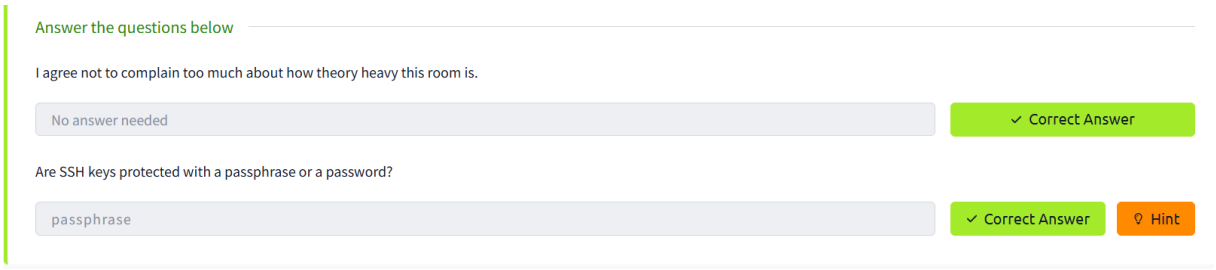
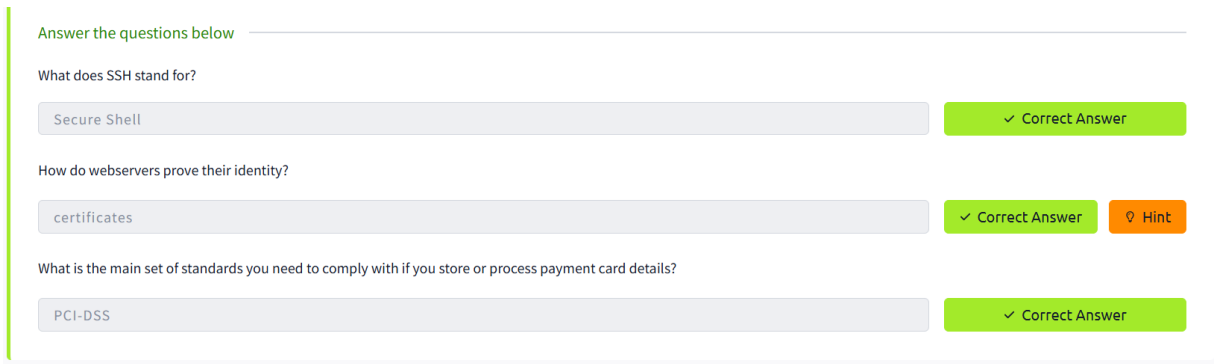
Sprawozdanie

Zadanie1

Zadanie 2.



Zadanie3.



Zadanie4.

Zadanie5.

Answer the questions below

Should you trust DES? Yea/Nay

Nay — ✓ Correct Answer — Hint

What was the result of the attempt to make DES more secure so that it could be used for longer?

Triple DES — ✓ Correct Answer — Hint

Is it ok to share your public key? Yea/Nay

Yea — ✓ Correct Answer

Zadanie6.

Answer the questions below

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

29239669 — ✓ Correct Answer — Hint

I understand enough about RSA to move on, and I know where to look to learn more if I want to.

No answer needed — ✓ Correct Answer

Zadanie8.

Answer the questions below

What can you use to verify that a file has not been modified and is the authentic file as the author intended?

Digital Signature — ✓ Correct Answer

Zadanie9.

Answer the questions below

Woop woop! Your answer is correct

I recommend giving this a go yourself. Deploy a VM, like Linux Fundamentals 2 and try to add an SSH key and log in with the private key.

No answer needed — ✓ Correct Answer — Hint

Download the SSH Private Key attached to this room.

No answer needed — ✓ Correct Answer

What algorithm does the key use?

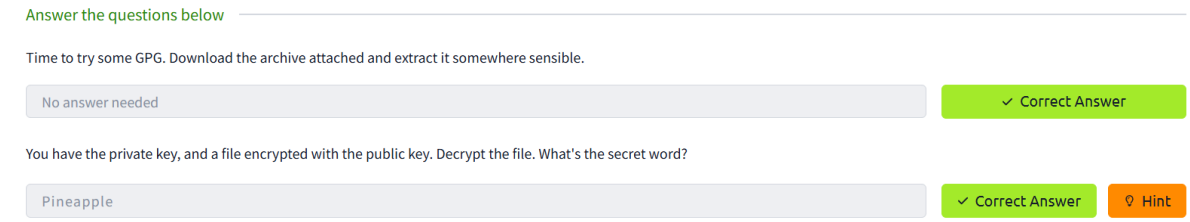
RSA — ✓ Correct Answer — Hint

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

delicious — ✓ Correct Answer — Hint

```
┌──(kali㉿kali)-[~/Desktop]
└─$ john --wordlist=/usr/share/wordlists/rockyou.txt sshjohn.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
Will run 4 OpenMP threads
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 (2025-06-16 07:13) 20.00g/s 78720p/s 78720c/s 78720C/s zamora..delicious
Use the "--show" option to display all of the cracked passwords reliably
Session completed.

┌──(kali㉿kali)-[~/Desktop]
└─$
```

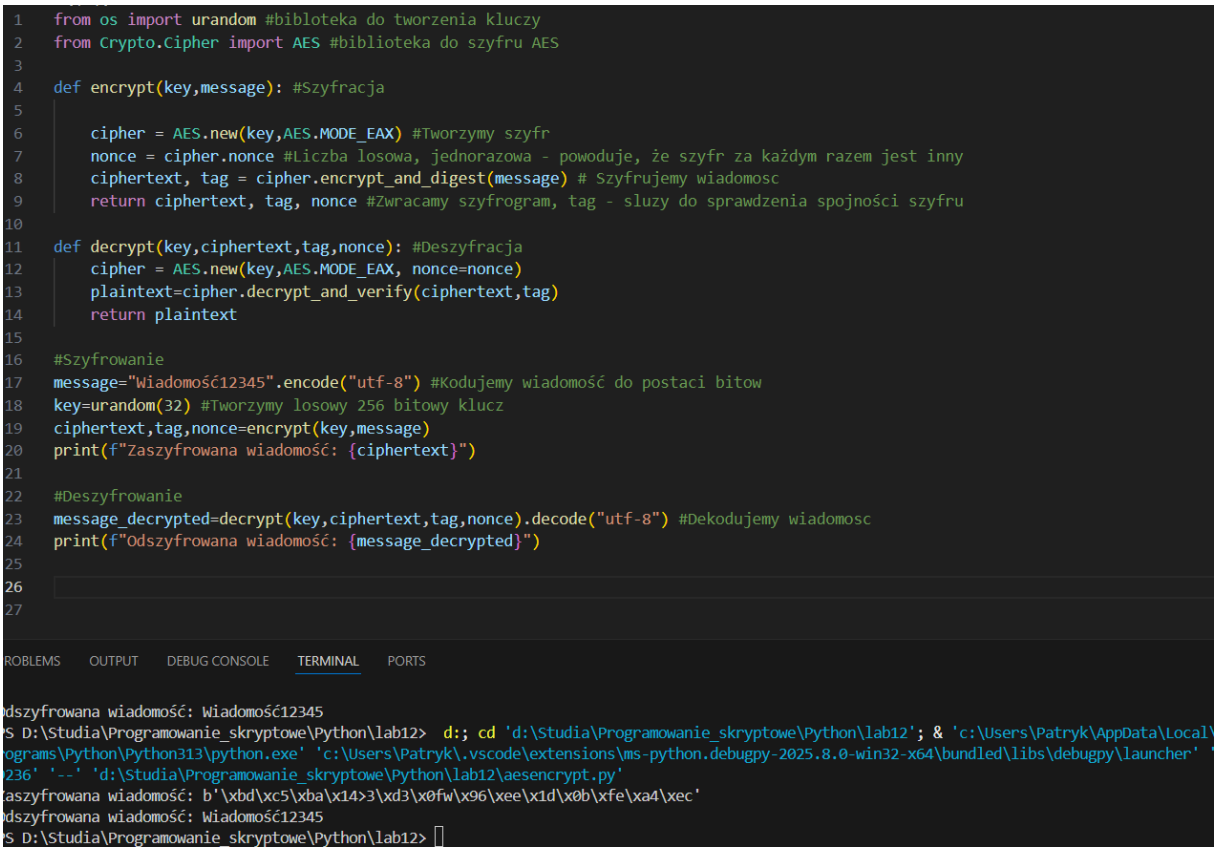
Zadanie11.




```
┌──(kali㉿kali)-[~/Desktop]
└─$ gpg --import tryhackme.key
gpg: key FFA4B5252BAEB2E6: "TryHackMe (Example Key)" not changed
gpg: key FFA4B5252BAEB2E6: secret key imported
gpg: Total number processed: 1
gpg:              unchanged: 1
gpg:       secret keys read: 1
gpg:  secret keys unchanged: 1

┌──(kali㉿kali)-[~/Desktop]
└─$ gpg --decrypt message.gpg
gpg: encrypted with 1024-bit RSA key, ID 2A0A5FDC5081B1C5, created 2020-06-30
      "TryHackMe (Example Key)"
You decrypted the file!
The secret word is Pineapple.
```

Zadanie2

https://github.com/wesoly772/Studia/blob/cd1ff73c9fc1a45c3248d66de6e1b89158cc0e9c/Programowanie_skryptowe/lab12/AESEncryption.py

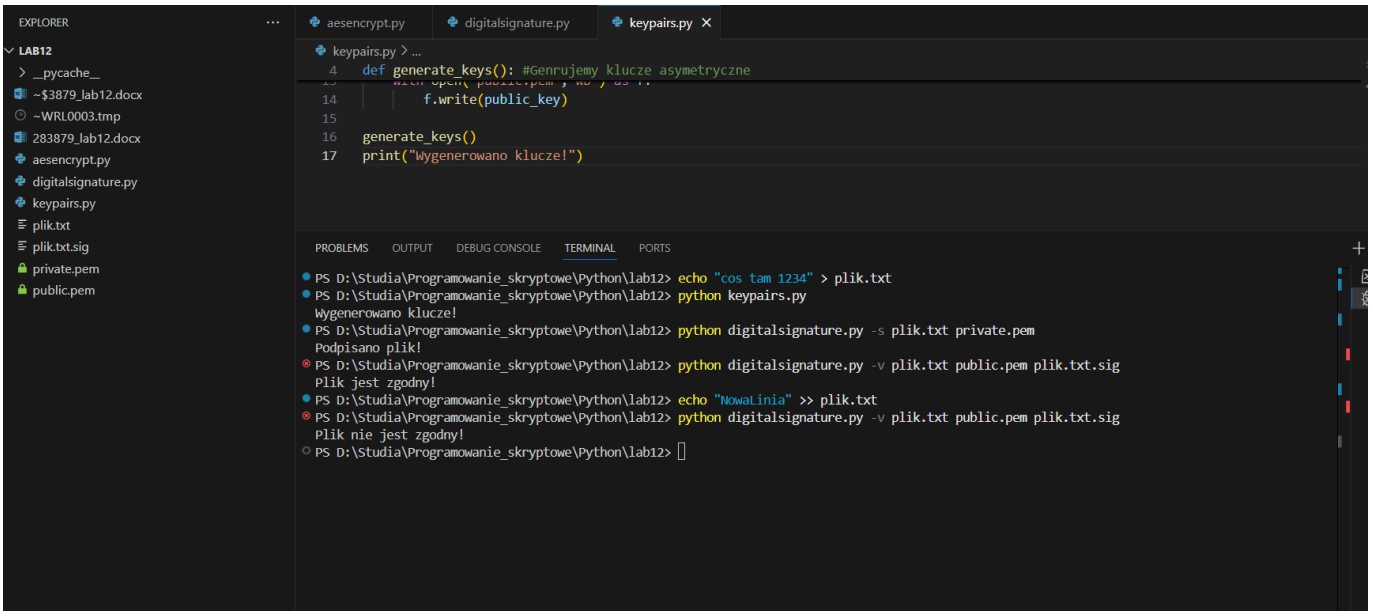


```python
from os import urandom #bibloteka do tworzenia kluczy
from Crypto.Cipher import AES #biblioteka do szyfru AES

def encrypt(key,message): #Szyfracja

    cipher = AES.new(key,AES.MODE_EAX) #Tworzymy szyfr
    nonce = cipher.nonce #Liczba losowa, jednorazowa - powoduje, że szyfr za każdym razem jest inny
    ciphertext, tag = cipher.encrypt_and_digest(message) # Szyfrujemy wiadomosc
    return ciphertext, tag, nonce #Zwracamy szyfrogram, tag - sluzy do sprawdzenia spojności szyfru

def decrypt(key,ciphertext,tag,nonce): #Deszyfracja
    cipher = AES.new(key,AES.MODE_EAX, nonce=nonce)
    plaintext=cipher.decrypt_and_verify(ciphertext,tag)
    return plaintext

#Szyfrowanie
message="Wiadomość12345".encode("utf-8") #Kodujemy wiadomość do postaci bitow
key=urandom(32) #Tworzymy losowy 256 bitowy klucz
ciphertext,tag,nonce=encrypt(key,message)
print(f"Zaszyfrowana wiadomość: {ciphertext}")

#Deszyfrowanie
message_decrypted=decrypt(key,ciphertext,tag,nonce).decode("utf-8") #Dekodujemy wiadomosc
print(f"Odszyfrowana wiadomość: {message_decrypted}")


```

```
ROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

dszyfrowana wiadomość: Wiadomość12345
S D:\Studia\Programowanie_skryptowe\Python\lab12>  d:; cd 'd:\Studia\Programowanie_skryptowe\Python\lab12'; & 'c:\Users\Patryk\AppData\Local\
rograms\Python\Python313\python.exe' 'c:\Users\Patryk\.vscode\extensions\ms-python.debugpy-2025.8.0-win32-x64\bundled\libs\debugpy\launcher' '
236' '--' 'd:\Studia\Programowanie_skryptowe\Python\lab12\aesencrypt.py'
aszyfrowana wiadomość: b'\xbd\xc5\xba\x14>3\xd3\x0fw\x96\xee\x1d\x0b\xfe\xa4\xec'
dszyfrowana wiadomość: Wiadomość12345
S D:\Studia\Programowanie_skryptowe\Python\lab12>
```

Zadanie3.

https://github.com/wesoly772/Studia/tree/cf0b963b165fa3e2745334030d7a476ecec6dfae/Programowanie_skryptowe/lab12/DigitalSignature


EXPLORER
aesencrypt.py
digitalsignature.py
keypairs.py
LAB12
__pycache__
~$3879_lab12.docx
~WRL0003.tmp
283879_lab12.docx
aesencrypt.py
digitalsignature.py
keypairs.py
plik.txt
plik.txt.sig
private.pem
public.pem
keypairs.py > ...
4 def generate_keys(): #Genrujemy klucze asymetryczne
14 f.write(public_key)
16 generate_keys()
17 print("Wygenerowano klucze!")
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
PS D:\Studia\Programowanie_skryptowe\Python\lab12> echo "cos tam 1234" > plik.txt
PS D:\Studia\Programowanie_skryptowe\Python\lab12> python keypairs.py
Wygenerowano klucze!
PS D:\Studia\Programowanie_skryptowe\Python\lab12> python digitalsignature.py -s plik.txt private.pem
Podpisano plik!
PS D:\Studia\Programowanie_skryptowe\Python\lab12> python digitalsignature.py -v plik.txt public.pem plik.txt.sig
Plik jest zgodny!
PS D:\Studia\Programowanie_skryptowe\Python\lab12> echo "NowaLinia" >> plik.txt
PS D:\Studia\Programowanie_skryptowe\Python\lab12> python digitalsignature.py -v plik.txt public.pem plik.txt.sig
Plik nie jest zgodny!
PS D:\Studia\Programowanie_skryptowe\Python\lab12>