Capítulo 5 – Criptografia

- ✔ 5.1 – Introdução
- ✔ 5.2 – Criptografia - algoritmos e chaves
- ✔ 5.3 – Tipos de criptografia
  - 5.3.1 – Criptografia simétrica
  - 5.3.2 – Criptografia assimétrica
  - 5.3.3 - Funções *hash*
- ✔ 5.4 - Assinaturas digitais
- ✔ 5.5 – Certificação digital
- ✔ 5.6 - Exemplo completo

5.1 – Introdução

- ***Criptografia*** é a ciência de escrever e ler mensagens cifradas.
- É a base de muitas aplicações que envolvem segurança digital:
  - autenticação
  - comunicação segura
  - dinheiro eletrônico
  - certificados digitais
  - identidade digital
  - assinatura digital
  - etc

- Com a criptografia é possível garantir três propriedades básicas de segurança:
  - autenticação
  - integridade
  - confiabilidade
- Começou no Império Romano, com Júlio César
- Evoluiu bastante na Segunda Guerra Mundial:
  - Alemães tinham a necessidade de enviar mensagens secretas
  - Os aliados desenvolveram máquinas para “quebrar” a criptografia dos alemães
- ***Criptoanálise***: ciência de quebrar códigos e decifrar informação sem conhecer a chave
- Criptografia e criptoanálise evoluem à medida que novos processadores mais rápidos são criados

5.2 – Criptografia - algoritmos e chaves

- Elementos básicos indispensáveis à criptografia:
  - Algoritmo de criptografia: função matemática usada para cifrar e reverter o processo
  - Chave: valor secreto necessário para realizar as operações de criptografia. É composta de um número específico de bits
- Um bom algoritmo de criptografia é aquele conhecido publicamente e cuja segurança foi atestada por um grande número de criptoanalistas
- Sem um algoritmo e pelo menos uma chave não é possível realizar criptografia

- Quanto maior a chave, mais difícil é o sucesso de um ataque de força bruta
  - Ataque de força bruta consiste em testar todas as combinações possíveis de chave, no intuito de encontrar a chave certa para decifrar a informação
- Quanto maior o tamanho da chave, maior a segurança da criptografia, maior o tempo necessário para cifrar e decifrar a informação
  - um valor ótimo é definido de acordo com a capacidade de processamento dos computadores da época
  - de tempos em tempos o tamanho da chave deve ser aumentado

5.3 – Tipos de criptografia

- Algoritmos de criptografia são divididos em três categorias distintas:
  - Algoritmos simétricos: criptografia de chave privada
  - Algoritmos assimétricos: criptografia de chave pública
  - Funções hash

5.3.1 – Criptografia simétrica

- Utiliza a mesma chave para cifrar e decifrar uma mensagem
- Nesse caso, a chave precisa ser pré-combinada entre os participantes da criptografia
- A chave deve ser mantida em segredo entre os participantes
- Não deve ser transmitida via Internet, pois o risco de alguém farejar a rede e se apoderar da chave é muito grande
  - utiliza-se disquetes, cd-roms, pendrives entregues pessoalmente ao destinatário => out of band
- Quando for necessária a transmissão via rede, outros métodos de criptografia devem ser utilizados em conjunto para a transmissão dessa chave

- Normalmente, os algoritmos simétricos operam em blocos de mensagem de 64 bits cada
  - necessário quebrar a mensagem em blocos de 64 bits e agrupá-los de alguma forma
- Existem 4 métodos de combinar a mensagem original, a chave e a mensagem cifrada:
  - **ECB:** cifra cada bloco de 64 bits de forma independente, usando a mesma chave
    - não é considerado seguro
    - uma mesma mensagem cifrada várias vezes vai gerar sempre o mesmo código
  - **CBC:** cada bloco depende do resultado do bloco anterior, ou seja, utiliza-se o bloco anterior no processo de cifragem do bloco seguinte
    - o método deve garantir que todos blocos cheguem corretamente, pois um erro em um bloco se propagará para todos os outros blocos

- **CFB:** capaz de cifrar dados de qualquer tamanho, independente do bloco
  - útil para cifrar pequenas quantidades de informação, ou informações que devem ser imediatamente transmitidas independentemente de completar um bloco ou não
- **OFB:** Parecido com o CBC, porém não existe dependência entre os blocos
  - múltiplas cifragens da mesma mensagem produzem mensagens diferentes, ainda que utilizando a mesma chave

✔ Algoritmos conhecidos de criptografia simétrica:

- DES, 3DES, RC-4, IDEA, e o AES

DES e 3DES

- ✔ Foi, por muitos anos, o algoritmo padrão para criptografia simétrica
- ✔ Opera com blocos de 64 bits
- ✔ Na sua forma padrão, usa chave de 64 bits:
  - 56 bits são randômicos e 8 são bits de paridade para verificar a integridade da chave
- ✔ Pode ser usado nos quatro modos: ECB, CBC, CFB e OFB
- ✔ Variação do DES de 40 bits:
  - criada pelo governo americano para permitir seu uso fora dos EUA
    - na época, os EUA não permitiam a exportação de qualquer sistema de criptografia com mais de 40 bits de chave
  - era o próprio algoritmo DES, sendo que 24 bits da chave eram fixos

- 3DES é outra variação do DES, com o intuito de dificultar os ataques de força bruta
- Realiza 3 operações em um bloco de 64 bits: cifra, decifra e cifra de novo
- Pode usar uma, duas ou três chaves diferentes
- É comum ouvir que o 3DES usa uma chave de 192 bits, visto que pode usar 3 chaves de 64 bits diferentes
- Não é tão seguro quanto um algoritmo que efetivamente use uma chave de 192 bits
- Tanto o DES quanto o 3DES são públicos e sua especificação pode ser encontrada em:
  - http://www.itl.nist.gov/fipspubs/fip46-2.htm

RC-4

- ✔ Algoritmo proprietário criado pelo MIT
- ✔ É de propriedade da RSA Data Security
- ✔ Normalmente usa chave de 128 bits

IDEA

- ✔ Surgiu com o propósito de substituir o DES
- ✔ Opera em blocos de 64 bits
- ✔ Usa chave de 128 bits
- ✔ Pode operar nos quatro modos: ECB, CBC, CFB e OFB
- ✔ Desenvolvido para ser eficiente tanto em hardware quanto em software
- ✔ Patenteado e requer licença de uso

AES

- ✔ Foi o último algoritmo aprovado pelo FIPS (*Federal Information Processing Standards*)
- ✔ Resultado de uma seleção de diversos algoritmos
- ✔ Aprovado em 2001
- ✔ Trabalha com blocos de 128 bits
- ✔ Capaz de utilizar chaves criptográficas de 128, 192 e 256 bits

Algoritmo Diffie-Hellman de troca de chaves simétricas

- ✔ Permite a troca de chaves em um canal inseguro sem nenhum conhecimento prévio de qualquer segredo
  - http://www.rsasecurity.com/rsalabs/node.asp?id=2248

5.3.2 - Criptografia assimétrica

- ✔ Pode usar o mesmo algoritmo ou algoritmos diferentes para cifrar e decifrar a informação
- ✔ Utiliza duas chaves distintas para cifrar e decifrar
  - **Chave Pública:** distribuída livremente
  - **Chave Privada**: pertence apenas ao dono do par de chaves
- ✔ O par de chaves possui propriedades importantes:
  - Uma mensagem cifrada com a chave pública só pode ser decifrada com a chave privada
  - Uma mensagem cifrada com a chave privada só pode ser decifrada com a chave pública
  - Derivar a chave privada a partir da chave pública é computacionalmente inviável

- Duas pessoas (Alice e Bob) se comunicando usando criptografia assimétrica:
  - 1. Alice e Bob, cada um deles cria seu próprio par de chaves (pública e privada)
  - 2. Alice e Bob trocam sua chaves públicas
  - 3. Alice escreve uma mensagem para Bob e usa a chave pública de Bob para cifrar a mensagem. Alice envia o resultado para Bob
  - 4. Bob usa a sua chave privada para decifrar a mensagem
  - 5. Bob escreve uma resposta, cifra com a chave pública de Alice e envia para Alice
  - 6. Alice usa a sua chave privada para decifrar a resposta
- **Dúvida:** como garantir que a chave pública de Alice realmente pertence a Alice, e não a uma pessoa se fazendo passar por Alice?
  - Qualquer pessoa poderia gerar um par de chaves como se fosse Alice e distribuir uma chave pública falsa

- Solução?
  - necessário um mecanismo para atestar a autenticidade de uma chave pública, chamado de ***certificação digital***
- Algoritmo mais conhecido de chave pública é o RSA, que se tornou público em 6 de setembro de 2000
- Maiores informações sobre o RSA pode ser obtidas no padrão PKCS #1 (*Public Key Cryptography Standard #1*), encontrado em:
  - http://www.rsasecurity.com/rsalabs/node.asp?id=2125

5.3.3 - Funções *hash*

- ✔ É uma função especial que, a partir de uma entrada de tamanho arbitrário, produz uma saída de tamanho fixo
- ✔ A saída é normalmente chamada de *hash* ou *digest* da mensagem original
- ✔ Um algoritmo adequado para uma função *hash* deve possuir as seguintes propriedades:
  - 1. Consistente: a mesma entrada deve gerar sempre a mesma saída (???)
  - 2. Randômico: gerar uma saída que não permita descobrir informações sobre a mensagem original
  - 3. Único: deve ser quase impossível que duas mensagens diferentes produzam o mesmo *hash*
    - quando isso ocorre dizemos que houve “colisão”
    - em algoritmos de *hash* usados em criptografia, uma “colisão” pode indicar fraqueza no algoritmo

- 4. *One way*: a partir da saída, deve ser quase impossível descobrir a mensagem original

- ✔ Para que serve?
- ✔ Uma função *hash* gera uma representação de uma mensagem ou arquivo
- ✔ Análogo à impressão digital humana => teoricamente único
- ✔ “Representa” a mensagem original em um espaço de armazenamento menor
- ✔ Usado para garantir a integridade de uma mensagem

- Duas pessoas (Alice e Bob) se comunicando usando *hash*:
  - 1. Alice escreve uma mensagem. Após a mensagem pronta, um *hash* é calculado para representar unicamente a mensagem
  - 2. A mensagem original é enviada para Bob junto com o hash
  - 3. Bob pega a mensagem original e calcula um novo *hash* da mensagem
  - 4. Caso os dois *hashes* sejam iguais, a mensagem não foi alterada
- Um usuário malicioso pode mudar a mensagem e o *hash*, tornando a nova mensagem “válida”
- Para garantir a confiabilidade das funções *hash* como assinaturas, elas devem ser combinadas com a criptografia pública

- Algoritmos mais comuns de *hash*: MD5 e SHA
- MD5:
  - processa entrada em blocos de 512 bits e produz uma saída de 128 bits
- SHA:
  - processa entrada em blocos de 512 bits e produz uma saída de 160 bits
  - processamento mais intensivo que o MD5
  - mais lento que o MD5
  - SHA-1: atualização por questões de segurança
  - SHA-2: variações do SHA → SHA-224, SHA-256, SHA-384 e SHA-512
- Pesquisadores chineses anunciaram, em 2004, uma série de colisões no MD5
  - http://eprint.iacr.org/2004/199.pdf
  - http://eprint.iacr.org/2004/146.ps

- ✔ Ainda não há nenhuma colisão conhecida no algoritmo SHA-1
- ✔ Através de um ataque de colisão, um *hacker* poderia substituir uma mensagem por outra, desde que elas tenham o mesmo *hash* resultante
- ✔ Como o *hacker* não pode escolher uma mensagem arbitrária, este tipo de ataque é difícil de ser realizado, apesar de não ser impossível
- ✔ Um típico ataque bem sucedido poderia ser a substituição de um valor de transferência bancária por um outro de valor mais alto, mas que produziria o mesmo *hash*
- ✔ Informações sobre colisões:
  - http://www.criptography.com/cnews/hash.html

5.4 - Assinaturas digitais

- Por meio de uma assinatura digital, é possível atestar a identidade do signatário de um documento (***autenticação***) e garantir que um determinado documento não foi alterado (***integridade***)
- Assinatura digital necessita de criptografia assimétrica e funções *hash*
- Os algoritmos de assinatura digital mais comuns são: RSA e o DSS
- Assinatura digital não garante ***confidencialidade***
- Pode ser combinada com um algoritmo de criptografia para garantir a confidencialidade
- Assinaturas digitais sofrem do mesmo problema de autenticidade das chaves públicas

- Duas pessoas (Alice e Bob) se comunicando usando assinaturas digitais:
  - 1. Alice cria um par de chaves
  - 2. Alice envia sua chave pública para Bob
  - 3. Alice escreve uma mensagem para Bob e usa a mensagem como entrada para uma função *hash*, que produzirá uma saída de tamanho fixo (*message digest*)
  - 4. Alice cifra a *message digest* com sua chave privada, resultando na assinatura digital
  - 5. Alice envia a mensagem original com a assinatura digital para Bob
  - 6. Bob utiliza a chave pública de Alice para decifrar a assinatura, recuperando a *message digest*
  - 7. Bob calcula a *message digest* da mensagem recebida, usando a mesma função usada por Alice
  - 8. Caso os dois *hashes* (*message digest*) sejam iguais, a mensagem foi enviada por uma pessoa com a chave privada de Alice e não foi modificada na transmissão

5.5 - Certificação digital

- Um certificado digital consiste em uma mensagem assinada digitalmente, usada para atestar a validade de uma chave pública
- Certificados digitais normalmente seguem o padrão X.509 do ITU-T
  - Versão do X.509
  - Número serial do certificado
  - Informação de algoritmos do gerador do certificado
  - Identificação do gerador do certificado
  - Datas de validade
  - Informação sobre o algoritmo assimétrico do dono do certificado
  - Assinatura digital da autoridade geradora do certificado

- Um certificado deve ser assinado por uma entidade certificadora (*Certification Authorities* - **CA**) para ser considerado válido
- CA's são entidades confiáveis que emitem e atestam certificados
  - análogos a cartórios que verifica assinaturas normais
- A chave pública da CA deve ser amplamente distribuída
  - e.g.: *browsers* já trazem essas chaves pré-instaladas
- Uma CA é responsável por criar, distribuir e invalidar certificados
  - invalidação de certificados é feita através de listas especiais chamadas de **CRL** (*Certificate Revogation Lists*)
- **PKI** (*Public Key Infrastructure*) ou **ICP** (*Infra-estrutura de Chaves Públicas*)
  - conjunto de hardware e software, pessoal, políticas e procedimento para se criar uma infra-estrutura de certificação digital
  - Governo Federal Brasileiro possui o ICP Brasil

5.6 - Exemplo completo

- 1. Alice e Bob geram seus respectivos pares de chaves
- 2. Alice e Bob entram em contato com uma CA para obter seus certificados. A CA, após certificar a identidade de cada um, gera um certificado assinado com a chave privada da CA
- 3. Alice e Bob trocam certificados (ou solicitam da CA). Verificam a autenticidade com a chave pública da CA (amplamente divulgada) e passam a confiar nas respectivas chaves públicas
- 4. Alice prepara uma chave simétrica randômica e calcula o seu *hash* utilizando uma função conhecida por Bob. Alice cifra o *hash* resultante com sua chave privada, formando uma assinatura digital

- 5. Alice cifra a chave simétrica + assinatura, utilizando a chave pública de Bob, e transmite o conjunto resultante para Bob
- 6. Bob, utilizando sua chave privada, decifra o conjunto chave simétrica + assinatura
- 7. Utilizando a chave pública de Alice, Bob decifra a assinatura. Bob calcula o *hash* da chave simétrica utilizando a mesma função de Alice. Caso os dois *hashes* sejam iguais, a chave simétrica foi enviada por Alice e não foi alterada
- 8. Utilizando a chave simétrica conhecida pelas duas partes, Alice e Bob realizam uma conversa cifrada. Para garantir integridade, cada mensagem é cifrada junto com um *hash*