Nel laboratorio di oggi è stato scirtto di fare quanto segue:

Esercizio di crittografia:
Messaggio cifrato: " HSNFRGH" e “QWJhIHZ6b2VidHl2bmdyIHB1ciB6ciBhciBucHBiZXRi”

Esercizio su kali
Criptazione e Firmatura con OpenSSLe Python:
Obiettivi dell'esercizio:

- Generare chiavi RSA.
- Estrarre la chiave pubblica da chiave privata.
- Criptare e decriptare messaggi.
- Firmare e verificare messaggi.

Strumenti utilizzati:

- OpenSSLper la generazione delle chiavi.
- Libreria cryptographyin Python.

Per il primo esercizio, il messaggio era codificato, e dovevo decifrarlo. Ammetto che non sapevo esattamente da dove partire, quindi ho provato a ragionare sul contesto. Ho pensato che potesse avere a che fare con Epicode, l’azienda che organizza il corso, e fortunatamente ci ho azzeccato! È stato un mix di intuito e fortuna, ma alla fine il messaggio aveva senso.

Se non si riesce a decifrare il messaggio con l’intuito, si può usare il “Cifrario di Cesare”. In questo metodo, ogni lettera viene spostata di tre posizioni nell’alfabeto. Ad esempio, “A” diventa “D”.

Per il secondo messaggio, ho usato il modulo base64 in Python per decodificarlo.

```
import base64

codificato = "QWJhIHZ6b2vIDhi2BMDYihb1CIb6ciBhciBucHBiZXRi"

decodificato = base64.b64decode(codificato)

print(decodificato)
```

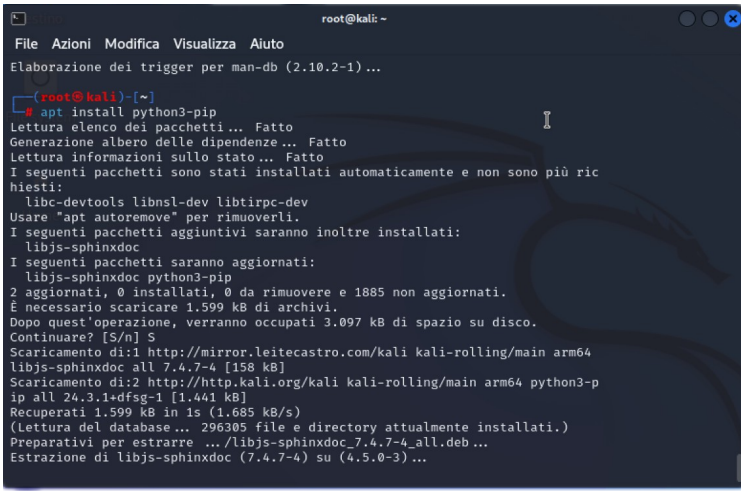
Dopo aver dato un’occhiata alle slide e fatto un po’ di ricerca sul web, sono riuscito a decodificarlo in modo grezzo.

```
┌──(kali㉿kali)-[~]
└─$ nano dec.py

┌──(kali㉿kali)-[~]
└─$ python dec.py
b'Aba vzok\xc8\x0e\x18\xb6\x04\xc0\xd8\x8a\x16\xf5\x08\x86\xfar ar nppbetb'
```

Nel terzo e ultimo esercizio ci è stato chiesto di utilizzare diversi strumenti, tra cui uno in Python per decrittografare delle chiavi e anche per gestire una firma.

In questa prima immagine, ci è stato chiesto di fare un aggiornamento generale dei pacchetti. Successivamente, dovevamo installare OpenSSL.

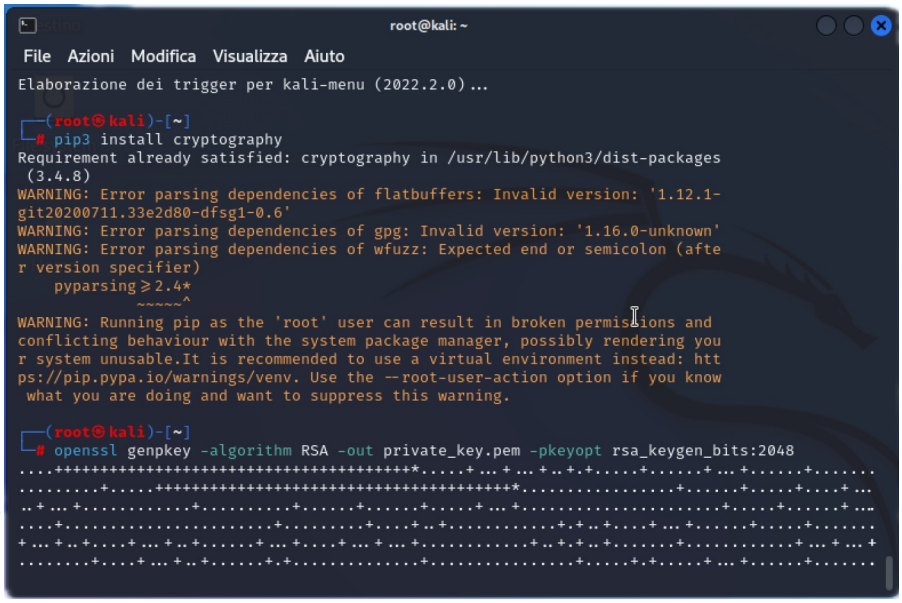
```
root@kali: ~
File  Azioni  Modifica  Visualizza  Aiuto
┌──(root㉿kali)-[~]
└─# apt update
Trovato:1 http://http.kali.org/kali kali-rolling InRelease
Lettura elenco dei pacchetti... Fatto
Generazione albero delle dipendenze... Fatto
Lettura informazioni sullo stato... Fatto
1905 pacchetti possono essere aggiornati: eseguire "apt list --upgradable" pe
r vederli.

┌──(root㉿kali)-[~]
└─# apt install openssl
Lettura elenco dei pacchetti... Fatto
Generazione albero delle dipendenze... Fatto
Lettura informazioni sullo stato... Fatto
I seguenti pacchetti sono stati installati automaticamente e non sono più ric
hiesti:
  libc-devtools libnsl-dev libtirpc-dev
Usare "apt autoremove" per rimuoverli.
I seguenti pacchetti aggiuntivi saranno inoltre installati:
  libc-bin libc-dev-bin libc-l10n libc6 libc6-dev libcryptsetup12
  libnss-systemd libpam-systemd libssl3t64 libsystemd-shared libsystemd0
  libzstd1 linux-sysctl-defaults locales openssh-client openssh-server
  openssh-sftp-server openssl-provider-legacy runit-helper systemd
  systemd-cryptsetup systemd-timesyncd
Pacchetti suggeriti:
  libc-devtools glibc-doc libnss-nis libnss-nisplus libtss2-rc0t64
  libarchive13t64 libbpf1 libdw1t64 libelf1t64 libpwquality1 keychain
```

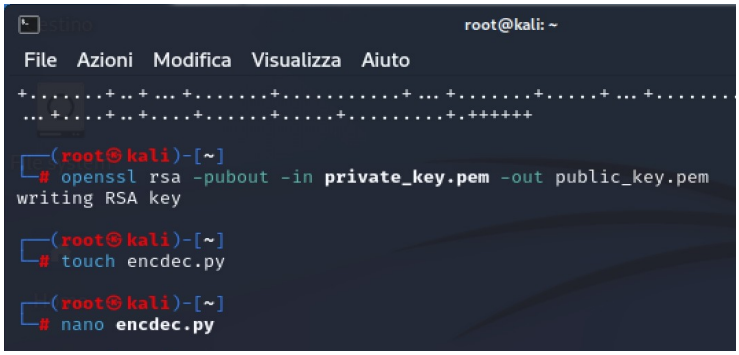


Dopo aver installato OpenSSL, il passo successivo è stato procedere con l'installazione di Python 3.

Una volta completata l'installazione di Python, il passo successivo è stato l'installazione della libreria criptografica di Python. Per fare ciò, abbiamo utilizzato il comando pip per installare la libreria cryptography, che ci ha permesso di eseguire le operazioni di criptografia e decrittografia necessarie per il proseguire con l'esercizio.

PS: anche se durante l'installazione sono comparsi alcuni errori, ho deciso di continuare comunque, poiché il sistema mi ha permesso di proseguire senza blocchi.

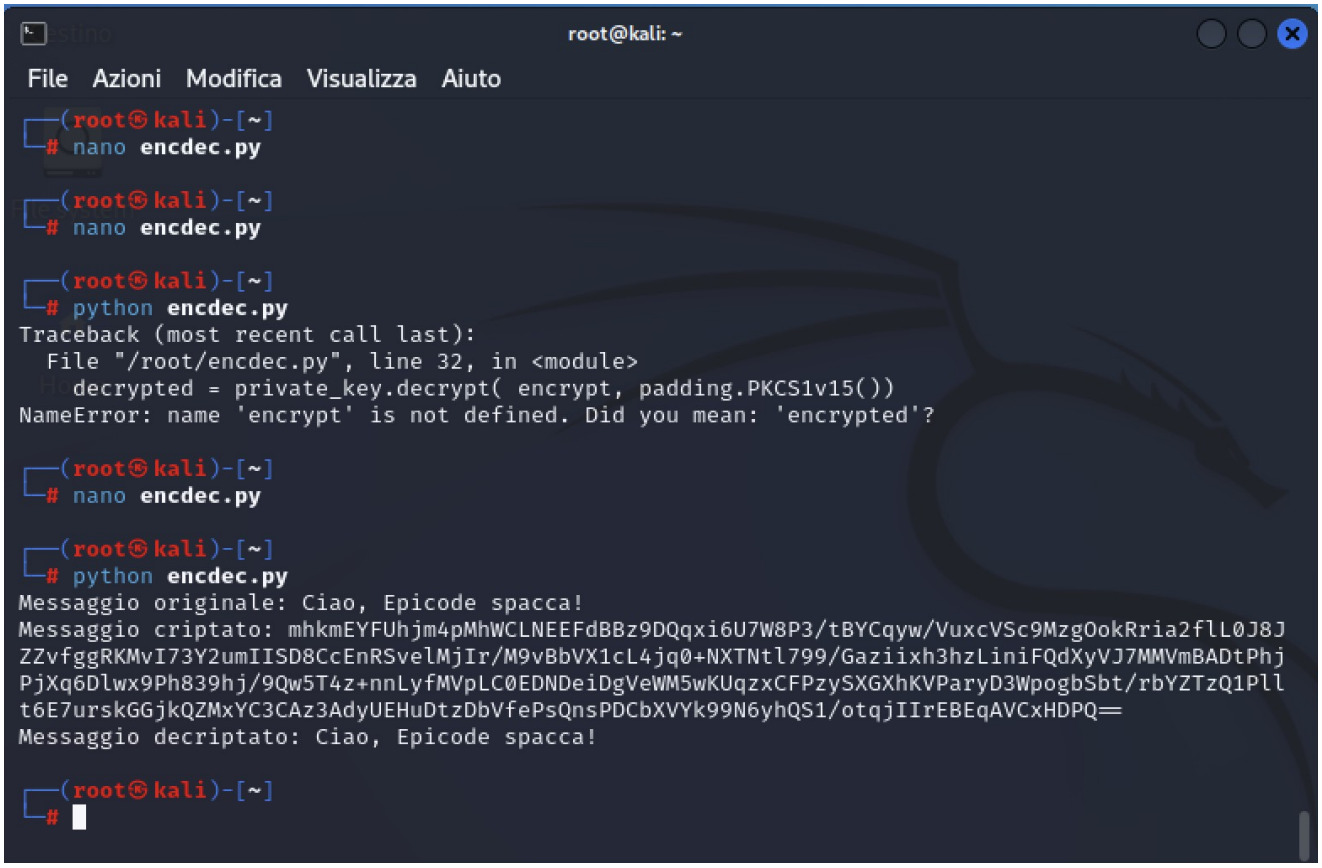




Una volta completata l'installazione di Python e della libreria cryptography, insieme a OpenSSL, come nella precedente immagine, ho creato tramite touch e nano per modificare la prima firma

Una volta completati tutti i passaggi mostrati nelle slide dell'esercizio, è il momento di testare la firma

Come si vede nell'immagine seguente, dopo vari tentativi, Python ha segnalato un errore che impediva la corretta firma. L'errore era dovuto al fatto che, invece di "ENCTYPE", era stato scritto "ENCTYPED". Una volta corretta questa svista, la firma è stata effettuata correttamente, come confermato dal test successivo.



Nell'immagine successiva, una volta scritto correttamente il codice sorgente (come indicato nella seconda parte delle slide), e dopo aver effettuato le necessarie correzioni, siamo riusciti finalmente a ottenere la firma.