

# Exercícios - semana 2 (individual)

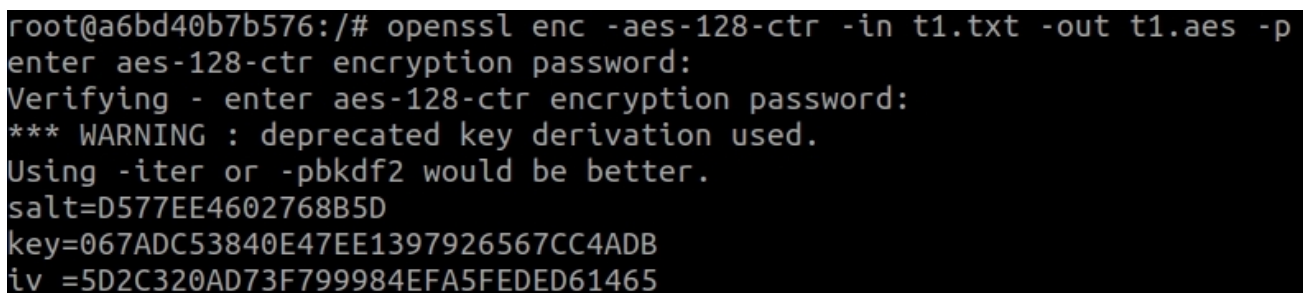
---

**Bruno Aurélio Rôzza de Moura Campos (14104255)**

## Questão 73

(Usando o openssl) Crie o arquivo t1.txt no gedit e escreva algum conteúdo dentro do arquivo. Cifrar o arquivo t1.txt com o algoritmo AES no modo CTR. Use screenshot para documentar a execução do comando. Ver modos de cifragem usando: openssl enc -help Use o comando:

```
openssl enc -aes-128-ctr -in t1.txt -out t1.aes -p
```



```
root@a6bd40b7b576:/# openssl enc -aes-128-ctr -in t1.txt -out t1.aes -p
enter aes-128-ctr encryption password:
Verifying - enter aes-128-ctr encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
salt=D577EE4602768B5D
key=067ADC53840E47EE1397926567CC4ADB
iv =5D2C320AD73F799984EFA5FEDED61465
```

**a. Qual chave foi usada para cifrar o arquivo?**

key=067ADC53840E47EE1397926567CC4ADB

**b. Qual IV foi usado?**

iv =5D2C320AD73F799984EFA5FEDED61465

**c. Como foram gerados a chave e o IV?**

O próprio openssl é quem gerou a chave e iv com base na senha passada no momento de criptografar o arquivo

**d. Onde ficam guardados a chave e o IV?**

Somente é guardado o salt (salt=D577EE4602768B5D) e a partir do salt + senha é possível gerar novamente a **chave** e **IV**.

## Questão 74

(Usando o openssl) Agora, decifre o arquivo t1.aes. Use screenshot para documentar a execução do comando. Use o comando:

```
openssl enc -aes-128-ctr -d -in t1.aes -p
```

O argumento -d significa “decifrar”.

```
root@a6bd40b7b576:/# openssl enc -aes-128-ctr -d -in t1.aes -p
enter aes-128-ctr decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
salt=D577EE4602768B5D
key=067ADC53840E47EE1397926567CC4ADB
iv =5D2C320AD73F799984EFA5FEDED61465
questao 73
```

### Questão 75

(Usando o openssl) Gerar uma chave secreta usando o comando (coloque o seu nome):

```
openssl rand -out chaveSecretaNomeAluno.bin -base64 128
```

```
root@a6bd40b7b576:/seguranca# cat chave_secreta_brunocampos.bin
Kf03Bsv1owXtjr0DvZSNaeVxjrlrJoFQ3S9QnD39RHTie+pib7RyNEndJ0EJwbGs
TlhWeNica21QMoMyXXW0RLZiizBFFkDoFUykStkd8F7qugYyg5RUW9VaNrRq8P96
8Gidchz3ERgR0b49EG8cz27QiETZA/qQwcs4HTcrXD0=
```

### Questão 76

(Usando o openssl) Cifrar o arquivo msgPlana.txt (crie o arquivo com algum conteúdo) com a chave secreta criada na questão anterior

```
openssl enc -aes-128-ctr -in msgPlana.txt -out msgCifrada -k
file:./chaveSecretaNomeAluno.bin
```

```
root@a6bd40b7b576:/seguranca# cat msgCifrada
Salted__9ت*|{
F,croot@a6bd40b7b576:/seguranca#
```

### Questão 77

(Usando o openssl) Monte o comando para decifrar o arquivo msgCifrada. Mostre o comando e sua execução

```
root@a6bd40b7b576:/seguranca# openssl enc -aes-128-ctr -d -in msgCifrada -k file:chave_secreta_brunocampos.bin
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
boa noite !!!
```

### Questão 78

(Usando o openssl) Crie o arquivo t1.txt no gedit e escreva algum conteúdo dentro do arquivo. Para gerar o hash deste arquivo usando o algoritmo sha256, pode ser usado o seguinte comando:

```
openssl dgst -sha256 t1.txt
```

a. Obtenha a saída do comando: `openssl dgst -sha256 -c t1.txt`

```
root@a6bd40b7b576:/seguranca# openssl dgst -sha256 t1.txt
SHA256(t1.txt)= cd855b5a0f64d98e73957341882b94b19604fb8b26e6b63a40c1b0e73e0e1a30
```

b) Encontre um arquivo em alguma página web que tenha o valor do hash SHA-256 listado na página (Sugestão de página: <http://httpd.apache.org/download.cgi#apache24>). Baixe o arquivo e recalcule o valor do SHA-256 com o openssl para conferir se o valor calculado é igual ao listado na página web. Mostre a tela da execução desse comando e indique o site usado.

```
root@a6bd40b7b576:/seguranca# cat httpd-2.4.46.tar.bz2.sha256
740eddf6e1c641992b22359cab66e6325868c3c5e2e3f98faf349b61ecf41ea *httpd-2.4.46.tar.bz2
root@a6bd40b7b576:/seguranca# openssl dgst -sha256 httpd-2.4.46.tar.bz2
SHA256(httpd-2.4.46.tar.bz2)= 740eddf6e1c641992b22359cab66e6325868c3c5e2e3f98faf349b61ecf41ea
```

79

(Usando o openssl) Para gerar o MAC do arquivo use o comando abaixo, mostrando a tela de execução do comando. Explique os parâmetros usados no comando.

a. `openssl dgst -sha256 -mac HMAC -macopt hexkey:aabbcc t1.txt`

- HMAC: é um tipo de MAC
- macopt: é um parâmetro de algoritmo de MAC
- hexkey: é uma chave hexa com o valor aabbcc
- t1.txt: arquivo de entrada

b. Para gerar uma chave de 128 bits, use o comando: `openssl rand -hex 32`

c. Agora repita o comando do HMAC usando esta chave e mostre a execução do comando.

```
root@a6bd40b7b576:/seguranca# openssl rand -hex 32
e22f3620d3edc7638cb26d363d684a4b4886b7586fe623c59976d7048be29f45
root@a6bd40b7b576:/seguranca#
root@a6bd40b7b576:/seguranca#
root@a6bd40b7b576:/seguranca# openssl dgst -sha256 -mac HMAC -macopt hexkey:e22f3620d3edc7638cb26d363d684a4b488
6b7586fe623c59976d7048be29f45 t1.txt
HMAC-SHA256(t1.txt)= 1fb706cb2087297112c6455ed9f7c1618da9c8542bc4fc9f4e4fd5786045bbc8
```