Criptografia Aplicada

Assinaturas digitais

LabSEC
Laboratório de Segurança em Computação

cpqd

Sumário

- Princípios básicos
- Assinatura digital com RSA
- Segurança
- Assinaturas RSA na prática

Assinaturas Digitais

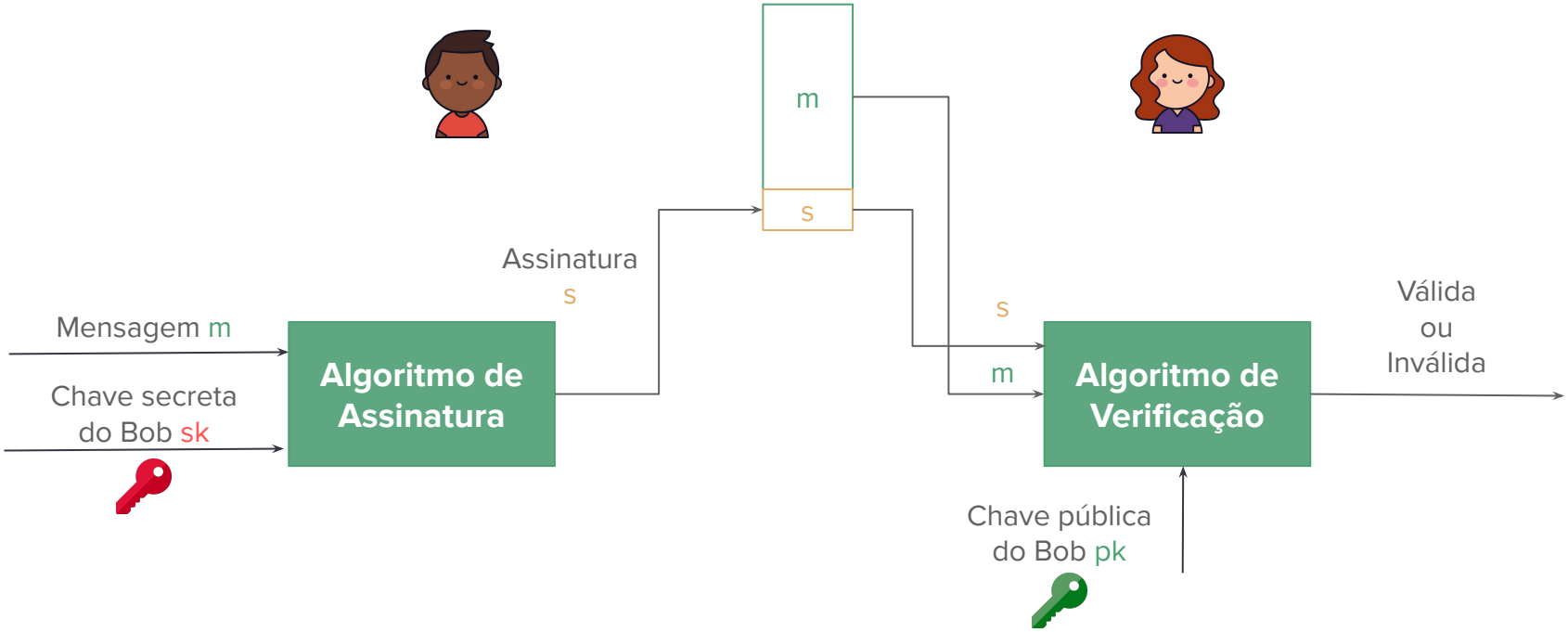
- Documentos digitais são facilmente manipuláveis
  - colar uma imagem de uma assinatura não provê segurança suficiente
  - qualquer pessoa pode colar essa imagem em diversos documentos
- Queremos obter o equivalente à assinaturas em papel
  - assinatura deve estar fortemente ligada ao documento
  - a identidade do assinante deve ser clara
  - apenas aquele assinante deve ser capaz de criar uma assinatura no nome dele
- Garantias com assinatura digital:
  - Integridade
  - Autenticidade
  - Não-repúdio

Características

Um esquema de assinatura possui os seguintes ingredientes:

- mensagem *m* a ser assinada
- par de chaves (uma pública (pk) e outra privada (sk))
- algoritmo de assinatura
- algoritmo de verificação de assinatura
- extra: uma função de hash

Assinatura Digital



Requisitos

- A assinatura deve depender de cada bit da mensagem
- Deve usar algo único do assinante
  - para prevenir falsificação e negação de assinatura por parte do assinante
- Deve ser fácil de produzir, reconhecer e verificar
- Deve ser computacionalmente inviável falsificar uma assinatura
  - tanto construir uma nova mensagem para uma assinatura existente
  - quanto construir uma assinatura falsificada para uma dada mensagem
- Dever ser possível reter uma cópia da assinatura de maneira prática

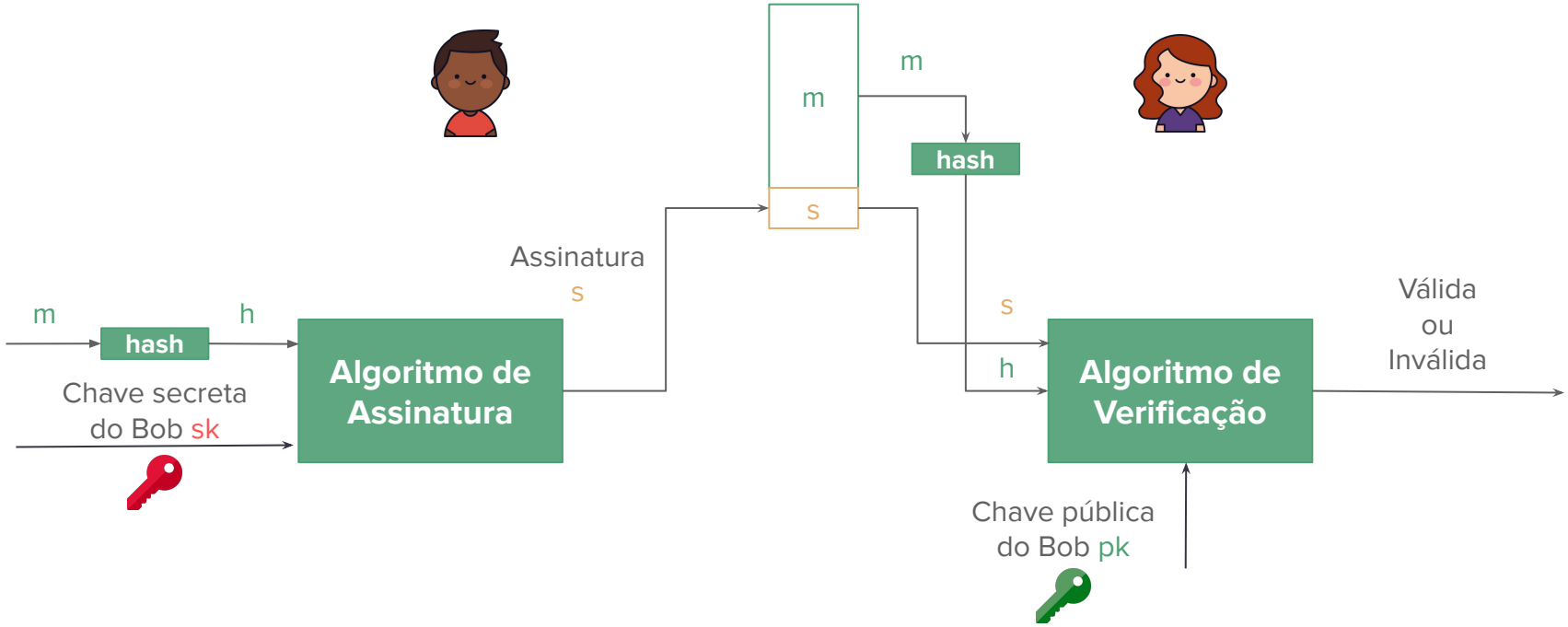
Algoritmos necessários

- Geração de chaves: (pk,sk) = KeyGen
- Assinatura: s = Sign(m, sk)
- Verificação: Verify(m, s, pk)
  - “válida” se s = Sign(m, sk)
  - “inválida” caso contrário

Assinatura digital e hash

- Normalmente, esquemas de assinatura são usados juntamente com uma função de hash criptográfica
- O hash da mensagem é assinado, ao invés da mensagem
  - Assinatura: s = Sign(h(m), sk)
  - Verificação: Verify(h(m), s, pk)

Assinatura digital e hash



Esquemas de assinatura digital

- O funcionamento interno dos algoritmos KeyGen, Sign e Verify dependem do esquema de assinatura utilizado
- Neste curso, estudaremos:
  - RSA
  - DSA
  - ECDSA
- Estes esquemas são apresentados no FIPS 186-5
- Existem diversos outros esquemas de assinatura digital
  - alguns, inclusive, resistentes contra computadores quânticos!

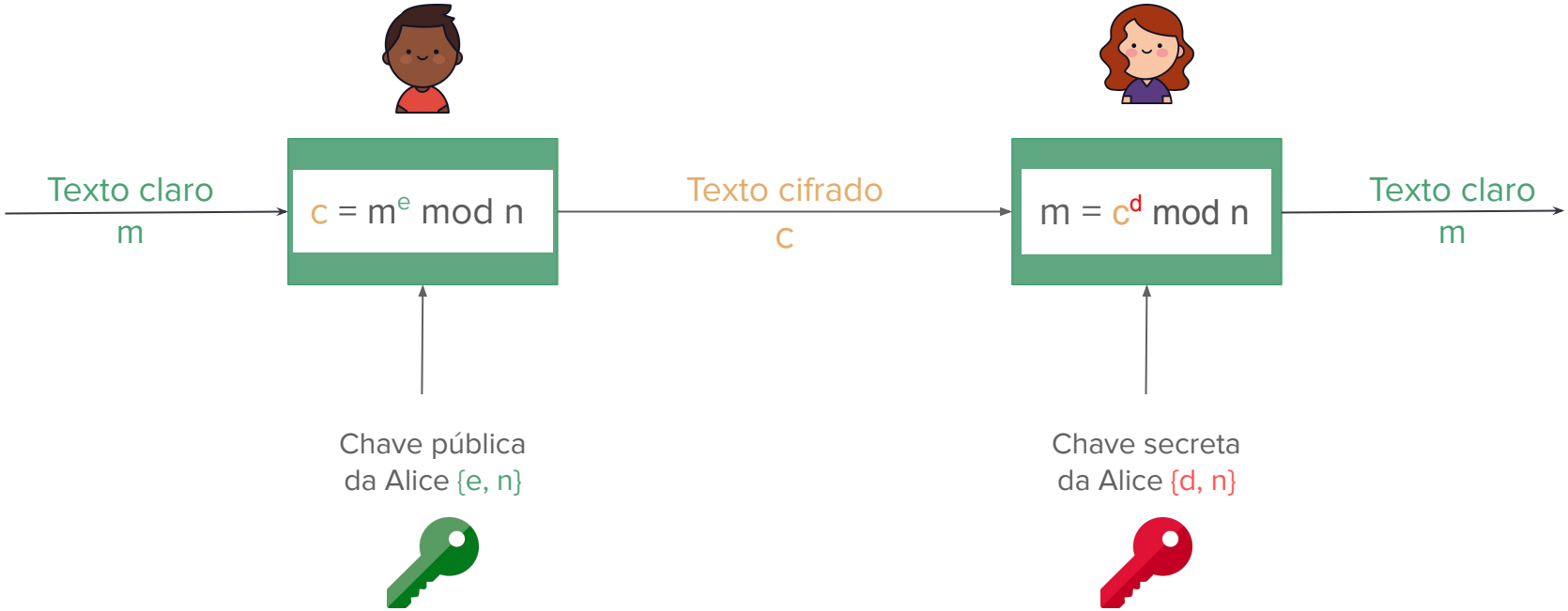
Sumário

- Princípios básicos
- **Assinatura digital com RSA**
- Segurança
- Assinaturas RSA na prática

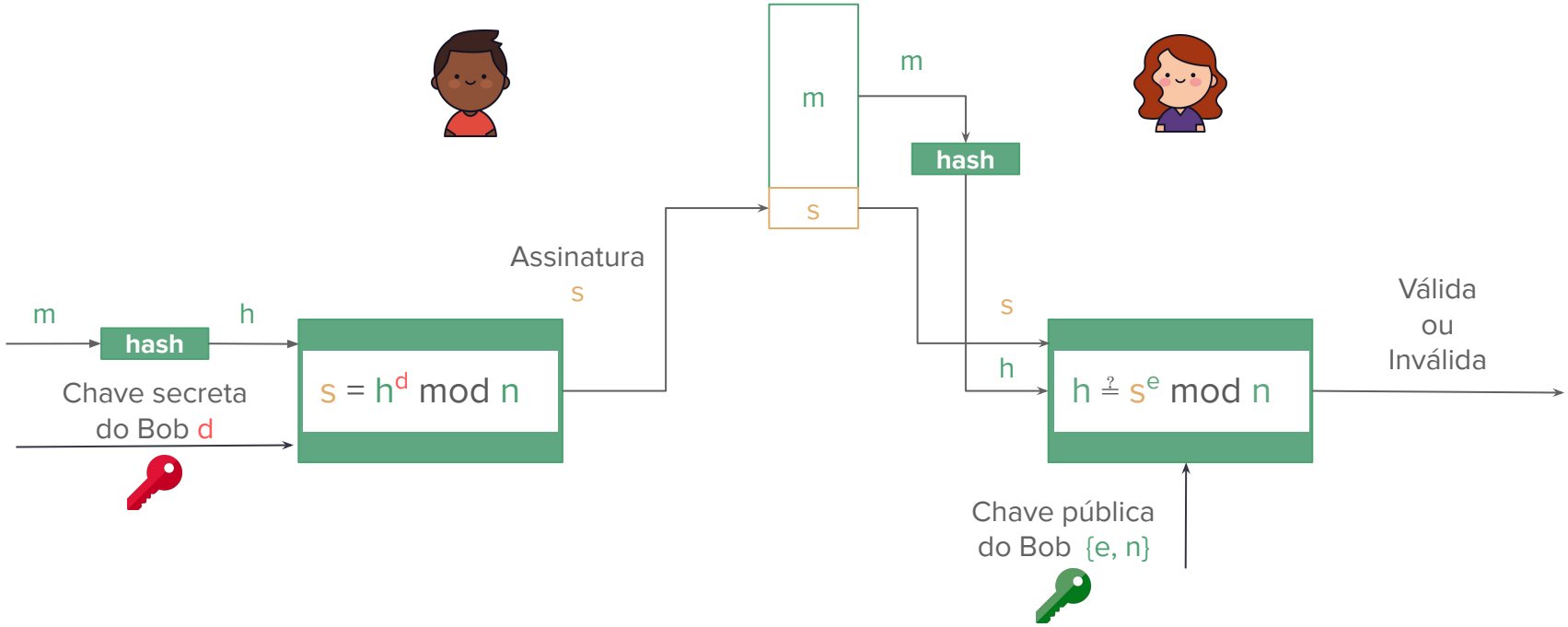
RSA

- Nas aulas anteriores, vimos geração de chaves, cifragem e decifragem com o criptossistema RSA
- Ele também pode ser utilizado para prover assinaturas digitais
  - com a modificação de que a chave privada é utilizada na geração da assinatura e a chave pública na verificação

RSA: Cifragem e Decifragem



RSA: Assinatura Digital



RSA: algoritmos

- Geração de chaves: KeyGen()
  - Sejam p e q números primos secretos, $n = pq$ e $\phi(n) = (p-1)(q-1)$, e e d tais que $ed \equiv 1 \pmod{\phi(n)}$
  - retorna {e, n} como chave pública e {d, n} como chave privada

- Assinatura: $s = \text{Sign}(h, d) = h^d \bmod n$

- Verificação: Verify(m, s, e)

  “válida” se $h(m) \equiv s^e \pmod n$
  “inválida” caso contrário

Considerações

- O expoente *d* precisa ser mantido em segredo
- O módulo *n* precisa ter pelo menos 2048 bits
- Apenas funções de hash aprovadas devem ser utilizadas na geração de assinaturas

Sumário

- Princípios básicos
- Assinatura digital com RSA
- **Segurança**
- Assinaturas RSA na prática

Segurança em assinaturas digitais

- Dada uma mensagem *m*, deve ser computacionalmente inviável para qualquer outra pessoa além do Bob produzir uma assinatura válida s tal que Verify(m, s, pk) = válida
- Se um atacante conseguir calcular um par (m, s) válido para uma mensagem *m* que não foi previamente assinada por Bob, dizemos que isso é uma ***falsificação***.
- Uma assinatura ***falsificada*** é uma assinatura válida produzida por alguém que não é dono da chave privada correspondente.

Modelos de ataques em assinaturas digitais

No caso de assinaturas digitais, os seguintes modelos de ataques são considerados:

- **key-only attack**: o atacante conhece apenas a chave pública
- **known message attack**: o atacante conhece uma lista de mensagens e assinaturas previamente assinadas por Bob (m_1, s_1), (m_2, s_2), ...
- **chosen message attack**: o atacante escolhe uma lista de mensagens m_1, m_2, .. e consegue a assinatura do Bob nessas mensagens
- **selective forgery**: com probabilidade não-negligenciável, o atacante consegue produzir uma assinatura válida para uma mensagem *m* escolhida por alguém e não previamente assinada por Bob
- **existential forgery**: o atacante é capaz de criar uma assinatura válida para pelo menos uma mensagem, ou seja, criar um par (m, s) para uma *m* não previamente assinada por Bob.
- **total break:** o atacante é capaz de determinar a chave secreta do Bob

Segurança em assinaturas digitais

- Um esquema de assinatura não pode ser *incondicionalmente* seguro, já que o atacante pode testar todas as possíveis assinaturas *s* para dada mensagem *m*
  - ou seja, executar Verify(m, s, pk) para todo possível s
- O objetivo é o de encontrar esquemas de assinatura que sejam *computacionalmente* seguros

Hash e segurança

- Precisamos garantir que o uso da função de hash não enfraqueça o esquema
- Por exemplo, dado o par (m,s), um atacante pode tentar encontrar uma m' tal que h(m) = h(m')
  - nesse caso, a assinatura s seria uma assinatura válida para a nova mensagem m'
- Para evitar esse ataque, a função h(.) precisa satisfazer as propriedades de hash já vistas
  - neste caso, precisa ser resistente à 2a pré-imagem

Segurança do RSA

- O RSA que aprendemos aqui pode tem algumas vulnerabilidades
- A maioria dessas vulnerabilidades são evitadas se usarmos funções de hash seguras
- Outros cuidados também são tomados na implementação dos algoritmos
    - Assinaturas digitais com RSA são tão seguras quando o esquema de cifragem/decifragem RSA se acompanhadas de uma formatação com *padding* e aleatoriedade
    - Esse esquema é conhecido como RSA-PSS (RSA Probabilistic Signature Scheme)
    - Mais detalhes: FIPS 186-5

Segurança do RSA

Exemplo 1: um atacante pode construir uma assinatura válida para uma mensagem se ele primeiro escolher s e depois calcular a mensagem como
$m = s^e \pmod n$.

- neste caso, a mensagem *m* possivelmente não teria um significado relevante
- além disso, o uso de funções de hash evitaria esse problema, pois $h = s^e \pmod n$, e o atacante precisaria descobrir *m* tal que h = h(m)
- note que, aparentemente, não há uma forma de escolher o *m* e depois criar a assinatura falsificada s, caso contrário o RSA seria inseguro.

Segurança do RSA

Exemplo 2: Um atacante obtém um par válido (m,s) previamente assinado por Bob.

- o atacante calcula h = h(m) e tenta encontrar m' ≠ m tal que h(m') = h(m)
- a assinatura *s* também será válida para m'
- esse seria um *existential forgery* usando um *known message attack*
- o ataque pode ser prevenido usando uma função de hash resistente à segunda pré-imagem

Segurança do RSA

Exemplo 3: Um atacante pode encontrar duas mensagens m ≠ m' tal que h(m) = h(m') e persuadir Bob a assinar *m*.

- essa assinatura gerada por Bob seria também uma assinatura válida para m'
- esse seria um *existential forgery* usando um *chosen message attack*
- o ataque pode ser prevenido usando uma função de hash resistente à colisões

Recomendações de segurança

- Quando uma função de hash e um algoritmo de assinatura são combinados, a segurança da assinatura é determinada pelo algoritmo mais fraco (NIST SP 8000-57)
- Apenas funções de hash aprovadas devem ser utilizadas na geração de assinaturas
- Quando os parâmetros são gerados aleatoriamente (ex: e), deve ser usado um gerador de números aleatórios aprovado
- O par de chaves gerado deve ser utilizado apenas para gerar e verificar assinaturas
- Chaves privadas devem ser protegidas de acesso não autorizado
- Chaves públicas devem ser protegidas de modificações não autorizadas
- Mais recomendações e detalhes em: NIST FIPS 186-5

Sumário

- Princípios básicos
- Assinatura digital com RSA
- Segurança
- **Assinaturas RSA na prática**

Atividade: assinatura RSA com openssl

- Faça o download dos arquivos assinatura.bin, msg.txt e thais.publica.pem
- Verifique a assinatura do arquivo de texto usando a chave pública:

```
openssl dgst -sha256 -verify thais.publica.pem -signature assinatura.bin msg.txt
```

- Se estiver OK, o openssl vai retornar "Verified OK"

Atividade: assinatura RSA com openssl

- Vamos utilizar as chaves RSA criadas anteriormente:
  - `openssl genrsa -aes256 -out seunome.privada.pem 2048`
  - `openssl rsa -pubout -in seunome.privada.pem -out seunome.publica.pem`
- Crie um arquivo de texto qualquer com uma mensagem
  - `echo "Mensagem autentica" > msg.txt`
- Assine a mensagem usando a chave privada
  - `openssl dgst -sha256 -sign seunome.privada.pem -out assinatura.bin msg.txt`
- Verifique a assinatura
  - `openssl dgst -sha256 -verify seunome.publica.pem -signature assinatura.bin msg.txt`

Atividade: assinatura RSA com openssl

- Modifique o arquivo de texto e tente verificar a assinatura novamente
  - o que acontece?
- Gere uma assinatura de um documento e peça para um colega verificar
  - que informações são necessárias?

Resumo

- Princípios básicos de assinatura digital
  - requisitos, algoritmos, uso do hash
- Assinatura digital com RSA
  - primeiro esquema de assinatura digital
  - baseado no problema de fatoração
- Segurança
  - diversos modelos de ataque
  - o uso de uma função de hash segura é fundamental

Referências

- W. Stallings. *Cryptography and network security*. 7a edição.
  - Definições, requisitos e ataques: 13.1
  - RSA-PSS: 13.6
- D. Stinson e M. Paterson. *Cryptography: Theory and Practice*. 4a edição.
  - Introdução: 1.2.2
  - RSA: 8.1
  - Segurança: 8.2
- Joachim von zur Gathen. *CryptoSchool*. 1a edição.
  - Assinaturas: 8
- imagem: Flaticon.com