**Universidade da Beira Interior**
Faculdade de Engenharia
Departamento de Informática



# Segurança de Sistemas Informáticos
Guia para Aula Laboratorial 1
2º Ciclo em Engenharia Informática
2º Ciclo em Eng. Eletrotécnica e de Computadores
2º Ciclo em Matemática e Aplicações

# Computer Systems Security
Guide for Laboratory Class 1
M.Sc. in Computer Science and Engineering
M.Sc. in Electrical and Computer Engineering
M.Sc. in Mathematics and Applications

**Sumário**
Exercícios de revisão de conceitos relativos a funções de *hash*, criptografia de chave simétrica e códigos de autenticação de mensagens, bem como da forma de manuseamento da ferramenta `OpenSSL`.

***Summary***
*Review exercises for concepts related with hash functions, symmetric key cryptography and message authentication codes, as well as of the way of handling the `OpenSSL` toolkit.*

**Pré-requisitos:**
A maior parte das tarefas enunciadas em baixo requerem a utilização da ferramenta `OpenSSL` (`http://www.openssl.org/`). Um ambiente linha de comandos robusto também facilita a execução de algumas dessas tarefas. Sugere-se, assim, o uso de uma distribuição comum de Linux, onde todas estas condições estarão provavelmente preenchidas ou pressupõe-se o acesso a um sistema com a possibilidade de instalar o *software* necessário.

## 1 Cifras de Chave Simétrica
*Symmetric-key Criptography*

### Tarefa 1 *Task 1*

Crie o ficheiro `confidential.txt` com a sequência `111111111111222222222222` (doze 1s e doze 2s) repetida 3 vezes.

### Tarefa 2 *Task 2*

Cifre o ficheiro com a cifra ChaCha20, usando a chave em hexadecimal `fff22f2fff`. O ficheiro cifrado deve ter a designação `confidential.cc20`. Escreva o comando que utilizar no espaço seguinte:

```
openssl enc -e -chacha20 -in confidential.txt -out confidential.cc20 -K fff22f2fff -iv 0
```

### Tarefa 3 *Task 3*

**Q1.: Experimentou abrir o ficheiro cifrado?**
☒ Sim.
☐ Não, nem sei como fazer isso convenientemente, visto que está cifrado.
Sugestão: use o comando `hexdump`:

```
> hexdump confidential.cc20
```

**Q2.: Quantos caracteres hexadecimais diferentes são produzidos pelo comando anterior para um ficheiro cifrado?**
☐ 2 ☐ 4 ☐ 8 ☒ 16 ☐ 32 ☐ hexilions

Considere que lhe pediam para calcular o valor da entropia para os bytes do ficheiro resultante usando o logaritmo de base natural. **Q3.: Qual lhe parece ser o valor mais provável para a entropia deste ficheiro?**
☒ 5.50 ☐ 4.32 ☐ 2.40 ☐ 0 ☐ 10 ☐ $+\infty$ ☐ $-\infty$

Antes de prosseguir, responda à seguinte pergunta: **Q4.: Se voltar a cifrar o ficheiro que já foi cifrado, vai conseguir ler facilmente o seu conteúdo ou não?**
☒ Claro que sim. ☐ Acho que não.

### Tarefa 4 *Task 4*

Volte a cifrar o ficheiro (com a mesma chave) e tente abrir o ficheiro resultante com um editor de texto. Chame ao novo ficheiro `confidential.2times.cc20`.

**Q5.: O resultado coaduna-se com o que respondeu na pergunta anterior?** Para seu próprio proveito, procure justificar o que observou.

**Q6.: Que tipo de cifra é a ChaCha20?**

☒ Cifra de chave simétrica contínua.
☐ Cifra de chave simétrica por blocos.
☐ Cifra de chave pública.
☐ Cifra de chave pública contínua.
☐ Cifra de chave pública por blocos.

A ChaCha20 é uma modificação de uma outra cifra desenvolvida por *Daniel J. Bernstein*. **Q7.: Qual o nome dessa cifra inicial?**

☐ Bachata20 ☐ Merengue20 ☐ Mambo N. 5
☐ Kizomba20 ☒ Salsa20

**Q8.: Qual o tamanho máximo que uma mensagem pode ter para que a sua cifra ainda possa ser considerada segura com a ChaCha20?**

☐ 100MB ☐ 200GB ☒ 256GiB
☐ 10MB ☐ 1Tebibyte

### Tarefa 5 *Task 5*

Cifre o ficheiro `confidential.txt` com a cifra AES no modo *Electronic Code Book* (ECB) (i.e. `-aes-128-ecb`) com a chave de cifra hexadecimal `11232233`. O ficheiro resultante deve chamar-se `confidential.aes-ecb`. Tente visualizar o conteúdo este ficheiro e escreva todos os comandos que utilizar a seguir:

```
openssl enc -e -aes-128-ecb -in confidential.txt -out confidential.aes-ecb -K 11232233
hexdump confidential.aes-ecb
cat confidential.aes-ecb
```

**Q9.: Consegue detetar padrões no ficheiro cifrado?**

☒ Claramente. ☐ Nem por isso.

**Nota:** se respondeu que não, considere uma visita ao oculista. Justifique a sua resposta:

os blocos de texto que se repetem no .txt, repetem.se também na crifra, pois esta cifra trabalha por blocos e vlocos iguais significam textos iguais

**Q10.: De acordo com a experiência que fez antes, qual é o tamanho do bloco que esta cifra utiliza?** ☐ 8 bytes ☒ 16 bytes ☐ 32 bytes ☐ 64 bytes

**Q11.: O modo ECB precisa de vetor de inicialização?**

☐ Sim. ☒ Não.

**Q12.: Que tipo de cifras concretizam a família de algoritmos especificados pela AES?**

☐ Cifra de chave simétrica contínua.
☒ Cifra de chave simétrica por blocos.
☐ Cifra de chave pública.
☐ Cifra de chave pública contínua.
☐ Cifra de chave pública por blocos.

**Q13.: O que significa AES?**

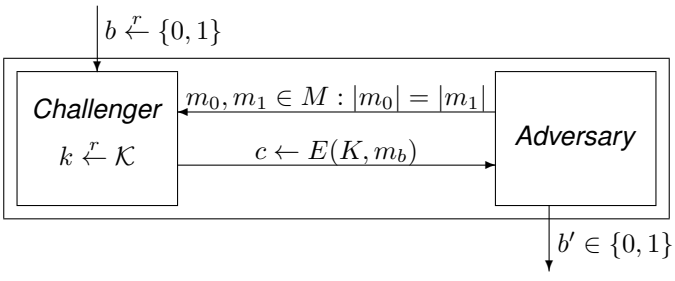
A dvanced

E ncryption

S tandart

### Tarefa 6 *Task 6*

O objetivo desta tarefa é simular o jogo que define o ataque em que apenas o criptograma é conhecido, tipicamente representado por um esquema semelhante a:




Junte-se com outro(a) colega. Um(a) dos(as) dois(uas) faz de *Challenger*, o(a) outro(a) faz de *Adversary*. O jogo funciona da seguinte forma:

1. O(a) *Adversary* tem de enviar dois ficheiros ao(à) *Challenger*.

2. O(a) *Challenger* gera duma chave de cifra (qualquer) sem a dizer ao *Adversary* e cifra um dos dois ficheiros (mas também não lhe diz qual).

3. O(a) *Challenger* envia o criptograma do ficheiro ao *Adversary*.

4. O(a) *Adversary* tem de adivinhar qual o ficheiro que foi cifrado.

O(a) *Adversary* tem total liberdade para escolher os dois ficheiros e o seu objetivo é acertar com 100% de certeza.

**Q14.: É o(a) *Challenger* ou o(a) *Adversary*?**
☐ *Challenger* ☐ *Adversary*

**Q15.: Consegue ganhar sempre este jogo?**
☒ Canja de galinha. ☒Não.

### Tarefa 7 *Task 7*

Use o seguinte comando para cifrar o ficheiro `confidential.txt`:

```
> openssl enc --aes-128-cbc -K 11232233 -in confidential.txt -out confidential.aes-cbc
```

**Q16.: O comando antes enunciado funcionou?**
☐ Sim, funcionou. ☒Não, não funcionou.

**Q17.: Como é que se resolve a situação?**
☐ Mudando a opção `enc` para `dec`.
☐ Adicionando um *salt* com a opção `-salt`.
☐ Definindo outra chave de cifra.
☒ Definindo um vetor de inicialização com a opção `-iv`.

**Nota:** quando souber como resolver a situação, emita novamente o comando, porque vai precisar do ficheiro a seguir.

**Q18.: O que significa CBC?**

C _____

B _____

C _____

**Q19.: O que é o CBC?**
☐ O nome de um algoritmo de cifra.
☒ Um modo de utilização de uma cifra.
☐ O nome de uma cadeia de televisão norte americada, que produz umas séries impecáveis.

Antes de prosseguir, responda à seguinte pergunta:

**Q20.: Se voltar a cifrar o ficheiro que já foi cifrado com `--aes-128-cbc`, vai conseguir ler facilmente o seu conteúdo ou não?**
☐ Claro que sim. ☒ Acho que não.

### Tarefa 8 *Task 8*

Então, volte a cifrar o ficheiro e tente abrir o ficheiro resultante com um editor de texto. Chame ao novo ficheiro `confidential.2times.aes-cbc`.

**Q21.: O resultado coaduna-se com o que respondeu na pergunta anterior?**
☐ Sim, impecável. ☒Ups...
Para seu próprio proveito, procure justificar o que observou.

_____

_____

_____

**Q22.: Como é que se decifra um ficheiro cifrado no modo CBC?**
openssl enc -d --aes-128-cbc -K 11232233 -in confidential.aes-cbc -out confidential.2times.txt -iv 0

_____

### Tarefa 9 *Task 9*

Quando se especifica uma palavra-passe em vez de uma uma chave de cifra (i.e., usar a opção `-k` em vez de `-K`), é conveniente definir uma *pitada de sal* usando a opção `-S` (para além do `iv`).

**Q23.: Porque?**
técnicas de força bruta ou tabelas de hash pré-calculadas

_____

_____

**Q24.: Acha que é preciso inserir o mesmo valor do salt quando se vai decifrar o ficheiro, ou basta a palavra-chave e o `iv`?**
☒ É preciso inserir outra vez o *salt*.
☐ Deixa-me mas é experimentar...
☐ Deixa-me mas é experimentar e... sei lá, fazer `> cat` ao ficheiro cifrado para ver o que lá vai dentro.

## 2 Maneabilidade

*Maleability*

Considere que queria enviar a mensagem `Enviar 2 euros para o Bob` ao seu banco. Esta mensagem é sensível, porque diz respeito ao seu dinheiro. Para a proteger, decidiu cifrá-la antes de a enviar.

### Tarefa 10 *Task 10*

Crie o ficheiro `texto-limpo.txt` e coloque lá a mensagem referida em cima. De seguida, cifre o ficheiro com uma cifra por blocos:

```
> openssl enc -aes-128-cbc -K
0123456789abcdef0123456789abcdef -in
texto-limpo.txt -out cifrado.aes -iv 0
```

### Tarefa 11 *Task 11*

Use o programa incluído a seguir para alterar um só byte (na verdade altera um só bit) do ficheiro cifrado obtido (considere analisar o programa com calma):

```c
#include<stdio.h>

void alterbyte(char * in, char * out, int
    ibytenumber){

  FILE *fIN, *fOUT;
  fIN  = fopen(in,  "r");
  fOUT = fopen(out, "w");
  if( (fIN == NULL) | (fOUT == NULL) ) {
    fprintf(stderr, "Problem with file\n");
    exit(1);
  }

  int b, i = 1;
  b = fgetc(fIN);
  while (!feof(fIN)) {
    if(i == ibytenumber)
      b = b ^ 0x01;
    fputc(b, fOUT);
    b = fgetc(fIN);
    i++;
  }
}

int main(int argc, char **argv){
  alterbyte(argv[1],argv[2],atoi(argv[3]));
}
```

Invoque o programa da seguinte forma:

```
> ./a.out cifrado.aes cifrado-2.aes 8
```

Depois volte a decifrar o ficheiro com:

```
> openssl enc -d -aes-128-cbc -K
0123456789abcdef0123456789abcdef -in
cifrado-2.aes -out texto-limpo-2.txt -iv
0
```

Abra o ficheiro `texto-limpo-2.txt` e verifique o que lá tem dentro. **Q25.: Consegue ler a mensagem original?**

☐ Sim, claro que consigo... afinal, quase não alterei nada.

☒ Não... de maneira nenhuma. Que estranho!?

**Q26.: Diria que, se alguém alterasse qualquer byte no ficheiro durante a sua transmissão, o recetor iria notar essa alteração?**

☒ Sim, neste caso poder-se-ia dizer isso.

☐ Não, neste caso não se nota nada.

### Tarefa 12 *Task 12*

Apague os ficheiros `cifrado.aes`, `cifrado-2.aes` e `texto-limpo-2.txt` e recomece, desta feita com o modo *CounTeR* (CTR). **Q27.: Em que tipo de cifra se transforma qualquer cifra por blocos quando se usa este modo?**

☒ Cifra de chave simétrica contínua.

☐ Cifra de chave simétrica por blocos.

☐ Cifra de chave pública.

☐ Cifra de chave pública contínua.

☐ Cifra de chave pública por blocos.

Cifre o ficheiro `texto-limpo.txt` com um comando semelhante a:

```
> openssl enc -aes-128-ctr -K
0123456789abcdef -in texto-limpo.txt -out
cifrado.aes-ctr -iv 0123456789
```

Verifique o conteúdo do ficheiro cifrado com `> cat cifrado.aes-ctr`.

### Tarefa 13 *Task 13*

Use o programa incluído antes para alterar o byte 8 do ficheiro via

```
> ./a.out cifrado.aes-ctr cifrado-2.aes-ctr
8
```

,

e volte a decifrá-lo usando:

```
> openssl enc -aes-128-ctr -K
0123456789abcdef -in cifrado-2.aes-ctr -out
texto-limpo-2.txt -iv 0123456789
```

**Q28.: Nota algo estranho no ficheiro decifrado?**

☒ Sim, noto algo até meio (vá) assustador!

☐ Não... nada.

**Q29.: Será que um atacante poderia mudar o texto-limpo sem o conhecer, alterando apenas o criptograma respetivo, e sem ser notado?**
☒ Sim, pelos vistos sim.
☐ Não... nunca neste caso.

**Q30.: De 0 a 5, em que 0 é muito má e 5 é muito boa, como classifica o facto de uma cifra ser maneável?**
☒ 0 ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5.

**Q31.: Usaria uma cifra simétrica contínua para transmitir uma mensagem confidencial sobre um canal que está sujeito a escutas, mas não a alteração das mensagens?**
☒ Sim, sem problema. ☐ Não, nem pensar.

## 3 Funções de *Hash*

*Hash Functions*

### Tarefa 14 *Task 14*

No terminal, calcule o valor resumo (*hash*) da cadeia de caracteres vazia com o algoritmo MD5 e com o SHA1. Guarde para a posteridade os comandos que utilizar:

```
openssl dgst -sha1 texto-limpo.txt
openssl dgst -md5 texto-limpo.txt
```

**Q32.: Obteve os valores**
`d41d8cd98f00b204e9800998ecf8427e` **e**
`da39a3ee5e6b4b0d3255bfef95601890afd80709`**?**
☒ Sim. ☐ Não.

Calcule também os valores de *hash* com o SHA1 e SHA256 da cadeia de caracteres `The quick brown fox jumps over the lazy dog` e indique em baixo os comandos que utilizou para esse efeito:

Confira se obteve os mesmos valores que são indicados, e.g., na Wikipédia, para os dois algoritmos pedidos.

**Q33.: O que abrevia a letra S do acrónimo SHA1?**
☐ *Safety* ☐ *Safer* ☐ *Security* ☐ *Salt*
☐ *Sardinha* ☒ *Secure* ☐ *Segurança* ☐ *Socket*

**Q34.: O que tem a dizer acerca da seguinte frase?**

*O autor do RC4 é o mesmo do SHA1.*

☐ A frase é verdadeira. ☒ A frase é falsa.

### Tarefa 15 *Task 15*

Numa tentativa de aliciar os estudantes a estudar (passo a redundância), o Professor disse que tinha cifrado o ficheiro *frequencia.pdf* com uma palavra-passe cujo valor de *hash* é
`3733bbd43bd029e3bd660b44a20dac43e9c922e3`

**Q35.: Acha que, sabendo isto, consegue descobrir a palavra-passe ou decifrar o raça da frequência?**

☒ Canja de galinha. ☐ Não, nem pensar...

**Q36.: Acha correto que os autores de sites de *cracking* de valores de *hash* usem a palavra *decrypt* (decifrar) para se referirem ao serviço que disponibilizam?**
☐ Sim, acho.
☒ Não, não acho, nomeadamente porque:

porque decifrar dá a entender que invertem o algoritmo, o que é algo que não fazem.

**Q37.: Caso tenha respondido *canja de galinha* duas questões acima, o que é que recomenda que o Professor faça na próxima vez para proteger melhor o documento?**
☐ Usar um `salt` juntamente com a palavra-chave.
☒ Usar uma palavra-chave ainda mais comprida do que a que foi usada.
☐ Usar uma palavra-chave contendo caracteres especiais.
☒ Usar o `bcrypt` para guardar uma representação segura da palavra-passe, depois de gerar um *salt*.
☐ Não mostrar qualquer informação que possa dizer qual será a palavra-passe aos alunos, como acabou de fazer aqui.

bcrypt -> salt + 100x hash da pass

### Tarefa 16 *Task 16*

Crie um ficheiro chamado `ficheiro.txt` com a letra `b`.

**Q38.: O que é que acontece ao valor de *hash* do ficheiro, se mudar a letra `b` pela letra `c`, no caso em que a função de *hash* utilizada é de qualidade comprovada?**

☐ Muda tudo. ☐ Não muda nada.
☐ Muda exatamente metade.
☒ Mudam aproximadamente metade dos bits de acordo com uma distribuição binomial.

**Q39.: Que nome se dá ao efeito que provoca a alteração mencionada na questão anterior?**

propriedade de dispersão (diffusion)

**Q40.: Só por curiosidade, quantos bits mudaram no ficheiro quando trocou a letra `b` pela letra `c`?**

☐ Nenhum :S. ☒ Só 1. ☐ Foram 2.
☐ 3? ☐ 255! ☐ Peço a ajuda do público.

### Tarefa 17 *Task 17*

Por curiosidade, procure saber se existem comandos disponíveis em Linux para cálculo de valores de *hash*, e anote-os aqui:

md5sum: Calcula e exibe o hash MD5 de arquivos.
Exemplo de uso: md5sum arquivo.txt

sha1sum: Calcula e exibe o hash SHA-1 de arquivos.
Exemplo de uso: sha1sum arquivo.txt

sha256sum: Calcula e exibe o hash SHA-256 de arquivos.
Exemplo de uso: sha256sum arquivo.txt

sha512sum: Calcula e exibe o hash SHA-512 de arquivos.
Exemplo de uso: sha512sum arquivo.txt

## 4 Códigos de Autenticação de Mensagens

*Message Authentication Codes*

### Tarefa 18 *Task 18*

Se ainda não o fez, crie um ficheiro chamado `ficheiro.txt` com a letra `b`. De seguida, calcule o *Hash based Message Authentication Code* (HMAC) para este ficheiro com a função SHA512. **Q41.: Precisa de alguma informação adicional?**

☐ Sim, da f5a3c79b42e64d9185f90f790ad6e3ab.
☐ Não.

**Q42.: Para que é que serve um MAC?**

☒ Para garantir que os dados não são alterados, de forma intencional, enquanto são transmitidos entre duas entidades.
☒ Para garantir que os dados não sofrem erros aleatórios durante uma transmissão.
☐ Para garantir que os dados não são vistos por ninguém que não as entidades com a devida chave de integridade.

**Q43.: Quantas entidades podem calcular um HMAC?**

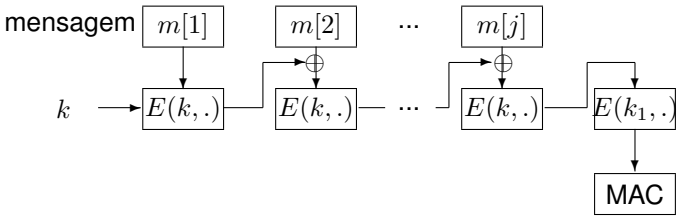
☐ Qualquer entidade envolvida numa comunicação (i.e., emissor, recetor e potenciais intrusos).
☐ Só a entidade que transmite o ficheiro.
☐ Só a entidade que recebe o ficheiro.
☒ Só as entidades que possuem a chave (chave?, mas qual chave?).

**Q44.: Quantas entidades podem calcular o valor de *hash* SHA1 de um ficheiro?** Compare com a resposta que deu na questão anterior.

☒ Qualquer entidade envolvida numa comunicação (i.e., emissor, recetor e potenciais intrusos).
☐ Só a entidade que transmite o ficheiro.
☐ Só a entidade que recebe o ficheiro.
☐ Só as entidades que possuem a chave (chave?, mas qual chave?).

## Tarefa 19 *Task 19*

A figura seguinte ilustra a construção de um *Encrypted Cipher Block Chaining* MAC (ECBC-MAC):




Observe a figura atentamente. **Q45.: Quantas chaves de integridade estão envolvidas na geração de um ECBC-MAC?**

☐ Nenhuma ☐ 1 ☒ 2 ☐ 3 ☐ 4

**Q46.: O *OpenSSL* permite calcular este MAC diretamente?**

☒ Sim, permite. ☐ Não, não permite.

Independentemente da resposta à pergunta anterior, indique uma sequência de comandos que lhe permita calcular este MAC com as funcionalidades de cifra do *OpenSSL*. **Sugestão:** use o comando `tail`. Escreva os comandos nos espaços a seguir:

```
openssl enc -aes-128-cbc -K 0123456789abcdef -iv
0 -in ficheiro.txt | tail -c 16 >> inc-mac.cbc


openssl enc -aes-128-cbc -K abcdef0123456789 -iv
0 -in inc-mac.cbc -out aes-cbc.mac
```

## Tarefa 20 *Task 20*

**Q47.: É possível produzir um MAC com o comando `openssl dgst` que não seja um HMAC?**

☐ Não. O *OpenSSL* só suporta HMACs.
☒ Sim, nomeadamente o MAC produzido com um comando parecido com:

```
openssl dgst ghost mac
```