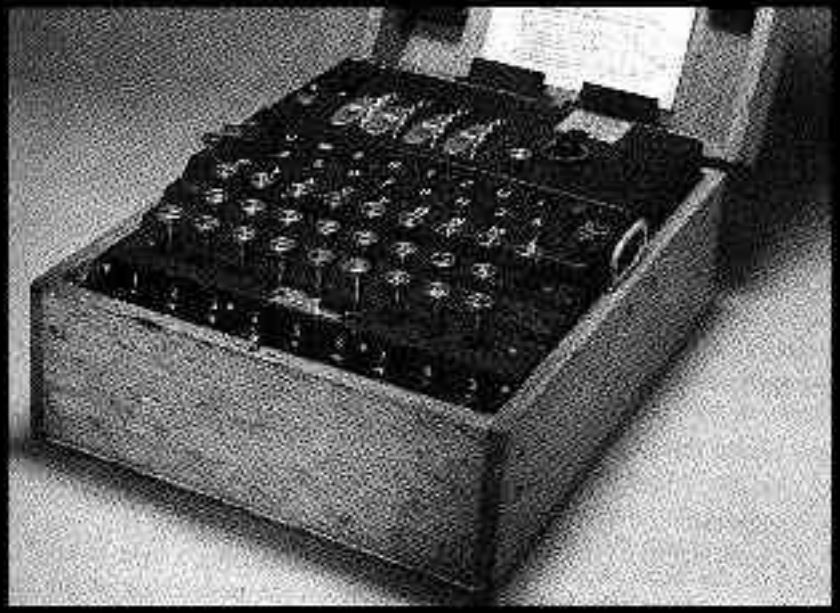
APRESENTA

http://nerv.5p.org.uk/

CRYPTO & CLUSTER

(PARTE 1)



Objetivo: desmistificar a criptoanálise, introduzir técnologias de clusters.

Tópicos

Steganografia x Criptografia

- Histórico de Ataques
  - Criptografia Clássica

  Dividindo as águas: Enigma
  - Criptografia Moderna
- Ambientes Distribuídos
  - Clusters OpenMosix
  - Clusters Beowulf

Steganografia != Criptografia

Diz a lenda que Julio Cezar escrevia mensagem na cabeça de soldados (que após o cabelos crescer) partiam para entregá-las aos seus generais...

Isso é **STEGANOGRAFIA**!

http://cdm.frontthescene.com.br/artigos/stego1.pdf

Criptografia Clássica

Desmistificando ataques
de criptoanálises

Preliminares de um ataque

Primeiras Perguntas...

1. Como identificar criptografia?
2. Como identificar algoritmos?

Preliminares de um ataque

Como identificar criptografia?

Bons algoritmos de criptografia produzem cyphertexts que não podem ser comprimidos!

1. Tentar fazer uma compressão.
2. Tentar fazer descompressão com algoritmos populares (zip, tar, rar...)

Preliminares de um ataque

Como identificar criptografia?

Os *Headers* dos arquivos ajudam na identificação de formato populares!

1. Catalogar *headers* conhecidos criptados com algoritmos populares e comparar este com arquivo alvo.

Listas de Formatos em http://www.wotsit.org/

Preliminares de um ataque

Como identificar?

Algumas das ferramentas
"deixam marcas"
nos arquivos encriptados...

Pois a segurança não é baseada na obscuridade.

```c
#include <stdio.h>

int main(int ac, char **av) {
    FILE *file;
    int i, keys[5];

    if (ac == 1) {
        fprintf(stdin, "Find out crypted files.\n");
        fprintf(stdin, "use: %s <file>\n", av[0]);
        exit(0);
    }

    if ((file = fopen(av[1], "r+b")) == NULL) {
        fprintf(stderr, "Unable to open %s\n", av[1]); exit(1);
    }

    for(i = 0; i <= 5; i++) {
        keys[i] = fgetc(file);
    }

    printf("Probably... ");

    if ((keys[0] == 140) && (keys[1] == 13) &&
          (keys[2] == 4) && (keys[3] == 3)) {
        printf("is GPGed with \"gpg -c\".\n");
    }
    else if ((keys[0] == 133) && (keys[1] == 2)   &&
                (keys[2] == 14) && (keys[3] == 3)) {
        printf("is GPGed by Enigmail Thunderbird Plugin.\n");
    }
    else if ((keys[0] == 166) && (keys[1] == 0)   &&
                (keys[0] == 166) && (keys[1] == 0)) {
        printf("is PGPed with \"pgp -c\".\n");
    }
    else { fprintf(stderr, "sorry, no tips!\n"); }

    printf("fingerprint %.2x %.2x %.2x %.2x %.2x\n", \
              keys[0], keys[1], keys[2], keys[3], keys[4]);

    fclose(file);
    return 0;
}
```

crypto_reveal.c

Procura por *headers* previamente identificados.

Parte 1

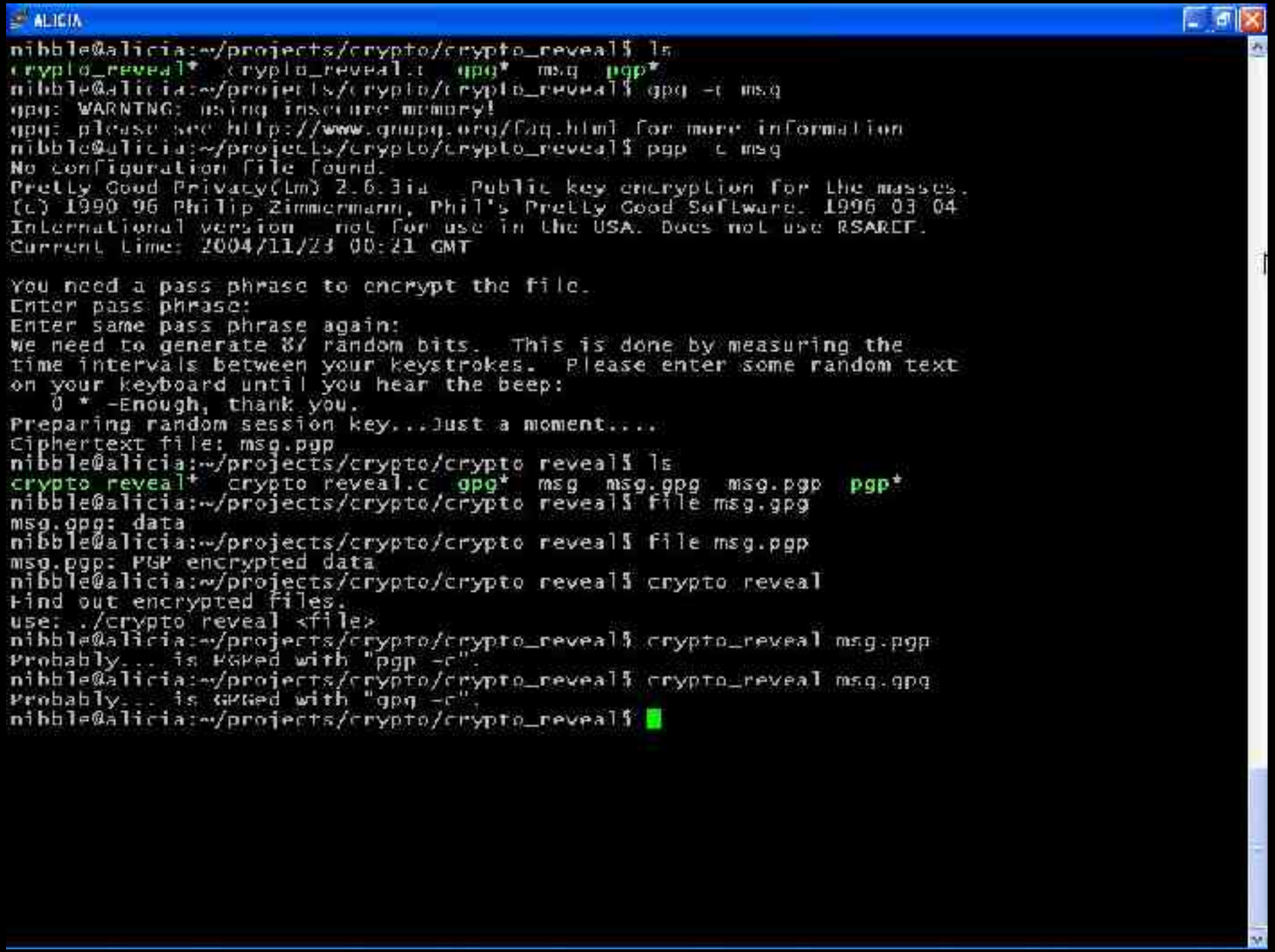
```c
for(i = 0; i <= 5; i++) {
    keys[i] = fgetc(file);
}
```

Parte 2

```c
if ((keys[0] == 140) && (keys[1] == 13) &&
      (keys[2] == 4) && (keys[3] == 3)) {
    printf("is GPGed with \"gpg -c\".\n");
}
```

h2hc_reveal.avi

0’ 54’’


ALICIA
nibble@alicia:~/projects/crypto/crypto_reveal$ ls
crypto_reveal* crypto_reveal.c gpg* msg pgp*
nibble@alicia:~/projects/crypto/crypto_reveal$ gpg -c msg
gpg: WARNING: using insecure memory!
gpg: please see http://www.gnupg.org/faq.html for more information
nibble@alicia:~/projects/crypto/crypto_reveal$ pgp -c msg
No configuration file found.
Pretty Good Privacy(tm) 2.6.3ia - Public key encryption for the masses.
(c) 1990-96 Philip Zimmermann, Phil's Pretty Good Software. 1996-03-04
International version - not for use in the USA. Does not use RSAREF.
Current time: 2004/11/23 00:21 GMT
You need a pass phrase to encrypt the file.
Enter pass phrase:
Enter same pass phrase again:
We need to generate 87 random bits. This is done by measuring the
time intervals between your keystrokes. Please enter some random text
on your keyboard until you hear the beep:
0 * -Enough, thank you.
Preparing random session key...Just a moment....
Ciphertext file: msg.pgp
nibble@alicia:~/projects/crypto/crypto_reveal$ ls
crypto_reveal* crypto_reveal.c gpg* msg msg.gpg msg.pgp pgp*
nibble@alicia:~/projects/crypto/crypto_reveal$ file msg.gpg
msg.gpg: data
nibble@alicia:~/projects/crypto/crypto_reveal$ file msg.pgp
msg.pgp: PGP encrypted data
nibble@alicia:~/projects/crypto/crypto_reveal$ crypto_reveal
Find out encrypted files.
use: ./crypto_reveal <file>
nibble@alicia:~/projects/crypto/crypto_reveal$ crypto_reveal msg.pgp
Probably... is PGPed with "pgp -c".
nibble@alicia:~/projects/crypto/crypto_reveal$ crypto_reveal msg.gpg
Probably... is GPGed with "gpg -c".
nibble@alicia:~/projects/crypto/crypto_reveal$

Preliminares de um ataque

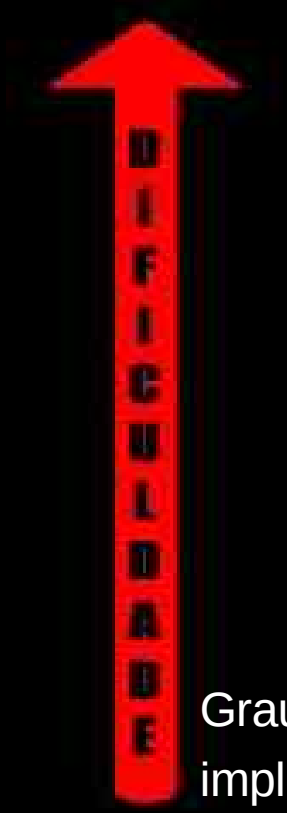
E agora...?

Superado este desafio, busca-se então determinar qual é a chave usada!

Os ataques de criptoanálises podem ser classificados conforme o grau de acesso a informações específicas...

Preliminares de um ataque

Ataques de Criptoanálises



1. Cyphertext-only attack
2. Known-plaintext attack.
3. Chosen-plaintext attack.
4. Chosen-cyphertext attack.

Grau de dificuldade para se implementar um ataque efetivo.

Exemplos de Criptografia Clássica

- Substitution Cypher
- Permutation Cypher
- Vigenère Cypher
- Rotation Cypher
- ...

Criptografia Clássica

Rot13: implementação

Exemplo:

H2HC ⟶ 8-2-8-3

Y = (X + 13) MOD 26

[e.g. 21 = (8+13) mod 26]

UOUP ⟵ 21-15-21-16

TABELA			
A	1	N	14
B	2	O	15
C	3	P	16
D	4	Q	17
E	5	R	18
F	6	S	19
G	7	T	20
H	8	U	21
I	9	V	22
J	10	W	23
K	11	X	24
L	12	Y	25
M	13	Z	26

Rotation: busca extensiva?

A quantidade reduzida de chaves em algoritmos de rotação simples possibilita uma busca extensiva pela chave...

```
 1  #!/bin/bash
 2
 3  if [ -z $2 ]; then
 4     echo "Break ROTation"
 5     echo "use: ./break_rot.sh <file> <guess>"
 6     exit
 7  fi
 8
 9  cat $1 | tr '[b-za-bB-ZA]' '[a-zA-Z]'> rot1.out
10  cat $1 | tr '[c-za-bC-ZA-B]' '[a-zA-Z]'> rot2.out
(...)
33  cat $1 | tr '[z-za-yZ-ZA-Y]' '[a-zA-Z]'> rot25.out
34  cat $1 | tr '[a-zA-Z]' '[a-zA-Z]'> rot26.out
35
36  echo "Break ROTations"
37  echo ""
38  var=1;
39  until [ $var -gt 25 ]; do
40     cat rot$var.out | grep $2 > tmp;
41        T=`cat tmp`
42        if [ -s tmp ]; then
43           echo -n "rot$var match: "
44           echo -n $T
45           echo ""
46        fi
47     var=`expr $var + 1`
48  done
49
50  rm -rf *.out tmp
```

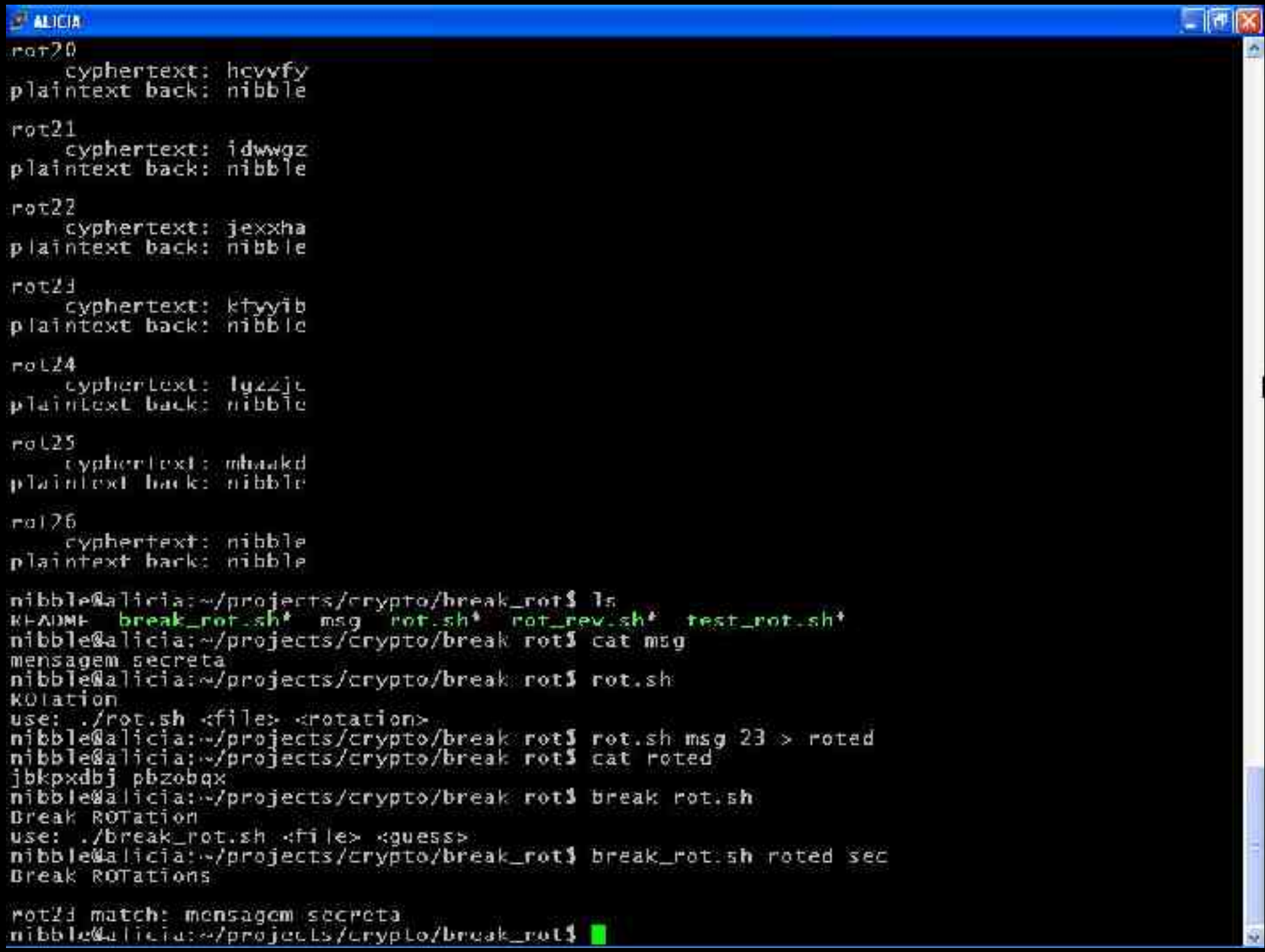
break_rot.sh

Busca extensiva por texto conhecido...

```
36          echo "Break ROTations"
37       echo ""
38       var=1;
39      until [ $var -gt 25 ]; do
40             cat rot$var.out | grep $2 > tmp;
41            T=`cat tmp`
42               if [ -s tmp ]; then
43                  echo -n "rot$var match: "
44                  echo -n $T
45                  echo ""
46               fi
47             var=`expr $var + 1`
48          done
```

h2hc_rot.avi

0’ 51’’


ALICIA
rot20
ciphertext: hcvvfy
plaintext back: nibble
rot21
ciphertext: idwwgz
plaintext back: nibble
rot22
ciphertext: jexxha
plaintext back: nibble
rot23
ciphertext: kfyyib
plaintext back: nibble
rot24
ciphertext: lgzzjc
plaintext back: nibble
rot25
ciphertext: mhaakd
plaintext back: nibble
rot26
ciphertext: nibble
plaintext back: nibble
nibble@alicia:~/projects/crypto/break_rot$ ls
README break_rot.sh* msg rot.sh* rot_rev.sh* test_rot.sh*
nibble@alicia:~/projects/crypto/break_rot$ cat msg
mensagem secreta
nibble@alicia:~/projects/crypto/break_rot$ rot.sh
ROTation
use: ./rot.sh <file> <rotation>
nibble@alicia:~/projects/crypto/break_rot$ rot.sh msg 23 > roted
nibble@alicia:~/projects/crypto/break_rot$ cat roted
jbkpxdbj pbzobqx
nibble@alicia:~/projects/crypto/break_rot$ break_rot.sh
Break ROTation
use: ./break_rot.sh <file> <guess>
nibble@alicia:~/projects/crypto/break_rot$ break_rot.sh roted sec
Break ROTations
rot23 match: mensagem secreta
nibble@alicia:~/projects/crypto/break_rot$

Criptografia Clássica

Substituição: implementação

Baseado em uma tabela basta, troca uma letra por outra!

Plaintext: **CRIPTOGRAPHY**

Cyphertext: **YCZLMFOCXPGD**

TABELA			
A	X	N	S
B	N	O	F
C	Y	P	L
D	A	Q	R
E	H	R	C
F	P	S	V
G	O	T	M
H	G	U	U
I	Z	V	E
J	Q	W	K
K	W	X	J
L	B	Y	D
M	T	Z	I

Criptografia Clássica

Substituição: plaintexts attack

Com acesso a um dado *plaintext* e seu respectivo *cyphertext* o araponga pode começar a montar sua tabela dicionário...

Plaintext: **CRIPTOGRAPHY**

Cyphertext: **YCZLMFOCXPGD**

Tabela			
A	X	O	F
C	Y	P	L
F	P	R	C
G	O	T	M
H	G	Y	D
I	Z		

Criptografia Clássica

Substituição: cypher-only attack

Apenas com o cyphertext o *araponga* tenta então substituições com base na freqüência das letras visto que esta informação não se perde (BINGO!)

Cyphertext:
MSONBVCXZLJHGFDSAPISYTREWQ
WQSERTYIPASDFGOJLZSXCVBONM
SKOJGLFDSAPOIYTREWQMNBVCXZ

Plaintext: (maybe..)
-EA----Z---E----E----T-
T-E------E---A-E----A--
EXA------E--A---T------

Freqüência das Letras			
Mais (+)		Menos (-)	
E	S	Z	H
T	W	X	K
A	O	Q	U

http://www.sonic.net/~sjl/codes/workbench.html

```
(...)
35    ch = getc(file); t[0] = ch;
36        ch = getc(file); t[1] = ch;
37        ch = getc(file); t[2] = ch;
38        t[3] = '\0';
39
40        while(!feof(file)) {
41           i = 0;
42           for(x = 0; x < sizeof(alpha); x++) {
43              triple[0] = alpha[x];
44              for(y = 0; y < sizeof(alpha); y++) {
45                 triple[1] = alpha[y];
46                 for(z = 0; z < sizeof(alpha); z++) {
47                    triple[2] = alpha[z];
48                    i++;
49                    if (!strcmp(triple, t)) {
50                       match_table[i]++;
51           } } } }
52           ch1 = t[1]; t[0] = ch1;
53           ch2 = t[2]; t[1] = ch2;
54           ch = getc(file);
55           if (ch == '\n') ch = getc(file);
56           t[2] = ch; t[3] = '\0';
57        }
(...)
```

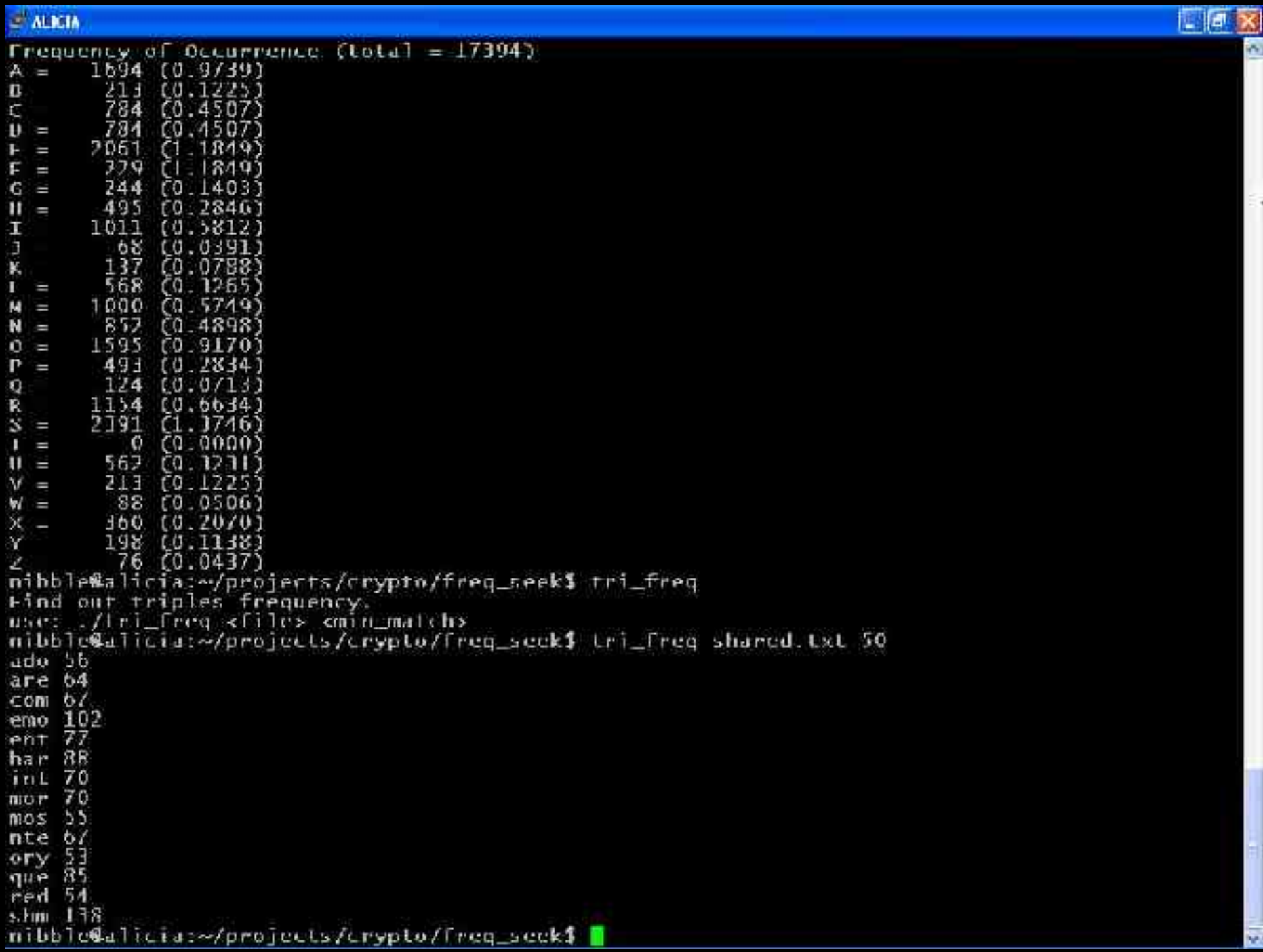
freq_triples.c

Útil contra algoritmos clássicos de chave baseado em palavras.

Este trecho de código foi baseado no algoritmo KMP.

h2hc_freq.avi

0’ 56’’


ALICIA
Frequency of Occurrence (total = 17394)
A = 1694 (0.9739)
B 213 (0.1225)
C 784 (0.4507)
D = 784 (0.4507)
E = 2061 (1.1849)
F = 2061 (1.1849)
G = 244 (0.1403)
H = 495 (0.2846)
I 1011 (0.5812)
J 68 (0.0391)
K 137 (0.0788)
L = 568 (0.1265)
M = 1000 (0.5749)
N = 852 (0.4898)
O = 1595 (0.9170)
P = 493 (0.2834)
Q 124 (0.0713)
R 1154 (0.6634)
S = 2191 (1.1746)
T = 0 (0.0000)
U = 562 (0.1231)
V = 213 (0.1225)
W = 88 (0.0506)
X = 360 (0.2070)
Y 198 (0.1138)
Z 76 (0.0437)
nibble@alicia:~/projects/crypto/freq_seek$ tri_freq
Find out triples frequency.
use: ./tri_freq <files> <min_matchs>
nibble@alicia:~/projects/crypto/freq_seek$ tri_freq shared.txt 50
ado 56
are 64
com 67
emo 102
ent 77
har 88
inL 70
mor 70
mos 55
nte 67
ory 53
que 85
red 54
shm 198
nibble@alicia:~/projects/crypto/freq_seek$

Wake up!

Algoritmos clássicos ainda são bastante usados tanto em aplicações desktop, assim como corporativas...

Criptografia Clássica

Exemplo de Vulnerabilidade

- Em Aplicação Desktop:

FTP Commander (versão 5.8) é um exemplo de aplicação desktop bastante popular que faz uso de criptografia de substituição para guardar senhas de contas em arquivos de texto...

http://www.internet-soft.com/ftpcomm.htm

Criptografia Clássica

Exemplo de Vulnerabilidade

- Em Aplicação Corporativa:

Foram reportados algoritmos de substituição até mesmo em aplicações do Oracle9i, *seria apenas negligência??*

http://www.securityfocus.com/bid/9515

Criptografia Clássica

Material Relacionado

- Wikipedia

http://en.wikipedia.org/wiki/Cryptography/

- "National Cryptologic Museum" da NSA

http://www.nsa.gov/museum/

- Publicação "Solving the Enigma" da NSA

http://www.nsa.gov/publications/publi00016.cfm

Enigma

"Dividindo as àguas" entre a criptografia clássica e moderna teremos a enigma. Concebida em 1919 pelo Holandês Hugo Koch, tornou-se a maquina de encriptação oficial do exercíto alemão durante a segunda guerra.

Enigma: a very famous story of cryptology

http://www.mlb.co.jp/linux/science/genigma/enigma-referat/enigma-referat.html

Enigma: o segredo

O “rotor” da enigma é seu maior trunfo, este implementa uma substituição polialfabética que gera um quantidade astrônimas de possibilidades.

http://w1tp.com/enigma/

http://homepages.tesco.net/~andycarlson/enigma/enigma_j.html

Criptografia Moderna

Bit-a-Bit

Criptografia Moderna

Idéia Geral

- A criptografia moderna busca fazer ofuscação de elementos digitais (contuídos de 0's e 1's).
- Por eficiência tarefas computacionalmente baratas (substituir, permutar...) são usadas.

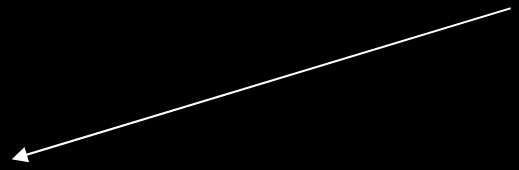
Criptografia Moderna

XOR: ou-excludente

key = 0011

Encrypt

0101 xor 0011 = 0110



0110 xor 0011 = 0101

Decrypt

Tabela Verdade

1 xor 0 = 1

0 xor 1 = 1

1 xor 0 = 1

1 xor 1 = 1

xor.c

Uma chave, um arquivo de entrada e outro de saída são passada em argumento.

```
1   #include <stdio.h>
2
3   // use: xor <key> <in> <out>
4   int main(int ac, char **av)
5   {
6   FILE *in, *out;
7   char *s;
8   int c;
9
10          if ((in = fopen(av[2], "rb")) != NULL) {
11              if ((out = fopen(av[3], "wb")) != NULL) {
12                  while ((c = getc(in)) != EOF) {
13                      if (!*s) s = av[1];
14                      c ^= *(s++);
15                      putc(c, out);
16          } } } // nasty code
17
18          fclose(out);
19          fclose(in);
20      }
```

h2hc_xor.avi

0’ 24’’


ALICIA
nibble@alicia:~/crypto/xor$ ls
msg xor* xor.c
nibble@alicia:~/crypto/xor$ xor key msg msg2
nibble@alicia:~/crypto/xor$ ls
msg msg2 xor* xor.c
nibble@alicia:~/crypto/xor$ file msg2
msg2: data
nibble@alicia:~/crypto/xor$ xor key msg2 msg3
nibble@alicia:~/crypto/xor$ ls
msg msg2 msg3 xor* xor.c
nibble@alicia:~/crypto/xor$ cat msg
Can you see me?
nibble@alicia:~/crypto/xor$ cat msg3
Can you see me?
nibble@alicia:~/crypto/xor$

One-time-pad

Com **key** e **paintext** de mesmo tamanho....

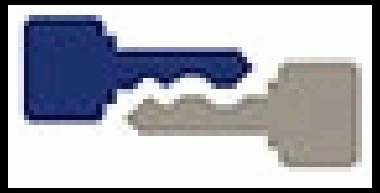
plaintext: 0010101010101
key: 1010111101001
cyphertext: 1000010111110

...temos a criptografia inquebrável,
tanto como inviável !!

Criptografia Moderna

Por que?

Não adianta ter uma boa criptografia simétrica se troca de chaves não é efetuada de forma segura.



Surge então a criptografia assimétrica...

Criptografia Moderna

Simétrica vs. Assimétrica

Algoritmo Assimétrico (duas chaves)

- Possui **duas chaves**, uma chave pública usada na codificação e outra chave privada para decodificação.

Algoritmo Simétrico (chave única)

- Possui **uma única chave** relativamente pequena aos algoritmo assimétricos e são, também, bem mais rápidos.

Criptografia Moderna

Algoritmos “Clássicos”

- DES - Data Encryption Standart

  Algoritmo simétrico alvo de criptoanálise durante décadas. É muito mais rápido, tanto em hardware como software, que RSA.

- RSA – Rivest, Shamir, Adleman

  Algoritmo assimétrico bastante difundido. Possui uma chave bem maior para garantir a mesma segurança que DES.

Criptografia Moderna

Tamanho de Chaves

Equivalência entre resistência de chaves:

Key Length	
Symmetric	**Public-Key**
56 bits	384 bits
64 bits	512 bits
80 bits	768 bits
112 bits	1792 bits
128 bits	2034 bits

ftp://ftp.research.att.com/dist/mab/keylength.txt

Criptografia Moderna

Algoritmos Simétricos

"The security of a symmetric cryptosystem is a function of two things: the strength of the algorithm and the length of the key. The former is more important, but latter is easier to demonstrate."

- Bruce Schneier

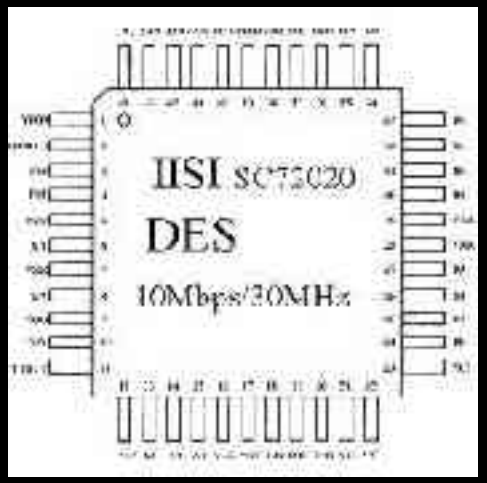
Exemplos: Simétrico

- DES (usado em crypt)
  bloco de 64 bits, chave de 56 bits

- IDEA (usado no pgp)
  bloco de 64 bits, chave de 128 bits

Criptografia Moderna

DES: implementação (1)



DES é implementado tanto em Software como em Hardware.

A segurança do DES é atribuída às S-Boxes...

http://www.allproducts.com/manufacture11/iisi/sc72020.html

DES: histórico (2)

DES foi criado na IBM com base no Lúcifer na década de 70, contudo a NSA fez modificações antes de publicá-lo...

1. Mudaram os valores das S-Box.
2. Diminuíram o tamanho da chave.

DES: controvérsias (3)

Além de aprovar publicamente (algo inédito!) a segurança do DES, a NSA fez alterações no código original...

Desde então, acusam a NSA de ter implantado uma trapdoor no DES!

DES: criptoanálise (4)

A história do DES e da criptoanálise moderna se confundem. Técnicas como...

- **criptoanálise linear**
  http://www.ciphersbyritter.com/RES/LINANA.HTM

- **criptoanálise diferencial**
  http://home.earthlink.net/~mylnir/desdoc.html

...tornaram-se publicas em pesquisa sobre o DES.

Criptografia Moderna

DES: quão seguro afinal?

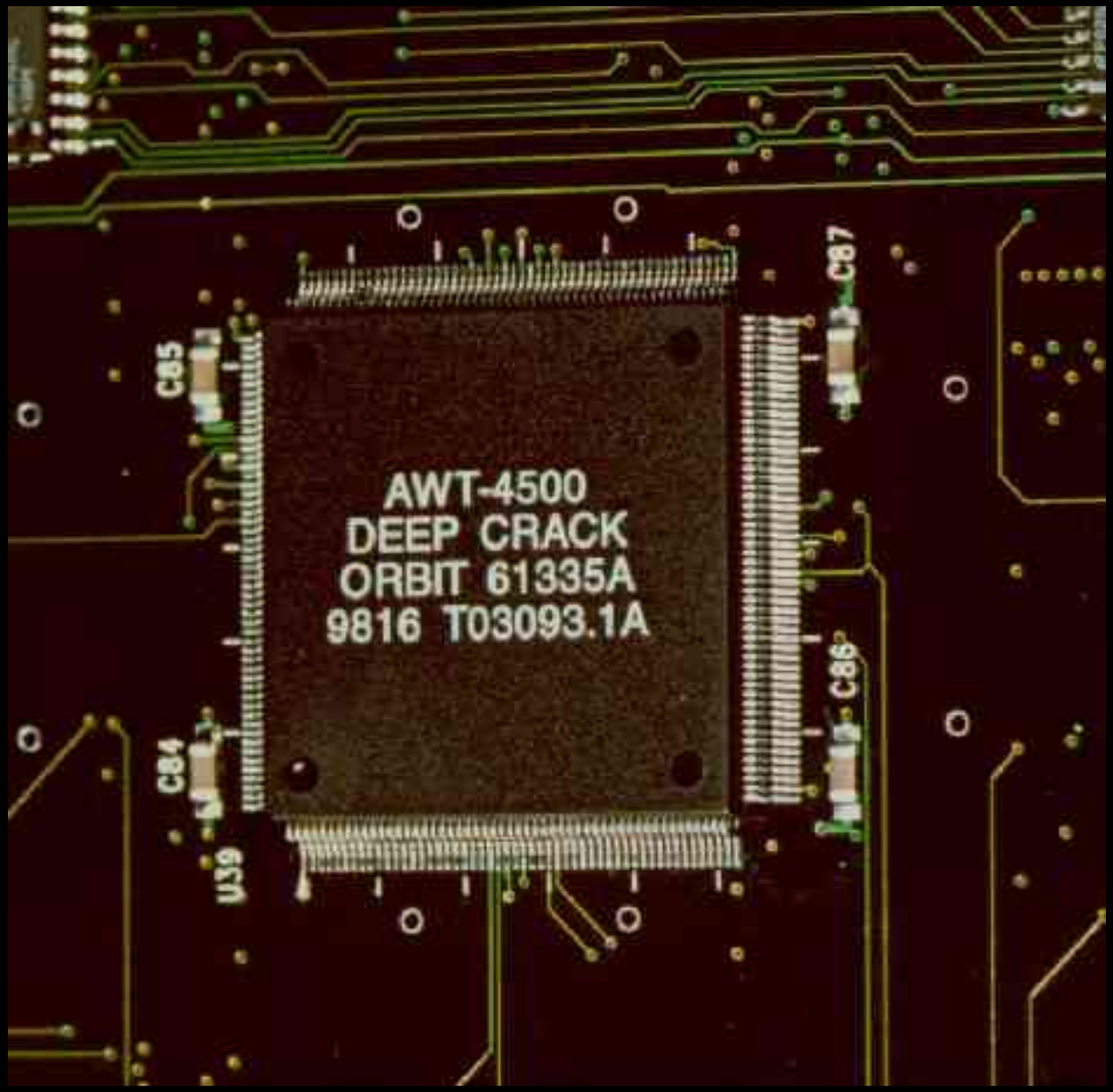
Ataques de força bruta contra DES (**56 bits**) são uma relalidade desde 98!

Coming in at 22 hours 15 minutes, the DES Challenge III was solved by the Electronic Frontier Foundation's (**EFF**) "Deep Crack" in a combined effort with **distributed.net**.

- **RSA Laboratories**

http:/www.eff.org/descracker.html

http://www.distributed.net/des/


AWT-4500
DEEP CRACK
ORBIT 61335A
9816 T03093.1A
C87
C85
C86
C84
U39

Histórico de Ataques

...mas agora é AES!

Aprendeu-se **MUITO** com DES, porém não é mais o atual padrão, vide AES.

http://csrc.nist.gov/CryptoToolkit/aes/

E os cryptoanalistas já correm atrás com novas técnicas, tal como XLS...

http://www.schneier.com/crypto-gram-0210.html#2

Criptografia Moderna

Algoritmos Assimétricos

A segurança da criptografia assimétrica se sustenta sobre variáveis de problemas matemáticos de resolução custosa.

Exemplos: Assimétrico

- RSA_
  (problema: *fatoração*)

- ElGamal
  (problema: *logaritmos discretos*)

Criptografia Moderna: RSA

A segurança do RSA reside na dificuldade de fatorar números grandes.

Simétrico Vs. Assimétrico

Desvantagem do Simétricos

- É necessário passar a chave ao destinatário por um caminho seguro.

Desantagem do Assimétricos

- É muito mas lento e sua chave é bem maior em relação aos algoritmos simétricos.

Criptografia Moderna

Material Relacionado

- A Cryptographic Compendium

http://home.ecn.ab.ca/~jsavard/crypto/intro.htm

- Rijndael becomes AES

http://www.esat.kuleuven.ac.be/~rijmen/rijndael/

- LASEC

http://lasecwww.epfl.ch/

Ambientes Distribuídos

Dividir para conquistar.

Ambientes Distribuídos

O que é um cluster?

Um cluster é um conjunto de computadores que se comportam como um sistema único dedicados a uma dada tarefa...

http://media.lug-marl.de/images/linux/cluster/

Exemplos (ferramentas)

- Beowulf (http://ww.beowulf.org/)
- OpenMosix (http://openmosix.sf.net/)
- MyGrid (http://mygrid.sf.net/)
- DIPC (http://wallybox.cei.net/dipc/)
- Etc.

Ambientes Distribuídos

Beowulf: histórico (1)

- Criado pela NASA por volta 1994.
- Primeiro cluster de alto desempenho e baixo custo disponível para as massas.
- Primeira implementação tinha apenas 16 pcs 486 rodando Linux numa rede Ethernet.

Ambientes Distribuídos

Beowulf: características (2)

- Possui um nó principal (mestre) que controla os demais nós (escravos).
- O mestre é atribui tarefas aos escravos que limitam-se a processa-las.
- Os programas são dedicados, *precisam* fazer uso das bibliotecas específicas.

DESVANTAGEMS: não portabilidade do código, necessidade de adaptação de programa existentes às bibliotecas.

Ambientes Distribuídos

Beowulf: instalação (3)

1. Instalar e Configurar rede ethernet Linux...
2. Instalar e configurar bibliotecas PVM/MPI...
3. Configurar RSH ou SSH para estabelecer uma relação de confiança...
4. Configurar NFS para criar sistema de arquivo centralizado...

Ambientes Distribuídos

OpenMosix: histórico (1)

- Mosix (Multicomputer Operating System unIX) foi criado em Israel na década de 80 para aplicações militares americanas.

- OpenMosix é uma extensão Open Source do projeto Mosix criada em 10/02/2002

Ambientes Distribuídos

OpenMosix: características (2)

- Estrutura totalmente descentralizada, não existe *mestre* e *escravos*.
- Transparente aos usuários em balancemento de carga, controle de memória... A idéia principal do projeto é *fork() and forget* !

DESVANTAGEMS: processo com I/O elevado tendem a não Migrar, ou seja, não fazem uso do ambiente distribuído.

OpenMosix: instalação (3)

1. Instalar e configurar rede ethernet Linux...
2. Aplicar um patch no kernel (do Linux) para o habilitar na migração de tarefas...
3. Usar as diversas aplicações (inclusives gráficas como *OpenMosixView)* inclusas no pacote para ajudar no gerenciamento do sistema...

Ambientes Distribuídos

O que seria um “Grid”?

Grid Computing é a evolução dos cluster, descentralizado, não tão ostensivo e com características interessantes tal como o uso apenas de subsistemas em IDLE...

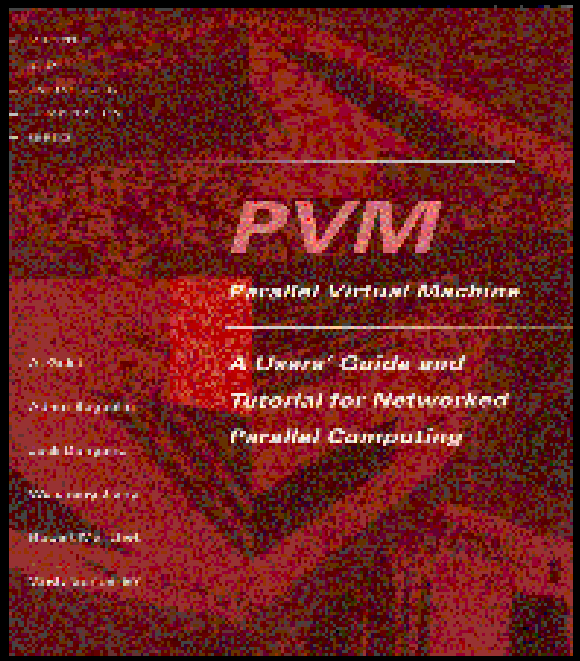
http://www.gridcomputing.com/gridfaq.html

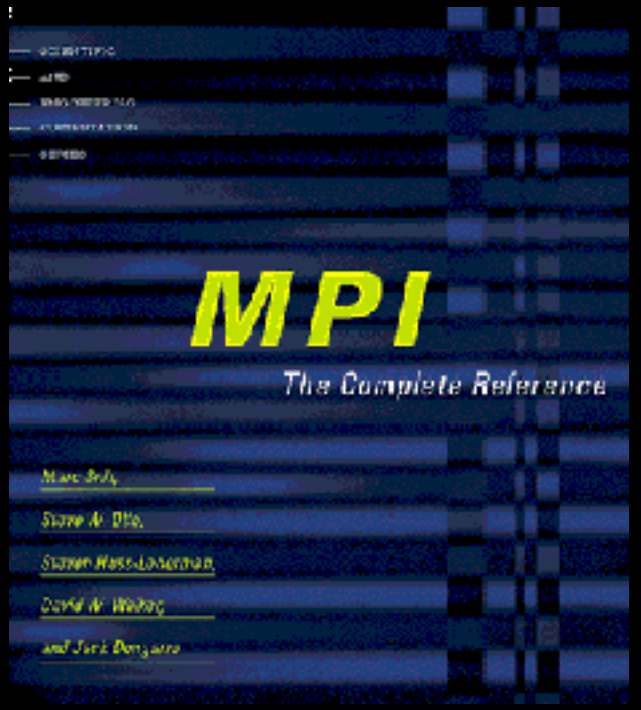
Seria o “IDLE Process” do RuimXP parte de um imenso GRID?

Livros de Referência

PVM - Parallel Virtual Machine

http://www.netlib.org/pvm3/book/pvm-book.html





MPI - Message Passing Interface

http://www.netlib.org/utk/papers/mpi-book/mpi-book.html

Material Relacionado

- Em português

http://www.clubedohardware.com.br/super.html
http://www.revistadolinux.com.br/ed/002/beowulf.php3

- Em Inglês

http://www.top500.org/
http://www.linux-ha.org/
http://www.linux-vs.org/
http://www.mpi.nd.edu/lam/

Agradecimentos

Hackers 2 Hackers Conference

http://www.h2hc.com.br/

“Cryptography products may be declared illegal, but the information will never be.”

- Bruce Schneier in Applied Cryptography