Apêndices

Apêndice A

Índice alfabético de tópicos

Índice alfabético de protocolos

Índice alfabético de algoritmos

Tradução de termos técnicos

alçapão .....trapdoor
arbitragem.....arbitration
ataque de espelhamento.....man-in-the-middle (interception) atack
ataque de número sequencial.....number sequence attack
bloqueio ou sobrecarga intencional….denial of service, resource exaustion
chave .....key
cifra .....cipher
desvio de controle.....control bypassing, hacking
e-mail em massa e não solicitado.....spam
embuste ou trapaça.....scam
embusteiro.....rougue
escuta ativa.....interception
escuta passiva.....scan, snif
falha .....breach
forja de identificação.....spoof
fraude .....fraud, integrity violation
gancho .....backdoor
grampo .....eavesdrop, wiretap
invasão ou violação de autorização.....invasion, authorization violation
mediação .....adjudication
objeto ou rótulo identificador.....token
penetração.....intrusion
personificação.....masquerade, impersonation;
reprise .....replay
repudiação.....repudiation
Terceiro Confiável.....TTP- Trusted Third Party
troiano .....trojan horse
varredura .....media scavenging
vazamento.....leakage, disclosure

Tabela comparativa de probabilidades e grandes números

Segurança computacional é um jogo de probabilidades. Para se ter uma noção comparativa acerca dos graus de possibilidade associados às faixas de probabilidade com que lida a criptografia, esta tabela de estimativas de ocorrências de eventos do mundo físico em que vivemos pode ser útil.

Fenômeno Físico	**Número**
Probabilidade de ser morto por um raio, em um dia:	1 em 9 bilhões ($\sim 2^{33}$)
Probabilidade de ganhar em uma loteria estadual americana:	1 em 4 milhões ($\sim 2^{22}$)
Prob. de, num dia, ganhar na loteria estadual e ser morto por um raio:	1 em 2^{55}
Prob. de se morrer afogado, em um ano:	1 em 59 mil ($\sim 2^{16}$)
Prob. de se morrer em acidente de transito nos EUA em 1993:	1 em 6100 ($\sim 2^{13}$)
Prob. de se morrer em acidente de transito nos EUA durante uma vida:	1 em 88 ($\sim 2^{7}$)
Tempo até a próxima era glacial:	14000 ($\sim 2^{14}$) anos
Tempo até que o sol se transforme em uma supernova:	10^{9} ($\sim 2^{30}$) anos
Idade estimada do planeta terra:	10^{9} ($\sim 2^{30}$) anos
Idade estimada do universo:	10^{10} ($\sim 2^{34}$) anos
Número de átomos no planeta terra:	10^{51} ($\sim 2^{170}$)
Número de átomos no sol:	10^{57} ($\sim 2^{190}$)
Número de átomos na galáxia:	10^{67} ($\sim 2^{223}$)
Número de átomos no universo (sem a matéria escura):	10^{77} ($\sim 2^{265}$)
Volume do universo:	10^{84} ($\sim 2^{280}$) cm^3

Se o Universo for fechado	10^{11} ($\sim 2^{37}$) anos
Tempo estimado de vida do universo:	10^{18} ($\sim 2^{61}$) segundos

Se o Universo for aberto	
Tempo estimado até que estrelas de pouca massa esfriem	10^{14} ($\sim 2^{47}$) anos
Tempo estimado até que planetas se destaquem das estrelas	10^{15} ($\sim 2^{50}$) anos
Tempo estimado até que estrelas se destaquem das galáxias	10^{19} ($\sim 2^{64}$) anos
Tempo de decaimento de orbitas por irradiação gravitacional	10^{20} ($\sim 2^{67}$) anos
Tempo de decaimento de buracos negros pelo processo Hawking	10^{64} ($\sim 2^{213}$) anos
Tempo até que toda a matéria se torne líquida à temperatura zero	10^{64} ($\sim 2^{213}$) anos
Tempo até que toda a matéria decaia em ferro	10^{1026} anos
Tempo até que toda a matéria se colapse em buracos negros	10^{1076} anos

Bruce Schneier, 1996

Cálculo modular na Criptografia Assimétrica com exemplos

• Aritmética modular

a mod b = resto da divisão inteira de a por b (Ex.: 33 mod 7 = 5)

O resto da divisão inteira por b é chamado **resíduo**. Resíduos mod b formam um sistema aritmético semelhante ao dos números inteiros (próxima página), mas finito, o **anel de resídos** Z_b

Escolha de primos

Lista dos números primos no intervalo entre 2000000000 e 2000001000:
2000000011, 2000000033, 2000000063, 2000000087, 2000000089, 2000000099, 2000000137, 2000000141, 2000000143, 2000000153, 2000000203, 2000000227, 2000000239, 2000000243, 2000000269, 2000000273, 2000000279, 2000000293, 2000000323, 2000000333, 2000000357, 2000000381, 2000000393, 2000000407, 2000000413, 2000000441, 2000000503, 2000000507, 2000000531, 2000000533, 2000000579, 2000000603, 2000000609, 2000000621, 2000000641, 2000000659, 2000000671, 2000000693, 2000000707, 2000000731, 2000000741, 2000000767, 2000000771, 2000000773, 2000000789, 2000000797, 2000000809, 2000000833, 2000000837, 2000000843, 2000000957, 2000000983.
Tamanho do intervalo: 1001 Primos encontrados no intervalo: 53

Estimativa de quantos primos deve haver no intervalo, pelo teorema dos números primos:

$2*10^9 / \ln(2*10^9) - 2.000001*10^9 / \ln(2.000001*10^9) = 44.56$

Caso seja escolhido o primo **q** = 2000000983 (representação decimal), sua representação interna binária em 4 bytes (32 bits) será: 01110111 00110101 10010111 11010111

• Execução do protocolo de Diffie & Hellman: Exemplo

1- Escolha da aritmética modular

número *primo* para o **módulo**: **q** = 32693: (em 16 bits: 0111111 110110101)
número menor que o módulo para **base**: **a** = 27911 (em 16 bits: 0110110 100000111)

2- Geração de sementes e transmissão de criptogramas

nº randômico **x** gerado por **A**: 20589
nº randômico **y** gerado por **B**: 17391

Criptograma $\mathbf{c_A} = \mathbf{a}^{\mathbf{x}} \bmod \mathbf{q} = 27911^{20589} \bmod 32693 = 26097$ (bits: 01100101 11110001)

Criptograma $\mathbf{c_B} = \mathbf{a}^{\mathbf{y}} \bmod \mathbf{q} = 27911^{17391} \bmod 32693 = 19370$ (bits: 01001011 10101010)

3- Cálculo da chave de sessão

Chave $\mathbf{k} = \mathbf{c_A}^{\mathbf{y}} \bmod \mathbf{q} = 26097^{17391} \bmod 32693 = 18574$ (bits 01001000 10001110)

Chave $\mathbf{k} = \mathbf{c_B}^{\mathbf{x}} \bmod \mathbf{q} = 19370^{20589} \bmod 32693 = 18574$ (bits 01001000 10001110)

Exponenciação modular rápida com exemplo

1- O expoente é decomposto de acordo com sua representação binária

Exemplo: cálculo de $4^{37} \bmod 7 =$

$$37 = 2^5 + 2^2 + 2^0 = 100101_{(2)}$$

2- A exponenciação é fatorada conforme a decomposição do expoente:

Para cada posição binária do expoente, calcula-se o quadrado do resultado anterior pelo produto por 1 se o bit do expoente é 0, ou pela base se o bit é 1

$$6^{37} \bmod 7 = 6^{(2^5 + 2^2 + 2^0)} \bmod 7$$

$$6^{37} \bmod 7 = ((((((1*6)^2*1)^2*1)^2*6)^2*1)^2*6) \bmod 7$$

posição do bit do expoente → **5** 4 3 **2** 1 **0**

3- Como mod é homomorpismo, os resídos podem são calculados após cada operação:

$$(1*6)^2*1)^2*1)^2*6)^2*1)^2*6) \bmod 7 =$$

$$((...(1*6)^2*1 \bmod 7)^2*1 \bmod 7)^2*6 \bmod 7)^2*1 \bmod 7)^2*6 \bmod 7 = 6$$

- ## Complexidade do algoritmo de exponenciação rápida

O número de operações de multiplicação efetuadas durante a exponenciação modular é proporcional ao número de bits do expoente.

O número de operações de multiplicação e divisão efetuadas durante a exponenciação modular, é proporcional ao logaritmo do expoente: no máximo o dobro do número de bits do expoente.

Aritmética finita de Z_p com exemplos

1- Quando p é composto

- Nem todo resíduo ≠ 0 possui inverso multiplicativo
- Menos da metade dos resíduos possuem raizes quadradas (diagonal abaixo)

Exemplo: multiplicação em $\mathbf{Z_8}$. Resíduos = {0, 1, 2, 3, 4, 5, 6, 7}

*	1	2	3	4	5	6	7
1	**1**	2	3	4	5	6	7
2	2	**4**	6	0	2	4	6
3	3	6	**1**	4	7	2	5
4	4	0	4	0	4	0	4
5	5	2	7	4	**1**	6	3
6	6	4	2	0	6	**4**	2
7	7	6	5	4	3	2	**1**

$\sqrt{1} \bmod 8 = 1, 3, 5$ ou 7;

$\sqrt{4} \bmod 8 = 2$ ou 6;

$5^{-1} \bmod 8 = 5$

$7^{-1} \bmod 8 = 7$

1- Quando p é primo

- Todos os resíduos ≠ 0 possuem inverso multiplicativo (pela definição de mdc)
- Metade dos resíduos não nulos possuem raizes quadradas

Exemplo: multiplicação em $\mathbf{Z_7}$ = {0, 1, 2, 3, 4, 5, 6}

*	1	2	3	4	5	6
1	**1**	2	3	4	5	6
2	2	**4**	6	1	3	5
3	3	6	**2**	5	1	4
4	4	1	5	**2**	6	3
5	5	3	1	6	**4**	2
6	6	5	4	3	2	**1**

$\sqrt{1} \bmod 7 =$ 1 ou 6;

$\sqrt{2} \bmod 7 =$ 3 ou 4;

$\sqrt{4} \bmod 7 =$ 2 ou 5;

$3^{-1} \bmod 7 = 5$

Exemplo de geração de chaves para o RSA

1- Escolhem-se dois números primos: por exemplo `p = 3 ; q = 11`

- Calcula-se `n = 3*11 = 33 ;` ϕ`(n) = (3-1)*(11-1) = 20`

2- Escolhe-se um inteiro primo com ϕ`(n)` **e calcula-se seu inverso em** Z_{ϕ}:

- **e** `= 17;` Euclides extendido resolve `17*`**d** `+ 20*`**y** `=` $\pm$`1:` **d**`=13,` **y**`=-11`

Tabela multiplicativa de Z_{20}

*	2	**3**	4	5	6	**7**	8	**9**	10	**11**	12	**13**	14	15	16	**17**	18	**19**
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
2	4	6	8	10	12	14	16	18	0	2	4	6	8	10	12	14	16	18
3	6	9	12	15	18	**1**	4	7	10	13	16	19	2	5	8	11	14	17
4	8	12	16	0	4	8	12	16	0	4	8	12	16	0	4	8	12	16
5	10	15	0	5	10	15	0	5	10	15	0	5	10	15	0	5	10	15
6	12	18	4	10	16	2	8	14	0	6	12	18	4	10	16	2	8	14
7	14	**1**	8	15	2	9	16	3	10	17	4	11	18	5	12	19	6	13
8	16	4	12	0	8	16	4	12	0	8	16	4	12	0	8	16	4	12
9	18	7	16	5	14	3	12	**1**	10	19	8	17	6	15	4	13	2	11
10	0	10	0	10	0	10	0	10	0	10	0	10	0	10	0	10	0	10
11	2	13	4	15	6	17	8	19	10	**1**	12	3	14	5	16	7	18	9
12	4	16	8	0	12	4	16	8	0	12	4	16	8	0	12	4	16	8
13	6	19	12	5	18	11	4	17	10	3	16	9	2	15	8	**1**	14	7
14	8	2	16	10	4	18	12	6	0	14	8	2	16	10	4	18	12	6
15	10	5	0	15	10	5	0	15	10	5	0	15	10	5	0	15	10	5
16	12	8	4	0	16	12	8	4	0	16	12	8	4	0	16	12	8	4
17	14	11	8	5	2	19	16	13	10	7	4	**1**	18	15	12	9	6	3
18	16	14	12	10	8	6	4	2	0	18	16	14	12	10	8	6	4	2
19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	**1**

3- Teste do par de chaves RSA gerada para o titular A para cifrar blocos de 5 bits: Toma-se, por exemplo, m = $(10100)_2$ `= 20`

$$\mathbf{E_A(m) = m^e} \bmod \mathbf{n} = 20^{17} \bmod 33 = 26 = (11010)_2 = \mathbf{c;}$$

$$\mathbf{D_A(c) = c^d} \bmod \mathbf{n} = 26^{13} \bmod 33 = 20 = (10100)_2 = \mathbf{m.}$$

4- Armazena-se $\mathbf{D_A = (d,n) =}$ `(13,33);` **Publica-se** $\mathbf{E_A = (e,n) =}$ `(17,33)`

Outras possíveis escolhas de pares de chaves para o módulo `n = 33:` `{(7,33),(3,33)};` etc.

• Aritmética das curvas elípticas sobre corpos finitos

Na geometria analítica, o conjunto de pontos de um espaço vetorial com coordenadas (x,y) que satisfazem uma dada equação da forma.

$$y^2 = x^3+ax+b$$

é chamado de *curva elíptica,* caso os coeficientes satisfaçam $4a^3+27b^2 \neq 0$.

Em 1985 *N. Koblitz* e *V. Miller* descobriram que, se aplicada a um espaço onde as coordenadas são elementos de um corpo finito (ex: Z_p), a definição de curva elíptica seleciona pontos discretos que, incluindo-se a eles um "ponto no infinito" (O), formam um grupo algébrico sob a operação de composição inspirada na geometria das secantes dos espaços métricos. Esta operação substitui a operação de exponenciação em algoritmos assimétricos.

Grupos de curvas elípticas $E(Z_p)$ -

$$E(Z_p) = \{ P=(x,y) \in \mathbf{Z_p} \times \mathbf{Z_p} \mid y^2 = x^3+ax+b \} \cup \{O\}$$

Com a operação algébrica do grupo, denotada por "+", definida pelas regras:

1. $P+O = O+P = P$
2. Dado $P=(x,y)$, denotando $-P=(x,-y)$, $P+(-P) = O$
3. Dados $P=(x_1,y_1)$, $Q=(x_2,y_2)$, $P+Q=(x_3,y_3)$ é dado por

   $x_3 = \lambda^2-x_1-x_2$;

   $y_3 = \lambda(x_1-x_3)-y_1$ onde

   $\lambda = (y_2-y_1)/(x_2-x_1)$ se $P \neq Q$, ou

   $\lambda = (3x_1^2+a)/(y_1+y_1)$ se $P=Q$.

4. $nP = P+P+...+P$ (n vezes) é o produto escalar (sobre $Z \times E(Z_p)$)

Exemplo de um grupo de curva elíptica $E(Z_p)$

1- Escolhem-se um primo e uma equação elíptica: $p = 23$; $y^2 = x^3+x+1$

- A equação $y^2 = x^3+ax+b$ escolhida ($a=1; b=1$) satisfaz $4a^3+27b^2 \neq 0$

2- Aos pontos $(x,y) \in \mathbf{Z_p} \times \mathbf{Z_p}$ satisfazendo a equação, junta-se $\mathcal{O}$ para obter $E(Z_{23})$

- O ponto $(5,4)$ por exemplo, satisfaz $4^2 \bmod 23 = (5^3+5+1) \bmod 23$

Pontos que formam o grupo $\mathbf{E(Z_{23})}$ da curva $y^2 = x^3+x+1$

(0,1)	(3,13)	(6,19)	(11,3)	(13,7)	(18,3)
(0,22)	(4,0)	(7,11)	(11,20)	(13,16)	(18,20)
(1,7)	(5,4)	(7,12)	(12,4)	(17,3)	(19,5)
(1,16)	(5,19)	(9,7)	(12,19)	(17,20)	(19,18)
(3,10)	(6,4)	(9,16)	$\mathcal{O}$	"ponto no infinito"	

- A "soma" dos pontos $(3,10) + (9,7) = (y_3, x_3)$ por exemplo, é dada por:

$(P \neq Q)$ $\lambda = (y_2-y_1)/(x_2-x_1) = (7-10)/(9-3) = -1*2^{-1} \bmod 23 = 11 \in \mathbf{Z_{23}}$

$x_3 = \lambda^2-x_1-x_2 = 121-3-9 = 109 \bmod 23 = 17 \in \mathbf{Z_{23}}$

$y_3 = \lambda*(x_1-x_3)-y_1 = 11*(3-17)-10 = 141 \bmod 23 = 20 \in \mathbf{Z_{23}}$

- O "produto escalar" $2 \cdot (3,10) = (y_4, x_4)$ por soma iterada $(3,10)+(3,10)$:

$(P=Q)$ $\lambda = (3x_1^2+a)/(2y_1) = (27+1)/(20) = 5*20^{-1} \bmod 23 = 6 \in \mathbf{Z_{23}}$

$x_4 = \lambda^2-x_1-x_2 = 36-3-3 = 30 \bmod 23 = 7 \in \mathbf{Z_{23}}$

$y_4 = \lambda*(x_1-x_3)-y_1 = 6(3-7)-10 = -34 \bmod 23 = 12 \in \mathbf{Z_{23}}$

Nesta curva portanto, $(3,10)+(9,7) = (17,20)$; $2 \cdot (3,10) = (7,12)$

- **Comparações entre operações aritméticas em Z_p e $E(Z_p)$ -**

Operação	Z_p	$E(Z_p)$
"Produto"	$a*b \bmod p$	$P+Q$
"Exponenciação"	$a^n \bmod p$	nP
Logaritmo discreto	Encontrar n tal que $a^n \bmod p = b$	Encontrar n tal que $nP = Q$

- **Comparações por níveis de robustez equivalentes -**

  Nível medido pelo custo de ataque ao algoritmo de cifra para obter a chave privada da chave pública usando algoritmo mais eficiente conhecido para o problema matemático que modela o ataque:

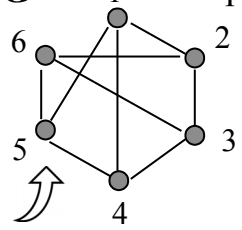
Esforço para obter chave privada mapeado para tamanho da chave	RSA Fatoração em Z_p: Number Field Sieve (NFS)	ECC genérico Logaritmo em $E(Z_p)$: Pollard - Rho
$3x10^8$ MIPS - ano	~960 bits	155 bits
$3x10^{18}$ MIPS - ano	~1820 bits	210 bits
$3x10^{28}$ MIPS - ano	~2500 bits	239 bits

  Devido a esse diferencial de eficiência no custo de operação das primitivas computacionais entre as famílias de algoritmos assimétricos clássicos (como o RSA, DSA ou ElGamal sobre $\mathbf{Z_p}$) e os baseados em aritmética de curvas elípiticas (ECC) é que as mais recentes inovações em PKIs – como as redes de criptomoedas, por exemplo – empregam algoritmos da segunda família.

Exemplo de protocolo 0-K com isomorfismo de grafos

Visão do Verificador

G

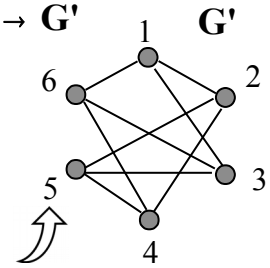


O provador diz conhecer o isomorfismo **H : G → G'**

H	1	2	3	4	5	6
↓	?	?	?	?	?	?

H

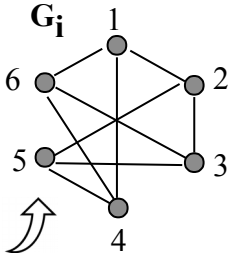
G'



G	1	2	3	4	5	6
1	0	1	0	1	1	0
2	1	0	1	0	0	1
3	0	1	0	1	0	1
4	1	0	1	0	1	0
5	1	0	0	1	0	1
6	0	1	1	0	1	0

G'	1	2	3	4	5	6
1	0	1	1	0	0	1
2	1	0	0	1	1	0
3	1	0	0	0	1	1
4	0	1	0	0	1	1
5	0	1	1	1	0	0
6	1	0	1	1	0	0

G_i



h_i

s_i

No **i**-ésimo desafio, alegando ter construído **G_i** isomorfo a **G** e a **G'**

o provador apresenta **G_i** e o verificador escolhe qual isomorfismo quer verificar**:** **h_i: G → G_i** ou **s_i: G' → G_i**

G_i	1	2	3	4	5	6
1	0	1	0	1	0	1
2	1	0	1	0	1	0
3	0	1	0	0	1	1
4	1	0	0	0	1	1
5	0	1	1	1	0	0
6	1	0	1	1	0	0

Supondo esolhido h_i :

h_i	1	2	3	4	5	6
↓	5	4	6	3	2	1

s_i	1	2	3	4	5	6
↓	?	?	?	?	?	?

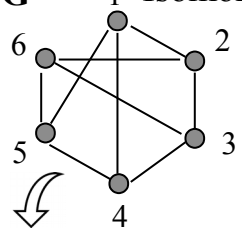
O provador revela a permutação escolhida (supondo aqui **h_i**) e o verificador testa se é isomorfismo: $h_i^{-1} \circ G \circ h_i =? G_i$ (alternativamente, se $s_i^{-1} \circ G' \circ s_i =? G_i$) Em representação matricial: $h_i^T * G * h_i =? G_i$ (**h** matriz permutação => h^{-1} = transposta de **h**)

$$\begin{bmatrix} 0&0&0&0&0&1 \\ 0&0&0&0&1&0 \\ 0&0&0&1&0&0 \\ 0&1&0&0&0&0 \\ 1&0&0&0&0&0 \\ 0&0&1&0&0&0 \end{bmatrix} * \begin{bmatrix} 0&1&0&1&1&0 \\ 1&0&1&0&0&1 \\ 0&1&0&1&0&1 \\ 1&0&1&0&1&0 \\ 1&0&0&1&0&1 \\ 0&1&1&0&1&0 \end{bmatrix} * \begin{bmatrix} 0&0&0&0&1&0 \\ 0&0&0&1&0&0 \\ 0&0&0&0&0&1 \\ 0&0&1&0&0&0 \\ 0&1&0&0&0&0 \\ 1&0&0&0&0&0 \end{bmatrix} = ? \begin{bmatrix} 0&1&0&1&0&1 \\ 1&0&1&0&1&0 \\ 0&1&0&0&1&1 \\ 1&0&0&0&1&1 \\ 0&1&1&1&0&0 \\ 1&0&1&1&0&0 \end{bmatrix}$$

Algoritmo **Monte Carlo**: desafios acumulam evidência de que Provador conhece **H**

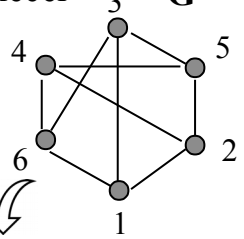
Exemplo de protocolo 0-K com isomorfismo de grafos

Visão do Provador

G 

Isomorfimo **H** que ele (provador) precisa/alega conhecer

H	**1**	**2**	**3**	**4**	**5**	**6**
↓	**3**	**5**	**2**	**1**	**6**	**4**

G' 

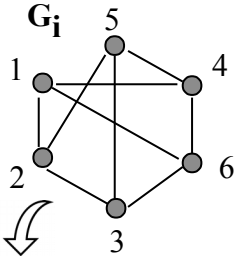
H

H^{-1}

G	**1**	**2**	**3**	**4**	**5**	**6**
1	0	1	0	1	1	0
2	1	0	1	0	0	1
3	0	1	0	1	0	1
4	1	0	1	0	1	0
5	1	0	0	1	0	1
6	0	1	0	1	0	1

G'	**1**	**2**	**3**	**4**	**5**	**6**
1	0	1	1	0	0	1
2	1	0	0	1	1	0
3	1	0	0	0	1	1
4	0	1	0	0	1	1
5	0	1	1	1	0	0
6	1	0	1	1	0	0

h_i

G_i 

s_i

Isomorfismos de rodada para sequencia de desafios

G_i é apresentado ao Verificador, que escolhe se quer conhecer **h_i** ou **s_i**

G_i	**1**	**2**	**3**	**4**	**5**	**6**
1	0	1	0	1	0	1
2	1	0	1	0	1	0
3	0	1	0	0	1	1
4	1	0	0	0	1	1
5	0	1	1	1	0	0
6	1	0	1	1	0	0

h_i	**1**	**2**	**3**	**4**	**5**	**6**
↓	**5**	**4**	**6**	**3**	**2**	**1**

Supondo esolhido **s_i** :

s_i	**1**	**2**	**3**	**4**	**5**	**6**
↓	**?**	**?**	**?**	**?**	**?**	**?**

Para revelar o isomorfismo escolhido (supondo, **s_i**), o Provador calcula a correspondente permutação de vértices a partir dos dois isomorfismos que conhece: o aleatório gerado para este desafio e o secreto: **$s_i = h_i \circ H^{-1}$** (ou, ao revés, **$h_i = s_i \circ H$**). Em representação matricial:

$h_i * H^T = s_i$

s_i	**1**	**2**	**3**	**4**	**5**	**6**
↓	**3**	**6**	**5**	**1**	**4**	**2**

$$\begin{bmatrix} 0&0&0&0&1&0 \\ 0&0&0&1&0&0 \\ 0&0&0&0&0&1 \\ 0&0&1&0&0&0 \\ 0&1&0&0&0&0 \\ 1&0&0&0&0&0 \end{bmatrix} * \begin{bmatrix} 0&0&0&1&0&0 \\ 0&0&1&0&0&0 \\ 1&0&0&0&0&0 \\ 0&0&0&0&0&1 \\ 0&1&0&0&0&0 \\ 0&0&0&0&1&0 \end{bmatrix} = \begin{bmatrix} 0&0&1&0&0&0 \\ 0&0&0&0&0&1 \\ 0&0&0&0&1&0 \\ 1&0&0&0&0&0 \\ 0&0&0&1&0&0 \\ 0&1&0&0&0&0 \end{bmatrix}$$

Algoritmo **Monte Carlo**: O Provador pode blefar o conhecimento de **H**, desde que o Verificador escolha conhecer o isomorfismo gerado para o desafio. **n** desafios independentes atenuam a probabilidade de seguidos blefes, indicando conhecimento de **H**

Método de Monte Carlo

Método para construir uma classe de algoritmos que calculam probabilidades baseadas em uma série de amostragens aleatórias, prevendo as seguintes etapas (ou características) para o algoritmo:

Dada uma instância de um problema a ser resolvido pelo método:

1. Definir um domínio de amostragem contendo dados referentes a esta instância;
2. Gerar uma série amostral, escolhendo aleatoriamente vários dados desse domínio, onde cada amostra servirá como entrada para uma rodada da série;
3. Em cada rodada da série, executar um cálculo determinístico com sua entrada aleatória;
4. Combinar os resultados obtidos nas rodadas da série em um resultado final para a instância do problema, conforme uma distribuição de probabilidades relacionada ao problema e ao domínio de amostragem.

Quando o problema é de decisão, ou seja, quando o cálculo determinístico em cada rodada deve produzir resultado binário, diz-se que o algoritmo é :

- "de monte-carlo **com viés falso**" se um resultado de rodada falso implica em resultado final falso para a instância do problema com probabilidade = 1.
- "de monte-carlo **com viés verdadeiro**" se um resultado de rodada verdadeiro implica em resultado final verdadeiro para a instância do problema com probabilidade = 1.

Exemplos:

Para exemplificar o método, o primeiro problema abordado em sala de aula foi:

"O professor está falando sério ao sortear SS?"

Para a instância da aula, o problema foi resolvido implementando-se um *protocolo* (algoritmo de execução alternada entre dois ou mais agentes) no qual:

1. O domínio de amostragem era binário: em uma de duas mãos fechadas e apresentadas para escolha (ao estilo par-ou-impar), supostamente haveria um giz, dando direito à menção SS)
2. Cada rodada consistia da apresentação das duas mãos fechadas, e a respectiva entrada consistia da escolha, por um aluno, da mão na qual ele adivinhava que o giz estaria.
3. O calculo determinístico de cada rodada consistia em o professor abrir a mão escolhida e o aluno verificar se na mão escolhida estava ou não o (suposto) giz.
4. Ambas mãos ocultadas pelo professor antes de iniciar cada escolha, alunos e amostras diferentes para cada rodada, presumiam a aleatoriedade exigida pela característica 2 acima (para monte carlo). Assim, a combinação dos resultados de n rodadas foi dada por

   Prob(sim)=1 se algum aluno acertasse;
   Prob(não)=$1/2^n$ se todos os alunos errassem.

Portanto, um protocolo de monte carlo **com viés positivo**

Outro exemplo

O problema abordado como "Exemplo de protocolo 0-k" no diagrama XXX da apostila (pg 55 do Cap. 4 e pgs A-16,17 deste Apêndice):

> "**O cliente que cadastrou o par de grafos (G,G') para com ele identificar-se junto ao servidor, conhece isomorfismo entre G e G'**?" (bijeção entre vértices preservando incidências, i.e., conectividade ou não por aresta)

Para uma dada instância de (G,G'), o problema é resolvido por um **Protocolo de autenticação 0-k, baseado em isomorfismo de grafos**, no qual:

1. O domínio é combinatório: Para uma série de grafos G_i haveria isomorfismos tanto entre G_i e G quanto entre G_i e G'

2. Cada rodada consiste da apresentação de um tal grafo G_i, e a respectiva entrada consiste da escolha, pelo servidor de autenticação, de qual dos dois isomorfismos (G_i <-> G ou G_i <-> G') o servidor deseja conhecer (equivalente, no problema anterior, à escolha de qual mão estaria o giz).

3. O calculo determinístico de cada rodada consiste em o cliente exibir a permutação de vértices que corresponde ao isomorfismo escolhido (equivalente a abrir a mão) e o servidor verificar se esta permutação corresponde exatamente ao isomorfismo escolhido (G_i <-> G ou G_i <-> G'), ou não.

4. Um novo G_i a cada rodada, e o servidor agindo independentemente para cada escolha, presumem a aleatoriedade exigida pela característica 2 acima (para monte carlo). Assim, a combinação dos resultados de n rodadas é dada por

   Prob(Não)=1 se algum isomorfismo escolhido não "batesse" com a permutação apresentada;

   Prob(Sim)=$1/2^n$ se todos os isomorfismos escolhidos "batessem".

Portanto, um protocolo de monte carlo **com viés negativo**

Exemplo de análise de freqüência com o Vigenère

Cifragem com o algortmo da pág 16 (Vigenère com monoalfabética XOR), chave de 7 bytes

Distribuição de ocorrencias dos 571 caracteres em texto e criptograma:

TEXTO		CRIPTOGRAMA	
num. ASCII – char –	Número de ocorrencias	num. ASCII –	Número de ocorrências
1°- chr(32) ' ':	88 (prob=0.1541)	chr(19)	22 (prob=0.0385)
2°- chr(97) 'a':	65 (prob=0.1138)	chr(114)	20 (prob=0.0350)
3°- chr(101) 'e':	53 (prob=0.0928)	chr(70)	18 (prob=0.0315)
4°- chr(111) 'o':	40 (prob=0.0701)	chr(86)	18 (prob=0.0315)
5°- chr(115) 's':	40 (prob=0.0701)	chr(9)	17 (prob=0.0298)
6°- chr(114) 'r':	39 (prob=0.0683)	chr(126)	17 (prob=0.0298)
7°- chr(100) 'd':	33 (prob=0.0578)	chr(63)	16 (prob=0.0280)
8°- chr(105) 'i':	31 (prob=0.0543)	chr(30)	15 (prob=0.0263)
9°- chr(109) 'm':	24 (prob=0.0420)	chr(51)	15 (prob=0.0263)
10°- chr(116) 't':	23 (prob=0.0403)	chr(93)	15 (prob=0.0263)
11°- chr(117) 'u':	20 (prob=0.0350)	chr(20)	14 (prob=0.0245)
12°- chr(110) 'n':	18 (prob=0.0315)	chr(25)	14 (prob=0.0245)
13°- chr(112) 'p':	17 (prob=0.0298)	chr(55)	14 (prob=0.0245)
14°- chr(99) 'c':	12 (prob=0.0210)	chr(92)	14 (prob=0.0245)
15°- chr(103) 'g':	12 (prob=0.0210)	chr(7)	13 (prob=0.0228)
16°- chr(67) 'C':	5 (prob=0.0088)	chr(8)	12 (prob=0.0210)
17°- chr(102) 'f':	5 (prob=0.0088)	chr(59)	12 (prob=0.0210)
18°- chr(44) ',':	4 (prob=0.0070)	chr(88)	11 (prob=0.0193)
19°- chr(108) 'l':	4 (prob=0.0070)	chr(26)	10 (prob=0.0175)
20°- chr(227) 'ã':	4 (prob=0.0070)	chr(64)	10 (prob=0.0175)
21°- chr(46) '.':	3 (prob=0.0053)	chr(84)	10 (prob=0.0175)
22°- chr(65) 'A':	3 (prob=0.0053)	chr(91)	10 (prob=0.0175)
23°- chr(98) 'b':	3 (prob=0.0053)	chr(3)	9 (prob=0.0158)
24°- chr(118) 'v':	3 (prob=0.0053)	chr(61)	9 (prob=0.0158)
25°- chr(122) 'z':	3 (prob=0.0053)	chr(33)	8 (prob=0.0140)
26°- chr(231) 'ç':	3 (prob=0.0053)	chr(80)	8 (prob=0.0140)
27°- chr(233) 'é':	3 (prob=0.0053)	chr(82)	8 (prob=0.0140)
28°- chr(113) 'q':	2 (prob=0.0035)	chr(15)	7 (prob=0.0123)
29°- chr(225) 'á':	2 (prob=0.0035)	chr(21)	7 (prob=0.0123)
30°- chr(9) tab:	1 (prob=0.0018)	chr(44)	7 (prob=0.0123)
31°- chr(45) '-':	1 (prob=0.0018)	chr(54)	7 (prob=0.0123)
32°- chr(83) 'S':	1 (prob=0.0018)	chr(65)	7 (prob=0.0123)
33°- chr(85) 'U':	1 (prob=0.0018)	chr(11)	6 (prob=0.0105)
34°- chr(104) 'h':	1 (prob=0.0018)	chr(14)	6 (prob=0.0105)
35°- chr(106) 'j':	1 (prob=0.0018)	chr(31)	6 (prob=0.0105)
36°- chr(224) 'à':	1 (prob=0.0018)	chr(45)	6 (prob=0.0105)
37°- chr(234) 'ê':	1 (prob=0.0018)	chr(58)	6 (prob=0.0105)
38°- chr(250) 'ú':	1 (prob=0.0018)	chr(74)	6 (prob=0.0105)
38°-39°		chr(77,94)	6 (prob=0.0105)
40°-48°	chr(2,18,32,34,42,49,71,73,90)		5 (prob=0.0088)
49°-53°	chr(5,23,39,48,75)		4 (prob=0.0070)
54°-61°	chr(29,38,43,46,53,57,67,87)		3 (prob=0.0053)
62°-67°	chr(13,22,52,56,208,218)		2 (prob=0.0035)
68°-95°	chr(1,4,10,16,24,28,37,40,50,66,72,76, 85,95,102,122,124,129,135,145,152, 156,177,179,183,185,217,222)		1 (prob=0.0018)

Tamanho médio de palavras do texto = 5.489; Total de bytes do criptograma: 571

Lista de exercícios

1 - Segundo a classificação de Warwick Ford para tipos de ataque a sistemas computacionais, o que caracteriza ataque subjacente, ataque primário ou ameaça básica a um sistema computacional?

2 - Em que consistem os ataques por *spoofing* de IP e de número seqüencial ao TCP?

3 - Porque o ataque de número seqüencial associado ao *spoofing* de IP habilita a transmissão de pacotes falsos apenas em uma direção do tráfego de sessão TCP?

4 - O que distingue modelos de controle de acesso discricionário de modelos de controle de acesso mandatório?

5 - Descreva o tipo de ataque que expõe usuários de serviços que usam controle de acesso remoto, mas sem se valer de um serviço de autenticação distribuído (exemplo de serviço: Telnet, ftp, rlogin)

6 - Das afirmações abaixo, diga quais são verdadeiras e quais são falsas:
Para ser segura uma cifra precisa ter um grande espaço de chaves
Uma cifra cujo espaço de chaves é grande, é uma cifra segura
Cifras cujo algoritmo criptográfico é ocultado são mais seguras
Algoritmos criptográficos assimétricos estão expostos a mais tipos de ataques que os simétricos
Algoritmos criptográficos assimétricos são menos seguros que os algoritmos simétricos

7 - Descreva o que é uma cifra monoalfabética, e porque é insegura. Descreva a diferença entre um algoritmo criptográfico simétrico e um assimétrico

8 - O que significam dedução local e quebra total de um algoritmo criptográfico?

9 - O que caracteriza um protocolo criptográfico ser arbitrado, ajuizado ou auto-verificável?

10- Diga quais dos propósitos abaixo, nenhum protocolo criptográfico até hoje concebido tem condições de almejar
Transformar sigilo em confiança na integridade de dados
Transferir sigilo entre dados
Criar confiança na integridade de dados entre pontos de transmissão
Criar sigilo para a transmissão de dados

11- Quais inconvenientes e limitações existem para se estabelecer canais individuais com privacidade (sigilosos) entre pares de usuários de uma rede, usando apenas algoritmos criptográficos simétricos, se compararmos ao uso de sistemas assimétricos?

12- Em que consiste um envelope digital?

13- Como um algoritmo criptográfico assimétrico é usado para estabelecer um sistema criptográfico de chaves públicas?

14- Como um sistema criptográfico de chaves públicas pode ser usado para implementar o conceito de assinatura digital? Em que consiste a assinatura digital de um dado?

15- O que é uma função de hash?

16- Em que consiste um MAC (*message autentication code*)?

17- Qual a diferença principal entre a garantia de integridade oferecida por MACs e a oferecida por assinatura digital?

18- Que tipo de ataque pode sofrer redes onde os agentes usam chaves públicas para estabelecer canais sigilosos entre si?

19- Em que consiste a certificação de chaves públicas?

20- Em que consiste o ataque por dicionário a um arquivo de senhas?

21- Como pode ser dificultado o ataque por dicionário a um arquivo de senhas?

22- Como pode ser evitado o ataque de personificação através do vazamento de senhas durante login remoto por escuta passiva no meio físico da rede?

23- Como a posse de uma chave privada pode ser verificada ao início de cada sessão, em uma rede que usa sistema de chaves públicas?

24- Qual o propósito do protocolo de Diffie & Hellman?

25- Qual o propósito da escrituração de chaves? (*key escrow*)

26- Porque os algoritmos assimétricos são computacionalmente viáveis, ao passo que a fatoração de inteiros e o cálculo do logaritmo discreto não o são?

27- Compare o uso da criptografia entre nós (*link to link*) e entre aplicativos (*end do end*): fale de uma vantagem e uma desvantagem de cada um desses tipos de implementação.

28- Porque uma chave assimétrica precisa ser maior que uma chave simétrica de mesma robustez?

29- Como o DES (*Data Encription Standard*) se tornou o primeiro padrão criptográfico de domínio público?

30- Em que se baseia a segurança do algoritmo RSA?

31- Responda, justificando, se é seguro ou não usar o mesmo módulo para vários pares de chaves no RSA.

32- Responda, justificando, se é seguro ou não encriptar e depois assinar uma mensagem com o RSA.

33- Porque o RSA se tornou um algoritmo importante para a segurança das instituições financeiras?

34- Quais tipos de ação uma regra de filtragem de pacotes especifica?

35- Qual a maior vulnerabilidade representada por um *firewall* que liga uma rede corporativa à internet?

36- O que distingue Filtro de Pacotes, *Screening Router*, e *Gateway* de aplicação?

37- Cite duas limitações na segurança proporcionada pelos *firewalls*

38- O que caracteriza uma máquina como um *Bastion Host*?

39- Qual função do sistema operacional unix precisa ser desabilitada numa máquina com múltiplas interfaces que hospede um *gateway* de aplicação?

40- Como este curso mudou sua idéia do que seja segurança computacional?