Encryption - Crypto 101

An introduction to encryption, as part of a series on crypto

Task 2 Key terms

Question	Answer
Are SSH keys protected with a passphrase or a password?	passphrase

Task 3 Why is Encryption important?

Question	Answer
What does SSH stand for?	Secure Shell
How do webservers prove their identity?	certificates
What is the main set of standards you need to comply with if you store or process payment card details?	PCI-DSS

Task 4 Crucial Crypto Maths

What's 30 % 5?
Answer: 0

- ➔ 30 is divisible by 5, so there is no remainder.

What's 25 % 7

Answer: 4

- ➔ When 25 is divided by 7, the quotient is 3 with a remainder of 4.

What's 118613842 % 9091

Answer: 3565

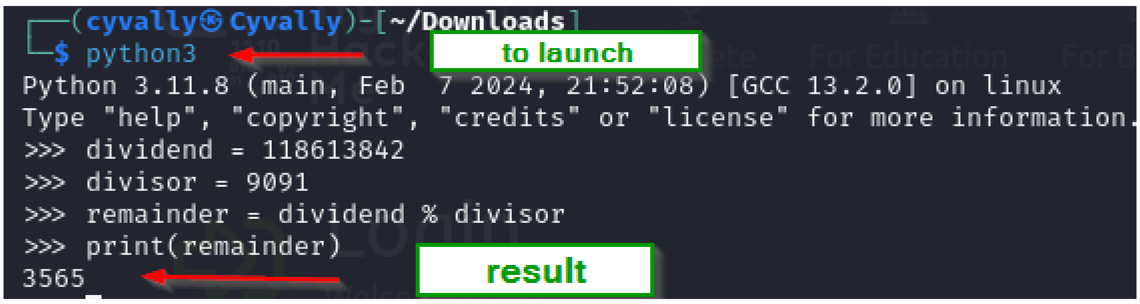
Hint: Use Python.

➔ I have python3 already installed in my local machine, so i launched it

Command: python3

➔ Then i ran the code to calculate the question

dividend = 118613842
>>> divisor = 9091
remainder = dividend % divisor
print(remainder)



```
┌──(cyvally㉿Cyvally)-[~/Downloads]
└─$ python3
Python 3.11.8 (main, Feb  7 2024, 21:52:08) [GCC 13.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> dividend = 118613842
>>> divisor = 9091
>>> remainder = dividend % divisor
>>> print(remainder)
3565
```

Task 5 Types of Encryption

Question	Answer
Should you trust DES? Yea/Nay	Nay
What was the result of the attempt to make DES more secure so that it could be used for longer?	Triple DES
Is it ok to share your public key? Yea/Nay	Yea

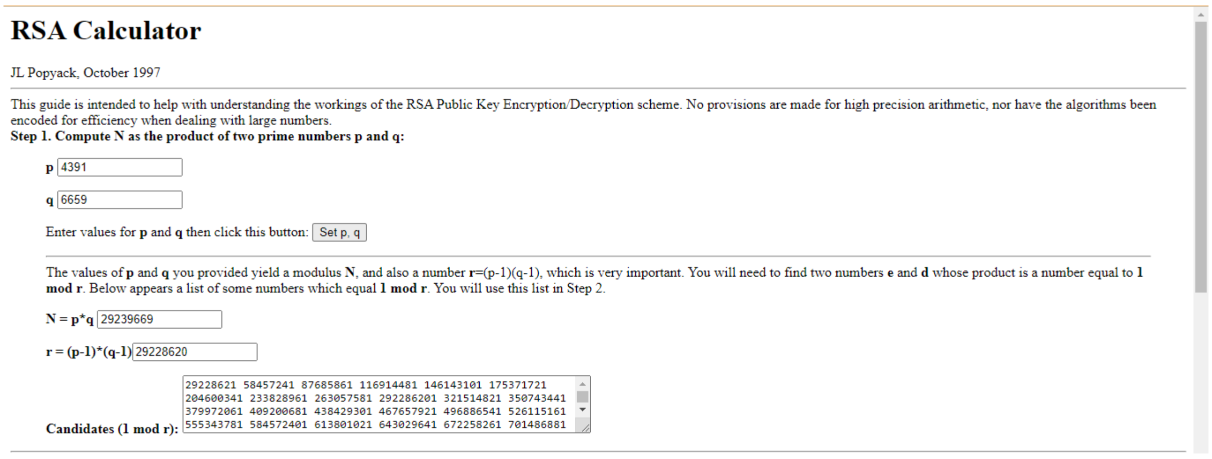
Task 6 RSA - Rivest Shamir Adleman

p = 4391, q = 6659. What is n?

Answer: 29239669

- I used RSA Calculator

- Then i entered the value of p and q



RSA Calculator

JL Popyack, October 1997

This guide is intended to help with understanding the workings of the RSA Public Key Encryption/Decryption scheme. No provisions are made for high precision arithmetic, nor have the algorithms been encoded for efficiency when dealing with large numbers.
Step 1. Compute N as the product of two prime numbers p and q:

p 4391

q 6659

Enter values for **p** and **q** then click this button: Set p, q

The values of **p** and **q** you provided yield a modulus **N**, and also a number **r**=(p-1)(q-1), which is very important. You will need to find two numbers **e** and **d** whose product is a number equal to **1 mod r**. Below appears a list of some numbers which equal **1 mod r**. You will use this list in Step 2.

N = p*q 29239669

r = (p-1)*(q-1) 29228620

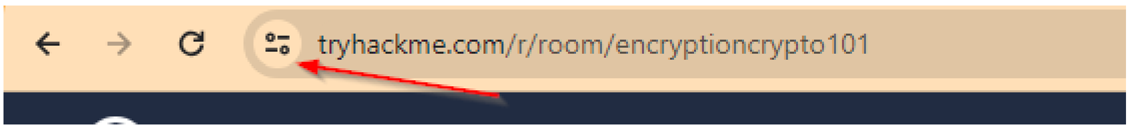
Candidates (1 mod r):
```
29228621 58457241 87685861 116914481 146143101 175371721
204600341 233828961 263057581 292286201 321514821 350743441
379972061 409200681 438429301 467657921 496886541 526115161
555343781 584572401 613801021 643029641 672258261 701486881
```

Task 8 Digital signatures and Certificates

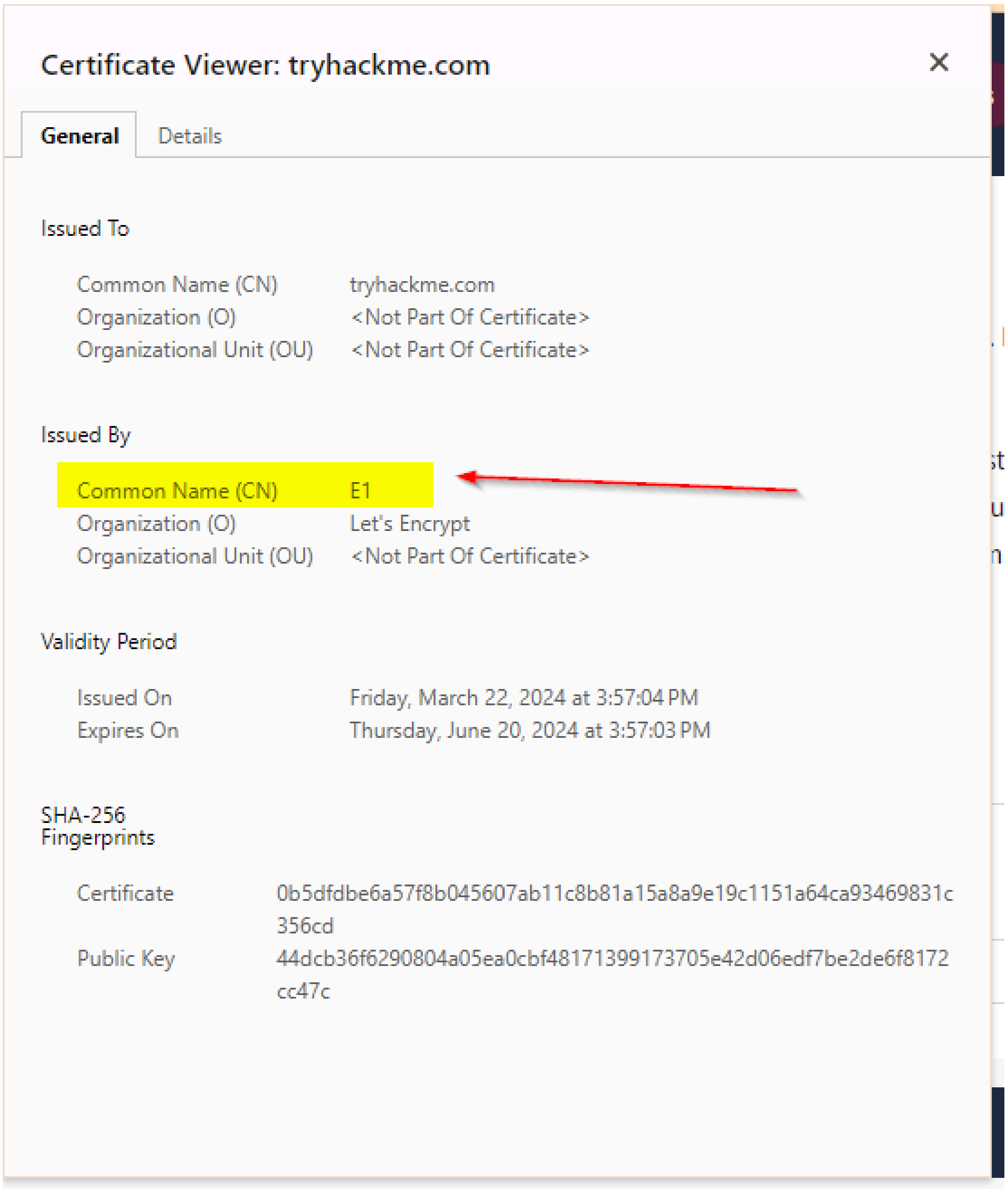
Who is TryHackMe's HTTPS certificate issued by?

Answer: E1

- I Clicked on the view site information button



- Then i clicked on “connection is secure”
- Then to “certificates is valid” to show certificates



Certificate Viewer: tryhackme.com

General | Details

Issued To

Common Name (CN)	tryhackme.com
Organization (O)	<Not Part Of Certificate>
Organizational Unit (OU)	<Not Part Of Certificate>

Issued By

Common Name (CN)	E1
Organization (O)	Let's Encrypt
Organizational Unit (OU)	<Not Part Of Certificate>

Validity Period

Issued On	Friday, March 22, 2024 at 3:57:04 PM
Expires On	Thursday, June 20, 2024 at 3:57:03 PM

SHA-256 Fingerprints

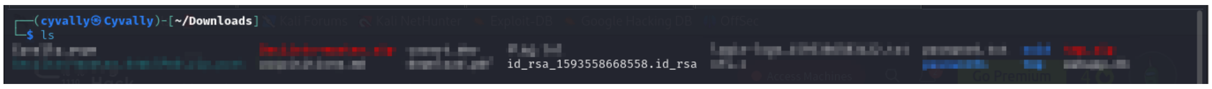
Certificate	0b5dfdbe6a57f8b045607ab11c8b81a15a8a9e19c1151a64ca93469831c356cd
Public Key	44dcb36f6290804a05ea0cbf48171399173705e42d06edf7be2de6f8172cc47c

Task 9 SSH Authentication

What algorithm does the key use?

Answer: **RSA**

➔ I downloaded the SSH Private Key attached



```
┌──(cyvally㉿Cyvally)-[~/Downloads]
└─$ ls
[redacted]
id_rsa_1593558668558.id_rsa
```

➔ I was able to know its RSA since the key was named id_rsa_1593558668558.id_rsa

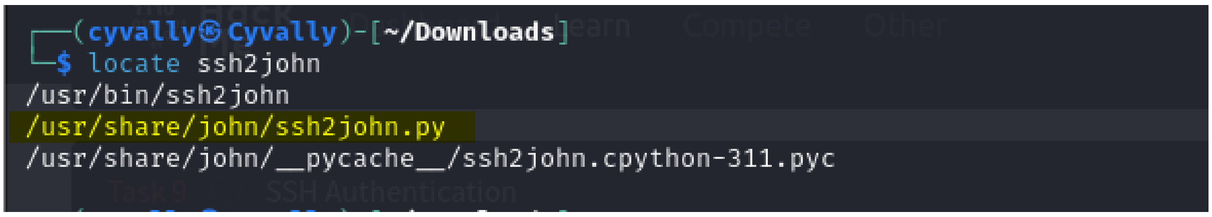
Crack the password with John The Ripper and rockyou, what's the passphrase for the key?

Answer: delicious

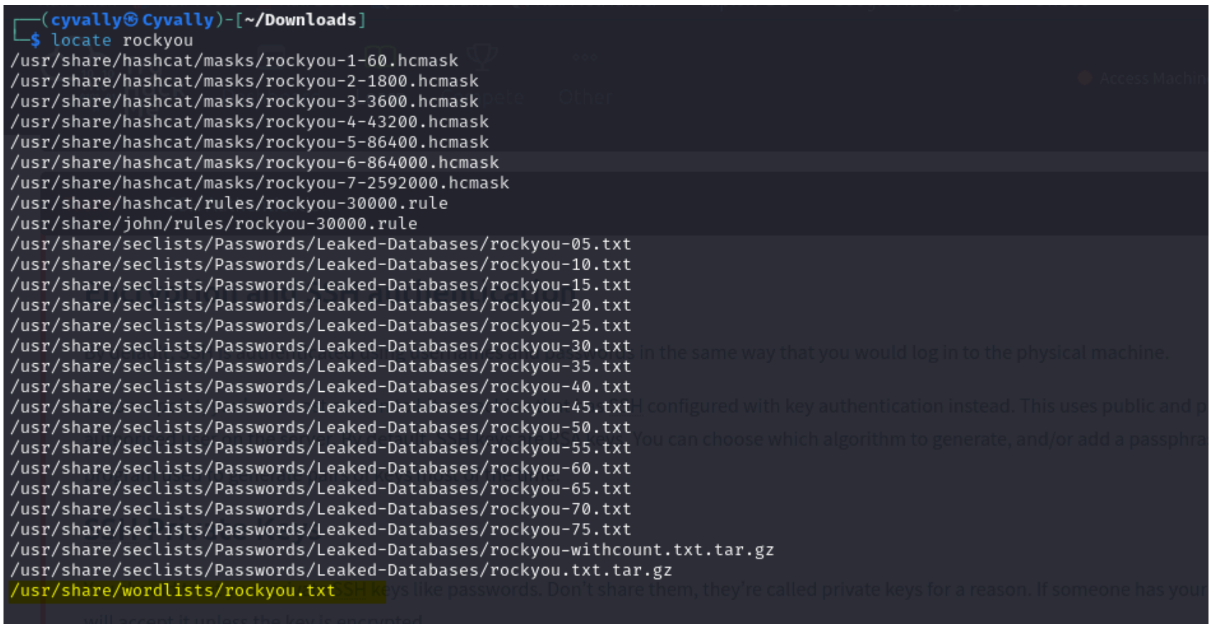
➔ First, i had to find where ssh2john and the wordlist rockyou is located

Command: locate ssh2john
Command: locate rockyou

➔ ssh2john is a tool for extracting password hashes from SSH private key files and converting them into a format that is compatible with John the Ripper. This helps to use John the Ripper to crack SSH private key passwords.



```
┌──(cyvally㉿Cyvally)-[~/Downloads]
└─$ locate ssh2john
/usr/bin/ssh2john
/usr/share/john/ssh2john.py
/usr/share/john/__pycache__/ssh2john.cpython-311.pyc
```



```
┌──(cyvally㉿Cyvally)-[~/Downloads]
└─$ locate rockyou
/usr/share/hashcat/masks/rockyou-1-60.hcmask
/usr/share/hashcat/masks/rockyou-2-1800.hcmask
/usr/share/hashcat/masks/rockyou-3-3600.hcmask
/usr/share/hashcat/masks/rockyou-4-43200.hcmask
/usr/share/hashcat/masks/rockyou-5-86400.hcmask
/usr/share/hashcat/masks/rockyou-6-864000.hcmask
/usr/share/hashcat/masks/rockyou-7-2592000.hcmask
/usr/share/hashcat/rules/rockyou-30000.rule
/usr/share/john/rules/rockyou-30000.rule
/usr/share/seclists/Passwords/Leaked-Databases/rockyou-05.txt
/usr/share/seclists/Passwords/Leaked-Databases/rockyou-10.txt
/usr/share/seclists/Passwords/Leaked-Databases/rockyou-15.txt
/usr/share/seclists/Passwords/Leaked-Databases/rockyou-20.txt
/usr/share/seclists/Passwords/Leaked-Databases/rockyou-25.txt
/usr/share/seclists/Passwords/Leaked-Databases/rockyou-30.txt
/usr/share/seclists/Passwords/Leaked-Databases/rockyou-35.txt
/usr/share/seclists/Passwords/Leaked-Databases/rockyou-40.txt
/usr/share/seclists/Passwords/Leaked-Databases/rockyou-45.txt
/usr/share/seclists/Passwords/Leaked-Databases/rockyou-50.txt
/usr/share/seclists/Passwords/Leaked-Databases/rockyou-55.txt
/usr/share/seclists/Passwords/Leaked-Databases/rockyou-60.txt
/usr/share/seclists/Passwords/Leaked-Databases/rockyou-65.txt
/usr/share/seclists/Passwords/Leaked-Databases/rockyou-70.txt
/usr/share/seclists/Passwords/Leaked-Databases/rockyou-75.txt
/usr/share/seclists/Passwords/Leaked-Databases/rockyou-withcount.txt.tar.gz
/usr/share/seclists/Passwords/Leaked-Databases/rockyou.txt.tar.gz
/usr/share/wordlists/rockyou.txt
```

➔ Then i extracted password hashes from SSH private key file and save it as sshkey.txt

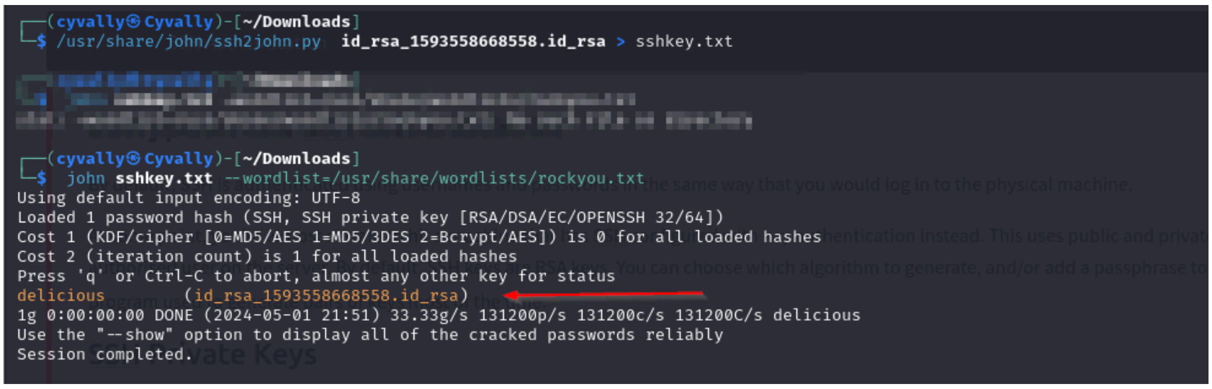
Command: /usr/share/john/ssh2john.py id_rsa_1593558668558.id_rsa > sshkey.txt

- ➔ I cracked the password using john the ripper

Command: john sshkey.txt –wordlist=/usr/share/wordlists/rockyou.txt

- ➔ Note: if you never used rockyou.txt file in linux before you have to unzip it. it located in /usr/share/wordlists/rockyou.txt.gz
- ➔ to unzip it

Command: gzip -d /usr/share/wordlists/rockyou.txt.gz



```
┌──(cyvally㉿Cyvally)-[~/Downloads]
└─$ /usr/share/john/ssh2john.py  id_rsa_1593558668558.id_rsa > sshkey.txt
[redacted]

┌──(cyvally㉿Cyvally)-[~/Downloads]
└─$ john sshkey.txt --wordlist=/usr/share/wordlists/rockyou.txt
Using default input encoding: UTF-8
Loaded 1 password hash (SSH, SSH private key [RSA/DSA/EC/OPENSSH 32/64])
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) is 0 for all loaded hashes
Cost 2 (iteration count) is 1 for all loaded hashes
Press 'q' or Ctrl-C to abort, almost any other key for status
delicious        (id_rsa_1593558668558.id_rsa)
1g 0:00:00:00 DONE (2024-05-01 21:51) 33.33g/s 131200p/s 131200c/s 131200C/s delicious
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

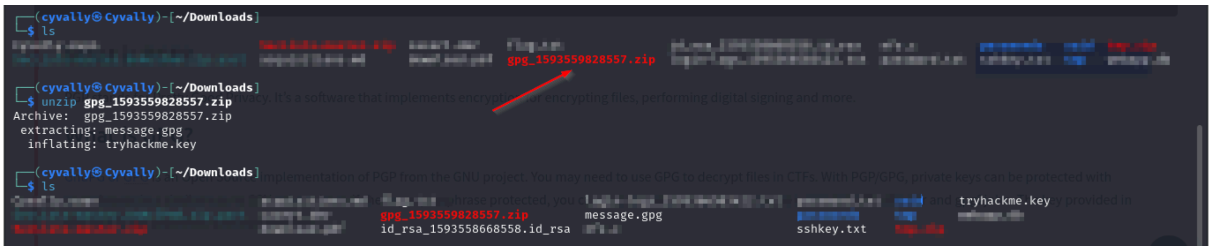
Task 11 PGP, GPG and AES

You have the private key, and a file encrypted with the public key. Decrypt the file. What's the secret word?

Answer: Pineapple

- ➔ I downloaded and unzipped the attached file and got the following

extracting: message.gpg
  inflating: tryhackme.key



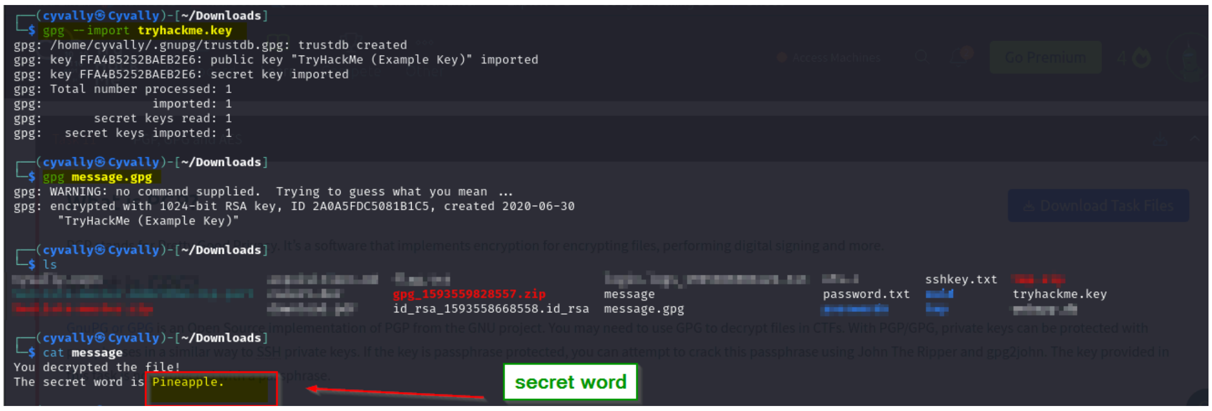
```
┌──(cyvally㉿Cyvally)-[~/Downloads]
└─$ ls
[redacted]                                       gpg_1593559828557.zip  [redacted]

┌──(cyvally㉿Cyvally)-[~/Downloads]
└─$ unzip gpg_1593559828557.zip
Archive:  gpg_1593559828557.zip
 extracting: message.gpg
  inflating: tryhackme.key

┌──(cyvally㉿Cyvally)-[~/Downloads]
└─$ ls
[redacted]                                                                   tryhackme.key
                          gpg_1593559828557.zip        message.gpg
                          id_rsa_1593558668558.id_rsa               sshkey.txt
```

➔ First, I imported the GPG (GNU Privacy Guard) key from the file named tryhackme.key into the local GPG keyring. This allows me to use the key for decryption

Command: gpg --import tryhackme.key

➔ Then i decrypted the file named message.gpg using GPG

Command: gpg message.gpg



```
┌──(cyvally㉿Cyvally)-[~/Downloads]
└─$ gpg --import tryhackme.key
gpg: /home/cyvally/.gnupg/trustdb.gpg: trustdb created
gpg: key FFA4B5252BAEB2E6: public key "TryHackMe (Example Key)" imported
gpg: key FFA4B5252BAEB2E6: secret key imported
gpg: Total number processed: 1
gpg:               imported: 1
gpg:       secret keys read: 1
gpg:   secret keys imported: 1

┌──(cyvally㉿Cyvally)-[~/Downloads]
└─$ gpg message.gpg
gpg: WARNING: no command supplied.  Trying to guess what you mean ...
gpg: encrypted with 1024-bit RSA key, ID 2A0A5FDC5081B1C5, created 2020-06-30
      "TryHackMe (Example Key)"

┌──(cyvally㉿Cyvally)-[~/Downloads]
└─$ ls
[redacted]              [redacted]  [redacted]                  [redacted]   [redacted]    sshkey.txt  [redacted]
[redacted]              [redacted]  gpg_1593559828557.zip       message      password.txt  [redacted]  tryhackme.key
[redacted]              [redacted]  id_rsa_1593558668558.id_rsa message.gpg  [redacted]    [redacted]  [redacted]

┌──(cyvally㉿Cyvally)-[~/Downloads]
└─$ cat message
You decrypted the file!
The secret word is Pineapple.
```

END!!!