Segurança da Informação

Modos de Operação e Aleatoriedade

Igor Machado Coelho

10/06/2024–19/06/2024

Section 1

Módulo: Modos de Operação e Aleatoriedade

Pré-Requisitos

São requisitos para essa aula o conhecimento de:

- Redes de Computadores (conceitos gerais)
- Módulo 1: princípios básicos
- Módulo 2: ameaças
- Módulo 3: requisitos
- Módulo 4: malware e vírus
- Módulo 5: worms
- Módulo 6: engenharia social e carga útil
- Módulo 7: contramedidas
- Módulo 8: negação de serviço
- Módulo 9: criptografia simétrica

Tópicos

- Modos de Operação

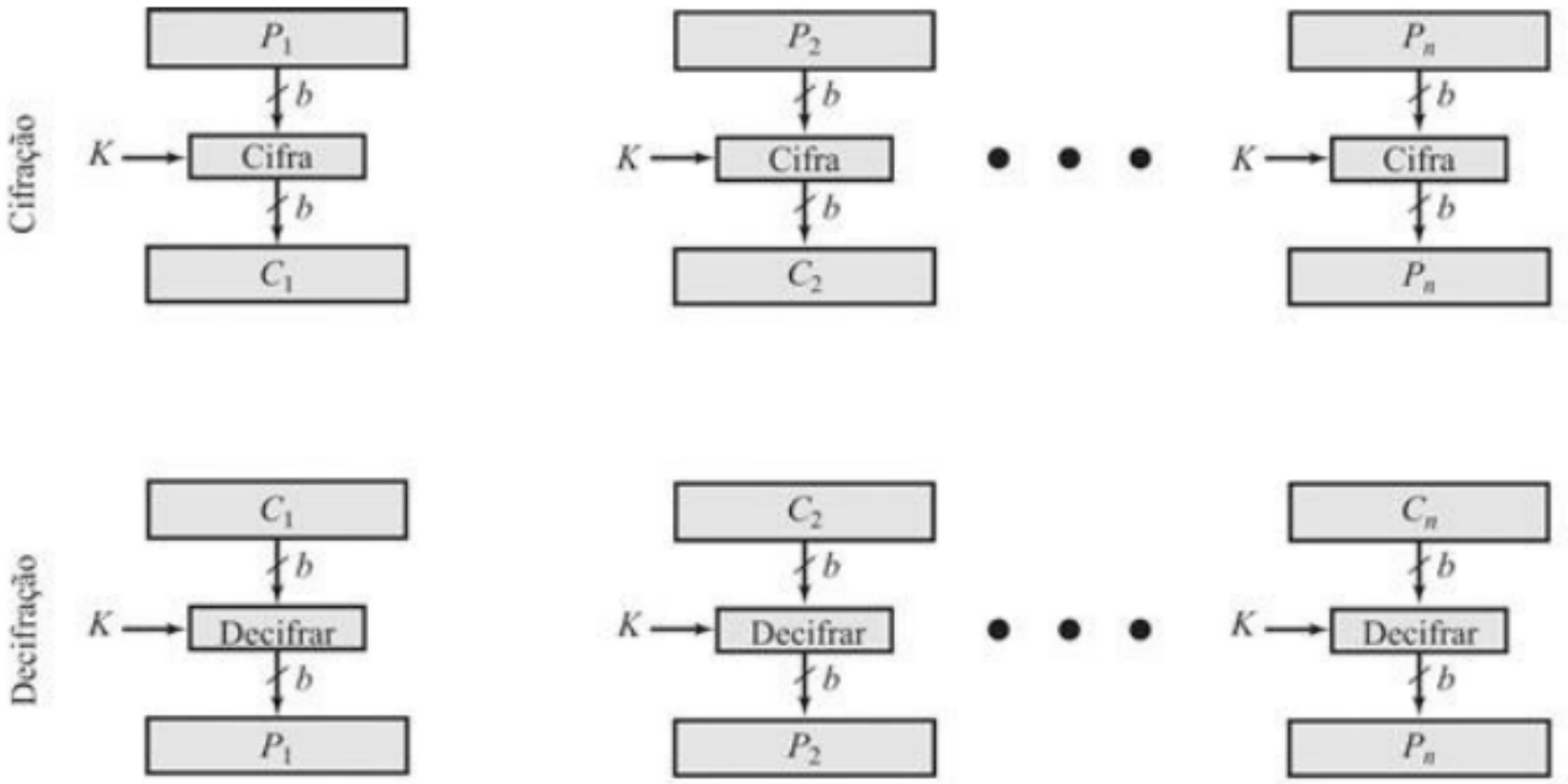
Section 2

Modos de Operação e Cifras de Bloco vs Fluxo

Questões práticas de segurança

- Normalmente, a cifração simétrica é aplicada a uma unidade de dados maior do que um bloco único de 64 bits ou 128 bits
- Mensagens de e-mail, pacotes de rede, registros de bancos de dados e outras fontes de texto às claras devem ser partidos em uma série de blocos de comprimento fixo para a cifração por uma cifra de bloco simétrica
- A abordagem mais simples para a cifração de múltiplos blocos é conhecida como modo de livro código eletrônico (**electronic codebook – ECB**)
  - texto às claras é processado b bits por vez e cada bloco de texto às claras é cifrado usando a mesma chave (normalmente b=64 ou b=128)
- Para mensagens longas, o modo ECB pode não ser seguro
  - Um criptoanalista pode conseguir explorar regularidades no texto às claras para facilitar a tarefa de decifração.
  - Por exemplo, se é sabido que a mensagem sempre começa com certos campos predefinidos, então o criptoanalista pode ter vários pares de texto às claras-texto cifrado conhecidos com os quais trabalhar.
- Utilização de **modos de operação** e também de **cifras de fluxo**

Ilustração da Cifração de Bloco



(a) Cifração por cifra de bloco (modo de livro código eletrônico)

Figure 1: Retirado do livro-texto

Ilustração da Cifração de Fluxo

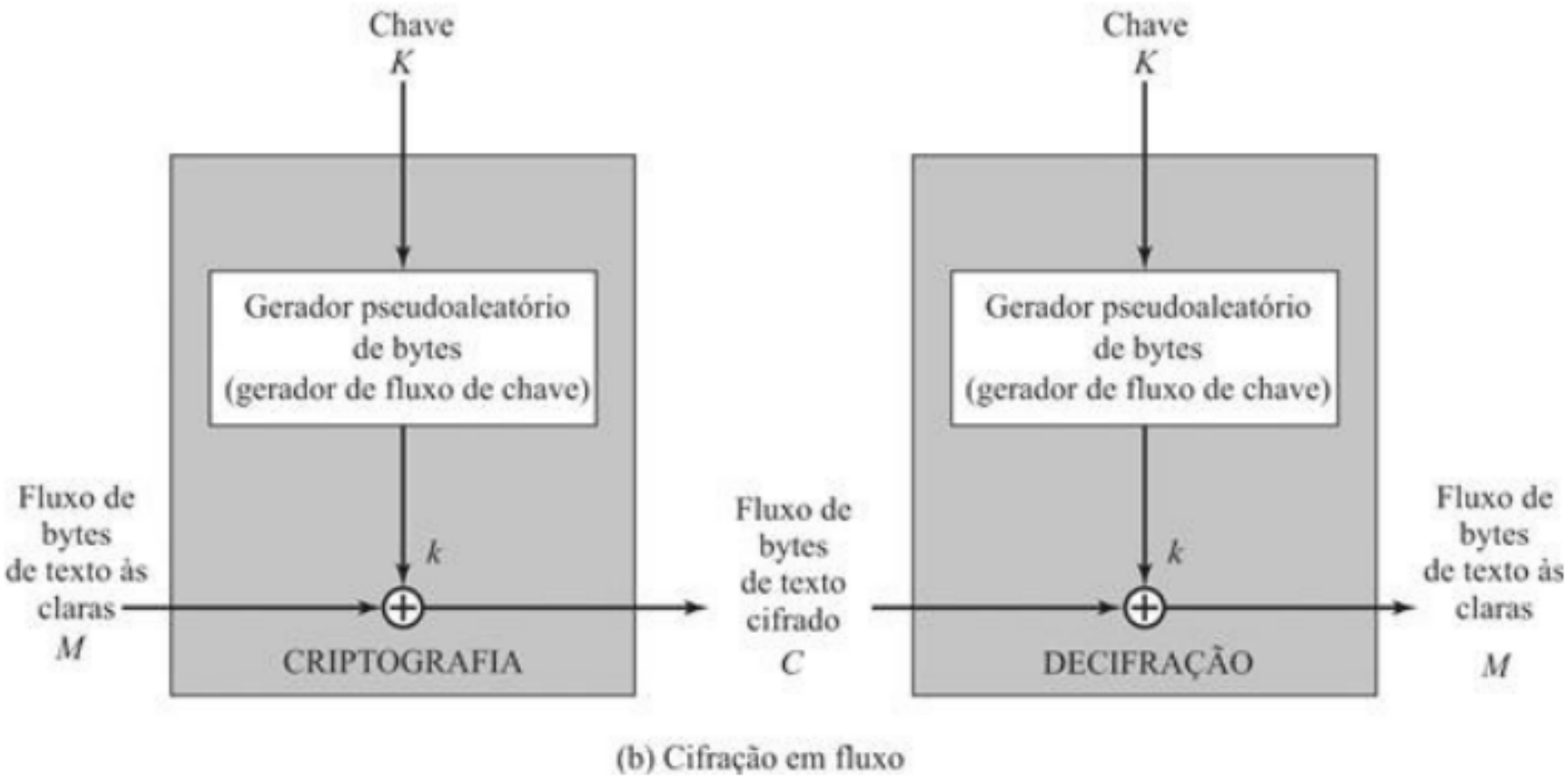


Figure 2: Retirado do livro-texto

CIFRAS DE FLUXO

- Processa os elementos de entrada continuamente
- Chave é entrada para um gerador de bits pseudoaleatório
  - Produz fluxo de números pseudoaleatórios (sequência de cifração)
  - Imprevisível sem conhecer a chave de entrada
  - XOR com bytes de texto às claras
- São mais rápidos e usam muito menos código
- Considerações de projeto:
  - Sequência de cifração deve ter um grande período
  - Deve ter propriedades próximas a de números aleatórios
  - Usar uma chave suficientemente longa
- Uma cifra de fluxo muito utilizada é o RC4 (Rivest Cipher 4) de 1987, *vazado* em 1994: https://en.wikipedia.org/wiki/RC4
- Bastante simples, mas considerada insegura (usada em protocolos inseguros como WEP)

RC4 vs cifras de bloco

Cifra	Comprimento da chave	Velocidade (Mbps)
DES	56	21
3DES	168	10
AES	128	61
RC4	Variável	113

Figure 3: Slides Kowada

MODOS DE OPERAÇÃO

- Em inglês, Block Cipher Mode Operation:
  - https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation
- Cifras de bloco processam dados em blocos
  - Por exemplo, 64 bits (DES, 3DES) ou 128 bits (AES)
- Para mensagens mais longas deve quebrar em blocos
  - E, possivelmente, o completar final do bloco para múltiplo (*padding*)
- Modos mais antigos: ECB, CBC, OFB, and CFB (FIPS 81 de 1981)
- 5 cinco modos de operação: NIST SP 800-38A de 2001
  - Livro-código eletrônico (ECB)
  - Encadeamento de blocos de cifra (CBC)
  - Realimentação de cifra (CFB)
  - Realimentação de saída (OFB)
  - *Contador (CTR)* - de 2001
- Modo XTS-AES adicionado na SP800-38E de 2010
- Modo PCBC também frequentemente usado (sem norma)
- Os modos pretendem cobrir praticamente todas as possíveis aplicações de cifração para as quais uma cifra de bloco poderia ser usada

Vetor de Inicialização

- Alguns modos exigem uma sequência binária única chamada de Vetor de Inicialização, ou *Initialization Vector* (IV)
  - Não deve se repetir, e para alguns modos, deve ser aleatório
- Utilizado para garantir que *plaintexts* repetidos gerem *ciphertexts* distintos, mesmo que a mesma chave seja utilizada
- Também chamado de *Starting Variable* (SV)
- O IV não precisa ser secreto, como uma chave privada
- Observar regras nas normas como SP800-38A
  - Modo CBC e CFB: reutilizar IV vaza informações no primeiro bloco de plaintext
  - Modo OFB e CTR: reutilizar IV quebra a segurança
  - Em CBC, o IV deve ser imprevisível

Modos: visão geral (seis modos incluindo PCBC)

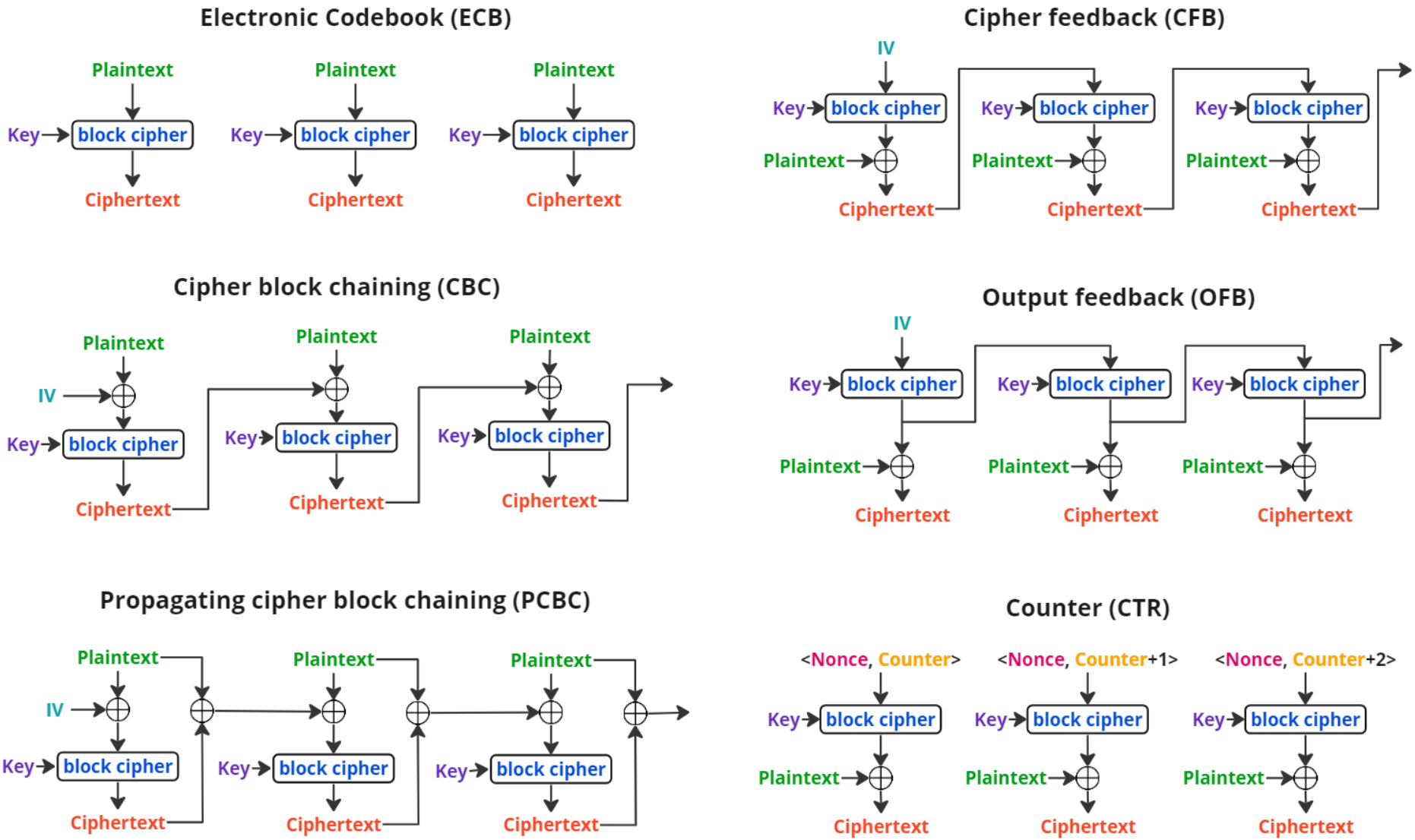


Figure 4: Wikipedia

Modo: ELECTRONIC CODEBOOK (ECB)

- Modo mais simples
- Dividir texto às claras em blocos de b bits cada
- Cifrar cada bloco usando a **mesma chave**
- "Codebook" (livro-código) porque tem valor de texto cifrado exclusivo para cada bloco de texto sem formatação
- Não é seguro para mensagens longas:
  - texto às claras **repetido** é visto no texto cifrado **também repetido**
  - um criptoanalista possivelmente conseguirá explorar essas regularidades
  - Por exemplo, sabendo que a mensagem sempre começa com certos campos predefinidos, o criptoanalista pode ter vários pares de textos às claras/textos cifrados conhecidos com os quais trabalhar
- Para superar as deficiências de segurança do ECB, seria bom se tivéssemos uma técnica na qual um mesmo bloco de texto às claras, caso repetido, produzisse blocos de texto cifrado diferentes
- Fraquezas conhecidas: veja no próximo slide a encriptação de bits de uma imagem

Ilustração Fraqueza ECB

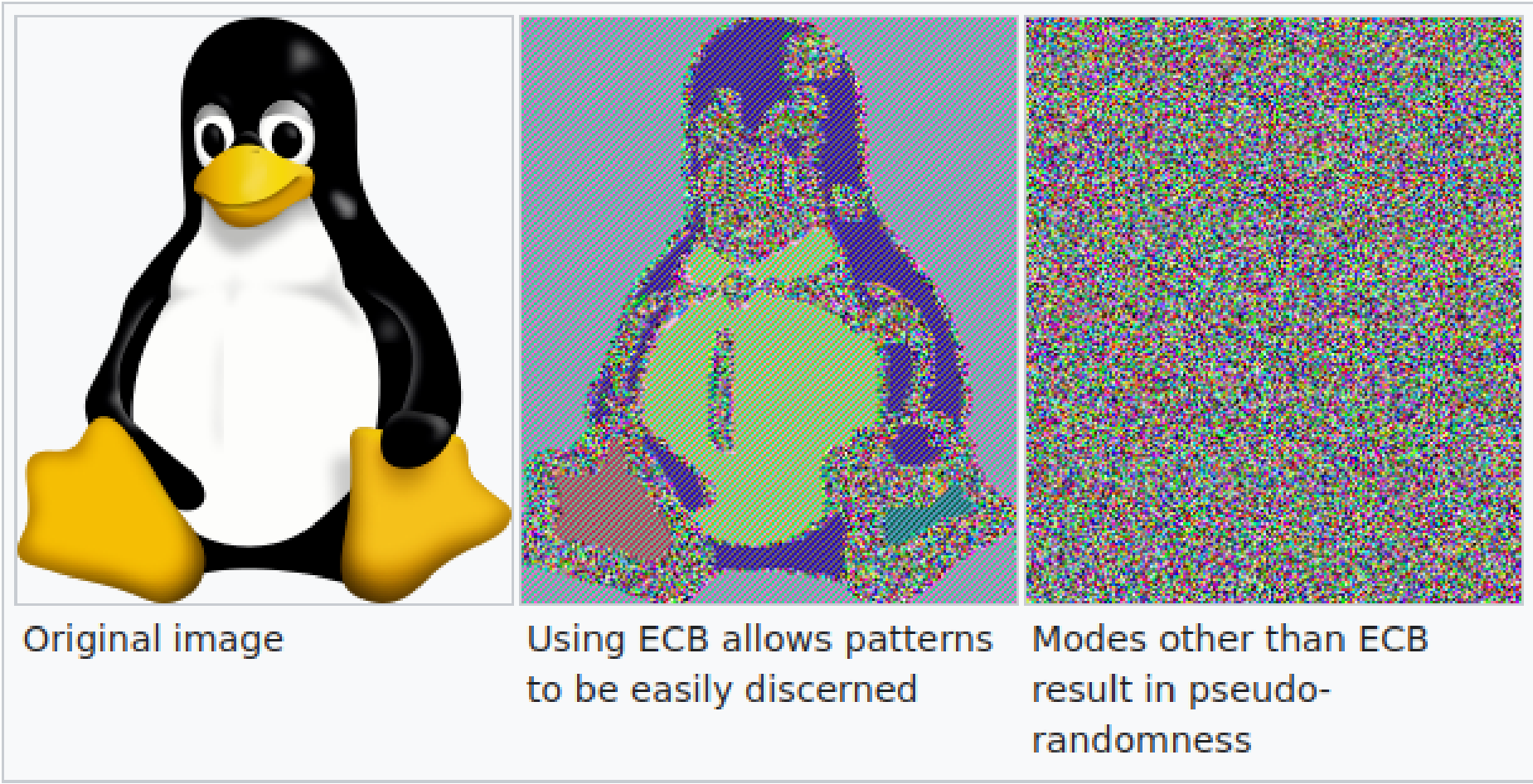
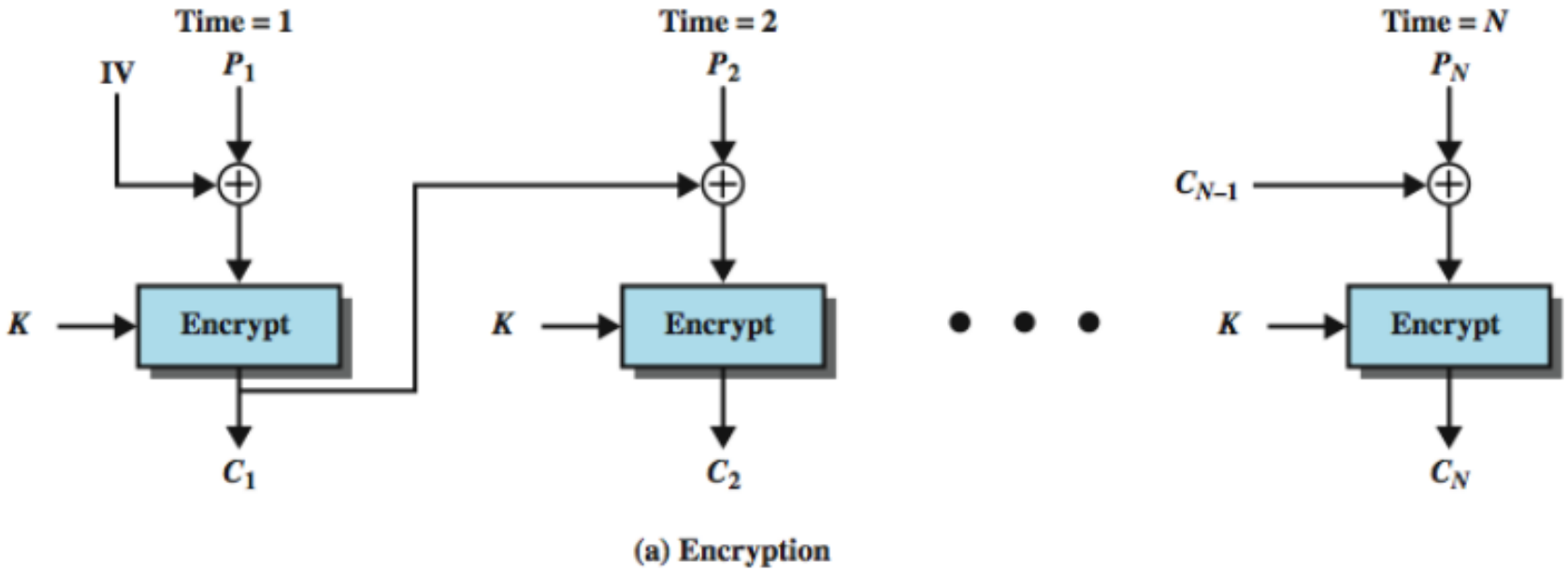


Figure 5: Fraqueza ECB - bits da imagem do Tux encriptados (wikipedia)

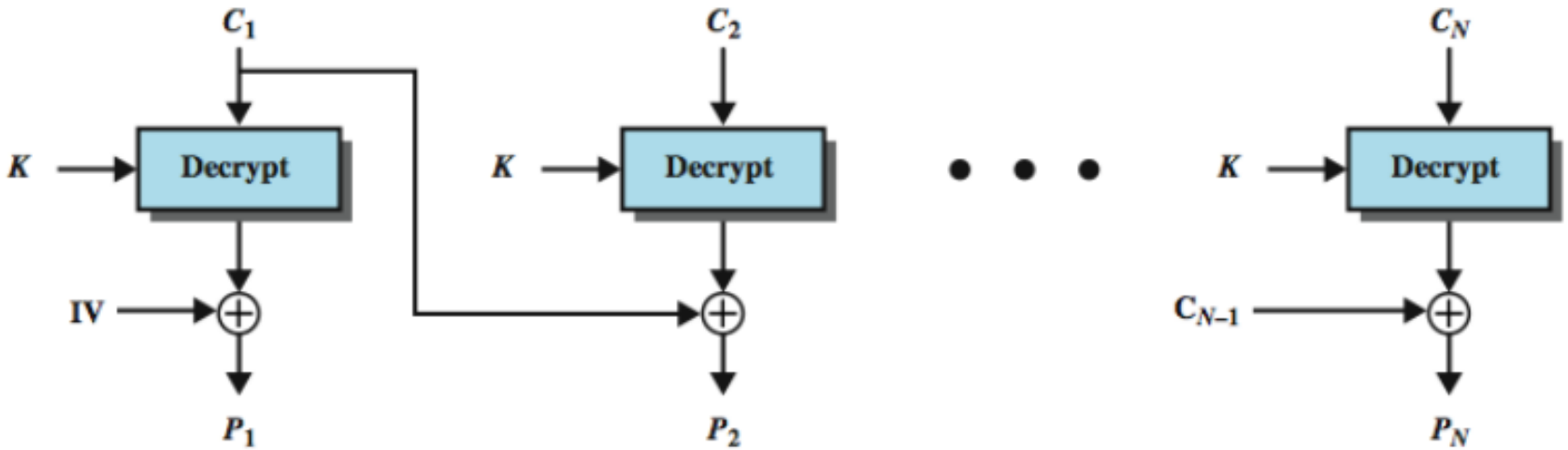
Modo: CIPHER BLOCK CHAINING (CBC)

- a entrada para o algoritmo de cifração é o resultado da operação de XOR entre o bloco de texto às claras e o bloco de texto cifrado precedente
- a mesma chave é usada para cada bloco
- encadeamos o processamento da sequência de blocos de texto às claras
- a entrada passada para a função de cifração para cada bloco de texto às claras não guarda qualquer relação fixa com o bloco de texto às claras
- padrões repetitivos de b bits não são expostos
- para fazer a decifração, cada bloco cifrado é passado pelo algoritmo de decifração
- O resultado passa por uma operação de XOR com o bloco de texto cifrado precedente para produzir o bloco de texto às claras

Ilustração CBC



(a) Encryption

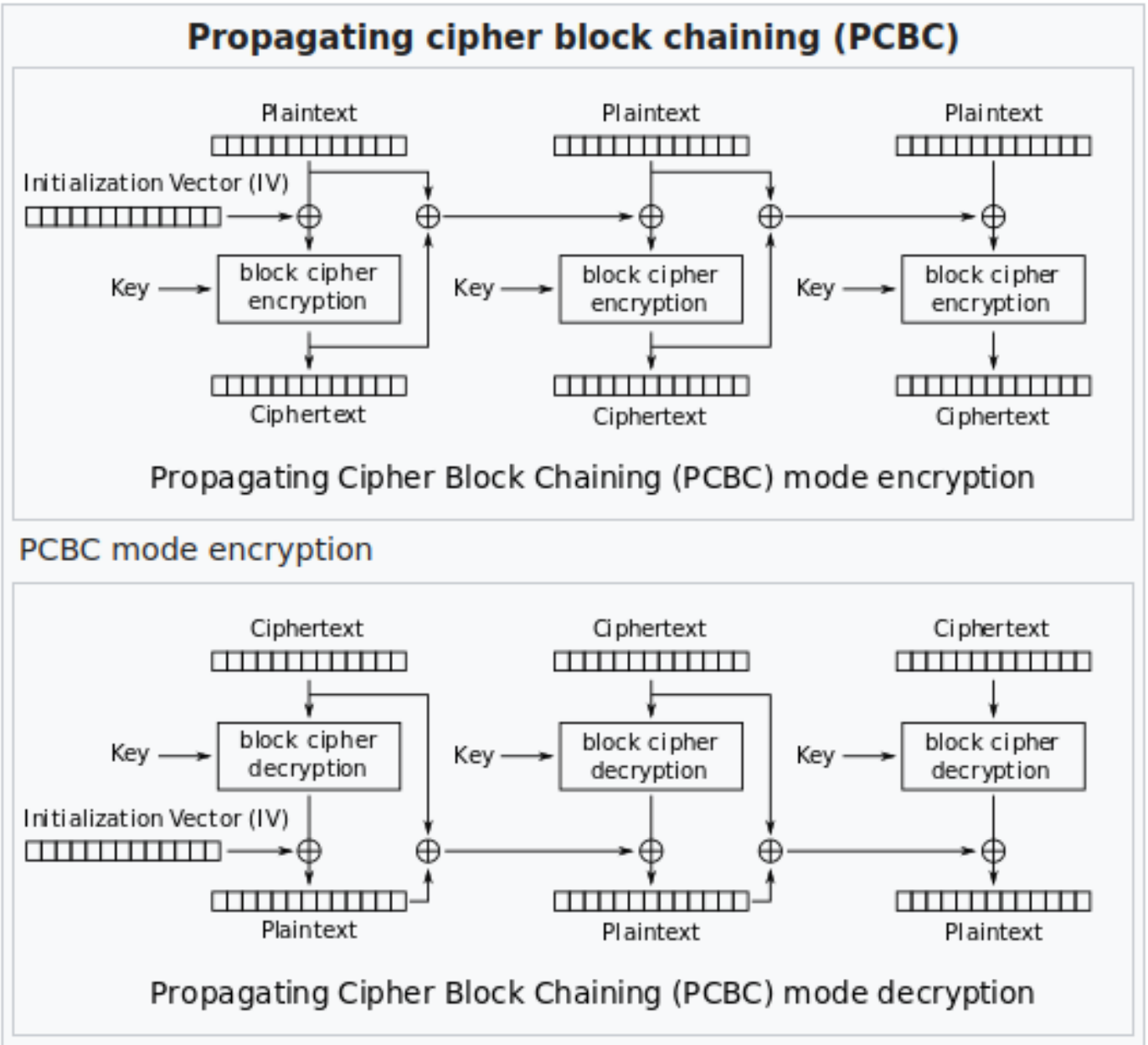


(b) Decryption

Modo: PCBC

- Alteração do IV no CBC somente corrompe o primeiro bloco, dado que ele permite decifração em paralelo (mas não a cifração em paralelo)
- Possibilidade no CBC de ataques tipo *padding oracle attacks*, como POODLE (no SSL 3.0)
  - https://en.wikipedia.org/wiki/POODLE
  - Solução: desabilitar SSL 3.0 (descoberto em 2014)
- Uso de PCBC mitiga esse tipo de ataque, propagando indefinidamente mudanças no plaintext
  - Efeito colateral: inviabiliza decifração paralela

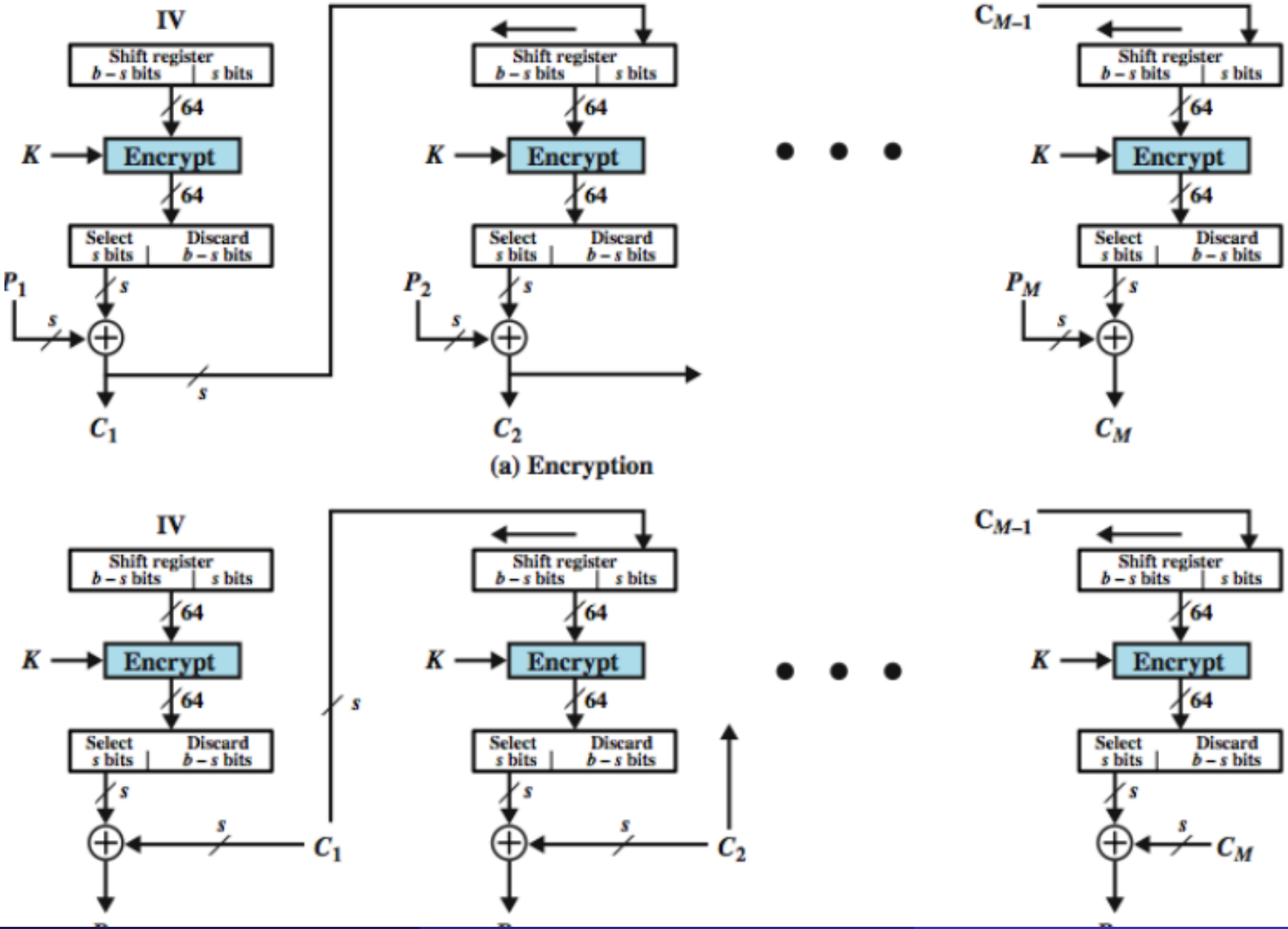
Ilustração PCBC



Modo: Realimentação de Cifra (CFB)

- É possível converter qualquer cifra de bloco (estilo CBC) em uma cifra de fluxo usando o modo de realimentação de cifra
- Uma cifra de fluxo elimina a necessidade de preenchimento de uma mensagem para que ela tenha um número inteiro de blocos
- Cifras de fluxo também podem operar em tempo real
- Assim, se um fluxo de caracteres está sendo transmitido, cada caractere pode ser cifrado e transmitido imediatamente usando uma cifra de fluxo orientada a caracteres
- Uma propriedade desejável de uma cifra de fluxo é que o texto cifrado seja do mesmo comprimento que o texto às claras
- Assim, se caracteres de 8 bits estão sendo transmitidos, cada caractere deve ser cifrado usando 8 bits
  - Se forem usados mais de 8 bits, desperdiça-se capacidade de transmissão.
- A entrada para a função de cifração é um registrador de deslocamento de b bits que inicialmente é ajustado para algum vetor de inicialização (IV)

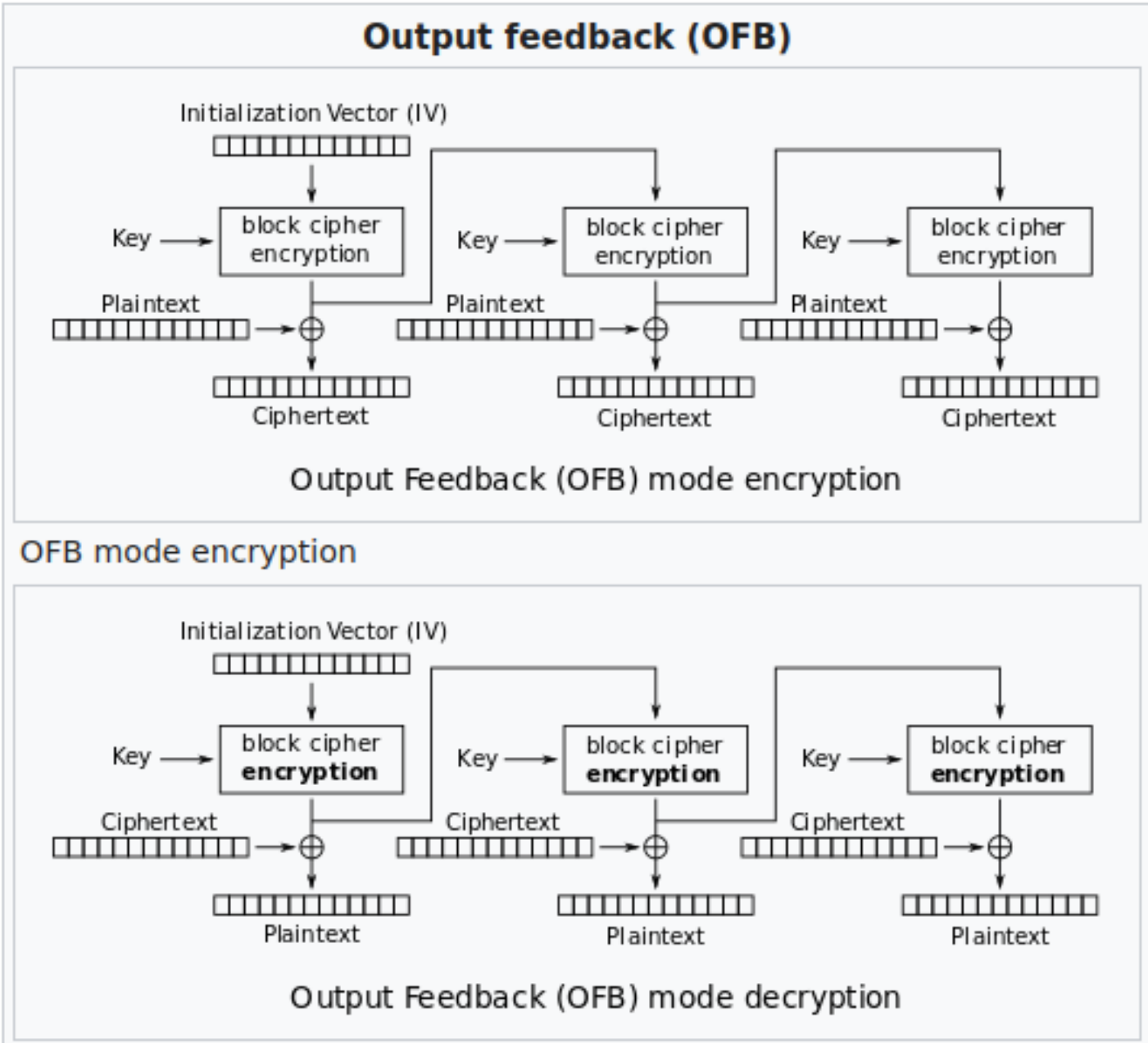
Ilustração CFB



Modo: Output Feedback (OFB)

- Funciona de forma idêntica na encriptação e decriptação
- Torna *síncrono* o processo de fluxo (inviabiliza paralelização)
- Permite que códigos de correção de erro funcionem normalmente, com o atraso da introdução do plaintext na execução do modo
- Possível encriptar antecipadamente e somente então aplicar XOR no dado plaintext, permitindo algum nível de paralelização

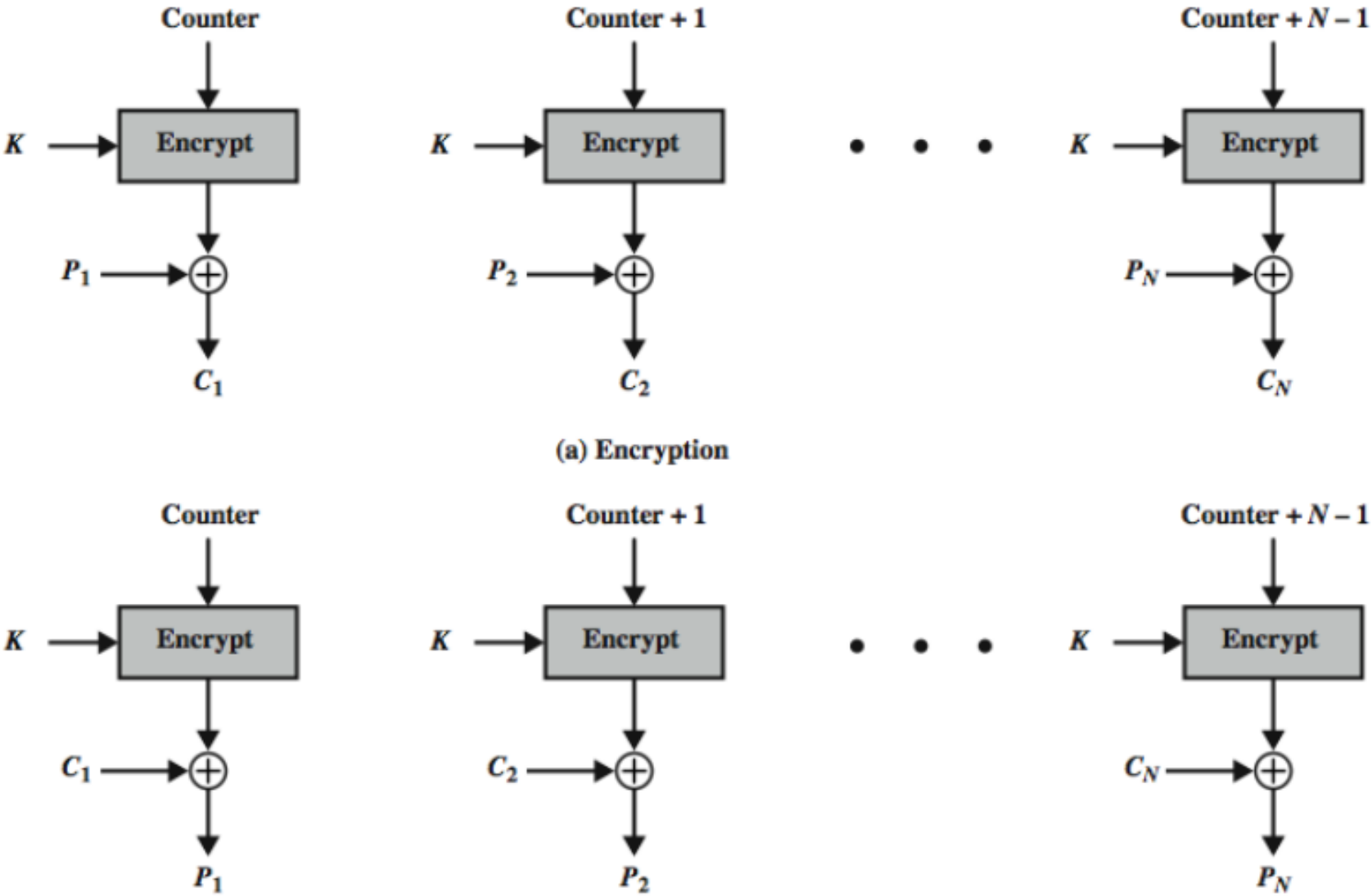
Ilustração OFB



Modo: Contador/*Counter* (CTR)

- Introduzido em 1979 por Whitfield Diffie e Martin Hellman
- Um contador igual ao tamanho do bloco de texto às claras é usado
- O único requisito declarado em SP 800-38A é que o valor do contador deve ser diferente para cada bloco de texto às claras que é cifrado
- Tipicamente, o contador é inicializado com algum valor e então incrementado de 1 para cada bloco subsequente (módulo 2^b, onde b é o tamanho do bloco)
- Para a cifração, o contador é cifrado e então passa por uma operação de XOR com o bloco de texto às claras para produzir o bloco de texto cifrado; não há encadeamento
- Para a decifração, a mesma sequência de valores de contador é usada, sendo que cada contador cifrado passa por uma operação de XOR com um bloco de texto cifrado para recuperar o bloco de texto às claras correspondente.

Ilustração CTR



Outros Modos

- Novos modos adicionados após publicação do livro
  - Busque em sites com atualização mais frequente!
- Leia mais em:
  https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation

Section 3

Aleatoriedade

Números aleatórios e pseudoaleatórios

- Números aleatórios desempenham importante papel na utilização de criptografia para várias aplicações de segurança de rede
- Geração de chaves para o algoritmo criptográfico
- Geração de um fluxo de chaves para uma cifra de fluxo simétrica
- Geração de uma chave simétrica para uso como chave de sessão temporária ou na criação de um envelope digital (a ver)
- Essas aplicações dão origem a dois requisitos distintos e não necessariamente compatíveis para uma sequência de números aleatórios: **aleatoriedade** e **imprevisibilidade**

Aleatoriedade

- Tradicionalmente, a preocupação na geração de uma sequência de números alegadamente aleatórios é que a sequência de números seja aleatória em algum sentido estatístico bem definido.
- Dois critérios são usados para validar que uma sequência de números é aleatória: **distribuição uniforme** e **independência**
- No contexto da nossa discussão, a utilização de uma sequência de números que *parecem estatisticamente aleatórios* ocorre frequentemente no projeto de algoritmos relacionados à criptografia

Distribuição uniforme

A distribuição de números na sequência deve ser uniforme, isto é, a frequência de ocorrência de cada um dos números deve ser aproximadamente a mesma.

Independência

Nenhum valor na sequência pode ser inferido dos outros.

Desafios Práticos de Aleatoriedade

- Existem testes bem definidos para determinar se uma sequência de números corresponde a uma distribuição particular, como a distribuição uniforme, mas não há teste para "provar" independência
  - Em vez disso, diversos testes podem ser aplicados para demonstrar se a sequência **não** exibe independência
  - A estratégia geral é aplicar vários desses testes até que a confiança na existência da independência seja suficientemente forte
- Por exemplo, um requisito fundamental do esquema criptográfico de chave pública RSA é a capacidade de gerar números primos
  - Em geral, é difícil determinar se um número grande N dado é primo. Uma abordagem de força bruta seria dividir N por todo ímpar inteiro menor do que $\sqrt{N}$
  - Se N for da ordem de, digamos, N^{150}, ocorrência que não é incomum em criptografia de chave pública, tal abordagem de força bruta está além do alcance de analistas humanos e de seus computadores.
  - Se a sequência for longa o suficiente (mas muito, muito menor que $\sqrt{N^{150}}$) a primalidade de um número pode ser determinada com *quase total certeza*

Aleatório versus pseudoaleatório

- Em aplicações como autenticação mútua e geração de chaves de sessão, o requisito não é tanto que a sequência de números seja estatisticamente aleatória, mas que os membros sucessivos da sequência sejam **imprevisíveis**
- Com sequências "verdadeiramente" aleatórias, cada número é estatisticamente independente de outros números na sequência e, por conseguinte, imprevisível
- números aleatórios verdadeiros nem sempre são usados; em vez disso, sequências de números que parecem ser aleatórias são geradas por algum algoritmo
  - Neste último caso, deve-se tomar cuidado para que um oponente não consiga prever elementos futuros da sequência com base em elementos anteriores
- algoritmos determinísticos produzem sequências de números que não são estatisticamente aleatórios
  - Todavia, se o algoritmo for bom, as sequências resultantes passarão em muitos testes razoáveis de aleatoriedade
- Tais números são denominados números **pseudoaleatórios**

Filosofando sobre números pseudoaleatórios

De acordo com HAMM91:

> *Para finalidades práticas somos forçados a aceitar o conceito esquisito de "relativamente aleatórios" significando que, com relação ao uso proposto, não podemos ver qualquer razão por que eles não funcionariam como se fossem aleatórios (como a teoria usualmente requer). Isso é altamente subjetivo e não muito palatável para os puristas, mas é um argumento ao qual os estatísticos regularmente apelam quando tomam "uma amostra aleatória" — eles esperam que quaisquer resultados que usarem terão aproximadamente as mesmas propriedades de uma contagem completa do espaço total da amostra que ocorre em sua teoria.*

Números verdadeiramente aleatórios

- Um gerador de números verdadeiramente aleatórios (*True Random Number Generator — TRNG*) usa uma fonte não determinística para produzir aleatoriedade
- A maioria opera medindo processos naturais imprevisíveis, como detectores de pulso de eventos de radiação ionizantes, tubos de descarga de gás e capacitores com fuga de fluxo
- A Intel desenvolveu um chip disponível comercialmente que toma amostras de ruído térmico amplificando a voltagem medida entre resistores não acionados
- Um grupo do Bell Labs desenvolveu uma técnica que usa as variações no tempo de resposta de solicitações de leitura de um setor de disco de um disco rígido
- LavaRnd é um projeto de código-fonte aberto para criar números verdadeiramente aleatórios usando câmeras baratas, código-fonte aberto e hardware barato. O sistema usa um dispositivo de carga acoplada saturado (CDD) dentro de uma lata hermeticamente fechada (à prova de luz) como fonte caótica para produzir aleatoriedade

Section 4

Discussão

Breve discussão

Cenário atual

- Pense em um problema prático que envolve cifração
- Quais técnicas são melhores para uso prático? Quais vantagens e desvantagens?

Leia mais

Livro:

- "Segurança de Computadores - Princípios e Práticas - 2012" - Stallings, William; Brown, Lawrie & Lawrie Brown & Mick Bauer & Michael Howard
  - Em Português do Brasil, CAMPUS - GRUPO ELSEVIER, 2ª Ed. 2014

Veja Capítulo 7, todas seções e finaliza o capítulo 7.

Section 5

Agradecimentos

Pessoas

Em especial, agradeço aos colegas que elaboraram bons materiais, como o prof. Raphael Machado, Kowada e Viterbo cujos conceitos formam o cerne desses slides.

Estendo os agradecimentos aos demais colegas que colaboraram com a elaboração do material do curso de Pesquisa Operacional, que abriu caminho para verificação prática dessa tecnologia de slides.

Software

Esse material de curso só é possível graças aos inúmeros projetos de código-aberto que são necessários a ele, incluindo:

- pandoc
- LaTeX
- GNU/Linux
- git
- markdown-preview-enhanced (github)
- visual studio code
- atom
- revealjs
- groomit-mpx (screen drawing tool)
- xournal (screen drawing tool)
- ...

Empresas

Agradecimento especial a empresas que suportam projetos livres envolvidos nesse curso:

- github
- gitlab
- microsoft
- google
- ...

Reprodução do material

Esses slides foram escritos utilizando pandoc, segundo o tutorial ilectures:

- https://igormcoelho.github.io/ilectures-pandoc/

Exceto expressamente mencionado (com as devidas ressalvas ao material cedido por colegas), a licença será Creative Commons.

Igor Machado Coelho

This Slide Is Intentionally Blank (for goomit-mpx)