# Tópicos de Teoria da Informação e de Teoria da Complexidade

MICEI/MSDPA

José Carlos Bacelar Almeida
(jba@di.uminho.pt)

## Avaliação da Segurança

- **Segurança Incondicional**

  A segurança do sistema criptográfico é avaliada sem se colocar qualquer limite na capacidade/recursos computacionais do intruso

  - Análise estatística (teoria da informação)

- **Segurança Computacional**

  A segurança do sistema criptográfico é avaliada impondo limites razoáveis na capacidade/recursos computacionais do intruso

  - Introduz aspectos da teoria da complexidade (e.g. redução a problemas tidos por intratáveis) no argumento de segurança.

# Teoria da Informação

- Teoria desenvolvida em 1947/49 por Claude Shannon.

> The fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another point. Frequently the messages have meaning; that is, they refer to or are correlated to some system with certain physical or conceptual entities. These semantic aspects of communication are irrelevant to the enginnering problem. The significant aspect is that the actual message is one *selected* from a set of possible messages.

- Comunicação é modelizada como um teste probabilístico sobre um canal $Y$ (variável aleatória), i.e. recepção de uma mensagem é identificada com a observação estatística $Y=y$. Admite-se que a mensagem é codificada numa linguagem bem definida ($L_Y$) com função de distribuição de probabilidade conhecida ($P_Y$).

# Da teoria das probabilidades...

- Varáveis aleatórias; funções de distribuição de probabilidades; probabilidades múltiplas; probabilidades condicionais.

$$P(x,y)=P(x|y)*P(y)$$

- Fórmula de Bayes

$$P(x \mid y) = \frac{P(x) * P(y \mid x)}{P(y)}$$

- Exercício: Demonstre que $P(x) = \sum_y P(x,y)$

# Confidencialidade

- Considerem-se as seguintes variáveis aleatórias:
  - T – **texto limpo -** ($t \in L_T$) com função de distribuição de probabilidades $P_T$.
  - K – **chaves** - ($k \in L_K$) com função de distribuição de probabilidades $P_K$

  (assume-se que T e K são variáveis independentes)
- Podemos considerar agora a variável **C** (criptograma). Como $c=e_k(t)$ podemos considerar o conjunto: $C(k)=\{e_k(t) : t \in L_T\}$. Assim definimos:

$$P_C(c) = \sum_{k:y \in C(k)} P_K(k) * P_T(d_k(c))$$

# Confidencialidade

- Podemos observar que:

$$P_C(c \mid t) = \sum_{k:t=d_k(c)} P_K(k)$$

- Daqui retiramos (fórmula Bayes):

$$P_C(t \mid c) = \frac{P_T(t) * \sum_{k:t=d_k(c)} P_K(k)}{\sum_{k:c \in C(k)} P_K(k) * P_P(d_k(c))}$$

# Exemplo

- $L_T:P_T$ = { a:1/4 ; b:3/4 }
- $L_K:P_K$ ={ k1:1/2 ; k2:1/4 ; k3:1/4}
- Cifra

| | a | b |
|---|---|---|
| $k_1$ | 1 | 2 |
| $k_2$ | 2 | 3 |
| $k_3$ | 3 | 4 |

$P_C(1)=1/8$ $P_C(2)=3/8*1/16=7/16$ $P_C(3)=3/16*1/16=1/4$ $P_C(4)=3/16$

- $P_T(a|1) = 1$ $P_T(b|1) = 0$
- $P_T(a|2) = 1/7$ $P_T(b|2) = 6/7$
- $P_T(a|3) = 1/4$ $P_T(b|3) = 3/4$
- $P_T(a|4) = 0$ $P_T(b|4) = 1$

# Cifra Perfeita

- Dizemos que um sistema criptográfico é uma **cifra perfeita** quando, para qualquer texto limpo *t* e criptograma *c*, P(*t*|*c*)=P(*t*).
- Ou seja, numa cifra perfeita, o conhecimento do criptograma não nos facilita em nada a tarefa de descobrir o texto limpo associado – a incerteza relativa à mensagem mantém-se.

# Cifra perfeita (cont.)

- Exercício: Demonstre que a cifra por deslocamento de um carácter com uma chave equiprovável é uma cifra perfeita.

**Observação**: Note que na definição de cifra perfeita se está implicitamente a assumir que uma chave só é utilizada para cifrar uma única vez. Se a chave for repetidamente utilizada (cifra de César) a cifra passa a ser facilmente atacável – voltaremos a este assunto adiante...

# Entropia

- Grandeza que pretende capturar o "conteúdo informativo" de uma variável aleatória e é expresso em *bit*.
- Pode ser entendida como o *número médio de bits requeridos para representar a informação da variável na codificação mais eficiente*.
- E.g.
  - *moeda ao ar*: H($X$)=1
  - $p$(X=a)=0.5; $p$(X=b)=0.25; $p$(X=c)=0.25

  (representação óptima: a-»0; b-»10; c-»11)

  H($X$)=1*0.5+2*0.25+2*0.25=1.5

# Entropia (cont.)

- Exemplo sugere que um evento com probabilidade $2^{-n}$ deve ser representado por uma sequência de $n$ bits.
- ...generalizando, um evento com probabilidade $p_i$ pode ser representado por uma sequência com comprimento $-\log_2(p_i)$
- Seja $X$ uma variável aleatória com $|X|=n$ e $p_i$ a probabilidade de $X=x_i$.

$$\mathrm{H}(X) = -\sum_{i=1}^{n} p_i * \log_2(p_i)$$

Obs: Quando $p_i=0$, $\log2(p_i)$ não está definido. Para o cálculo da entropia admitimos que 0*log2(0) é 0 (que, de resto, o é no limite).

# Entropia (cont.)

- Entropia mútua

$$\mathrm{H}(XY) = -\sum_{y}\sum_{x} P_{XY}(x,y) * \log(P_{XY}(x,y))$$

- Entropia condicional

$$\mathrm{H}(X \mid Y) = -\sum_{y}\sum_{x} P(y) * P(x \mid y) * \log(x \mid y)$$

# Propriedades da Entropia

Exercício: Quando é que as igualdades limite das seguintes inequações se verificam?

$$0 \leq H(X) \leq \log |L_X|$$

$$H(XY) \leq H(X) + H(Y)$$

Exercício: Demonstre as seguintes (des)igualdades.

$$H(XY) = H(Y) + H(X|Y)$$

$$H(X|Y) \leq H(X)$$

# Informação Mútua

- Captura o decremento da entropia de uma variável face ao conhecimento de uma outra.

$$I(X;Y) \leftrightarrow H(X) - H(X|Y)$$

- Analogamente, a **Informação Mútua Condicional**

$$I(X;Y|Z) \leftrightarrow H(X|Z) - H(X|YZ)$$

IExercício: Demonstre a seguinte igualdade.

$$I(X;Y) = I(Y;X)$$

# Algumas leis

- *H(XY)*; *H(X|Y)* e *I(X;Y)* são conceitos mutuamente dependentes. Todos podem ser expressos à custa de um único.
  - *H(XY) = H(X) + H(X|Y)*
  - *I(X;Y) = H(X) – H(X|Y)*
  - *H(X|Y) = H(XY) – H(X)*
  - *I(X;Y) = H(X) + H(Y) – H(XY)*
  - *H(XY) = H(X) + H(Y) – I(X;Y)*
  - *H(X|Y) = H(X) – I(X;Y)*

# Cifra Perfeita (novamente)

- Uma forma abstracta de exprimirmos uma cifra perfeita consiste em considerarmos as igualdades:
  - H(T|CK) = 0; i.e. T é conhecido se dispusermos do conhecimento de C=*c*, K=*k* – basta calcular $t=d_k(c)$
  - I(T ; C) = 0; i.e. o conhecimento de C não nos permite baixar a incerteza relativa a T.
- Mas assim rapidamente deduzir

H(T) = H(T|C) P H(TK|C) = H(K|C)+H(T|KC) = H(K|C) P H(K)

ou seja, a entropia da chave deve ser tão grande quanto a da mensagem a transmitir.

# Entropia de uma Linguagem

- Captura a entropia de um carácter de uma linguagem

$$H_L = \lim_{n \to \infty} \frac{H(P_L^n)}{n}$$

- **Redundância** – captura o "excesso de caracteres" de uma linguagem

$$R_L = 1 - \frac{H_L}{\log_2 |L|}$$

Valores típicos:

$1.0 \leq H_L \leq 1.5$

$R_L \approx .75$

# Unicity Distance

- Definido como o tamanho médio do criptograma a partir do qual existe uma única chave que o produz.

$$U \approx \frac{H(K)}{R_L * \log_2 |L|}$$

**Nota**: Ver derivação deste valor em [*Stinson*, pag.59-63].

**Observação**: Como se atribui ao intruso um poder computacional ilimitado, a *unicity distance* dá-nos uma medida da segurança do sistema critográfico. A partir do limite dado por esta grandeza, a chave é completamente caracterizada pelo criptograma pelo que podemos admitir que fica do conhecimento do intruso.

# Unicity Distance (cont.)

- Alguns valores aproximados...
  - Cifra de César: 1.33
  - Cifra por substituição mono-alfabética: 25
  - Cifra de Vigenère (chave de comp. *n*): n*1.33
  - Transposição por linhas (blocos comp. *n*): $n*\log_2(n/e)/3.5$
    - n=3 0.128
    - n=5 1.32
    - n=7 2.87

obs.: utiliza-se a aproximação $n! \approx \sqrt{2 \cdot \pi \cdot n} \cdot \left(\frac{n}{e}\right)^n$

# Códigos de Autenticação

- Como avaliar a confiança que a mensagem recebida
  - foi enviada pelo emissor pressuposto – **Autenticidade**
  - não foi manipulada por outras entidades –**Integridade**
- Às duas garantias podemos associar noções distintas de ataques:
  - Ataque por Personificação – o intruso, após escutar a transmissão das mensagens $Y_1..Y_n$, gera uma mensagem Z que tenta fazer passar por $Y_{n+1}$
  - Ataque por Substituição – o intruso, após escutar a transmissão de $Y_1..Y_{n-1}$, substitui a mensagem $Y_n$ por uma por si escolhida.

# Códigos de Autenticação (cont.)

- ...mais uma vez encontramos ferramentas na teoria das probabilidades que nos permitem atacar o problema: *o teste de hipóteses* (teoria da decisão)
- Para cada mensagem recebida, o receptor deve decidir:
  - Aceita a mensagem como legítima;
  - Rejeita a mensagem.
- Decisões a que podemos associar dois tipos de erros
  - Rejeitar mensagens legítimas;
  - Aceitar mensagens fraudulentas.

# Códigos de Autenticação (cont.)

- Se excluirmos a possibilidade de rejeitar mensagens legítimas...
- ...podemos derivar:

$$P_P \geq \Sigma 2^{-I(Yn;K|Y1..Yn-1)}$$

$$P_S \geq \Sigma 2^{-H(K|\ Y1..Yn)}$$

- Valores óptimos para n=1...

$P_P = \Sigma 2^{-H(Y)}$ quando $H(Y|K)=0$

$P_S = \Sigma 2^{-H(K)}$ quando $I(K;Y)=0$

# Tópicos de Teoria da Complexidade

- Problemas:
  - **Tratáveis** - existe um algoritmo eficiente para a resolução do problema
  - **Intratáveis** - o melhor algoritmo para resolver o problema requer recursos computacionais inviáveis
  - **Insolúveis** - não é possível estabelecer um algoritmo para a resolução do problema

# Computabilidade

- Início do século XX...
- Formaliza a noção de algoritmo.
- Diferentes modelos computacionais determinam a mesma noção de problema solúvel (e.g. λ-calculus; máquinas de Turing; máquinas de registos; linguagem *While*.
- Exemplos de problema indecidíveis: halting problem; reconhecimento de linguagens tipo 0; problema de correspondência de Post.

# Complexidade

- Procura avaliar a eficiência (temporal e/ou espacial) dos algoritmos em função do "tamanho" dos dados de entrada.
- Modelo computacional adoptado é normalmente a máquina de Turing (mas...)
- É normal conduzirmos a análise para o *pior caso*, mas também é possível considerar o *caso médio* (análise estatística).
- Análises assimptótica (e.g. $O(2^n)$; $O(n^5)$)

# Classes de Complexidade

- **P** – *Polinomial* – tempo de execução é limitado por um polinómio (sobre o tamanho dos dados de entrada).
- **NP** – *Não determinístico Polinomial* – tempo de execução é limitado por um polinómio numa máquina de Turing não determinística (modelo computacional pode criar execuções concorrentes para prosseguir diferentes alternativas...).
- **EXP** – *Exponencial* – tempo de execução é exponencial...

# P versus NP

- É normal identificar-se a classe **P** com a classe dos problemas *tratáveis*.
- Um problema diz-se P(NP)-completo quando qualquer problema em P(NP) dispõe de uma redução polinomial nele.
- Exemplo de problema NP-Completo: validade da lógica proposicional
- Conjectura **P≠NP** ...

# Funções de sentido único

- Em criptografia gostaríamos de dispor de funções que:
  - disponham de um algoritmo eficiente para o seu cálculo
  - não disponham de um algoritmo eficiente que calcule uma sua inversa

essas funções são designadas por

**Funções de Sentido Único**

# FSU (cont.)

- É óbvio que funções de sentido único com domínio finito pertencem à classe NP.
- Mas:
  - **P≠NP** não implica a existência de funções de sentido único.
  - Complexidade do "pior caso" não é adequada para garantir segurança.

# FSU com segredo (cont.)

- Por vezes necessitamos de uma especificação mais apertada (na criptografia de chave pública): Funções de Sentido Único Com Segredo
  - Função injectiva que dispõe de um algoritmo eficiente
  - Cujo cálculo da inversa seja um problema intratável
  - Mas, para quem dispuser de informação adicional (o segredo) pode calcular essa inversa

# Alguns exemplos...

- A teoria de números tem-se revelado uma boa fonte de funções intratáveis e de sentido único...
  - Funções (tidas por) intratáveis:
    - factorização de um inteiro
    - logaritmo discreto
  - Funções de sentido único
    - x*y com x<y e ambos primos
  - Funções de sentido único com segredo
    - RSA (estudada adiante)

# Funções de Hash criptográficas

- ...outro exemplo de funções onde se baseia a segurança em argumentos de natureza de *complexidade computacional*.
- Mapeam mensagens (de comprimento arbitrário) num contra-domínio de tamanho fixo.
- ... mas devem ser "de sentido único" no sentido em que não deve ser possível inverter essa função.

# Propriedades requeridas pelas Funções de Hash Criptoráficas

...um requisito mais forte pretendido nas funções de Hash Criptográficas

- **Fortemente livre de colisões**:

  *Uma função* h *diz-se fortemente livre de colisões quando não é computacionalmente viável determinar mensagens* $m_1$ *e* $m_2$ *tal que:*

$$(m_1 \neq m_2) \wedge (\mathrm{h}(m_1) = \mathrm{h}(m_2))$$

# Birthday attack

- Um resultado famoso da teoria das probabilidades indica-nos que necessitamos de um contra-domínio de "tamanho razoável"

  Quantas pessoas tenho (em média) que perguntar a idade numa festa de anos para encontrar duas com o mesmo dia de aniversário?

  - ...testando *sqrt(N)* valores aleatórios do domínio disponho de probabilidade superior a ½ de encontrar uma colisão!
- Valores típicos para contra-domínios de funções de Hash criptográficas: 128..512 bit

# MD5

- Proposto pela RSA Labs (Donald Rivest)
- Melhoramento da função MD4
- Tamanho do contra-domínio: 128 bit
- Opera por *rounds* (4)

# MD5 (cont.)

- A,B,C,D: 32 bit

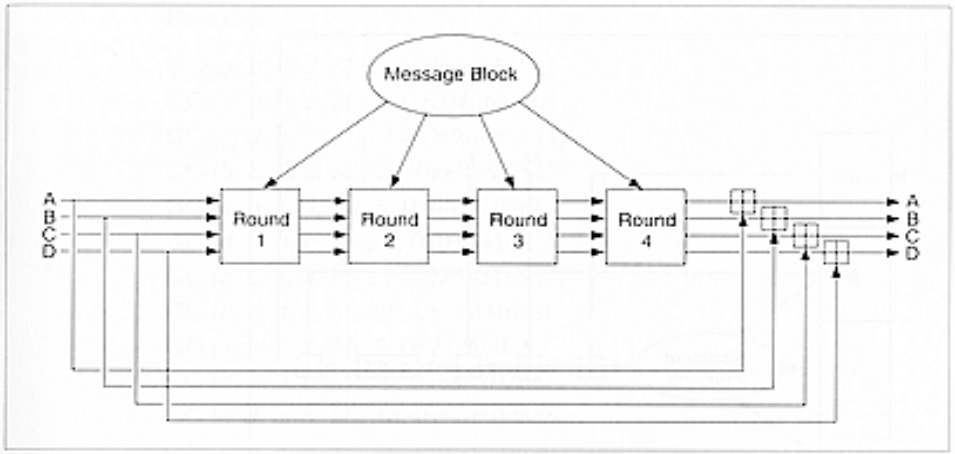


*Figure 18.5 MD5 main loop.*

# MD5 (cont.)

- Blocos 512 bit: ($M_0$-$M_{15}$ 32 bit sub-blocos)

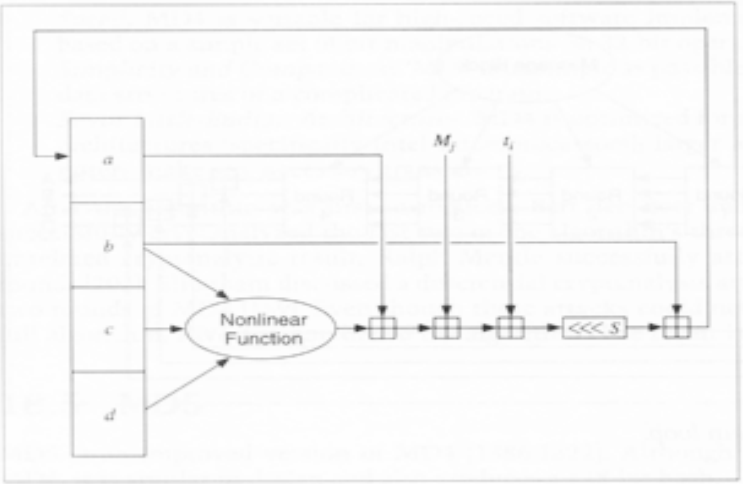


*Figure 18.6 One MD5 operation.*

# SHA-1

- Função de hash incluída no DSS (Digital Signature Standard)
- Desenvolvida pela NSA (para a NIST) – 1993/1995
- Tamanho do contra-domínio: 160 bit
- Optimizada para arquitecturas *big-endian*

# SHA-1 (cont.)

- A,B,C,D,E: 32 bit
- 4 rounds
- 20 operações/round

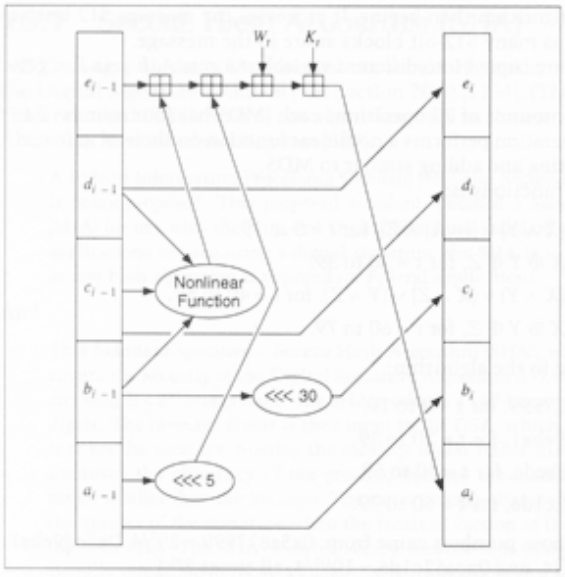


*Figure 18.7 One SHA operation.*

# Message Authentication Codes (MAC)

- As funções de hash, por si só, não garantem nem a integridade nem autenticidade!
- ... mas quando utilizadas com uma cifra já permitem estabelecer essas propriedades...
- Quando o único objectivo é produzir um código de autenticação, podemos utilizar:

HMAC: h(K,h(K,M))

# Referências

- Sobre Teoria da Informação...
  - Stinson, cap. 2
- Sobre Authentication Codes...
  - Stinson, cap. 10
  - G. J. Simmons, A survey of information authentication, 1992.
- Sobre Teoria da complexidade...
  - Computational Complexity, C. H. Papadimitriou – 1994, Addison-Wesley.
- Sobre funções de Hash...
  - Schneier, cap. 17
  - Stinson, cap. 7