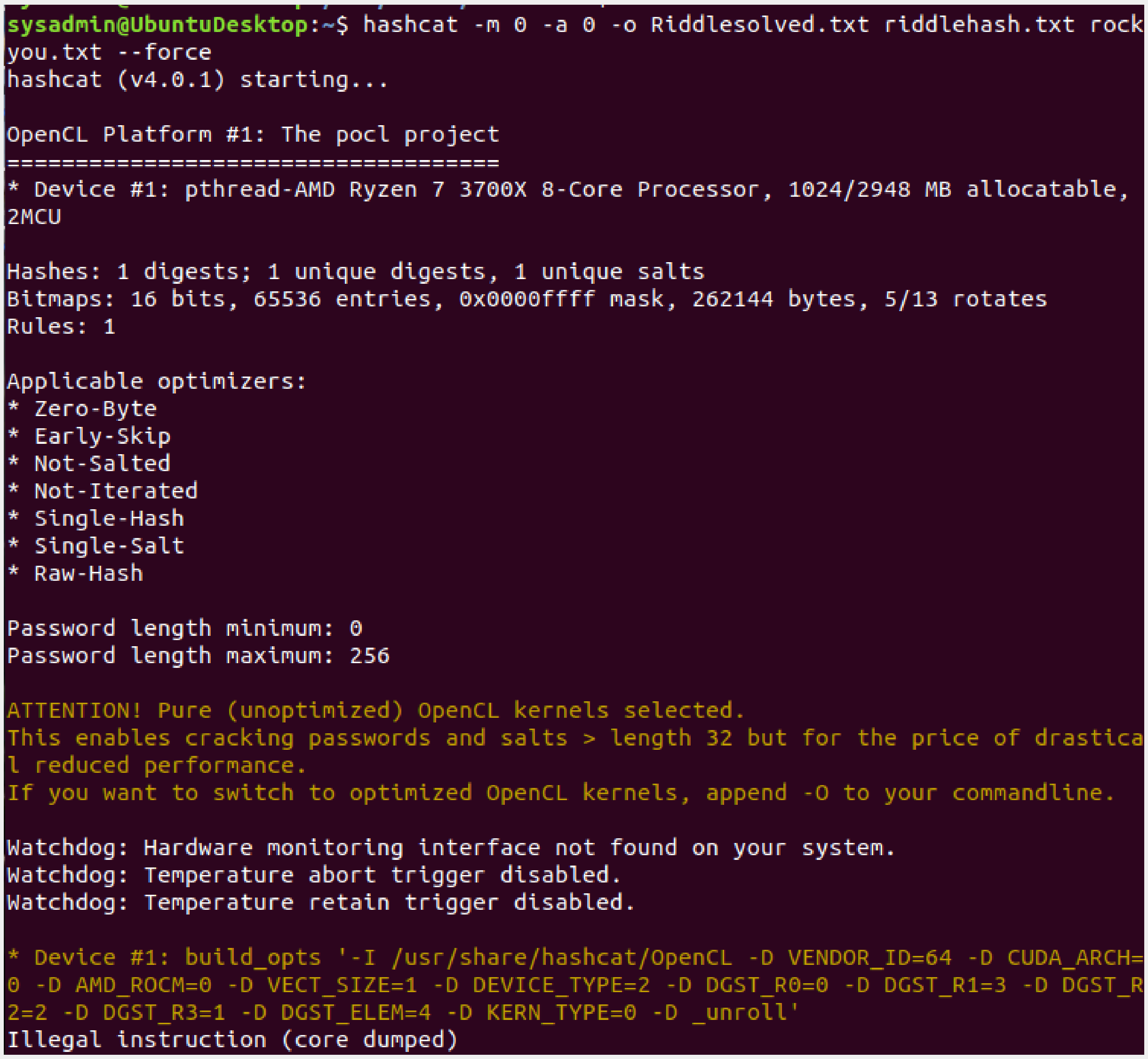
5. Riddle 5:

```
riddlehash.txt = 3b75cdd826a16f5bba0076690f644dc7

Hashcat -m 0 -a 0 -o Riddlesolved.txt riddlehash.txt rockyou.txt –force

Result = argyle

Key = ajy39d2
```



```
┌──(kali㉿kali)-[~]
└─$ hashcat -m 0 -a 0 -o Riddlesolved.txt riddlehash.txt rockyou.txt --force
hashcat (v6.2.6) starting

You have enabled --force to bypass dangerous warnings and errors!
This can hide serious problems and should only be done when debugging.
Do not report hashcat issues encountered when using --force.

OpenCL API (OpenCL 3.0 PoCL 3.0+debian  Linux, None+Asserts, RELOC, LLVM 13.0.1, SLEEF, DISTRO, POCL_DEBUG) - Plat
form #1 [The pocl project]
==================================================================================================================
==========================
* Device #1: pthread-AMD Ryzen 7 3700X 8-Core Processor, 1441/2947 MB (512 MB allocatable), 1MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Optimizers applied:
* Zero-Byte
* Early-Skip
* Not-Salted
* Not-Iterated
* Single-Hash
* Single-Salt
* Raw-Hash

ATTENTION! Pure (unoptimized) backend kernels selected.
Pure kernels can crack longer passwords, but drastically reduce performance.
If you want to switch to optimized kernels, append -O to your commandline.
See the above message to find out about the exact limits.

Watchdog: Temperature abort trigger set to 90c

Host memory required for this attack: 0 MB

Dictionary cache built:
* Filename..: rockyou.txt
* Passwords.: 14344392
* Bytes.....: 139921507
* Keyspace..: 14344385
* Runtime...: 1 sec


Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 0 (MD5)
Hash.Target......: 3b75cdd826a16f5bba0076690f644dc7
Time.Started.....: Thu Nov 17 23:59:48 2022, (0 secs)
Time.Estimated...: Thu Nov 17 23:59:48 2022, (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:   288.5 kH/s (0.04ms) @ Accel:256 Loops:1 Thr:1 Vec:4
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.........: 18432/14344385 (0.13%)
Rejected.........: 0/18432 (0.00%)
Restore.Point....: 18176/14344385 (0.13%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#1....: 271088 -> tanika
Hardware.Mon.#1..: Util:100%

Started: Thu Nov 17 23:59:14 2022
Stopped: Thu Nov 17 23:59:50 2022

┌──(kali㉿kali)-[~]
└─$ ls
DEBIAN   Documents  etc    myTextFile.txt  Pictures  riddlehash.txt    rockyou.txt  usr
Desktop  Downloads  Music  opt             Public    Riddlesolved.txt  Templates    Videos

┌──(kali㉿kali)-[~]
└─$ cat Riddlesolved.txt
3b75cdd826a16f5bba0076690f644dc7:argyle
```

Had to use my kali linux instead of vagrant because i got an error:

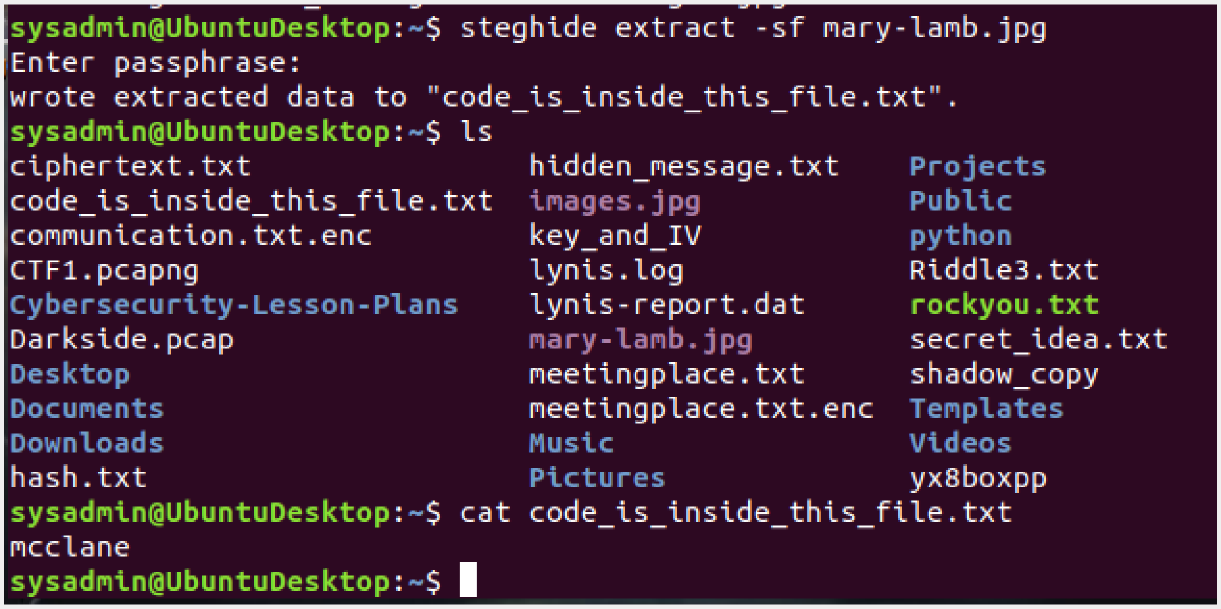


```
sysadmin@UbuntuDesktop:~$ hashcat -m 0 -a 0 -o Riddlesolved.txt riddlehash.txt rock
you.txt --force
hashcat (v4.0.1) starting...

OpenCL Platform #1: The pocl project
====================================
* Device #1: pthread-AMD Ryzen 7 3700X 8-Core Processor, 1024/2948 MB allocatable,
2MCU

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Applicable optimizers:
* Zero-Byte
* Early-Skip
* Not-Salted
* Not-Iterated
* Single-Hash
* Single-Salt
* Raw-Hash

Password length minimum: 0
Password length maximum: 256

ATTENTION! Pure (unoptimized) OpenCL kernels selected.
This enables cracking passwords and salts > length 32 but for the price of drastica
l reduced performance.
If you want to switch to optimized OpenCL kernels, append -O to your commandline.

Watchdog: Hardware monitoring interface not found on your system.
Watchdog: Temperature abort trigger disabled.
Watchdog: Temperature retain trigger disabled.

* Device #1: build_opts '-I /usr/share/hashcat/OpenCL -D VENDOR_ID=64 -D CUDA_ARCH=
0 -D AMD_ROCM=0 -D VECT_SIZE=1 -D DEVICE_TYPE=2 -D DGST_R0=0 -D DGST_R1=3 -D DGST_R
2=2 -D DGST_R3=1 -D DGST_ELEM=4 -D KERN_TYPE=0 -D _unroll'
Illegal instruction (core dumped)
```

6. Riddle 6:

```
Passphrase = ABC

Result:
    -   Steghide extract -sf mary-lamb.jpg
    -   Passphrase = ABC
    -   Ls
    -   Cat code_is_inside_this_file.txt

Code = mcclane
```

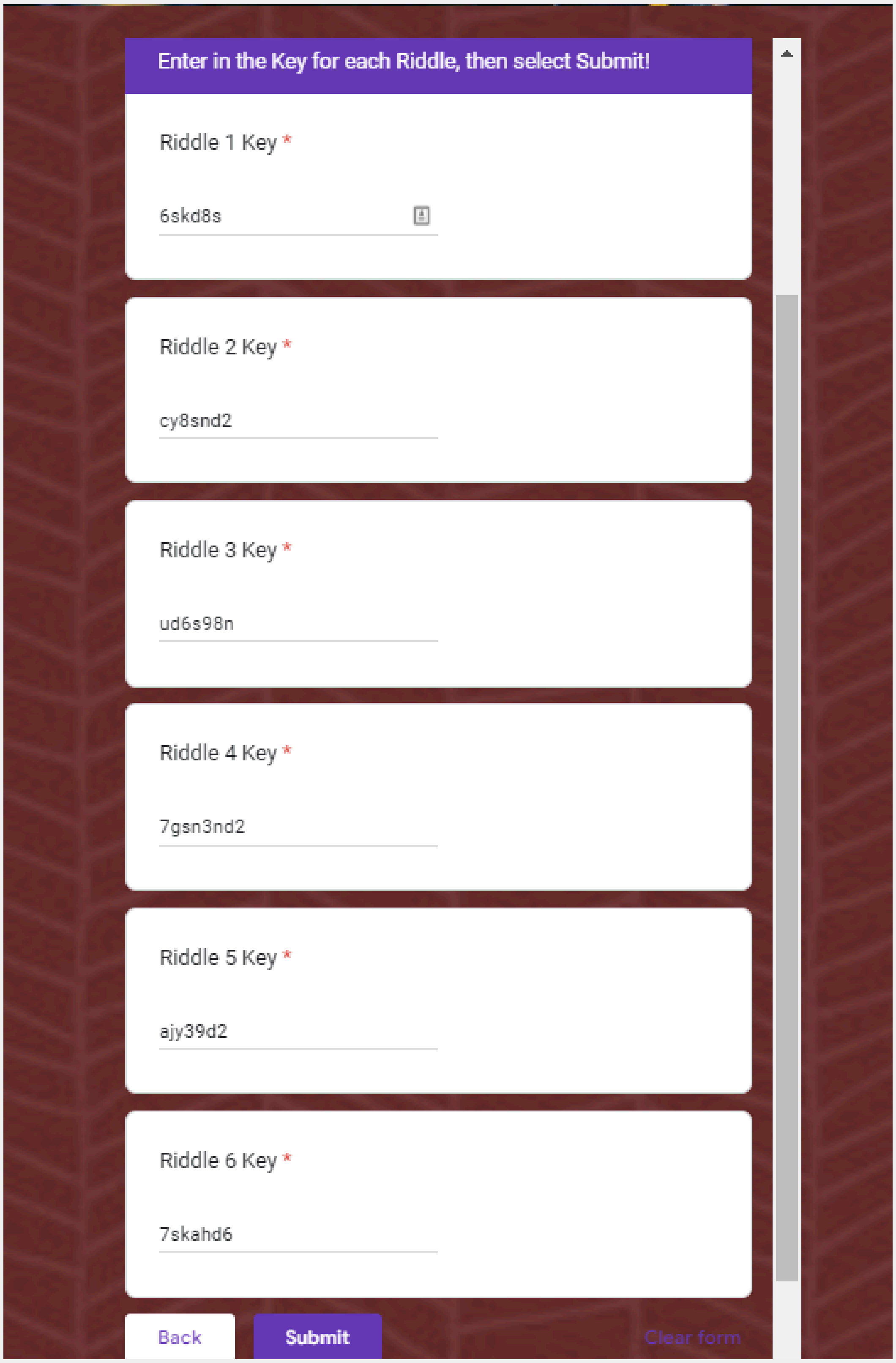
```
Key = 7skahd6
```



```
sysadmin@UbuntuDesktop:~$ steghide extract -sf mary-lamb.jpg
Enter passphrase:
wrote extracted data to "code_is_inside_this_file.txt".
sysadmin@UbuntuDesktop:~$ ls
ciphertext.txt                hidden_message.txt    Projects
code_is_inside_this_file.txt  images.jpg            Public
communication.txt.enc         key_and_IV            python
CTF1.pcapng                   lynis.log             Riddle3.txt
Cybersecurity-Lesson-Plans    lynis-report.dat      rockyou.txt
Darkside.pcap                 mary-lamb.jpg         secret_idea.txt
Desktop                       meetingplace.txt      shadow_copy
Documents                     meetingplace.txt.enc  Templates
Downloads                     Music                 Videos
hash.txt                      Pictures              yx8boxpp
sysadmin@UbuntuDesktop:~$ cat code_is_inside_this_file.txt
mcclane
sysadmin@UbuntuDesktop:~$
```

Ransomware Screenshot

1. Include a screenshot showing proof that you've decrypted the ransomware:


Enter in the Key for each Riddle, then select Submit!

Riddle 1 Key *

6skd8s

Riddle 2 Key *

cy8snd2

Riddle 3 Key *

ud6s98n

Riddle 4 Key *

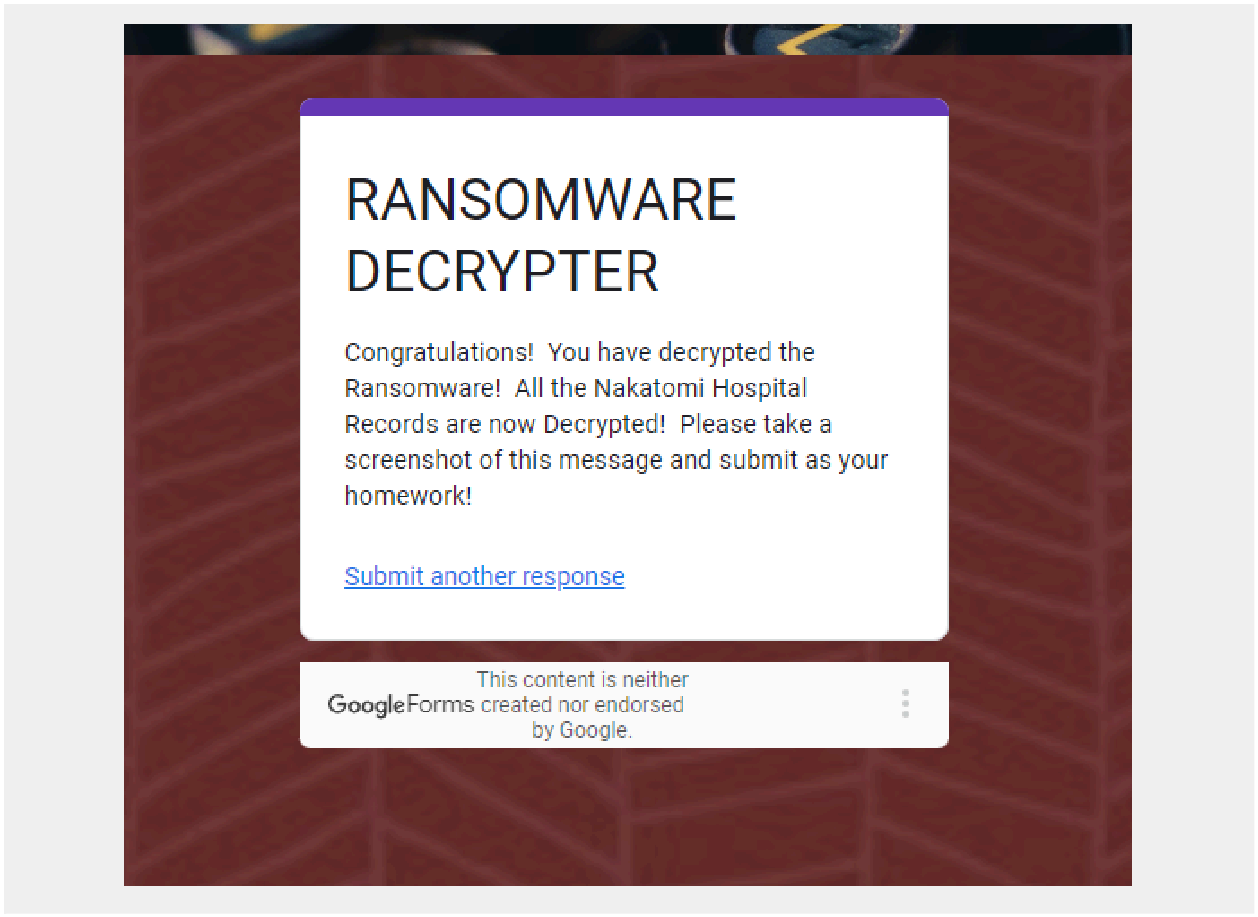
7gsn3nd2

Riddle 5 Key *

ajy39d2

Riddle 6 Key *

7skahd6



RANSOMWARE DECRYPTER

Congratulations! You have decrypted the Ransomware! All the Nakatomi Hospital Records are now Decrypted! Please take a screenshot of this message and submit as your homework!

Submit another response