CONTROLE DE ACESSO E FUNÇÕES DE HASH

- Controle de Acesso.
- Uso de Senhas.
- Funções de Hash, salting, HMAC.

CONTROLE DE ACESSO – TIPOS DE CONTROLE DE ACESSO

Controles de acesso físico - barreiras reais implantadas para evitar o contato direto com sistemas. A meta é prevenir que usuários não autorizados acessem fisicamente as instalações, equipamentos e outros ativos organizacionais. O controle de acesso físico determina quem pode entrar (ou sair), onde podem entrar (ou sair) e quando podem entrar (ou sair).

Controles de acesso lógicos - soluções de hardware e software usadas para gerenciar o acesso aos recursos e sistemas. Essas soluções baseadas em tecnologia incluem ferramentas e protocolos que os sistemas de computador usam para identificação, autenticação, autorização e responsabilidade.

Controles de acesso administrativos - políticas e procedimentos definidos pelas empresas para implementar e aplicar todos os aspectos do controle de acesso não autorizado. Os controles administrativos focam em práticas pessoais e de negócios.

CONTROLE DE ACESSO – IDENTIFICAÇÃO

A identificação aplica as regras estabelecidas pela política de autorização:

- Um indivíduo solicita acesso a um recurso do sistema.
- Sempre que o indivíduo solicita acesso a um recurso, os controles de acesso determinam se devem conceder ou negar o acesso.
- As políticas de segurança cibernética determinam quais controles de identificação devem ser usados.
- A confidencialidade das informações e os sistemas de informações determinam o nível de exigência dos controles.
- O aumento nas violações de dados forçou muitas empresas a reforçar os controles de identificação.

CONTROLE DE ACESSO – MÉTODOS DE AUTENTICAÇÃO

- **O que você sabe** - senhas, frases secretas ou PINs são exemplos de algo que o usuário sabe. As senhas são o método mais popular usado para autenticação.
- **O que você tem** - cartões inteligentes, tokens são exemplos de algo que os usuários têm.
- **Quem você é** - uma característica física única, como uma impressão digital, retina ou voz, que identifica um usuário específico, é chamada de biometria.
- **Autenticação multifator** - usa pelo menos dois métodos de verificação. Uma chave de segurança é um bom exemplo. Os dois fatores são algo que você sabe, como uma senha, e algo que você tem, como uma chave de segurança.

CONTROLE DE ACESSO – AUTORIZAÇÃO

A autorização controla o que um usuário pode ou não fazer na rede após a autenticação:

- Depois que um usuário revelar sua identidade, o sistema verifica quais recursos da rede o usuário pode acessar e o que os usuários podem fazer com os recursos.
- A autorização usa um conjunto de atributos que descrevem o acesso do usuário à rede.
- O sistema compara esses atributos às informações contidas no banco de dados de autenticação, determina um conjunto de restrições para o usuário e o entrega no roteador local em que o usuário está conectado.
- Definir as regras de autorização é a primeira etapa no controle de acesso. Uma política de autorização estabelece essas regras.

CONTROLE DE ACESSO – AUDITORIA

A auditoria rastreia uma ação até a pessoa ou processo que está efetuando a mudança em um sistema, coleta essas informações e reporta os dados de uso.

- A empresa pode usar esses dados para determinadas finalidades, como auditoria ou cobrança.
- Os dados coletados podem incluir a hora de login para um usuário, independentemente de o login de usuário ter sido bem ou malsucedido ou quais recursos de rede o usuário acessou.
- Isso permite que a empresa rastreie as ações e os erros durante uma auditoria ou investigação.
- A implementação da auditabilidade consiste em tecnologias, políticas, procedimentos e educação.
- Os arquivos de log fornecem informações detalhadas com base nos parâmetros escolhidos.

SENHAS

- Gerador de Senhas Fortes.
  - https://www.security.org/how-secure-is-my-password
  - https://strongpasswordgenerator.com
  - https://passwordsgenerator.net
  - https://lastpass.com/pt/password-generator

FUNÇÕES DE HASH - O QUE É HASH?

- *hash*, também chamado de "digestor", é uma espécie de "assinatura" ou "impressão digital" que representa o conteúdo de um fluxo de dados. Com certa frequência os hashes são chamados de *checksum*. Um *hash* pode ser comparado com um selo de embalagem que indica clara e inequivocamente se a embalagem já foi aberta ou violada.
- *Hashes* não são cifragens, são digestos! As cifragens transformam os dados do texto claro num criptograma e vice-versa. *Hashes*, transformam os dados do texto (claro ou cifrado) num pequeno digesto, de **tamanho fixo**, numa **operação de mão única**.

O QUE É HASH?

- Os *hashes* produzem "selos de segurança" de comprimento fixo, não importa o comprimento do fluxo de dados ou do arquivo que representem. Qualquer alteração efetuada no arquivo, por mínima que seja, altera substancialmente o resultado hash. Isto ocorre porque, mesmo se apenas um dos bits do arquivo for alterado, muitos bits do resultado serão afetados. Este comportamento é conhecido como **efeito avalanche**.

O QUE É HASH?

- md5

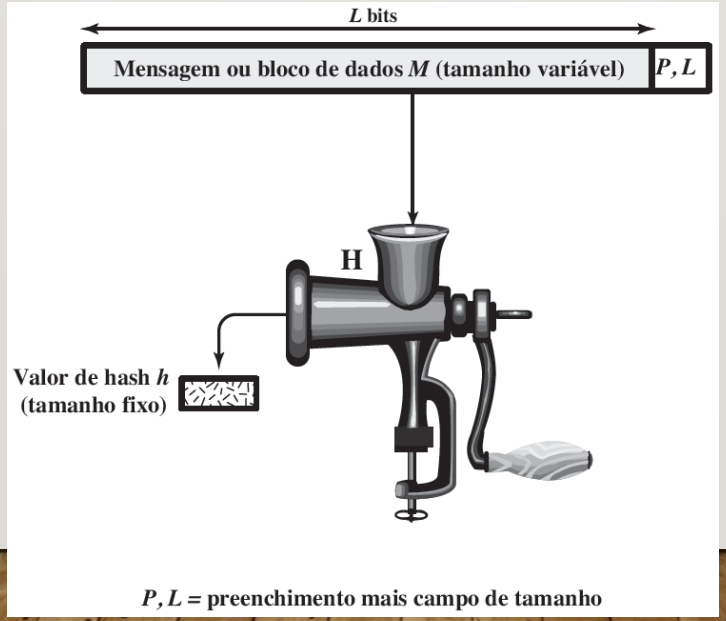
Caracter	ASCII (decimal)	ASCII (binário)
A	65	0100 0001
a	97	0110 0001

```
Aldeia NumaBoa     3cdb658425ee484e4bff3d4583f6f851
aldeia NumaBoa     9c1f4lef263026b0283676d63df21fdl
```

```
3cdb6584     0011 1100 1101 1011 0110 0101 1000 0100
9c1f4lef     1001 1100 0001 1111 0100 0001 1110 1111
             x.x. .... xx.. .x.. ..x. .x.. .xx. x.xx     12 bits diferentes

25ee484e     0010 0101 1110 1110 0100 1000 0100 1110
263026b0     0010 0110 0011 0000 0010 0110 1011 0000
             .... ..xx xx.x xxx. .xx. xxx. xxxx xxx.     20 bits diferentes

4bff3d45     0100 1011 1111 1111 0011 1101 0100 0101
283676d6     0010 1000 0011 0110 0111 0110 1101 0110
             .xx. ..xx xx.. x..x .x.. x.xx x..x ..xx     16 bits diferentes

83f6f851     1000 0011 1111 0110 1111 1000 0101 0001
3df21fdl     0011 1101 1111 0010 0001 1111 1101 0001
             x.xx xxx. .... .x.. xxx. .xxx x... ....     14 bits diferentes
```

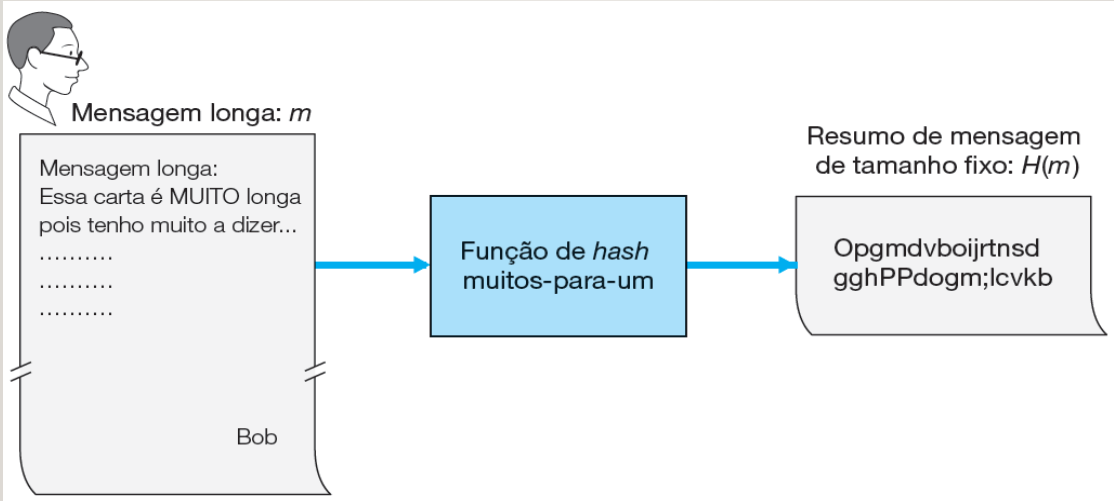
FUNÇÕES DE HASH CRIPTOGRÁFICAS

- Uma função de hash aceita uma mensagem de tamanho variável M como entrada e produz um valor de hash de tamanho fixo h = H(M).
- O tipo de função de hash necessária para aplicações de segurança é conhecido como função de hash criptográfica:



FUNÇÕES DE HASH CRIPTOGRÁFICAS

- Uma **função *hash* criptográfica** deve apresentar a seguinte propriedade adicional:
- Em termos de processamento, é impraticável encontrar duas mensagens diferentes x e y tais que $H(x) = H(y)$.

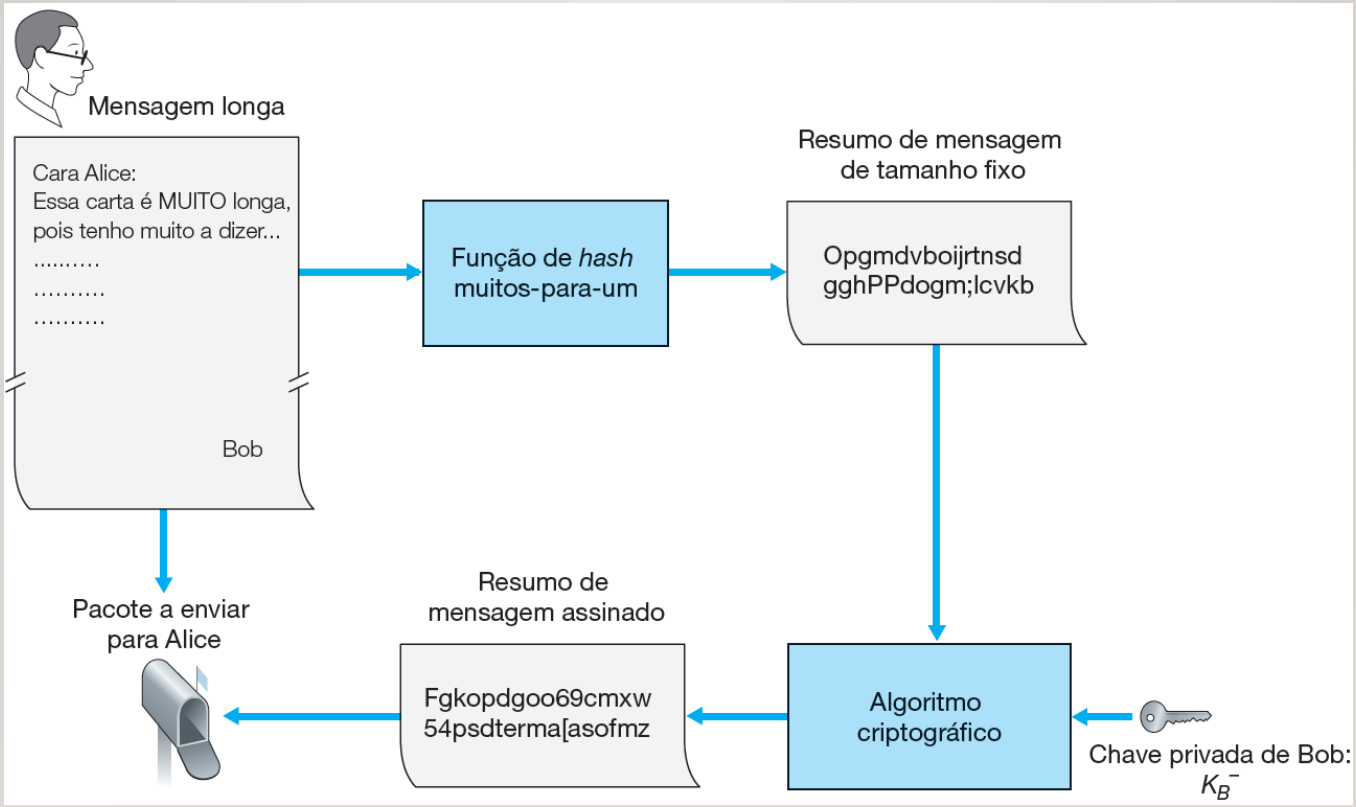


FUNÇÕES DE HASH CRIPTOGRÁFICAS

- Aplicações
  - Integridade de dados: qualquer tipo de arquivo, e um fluxo de dados que produz um resultado hash único. Uma das maneiras de poder verificar se o arquivo baixado é idêntico ao disponibilizado é conhecer o hash do arquivo original.
  - Segurança de senhas: armazenar os resultados hash das senhas do que as próprias senhas. O uso de uma senha pressupõe que um usuário a digite. Tendo a senha como entrada, é fácil e rápido calcular o resultado hash da senha fornecida e compará-lo com o valor arquivado.
  - Assinaturas digitais: Para se obter uma assinatura digital válida são necessárias duas etapas. A primeira é criar um hash do documento. Este hash identifica unicamente e inequivocamente o documento do qual ele se originou. A seguir, o assinante submete o hash a um método criptográfico usando sua chave privada. Como o hash criptografado só pode ser recuperado usando a chave pública do assinante, isto comprova a identidade da pessoa que assinou - é a chamada assinatura digital - e como o hash recuperado identifica o documento, a assinatura está associada unicamente a este documento.
  - Detecção de intrusão e detecção de vírus: armazena os arquivos e se houver alterações no sistema, são identificadas.

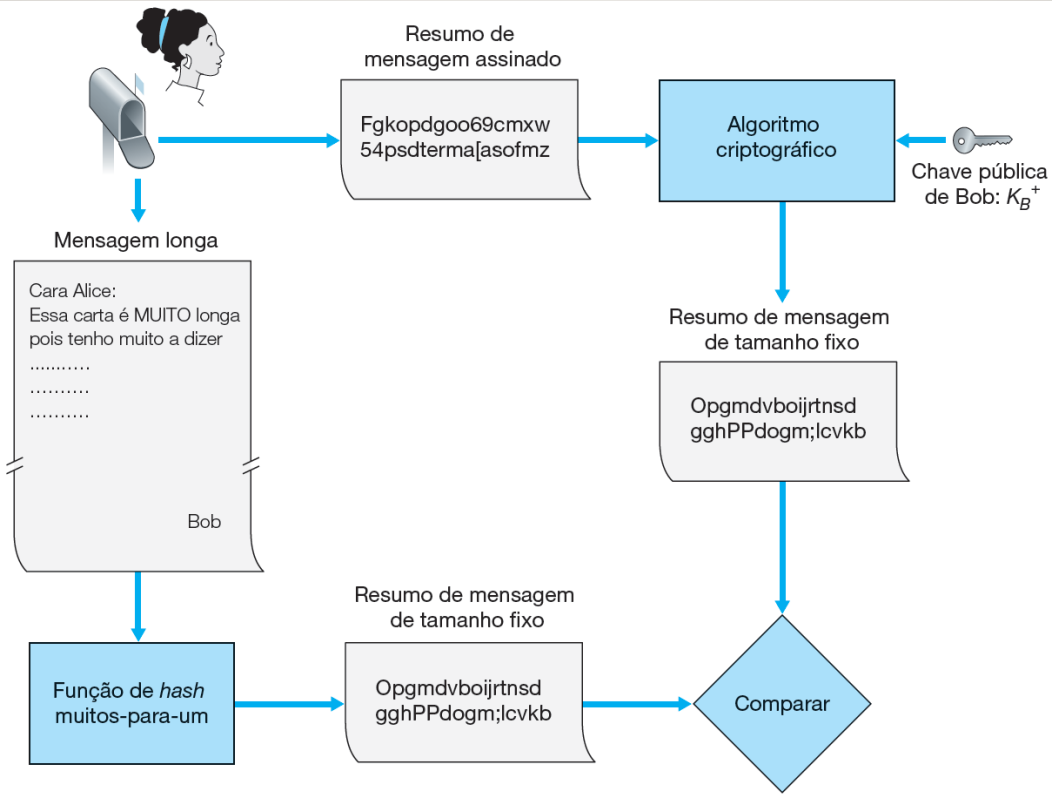
FUNÇÕES DE HASH CRIPTOGRÁFICAS

- Enviando uma mensagem assinada digitalmente



FUNÇÕES DE HASH CRIPTOGRÁFICAS

- Verificando uma mensagem assinada



REQUISITOS E SEGURANÇA

- Requisitos para função de hash criptográfica H:

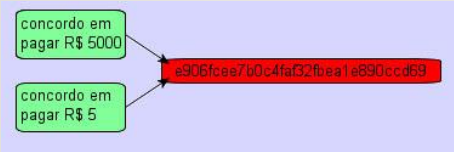
Requisito	Descrição
Tamanho de entrada variável	H pode ser aplicado em um bloco de dados de qualquer tamanho.
Tamanho da saída fixo	H produz uma saída de tamanho fixo.
Eficiência	$H(x)$ é relativamente fácil de calcular para qualquer valor de x informado, através de implementações tanto em hardware quanto em software.
Resistência à pré-imagem (propriedade de mão única)	Para qualquer valor de hash h informado, é computacionalmente impossível encontrar y, de modo que $H(y) = h$.
Resistência à segunda pré-imagem (resistência à colisão fraca)	Para qualquer bloco x informado, é computacionalmente impossível encontrar $y \neq x$ com $H(y) = H(x)$.
Resistência à colisão forte	É computacionalmente impossível encontrar qualquer par (x, y), de modo que $H(x) = H(y)$.
Pseudoaleatoriedade	A saída de H atende os testes padrão de pseudoaleatoriedade.

REQUISITOS E SEGURANÇA

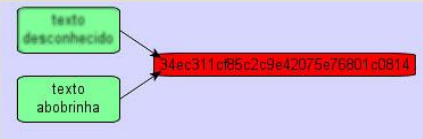
- Colisões
  - Usando métodos que produzam hashes de 128 bits, o número de hashes possíveis são 2^128 = 3,4 x 10^38 possíveis. Ou 340.282.366.920.938.463.463.374.607.431.768.211.456 hashes
  - Como o número de conjuntos de dados é praticamente infinito, a possibilidade de que dois conjuntos de dados diferentes produzam o mesmo hash não pode ser ignorada.

REQUISITOS E SEGURANÇA

- Resistência à colisões
  - Mede a dificuldade de encontrar duas entradas que produzam o mesmo resultado hash (procura de dois textos que produzam um hash qualquer)



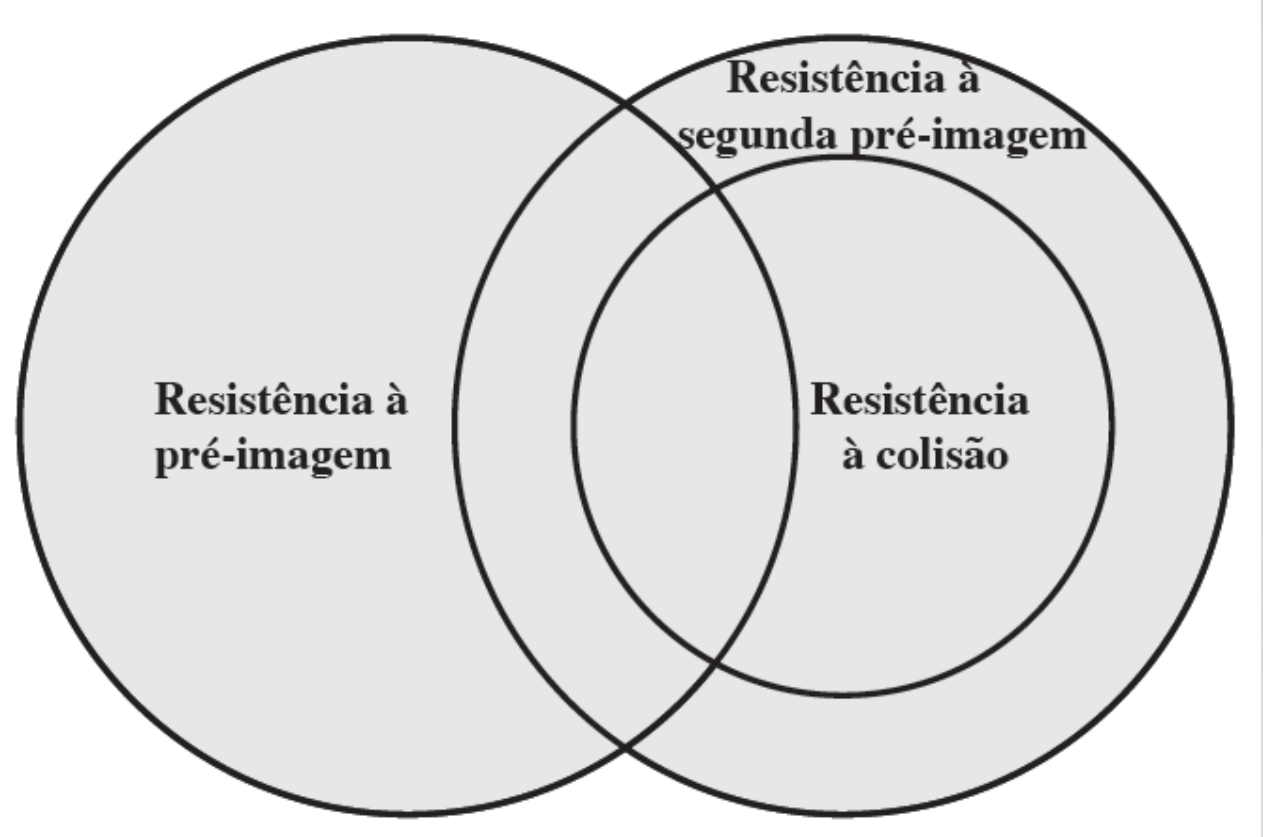
- Resistência de pré-imagem
  - Mede a dificuldade de criar um conjunto de dados que resulte num determinado valor hash, sem conhecer o texto que o originou (criação de determinado valor hash sem conhecer o texto original – hash de senhas)



- Resistência de segunda pré-imagem
  - Mede a dificuldade de criar um conjunto de dados que resulte num determinado valor hash, conhecendo o texto que o originou – arquivos de download.

REQUISITOS E SEGURANÇA

- Relação entre as propriedades das funções de hash:



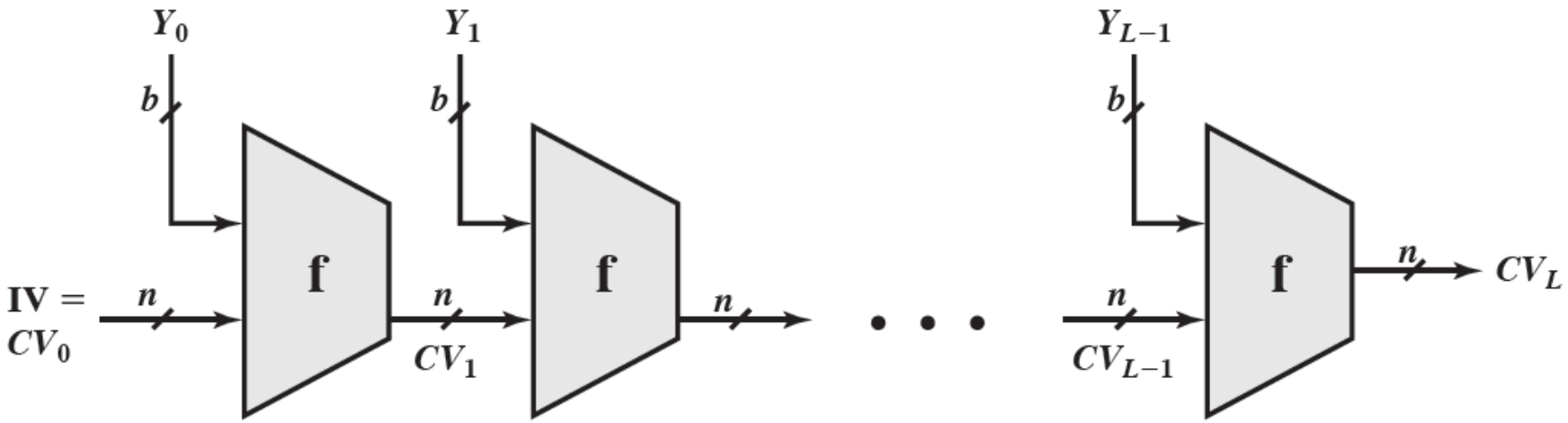
REQUISITOS E SEGURANÇA

- Propriedades de resistência necessárias para várias aplicações de integridade de dados:

	Resistência à pré-imagem	Resistência à segunda pré-imagem	Resistência à colisão
Hash + assinatura digital	sim	sim	sim*
Detecção de intrusão e detecção de vírus		sim	
Hash + encriptação simétrica			
Arquivo de senha de mão única	sim		
MAC	sim	sim	sim*

REQUISITOS E SEGURANÇA

- Estrutura geral do código de hash seguro:



IV = valor inicial
CV_i = variável de encadeamento
Y_i = i-ésimo bloco de entrada
f = algoritmo de compactação
L = Número de blocos de entrada
n = Tamanho do código de hash
b = Tamanho do bloco de entrada

CONTROLE DE INTEGRIDADE DE DADOS (SECURE HASH ALGORITHM - SHA)

- Comparação de parâmetros do SHA:

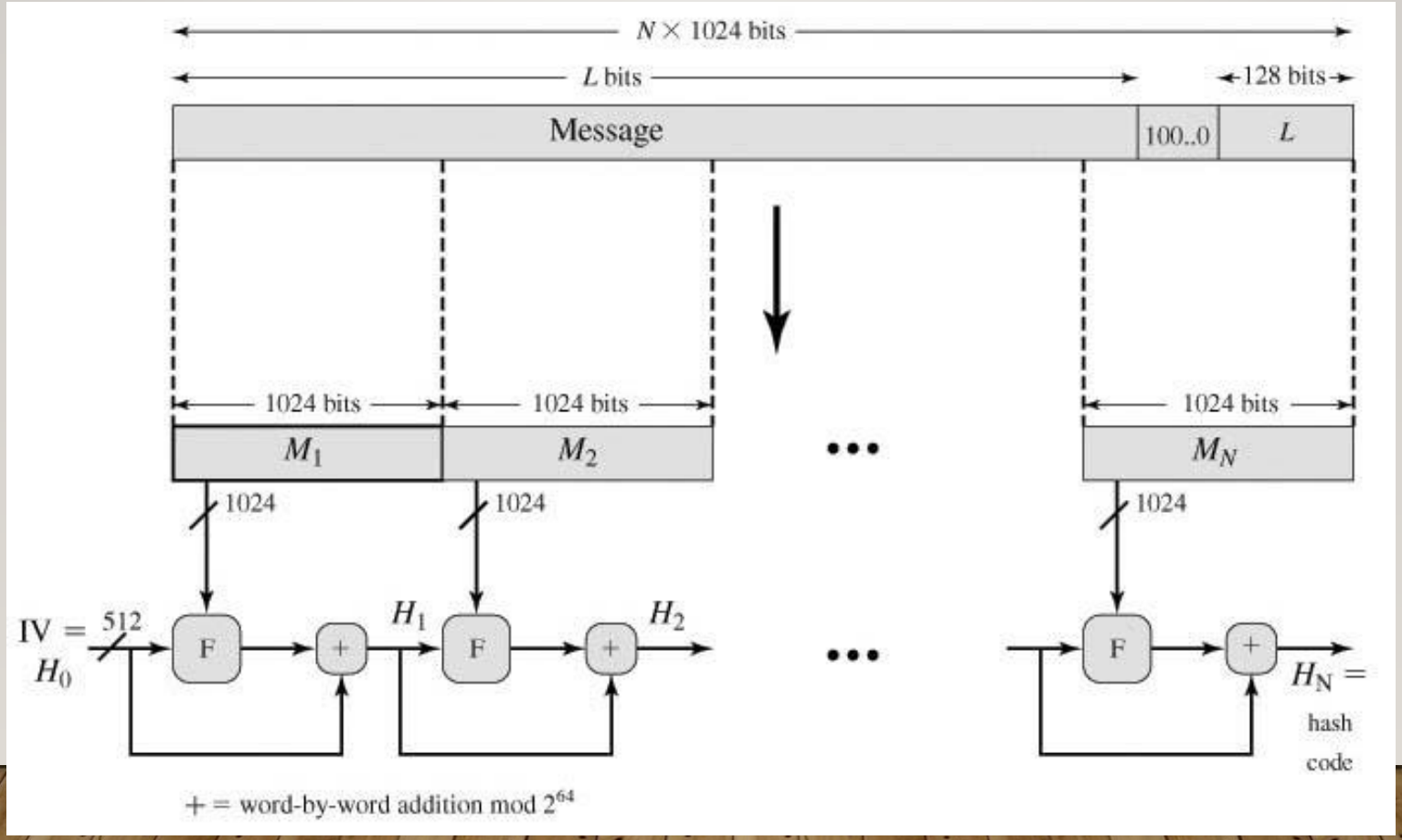
	SHA-1	SHA-224	SHA-256	SHA-384	SHA-512
Tamanho do resumo da mensagem	160	224	256	384	512
Tamanho da mensagem	$< 2^{64}$	$< 2^{64}$	$< 2^{64}$	$< 2^{128}$	$< 2^{128}$
Tamanho do bloco	512	512	512	1024	1024
Tamanho da word	32	32	32	64	64
Número de etapas	80	64	64	80	80

CONTROLE DE INTEGRIDADE DE DADOS (SECURE HASH ALGORITHM - SHA)

- Características para SHA-512:
  - Tamanho máximo para mensagem de entrada: <2^128
  - Produz na saída um digestor de mensagem de 512 bits
  - A entrada é processada em blocos de 1024 bits

CONTROLE DE INTEGRIDADE DE DADOS (SECURE HASH ALGORITHM - SHA)

- Geração de resumo da mensagem usando SHA-512:

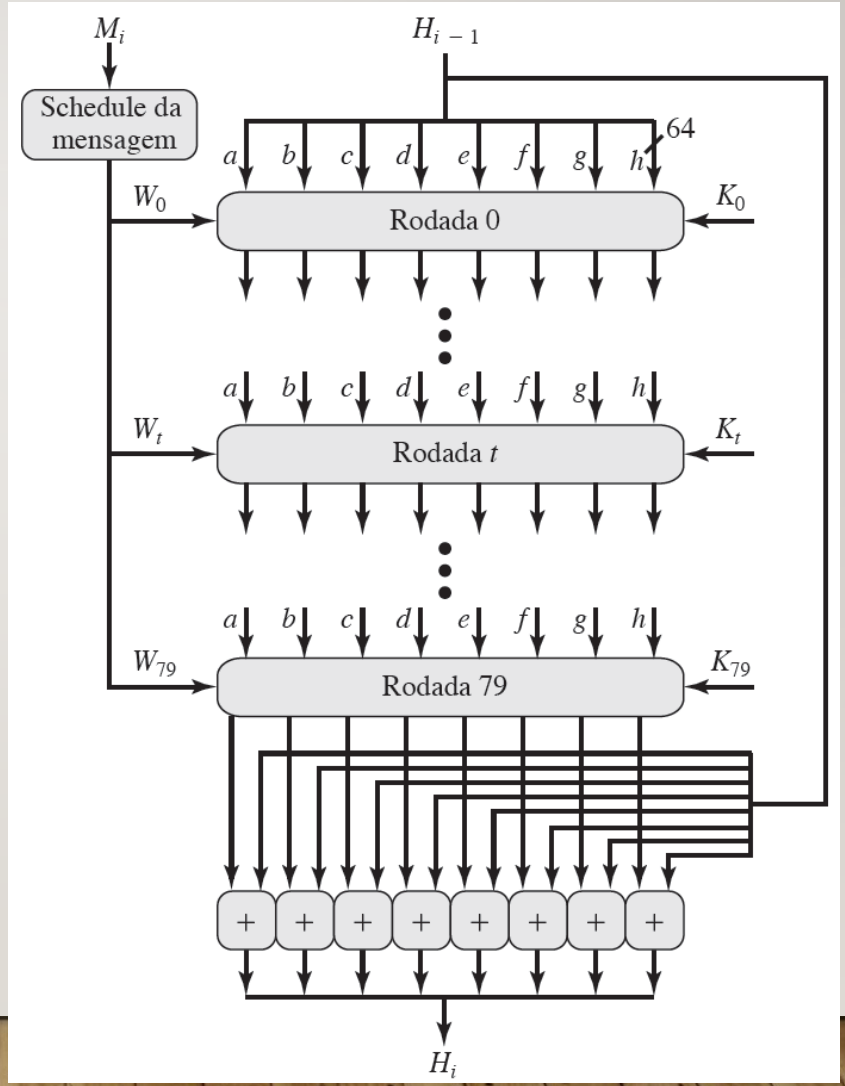


CONTROLE DE INTEGRIDADE DE DADOS (SECURE HASH ALGORITHM - SHA)

- Geração dos digestores de mensagens (SHA-512).
  - L: identifica o tamanho da mensagem (128 bits - formato “big-endian”)
  - Buffer hash inicial – buffer de 512 bits
    - a = 6A09E667F3BCC908 – registradores de 64 bits
    - b = BB67AE8584CAA73B – armazenados no formato big-endian
    - c = 3C6EF372FE94F82B
    - d = A54FF53A5F1D36F1
    - e = 510E527FADE682D1
    - f = 9B05688C2B3E6C1F
    - g = 1F83D9ABFB41BD6B
    - h = 5BE0CDI9137E2179

CONTROLE DE INTEGRIDADE DE DADOS (SECURE HASH ALGORITHM - SHA)

- Processando um bloco de 1024 bits (SHA-512).
  - H_0= IV
  - H_i= $Soma_{64}$ (H_{i-1}, $abcdefgh_i$)
  - MD= H_N
- Onde:
  - IV= Valor inicial do buffer abcdefgh
  - N= número de blocos na mensagem
  - MD= valor final do digestor de mensagem



CONTROLE DE INTEGRIDADE DE DADOS (SECURE HASH ALGORITHM - SHA)

$$T_1 = h + \mathrm{Ch}(e,f,g) + \left(\sum_1^{512} e\right) + W_t + K_t$$

$$T_2 = \left(\sum_0^{512} a\right) + \mathrm{Maj}(a,b,c)$$

$$a = T_1 + T_2$$

$$b = a$$

$$c = b$$

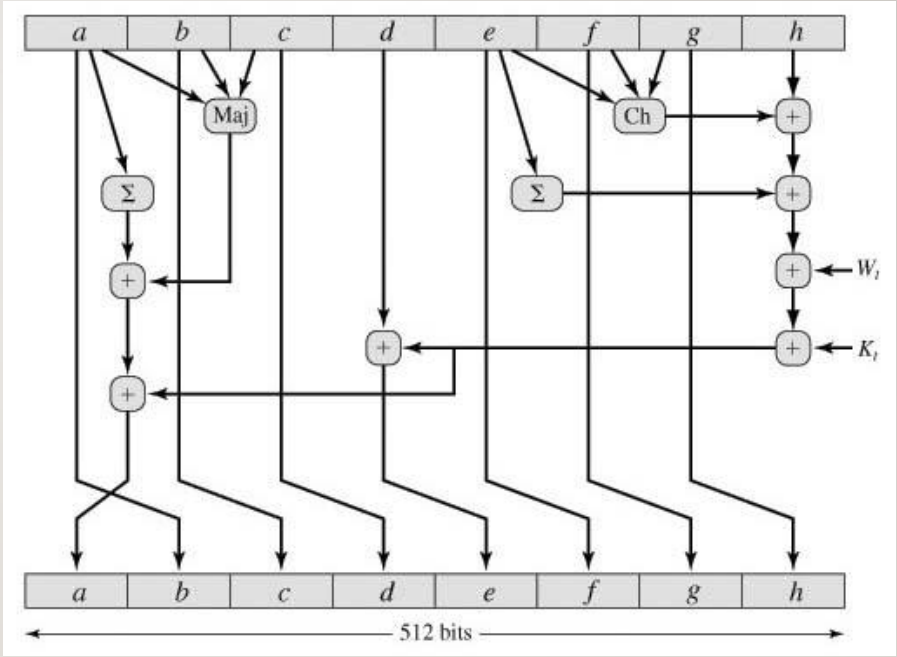
$$d = c$$

$$e = d + T_1$$

$$f = e$$

$$g = f$$

$$h = g$$



- Onde:
  - t: variando de 0 à 79
  - Ch(e,f,g)=(e AND f) xor (NOT e AND g)
  - Maj(a,c,b)=(a AND b) xor (a AND c) xor (b AND c)

$$\left(\sum_0^{512} a\right) = \mathrm{ROTR}^{28}(a) \oplus \mathrm{ROTR}^{34}(a) \oplus \mathrm{ROTR}^{39}(a)$$

$$\left(\sum_1^{512} e\right) = \mathrm{ROTR}^{14}(e) \oplus \mathrm{ROTR}^{18}(e) \oplus \mathrm{ROTR}^{41}(e)$$

CONTROLE DE INTEGRIDADE DE DADOS (SECURE HASH ALGORITHM - SHA)

- Exemplos:
- http://www.fileformat.info/tool/hash.htm
  - Mensagem: “abc”
  - Hash:
    - MD5: 900150983cd24fb0d6963f7d28e17f72
    - SHA-1: a9993e364706816aba3e25717850c26c9cd0d89d
    - SHA-256: ba7816bf8f01cfea414140de5dae2223b00361a396177a9cb410ff61f20015ad
    - SHA-512: ddaf35a193617abacc417349ae20413112e6fa4e89a97ea20a9eeee64b55d39a2192992a274fc1a836ba3c23a3feebbd454d4423643ce80e2a9ac94fa54ca49f

CONTROLE DE INTEGRIDADE DE DADOS (SALTING)

- Salting é usado para tornar o hash mais seguro. Se dois usuários tiverem a mesma senha, eles também terão os mesmos hashes de senha.
- Esse procedimento cria um resultado de hash diferente para as duas senhas. Um banco de dados armazena o hash e o salting.
  - https://www.symbionts.de/tools/hash/sha256-hash-salt-generator.html

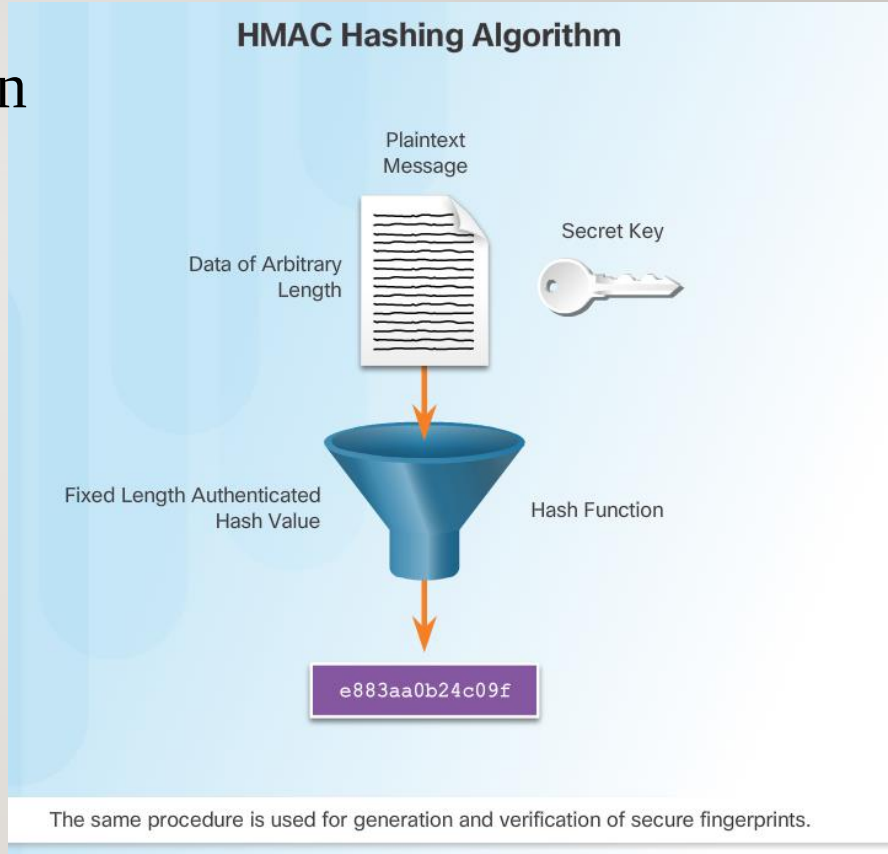
	Salt		Hash Value
Hash (“password” +	QxLUF1bIAdeQX)	=	b3bad1e5324f057753a4b8d7cef293e4
Hash (“password” +	R9PeIC7sxQXb8)	=	713c7beb54841a26a7c81eb06d6cf066

CONTROLE DE INTEGRIDADE DE DADOS (SALTING – PROTEÇÃO CONTRA ATAQUES)

- Salting evita que um atacante use um ataque de dicionário para tentar adivinhar senhas. Salting também torna impossível usar lookup tables e rainbow tables para quebrar o hash.
- Exemplo de Lookup Tables
  - Armazena os hashes pré-calculados de senhas em um dicionário de senha juntamente com a senha correspondente. Uma tabela de pesquisa
- https://crackstation.net

CONTROLE DE INTEGRIDADE DE DADOS (HMAC)

- Hash-Based Message Authentication Code - (HMACs) fortalecem os algoritmos de hash usando uma chave secreta adicional como entrada para a função hash.
- O uso do HMAC garante além da integridade, também a autenticação.
- Um HMAC usa um algoritmo específico que combina uma função de hash criptográfica com uma chave secreta.



- Usado em VPN (autentica a origem do pacote e garante a integridade dos dados), equipamentos de rede, AWS (assinatura HMAC-SHA + preenchimento dos campos)
- http://www.freeformatter.com/hmac-generator.html

BIBLIOGRAFIA

- **Bibliografia:**

  - KUROSE, James F; ROSS, Keith W. Redes de computadores e a internet: uma abordagem Top-Down. 6. ed. São Paulo: Pearson, c2014.
  - BURNETT, M. How Secure is my Password?. Disponível em: <https://www.security.org/how-secure-is-my-password>. Acesso em: 02.03.2024.
  - SPG. How Secure is my Password?. Disponível em: <https://strongpasswordgenerator.com>. Acesso em 02.03.2024.

BIBLIOGRAFIA

- **Bibliografia:**

  - INFO, F. F. Hash Functions. Disponível em: <http://www.fileformat.info/tool/hash.htm>. Acesso em: 02.03.2024.
  - COM, F. HMAC Generator / Tester Tool. Disponível em: <http://www.freeformatter.com/hmac-generator.html>. Acesso em: 02.03.2024.
  - SECURITY, D. Crack Station. Disponível em: <https://crackstation.net/>. Acesso em: 02.03.2024.
  - SALTING. Disponível em: <https://www.symbionts.de/tools/hash/sha256-hash-salt-generator.html> . Acesso em: 02.03.2024.
  - Notas de aula.