Tópicos de Segurança

Ano letivo 2019/2020

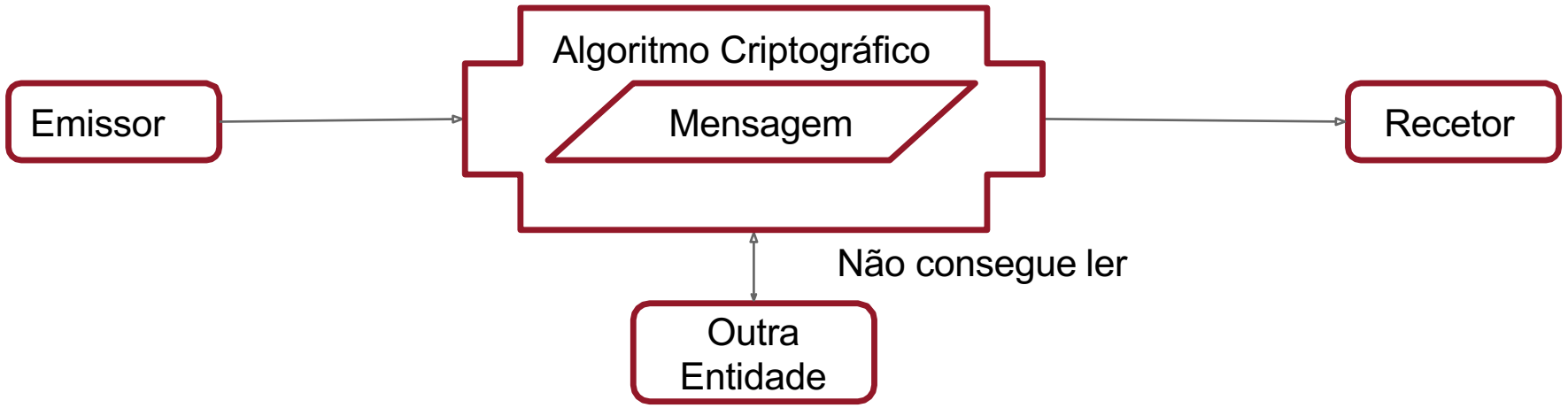
Cofinanciado por:

POCH PORTUGAL 2020 UNIÃO EUROPEIA Fundo Social Europeu

IPL instituto politécnico de leiria

Criptografia, o que é?

A criptografia é a escrita e estudo de informação codificada de forma secreta.



Cofinanciado por:

Terminologia (1)

- Criptografia Forte
  - Difícil de reverter o processo sem a chave
  - Está dependente apenas da chave e não do conhecimento do algoritmo usado
  - O poder computacional necessário para decifrar a informação é tal que torna a tarefa impraticável
    - o poder computacional existente não permite reverter o processo, sem a chave, durante o período de validade da informação
  - ou o custo de quebrar a cifra é mais elevado que o valor da informação
  - Exemplo: os documentos dos governos têm de permanecer secretos (para o público geral) durante 25 anos

Cofinanciado por: POCH PORTUGAL 2020 UNIÃO EUROPEIA Fundo Social Europeu

Terminologia (2)

- Cifragem
  - Transformação reversível dos dados por forma a torná-los ininteligíveis
  - Algoritmos de Criptografia ou Cifra
    - Funções matemáticas utilizadas para realizar as operações
    - Os algoritmos são públicos. A segurança baseia-se nas chaves utilizadas.
- Alternativas
  - Cifragem simétrica (ou de chave secreta)
  - Cifragem assimétrica (ou de chave pública)

- Criptoanálise - Utilização de várias técnicas de análise estatística com o objetivo de obter o texto e/ou a chave de uma mensagem cifrada

Cofinanciado por: POCH PORTUGAL 2020 UNIÃO EUROPEIA Fundo Social Europeu

Algoritmos

- Simétricos
  - Secret Key
  - Exemplos: AES; DES; TripleDES

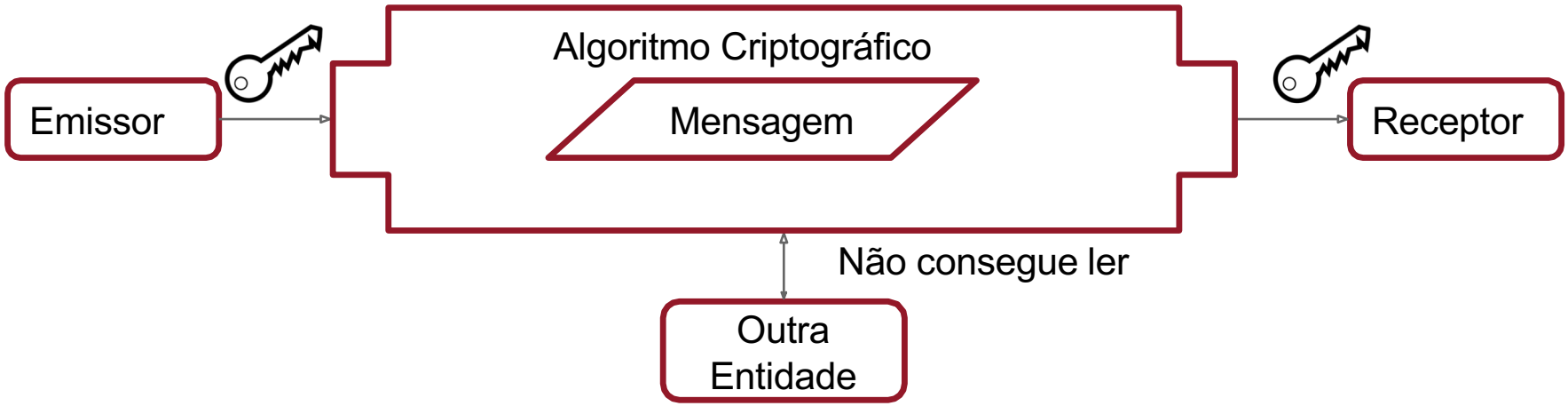
- Assimétricos
  - Public Key Cryptography
  - Exemplos: RSA; DAS; El Gamal
- Hash
  - One-Way
  - Exemplos: MD5; SHA (1,2,3)

Cofinanciado por: POCH PORTUGAL 2020 UNIÃO EUROPEIA Fundo Social Europeu

Criptografia simétrica (1)

A criptografia simétrica é o método convencional utilizado na troca de informação segura.

Existe uma chave partilhada entre emissor e recetor utilizada para cifrar/decifrar os dados.



O ato de cifrar é simétrico (inverso) ao ato de decifrar!

Cofinanciado por: POCH PORTUGAL 2020 UNIÃO EUROPEIA Fundo Social Europeu

Criptografia simétrica (2)

Vantagens:

- Eficientes (menos pesados computacionalmente);
- Facilmente implementáveis em hardware.

Desvantagens:

- Dificuldade na distribuição das chaves de forma segura;
- Menos resistente a técnicas de criptoanálise.
- Quantidade de chaves necessárias para permitir comunicações arbitrárias entre (N) interlocutores. Número de chaves = N x (N – 1) / 2.
- Exemplos: Entre 4 PCs existem 6 chaves diferentes. Entre 10 PCs existem 45 chaves diferentes.

Cofinanciado por: POCH PORTUGAL 2020 UNIÃO EUROPEIA Fundo Social Europeu

Criptografia simétrica (3)

Código em c# de exemplo:

Cifrar:

```
using (AesCryptoServiceProvider aes = new AesCryptoServiceProvider())
{
                    //Guardar e Partilhar key e IV  -> Terão que ser enviados para o servidor
                    this.key = aes.Key;
                    this.iv = aes.IV;

                    // Cifrar dados

                    // 1º - Converter os dados em bytes
                    byte[] msgEmBytes = Encoding.UTF8.GetBytes(textBoxEncrypt.Text);

                    // 2º - Criar variável para colocar o conteúdo da msg decifrada
                    byte[] msgCifradaEmBytes;

                    // 3º - Aplicar o algoritmo criptografico
                    using (MemoryStream ms = new MemoryStream())
                    {
                                        using (CryptoStream cs = new CryptoStream(ms, aes.CreateEncryptor(), CryptoStreamMode.Write))
                                        {
                                                            cs.Write(msgEmBytes, 0, msgEmBytes.Length);
                                        }
                                        msgCifradaEmBytes = ms.ToArray();
                    }
                                        // 4º - Mostrar dados cifrados
                                        textBoxEncrypted.Text = Convert.ToBase64String(msgCifradaEmBytes);
                    }
}
```


Cofinanciado por:

Criptografia simétrica (4)

Código em c# de exemplo:

Decifrar:

```
using (AesCryptoServiceProvider aes = new AesCryptoServiceProvider())
{
                    //Importar key e IV
                    aes.key = receivedKey;
                    aes.iv = receivedIV;

                    // Decifrar dados

                    // 1º - Converter os dados cifrados em bytes
                    byte[] msgCifradaEmBytes = Convert.FromBase64String(msgCifrada);

                    // 2º - Criar variável para colocar o conteúdo da msg decifrada
                    byte[] msgDecifradaEmBytes = new byte[msgCifradaEmBytes];
                    int bytesLidos = 0;

                    // 3º - Aplicar o algoritmo criptografico
                    using (MemoryStream ms = new MemoryStream())
                    {
                                        using (CryptoStream cs = new CryptoStream(ms, aes.CreateEncryptor(), CryptoStreamMode.Read))
                                        {
                                                            bytesLidos = cs.Read(msgDecifradaEmBytes, 0, msgDecifradaEmBytes.Length);
                                        }
                                        msgDecifradaEmBytes = ms.ToArray();
                    }
                                        // 4º - Mostrar dados decifrados
                                        textBoxEncrypted.Text = Encoding.UTF8.GetString(msgDecifradaEmBytes, 0, bytesLidos);
                    }
}
```

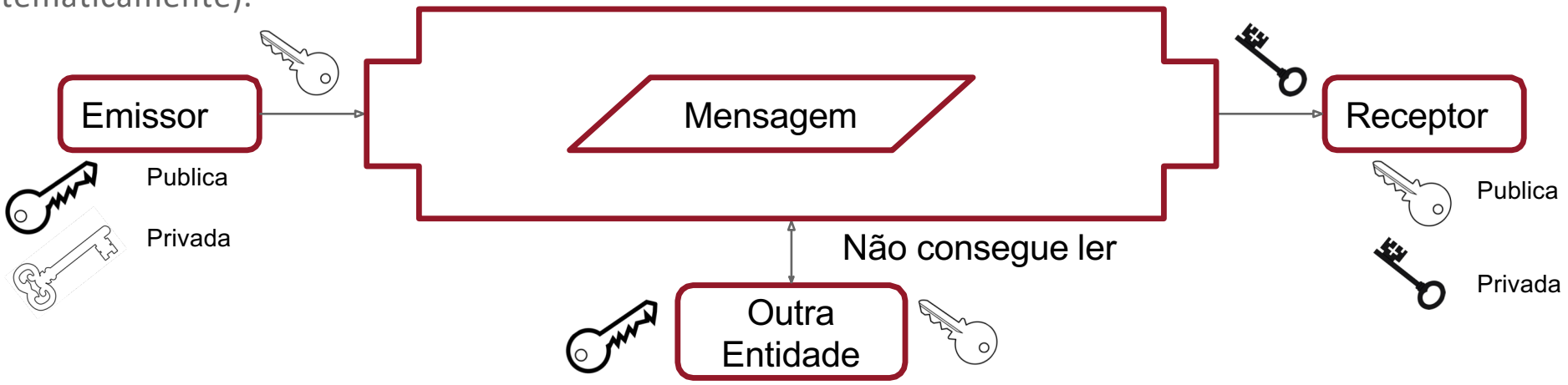

Cofinanciado por:

Criptografia assimétrica (1)

A criptografia assimétrica utiliza um par de chaves (segredos) para cifrar/decifrar os dados.

- Existe uma chave pública que é conhecida por todos e uma chave privada que apenas o "dono" conhece.
- Cifrar com uma chave obriga a decifrar com a outra (assimétrico).

Não é possível descobrir a chave privada a partir da pública, mas o inverso sim (as chaves estão relacionadas matematicamente).



Cofinanciado por: POCH PORTUGAL 2020 UNIÃO EUROPEIA Fundo Social Europeu

Criptografia assimétrica (2)

Vantagens:

- Resolve o problema da distribuição das chaves que existia nos algoritmos simétricos;
- Além da confidencialidade e integridade, permitem efetuar a autenticação do emissor e garantir o não repúdio.
- A quantidade de chaves necessárias para permitir comunicações arbitrárias entre (N) interlocutores é (2 x N). Por exemplo: Entre 10 PCs, existem 20 chaves diferentes.

Desvantagens:

- Muito pesado computacionalmente;
- Mais complexa, logo mais difícil de implementar.

Cofinanciado por: POCH PORTUGAL 2020 UNIÃO EUROPEIA Fundo Social Europeu

Criptografia assimétrica (3)

Código em c# de exemplo:

```
// 1º - Inicializar RSA e guardar as chaves se oportuno

RSACryptoServiceProvider rsa = new RSACryptoServiceProvider();

string publickey = rsa.ToXmlString(false); // Chave Pública
string bothkeys = rsa.ToXmlString(true); // Chave Privada + Pública
```

Cifrar:

```
// 1º - Converter dados para byte[]
string msg = "Exemplo";
byte[] dados = Encoding.UTF8.GetBytes(msg);

// 2º - Cifrar os dados e guardá-los numa variável
byte[] dadosEnc = rsa.Encrypt(dados, true);

// 3º - Apresentar os dados em Base64
textBox.Text = Convert.ToBase64String(dadosEnc);
```

Decifrar:

```
// 1º - Converter dados de Base64 para byte[]
byte[] dados = Convert.FromBase64String(msg);

// 2º - Decifrar os dados e guardá-los numa variável
byte[] dadosDec = rsa.Decrypt(dados, true);

// 3º - Apresentar os dados
tbSymmetricKeyDecrypted.Text = Encoding.UTF8.GetString(dadosDec);
```


Cofinanciado por: POCH PORTUGAL 2020 UNIÃO EUROPEIA Fundo Social Europeu

Hashing (1)

O hashing (ou criar uma hash) serve para duas coisas:

- Validar a integridade de algo;
- Validar uma *password* durante uma autenticação.

É apenas "one-way", ou seja, é virtualmente impossível reverter o resultado.

O tamanho é independente dos dados.

Por exemplo (com recurso ao algoritmo MD5):

MD5("Tópicos de **S**egurança") = 6818320d10fe14421d3f01e68ec8e6b1

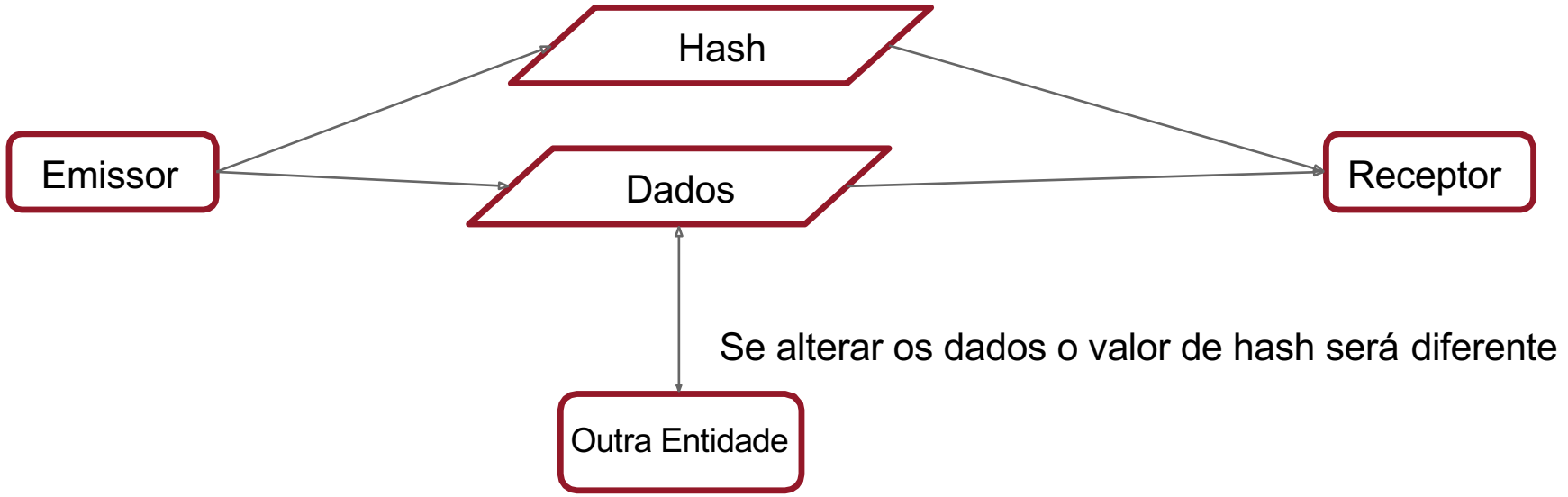
MD5("Tópicos de **s**egurança") = 4a21939d79c183a3cdb5d0f7bdd3a047

Cofinanciado por: POCH PORTUGAL 2020 UNIÃO EUROPEIA Fundo Social Europeu

Hashing (2) – Validação de Integridade

Algoritmos mais comuns:

- MD5 (128 bit)
- SHA1 (160 bit)
- SHA2 (256, 384 ou 512 bit)



Cofinanciado por:

Hashing (2) – Validação de Integridade

Código em c# de exemplo:

```
using (SHA512 sha = SHA512.Create()) {

        // 1º - Converter dados para byte[]
        byte[] data = Encoding.UTF8.GetBytes(tbData.Text);

        // 2º - Calcular o valor de hash
        byte[] hashBytes = sha.ComputeHash(data);

        // 3º - Converter o resultado de byte[] para hexadecimal
        string hash = BitConverter.ToString(hashBytes);
}

// Nota: Pode-se substituir a classe SHA512 por MD5, SHA1, SHA256, etc...
```


Cofinanciado por: POCH PORTUGAL 2020 UNIÃO EUROPEIA Fundo Social Europeu

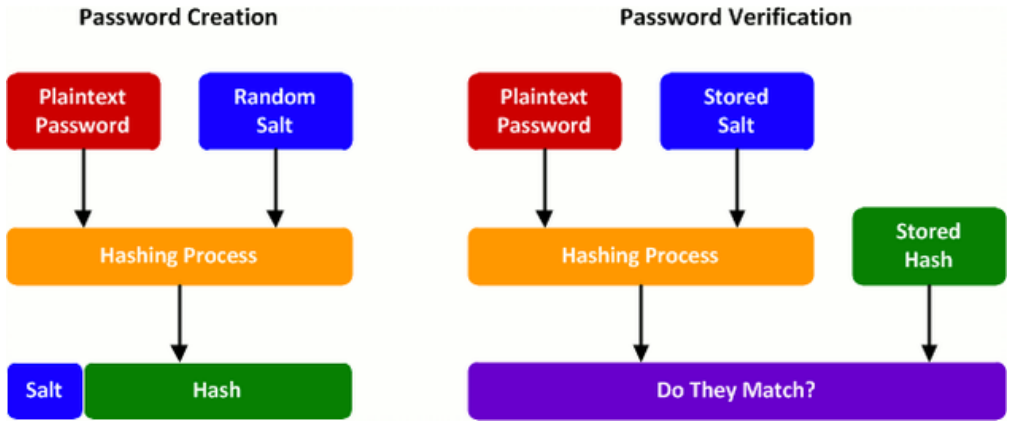
Hashing (3) – Autenticação

Deverá ser gerada e adicionada uma *string* aleatória e única (SALT) à *password.*

Username	Hash (MD5)
john98	5f4dcc3b5aa765d61d8327deb882cf99
master	d8578edf8458ce06fbc5bb76a58c5ca4
timmy	5f4dcc3b5aa765d61d8327deb882cf99
lopez	d8578edf8458ce06fbc5bb76a58c5ca4

passwords iguais

Username	Salt	Salted Hash (MD5)
john98	QxLUF1bgIAdeQX	056da86e4b284a13fcfcab44bdfb8cc9
master	bv5PehSMfV11Cd	28a0f959d2bf561ce0a0385d7fab56c9
timmy	YYLmfY6IehjZMQ	77653244c6961112e659dc6bf496340c
lopez	9fE5eF3b08Ha75	8ebe4950a721a6a307d98b58afa4c005

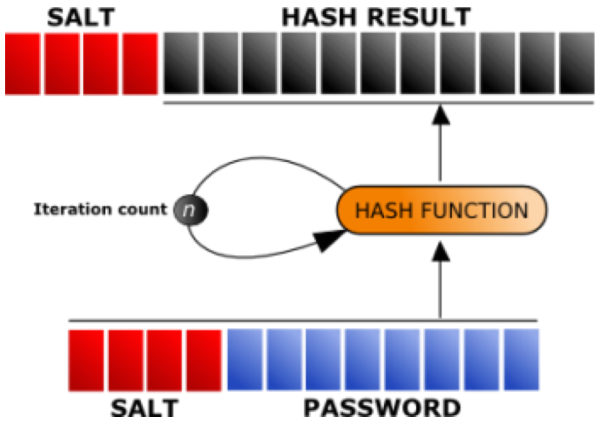


Cofinanciado por: POCH PORTUGAL 2020 UNIÃO EUROPEIA Fundo Social Europeu

Hashing (4) – Autenticação

Deverá ser utilizado um algoritmo cuja execução é muito lenta.

- O objetivo é tornar os ataques de força-bruta inúteis.
- Exemplos:
  - Argon2
  - PBKDF2
  - scrypt
  - Brcrypt
- Estes exemplos possuem a capacidade de serem ajustados ("iteration count")
  - Assim, o *hashing* da *password* pode demorar o tempo que se achar necessário



Cofinanciado por: