

## • Como o algoritmo de criptografia escolhido funciona?

Foi implementado o algoritmo Fernet do pacote cryptography para Python. Conceitualmente, o fernet recebe uma mensagem fornecida pelo usuário (uma sequência arbitrária de bytes), uma chave (256 bits) e a hora atual e produz um token, que contém a mensagem em um formato que não pode ser lido ou alterado sem a chave.

A chave é formada pela chave de assinatura (128 bits) e a chave de criptografia (128 bits)

O token é formado por:

**Versão:** Este campo indica qual versão do formato está sendo usada pelo token.

Atualmente existe apenas uma versão definida, com o valor 128 (0x80).

**Carimbo de data de hora:** Este campo é um inteiro big-endian não assinado de 64 bits. Ele registra o número de segundos decorridos entre 1º de janeiro de 1970 UTC e a hora em que o token foi criado.

**IV:** O vetor de inicialização de 128 bits usado na criptografia AES e na descriptografia do texto cifrado. Ao gerar novos tokens fernet, o IV deve ser escolhido exclusivamente para cada token. Com uma fonte de entropia de alta qualidade, a seleção aleatória fará isso com alta probabilidade.

**Texto cifrado:** Este campo tem tamanho variável, mas é sempre um múltiplo de 128 bits, o tamanho do bloco AES. Ele contém a mensagem de entrada original, preenchida e criptografada.

**HMAC:** Este campo é o HMAC SHA256 de 256 bits, sob chave de assinatura, da concatenação dos seguintes campos: version + timestamp + iv + ciphertext

**Criptografando:** Dada uma chave e uma mensagem, gere um token fernet com as seguintes etapas, em ordem:

Registre a hora atual para o campo de carimbo de data/hora.

Escolha um IV exclusivo.

Construa o texto cifrado:

Preencher a mensagem para um múltiplo de 16 bytes (128 bits) por RFC 5652, seção 6.3 . Essa é a mesma técnica de preenchimento usada no PKCS #7 v1.5 e em todas as versões do SSL/TLS (cf. RFC 5246, seção 6.2.3.2 para TLS 1.2).

Criptografe a mensagem preenchida usando AES 128 no modo CBC com o IV escolhido e a chave de criptografia fornecida pelo usuário.

Calcule o campo HMAC conforme descrito acima usando a chave de assinatura fornecida pelo usuário.

Concatene todos os campos juntos no formato acima.

base64url codifica todo o token.

**Descriptografando:** Dado uma chave e um token, para verificar se o token é válido e recuperar a mensagem original, execute as seguintes etapas, na ordem:

base64url decodificar o token.

Certifique-se de que o primeiro byte do token seja 0x80.

Se o usuário tiver especificado uma idade máxima (ou "tempo de vida") para o token, certifique-se de que o carimbo de data/hora gravado não seja muito antigo.

Recalcule o HMAC dos outros campos e da chave de assinatura fornecida pelo usuário.

Certifique-se de que o HMAC recalculado corresponda ao campo HMAC armazenado no token, usando uma função de comparação de tempo constante.

Descriptografar o campo de texto cifrado usando AES 128 no modo CBC com o IV gravado e a chave de criptografia fornecida pelo usuário.

Descompacte o texto simples descriptografado, produzindo a mensagem original

- **Justificar porquê da escolha de determinado algoritmo.**

Escolhi devido a facilidade de implementação.

- **Quais serão as vantagens de utilizar o algoritmo no sistema?**

A segurança, pois é gerada uma nova chave de criptografia randômica e nova a cada execução, bem como a facilidade em se implementar.

- **Quais serão as desvantagens que o uso do algoritmo poderá trazer para o sistema?**

O conteúdo completo da mensagem deve estar disponível na memória, o que torna o Fernet geralmente inadequado para arquivos muito grandes. Um outro fato na mesma linha é que seus tokens não são autolimitados, devendo o sistema também

- **Implementar o algoritmo escolhido.**

```
from cryptography.fernet import Fernet #IMPORTA MÓDULO pip install cryptography
```

```
# OBTEM A MENSAGEM
```

```
mensagemOriginal = ("Testando criptografia")
```

```
# GERA UMA CHAVE ALEATÓRIA DE CRIPTOGRAFIA E A EXIBE
```

```
chave = Fernet.generate_key()
```

```
fernet = Fernet(chave) # PREPARA A CHAVE PARA O USO
```

```
# CRIPTOGRAFA
```

```
mensagemCriptografada = fernet.encrypt(mensagemOriginal.encode())
```

```
# DESCRIPTOGRAFA
```

```
mensagemDescriptograda = fernet.decrypt(mensagemCriptografada).decode()
```

```
print ("A mensagem original é: "+mensagemOriginal)

print ("A chave de criptografia e:",chave)

print ("A mensagem criptografada é: ",mensagemCriptografada)

print ("A mensagem descriptografada é: "+mensagemDescriptografada)
```

## Resultado:

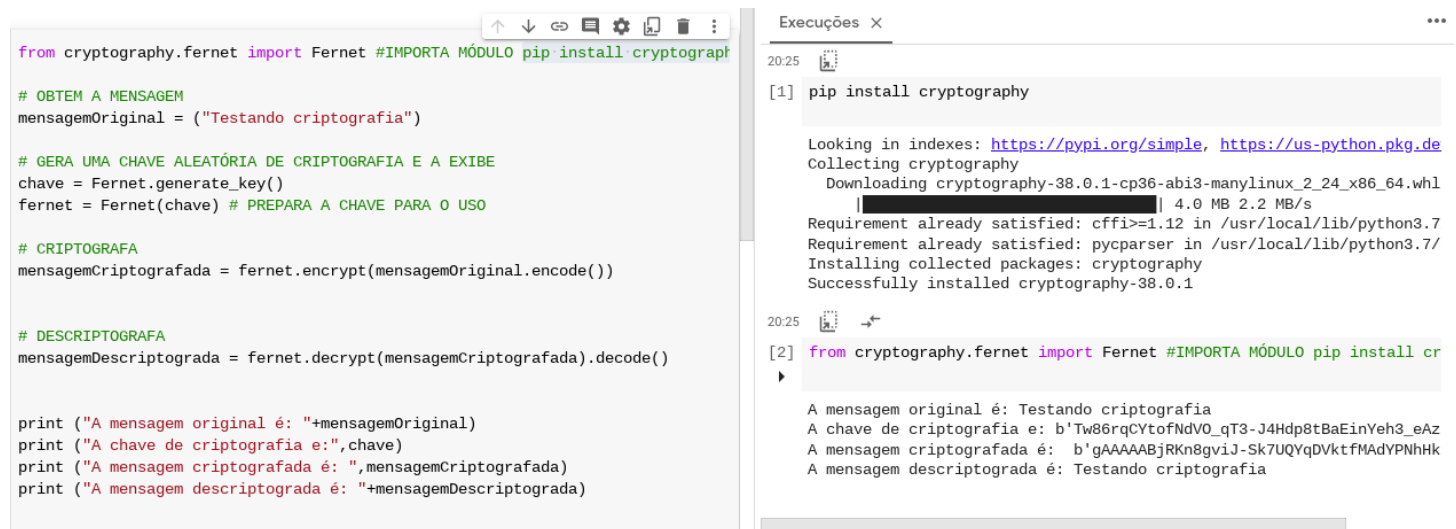
A mensagem original é: Testando criptografia

A chave de criptografia e: b'3UdSf31Gdpuqv1-09GMPHjbuy18U4Q4Ewxd\_TjNV8l8='

A mensagem criptografada é:

b'gAAAAABjRkK09a17bJMuWHgzSJsTOsJZT70HG81zM6VQgGbTROJyYZRwtF2cTX33mq\_FcRjPE1kNaw4rF8NHbY0VCuK  
ZiY\_m33LbSt9xEKCx8Wc2y4D9nKo='

A mensagem descriptografada é: Testando criptografia



```
from cryptography.fernet import Fernet #IMPORTA MÓDULO pip install cryptography

# OBTEN A MENSAGEM
mensagemOriginal = ("Testando criptografia")

# GERA UMA CHAVE ALEATÓRIA DE CRIPTOGRAFIA E A EXIBE
chave = Fernet.generate_key()
fernet = Fernet(chave) # PREPARA A CHAVE PARA O USO

# CRIPTOGRAFA
mensagemCriptografada = fernet.encrypt(mensagemOriginal.encode())

# DESCRIPTOGRAFA
mensagemDescriptografada = fernet.decrypt(mensagemCriptografada).decode()

print ("A mensagem original é: "+mensagemOriginal)
print ("A chave de criptografia e:",chave)
print ("A mensagem criptografada é: ",mensagemCriptografada)
print ("A mensagem descriptografada é: "+mensagemDescriptografada)
```

Execuções X

20:25 [1] pip install cryptography

Looking in indexes: <https://pypi.org/simple>, <https://us-python.pkg.de>  
Collecting cryptography  
 Downloading cryptography-38.0.1-cp36-abi3-manylinux\_2\_24\_x86\_64.whl  
 |██| 4.0 MB 2.2 MB/s  
Requirement already satisfied: cffi>=1.12 in /usr/local/lib/python3.7/  
Requirement already satisfied: pycparser in /usr/local/lib/python3.7/  
Installing collected packages: cryptography  
Successfully installed cryptography-38.0.1

20:25 [2] from cryptography.fernet import Fernet #IMPORTA MÓDULO pip install cr

A mensagem original é: Testando criptografia  
A chave de criptografia e: b'Tw86rqCYtofNdVO\_qT3-J4Hdp8tBaEinYeh3\_eAz  
A mensagem criptografada é: b'gAAAAABjRkN8gviJ-Sk7UQYqDVktfMAdYPNhHk  
A mensagem descriptografada é: Testando criptografia