



1

## Segurança em Sistemas: Ferramentas

- + Na aula de hoje vamos estudar ferramentas para um desenvolvedor mitigar vulnerabilidades, controlar ameaças e prevenir ataques.
- + Tal como um carpinteiro tem as suas ferramentas que utiliza frequentemente, o desenvolvedor terá também uma série de ferramentas/abordagens para a implementação de segurança num sistema de computação
- + Muitas destas medidas já conhecemos
  - + Passwords? Logins? Chaves? etc

2

## Segurança em Sistemas: Autenticação

- + Uma das bases para um sistema de segurança é o controle no acesso
  - + "Alguém tem acesso a algo"
- + Um sistema não tem forma de identificar um indivíduo "visualmente"
  - + Necessita de algum tipo de dados
- + Identificação: Quem é o indivíduo que quer aceder ao sistema
- + Autenticação: Confirmar que o indivíduo é quem afirma ser

Identification is asserting who a person is.  
Authentication is proving that asserted identity.

3

## Segurança em Sistemas: Autenticação vs Identificação

- + A identificação poderá ser facilmente conhecida ou pelo menos previsível
  - + Por exemplo quando recebemos um email, ficamos a conhecer o email do remetente
  - + O email é um dos métodos de identificação mais comuns em diversos sistemas online
  - + Da mesma forma o nosso IBAN é impresso em qualquer recibo de multibanco
- + Por outro lado, a autenticação deverá ser fiável
  - + Se a identificação afirma a identidade de um indivíduo/sistema a autenticação deverá confirmar essa identidade
  - + A autenticação é composta por 1 (ou mais) das seguintes qualidades:
    - + Algo que o utilizador sabe: Password, pin, handshake, nomes, datas de nascimento, etc
    - + Algo que o utilizador é: Dados biométricos, baseados em características do utilizador, impressão digital, reconhecimento de voz ou aparência
    - + Algo que o utilizador tem: distintivos de identificação, chaves físicas, cartões de identificação, etc.

4

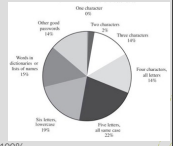
## Segurança em Sistemas: Algo que o utilizador sabe

- ✚ Passwords fornecem um **forma relativamente segura para assegurar a autenticação de um utilizador**
  - ✚ Todos já usamos/implementamos passwords
    - ✚ Se a password corresponder a um valor guardado é garantido o acesso
- ✚ Apesar da sua utilização muito disseminada as passwords apresentam alguns problemas:
  - ✚ **Utilização:** Fornecer uma password para cada acesso pode ser um inconveniente para o utilizador
  - ✚ **Divulgação:** Se o utilizador partilha a password com alguém não autorizado, esse indivíduo ganha acesso imediato ao sistema
  - ✚ **Revogação:** Para retirar o acesso a um utilizador bastará alterar a password
  - ✚ **Perda:** Regra geral se o utilizador perde uma password de acesso será necessária a criação de uma nova

5

## Segurança em Sistemas: Algo que o utilizador sabe

- ✚ Como proteger uma password
  - ✚ As password contém, geralmente, um número reduzido de bits, logo poderão apresentar vulnerabilidades.
    - ✚ Ou pior.... "querty" "password" "123456"
- ✚ Ataques a passwords mais sofisticados incluem:
  - ✚ **Ataques com base em dicionários:** Existem dicionários públicos que ajudam sysadmins a identificarem password fracas.
    - ✚ Estes dicionários poderão também ser utilizados para ataques
  - ✚ **Inferir a password do utilizador:** Se o Filipe escolhe um password, provavelmente não será 100% aleatório. Fará algum sentido para ele.
    - ✚ **Adivinhar password prováveis:** Atacantes que tentam descobrir passwords que geralmente estas possuem características humanas, são "coisas" reais.
      - ✚ Por exemplo, em teoria a probabilidade de um utilizador escolher a password "vlas" ou "mase" é a mesma que escolher "beer". Mas as pessoas tendem a escolher nomes que possam ser associados a "coisas" reais



Think of a word. Is the word you thought of long? Is it uncommon? Is it hard to spell or to pronounce? The answer to all three of these questions is probably no.

6

## Segurança em Sistemas: Algo que o utilizador sabe

- ✚ **Derrotar a Ocultação** : Os sistemas verificam a password introduzida contra um valor guardado
  - ✚ Se um ataque tem como alvo o local onde todas estas passwords estão guardadas. Terá acesso a todas as contas e passwords
  - ✚ Encriptação poderá resolver este problema, desde que bem implementada
- ✚ **Ataque por brute-force:** Neste tipo de ataque o atacante tenta aceder a um sistema testando todas as possíveis combinações de caracteres que formam uma password.
  - ✚ Se considerarmos passwords de 8 caracteres, entre 26 possíveis, existem  $26^8$
  - ✚ Se o sistema permitir passwords inferior a 8 então  $26^1 + 26^2 + 26^3 + 26^4 \dots + 26^8$
  - ✚ Um sistemas moderno pode encontrar a password correta em aproximadamente 1 mês

7

## Segurança em Sistemas: Algo que o utilizador é

- ✚ **Propriedades biológicas**, baseadas em características do corpo humano, por exemplo:
  - ✚ Fingerprints
  - ✚ Geometria da mão
  - ✚ Formato da retina ou iris
  - ✚ Voz
  - ✚ Escrita
  - ✚ ...
- ✚ Apesar do crescimento na utilização dos dados biométricos, investigadores apontam algumas críticas:
  - ✚ A utilização de autenticadores biométricos parecem, ao utilizador, um processo binário (autenticado/não autenticado)
  - ✚ No entanto **existe sempre um trade-off entre sensibilidade e seletividade dos sensores usados**
  - ✚ Além disso os identificadores biométricos **podem mudar ao longo do tempo** (diferente cor do cabelo, peso, etc)
  - ✚ Existem também indivíduos que facilmente conseguem fazer-se passar por terceiros
  - ✚ E indivíduos que têm dificuldades em utilizar dados biométricos



8

## Segurança em Sistemas: algo que o utilizador tem

- + Autenticação baseada em algo que o utilizador "tem" - tokens
  - + Para entrarmos em casa possuímos uma chave
  - + Ou distintivos e cartões de identificação, usados em muitos edifícios
- + Um Token poderá ser **passivo ou ativo**
  - + Um **token passivo** não preforma nenhuma ação, o seu conteúdo nunca muda (por exemplo uma chave)
  - + Um token ativo possui alguma variabilidade na interação.
    - + Por exemplo os **cartões de transportes públicos** - possuem que corresponde ao número de viagens ainda disponíveis, que é reescrito a cada viagem

9

## Segurança em Sistemas: algo que o utilizador tem

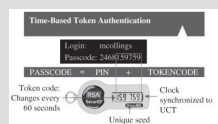
- + Tokens poderão também ser **dinâmicos ou estáticos**.
  - + **Tokens estáticos**: Os valores destes tokens mantêm-se, cartões de identificação, passaportes.
    - + Mais úteis para autenticações *no local*, por exemplo o "agente" verifica o CC e a foto
  - + **Tokens dinâmicos**: Neste caso o **valor da token pode mudar**
    - + Regra geral estes tokens são dispositivos que **geram valores aleatórios**
    - + Utilizados para autenticação remota (por exemplo em sistemas bancários)



10

## Segurança em Sistemas: algo que o utilizador tem

- + Tokens **são vulneráveis a ataques** como o *skimming*
  - + A utilização de algum dispositivo com o objetivo de copiar os dados de autenticação do token
  - + ATMs são muito vulneráveis a este tipo de ataques
- + Tokens dinâmicos tentam mitigar este problema
  - + Gera valores imprevisíveis que poderão ser utilizados para confirmar a identidade de um utilizador



11

## Segurança em Sistemas: algo que o utilizador ...

- + Sistemas mais avançado utilizam técnicas como
  - + **Multifactor-Authentication**
    - + AS abordagens discutidas até agora possuem algumas desvantagens
      - + Um *token* só funciona se não for partilhado, uma password só tem valor se for mantida secreta
    - + A abordagem **Multifactor** ou **Two-factor authentication** tenta mitigar a desvantagens de técnicas individuais
      - + Combinando-as
      - + Por exemplo o cartão de cidadão é um token de autenticação
        - + Mas a autenticação da identidade também é garantida pela foto, e pela assinatura

12

## Segurança em Sistemas: Criptografia

- + Além da autenticação outra poderá ser útil **esconder a informação para indivíduos não autenticados**
- + Importante quando os meios de comunicação são públicos
  - + E.g., Internet
- + Em criptografia uma mensagem **poderá ser enviada "à vista de todos"**
  - + Mas apenas o seu destinatário conseguirá compreendê-la

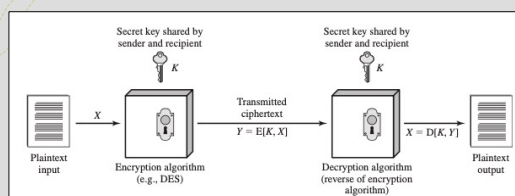
13

## Segurança em Sistemas: Criptografia

- + Uma das bases em qualquer sistema de computação é **garantir a confidencialidade dos dados transmitidos ou guardados**.
- + Para começar este aspeto mais prático de segurança de sistemas vamos considerar a **criptação simétrica**
  - + Também conhecida como **criptação convencional**, ou **criptação por chaves simples**
  - + Muito usada desde os tempos de Julio César, até aos U-boats na segunda guerra mundial
  - + Até **sistemas militares ou diplomáticos dos dias de hoje**

14

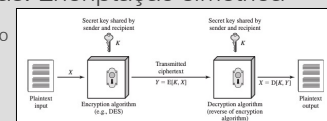
## Segurança em Sistemas: Criptografia



15

## Segurança em Sistemas: Criptação simétrica

- + A **criptação simétrica** segue o esquema apresentado na imagem à direita



- + **Entrada:** Conteúdo original, mensagem ou dados que são dados ao algoritmo:
- + **Algoritmo de encriptação:** Algoritmo que realiza diversas alterações e substituições ao conteúdo
- + **Chave secreta:** Também input do algoritmo. As mudanças e substituições feitas ao conteúdo original dependem desta chave
- + **Conteúdo cifrado (encriptado):** Mensagem "revolvida" resultante do algoritmo e chave.
- + **Algoritmo de desencriptação:** Basicamente é a execução reversa do algoritmo de encriptação. Recebe o conteúdo encriptado e a chave secreta, e retorna o conteúdo desencriptado

16

## Segurança em Sistemas: Encriptação simétrica

- + Existem **dois requisitos principais** na utilização de encriptação simétrica
  - + O algoritmo terá que ser bom.
    - + Por exemplo, mesmo que o código do algoritmo seja descoberto, um possível *hacker* não consiga decifrar a mensagem sem acesso à chave secreta
    - + Da mesma forma um hacker não deverá ser capaz de *reverse-engineer* o algoritmo com base no conteúdo encriptado e conteúdo descriptado
  - + O destinatário e remetente de uma mensagem encriptada deverão ter **obtido cópias da chave secreta de uma forma segura**.
    - + E estas deverão ser mantidas de uma forma segura

17

## Segurança em Sistemas: Encriptação simétrica - vulnerabilidades

- + Uma abordagem de encriptação simétrica é **suscetível a dois tipos de ataques**
  - + **Criptanálise**: Tenta, com base na natureza do algoritmo e quais inputs, ou pares, inputs/conteúdo encriptado, deduzir a chave ou outros pontos do algoritmo.
    - + Se um ataque é bem sucedido na criptanálise, **toda a segurança fica comprometida**
  - + **Força-Bruta**: Começando com o conteúdo encriptado, tentar todas as combinações de chaves secretas até que o output do algoritmo de desencriptação seja "legível"



18

## Segurança em Sistemas: Encriptação simétrica - algoritmos

- + Os algoritmos de encriptação simétrica mais comuns são **cifras de bloco (block ciphers)**
  - + **Processa o input em blocos com um tamanho fixo**, e cada bloco encriptado tem o mesmo tamanho que o input
- + Os algoritmos mais comuns são o **Data Encryption Standard (DES)** o **Triple DES** e o **Advanced Encryption Standard (AES)**

19

## Segurança em Sistemas: Encriptação simétrica - DES

- + Até recentemente o **DES foi o algoritmo mais usado**
  - + Recebe um bloco de 64 bits e uma chave de 56 bits para produzir um bloco encriptado de 64 bits
  - + O tamanho da chave é uma das vulnerabilidades de segurança do DES
    - + Existem  $2^{56}$  possíveis chaves
    - + Considerando a velocidade dos CPUs atuais um computador pode quebrar um algoritmo DES em pouco mais de um ano
      - + Este valor é muito mais reduzido quando consideramos uma abordagem paralela com mais sistemas
    - + Chaves maiores (e.g. 128 bits) tornariam uma ataque por força bruta efetivamente impraticável
  - + Por outro lado, dada a sua popularidade, o DES é um dos algoritmos mais estudados
    - + E apesar de inúmeras tentativas ainda não foi explorada nenhuma vulnerabilidade utilizando uma abordagem de criptanálise

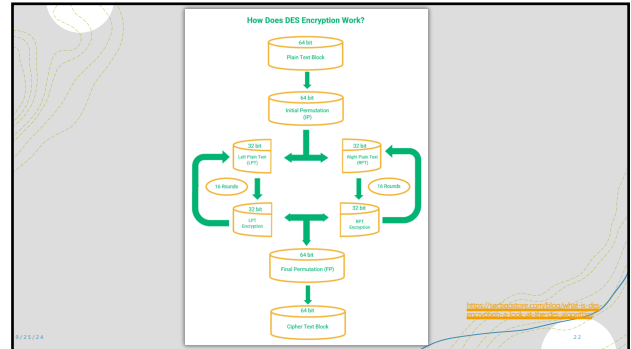
20

## Segurança em Sistemas: Encriptação simétrica - DES

- + O DES utiliza operações lógicas e aritméticas em dados binários até 64 bits
- + A encriptação ocorre em 16 passos
  - + Cada passo substitui e "baralha" os dados do input com base nos valores da chave
  - + Este processo utiliza uma tabela e é repetitivo
    - + Facilitando a implementação num chip criado à medida
    - + o DES é considerado uma Feistel Cipher



21



22

## Segurança em Sistemas: Encriptação simétrica - 3DES

- + O DES foi estendido através do triple DES (3DES)
  - + Proposto em 1985
- + Repete o algoritmo DES 3 vezes
  - + Com 2 ou 3 chaves secretas diferentes
- + Possui 2 principais vantagens em relação ao DES
  - + Chave de 168 (56\*3) bits mitiga o risco de um ataque por força bruta
  - + Estende a capacidade de resistência à criptoanálise do DES
- + Se assumirmos uma encriptação com o DES representada como:
  - +  $C = E(k, m)$  em que  $m$  é a mensagem,  $k$  é a chave, e  $E$  é o algoritmo
  - + O 3DES seria:  $C = E(k_3, E(k_2, E(k_1, m)))$ , onde o mesmo algoritmo  $E$ , é aplicado à mesma mensagem  $m$ , com 3 chaves diferentes  $k_1, k_2, k_3$

توضيح: DES هو خوارزمية تشفير متماثل تستخدم مفتاح سري واحد. 3DES هو امتداد لـ DES يستخدم ثلاثة مفاتيح سريّة مختلفة (أو اثنين) لتطبيق عملية التشفير ثلاث مرات على النص الأصلي. هذا يزيد من أمان النظام ويمنحه مقاومة أفضل لهجمات القوة الغاشية.

23

## Segurança em Sistemas: Encriptação simétrica - 3DES

- + A principal desvantagem do 3DES relaciona-se com a sua "idade"
  - + Estamos a repetir o DES 3 vezes
    - + Se considerarmos que o DES foi desenvolvido a "um nível baixo" nos anos 70, a sua implementação em sistemas modernos não é a mais eficiente
  - + Além disso, visto que tanto o DES como o 3DES utilizam blocos de 64bits a eficiência de encriptação/desencriptação não é a melhor
    - + Blocos com tamanho superior tornariam a encriptação/desencriptação mais rápida

توضيح: DES هو خوارزمية تشفير متماثل تستخدم مفتاح سري واحد. 3DES هو امتداد لـ DES يستخدم ثلاثة مفاتيح سريّة مختلفة (أو اثنين) لتطبيق عملية التشفير ثلاث مرات على النص الأصلي. هذا يزيد من أمان النظام ويمنحه مقاومة أفضل لهجمات القوة الغاشية.

24

## Segurança em Sistemas: Encriptação simétrica - AES

- As desvantagens do 3DES fazem com que **não sejam um candidato apropriado para utilização no longo termo**.
- Em 1997 foi criado um concurso de propostas para o **Advanced Encryption Standard**
- Deverá ser **tão seguro como o 3DES e significativamente mais eficiente**
  - Utilização de **blocos de 128 bits e chaves de 128, 192 e 256 bits**
- Em Novembro de 2001 o **algoritmo foi publicado** e é atualmente amplamente utilizado em produtos comerciais

25

## Segurança em Sistemas: Encriptação simétrica - Performance

Key size (bits)	Cipher	Number of Alternative Keys	Time Required at $10^9$ decryptions/ $\mu$ s	Time Required at $10^{13}$ decryptions/ $\mu$ s
56	DES	$2^{56} \approx 7.2 \times 10^{16}$	$2^{55} \mu s = 1.125$ years	1 hour
128	AES	$2^{128} \approx 3.4 \times 10^{38}$	$2^{127} \mu s = 5.3 \times 10^{21}$ years	$5.3 \times 10^{17}$ years
168	Triple DES	$2^{168} \approx 3.7 \times 10^{50}$	$2^{167} \mu s = 5.8 \times 10^{33}$ years	$5.8 \times 10^{29}$ years
192	AES	$2^{192} \approx 6.3 \times 10^{57}$	$2^{191} \mu s = 9.8 \times 10^{40}$ years	$9.8 \times 10^{36}$ years
256	AES	$2^{256} \approx 1.2 \times 10^{77}$	$2^{255} \mu s = 1.8 \times 10^{69}$ years	$1.8 \times 10^{56}$ years

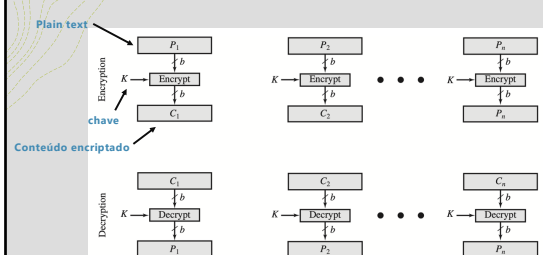
26

## Segurança em Sistemas: Encriptação simétrica - desafios

- Os algoritmos discutidos aplicam a **encriptação em blocos de 64 a 128 bits**
- No nosso dia a dia utilizamos informação com tamanho muito superior
  - Emails, mensagens, documentos, pacotes numa rede, etc
- Terão que ser divididos em blocos para a encriptação simétrica**
- O método mais comum para encriptar diversos blocos é o **ECB Electronic Codebook**
- Uma mensagem em *plaintext* é tratada *b* bits de cada vez (*b*=tamanho do bloco)
  - O conteúdo em cada bloco de *b* bits é encriptado utilizando a mesma chave
  - A desencriptação segue o processo inverso

27

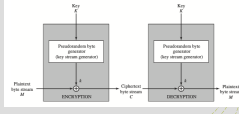
## Segurança em Sistemas: Encriptação simétrica - desafios



28

### Segurança em Sistemas: Encriptação simétrica - Cifras de fluxo – stream ciphers

- + A cifras de bloco processam o input em blocos 😊
- + Podem existir aplicações em que é necessário um processamento contínuo de dados
- + Podemos utilizar cifras de fluxo – *stream ciphers* nestes casos
- + Geralmente encriptam o conteúdo **um byte de cada vez**
- + Apesar de existirem variações a este padrão
- + É utilizada uma chave num *pseudo-random bit generator* que produz um fluxo (aparentemente aleatório)
- + O output deste processo é um **keystream**
  - + O resultado irá combinar o keystream com o conteúdo plaintext através de um XOR



29

### Segurança em Sistemas: Encriptação simétrica - Cifras de fluxo – stream ciphers

- + Estudos revelaram que uma cifra de fluxo poderá ser tão segura como uma cifra de bloco (assumindo chaves do mesmo tamanho)
- + Outras vantagens:
  - + Serão geralmente mais rápidas que cifras de bolos
  - + Úteis para streams...
    - + Em canais de comunicação por exemplo...
- + Mas o facto é que qualquer tipo de cifra poderá ser utilizada na grande maioria dos cenários
- + Desde que implementadas de forma competente

30

### Um exemplo prático

31

### Segurança em Sistemas: Encriptação simétrica – Um caso prático

- + Vamos implementar um **exemplo prático de encriptação simétrica**
- + Utilizando o algoritmo PA
  - + Muito fraco, desenvolvido apenas para esta aula
  - + Com a chave "!"#\$%&'()\*=?\*^&^\_ :;MNBVCDS><"
- + E depois o DES e AES
- + Neste exemplo a **comunicação é feita através de um ficheiro** que será gravado por um programa e lido por outro
  - + Encriptado antes da gravação e descriptado após a leitura
  - + Na prática vamos criar uma função que lê um ficheiro encripta o conteúdo e grava-o com outro nome
  - + E outra função que lê um ficheiro encriptado descripta-o e imprime na consola

32



## Segurança em Sistemas: Encriptação simétrica – Um caso prático

```
private static String key = "!@#$%^&*()=2^*^P^_!MNBVCDS><";
public static void encryptFile(String originFile, String destinationFile) {
    File original = new File(originFile);
    Scanner reader = null;
    try {
        reader = new Scanner(original);
        String encryptedContent = "";
        while (reader.hasNextLine()) {
            String input = reader.nextLine();
            encryptedContent = encryptedContent + "\n" + encrypt(input);
        }
        //System.out.println(encryptedContent);
        FileWriter encryptedFile = new FileWriter(destinationFile);
        encryptedFile.write(encryptedContent);
        encryptedFile.close();
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    }
}

private static String encrypt(String block) {
    String encryptedBlock = "";
    block = block.toLowerCase();
    for (int i = 0; i < block.length(); i++) {
        int index = (int) block.charAt(i);
        if (index == 32) {
            encryptedBlock = encryptedBlock + key.charAt(27); // o
        } else {
            encryptedBlock = encryptedBlock + key.charAt(index - 97);
        }
    }
    //System.out.println(encryptedBlock);
    return encryptedBlock;
}
```

33

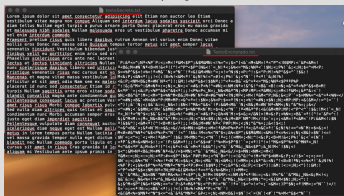
## Segurança em Sistemas: Encriptação simétrica – Um caso prático

- + O algoritmo de encriptação PA é muito simples
- + **Converte o conteúdo para minúsculas**
- + **Encripta linha a linha (cifra de "bloco")**
  - + Substituindo o conteúdo carater a carater com um valor da chave
  - + Neste caso **usamos o código ascii de cada carater de origem para selecionar um elemento da chave**
    - + Ao índice subtraímos o valor 97, para criar um índice que começa em 0
    - + Para o carater " " foi adicionado um caso especial
  - + A função de desencriptação realiza o processo inverso
- + Este algoritmo não deveria ser usada em nenhum tipo de sistema seguro
- + Contém vários problemas
- + **TPC: Identificar 4 problemas com este algoritmo**

34

## Segurança em Sistemas: Encriptação simétrica – Um caso prático

- + O algoritmo de encriptação PA é muito simples
- + Mas permite-nos observar a encriptação



35

## Segurança em Sistemas: Encriptação simétrica – Um caso prático

- + A utilização dos algoritmos DES e AES é semelhante
- + Existem diversas **bibliotecas que permitem a utilização destes algoritmos**
- + Por exemplo:
- + Criação de chaves aleatórias:

```
Random random = new Random();
byte[] iv = new byte[128/8]; //16 bytes
random.nextBytes(iv);
IvParameterSpec ivspec = new IvParameterSpec(iv);
KeyGenerator kgen = null;
kgen = KeyGenerator.getInstance("AES");
SecretKey skey = kgen.generateKey();
```

ou com base numa String:

```
key = "MinhaChaveTeste";
DESKeySpec dks = new DESKeySpec(key.getBytes());
SecretKeyFactory skf =
    SecretKeyFactory.getInstance("DES");
deskey = skf.generateSecret(dks);
des_cipher = Cipher.getInstance("DES");
des_cipher.init(mode, deskey);
```

36

## Segurança em Sistemas: Encriptação simétrica – Um caso prático

### + Gravação de ficheiro encriptado (AES):

```
FileOutputStream out = new
FileOutputStream(destinationFile);
//byte[] contentB = new byte[content.length()];
byte[] input = content.getBytes("UTF-8");
byte[] encoded = ci.doFinal(input);
out.write(encoded);
```

- + ci é um objeto da classe Cipher, que terá de ser inicializado com uma chave secreta e tipo de algoritmo
- + skey é criada no slide anterior

```
ci = Cipher.getInstance("AES/CBC/PKCS5Padding");
ci.init(Cipher.ENCRYPT_MODE, skey, ivspec);
```

37

PROGRAMAÇÃO AVANÇADA | UMA 23/24

## Segurança em Sistemas: Encriptação simétrica – Um caso prático

### + Leitura de ficheiro encriptado (AES):

```
String keyFile = "/Users/filipequintal/Desktop/key";
FileInputStream key = new FileInputStream(keyFile);
byte[] keyb = key.readAllBytes();
SecretKeySpec stored_key = new SecretKeySpec(keyb, "AES");
```

Neste exemplo, a chave tinha sido guardada no Desktop e carregada – Na realidade esta chave deve ser guardada num local seguro

O objeto ci terá de ser redefinido para leitura

```
ci.init(Cipher.DECRYPT_MODE, stored_key, ivspec);
byte[] encryptedContent = Files.readAllBytes(Paths.get(encryptedFile));
String text = new String(ci.doFinal(encryptedContent), "UTF-8");
```

Conteúdo encriptado em plain text

Gravação da chave

```
String key = "/Users/filipequintal/Desktop/key";
FileOutputStream out = new FileOutputStream(key);
byte[] keybytes = skey.getEncoded();
out.write(keybytes);
```

38

PROGRAMAÇÃO AVANÇADA | UMA 23/24

## Segurança em Sistemas: Encriptação simétrica – Um caso prático

+ Todos estes exemplos serão explorados nas próximas aulas práticas

39

PROGRAMAÇÃO AVANÇADA | UMA 23/24

## Segurança em Sistemas: Bibliografia

- + Computer Security: Principles and Practice, 4th Edition, William Stallings, Lawrie Brown, 2018
  - + 2. Cryptographic Tools
    - + 2.1 Confidentiality with Symmetric Encryption
  - + 3. User Authentication
    - + 3.1 Electronic User Authentication Principles
    - + 3.2 Password-Based Authentication
    - + 3.3 Token-Based Authentication
    - + 3.4 Biometric Authentication
- + Security in Computing, Charles Pfleeger, Shari Pfleeger, Jonathan Margulies, 2015
  - + 2.1 Authentication
  - + 2.3 Cryptography

40

PROGRAMAÇÃO AVANÇADA | UMA 23/24