Criptografia de Chave Simétrica

Capítulo 2

Introdução

- O primeiro tipo bastante prático de criptografia é chamado de **criptografia simétrica**.

- Um algoritmo utiliza **uma chave para converter as informações** em algo que se parece com bits aleatórios.

Introdução

- O mesmo algoritmo utiliza a **mesma chave** para **recuperar os dados originais**.

Introdução

- **Gwen** – diretora de vendas.
- **Pao-Chi** – representante de vendas.
- **Atividade** – venda de maquinário de impressão.
- **Produtos** – prensas, ferramentas, peças de reposição, serviços de reparo, treinamento.
- **Clientes** – jornais, gráficas, universidades, outras.

Introdução

- Pao-Chi acaba de receber um memorando de Gwen:

  “…a empresa passa por dificuldades …prepare seus números … ”

- Uma nova política de preços está sendo delineada pela empresa.

Introdução

- No passado, o percentual de desconto baseava-se no tamanho do pedido, nas expectativas de vendas futuras, e outros fatores.

Introdução

- A nova política lista os preços para todos os produtos e também indica o menor preço de venda que os representantes podem negociar.

Introdução

- Agora, o memorando afirma que os representantes de vendas têm autonomia para oferecer descontos ainda maiores.

Introdução

- Pao-Chi quer limitar ao máximo possível quem tem acesso as essas informações.

- Se os clientes potenciais souberem até onde ele está disposto a negociar os descontos, eles teriam vantagem nas negociações.

Introdução

- Os clientes existentes poderiam reivindicar reembolsos.
- Os concorrentes poderiam usar essas informações para ganhar concorrências.
- O mercado de ações da empresa poderia ser afetado …

Introdução

- Como Gwen e Pao-Chi podem manter essas informações em segredo ?

- Não deixar sair do escritório ?
- Memorizá-lo ?

- São 20 páginas.

Introdução

- Pao-Chi resolve **manter uma cópia eletrônica no seu laptop** e toma algumas medidas para proteger o arquivo.

- Medidas comuns de proteção não são suficientes.

Introdução

- Pao-Chi pode perder o seu laptop. Alguém pode furtá-lo.
- Alguém pode examinar seus arquivos enquanto ele está almoçando.
- Para proteger o arquivo, ele decide encriptá-lo.

Introdução

- Pao-Chi obtém um **programa para encriptar seus arquivos** sigilosos.
- Encripta ... Decripta ...
- Problema: **Se os invasor for capaz de obter o aquivo sigiloso, encriptado**, certamente, ele poderá obter o programa de conversão.

Introdução

- Onde Pao-Chi pode, **de maneira segura**, **armazenar** o programa ?

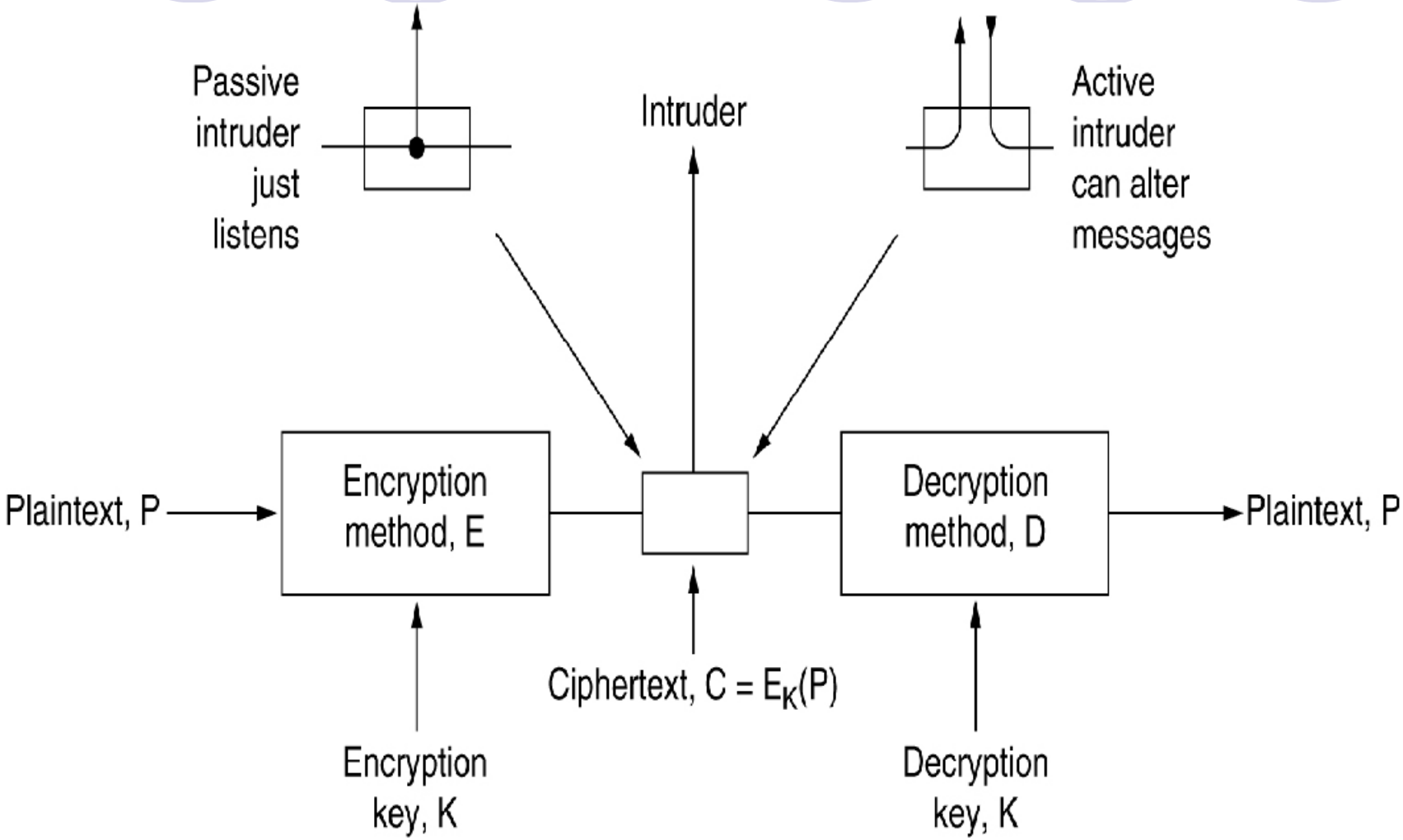
- Se ele puder manter **o programa fora do alcance do invasor**, por que não **armazenar o arquivo sigiloso nesse lugar** ?

Introdução

- Pao-Chi não tem um lugar seguro para tal.
- Se ele tem **acesso a esse lugar seguro**, certamente, **um invasor terá também acesso**.
- Esta é a razão principal porque Pao-Chi utiliza **criptografia**.

Modelo para Criptografia Simétrica

Fonte: Redes de Computadores, A. S. Tanenbaum, Cap. 8



Introdução

- Mas, um **programa de criptografia**, por si só, não pode proteger segredos.
- Pao-Chi precisa de **proteção adicional**.
- Essa proteção adicional é um **número secreto**.

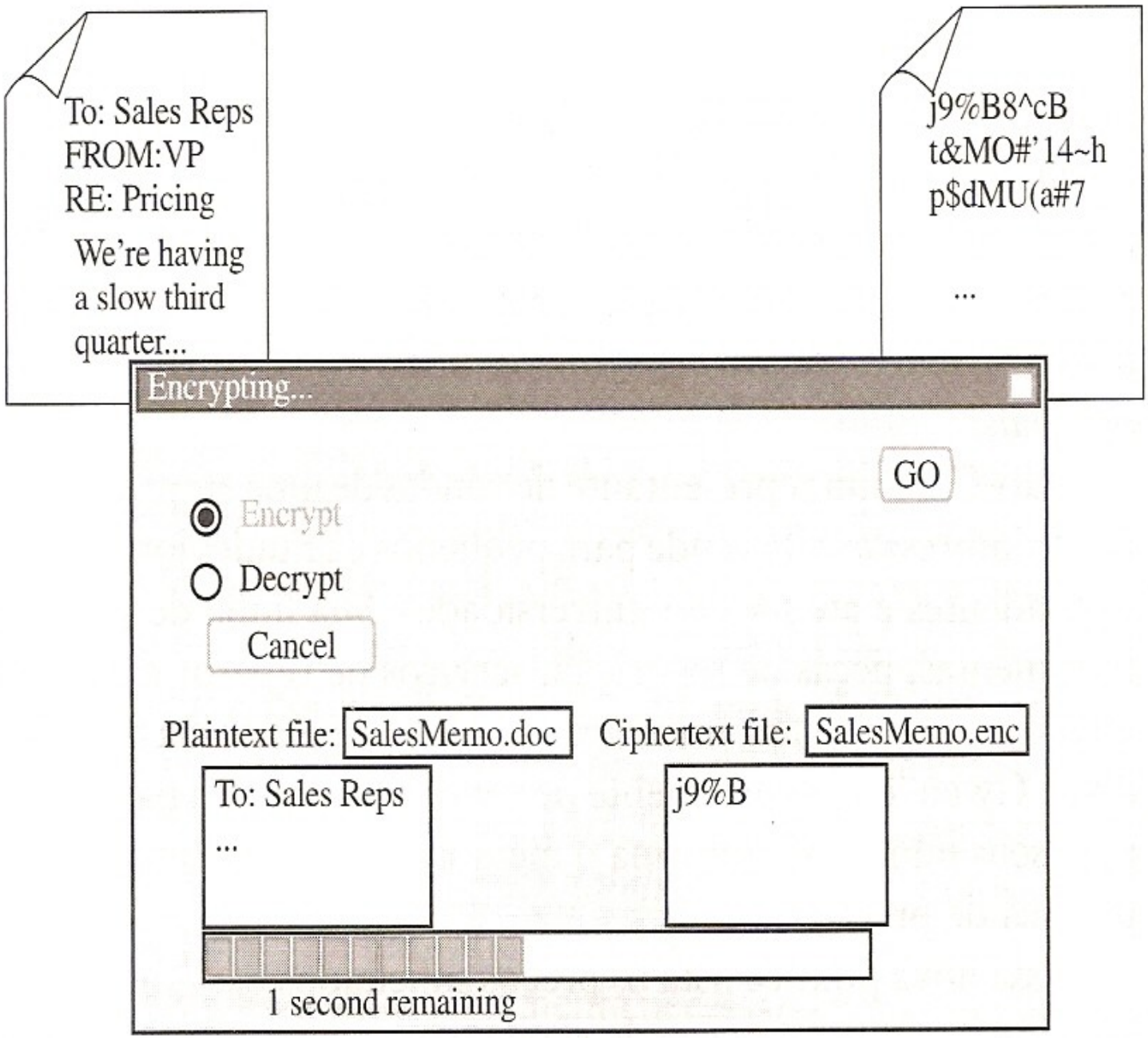
Introdução

- Se ele alimentar o programa com o **arquivo** e o **número secreto**, o programa encriptará o arquivo.

- Enquanto o programa não tenha esse número secreto, ele não será executado.

FIGURA 2-1

Se alimentar seu programa de criptografia com arquivos sigilosos você terá algo que parece não fazer sentido



Introdução

- O problema é que contanto que o arquivo não faça sentido, Pao-Chi também não será capaz de lê-lo.

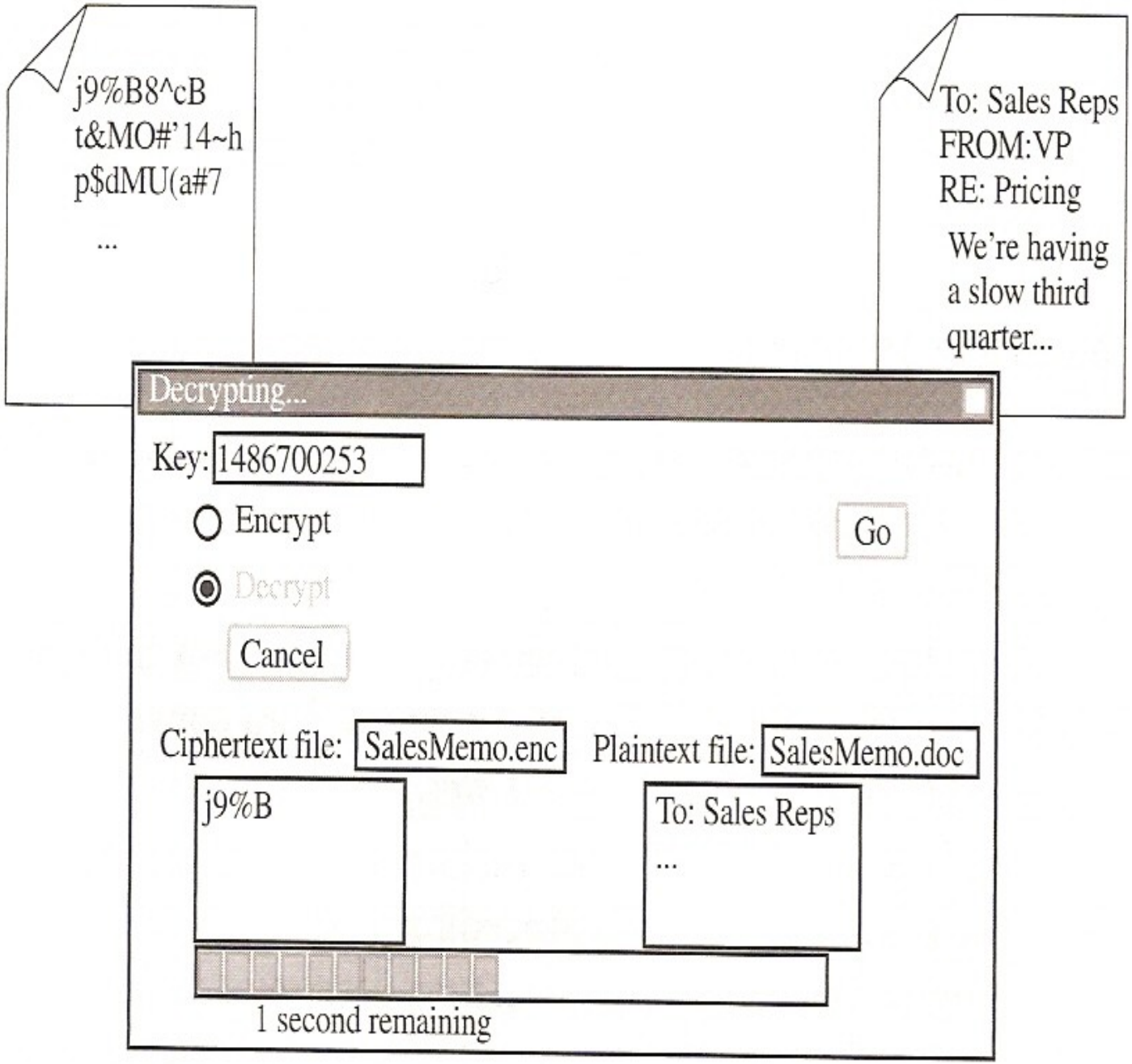
- Para ler o arquivo, Pao-Chi precisa, de alguma maneira, converter a sua forma original. Pao-Chi usa o recurso do Decrypt no programa.

Introdução

- Para **decriptar o arquivo**, Pao-Chi dá ao programa o arquivo encriptado (sem sentido) e **o mesmo número secreto**.

FIGURA 2-2

Para tornar a parte sem sentido encriptada você alimenta uma máquina de criptografia com os dados sigilosos e um número secreto. Para recuperar o arquivo, você pressiona "Decrypt" e então o alimenta com a parte sem sentido e com o número secreto

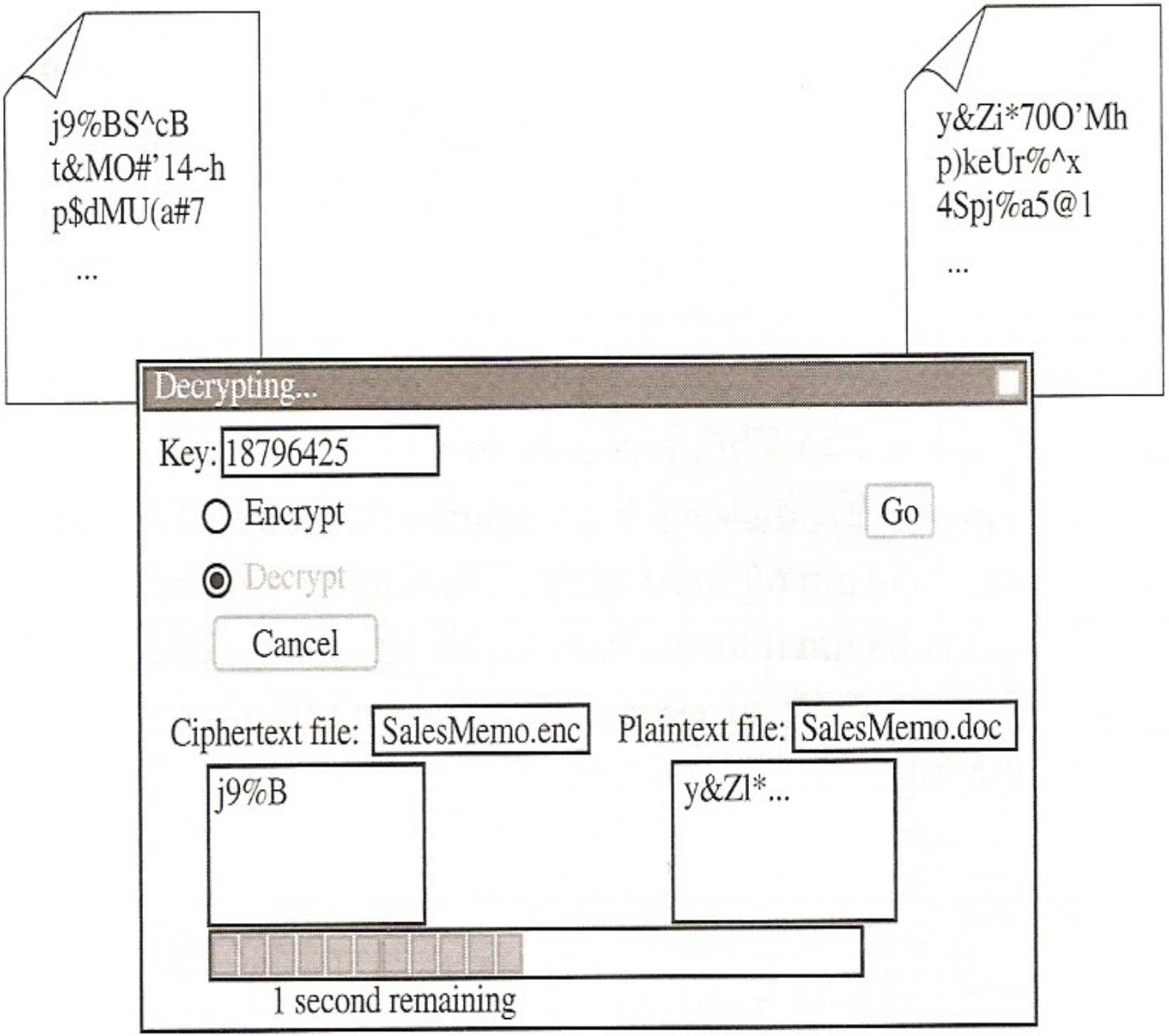


Forjando a chave secreta

- Se invasores tentarem forjar a chave secreta ?
- O que acontecerá ?

FIGURA 2-3

Se invasores tentarem outros números que não o valor secreto, eles obtêm apenas outras coisas sem sentido



O que é uma Chave

- O fato de que o **número secreto** que você escolhe **funcionar da mesma maneira que uma chave** convencional, faz aparecer o termo “**chave**”, para designar esse **número secreto**.

Analogia com o mundo real

Mundo Real	**Mundo Computacional**
Porta	Computador
Fechadura	Algoritmo de Cripto
Chave	Chave

Por que uma chave é necessária.

- Por que **não criar um algoritmo** que **não necessite de uma chave ?**
- O que é mais fácil: **guardar um algoritmo em segredo** ou **guardar uma chave ?**

Por que uma chave é necessária.

- **Aqui está a pergunta mais importante:**

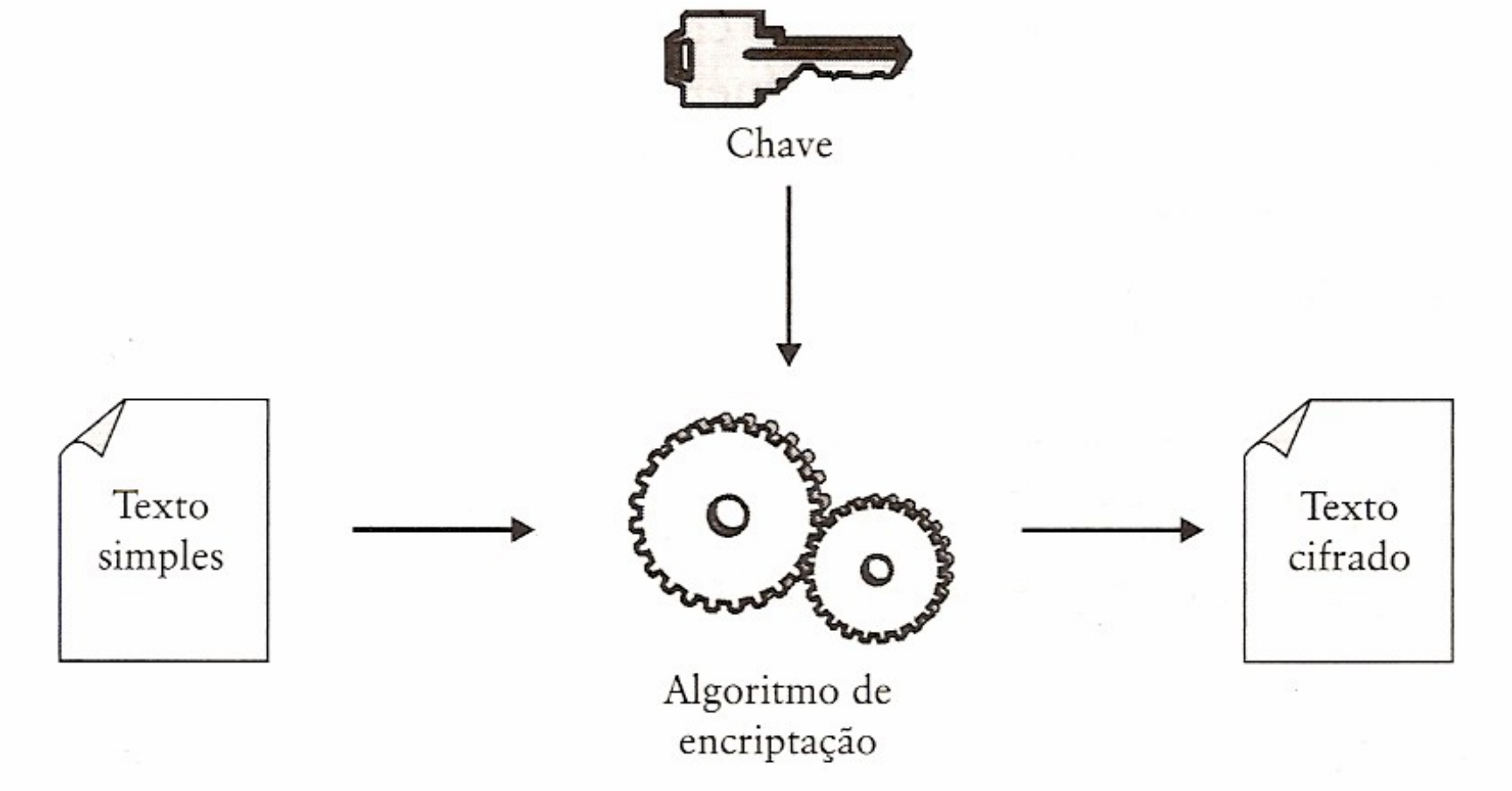
  No que você confiaria mais para manter em segredo?

  Um algoritmo mantido em sigilo ?

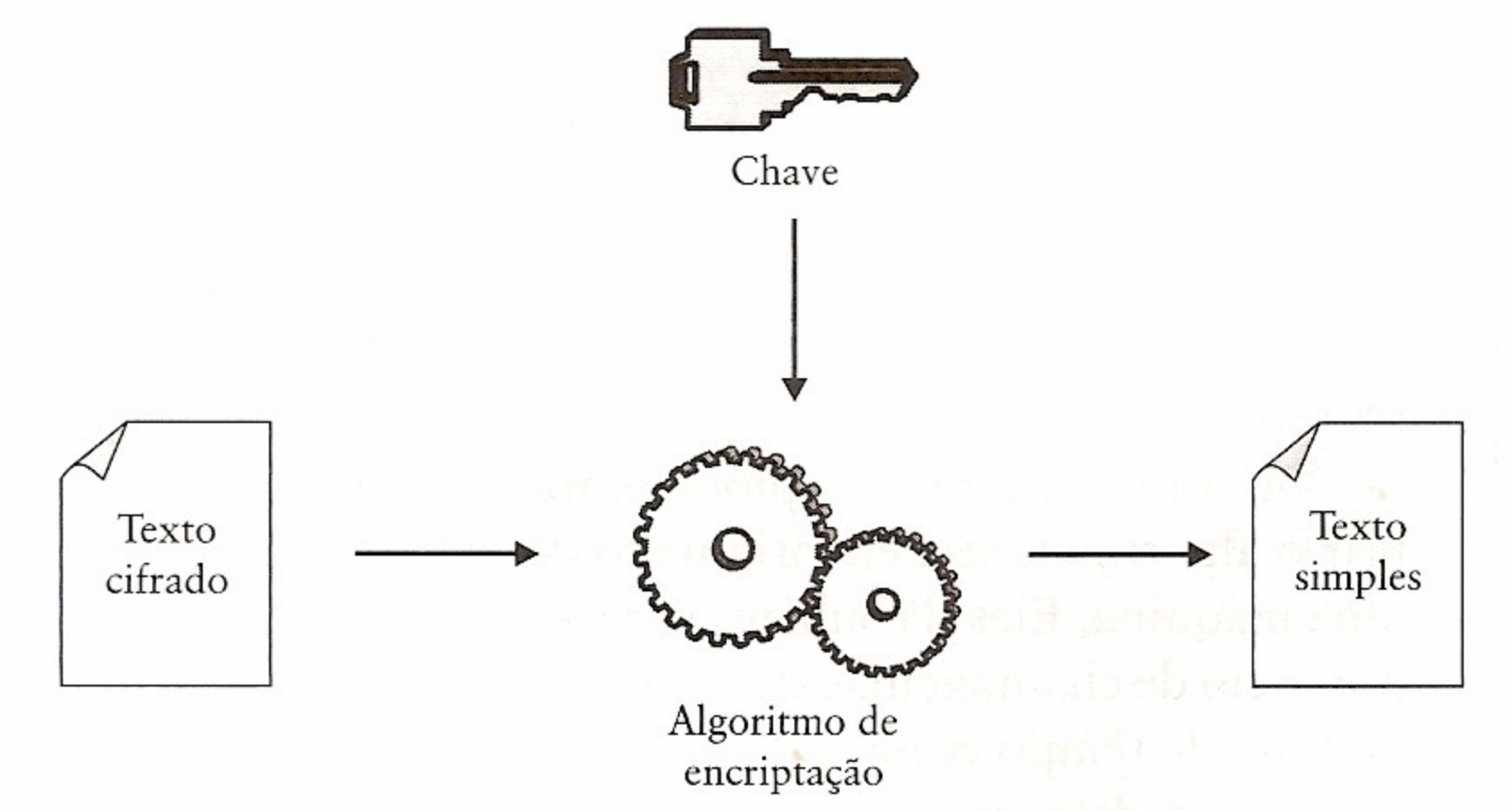
  Ou um algoritmo que pode fazer seu trabalho mesmo todo mundo sabendo como ele funciona?

- **É aqui que as chaves entram.**

Criptografia Simétrica - Criptografando



Criptografia Simétrica - Decriptografando



Por que uma chave é necessária

- As chaves aliviam-nos da necessidade de se preocupar em guardar um algoritmo.

- Se proteger seus dados com uma chave, precisamos apenas proteger a chave, que é mais fácil do que guardar um algoritmo em segredo.

Por que uma chave é necessária

- Se utilizar **chaves para proteger seus segredos** (dados), você poderá utilizar **diferentes chaves para proteger diferentes segredos**.

Por que uma chave é necessária

- Se alguém **quebrar uma das suas chaves**, os **outros segredos ainda estarão seguros**.

Por que uma chave é necessária

- Se você **depender de um algoritmo**, um invasor que **quebre esse algoritmo**, terá **acesso a todos os seus dados sigilosos**.

O segredo deve estar na chave

- A idéia de que **o criptoanalista conhece o algoritmo** e que **o segredo deve residir exclusivamente na chave** é chamada Princípio de Kerckhoff (1883):

Princípio de Kerckhoff

- **Todos os algoritmos devem ser públicos; apenas as chaves são secretas.**

Gerando uma chave

- Em um **sistema criptográfico simétrico**, a **chave** é apenas **um número qualquer**, contanto que tenha um **tamanho correto**.

Gerando uma chave

- Assim, sempre que precisar de uma chave, você deve selecionar um outro número, aleatoriamente.

- Mas, **como selecionar esse número aleatoriamente ?**

O que significa a palavra “aleatória”

- O que <u>não é</u> **aleatório**:

  “Se alguém souber quais são os números ***atuais***, é possível prever os números ***seguintes***?”

Valores Aleatórios (randômicos)

- São conjuntos de números que **não são repetíveis** e passam em **testes estatísticos de aleatoriedade**.

- **Entropia** é a **medida de aleatoriedade** de um conjunto de números.

Teste de aleatoriedade

- Imaginem um conjunto de milhares de números binários.

- Nos testes há perguntas como:
  - Há, aproximadamente a mesma contagem de “1s” e “0s” ?
  - Alguns padrões de “1s” e de “0s” aparecem com muita frequência?

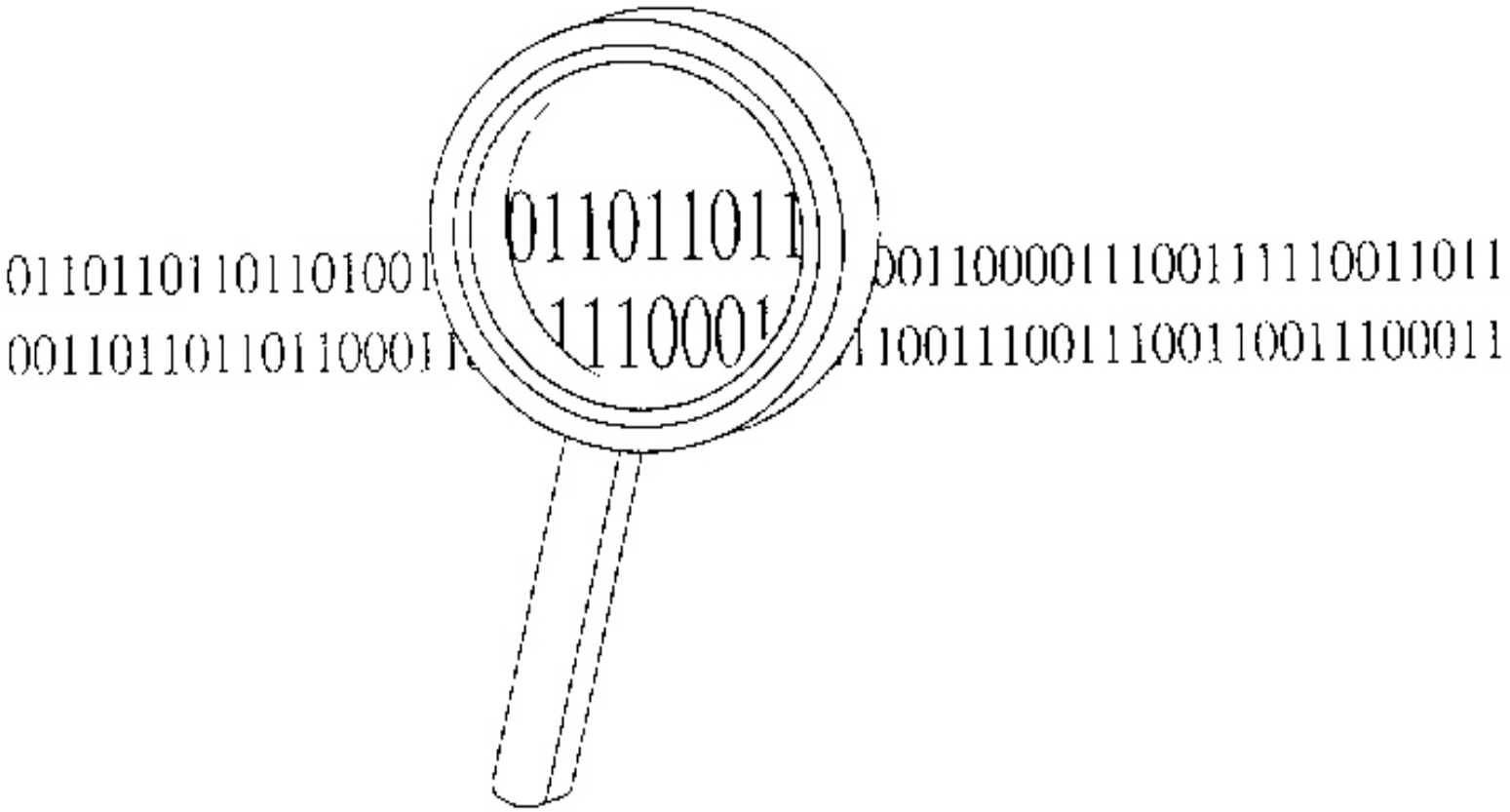
Teste de aleatoriedade

- Se esses números passarem nos testes, dizemos que **provavelmente** os números são aleatórios.

- "**Provavelmente**" aleatórios ?

- Não podemos dizer "**definitivamente**" aleatórios ? **Não podemos**.

Testando a aleatoriedade de níumeros. O padrão 110 aparece com muita frequência.



Um Gerador de Números Aleatórios

- De onde se obteve esses milhares de números ?
- Uma fonte é um RNG (Randon Number Generator).
- Um **RNG** funciona agrupando números de diferentes tipos de entradas imprevisíveis.

RNG

- Se solicitar ao RNG um segundo grupo de números**, praticamente nunca receberemos a mesma sequência** novamente.

- Isso ocorre porque **a saída de um RNG é baseada em uma entrada** que sempre está mudando **(variável e imprevisível)**. Os números não são repetíveis.

Gerador de Números Pseudo-Aleatórios (PRNG)

- Como podemos obter **números aleatórios** se **não tivermos um RNG** ?

- Existem algoritmos que produzem o que é chamado de números “pseudo-aleatórios”.

Gerador de Números Pseudo-Aleatórios (PRNG)

- O que torna esses números **pseudo-aleatórios** e **não aleatórios** é que eles **são repetíveis**.

- Aplicando-se testes estatísticos, esses números passam.

Gerador de Números Pseudo-Aleatórios (PRNG)

- Mas, **se os números são repetíveis**, **para que serve um PRNG** ?

- É que pode-se **alterar a saída** utilizando uma **entrada** (chamada de semente) que **precisamos nos certificar que essa entrada é alterada** todas as vezes que quisermos gerar novos números.

Gerador de Números Pseudo-Aleatórios (PRNG)

- Em um RNG, a entrada estará mudando constantemente, **por conta própria**, de maneira imprevisível.

Gerador de Números Aleatórios (RNG)

- Entrada RNG:
  - Desintegração espontânea de radiotividade,
  - Condições atmosféricas,
  - Minúsculas variâncias elétricas
- Entropia na entrada RNG, **é** muito maior que a entrada de um PRNG.

Gerador de Números Pseudo-Aleatórios (PRNG)

- Uma **entrada PRNG** pode ser:
  - Hora do dia em milisegundos, **ou**;
  - Medidas das constantes alterações do estado dos registradores de computador, **ou**;
  - Entrada de um usuário (pixels na tela dados pela posição de um cursor – um par de números).

.

Gerador de Números Pseudo-Aleatórios (PRNG)

- Uma entrada é construída por um **coletor de semente**.
- Entropia mais baixa que a de um PNG. Qualquer uma das entradas **não é suficiente** em termos de aleatoriedade, mas **agrupando-se** temos uma **imprevisibilidade**.

Gerador de Números Pseudo-Aleatórios (PRNG)

- Por que utilizar um PRNG e não apenas a semente ?
  - **Velocidade**. A coleção de sementes é um processo demorado.
  - **Entropia**. Quanto mais entropia na entrada, mais aleatória será a saída.

Gerador de Números Pseudo-Aleatórios (PRNG)

- Um bom PRNG sempre produz números pseudo-aleatórios independente da semente.
- Se temos uma **“boa”** semente, uma com bastante entropia, o PRNG produzirá números que passam em testes de aleatoriedade.

Gerador de Números Pseudo-Aleatórios (PRNG)

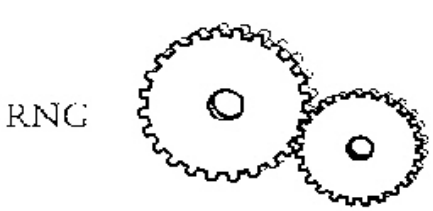
- Se temos uma semente **“ruím”** (ou praticamente **nenhuma semente** ou uma semente **com baixa entropia**), o **PRNG** ainda **produzirá bons números** que passam pelos testes de aleatoriedade.

Gerador de Números Pseudo-Aleatórios (PRNG)

- Mas, então, **por que precisamos de uma boa semente ?**
- **Chaves** são construídas a partir de um **PRNG**s e uma **semente**.
- Alguém **quer ler os dados** que você **criptografou.** E …

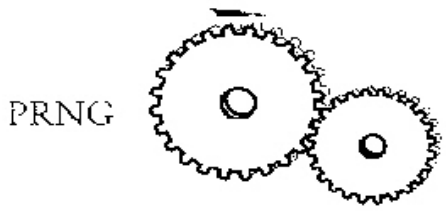
RNG e PRNG

As correntes elétricas são perfeitas. Com um bom dispositivo de medição, você pode ver que há leves variações entre os milissegundos



00111011001001100110101001001101110110000110011001

Informações de semente: horas do dia em milissegundos, informações de thread em registradores "ocultos", movimentos de mouse do usuário, sincronizações de digitação do usuário, mais



11101110100100001100111001110110100000111110130010000

Ataques contra Dados Criptográficos (1)

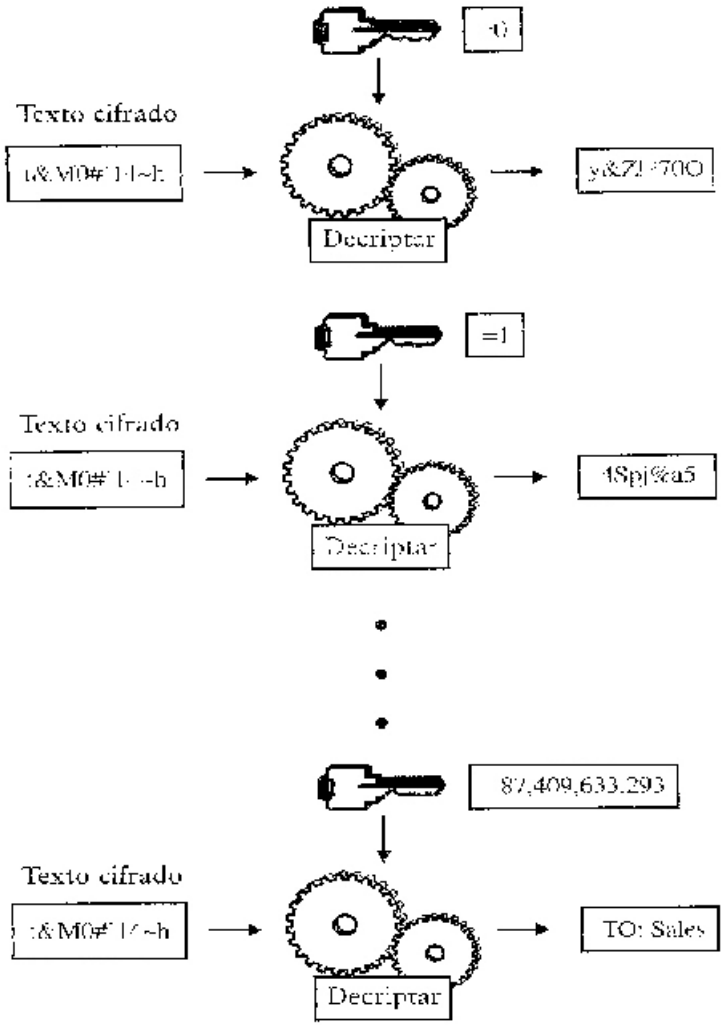
- **Atacando a chave** (Força Bruta) para **reproduzi-la** e **identificá-la.**

- Quebrando o algoritmo

Atacando a Chave

- O ataque de força bruta. Se soubesse que a chave é um número entre 1 e 100.000.000.000 você tentaria uma vez cada número até que produzisse um número, algo que não seja um texto sem sentido.

Atacando a chave (Força Bruta)



Tamanho da Chave

- Esse conceito sobre o intervalo de possíveis chaves é conhecido como **tamanho da chave**.

- Cada bit que você adicionar ao tamanho da chave dobrará o tempo requerido para ataque de força bruta.

Tamanho de Chaves

- Chaves criptográficas são medidas em bits: 40 bits, 56 bits, 64 bits, 128 bits, ...

- Uma chave de 40 bits tem 2^{40} chaves possíveis: aproximadamente, de 0 até 1 trilhão de chaves.

Tamanho de Chaves

- Uma chave de 56 bits tem um intervalo de 0 ate 2^{56} chaves (1 quatrilhão de chaves).

- O intervalo de chaves de 128 bits é tão grande que é mais fácil dizer que uma chave tem 128 bits.

Tempo para Força Bruta na Chave

- **Cada bit** acrescentado ao **tamanho da chave**, **dobrará o tempo** requerido para um ataque de força bruta.

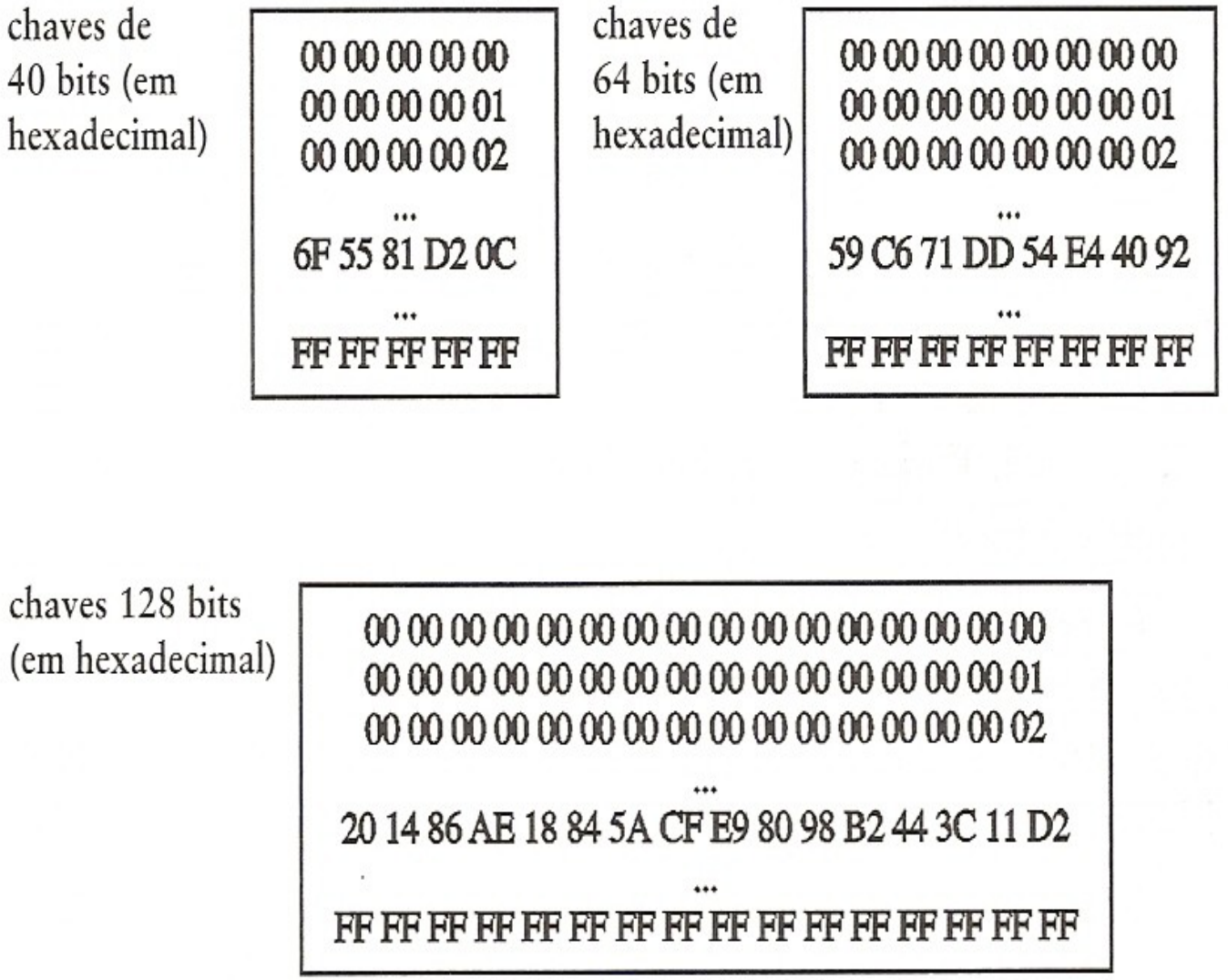
- Porque **cada bit adicional dobra o número de chaves possíveis**.

Tentativas para descoberta de chave

- Na média um invasor tentará a metade de todas as possíveis chaves encontradas.

FIGURA 2-8

Quanto maior o tamanho da chave, maior o intervalo dos valores possíveis que uma chave pode ter. Cada bit em cada posição, seja 0 ou 1, é significativo



Em resumo

- Se você quiser tornar o trabalho de um invasor mais difícil, você deve escolher uma chave maior.

Tabela 2.1 *Um cenário pior do que a pior das situações: quanto tempo um ataque de força bruta levaria quanto aos vários tamanhos de chaves*

Bits	1% do espaço de chave	50% do espaço de chave
56	1 segundo	1 minuto
57	2 segundos	2 minutos
58	4 segundos	4 minutos
64	4,2 minutos	4,2 horas
72	17,9 horas	44,8 dias
80	190,9 dias	31,4 anos
90	535 anos	321 séculos
108	140.000 milênios	8 milhões de milênios
128	146 bilhões de milênios	8 trilhões de milênios

Atacando a Semente (1)

- **Ou**, em vez de **tentar reproduzir** a chave, o invasor pode **tentar reproduzir o PRNG e a semente** que foi utilizada para **criar a chave**.

Eles sempre descobrem o algoritmo … (1)

- O invasor **conhece um PRNG** específico e o método de **coleta da semente** que foi utilizada.

- Se a **semente** for “**boa**”, maior dificudade terá o invasor para descobrí-la e **reconstruir a mesma chave**.

Ataques contra Dados Criptográficos (2)

- **Quebrando o algoritmo** (análise sobre possíveis fraquezas no resultado do algoritmo).

- A partir do **texto cifrado**, o invasor identifica certas “combinações de bits” e suas localizações no **texto cifrado**.

Ataques contra Dados Criptográficos (2)

- Um invasor poderá **examinar o texto cifrado** e **decifrar as partes do texto claro**, mesmo **sem conhecer a chave**.

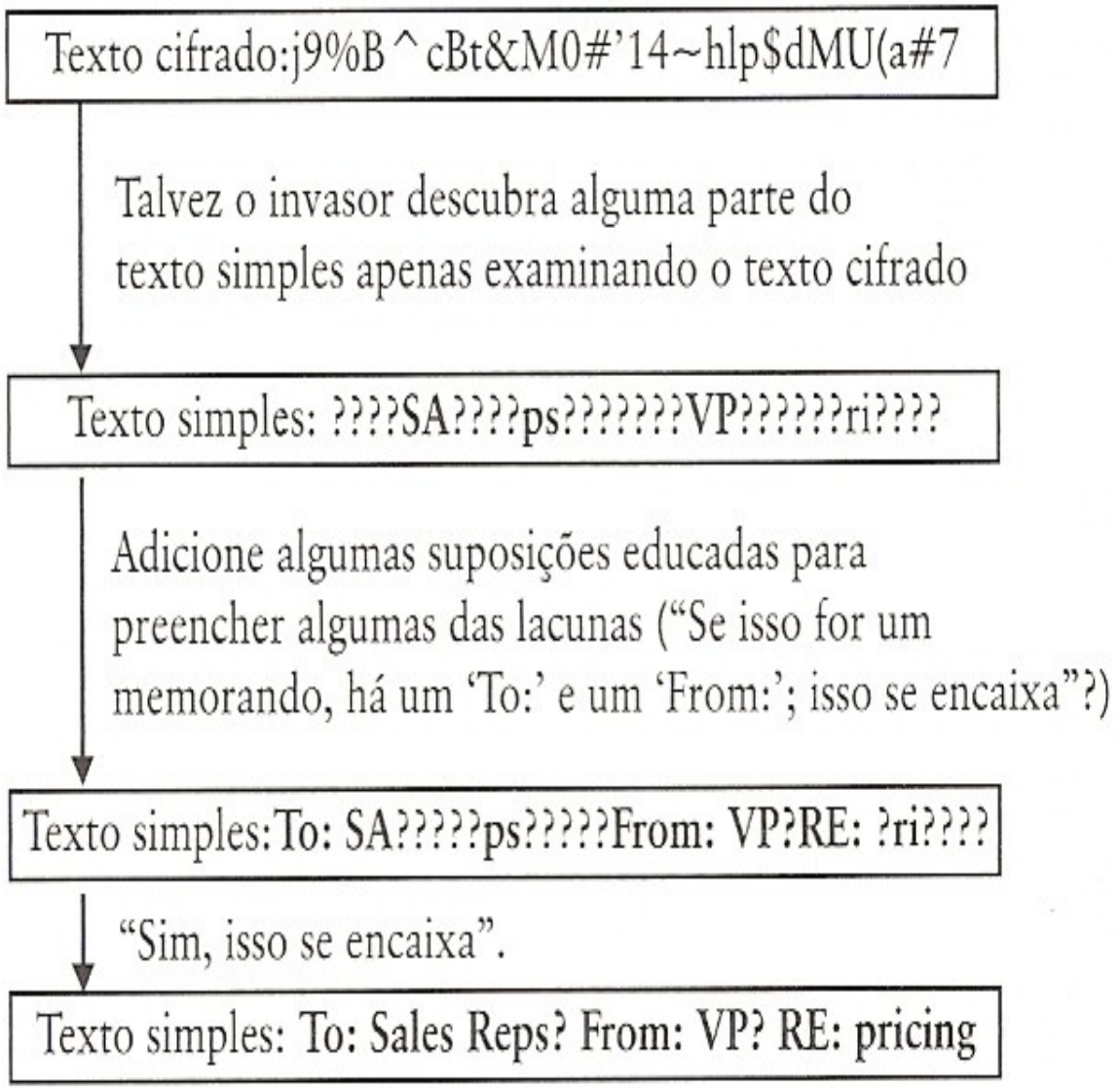
- **Parte da mensagem original** pode ser suficiente para causar danos.

Ataques contra Dados Criptográficos (2)

- Se alguém puder **computar a chave** a partir de um **pedaço do texto cifrado** e do **texto claro correspondente**, o restante da mensagem do texto claro poderá ser descoberta.

FIGURA 2-9

Se um algoritmo tem uma fraqueza, um invasor descobre partes do texto simples sem a chave, reconstruindo a maior parte ou toda a mensagem



Ataques contra Dados Criptográficos (3)

- **Quanto tempo se leva para decifrar uma mensagem ?**
- Em geral, quanto maior a chave, mais tempo levará.
- Entretanto, se o **algoritmo for fraco** não importa qual seja o **tamanho** desta.

Algoritmos Simétricos

- A Tabela de Chaves

  Todos os algoritmos simétricos usam a chave para construir uma **tabela de chaves**.

Tabela de Chaves

- A tabela é um **vetor** de elementos **pseudo-aleatórios** com um tamanho e formato específicos.

- A formação da tabela é chamada de **inicialização de chave** (Vetor de Inicialização)

- **É essa tabela que realiza a criptografia**.

Tabela de Chaves

- Chaves com diferentes comprimentos, num mesmo aplicativo.
- Evitar ataques contra o algoritmo.

Algoritmos de Chave Simétrica

- **Tradicionalmente**, as pessoas que criaram a criptografia (**substituição** e **transposição**), utilizaram algoritmos simples.

- Embora a **criptografia moderna** utilize as mesmas idéias básicas da criptografia tradicional (*substituição* e *transposição*), sua ênfase é diferente.

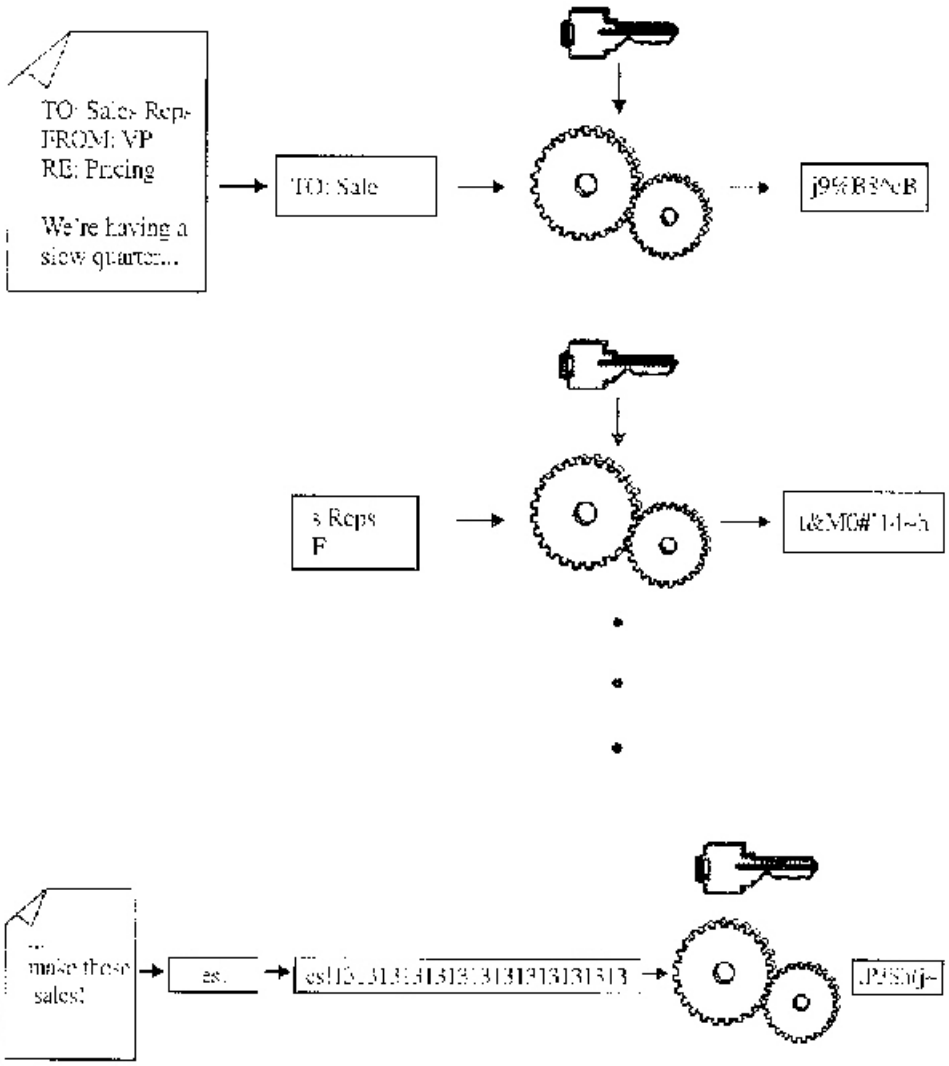
Criptografia Moderna

- Mas, atualmente, o objetivo é **tornar o algoritmo de criptografia tão complexo** que, mesmo que um criptoloanalista adquira volume significativo de texto cifrado, **sem a chave ele não será capaz de captar qualquer sentido** em tudo o que conseguir.

Algoritmos de Chave Simétrica

- Modos de Cifra
  - Cifra de Produto
  - Electronic Code Book
  - **Encadeamento de blocos de cifras**
  - Feedback de Cifra
  - **Cifra de fluxo**
  - Contador

Cifragem de Blocos



Nota técnica: XOR

O termo *XOR* significa "exclusive OR – OU exclusivo", um tipo de manipulação de bits. O primeiro conceito a ser apreendido é um OR (OU). Um OR é uma manipulação de bits que informa, "examine dois bits. Se um outro OR estiver configurado, configure o resultado".

```
0 OR 0 = 0 (zero OR zero é igual a 0)
0 OR 1 = 1 (zero OR um é igual a 1)
1 OR 0 = 1 (um OR zero é igual a 1)
1 OR 1 = 1 (um OR um é igual a 1)
```

Um OR exclusivo informa, "examine dois bits. Se um estiver exclusivamente configurado, OU se o outro estiver exclusivamente configurado, configure o resultado". Se ambos os bits estiverem configurados, então não há nenhuma exclusividade; portanto, o bit resultante não é configurado.

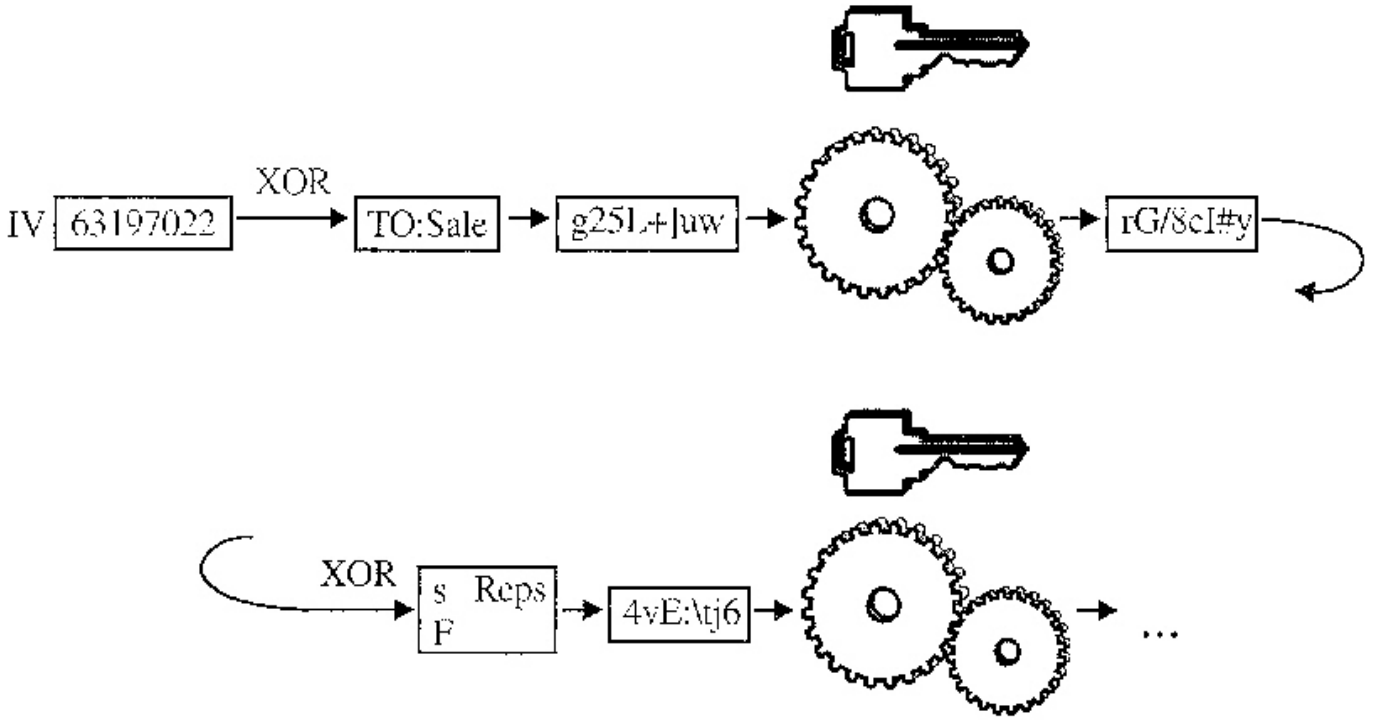
```
0 XOR 0 = 0 (zero XOR zero é igual a 0)
0 XOR 1 = 1 (zero XOR um é igual a 1)
1 XOR 0 = 1 (um XOR zero é igual a 1)
1 XOR 1 = 0 (um XOR um é igual a 0)
```

XOR é uma manipulação de bits útil na criptografia, visto que em 50% dos casos o resultado é 1 e nos outros 50% é 0. Se um bit for um texto simples e um bit for fluxo de chave, então o fluxo de chave às vezes altera o bit e às vezes não o altera.

No primário, aprendemos como adicionar, subtrair e multiplicar colunas utilizando:

```
1.482      77      204
+ 319      - 5     * 8
-----      ---     ----
1.801      72      1632
```

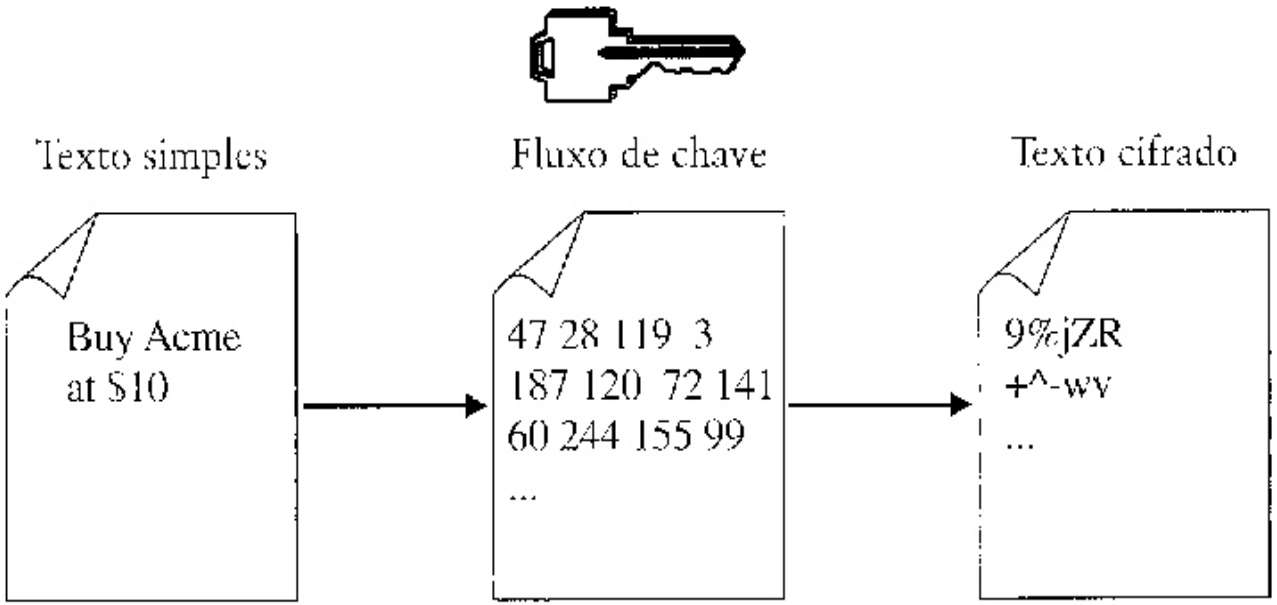
Cifragem de Blocos por Encadeamento



Cifragem de Fluxo

- O algoritmo gera, com base na chave, um padrão criptográfico, tão grande quanto necessário.
- XOR do texto simples com o padrão criptográfico gerado.
- Esse padrão é conhecido como **fluxo de chave** (Tabela de Chaves).

Cifragem de Fluxo



Cifragem de Fluxo

- **Para encriptar:** um byte do texto-simples é tomado, vai para a tabela de chaves, de alguma maneira obtém um byte do fluxo de chaves e opera um XOR com o byte do texto simples.

- Descarta o byte da tabela de chaves. Mescla a tabela novamente. Obtém o byte seguinte dos dados, e assim sucessivamente.

Qual o melhor ?

- Cifragem de fluxo é mais rápida.
- Têm menos código.

- Escolhe-se cifragem de bloco por ser um padrão. Todo mundo, em geral tem dois algoritmos: DES e AES.

Tabela 2.2 *Escolhendo um algoritmo por aplicação*

Aplicação	Cifragem a ser utilizada	Comentários
Banco de dados	Bloco	A interoperabilidade com um outro software não é uma questão, mas você precisará reutilizar as chaves.
E-mail	AES	Embora cada mensagem de e-mail tenha sua própria chave e você possa utilizar uma cifragem de fluxo, você ganha interoperabilidade em todos os pacotes de e-mail utilizando o AES padrão.
SSL (conexões seguras na Web)	RC4 (cifragem de fluxo)	A velocidade é extremamente importante, cada conexão pode ter uma nova chave e praticamente todos os navegadores e servidores da Web possuem uma RC4.
Criptografia de arquivo (armazenando seus arquivos de maneira segura)	Bloco	A interoperabilidade não é uma questão, porém você pode encriptar cada arquivo com a mesma chave e então proteger essa chave (consulte o Capítulo 3).

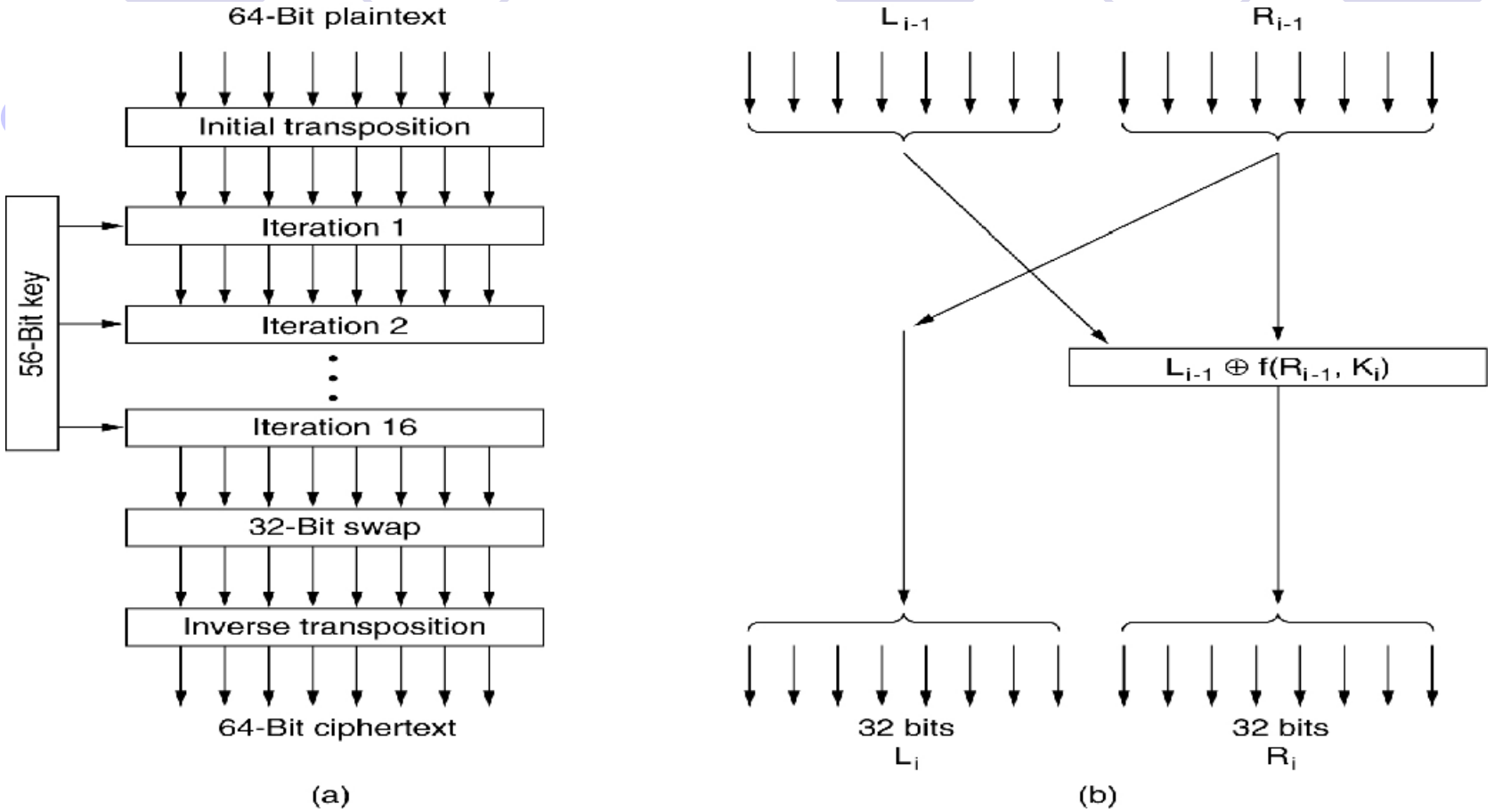
DES – Data Encryption Standard

- Autor: IBM, janeiro de 1977
- Chave: 56 bits
- Comentário: Muito fraco para uso atual.

Triple DES

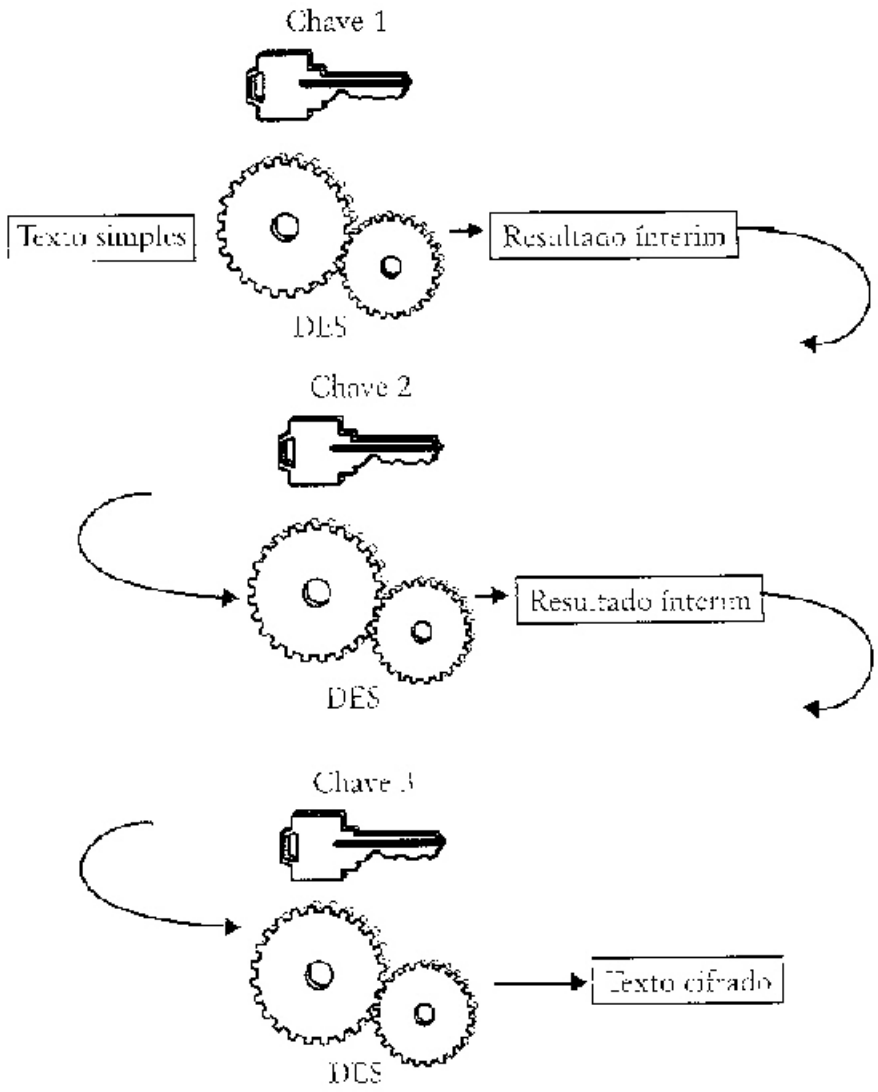
- Autor: IBM, início de 1979.
- Chave: 168 bits
- Comentário: **Segunda melhor escolha**.

Data Encryption Standard

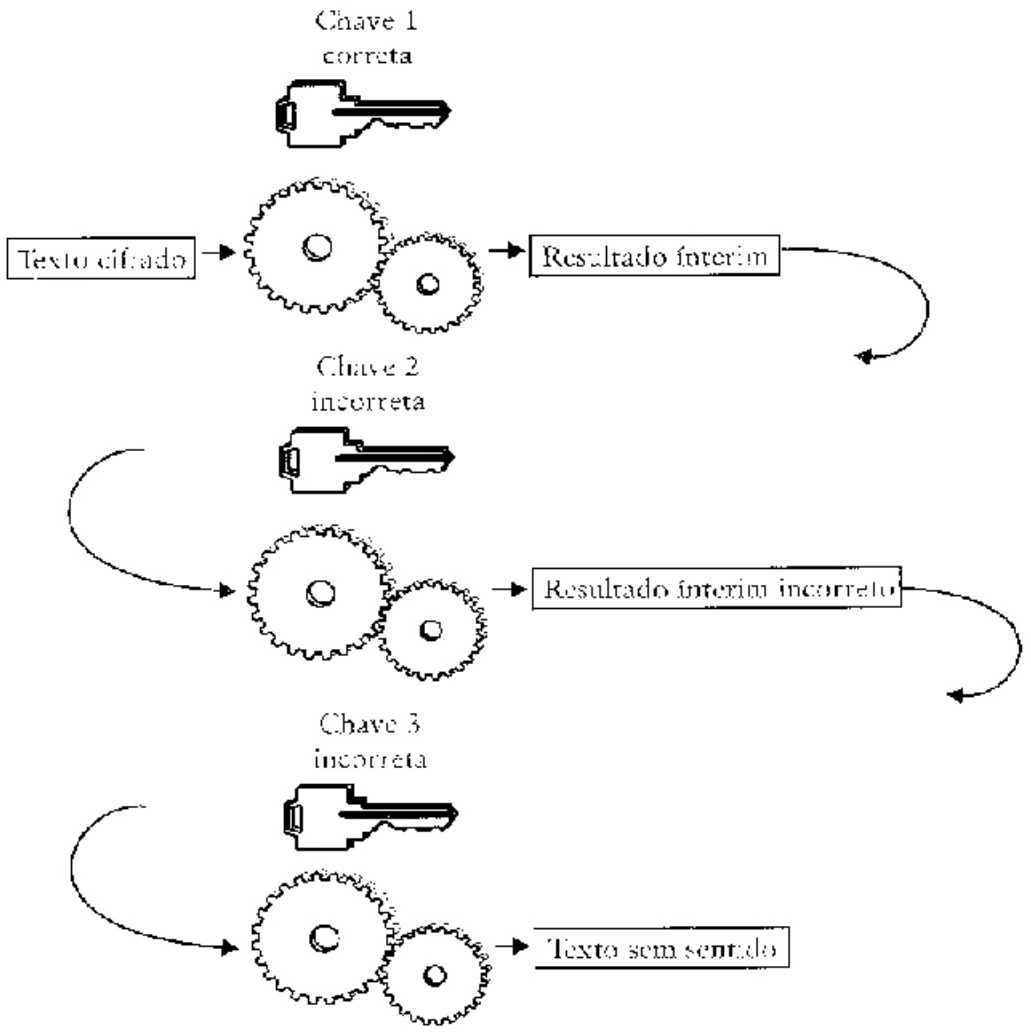


The data encryption standard. (a) General outline.
(b) Detail of one iteration. The circled + means exclusive OR.

Triple DES


Chave 1
Texto simples
DES
Resultado interim
Chave 2
DES
Resultado interim
Chave 3
DES
Texto cifrado

Quebrando Chaves no Triple DES



Substituições comerciais do DES

- Em resposta ao **tamanho da chave** e aos **problemas de desempenho** relacionados ao **Triple DES**, criptógrafos e empresas comerciais desenvolveram **novas cifras de bloco**.

Substituições comerciais do DES

- Blowfish (Counterpane Systems)
- RC2 (RSA)
- RC4 (RSA)
- IDEA (Ascon)
- Cast (Entrust)
- Safer (Cylink)
- RC5 (RSA)

Substituições comerciais do DES

- Enquanto **DES** e **Triple DES** requeriam **chaves de tamanho fixo** (40, 56 bits, respectivamente), suas substituições comerciais eram **mais rápidas** e capazes de operar com **chaves maiores e e tamanho variável**.

Substituições comerciais do DES

- Pode-se **escolher um tamanho de chave que seja suficientemente grande** para tornar o seu algoritmo criptográfico imune a um **ataque de força bruta sobre a chave**, ou ao menos **tornar o ataque de força bruta impraticável**.

Substituições comerciais do DES

- As diferentes substituições comerciais do DES prosperaram em algum grau e as empresas construiram produtos utilizando os algoritmos.

Substituições comerciais do DES

- Mas, **nenhuma proposta** se tornou **um padrão mundial comparável ao DES ou ao Triple DES**.

Substituições não comerciais

- Serpent
- Twofish

Blowfish

- Autor: Bruce Schneier
- Chave: 1 a 448 bits
- Comentário: Velho e lento.

RC2

- Autor: Ronald Rivest, RSA Data Security
  Meado dos anos 80.

- Chave: 1 a 2048 bits
  40 bits para exportação

- Comentário: quebrado em 1996.

RC4

- Autor: **Ronald Rivest,**
  RSA Data Security, 1987
- Chave: 1 a 2048 bits
- Comentário: **Algumas chaves são fracas**.
- Usado como componente do SSL (Netscape)

IDEA – International Data Encryption Algorithm

- Autor: **Massey & Xuejia, 1990**.
- Chave: 128 bits
- Comentário: **Bom, mas patenteado**.
- Usado no PGP.

RC5

- Autor: Ronald Rivest,
  RSA Data Security, 1994.
- Chave: 128 a 256 bits
- Comentário: **Bom, mas patenteado**.

Twofish

- Autor: Bruce Schneier, 1997
- Chave: 128 a 256 bits
- Comentário: **Muito forte, amplamente utilizado**.

Serpent

- Autor: Anderson, Biham, Knudsen 1997
- Chave: 128 a 256 bits
- Comentário: **Muito forte**.

Rijndael (Origem do AES)

- Janeiro de 1997,

- **NIST** (**National Institute of Standards and Technology**), órgão do Departamento de Comércio dos EUA, encarregado de aprovar padrões para o governo federal dos EUA,

Rijndael

- patrocinou um **concurso** para um **novo padrão criptográfico para uso não-confidencial**.

Rijndael

- A ser chamado **AES** (**Advanced Encrytion Standard**)

- Regras do concurso:
  - O algoritmo deveria ser uma cifra de bloco simétrica.
  - Todo o projeto deveria ser público.
  - Tamanho de chaves: 128, 192, 256 bits
  - Implementado, possivelmente, em SW e HW.
  - O algoritmo deveria ser público ou licenciado em termos não-discriminatórios.

Rijndael

- **15 propostas**, **conferências públicas**, **análises criptográficas** para encontrar falhas.

Rijndael

- **Agosto de 1998** foram selecionados 5 propostas finalistas.

- Requisitos de segurança:
  - Eficiência
  - Simplicidade
  - Flexibilidade
  - Memória (importante para sistemas embutidos)

Rijndael

- **Ultima votação**:
  - **Rijndael** (Daemen, Rijmen) – **86 votos**
  - Serpent (Anderson, Biham, Knudsen) – 59 votos
  - Twofish (Bruce Schneier) – 31 votos
  - RC6 (RSA) – 23 votos
  - MARS (IBM) – 13 votos

Rijndael

- Autor: Daemen & Rijmen
- Chave: 128 a 256 bits
- Comentário: **Melhor escolha**.

Rijndael

- **Outubro de 2000**, eleito pelo concurso com o voto do NIST.

- **Novembro de 2001**, o Rijndael **se tornou o padrão do governo dos EUA**, publicado como o Federal Information Processing Standard (FIPS 197).

Rijndael

- O algoritmo foi projetado não só por **segurança**, mas também para **aumentar a velocidade**.

- Uma **boa implementação de software** em uma máquina de **2 GHz** deve ser capaz de alcançar uma **taxa de criptografia** de **700 Mbps**, … … …

Rijndael

- … … … que é rápida o suficiente para codificar **mais de 100 vídeos MPEG-2** em tempo real.

AES (novo nome para o Rijndael)

- **A**dvanced **E**ncryption **S**tandard
- **Tamanho do Bloco**: 128 bits
- **Comprimento da Chave**: **128**, 192, **256** bits.

AES

- Atual: **128/128** bits ou **128/256** bits
- Um **tamanho de chave de 128 bits**, oferece um espaço de **2^{128} chaves**.

Segurança do AES

- Ainda que a **NSA** (**National Security Agency, EUA**) consiga construir uma máquina com **1 bilhão de processadores paralelos**,

Segurança do AES

- cada um capaz de **avaliar uma chave por pico-segundos**, tal máquina levaria cerca de **10^{10} anos** para **pesquisar esse espaço de chaves**.

A Matemática do AES

- Baseado na **Teoria de Campo** de **Galois** (matemático francês).

- O que proporciona ao algoritmo **propriedades de segurança demonstráveis**.