(CMPSEGS)

Segurança de Sistemas

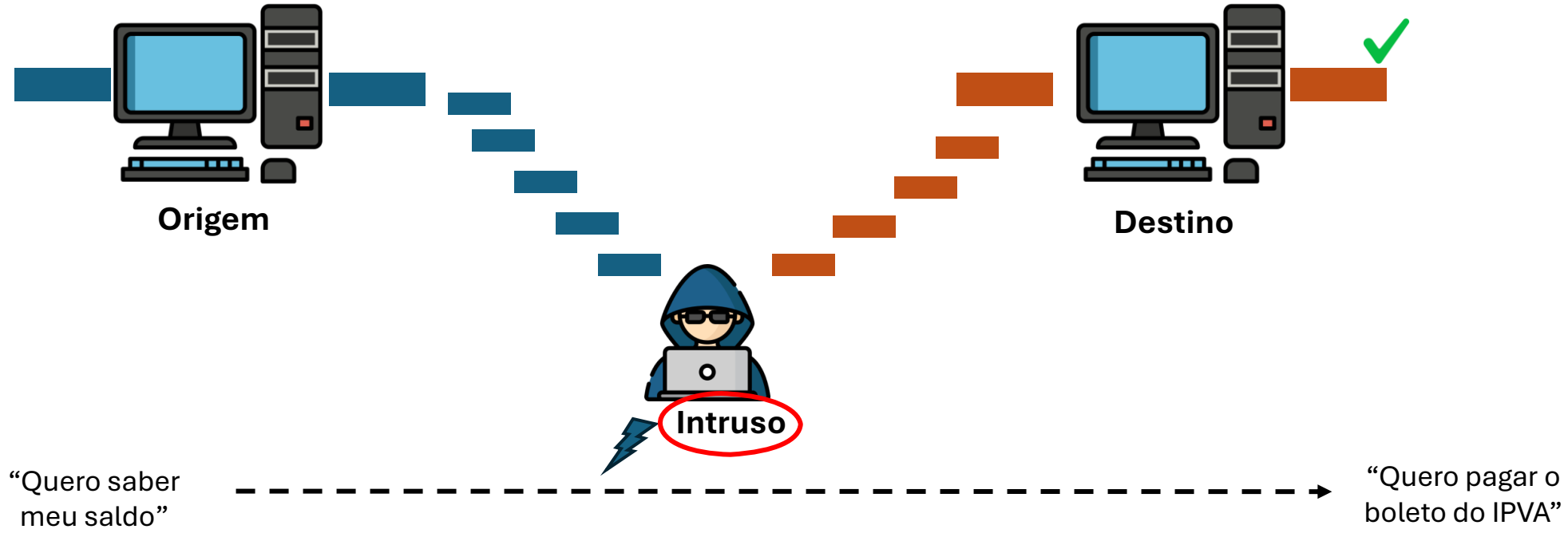
Tecnologia em Análise e Desenvolvimento de Sistemas

Noções de criptografia: funções de Hash, códigos de autenticação, números aleatórios
Algoritmos assimétricos e certificação digital

Prof. Me. Leonardo Arruda
leonardo.arruda@ifsp.edu.br

Nos episódios anteriores...

- **Integridade**
  - Capacidade de verificar se a informação foi alterada.



Integridade: redundância

Exemplo prático (não-criptográfico):

- RG/CPF: usa **digito verificador** (DV)

- **Método**: “mod 11”
  - Digito é multiplicado por sua posição, indo do menos significativo (peso 2) até o mais significativo
  - Os resultados são somados
  - **DV:** resto da divisão desta soma por 11

Exemplo simplificado

Entrada:	2	3	5	9	2
Posição:	6	5	4	3	2
Multiplicação:	12	15	20	27	4
Soma:	78				
Digito verificador:	78 mod 11 = 1				

Integridade: redundância

Exemplo prático (não-criptográfico):

- RG/CPF: usa **digito verificador** (DV)

- **Método**: “mod 11”
  - Digito é multiplicado por sua posição, indo do menos significativo (peso 2) até o mais significativo
  - Os resultados são somados
  - **DV:** resto da divisão desta soma por 11

Exemplo simplificado

Entrada:	2	3	5	9	2
Posição:	6	5	4	3	2
Multiplicação:	12	15	20	27	4
Soma:	78				
Digito verificador:	78 mod 11 = 1				

Integridade: redundância

Exemplo prático (não-criptográfico):

- RG/CPF: usa **digito verificador** (DV)

- **Método**: "mod 11"
  - Digito é multiplicado por sua posição, indo do menos significativo (peso 2) até o mais significativo
  - Os resultados são somados
  - **DV:** resto da divisão desta soma por 11

Exemplo simplificado

Entrada:	2	3	5	9	2
Posição:	6	5	4	3	2
Multiplicação:	12	15	20	27	4
Soma:	78				
Digito verificador:	78 mod 11 = 1				

Integridade: redundância

Exemplo prático (não-criptográfico):

- RG/CPF: usa **digito verificador** (DV)

- **Método**: "mod 11"
  - Digito é multiplicado por sua posição, indo do menos significativo (peso 2) até o mais significativo
  - Os resultados são somados
  - **DV:** resto da divisão desta soma por 11

Exemplo simplificado

Entrada:	2	3	5	9	2
Posição:	6	5	4	3	2
Multiplicação:	12	15	20	27	4
Soma:	78				
Digito verificador:	78 mod 11 = 1				

Integridade: redundância

Exemplo prático (não-criptográfico):

- RG/CPF: usa **digito verificador** (DV)

- **Método**: "mod 11"
  - Digito é multiplicado por sua posição, indo do menos significativo (peso 2) até o mais significativo
  - Os resultados são somados
  - **DV:** resto da divisão desta soma por 11

Exemplo simplificado

Entrada:	2	3	5	9	2
Posição:	6	5	4	3	2
Multiplicação:	12	15	20	27	4
Soma:	78				
Digito verificador:	78 mod 11 = 1				

Integridade: redundância

Exemplo prático (não-criptográfico):

Digito verificador (DV)

- É interessante do ponto de vista de verificação de integridade contra erros acidentais, como erros de digitação, mas não funciona tanto contra erros propositais, por exemplo:
  - Se o valor do DV for fixo é facilmente possível alterar os valores para conseguir atingir o mesmo DV.

Exemplo simplificado

Entrada:	2	3	5	9	2
Posição:	6	5	4	3	2
Multiplicação:	12	15	20	27	4
Soma:	78				
Digito verificador:	78 mod 11 = 1				

Integridade: redundância

Exemplo prático (não-criptográfico):

Digito verificador (DV)

- É interessante do ponto de vista de verificação de integridade contra erros acidentais, como erros de digitação, mas não funciona tanto contra erros propositais, por exemplo:
  - Se o valor do DV for fixo é facilmente possível alterar os valores para conseguir atingir o mesmo DV.

Exemplo simplificado

Entrada:	2	3	**5** (-1)	9	**2** (+2)
Posição:	6	5	4	3	2
Multiplicação:	12	15	20	27	4
Soma:	78				
Digito verificador:	78 mod 11 = 1				

Integridade: redundância

Exemplo prático (não-criptográfico):

Digito verificador (DV)

- É interessante do ponto de vista de verificação de integridade contra erros acidentais, como erros de digitação, mas não funciona tanto contra erros propositais, por exemplo:
  - Se o valor do DV for fixo é facilmente possível alterar os valores para conseguir atingir o mesmo DV.

Exemplo simplificado

Entrada:	2	3	**4**	9	**4**
Posição:	6	5	4	3	2
Multiplicação:	12	15	**16**	27	**8**
Soma:	78				
Digito verificador:	78 mod 11 = 1				

Integridade: redundância

Exemplo prático (não-criptográfico):

Digito verificador (DV)

- É interessante do ponto de vista de verificação de integridade contra erros acidentais, como erros de digitação, mas não funciona tanto contra erros propositais, por exemplo:
  - Se o valor do DV for fixo é facilmente possível alterar os valores para conseguir atingir o mesmo DV.
  - Não é usado para questão de criptografia porque o atacante tem objetivo claro que é violar a integridade do sistema.

Exemplo simplificado

Entrada:	2	3	**4**	9	**4**
Posição:	6	5	4	3	2
Multiplicação:	12	15	**16**	27	**8**
Soma:	78				
Digito verificador:	78 mod 11 = 1				

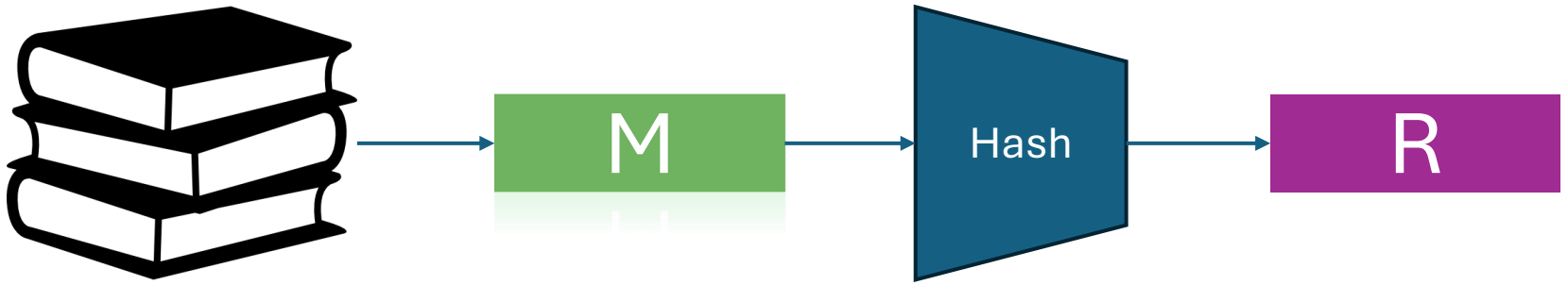
INSTITUTO FEDERAL
São Paulo
Campus Campinas

Funções de Hash

Funções de Hash

Geram **redundâncias** que são anexadas a mensagens com o propósito de detectar alterações: integridade.

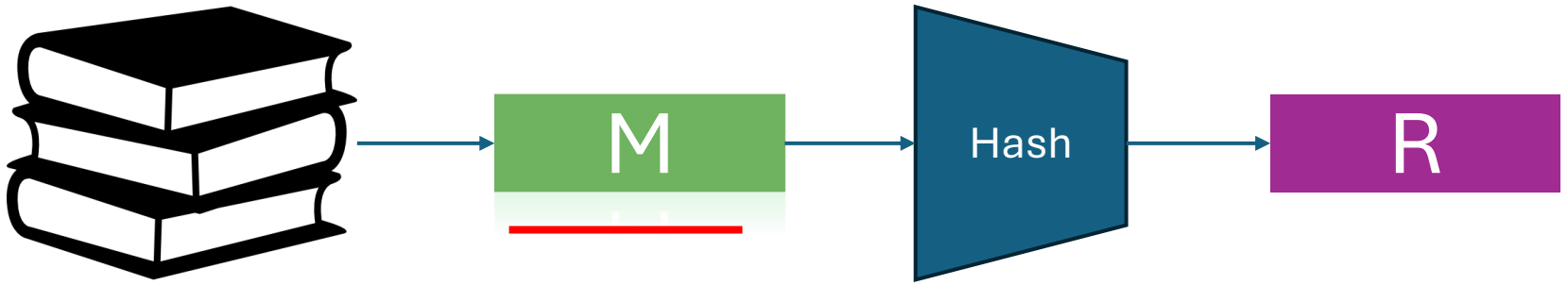
- A redundância é chamada de "***hash***" ou "***resumo criptográfico***" da mensagem.
- O hash tem tamanho fixo e seu valor depende exclusivamente da mensagem – processo **não envolve uma chave secreta**.



Funções de Hash

Geram **redundâncias** que são anexadas a mensagens com o propósito de detectar alterações: integridade.

- A redundância é chamada de "***hash***" ou "***resumo criptográfico***" da mensagem.
- O hash tem tamanho fixo e seu valor depende exclusivamente da mensagem – processo **não envolve uma chave secreta**.



Funções de Hash

Geram **redundâncias** que são anexadas a mensagens com o propósito de detectar alterações: integridade.

- A redundância é chamada de "***hash***" ou "***resumo criptográfico***" da mensagem.
- O hash tem tamanho fixo e seu valor depende exclusivamente da mensagem – processo **não envolve uma chave secreta**.

M

Tamanho de entrada variada

Hash

R

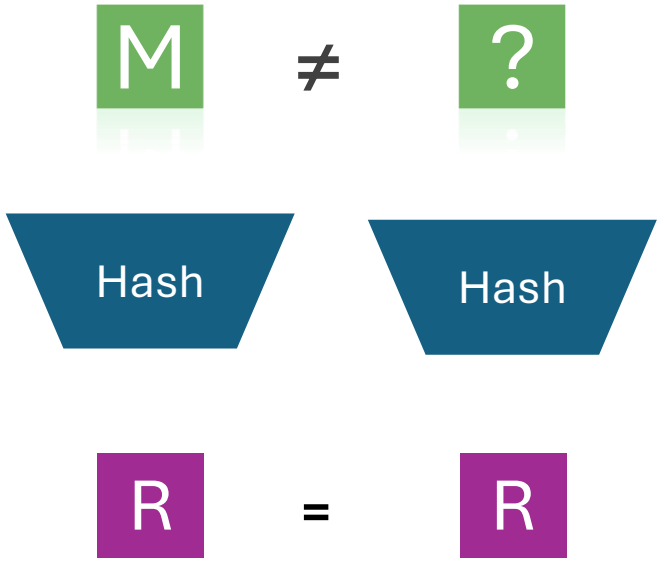
Tamanho Fixo

Propriedades fundamentais

Resistência a 1ª inversão

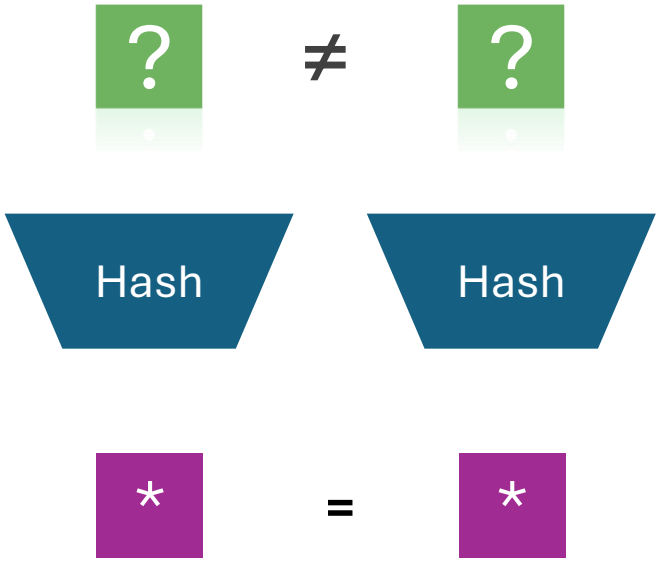
2^n

Resistência a 2ª inversão



2^n

Resistência a colisões



$2^{n/2}$

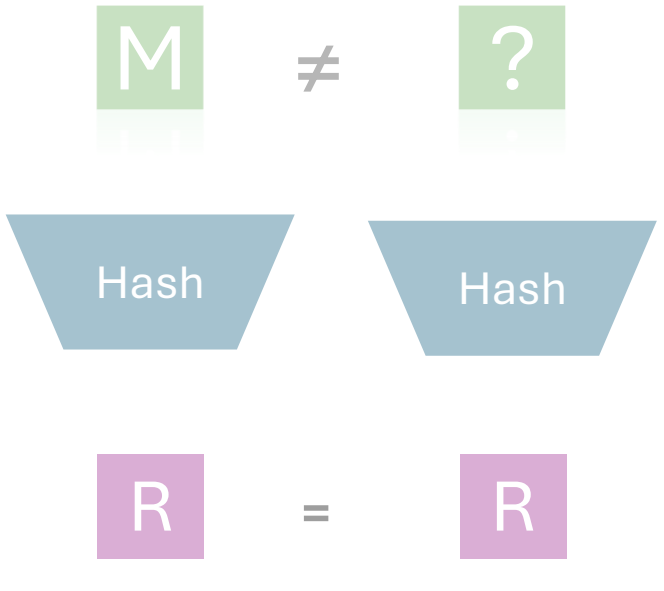
Custo do ataque

Propriedades fundamentais

Resistência a 1ª inversão

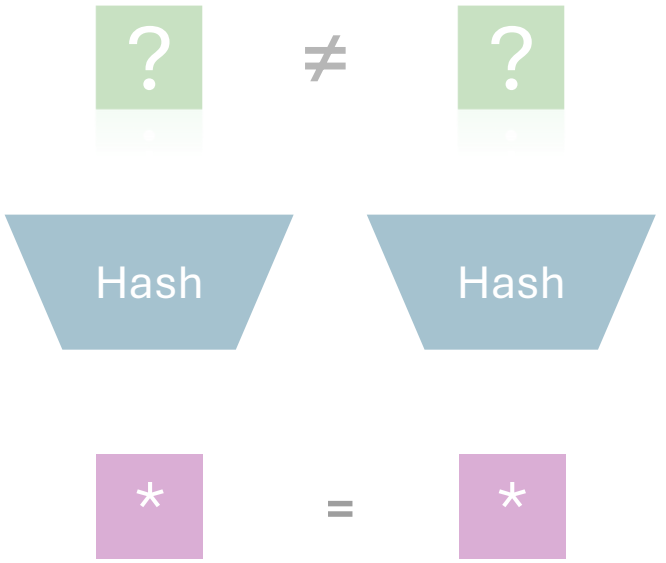
2^n

Resistência a 2ª inversão



2^n

Resistência a colisões

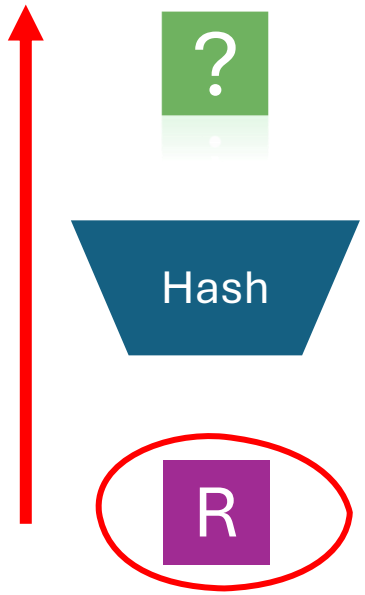


$2^{n/2}$

Custo do ataque

Propriedades fundamentais

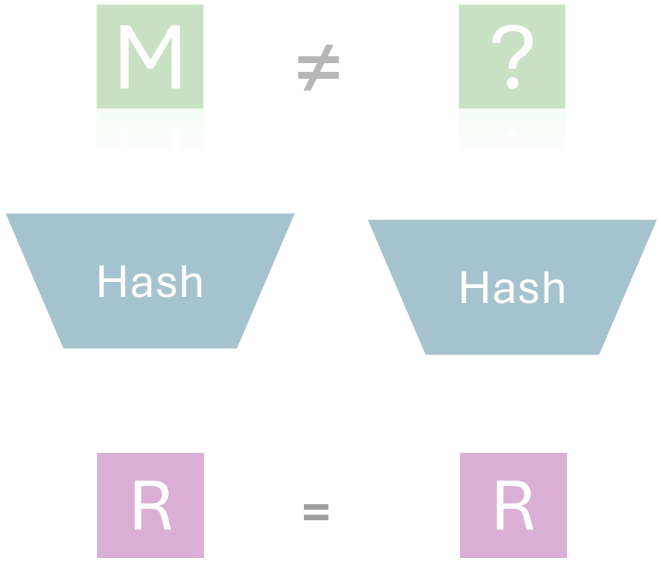
Resistência a 1ª inversão



Testes: 2^n

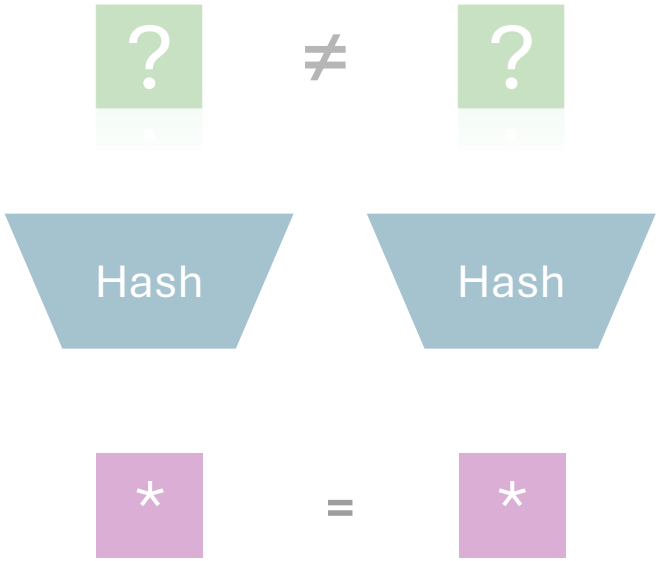
Obs.: Se um R (hash) for fornecido é difícil inverter para descobrir a mensagem que deu origem ao hash.

Resistência a 2ª inversão



2^n

Resistência a colisões

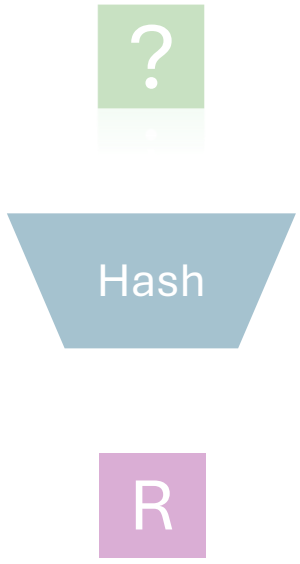


$2^{n/2}$

Custo do ataque

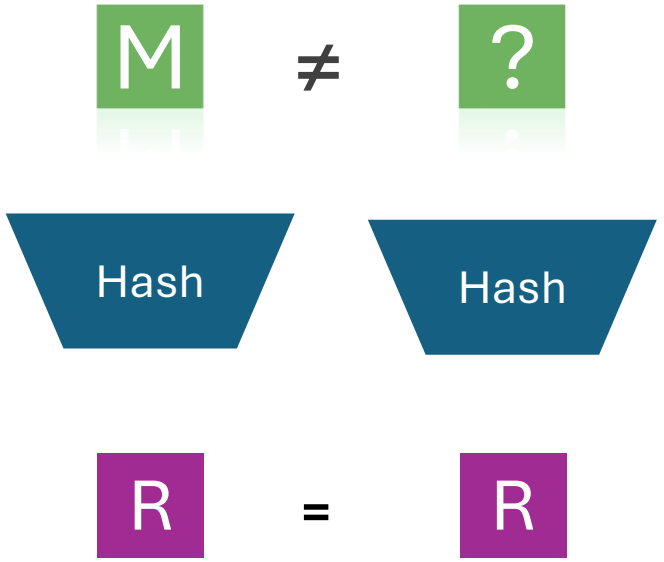
Propriedades fundamentais

Resistência a 1ª inversão



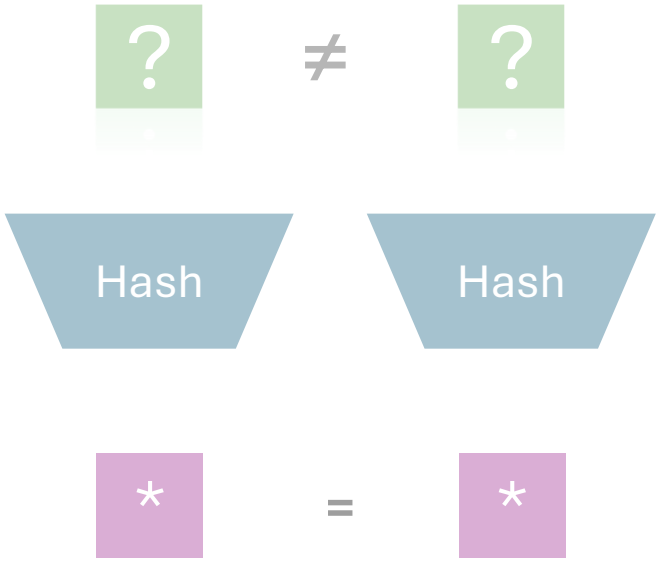
2^n

Resistência a 2ª inversão



2^n

Resistência a colisões

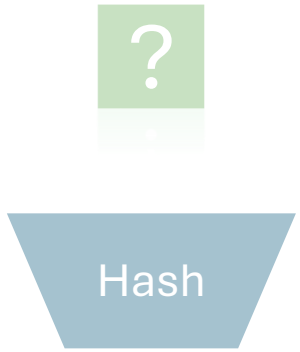


$2^{n/2}$

Custo do ataque

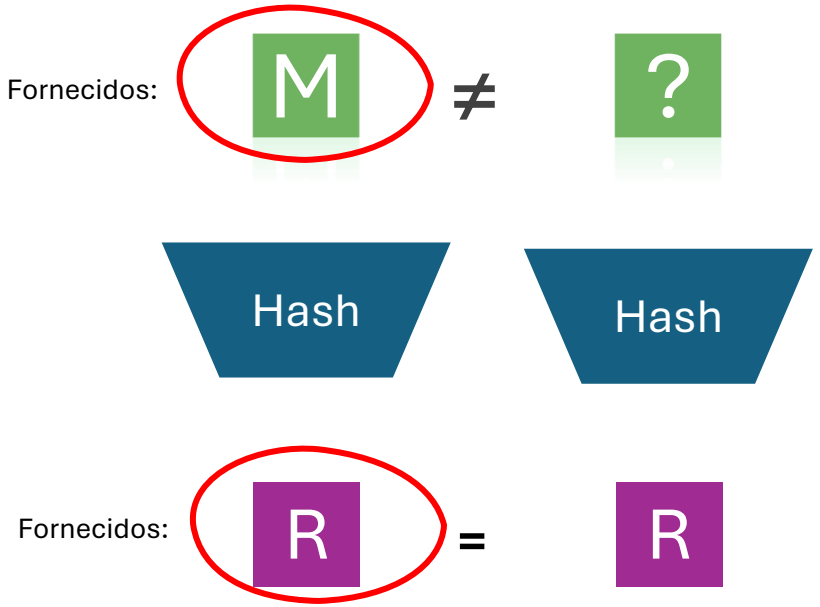
Propriedades fundamentais

Resistência a 1ª inversão



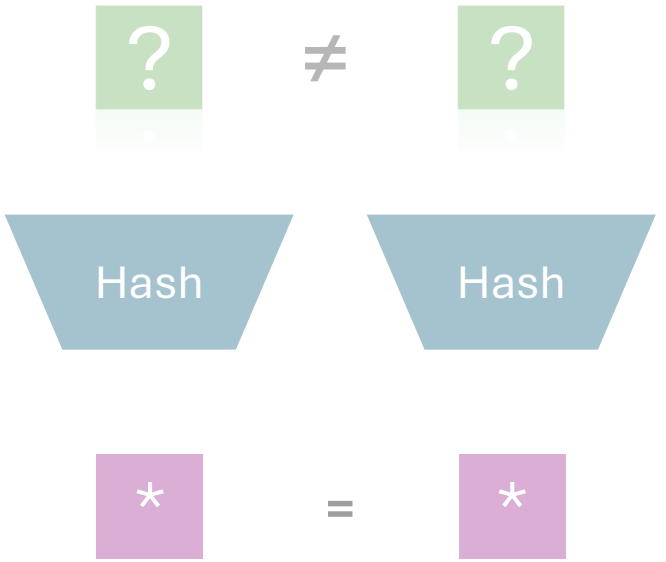
2^n

Resistência a 2ª inversão



2^n

Resistência a colisões

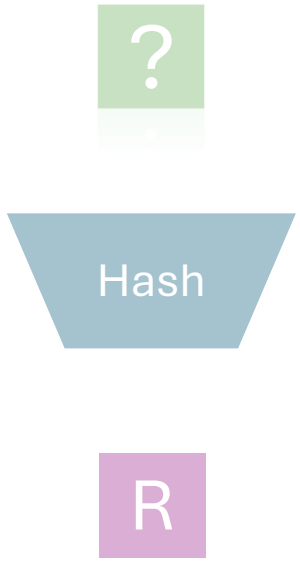


$2^{n/2}$

Custo do ataque

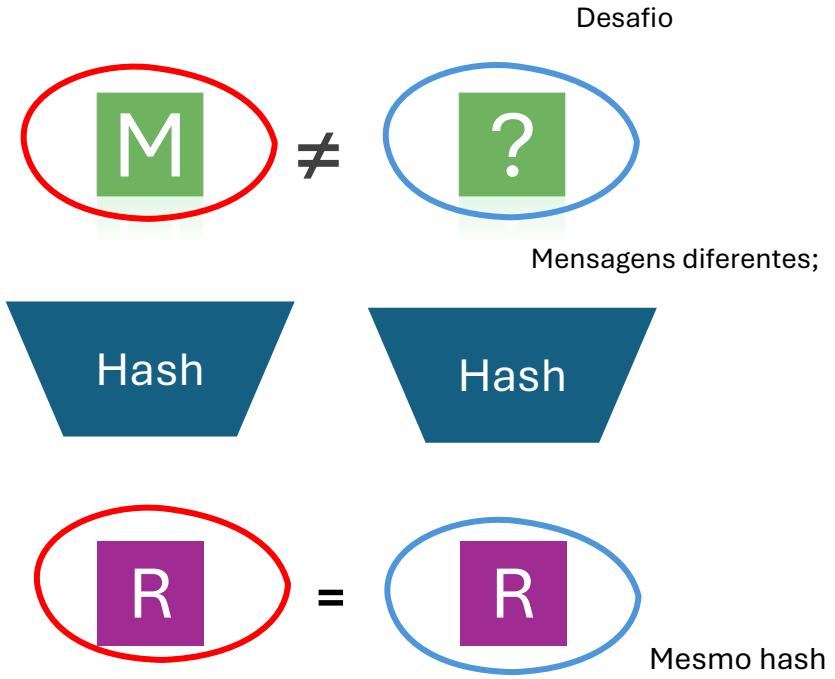
Propriedades fundamentais

Resistência a 1ª inversão



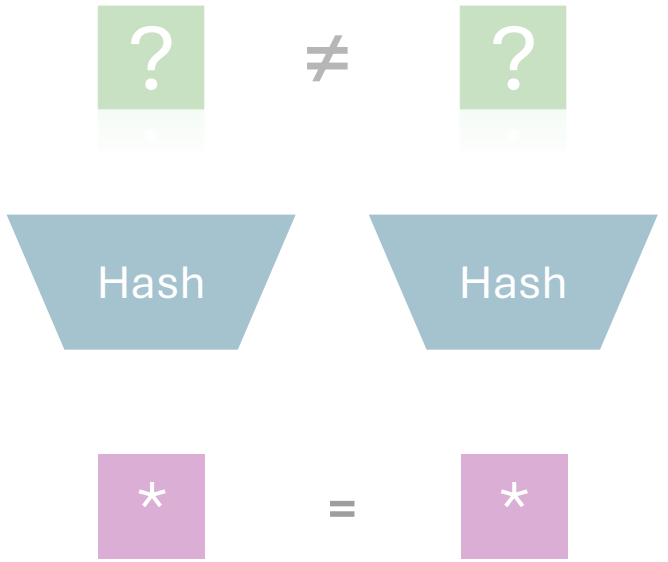
2^n

Resistência a 2ª inversão



2^n

Resistência a colisões

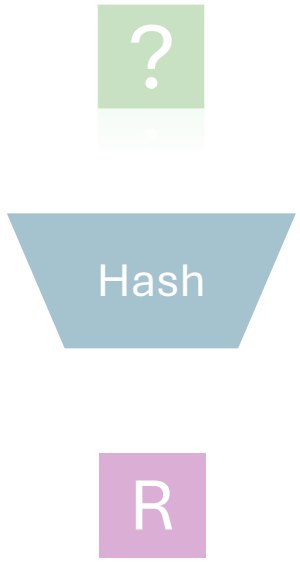


$2^{n/2}$

Custo do ataque

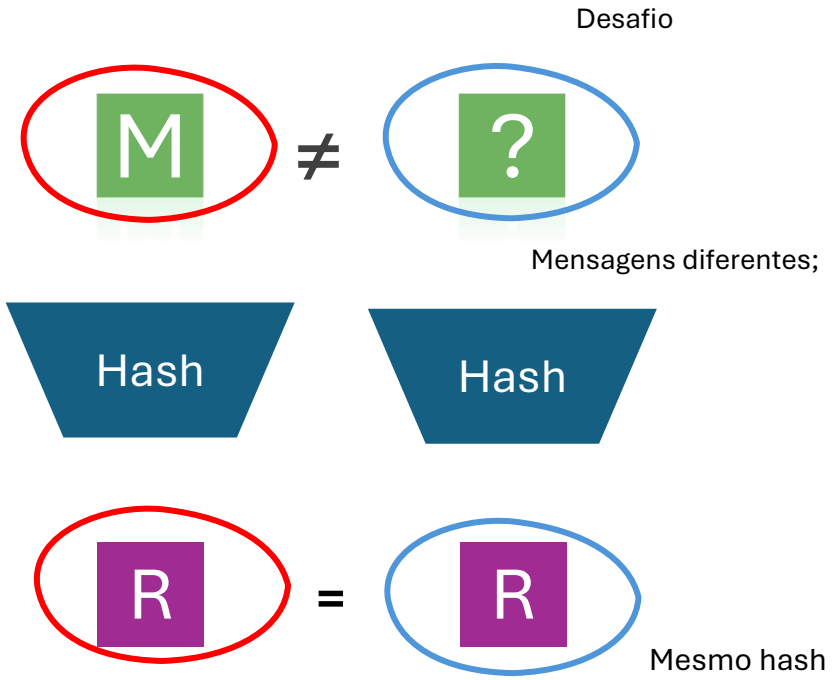
Propriedades fundamentais

Resistência a 1ª inversão



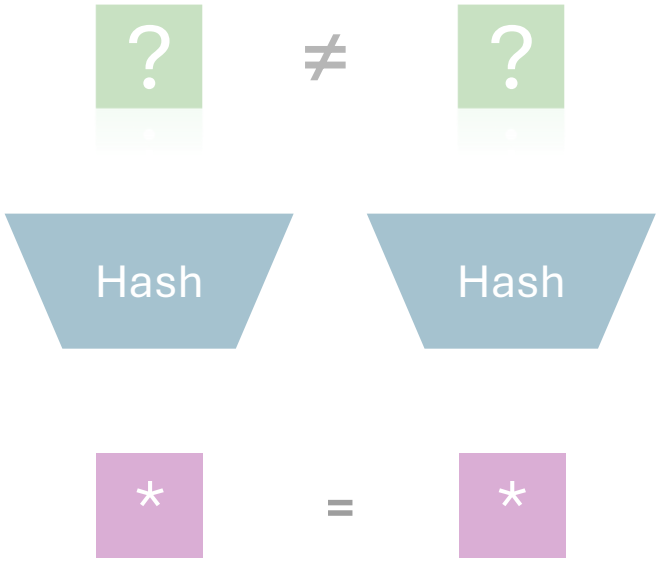
2^n

Resistência a 2ª inversão



Testes: 2^n
(custo alto)

Resistência a colisões



$2^{n/2}$

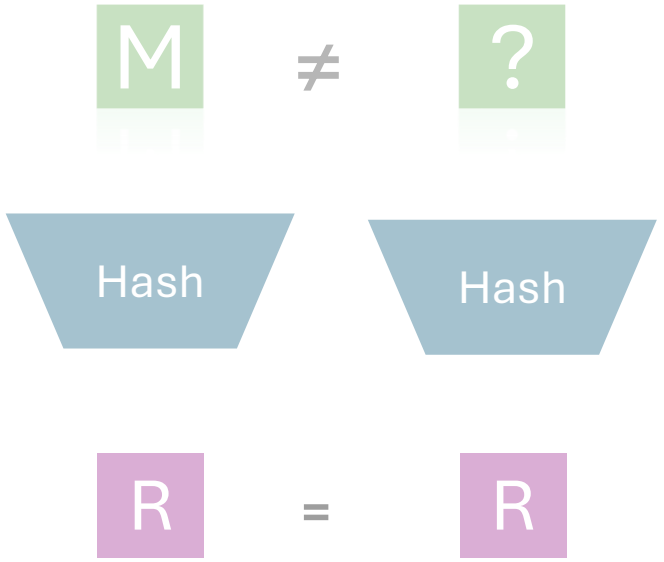
Custo do ataque

Propriedades fundamentais

Resistência a 1ª inversão

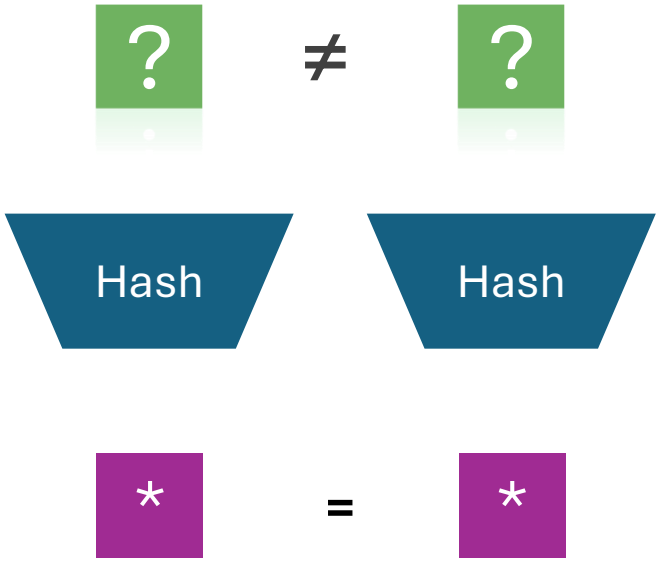
2^n

Resistência a 2ª inversão



2^n

Resistência a colisões



$2^{n/2}$

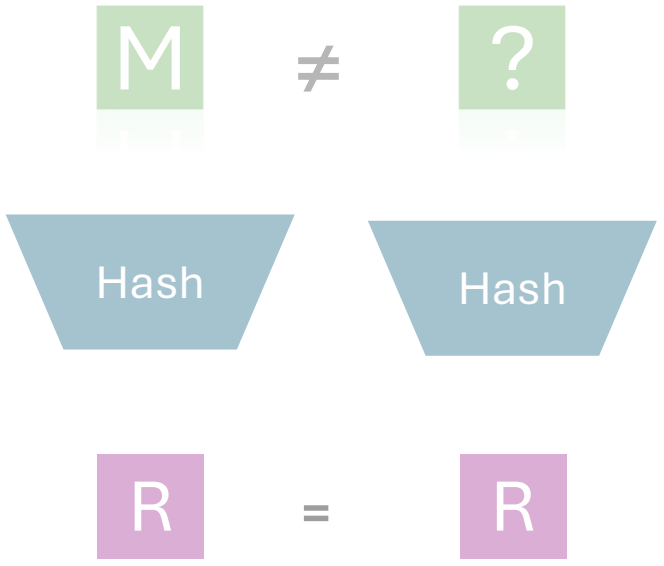
Custo do ataque

Propriedades fundamentais

Resistência a 1ª inversão

2^n

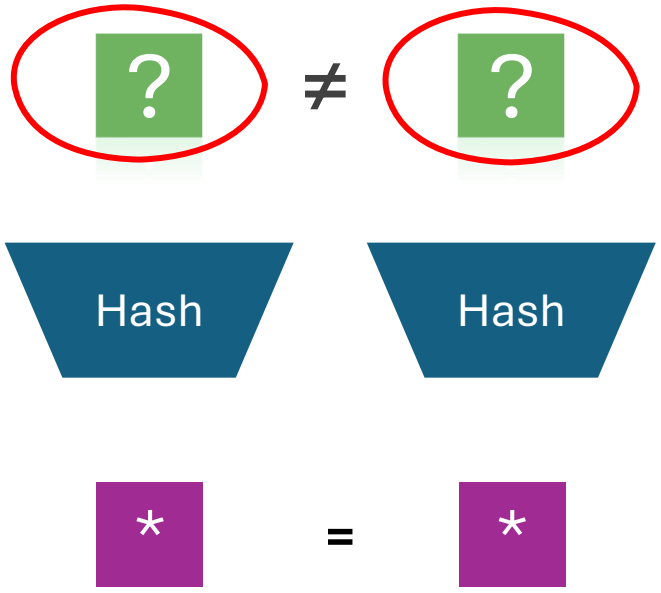
Resistência a 2ª inversão



2^n

Resistência a colisões

Mensagens completamente diferentes

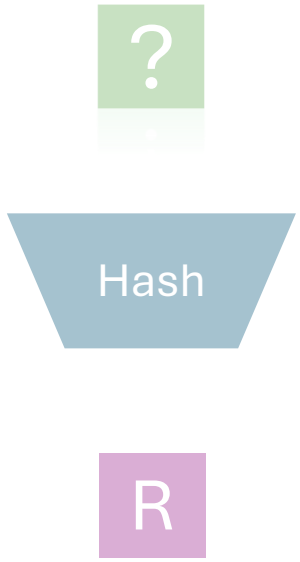


$2^{n/2}$

Custo do ataque

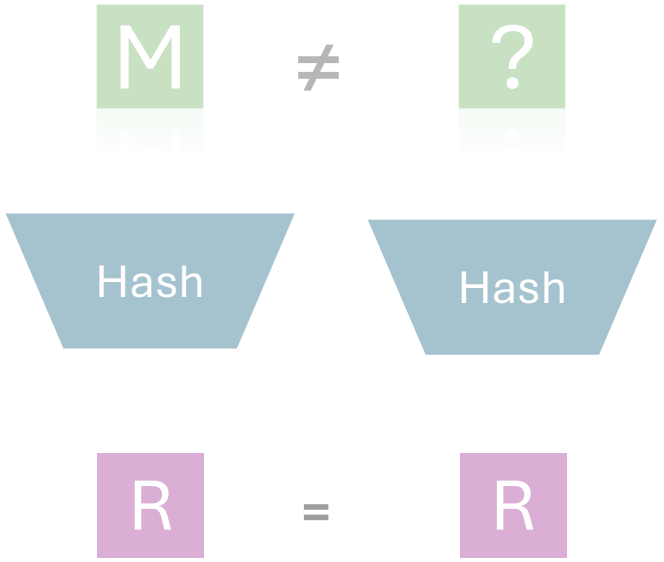
Propriedades fundamentais

Resistência a 1ª inversão



2^n

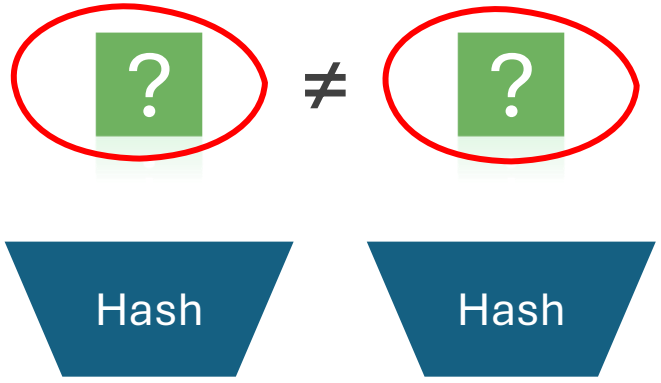
Resistência a 2ª inversão



2^n

Resistência a colisões

Mensagens completamente diferentes



Hashes completamente iguais

$2^{n/2}$

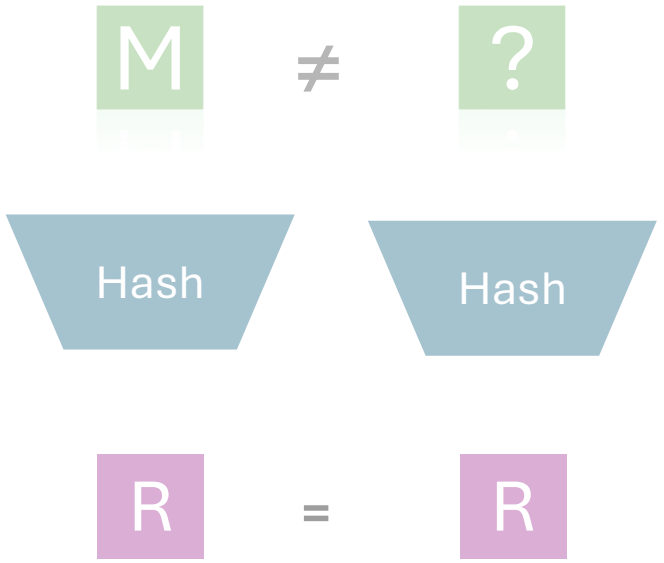
Custo do ataque

Propriedades fundamentais

Resistência a 1ª inversão

2^n

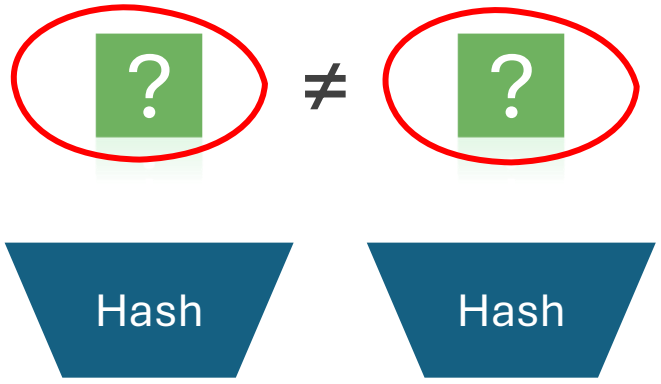
Resistência a 2ª inversão



2^n

Resistência a colisões

Mensagens completamente diferentes



Hashes completamente iguais

Testes: $2^{n/2}$

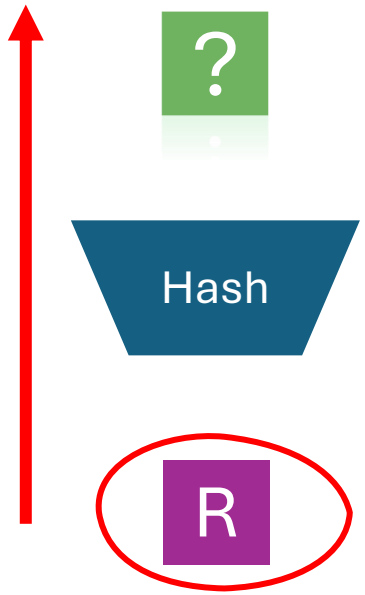
Retirado um pouco da segurança

Custo do ataque

Propriedades fundamentais

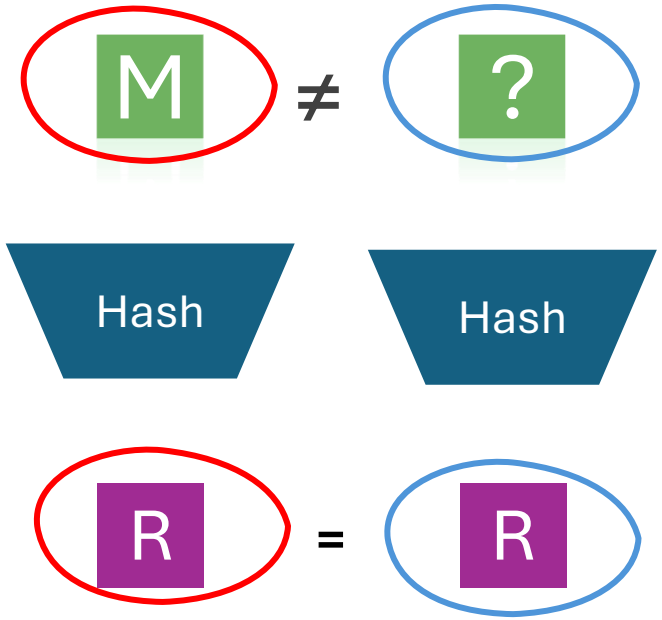
Importantes para as Assinaturas Digitais

Resistência a 1ª inversão



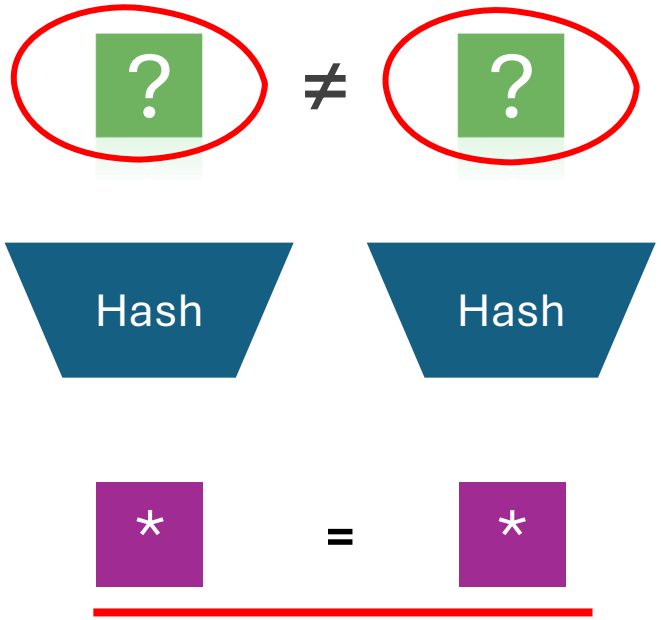
Testes: 2^n

Resistência a 2ª inversão



Testes: 2^n

Resistência a colisões



Testes: $2^{n/2}$

Custo do ataque

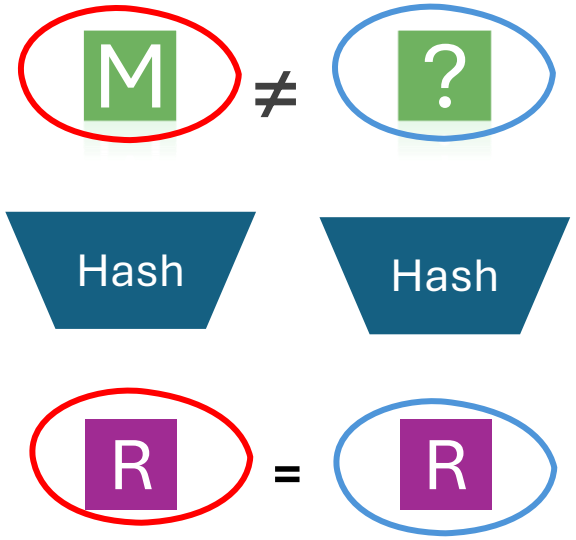
Propriedades fundamentais

Obs.:

As <u>assinaturas</u> digitais são calculadas não em cima da mensagem mas em cima dos hashes da mensagem.

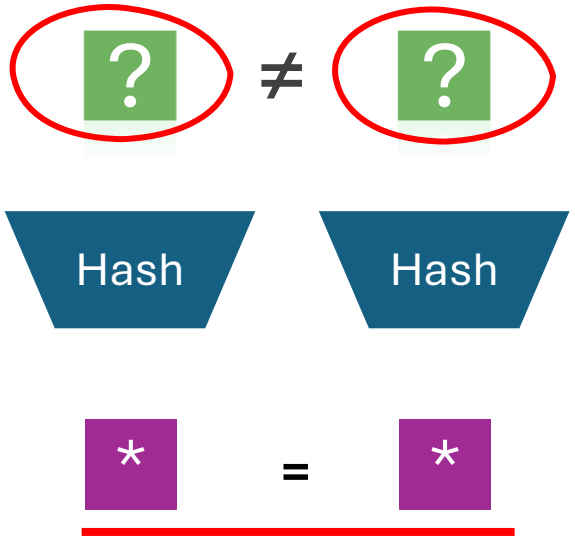
Portanto, se é fácil achar duas mensagens distintas com o mesmo hash, então é fácil achar duas mensagens diferentes em que ao assinar uma, a assinatura é válida para a outra.

Resistência a 2ª inversão



Testes: 2^n

Resistência a colisões



Testes: $2^{n/2}$

Custo do ataque

Integridade: funções de Hash

- Família MD:
  - MD2, MD4 e MD5: hashes de 128 bbits
    - Completamente quebrada (Wang et al., 2004);

- Família SHA
  - SHA-0: hashes com 160 bits
    - Não recomendado: colisão em 2^{39} passos x 2^{80} projetado.

  - SHA-1: hashes com 160 bits
    - Não recomendado: desde 2010, paa assinaturas.
    - Segurança: colisões em 2^{39} passos x 2^{80} projetado
  - SHA-2: hashes com 224, 256, 384 ou 512 bits
    - Paliativo atual: algoritmo baseado no SHA-1, mas hash grande dificulta ataque
  - SHA-3: hashes com 224, 256, 384 ou 512 bits
    - Concurso público finalizado em 2012: Keccak

Integridade: funções de Hash

- Família MD:
  - MD2, MD4 e MD5: hashes de 128 bbits
    - Completamente quebrada (Wang et al., 2004);

- Família SHA
  - SHA-0: hashes com 160 bits
    - Não recomendado: colisão em 2^{39} passos x 2^{80} projetado.
  - SHA-1: hashes com 160 bits
    - Não recomendado: desde 2010, paa assinaturas.
    - Segurança: colisões em 2^{39} passos x 2^{80} projetado
  - **SHA-2**: hashes com 224, 256, 384 ou 512 bits
    - Paliativo atual: algoritmo baseado no SHA-1, mas hash grande dificulta ataque
  - **SHA-3**: hashes com 224, 256, 384 ou 512 bits
    - Concurso público finalizado em 2012: Keccak

- **Não são usados na prática;**
- **Problemas na resistência a colisão**

Integridade: funções de Hash

- Família MD:
  - MD2, MD4 e MD5: hashes de 128 bbits
    - Completamente quebrada (Wang et al., 2004);

- Família SHA
  - SHA-0: hashes com 160 bits
    - Não recomendado: colisão em 2^{39} passos x 2^{80} projetado.
  - SHA-1: hashes com 160 bits
    - Não recomendado: desde 2010, paa assinaturas.
    - Segurança: colisões em 2^{39} passos x 2^{80} projetado

- **SHA-2**: hashes com 224, 256, 384 ou 512 bits
  - Paliativo atual: algoritmo baseado no SHA-1, mas hash grande dificulta ataque

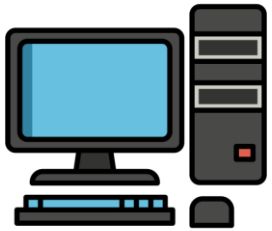
- **SHA-3**: hashes com 224, 256, 384 ou 512 bits
  - Concurso público finalizado em 2012: Keccak

Uso prático: download de arquivos

Download de aplicativos (via web, P2P...)

- Realizar em geral a verificação do hash para evitar receber um arquivo corrompido ou com malware (e.g., Torrent, Stremio).
- Os sistemas fazem verificação se os dados que está sendo recebido estão íntegros e sem nenhum tipo de modificação.

OpenSSL

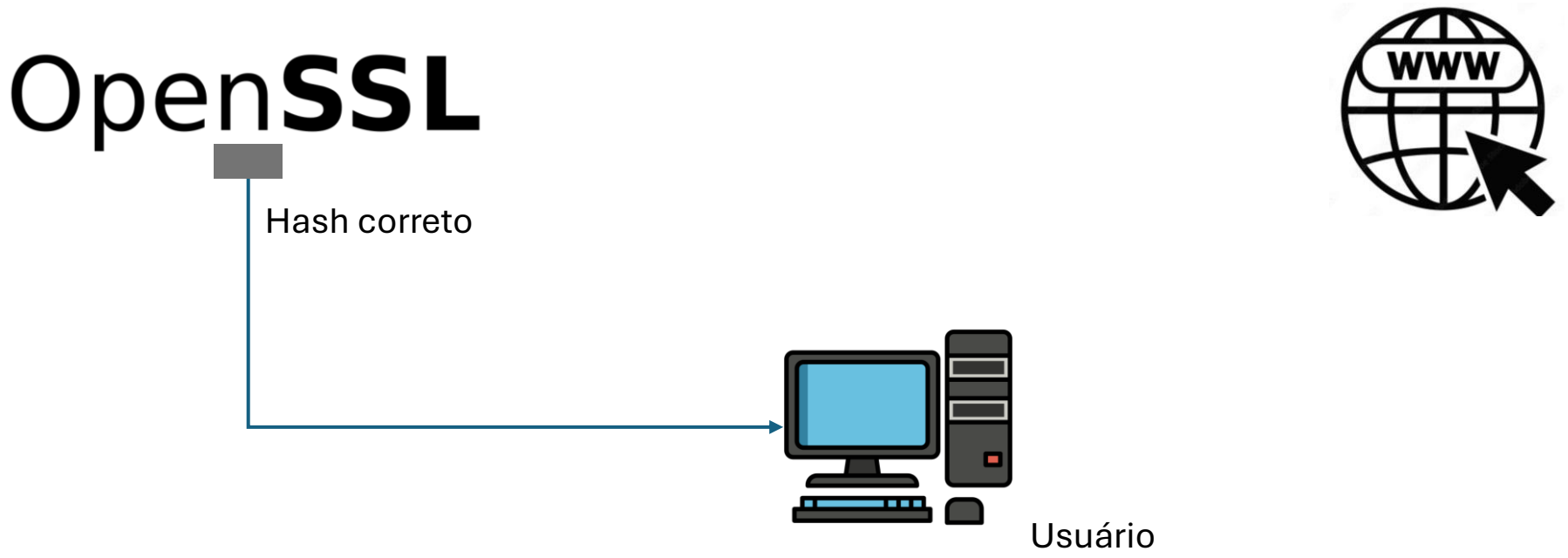


Usuário

Uso prático: download de arquivos

Download de aplicativos (via web, P2P...)

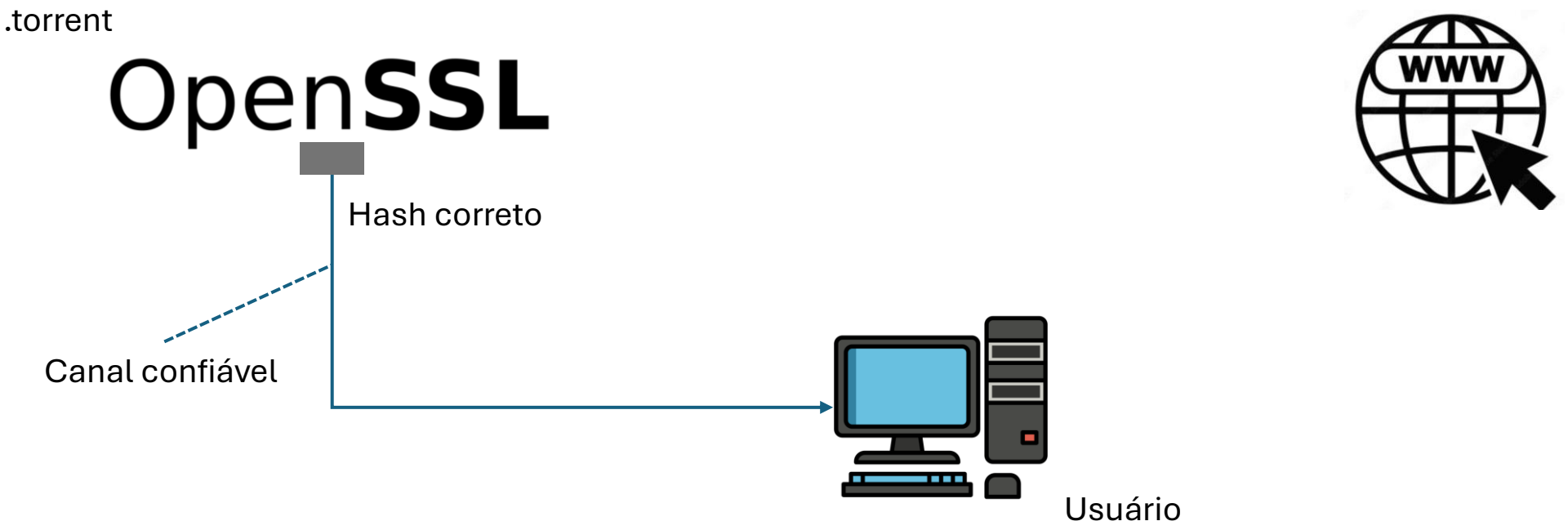
- Obs.: hash deve ser entregue via canal confiável, ou atacante, pode também modificar seu valor, fazendo com que a verificação não indique erro.



Uso prático: download de arquivos

Download de aplicativos (via web, P2P...)

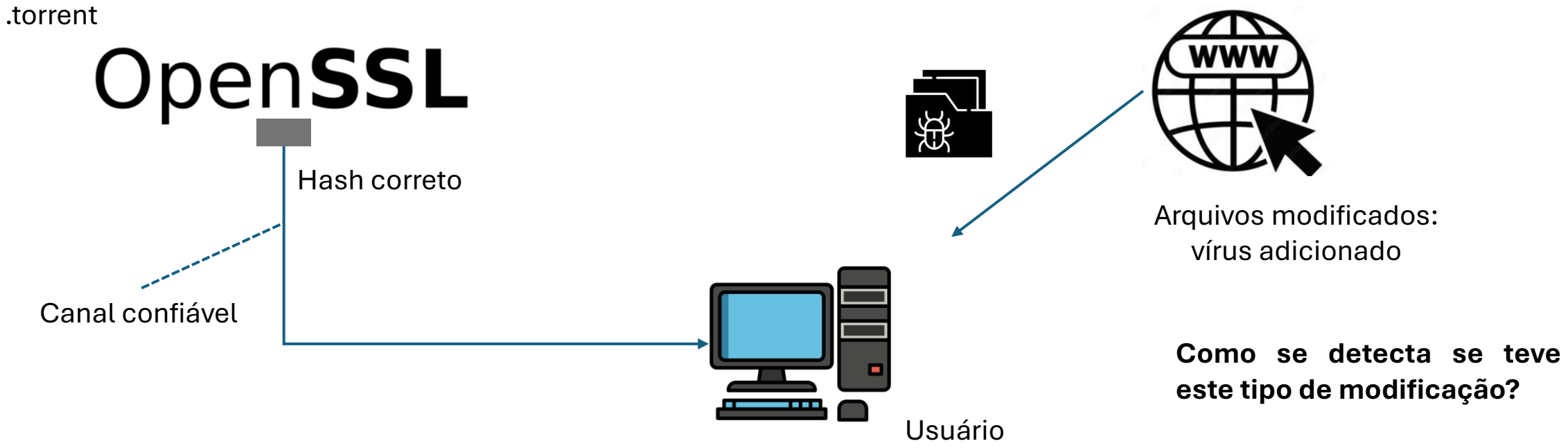
- Obs.: hash deve ser entregue via canal confiável, ou atacante, pode também modificar seu valor, fazendo com que a verificação não indique erro.



Uso prático: download de arquivos

Download de aplicativos (via web, P2P...)

- Obs.: hash deve ser entregue via canal confiável, ou atacante, pode também modificar seu valor, fazendo com que a verificação não indique erro.

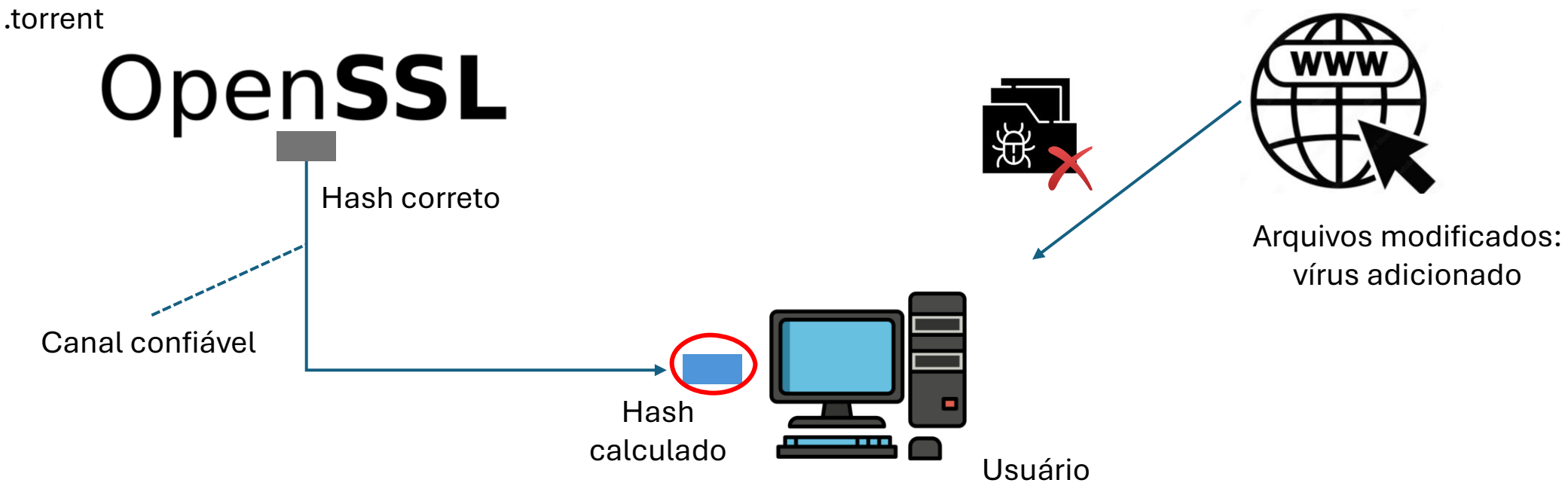


Como se detecta se teve este tipo de modificação?

Uso prático: download de arquivos

Download de aplicativos (via web, P2P...)

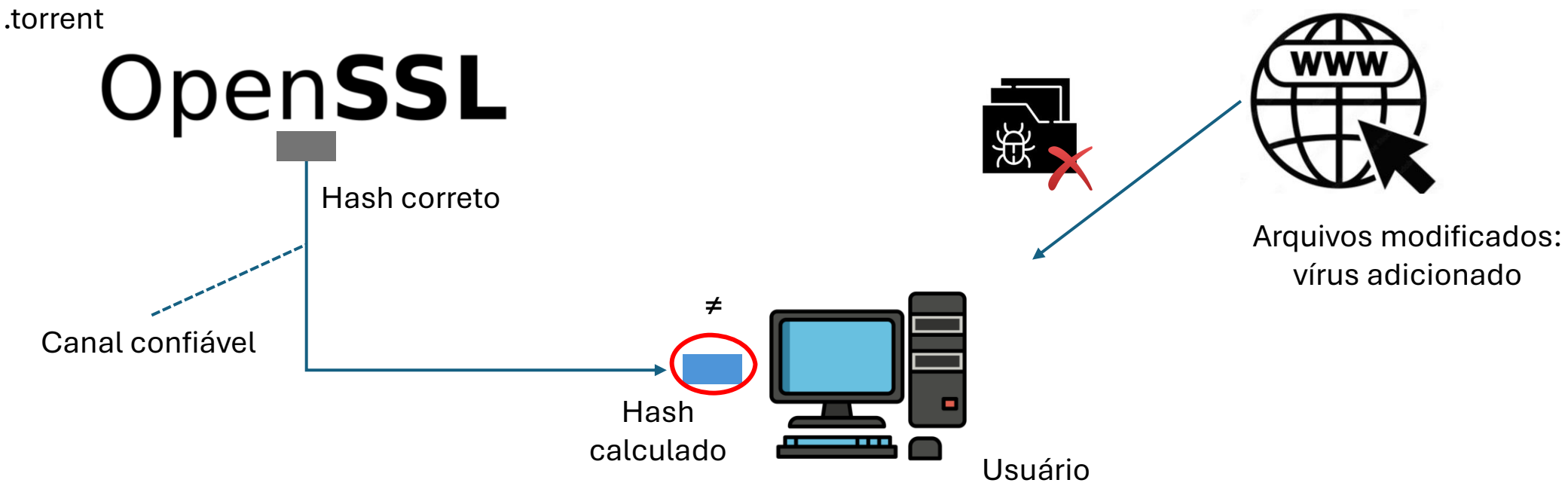
- Obs.: hash deve ser entregue via canal confiável, ou atacante, pode também modificar seu valor, fazendo com que a verificação não indique erro.



Uso prático: download de arquivos

Download de aplicativos (via web, P2P...)

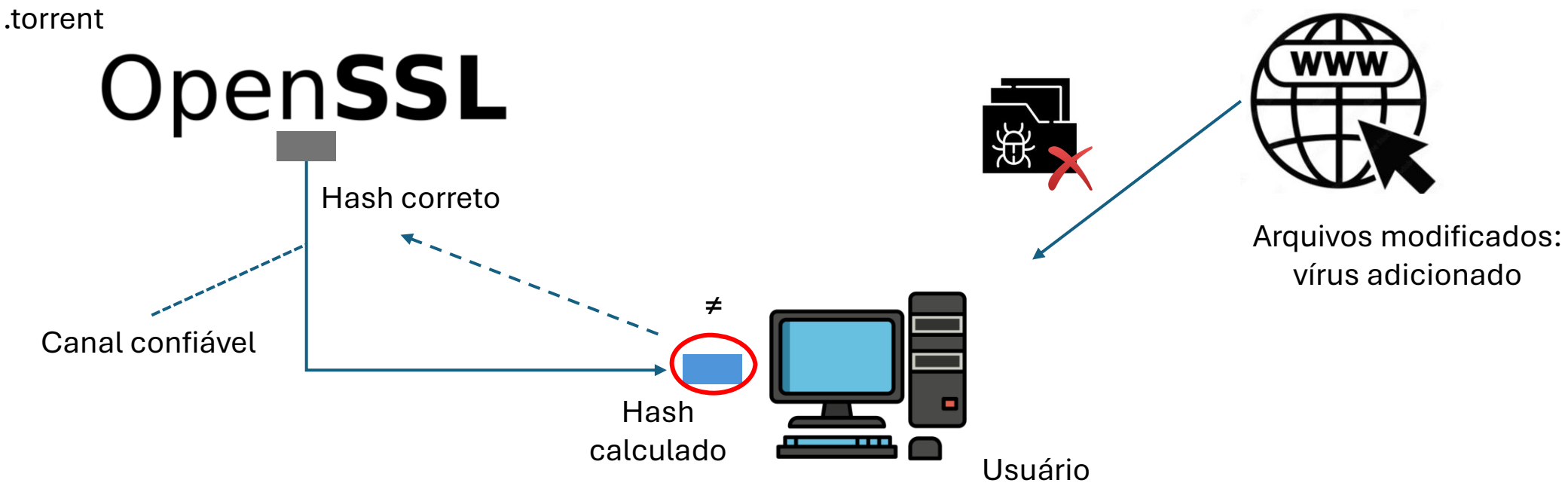
- Obs.: hash deve ser entregue via canal confiável, ou atacante, pode também modificar seu valor, fazendo com que a verificação não indique erro.



Uso prático: download de arquivos

Download de aplicativos (via web, P2P...)

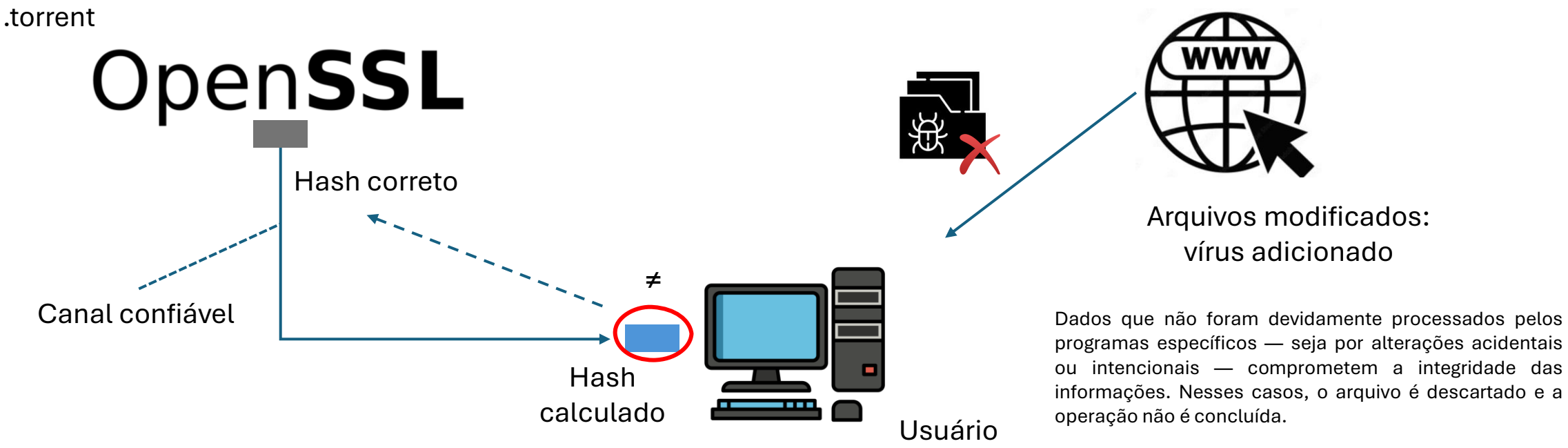
- Obs.: hash deve ser entregue via canal confiável, ou atacante, pode também modificar seu valor, fazendo com que a verificação não indique erro.



Uso prático: download de arquivos

Download de aplicativos (via web, P2P...)

- Obs.: hash deve ser entregue via canal confiável, ou atacante, pode também modificar seu valor, fazendo com que a verificação não indique erro.



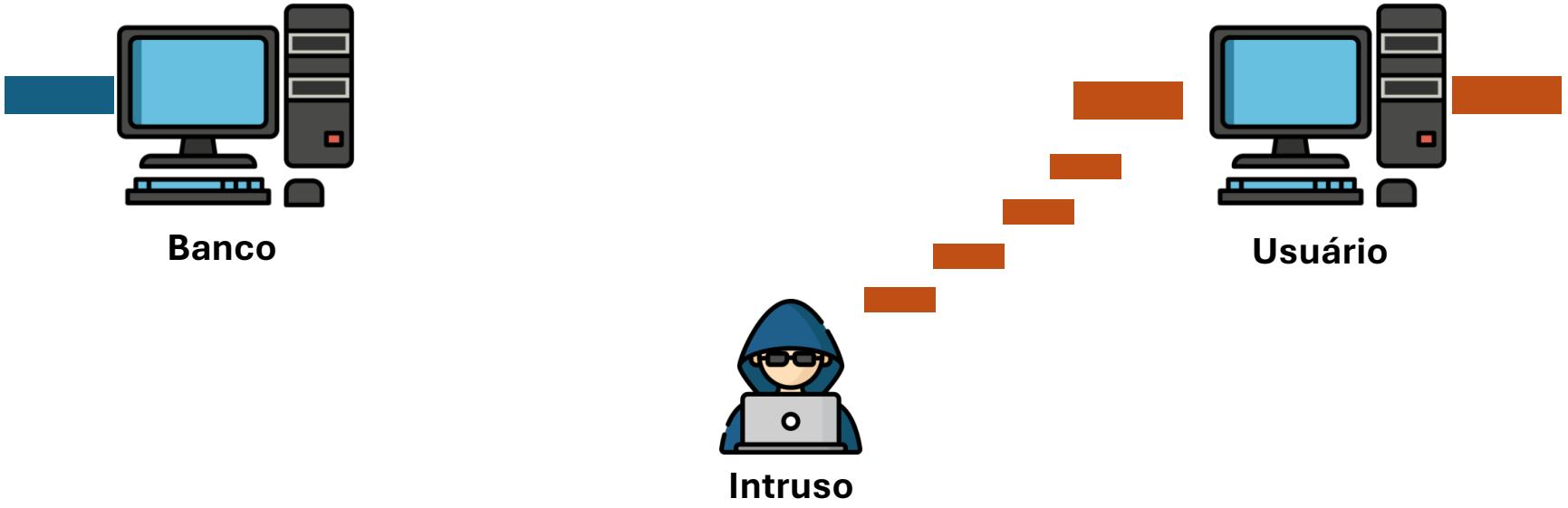
INSTITUTO FEDERAL
São Paulo
Campus Campinas

Códigos de autenticação

Autenticidade

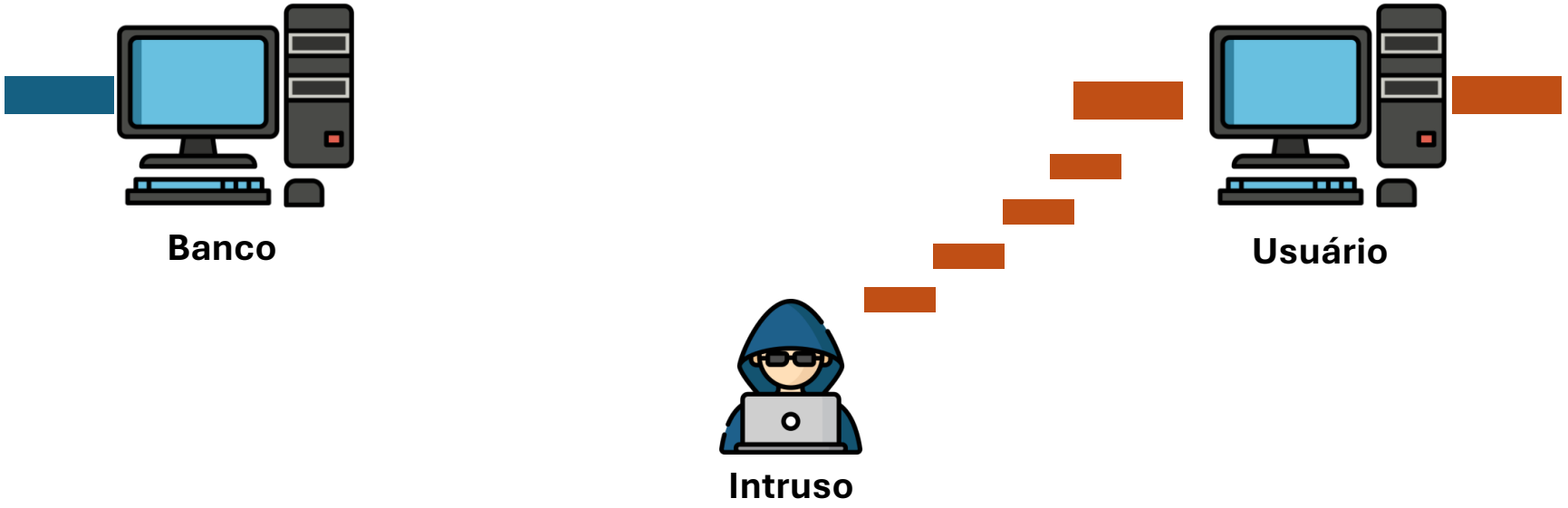
Serviço necessário

- Capacidade do receptor verificar quem é o emissor da mensagem

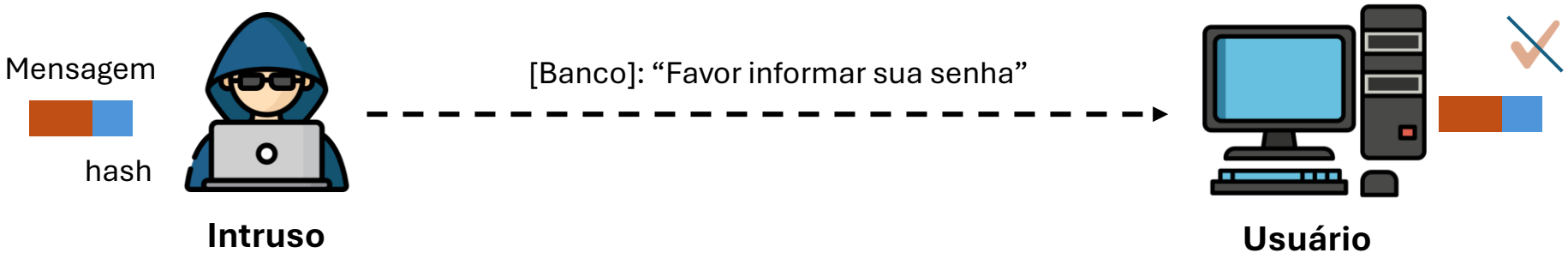


Autenticidade

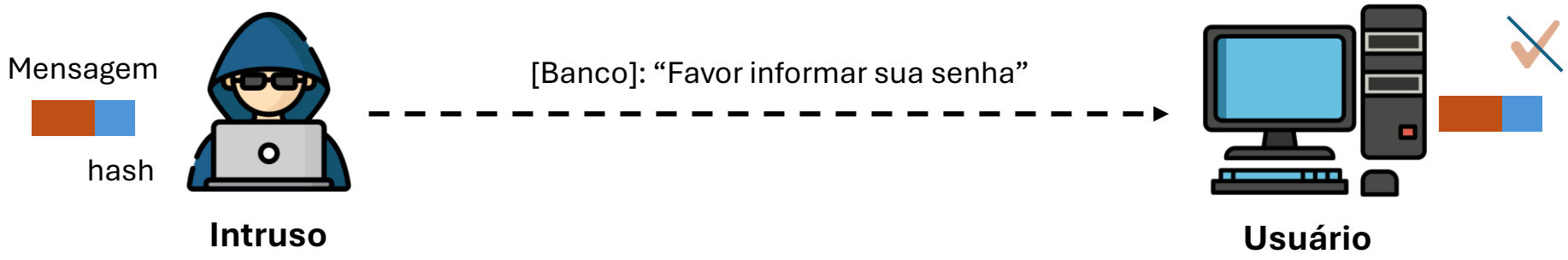
Como resolver?



Usar hash?


Mensagem
hash
[Banco]: “Favor informar sua senha”
Intruso
Usuário

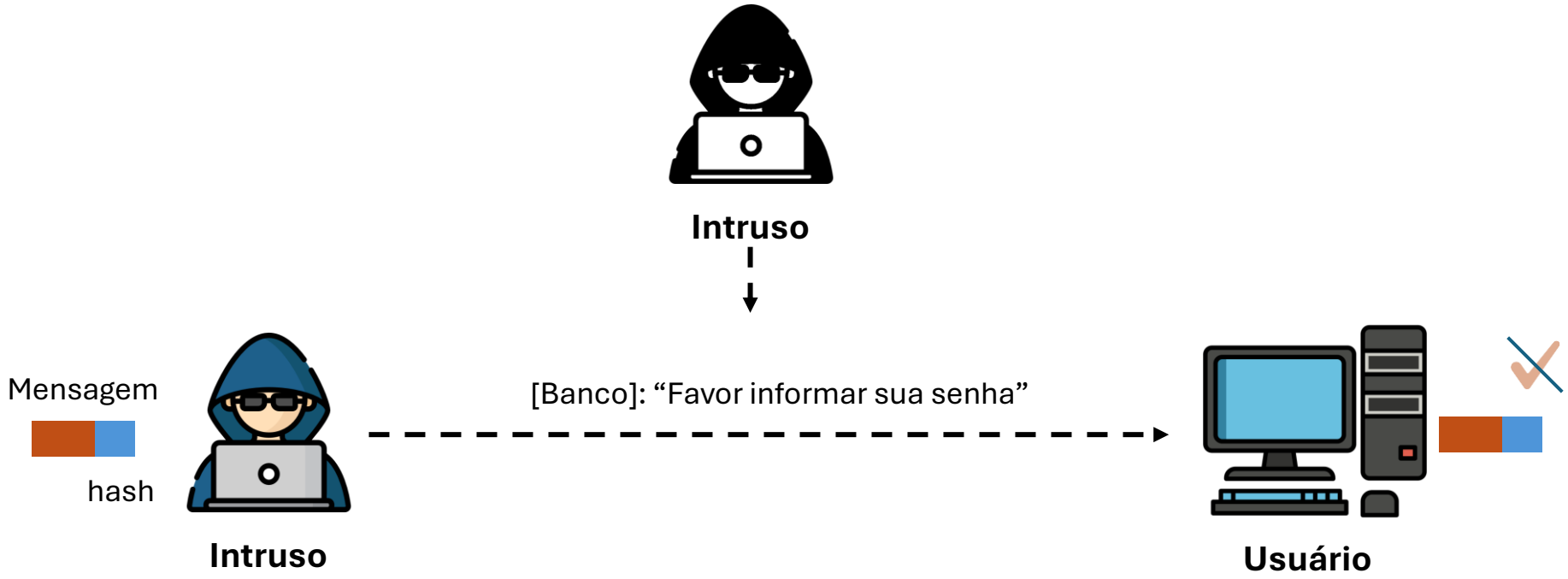
Usar hash?



Infelizmente não...

Como o cálculo do hash não depende de informação secreta, qualquer pessoa — inclusive um intruso — pode gerá-lo e enviar a mensagem adulterada. Assim, não há garantia de autenticidade.

Usar hash?



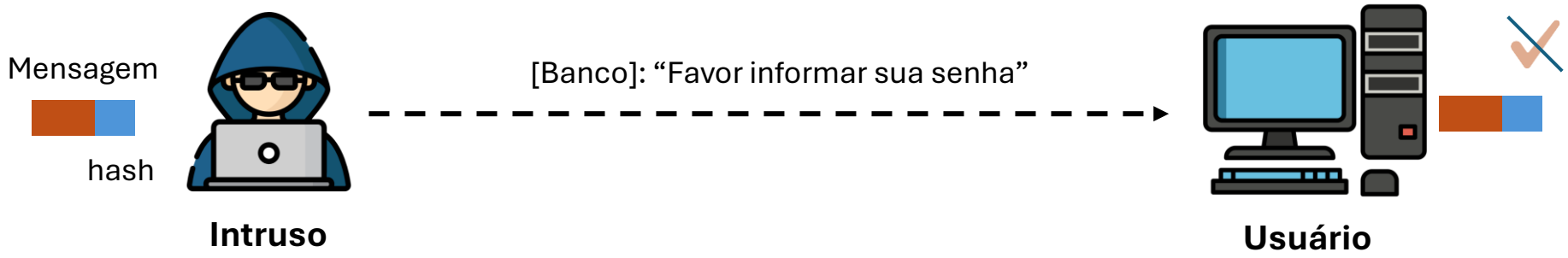
Infelizmente não...

Nesse cenário, o uso apenas do hash não garante a integridade da mensagem, pois outro intruso pode interceptá-la, modificar seu conteúdo, recalcular o hash e reenviá-la adulterada.

Usar hash?

Hash sozinho não funciona...

- Qualquer pessoa, incluindo intruso, pode calcular o hash da mensagem falsa.
- O fato da mensagem estar integra não significa que foi o banco quem a enviou.



INSTITUTO FEDERAL
São Paulo
Campus Campinas

Estratégia

Problema: hash pode ser calculado por qualquer pessoa.

- E se eu fizer um hash que pode ser calculado somente por alguns usuários?

INSTITUTO FEDERAL
São Paulo
Campus Campinas

Estratégia

Problema: hash pode ser calculado por qualquer pessoa.

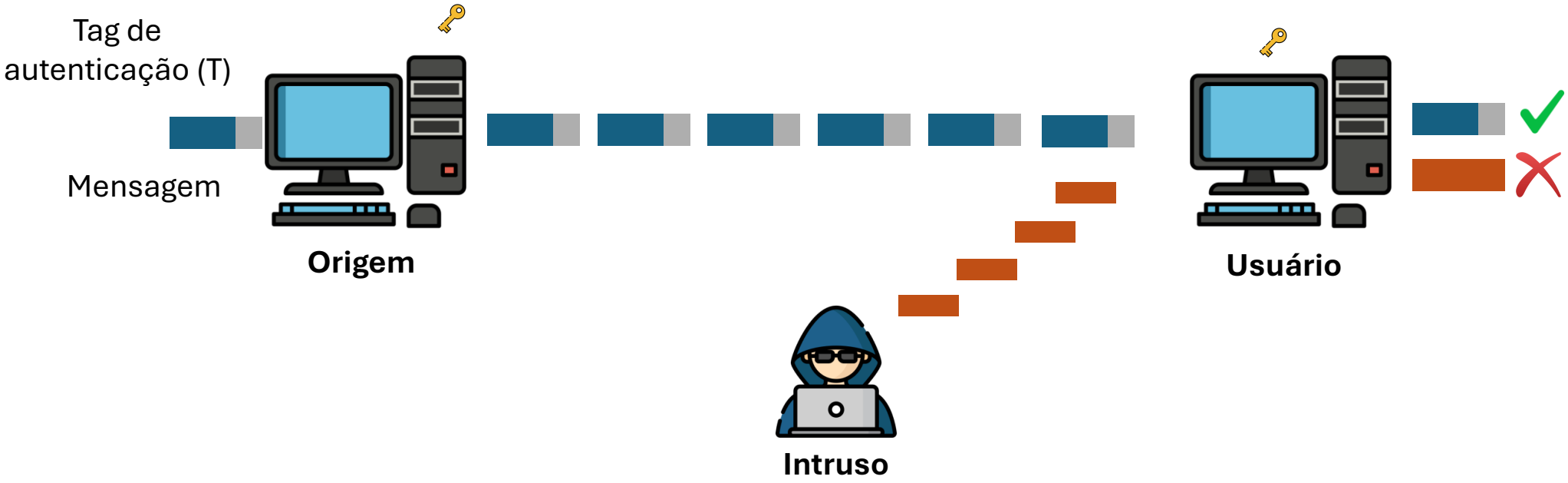
- E se eu fizer um hash que pode ser calculado somente por alguns usuários?

....conceito: autenticação de mensagens

Estratégia

Usar **redundância dependente de chave**

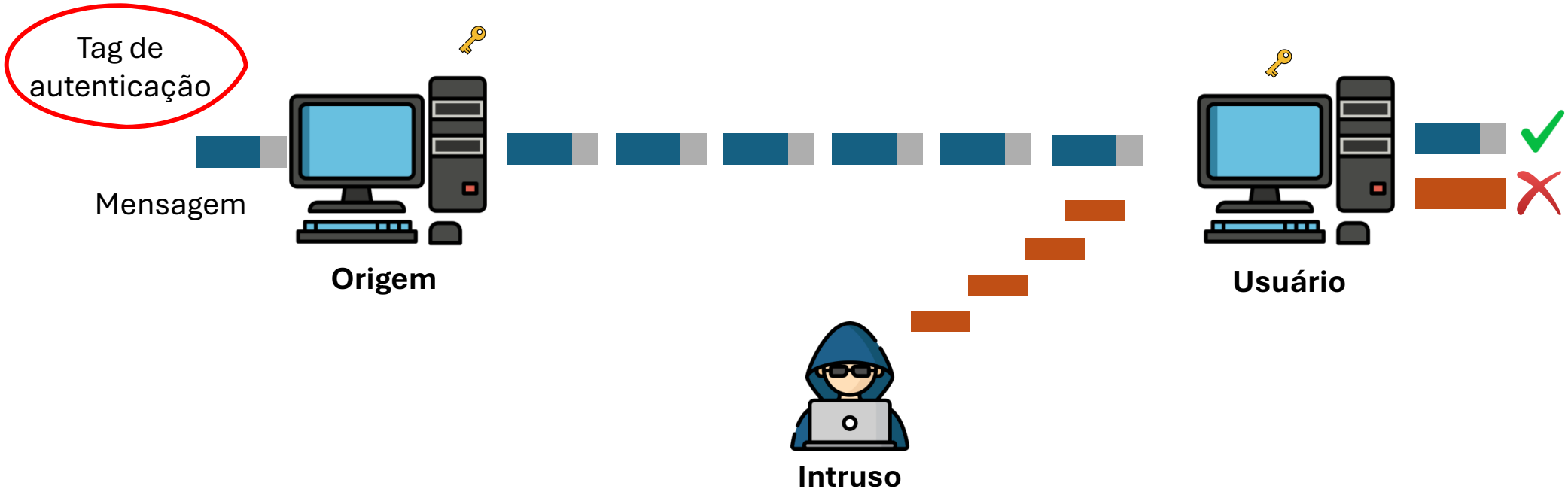
- **Apenas a origem e destino conhecem a chave** e conseguem calcular a redundância corretamente: autenticidade.
- Também garante **integridade** – alteração na mensagem detectada, como no caso das funções de hash



Estratégia

Usar **redundância dependente de chave**

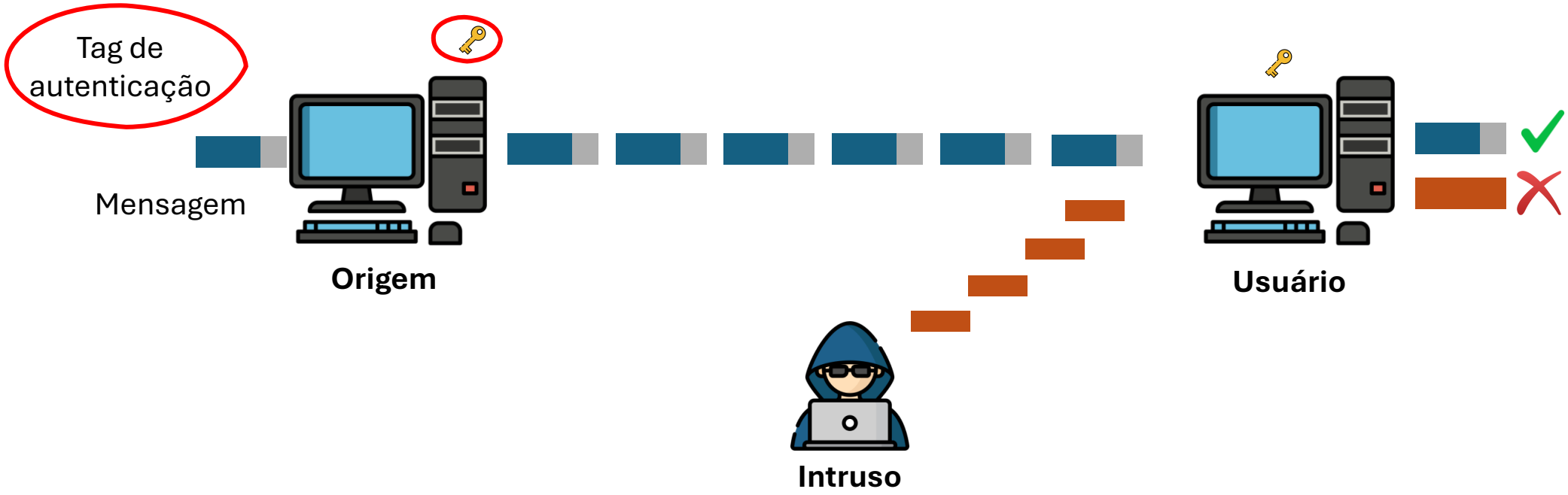
- **Apenas a origem e destino conhecem a chave** e conseguem calcular a redundância corretamente: autenticidade.
- Também garante **integridade** – alteração na mensagem detectada, como no caso das funções de hash



Estratégia

Usar **redundância dependente de chave**

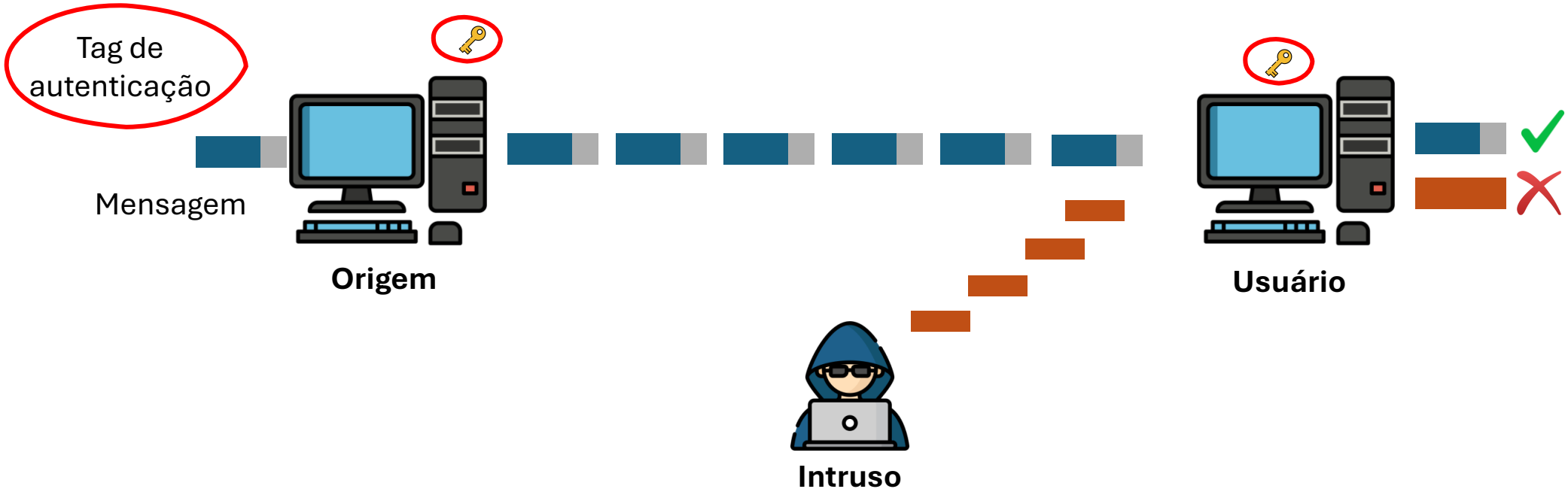
- **Apenas a origem e destino conhecem a chave** e conseguem calcular a redundância corretamente: autenticidade.
- Também garante **integridade** – alteração na mensagem detectada, como no caso das funções de hash



Estratégia

Usar **redundância dependente de chave**

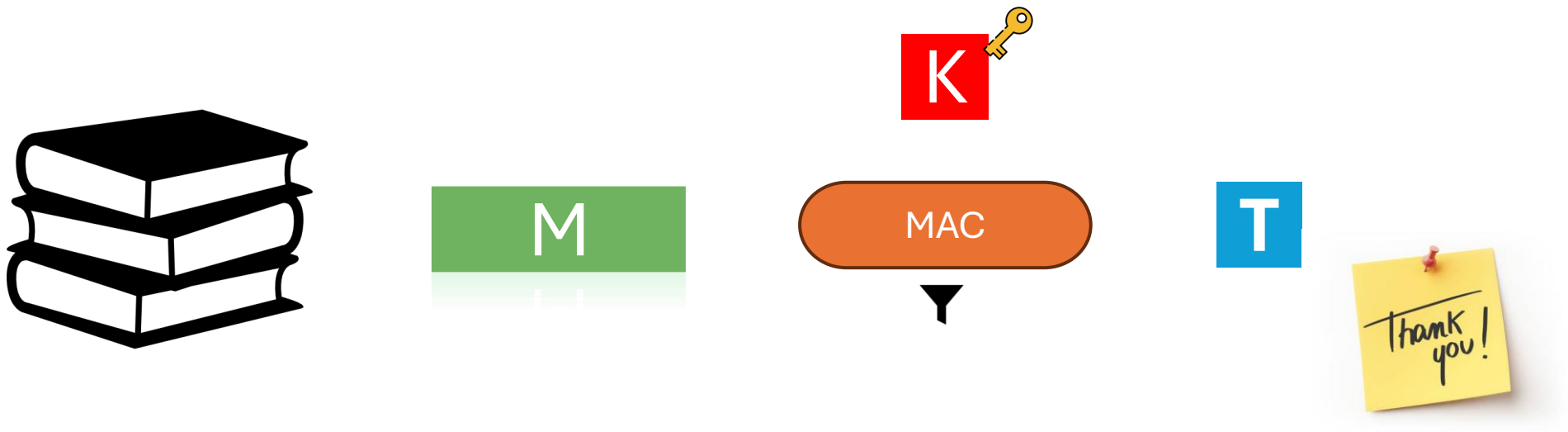
- **Apenas a origem e destino conhecem a chave** e conseguem calcular a tag de autenticação corretamente.



Código de autenticação

Redundância anexadas a mensagens de modo a detectar alterações (**integridade**) e garantir a **autenticidade** do remetente.

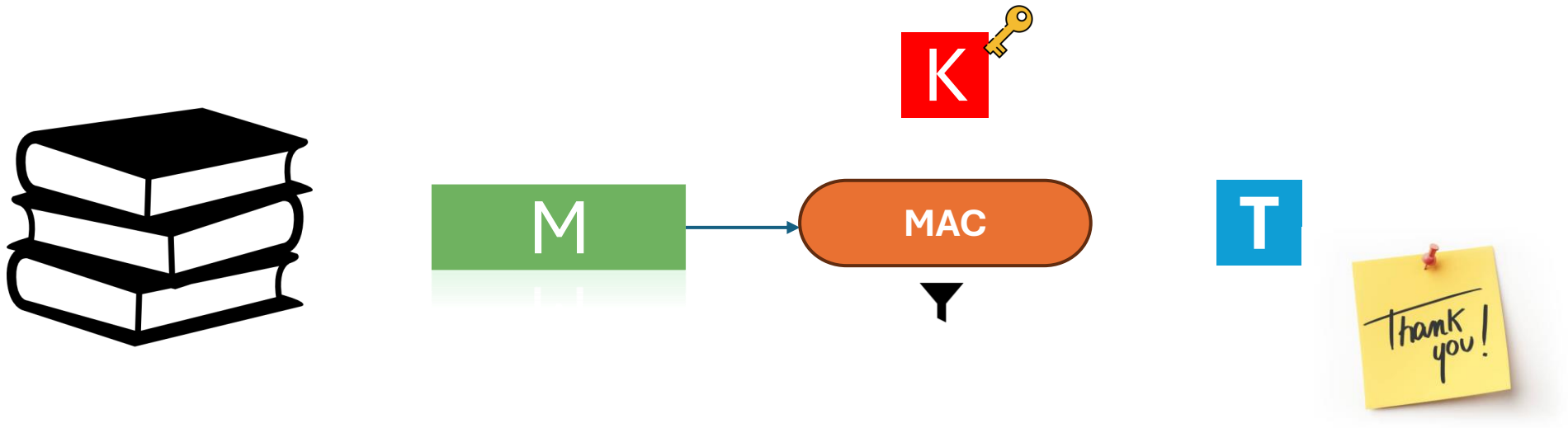
- Chamada de “tag (etiqueta) de autenticação”
- Dependem da mensagem e de uma chave secreta compartilhada entre o remetente e o destinatário



Código de autenticação

Redundância anexadas a mensagens de modo a detectar alterações (**integridade**) e garantir a **autenticidade** do remetente.

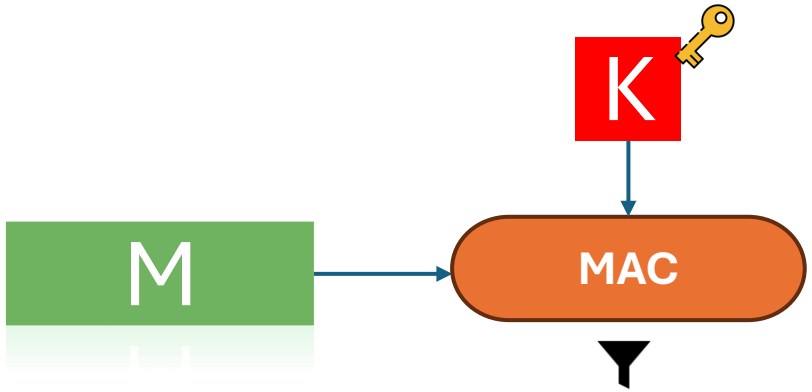
- Chamada de “tag (etiqueta) de autenticação”
- Dependem da mensagem e de uma chave secreta compartilhada entre o remetente e o destinatário



Código de autenticação

Redundância anexadas a mensagens de modo a detectar alterações (**integridade**) e garantir a **autenticidade** do remetente.

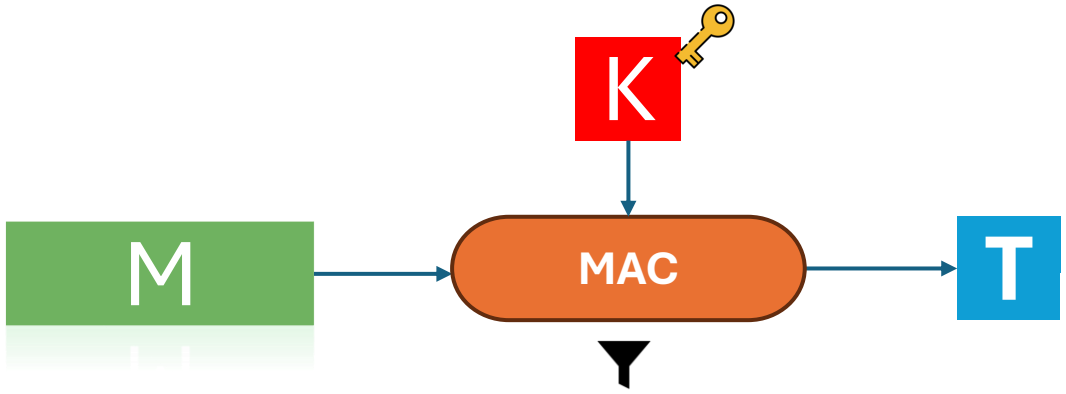
- Chamada de “tag (etiqueta) de autenticação”
- Dependem da mensagem e de uma chave secreta compartilhada entre o remetente e o destinatário



Código de autenticação

Redundância anexadas a mensagens de modo a detectar alterações (**integridade**) e garantir a **autenticidade** do remetente.

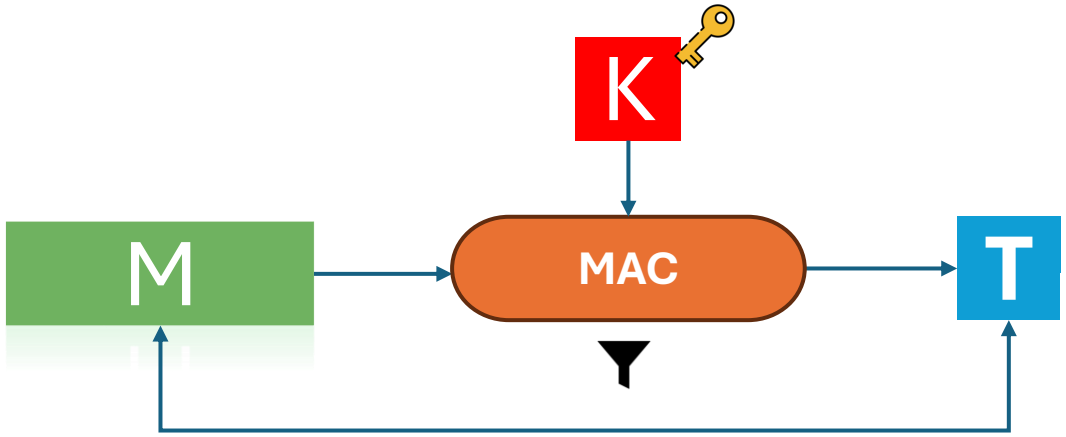
- Chamada de “tag (etiqueta) de autenticação”
- Dependem da mensagem e de uma chave secreta compartilhada entre o remetente e o destinatário



Código de autenticação

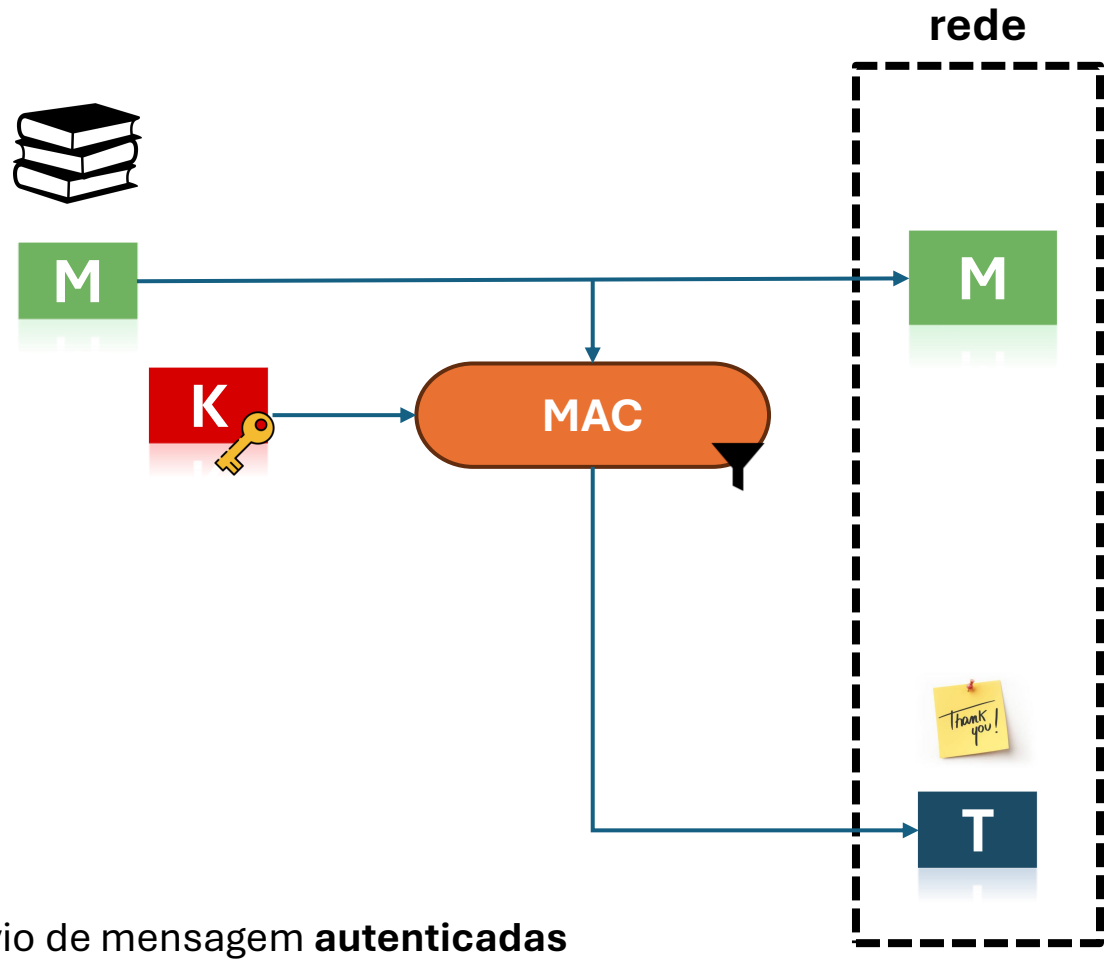
Redundância anexadas a mensagens de modo a detectar alterações (**integridade**) e garantir a **autenticidade** do remetente.

- Chamada de “tag (etiqueta) de autenticação”
- Dependem da mensagem e de uma chave secreta compartilhada entre o remetente e o destinatário



Somente usuários com as chaves podem calcular o **M** e o **T**

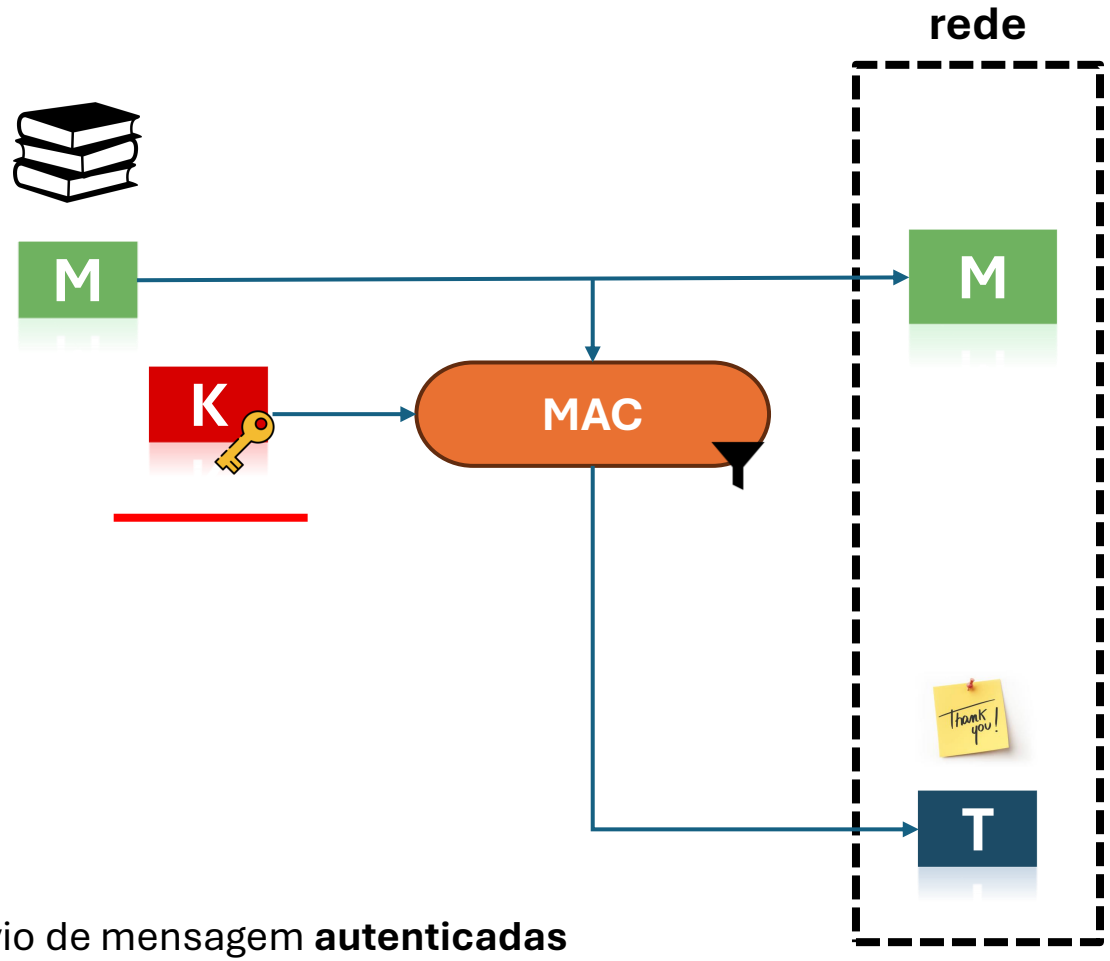
Código de autenticação: uso



Envio de mensagem **autenticadas**

- K: chave simétrica compartilhada
- T: tag -- garante integridade e autenticidade

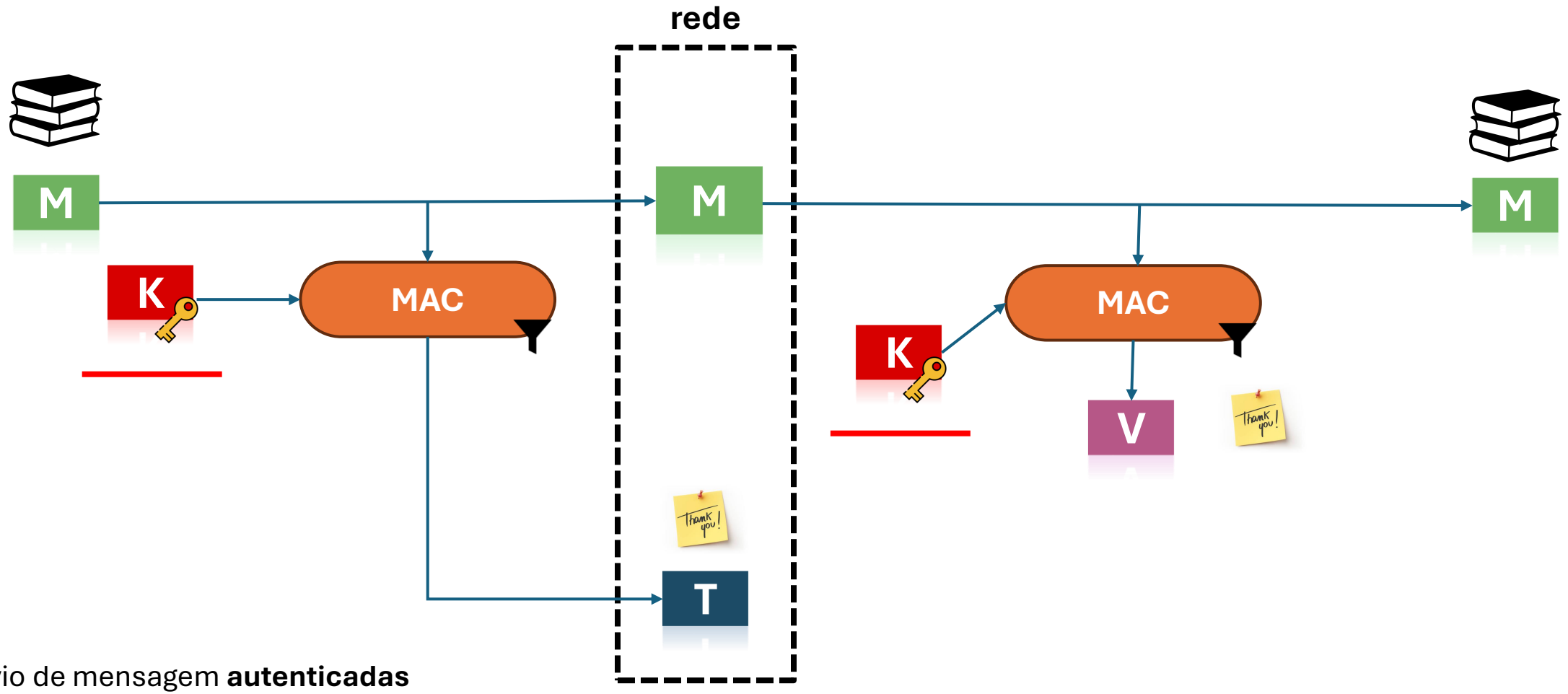
Código de autenticação: uso



Envio de mensagem **autenticadas**

- K: chave simétrica compartilhada
- T: tag -> garante integridade e autenticidade

Código de autenticação: uso

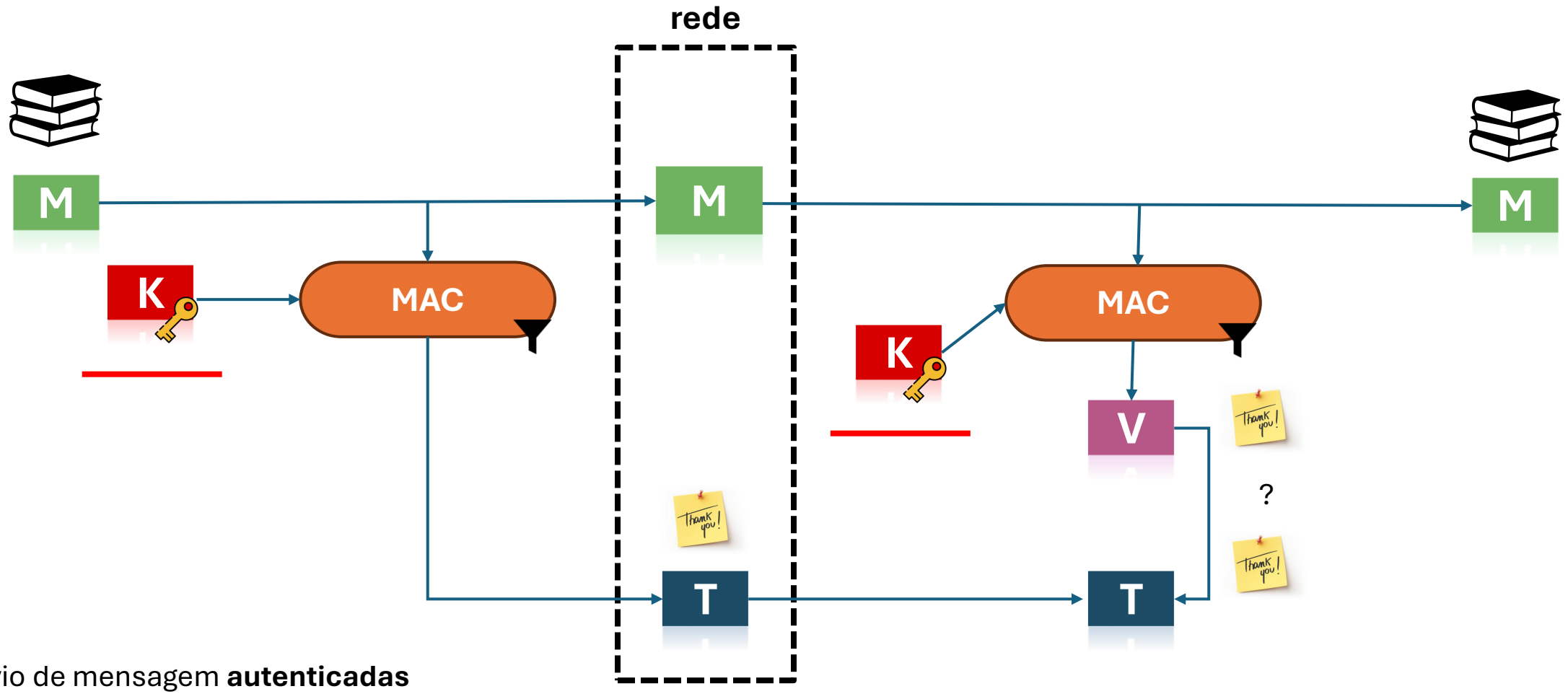


Envio de mensagem **autenticadas**

- K: chave simétrica compartilhada
- T: tag -> garante integridade e autenticidade

Somente integridade e autenticidade (caso anterior)

Código de autenticação: uso



Envio de mensagem **autenticadas**

- K: chave simétrica compartilhada
- T: tag -> garante integridade e autenticidade

Se o que foi calculado localmente bateu com o que foi enviado, então está tudo certo. Mensagem integra e autentica.

INSTITUTO FEDERAL
São Paulo
Campus Campinas

Algoritmos

Baseados em cifras de blocos:

- **CMAC** (NIST SP 800-38B)
- Pró: tamanho de código (reusam cifras de bloco, mas há custo de implementação, e desempenho pode ser inferior ao de baixo)

Baseados em funções hash:

- **HMAC** (FIPS 198)
- Pró: desempenho (funções de hash puras)

Combinados com cifras:

- **AEAD**: *Authenticaed Encryption with Associated Data* (confidencialidade de parte dos dados)
- Portifólio fixado: Caesar (http://competitions.cr.yp.to/caesar.html)

Algoritmos

Baseados em cifras de blocos:

- **CMAC** (NIST SP 800-38B)
- Pró: tamanho de código (reusam cifras de bloco, mas há custo de implementação, e desempenho pode ser inferior ao de baixo)

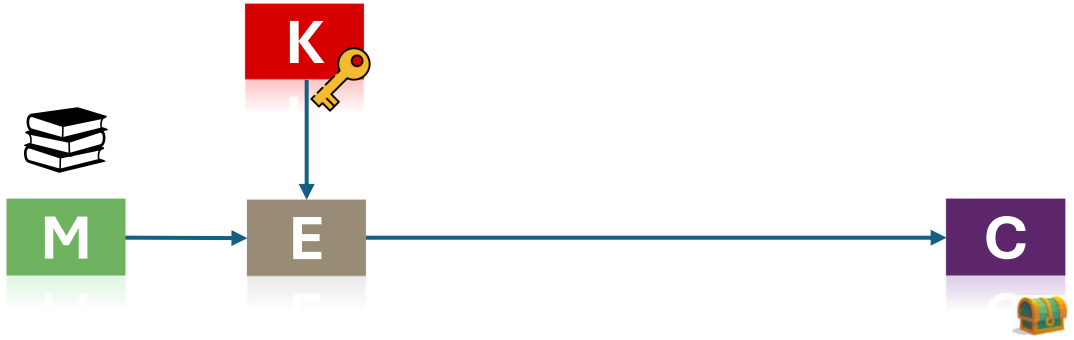
Baseados em funções hash:

- **HMAC** (FIPS 198) ← Rápido e bastante usado na prática (navegador – sessões https)
- Pró: desempenho (funções de hash puras)

Combinados com cifras:

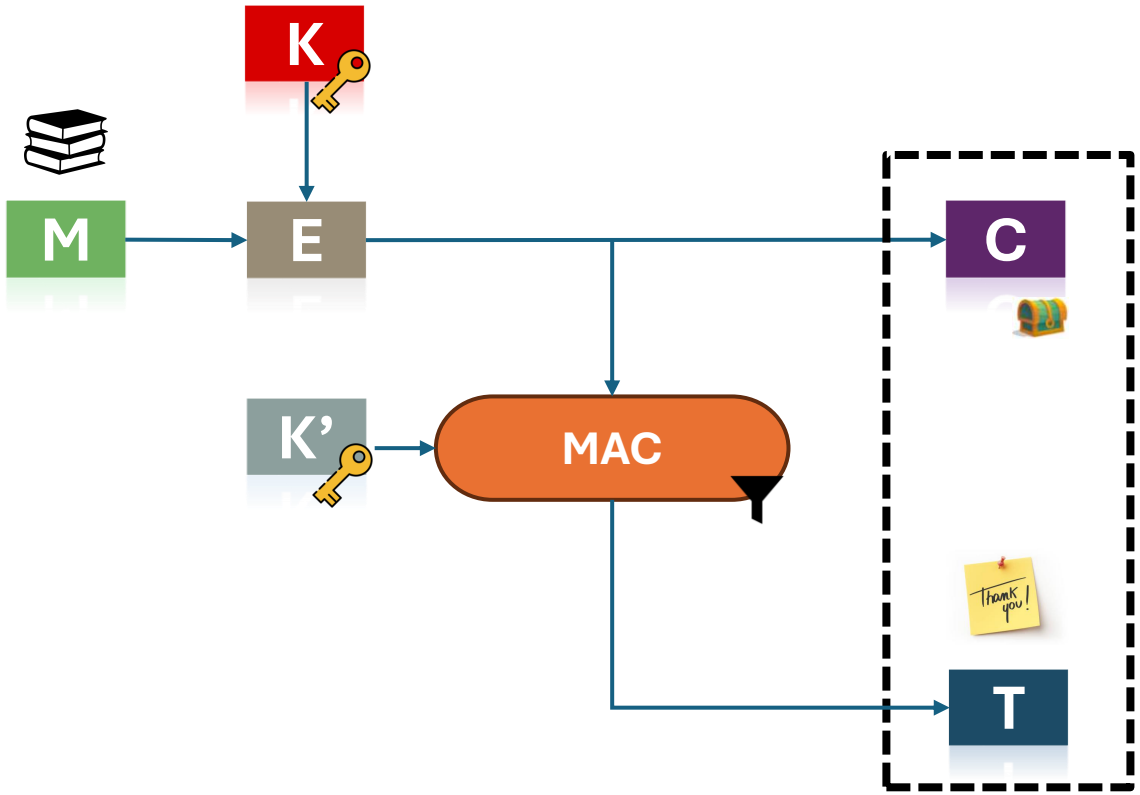
- **AEAD**: *Authenticaed Encryption with Associated Data* (confidencialidade de parte dos dados)
- Portifólio fixado: Caesar (http://competitions.cr.yp.to/caesar.html)

MAC + cifras (uso no HTTP/TLS)



Mensagem confidencial (C) e autenticada (T)

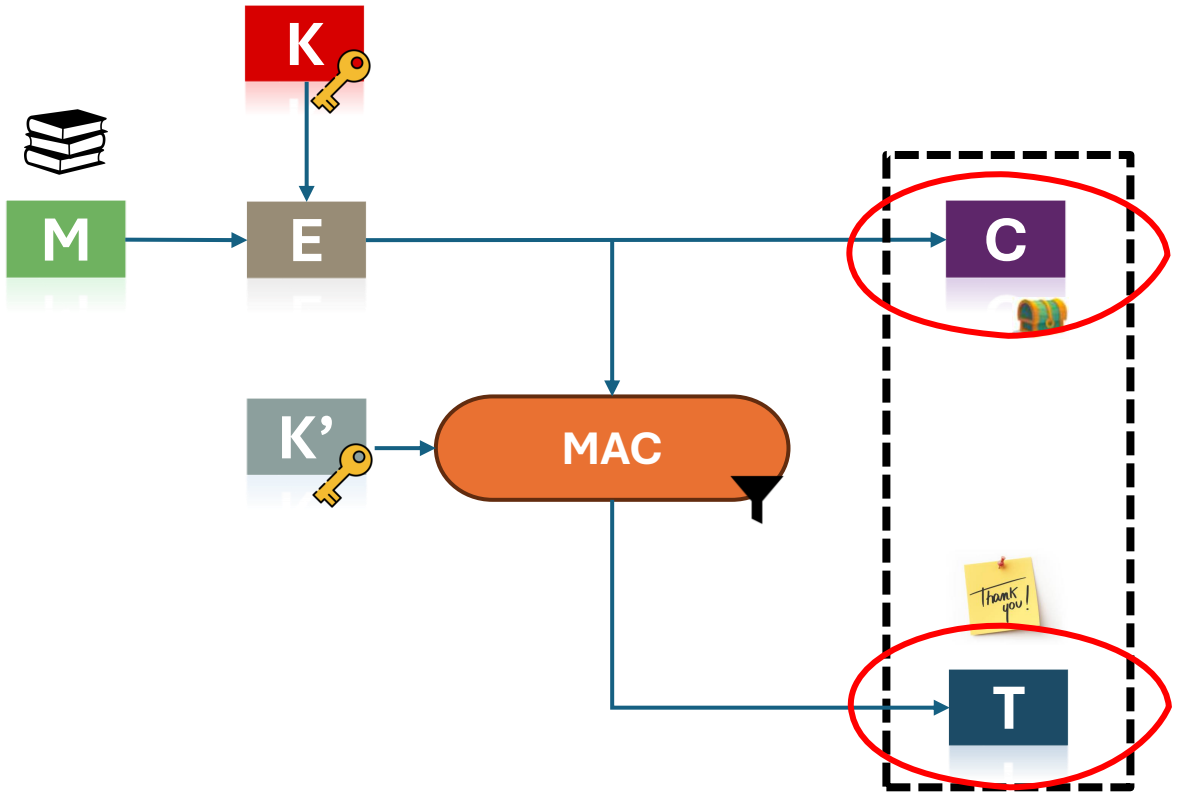
MAC + cifras (uso no HTTP/TLS)



Mensagem confidencial (C) e autenticada (T)

- K e K': chaves compartilhadas diferentes (ou uso de AEAD)
- E: algoritmo de cifração

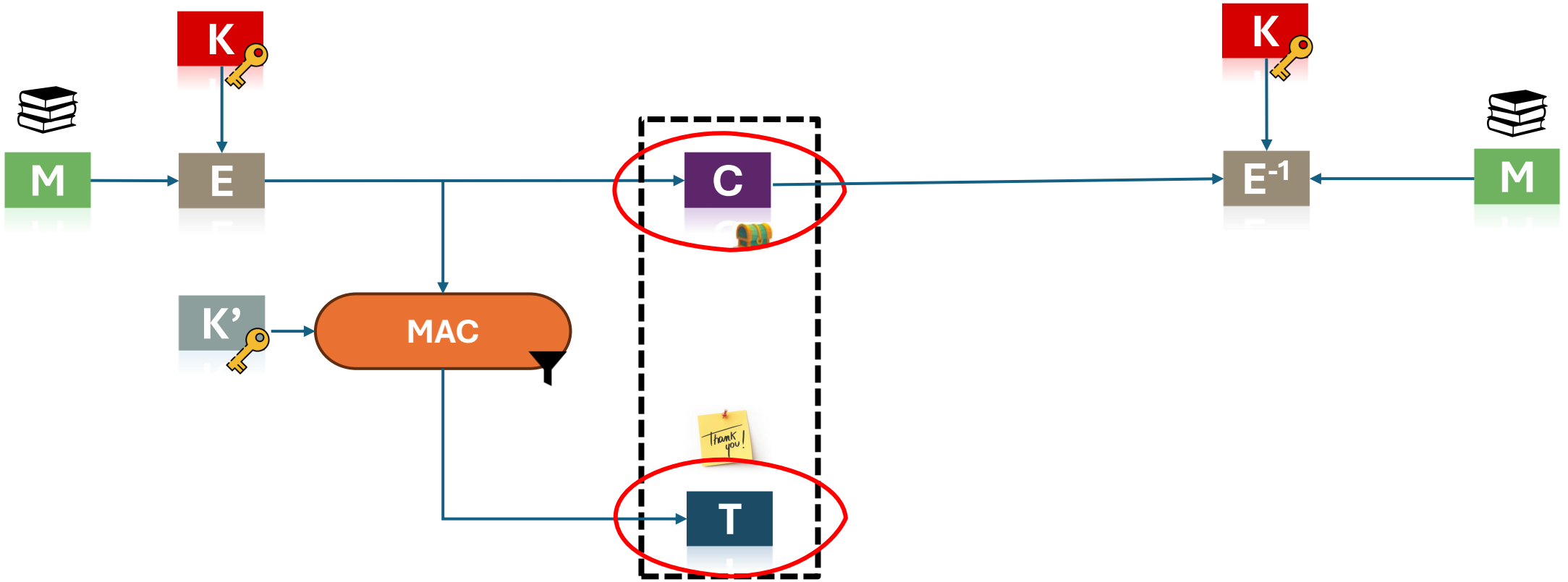
MAC + cifras (uso no HTTP/TLS)



Mensagem confidencial (C) e autenticada (T)

- K e K': chaves compartilhadas diferentes (ou uso de AEAD)

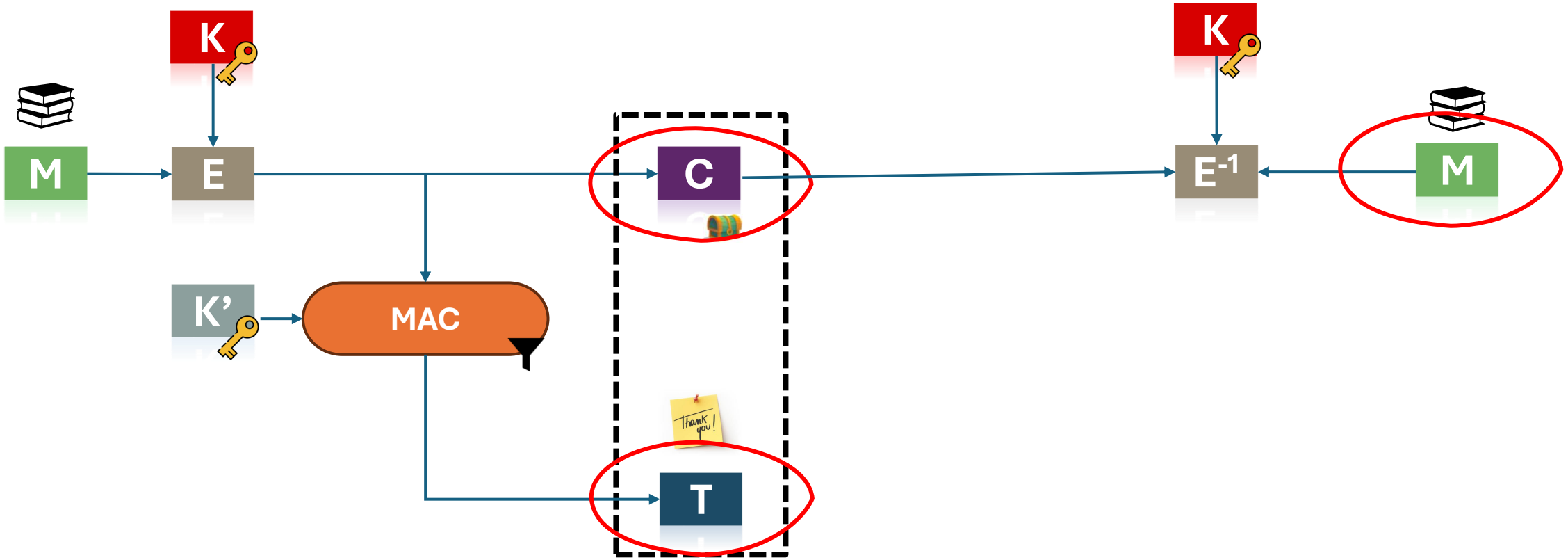
MAC + cifras (uso no HTTP/TLS)



Mensagem confidencial (C) e autenticada (T)

- K e K': chaves compartilhadas diferentes (ou uso de AEAD)

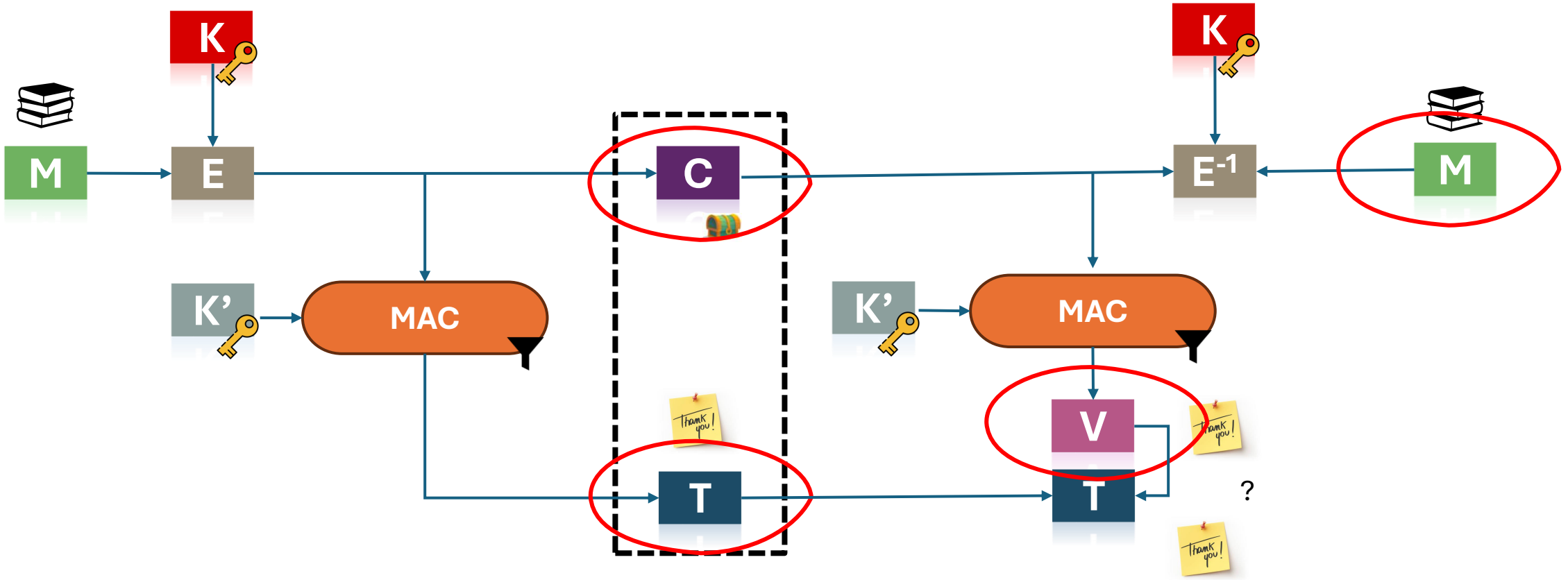
MAC + cifras (uso no HTTP/TLS)



Mensagem confidencial (C) e autenticada (T)

- K e K': chaves compartilhadas diferentes (ou uso de AEAD)

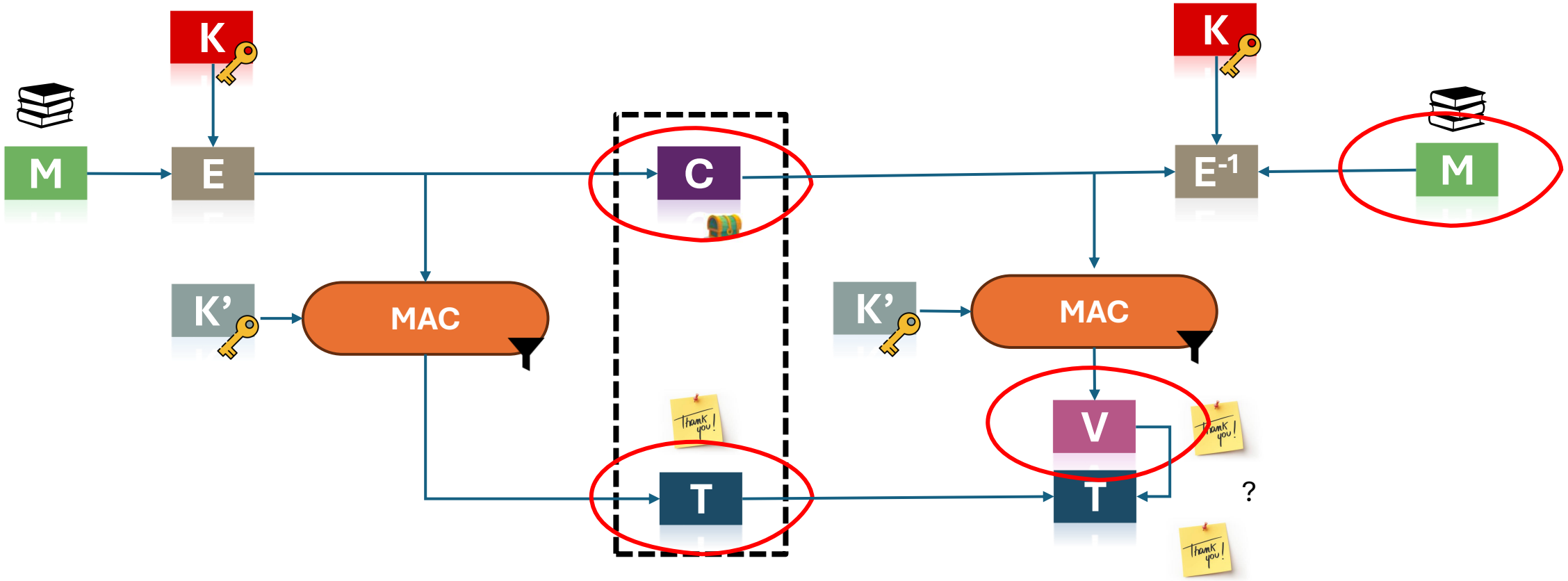
MAC + cifras (uso no HTTP/TLS)



Mensagem confidencial (C) e autenticada (T)

- K e K': chaves compartilhadas diferentes (ou uso de AEAD)

MAC + cifras (uso no HTTP/TLS)



Mensagem confidencial (C) e autenticada (T)

- K e K': chaves compartilhadas diferentes (ou uso de AEAD)
- Serviços: confidencialidade (cifra simétrica), integridade e autenticidade (algoritmo de MAC)

INSTITUTO FEDERAL
São Paulo
Campus Campinas

Geração de chaves: números aleatórios

Estudo de caso: Netscape

- Netscape 1.x (1995)
- Dois estudantes de Berkeley descrevem como quebrar a segurança do navegador, recuperando chaves usadas em sessões seguras (HTTPS) em 25 s.
- Chaves pequenas?
  - Não, chaves de 128 bits (tamanho atual!!)
- Pergunta: como isso é possível?

Estudo de caso: Netscape

- Netscape 1.x (1995)
- Dois estudantes de Berkeley descrevem como quebrar a segurança do navegador, recuperando chaves usadas em sessões seguras (HTTPS) em 25 s.
- Chaves pequenas?
  - Não, chaves de 128 bits (tamanho atual!!)
- Pergunta: como isso é possível?

... Elas não eram tão aleatórias

Análise de (in)segurança

Baixa **aleatoriedade das chaves de sessão:**

- Chaves geradas a partir do **relógio do sistema** (precisão de μs), sem acúmulo entre ativações;
- Conhecendo minuto da criação da sessão HTTPS: menos de 60 milhões de chaves possíveis.
  - Segurança cerca de 2^{26}, não 2^{128}

Análise de (in)segurança

Baixa **aleatoriedade das chaves de sessão**:

- Chaves geradas a partir do **relógio do sistema** (precisão de µs), sem acúmulo entre ativações;
- Conhecendo minuto da criação da sessão HTTPS: menos de 60 milhões de chaves possíveis.
  - Segurança cerca de 2^{26}, não 2^{128}
- Geração de chaves segura: **fontes de entropia** (navegadores)
  - Ex. (**físicas**): relógio, ruído térmico
  - Ex. (**comportamentais**): estatísticas de rede, pastas temporárias (Firefox), posição do mouse (**TrueCrypt**);
  - Soluções de sistema: “SecureRandom” (Java), “/dev/random” (Unix)

Geradores pseudo-aleatórios

Evitam necessidade de capturar entropia “bruta” repetidamente:

- Na prática: ganhos de desempenho

Basicamente:

- Coleta-se entropia baixa (de várias fontes!) para **criar/atualizar semente de tamanho adequado.**
  - A semente deve ser mantida secreta
  - Usa-se algoritmo determinístico para gerar uma **sequência “indistinguível” de bits aleatórios**.

- Algoritmos padrão: NIST-SP800-90A-Rev 1
  - Revisão removeu Dual EC DRBG (porta dos fundos da NSA)

Geradores pseudo-aleatórios

Evitam necessidade de capturar entropia “bruta” repetidamente:

- Na prática: ganhos de desempenho

Basicamente:

- Coleta-se entropia baixa (de várias fontes!) para **criar/atualizar semente de tamanho adequado.**
  - A semente deve ser mantida secreta
  - Usa-se algoritmo determinístico para gerar uma **sequência “indistinguível” de bits aleatórios**.

- Algoritmos padrão: NIST-SP800-90A-Rev 1
  - Revisão removeu Dual EC DRBG (porta dos fundos da NSA)

Conhecendo a semente, descubro todos eles.

CloudFlare

Cloudflare - https://www.cloudflare.com/learning/ssl/lava-lamp-encryption

https://www.youtube.com/watch?v=1cUUfMeOijg

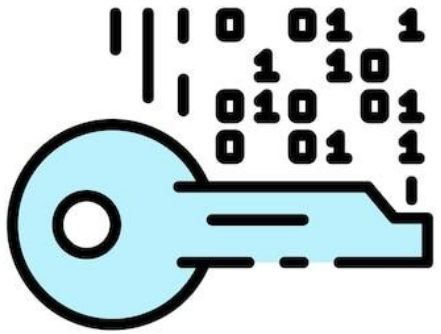
https://www.youtube.com/watch?v=eGaZpb3rP_I

INSTITUTO FEDERAL
São Paulo
Campus Campinas

Algoritmos assimétricos e certificação digital

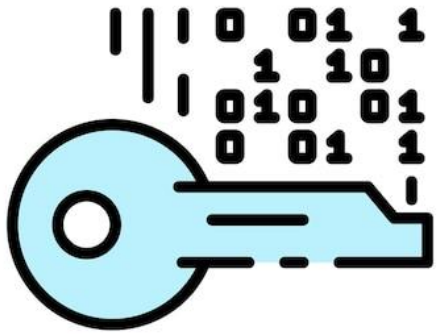
INSTITUTO FEDERAL
São Paulo
Campus Campinas

Distribuição de chaves



Como as chaves chegam nas duas entidades de comunicação?

Distribuição de chaves



Chaves devem ser distribuídas por meio de canal seguro

- Senão, eles podem ser interceptados

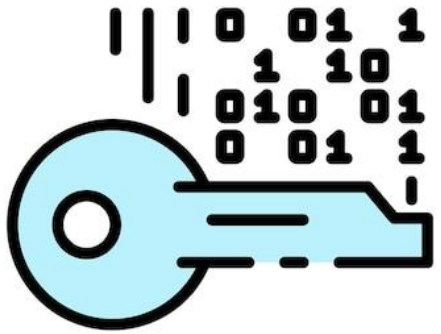
Abordagem 1: **manualmente**

- Comum em redes locais (exemplos: WPA2-PSK, Bluetooth)

- Cenário:
  - Sincronização entre celular e tv (cadastro em ambos com a mesma chave)

- Quando as entidades que precisam se comunicar uma com a outra estão próximas.

Distribuição de chaves



Chaves devem ser distribuídas por meio de canal seguro

- Senão, eles podem ser interceptados

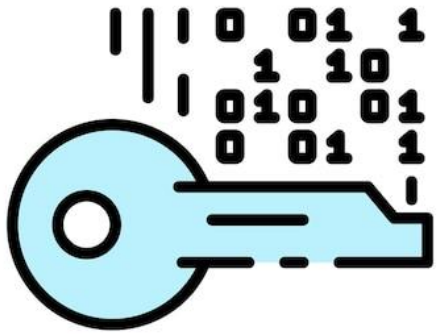
Abordagem 1: **manualmente**

- Comum em redes locais (exemplos: WPA2-PSK, Bluetooth)

Abordagem 2: com a ajuda de uma **entidade confiável**

- Exemplo: usando centro de distribuição de chaves (*Key Distribution Center* KDC), e.g. via protocolo Kerberos.
  - Servidor compartilha uma chave com todo mundo que é feita durante o cadastro do usuário.
  - Os usuários que não tem uma comum, ajuda a construir (aplicação Kerberos).

Distribuição de chaves



Chaves devem ser distribuídas por meio de canal seguro

- Senão, eles podem ser interceptados

Abordagem 1: **manualmente**

- Comum em redes locais (exemplos: WPA2-PSK, Bluetooth)

Abordagem 2: com a ajuda de uma **entidade confiável**

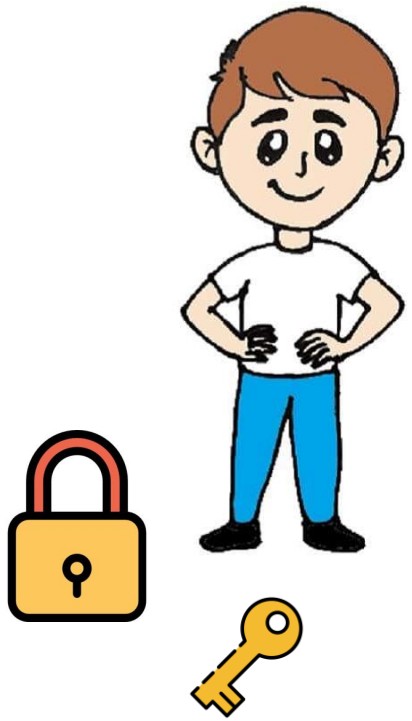
- Exemplo: usando centro de distribuição de chaves (*Key Distribution Center* KDC), e.g. via protocolo Kerberos.

Abordagem 3: sem a ajuda de uma entidade confiável (internet)

- Em geral, envolvem **algoritmos assimétricos**

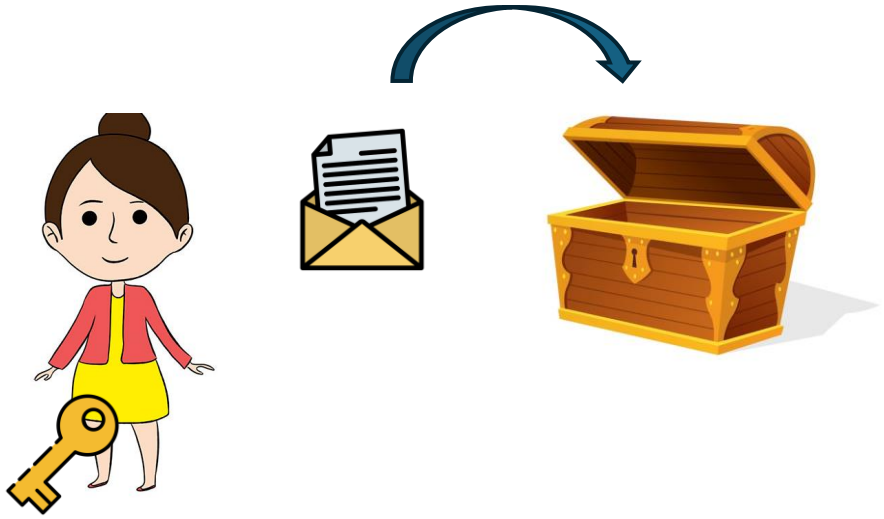
Exemplo didático: Protocolo Massey-Omura

Problema: Alice deseja enviar uma carta confidencial para Chico usando apenas um baú com cadeado e sua respectiva chaves.



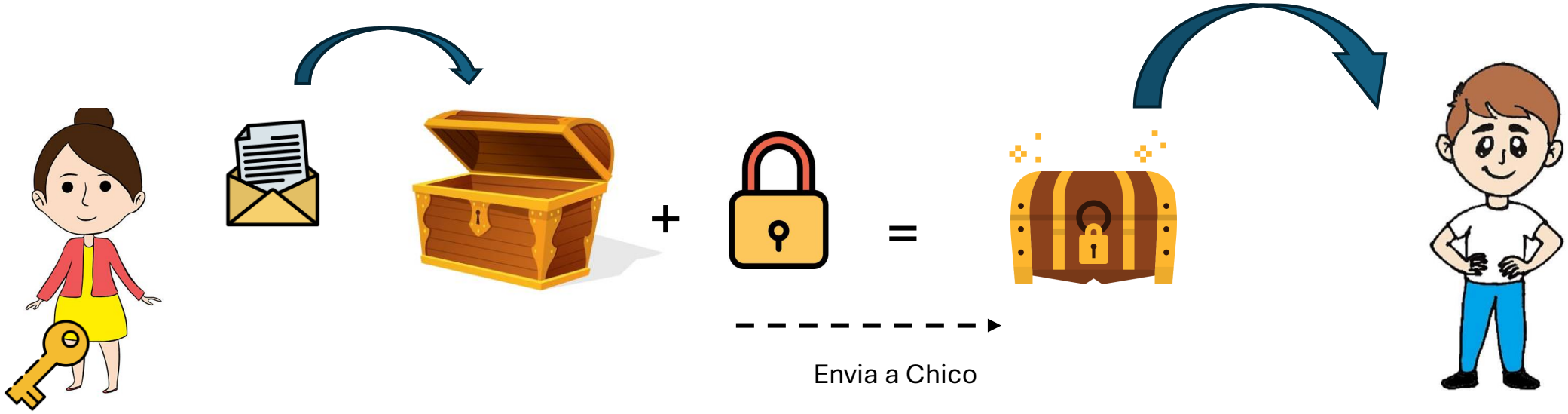
Exemplo didático: Protocolo Massey-Omura

Exemplo didático: Protocolo Massey-Omura



Exemplo didático: Protocolo Massey-Omura

Exemplo didático: Protocolo Massey-Omura

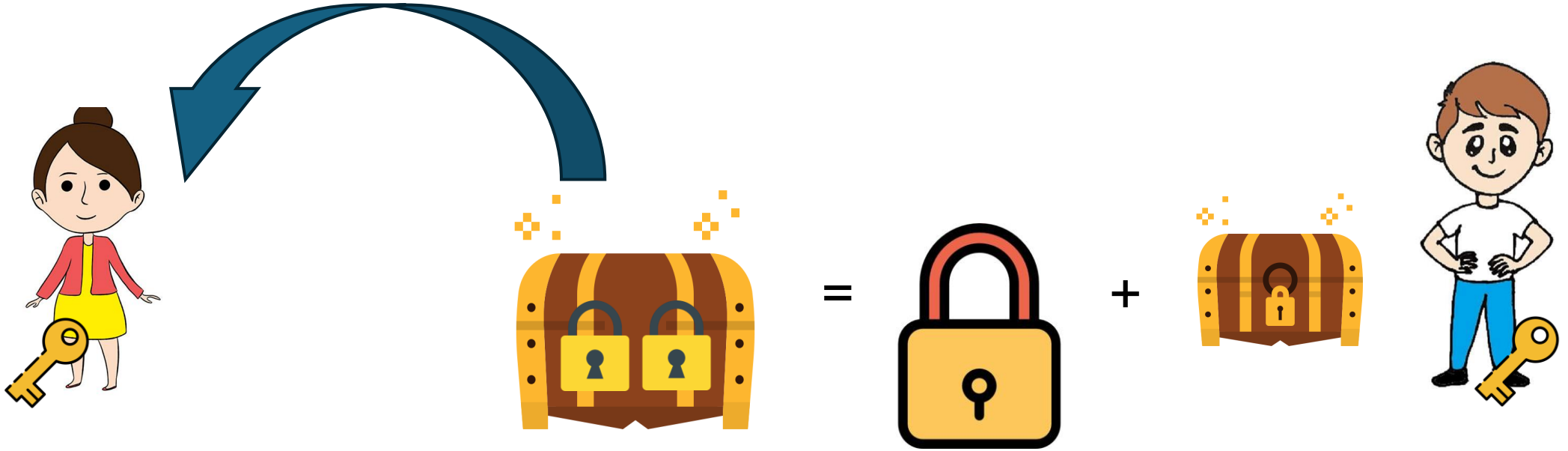


Exemplo didático: Protocolo Massey-Omura

Como abrir?

Exemplo didático: Protocolo Massey-Omura

Exemplo didático: Protocolo Massey-Omura

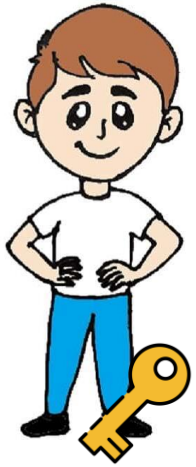


Exemplo didático: Protocolo Massey-Omura

Exemplo didático: Protocolo Massey-Omura

Exemplo didático: Protocolo Massey-Omura

Exemplo didático: Protocolo Massey-Omura



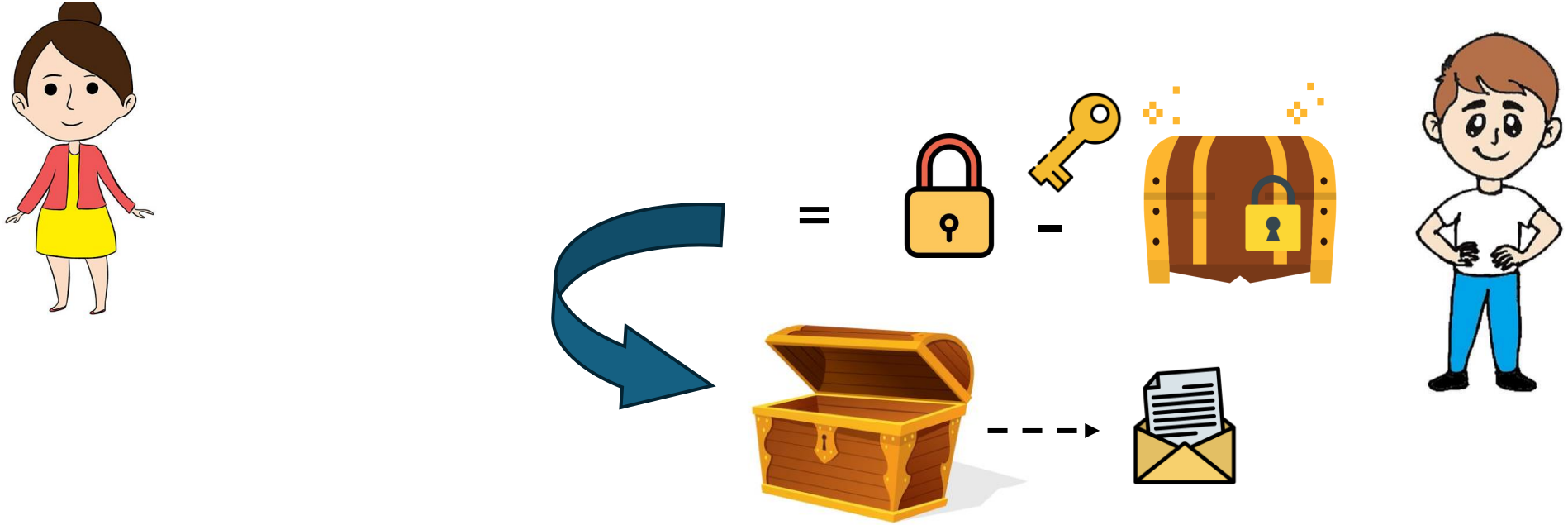
Exemplo didático: Protocolo Massey-Omura

=

-

Exemplo didático: Protocolo Massey-Omura

Exemplo didático: Protocolo Massey-Omura



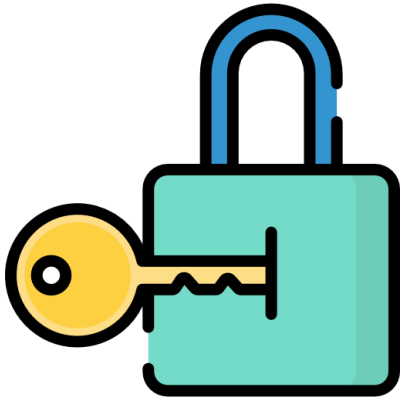
Vantagens didática:

- Conceitos de **chave pública** e **chave privada** (base da criptografia assimétrica)

INSTITUTO FEDERAL
São Paulo
Campus Campinas

Criptografia assimétrica

Criptografia assimétrica



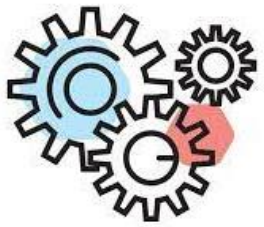
Duas chaves distintas:

- **Chave pública K_U**: divulgada abertamente
  - Paralelo: cadeado
- **Chave privada K_R**: conhecida apenas pelo seu dono (não passa pela rede)
  - Paralelo: a chave do cadeado

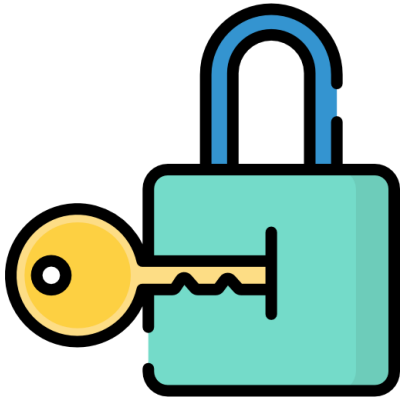
Transformações feitas **com uma chave** somente podem ser **invertidas com a outra chave.**

Ambas as chaves são geradas pelo seu dono

- Em um algoritmo seguro: deve ser **inviável calcular a chave privada a partir da chave pública.**
- Para se comunicar, os usuários devem obter, de alguma forma, a chave pública de seus interlocutores.



Criptografia assimétrica



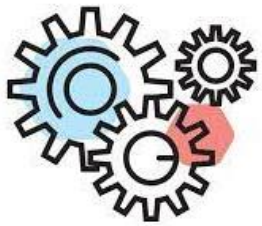
Duas chaves distintas:

- **Chave pública K_U**: divulgada abertamente
  - Paralelo: cadeado
- **Chave pública K_R**: conhecida apenas pelo seu dono (não passa pela rede)
  - Paralelo: a chave do cadeado

Transformações feitas **com uma chave** somente podem ser **invertidas com a outra chave.**

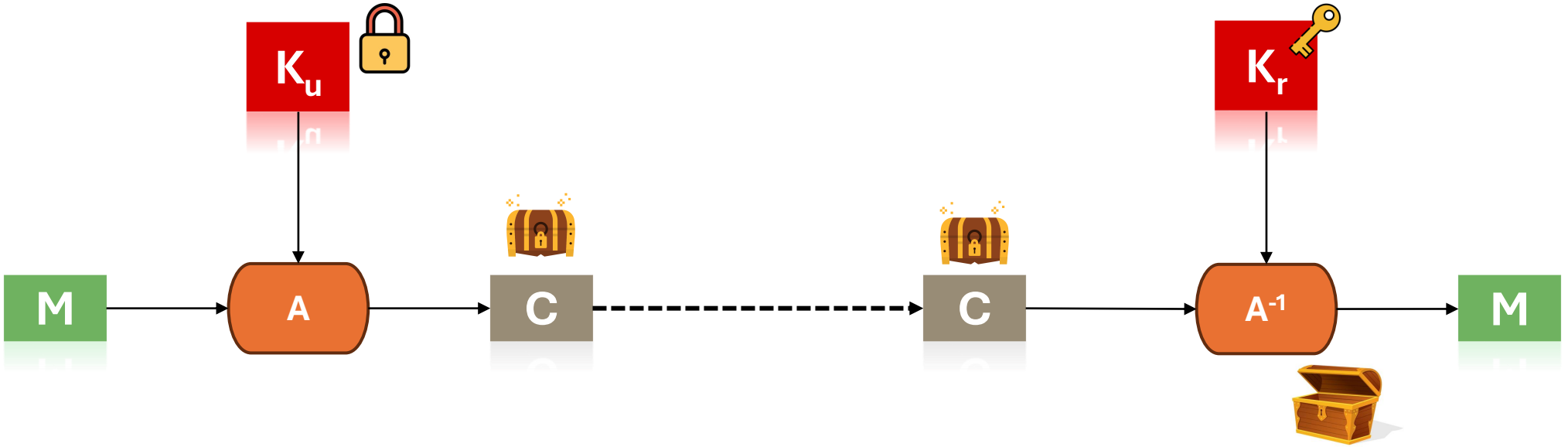
Ambas as chaves são geradas pelo seu dono

- Em um algoritmo seguro: deve ser **inviável calcular a chave privada a partir da chave pública.**
- Para se comunicar, os usuários devem obter, de alguma forma, a chave pública de seus interlocutores.



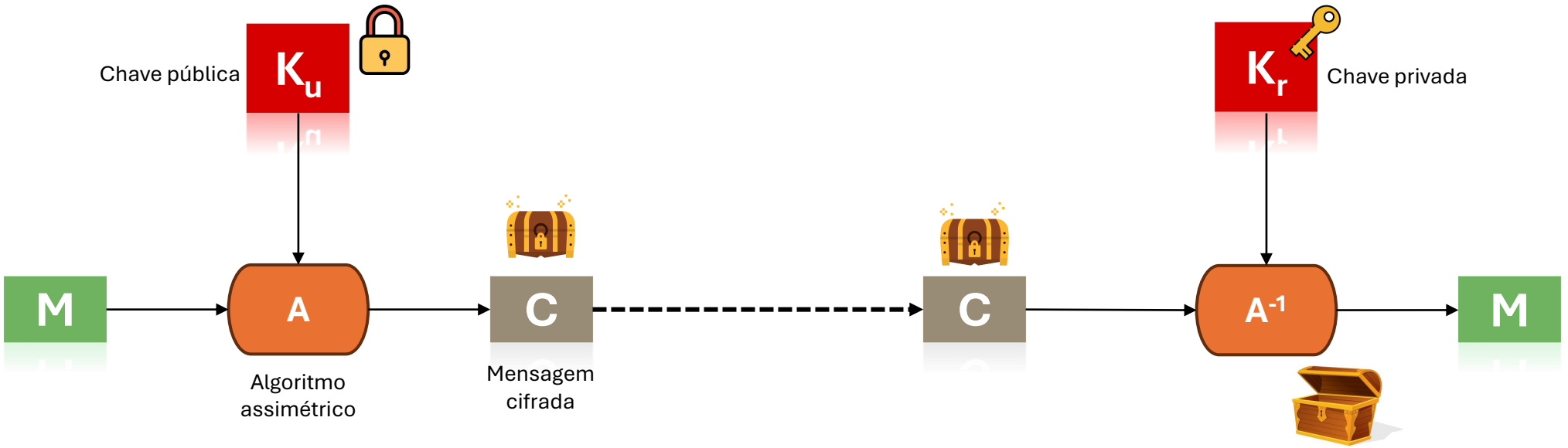
Criptografia assimétrica

Cifração: confidencialidade



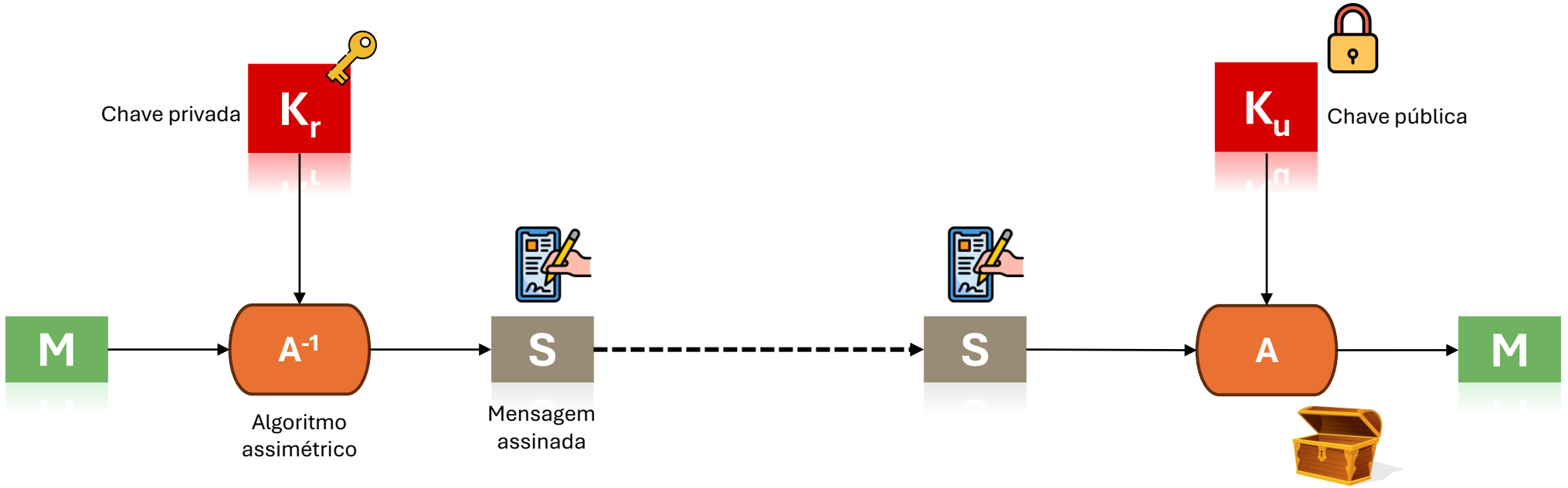
Criptografia assimétrica

Cifração: confidencialidade



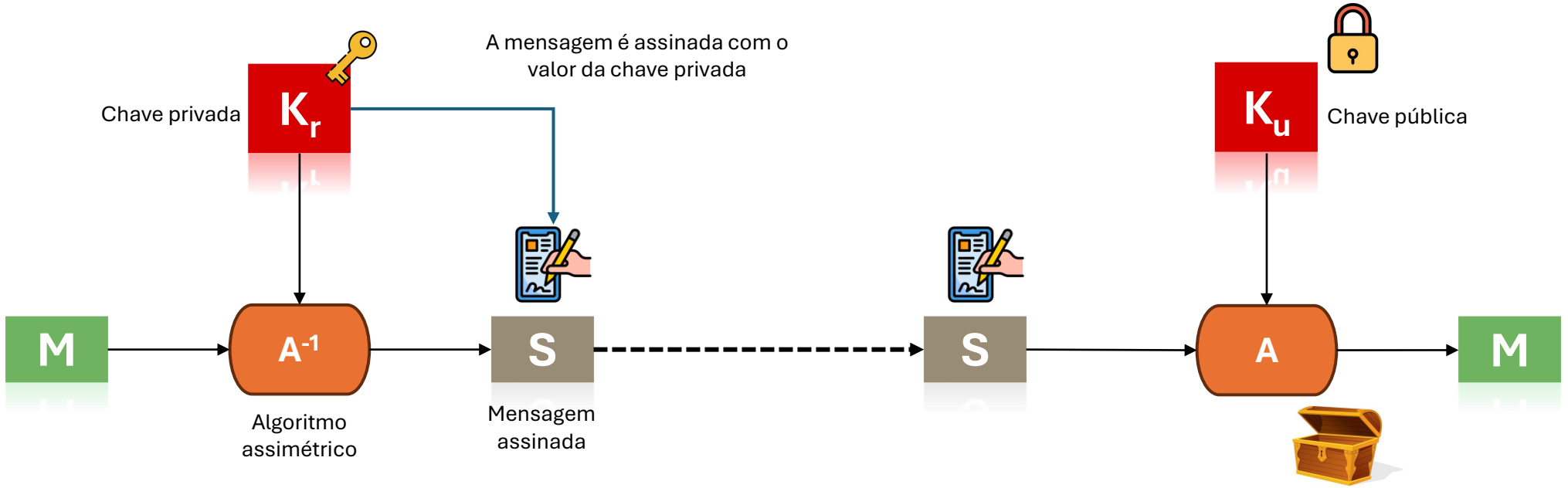
Criptografia assimétrica

Assinatura digital: integridade, autenticidade e irretratabilidade



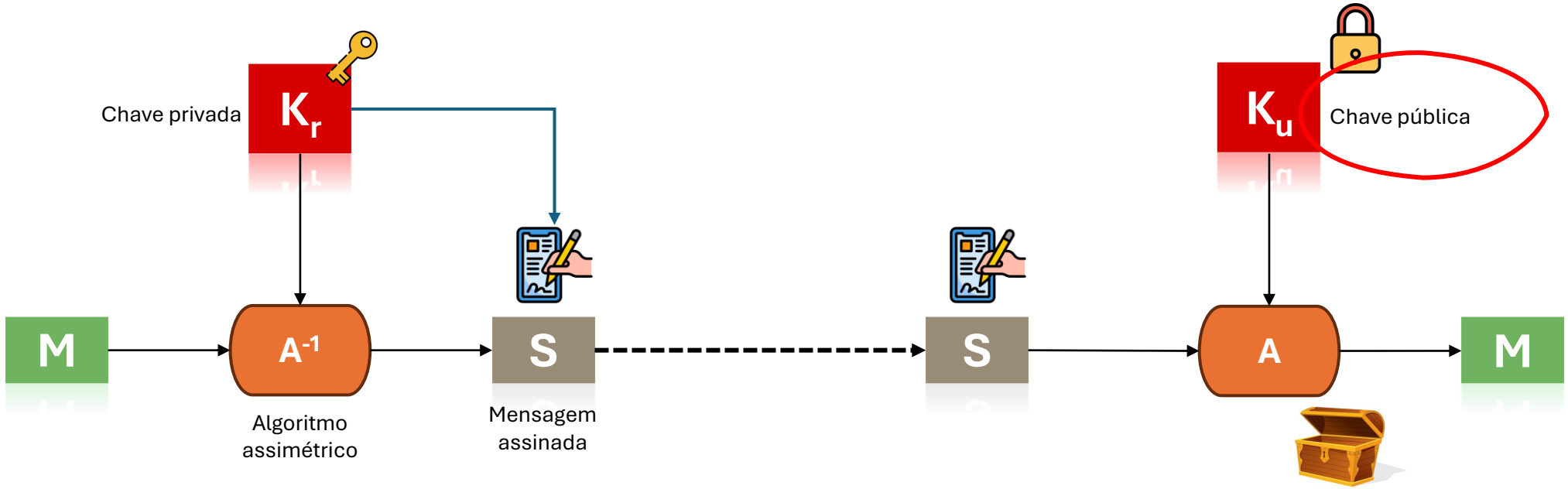
Criptografia assimétrica

Assinatura digital: integridade, autenticidade e irretratabilidade



Criptografia assimétrica

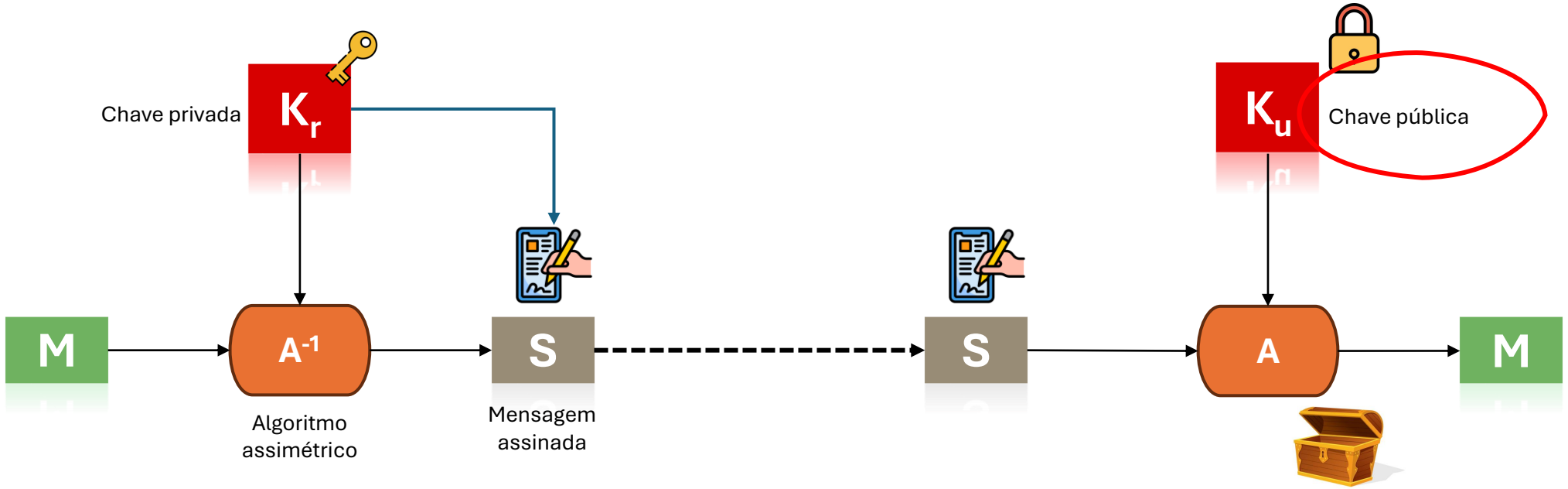
Assinatura digital: integridade, autenticidade e irretratabilidade



Qualquer pessoa de posse da minha chave pública, consegue verificar que o M e S é válido

Criptografia assimétrica

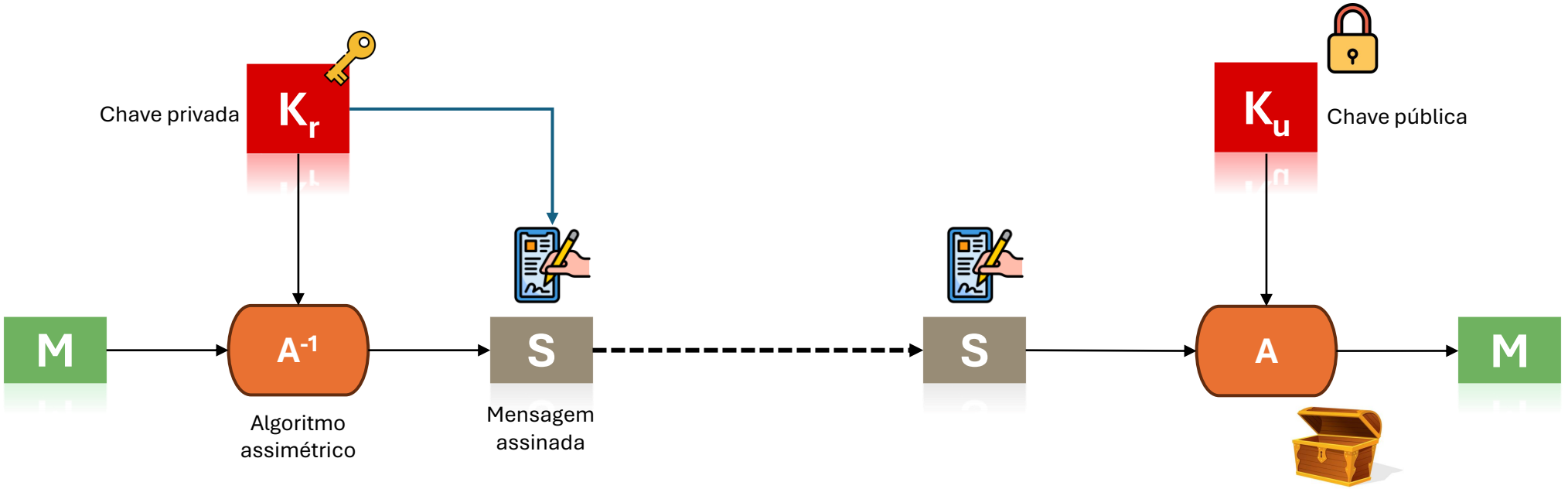
Assinatura digital: integridade, autenticidade e irretratabilidade



Apenas eu posso gerar a assinatura S correspondente à mensagem M, pois somente eu possuo a chave privada.

Qualquer pessoa de posse da minha chave pública, consegue verificar que o M e S é válido

Criptografia assimétrica

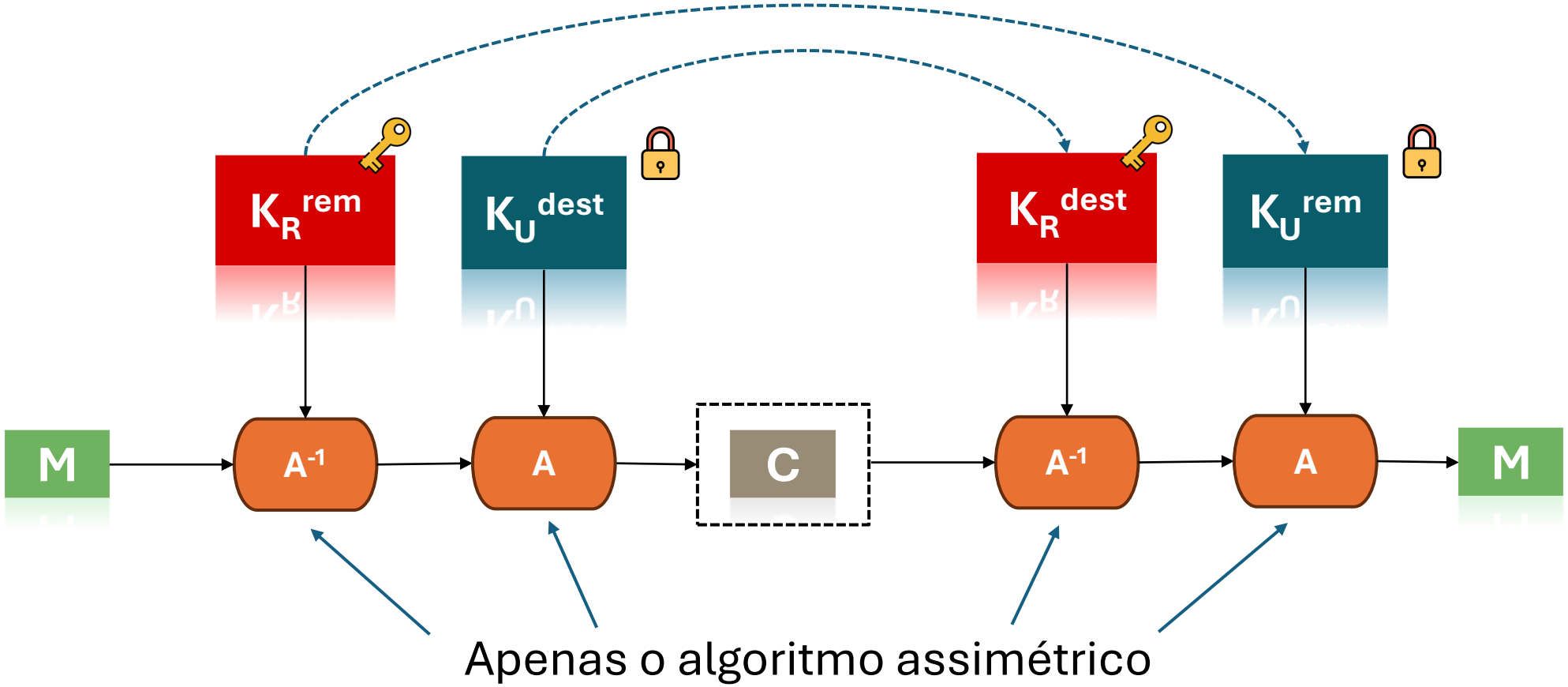


Criptografia são 4 serviços de segurança: confidencialidade, integridade, autenticidade e irretratabilidade.

É possível fazer tudo com a criptografia assimétrica

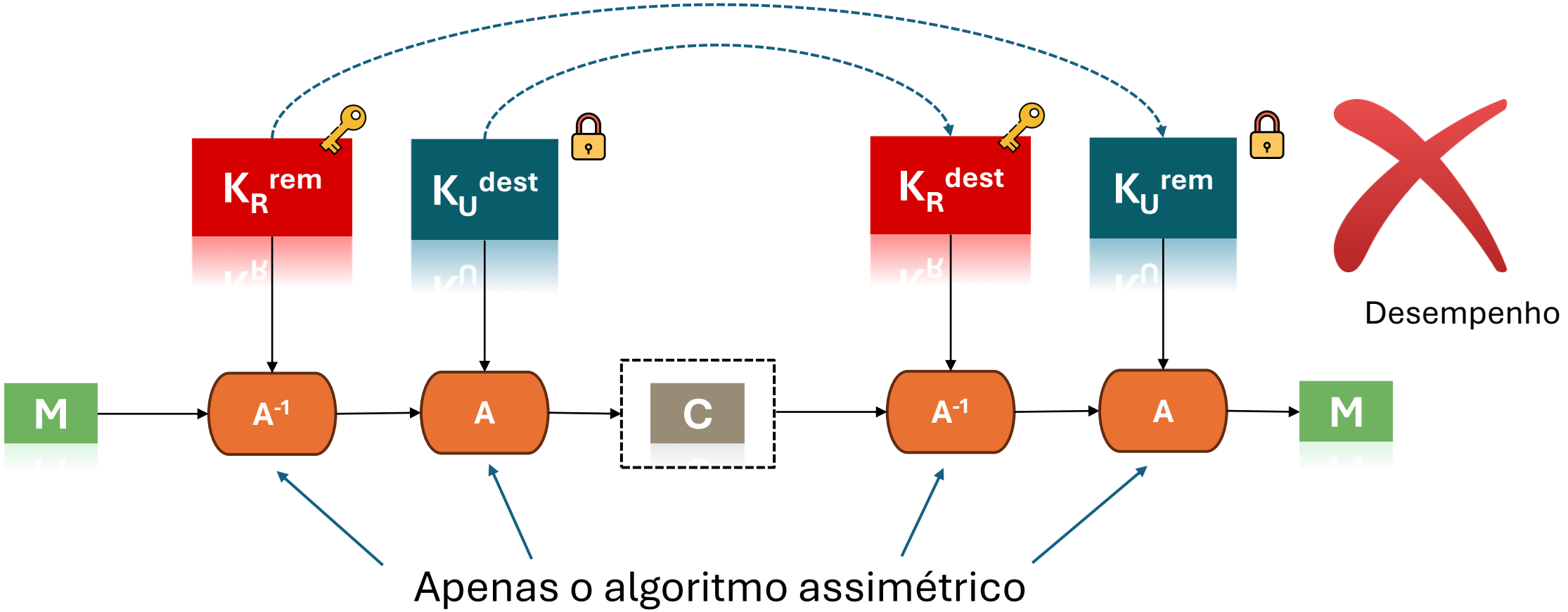
Envelope criptográfico assimétrico

É possível obter confidencialidade, integridade, autenticidade e irretratabilidade aplicando apenas criptografia assimétrica sobre o conteúdo completo da mensagem



Envelope criptográfico assimétrico

É possível obter confidencialidade, integridade, autenticidade e irretratabilidade aplicando apenas criptografia assimétrica sobre o conteúdo completo da mensagem



Criptografia assimétrica + simétrica

Algoritmos **assimétricos** costumam ser **combinados com simétricos por razões de desempenho**:

- Algoritmos simétricos costumam ser ~1000 vezes mais rápidos que os assimétricos;
- Prática: os algoritmos assimétricos são usados para dar o suporte necessário para o algoritmo simétrico ficar mais eficiente na solução como um todo.

Exemplos:

- **Estabelecimentos de chaves simétricas (confidencialidade)**: usadas por cifras simétricas e algoritmos de MAC (transferências de chaves entre entidades).
- **Assinatura digital do hash** da mensagem ao invés da mensagem em si: menor quantidade de dados processados pelo algoritmo assimétrico

Criptografia assimétrica + simétrica

Algoritmos **assimétricos** costumam ser **combinados com simétricos por razões de desempenho**:

- Algoritmos simétricos costumam ser ~1000 vezes mais rápidos que os assimétricos;
- Prática: os algoritmos assimétricos são usados para dar o suporte necessário para o algoritmo simétrico ficar mais eficiente na solução como um todo.

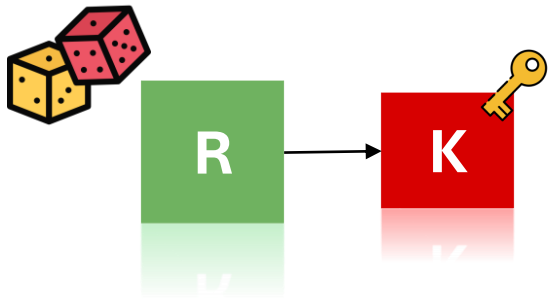
Exemplos:

- **Estabelecimentos de chaves simétricas (confidencialidade)**: usadas por cifras simétricas e algoritmos de MAC (transferências de chaves entre entidades).
- **Assinatura digital do hash** da mensagem ao invés da mensagem em si: menor quantidade de dados processados pelo algoritmo assimétrico.
  - **2ª inversão e colisões** (mensagens distintas não chegam no mesmo hash)

Transmissão de chave simétrica

Transmissão de chave simétrica K

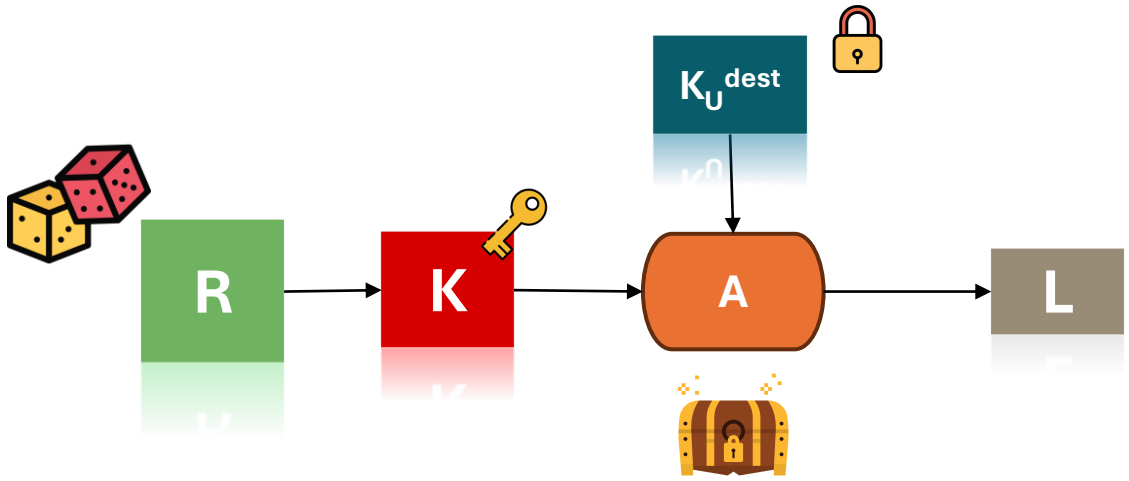
- 1º passo: construção de uma chave secreta com uma aleatoriedade (e.g, algoritmos pseudoaleatórios)
  - R: número aleatório [fonte de entropia] gera chave K



Transmissão de chave simétrica

Transmissão de chave simétrica K

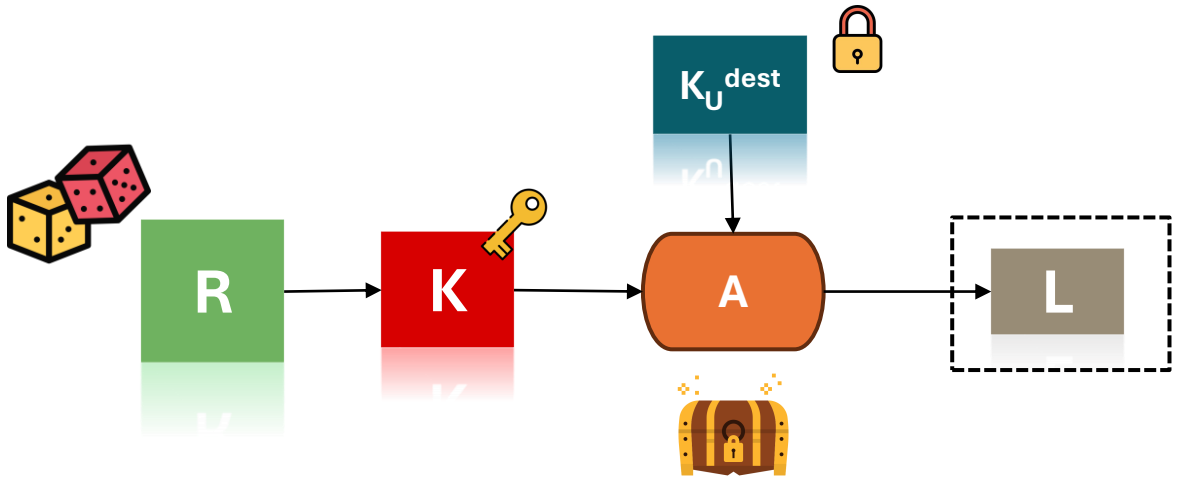
- R: número aleatório [fonte de entropia] gera chave K
- L: chave K protegida por chave pública do destinatário (K_u)



Transmissão de chave simétrica

Transmissão de chave simétrica K

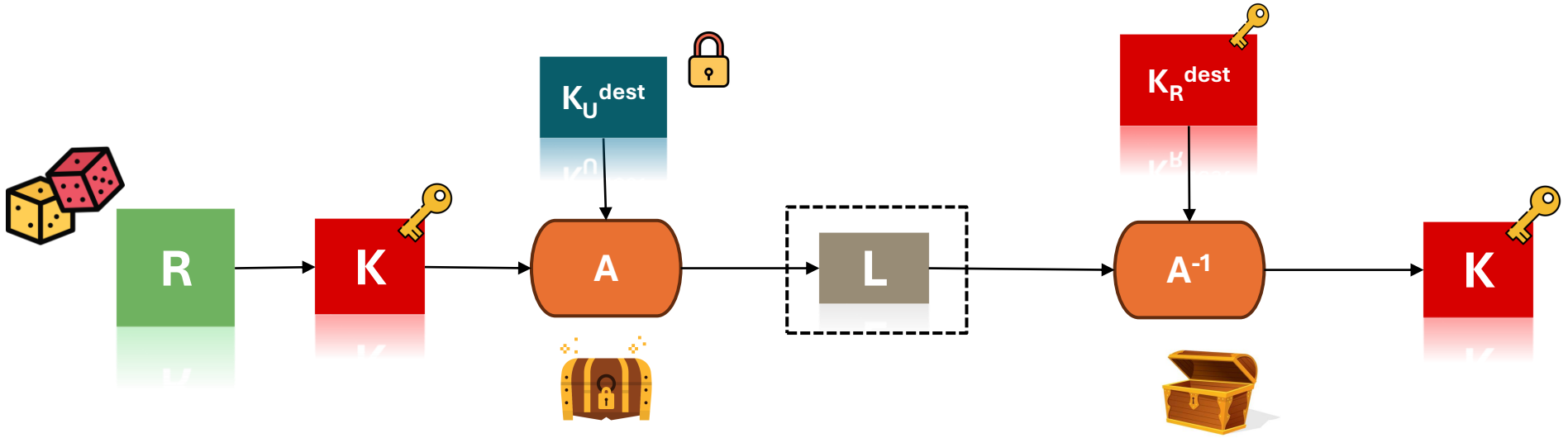
- R: número aleatório [fonte de entropia] gera chave K
- L: chave K protegida por chave pública do destinatário (K_u)
  - Enviado pela rede para o destinatário



Transmissão de chave simétrica

Transmissão de chave simétrica K

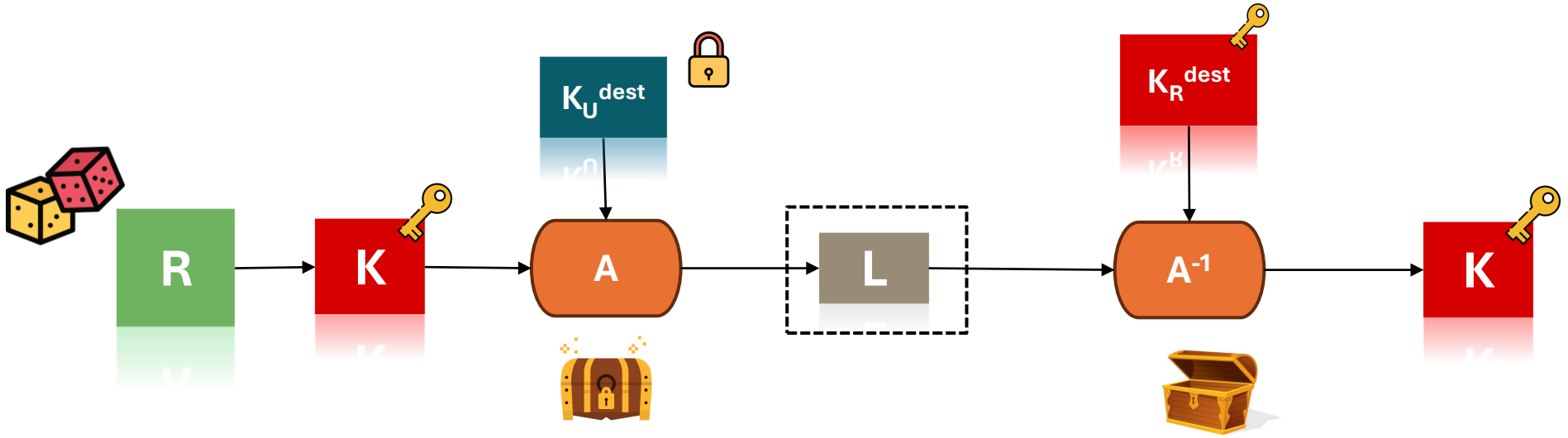
- R: número aleatório [fonte de entropia] gera chave K
- L: chave K protegida por chave pública do destinatário (K_u)
  - Enviado pela rede para o destinatário
- Apenas dono da chave K_R pode recuperar K
- Utilidade: cifras simétricas são mais eficientes



Transmissão de chave simétrica

Transmissão de chave simétrica K

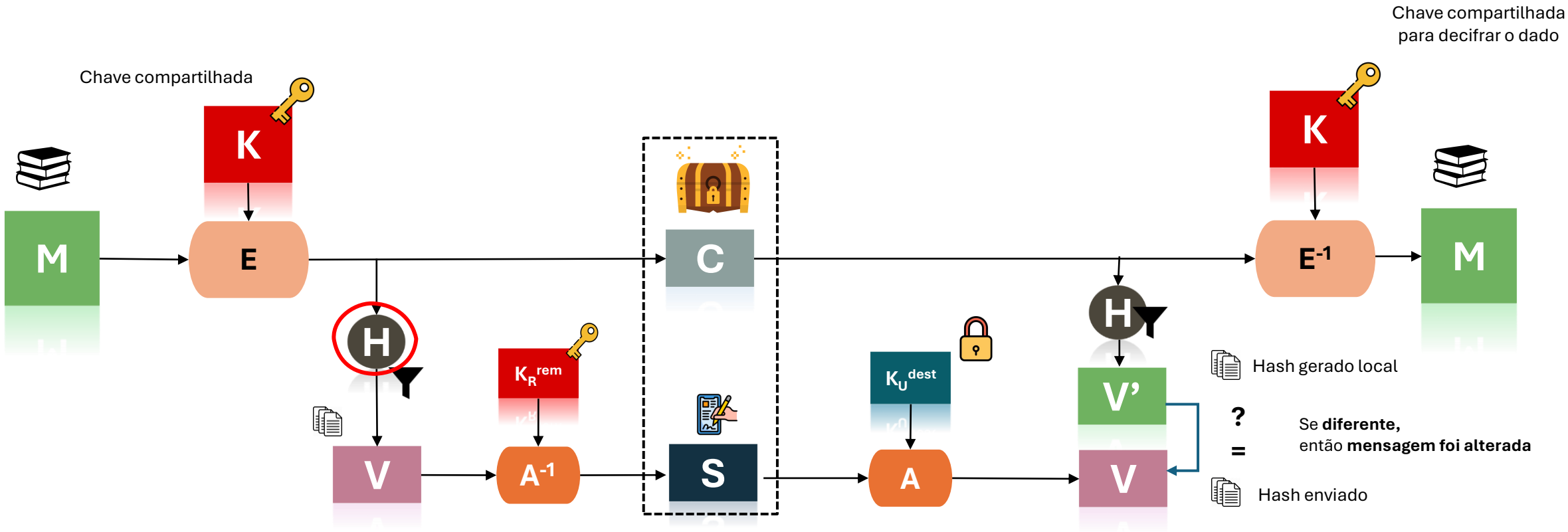
- R: número aleatório [fonte de entropia] gera chave K
- L: chave K protegida por chave pública do destinatário (K_u)
  - Enviado pela rede para o destinatário
- Apenas dono da chave K_R pode recuperar K
- Utilidade: cifras simétricas são mais eficientes



Envelope criptográfico

Mensagem confidencial (C) e Assinada (S)

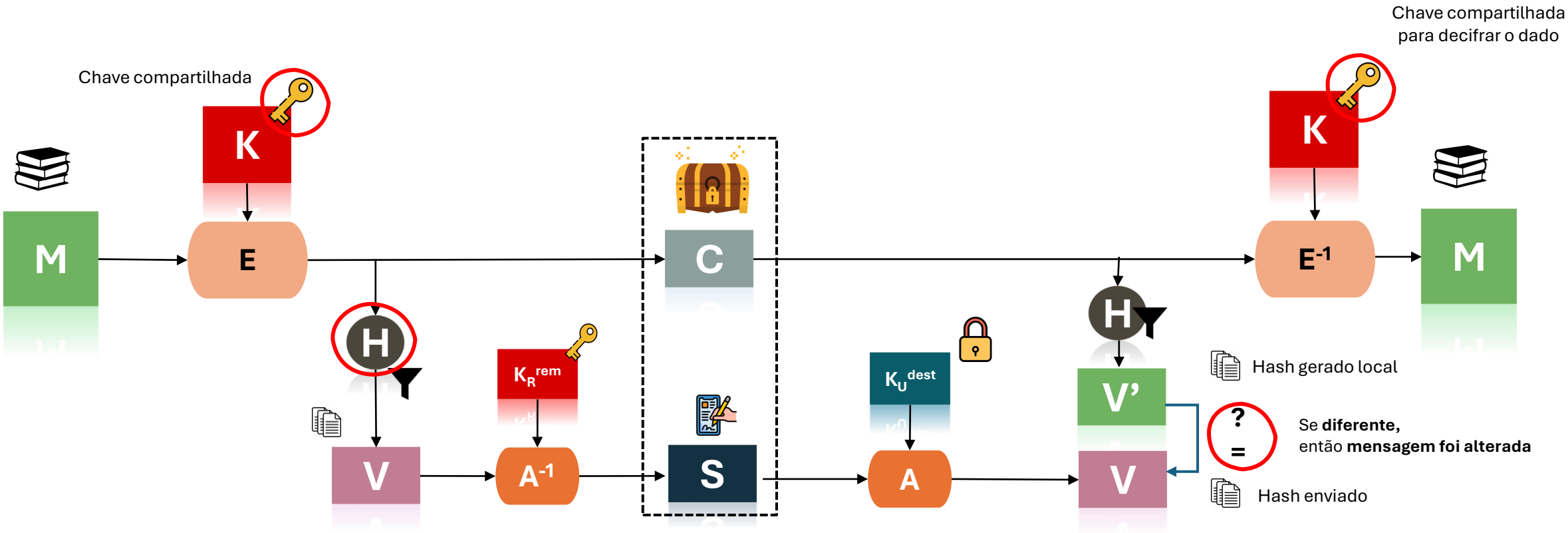
- Serviços: confidencialidade (cifra simétrica), integridade, autenticidade e irretratabilidade (assinatura digital)
- Utilidade: mais eficiente assinar hash das mensagens.



Envelope criptográfico

Mensagem confidencial (C) e Assinada (S)

- Serviços: confidencialidade (cifra simétrica), integridade, autenticidade e irretratabilidade (assinatura digital)
- Utilidade: mais eficiente assinar hash das mensagens.



INSTITUTO FEDERAL São Paulo Campus Campinas

Algoritmos assimétricos: exemplos

Protocolos para **estabelecimento de chaves simétricas**:

- Diffle-Helmman clássico (DH), com curvas elípticas (ECDH) ou com isogenias supersingulares (*SIDH*);

Protocolos para gerar **assinaturas digitais**:

- RSA, (EC)DS, EdDSA, *NTRUSign*

Protocolos para **cifração assimétrica**:

- RSA, ECIES, *McEliece*, *NTRUEncrypt*

- Obs.: algoritmos em *itálico* são propostas que resistem a ataques feitos com **computadores quânticos**.

Algoritmos assimétricos: exemplos

Protocolos para **estabelecimento de chaves simétricas**:

- Diffle-Helmman clássico (DH), com curvas elípticas (ECDH) ou com isogenias supersingulares (*SIDH*);

Protocolos para gerar **assinaturas digitais**:

- RSA, (EC)DS, EdDSA, *NTRUSign*

Protocolos para **cifração assimétrica**:

- RSA, ECIES, *McEliece*, *NTRUEncrypt*

- Obs.: algoritmos em *itálico* são propostas que resistem a ataques feitos com **computadores quânticos**.

INSTITUTO FEDERAL São Paulo Campus Campinas

Algoritmos assimétricos: exemplos

Protocolos para **estabelecimento de chaves simétricas**:

- Diffle-Helmman clássico (DH), com curvas elípticas (ECDH) ou com isogenias supersingulares (*SIDH*);

Protocolos para gerar **assinaturas digitais**:

- RSA, (EC)DS, EdDSA, *NTRUSign*

Protocolos para **cifração assimétrica**:

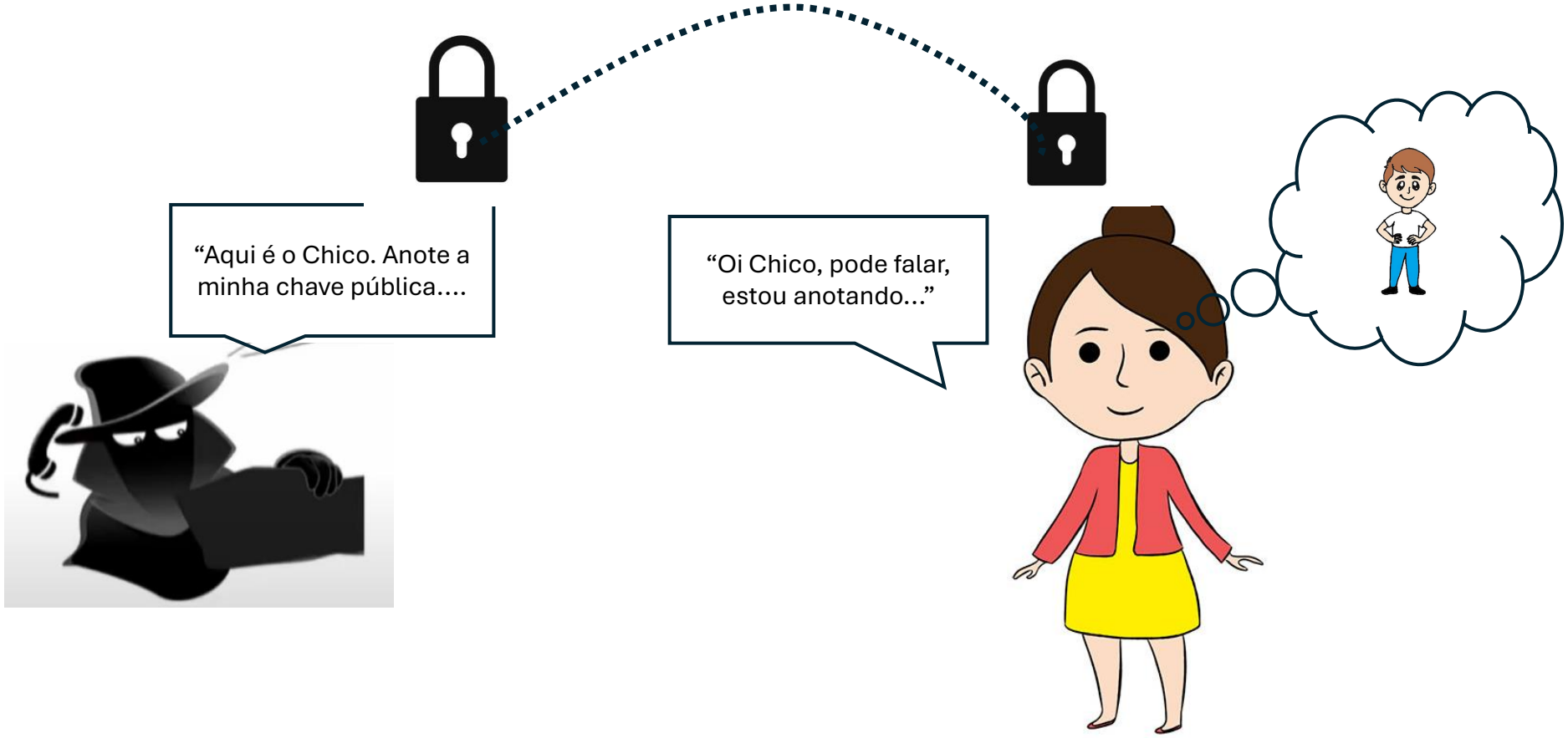
- RSA, ECIES, *McEliece*, *NTRUEncrypt*

- Obs.: algoritmos em *itálico* são propostas que resistem a ataques feitos com **computadores quânticos**.

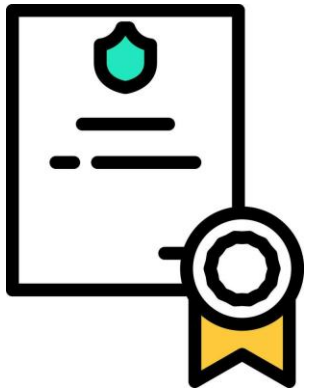
INSTITUTO FEDERAL
São Paulo
Campus Campinas

Certificados digitais

Distribuição de chaves públicas



Certificados digitais

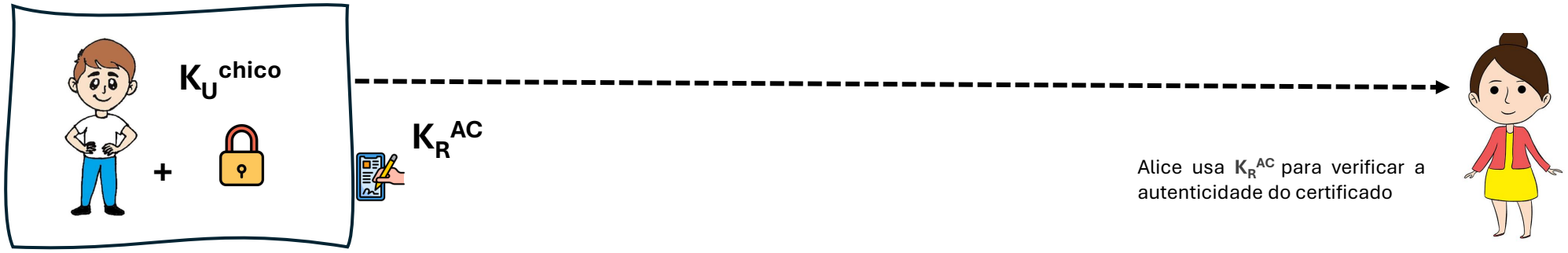


Associam chave pública a seu dono (cartório digital):

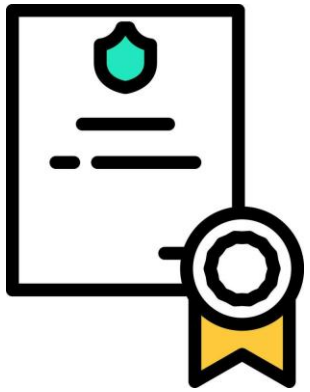
- Atestado dizendo qual é a chave pública de Chico

Modelo PKI: certificado contém chave pública de Chico assinada por uma **autoridade certificadora (AC)**

- **Premissa:** chave pública AC é amplamente conhecida.
- Na prática, **certificados das ACs são pré-estabelecidos** em sistemas computacionais, como navegadores web.
  - Também podem ser instalados por usuário



Certificados digitais

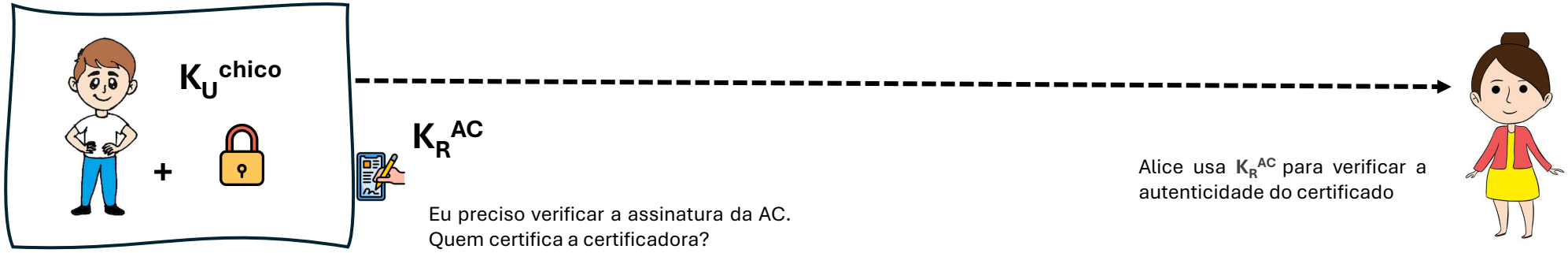


Associam chave pública a seu dono:

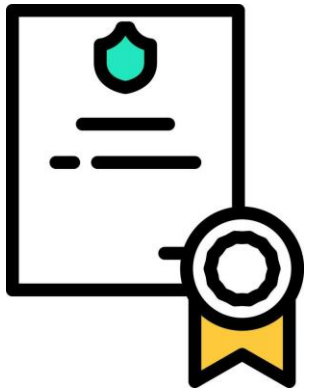
- Atestado dizendo qual é a chave pública de Chico

Modelo PKI: certificado contém chave pública de Chico assinada por uma **autoridade certificadora (AC)**

- **Premissa:** chave pública AC é amplamente conhecida.
- Na prática, **certificados das ACs são pré-estabelecidos** em sistemas computacionais, como navegadores web.
  - Também podem ser instalados por usuário



Certificados digitais

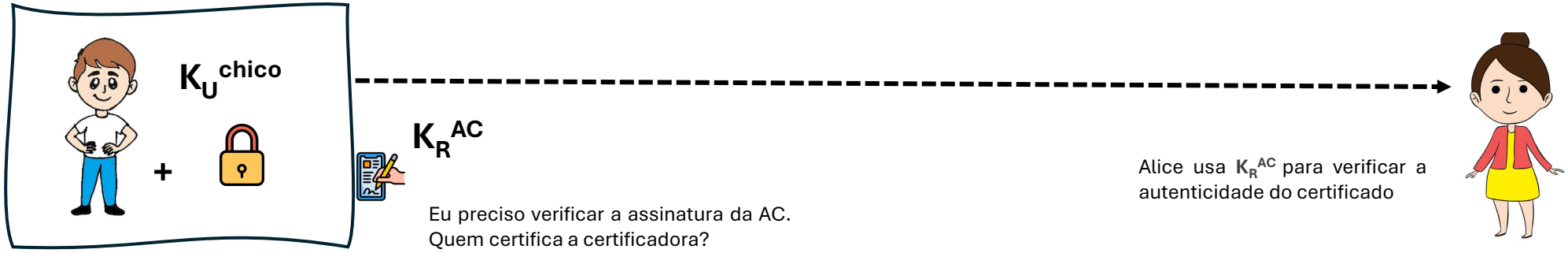


Associam chave pública a seu dono:

- Atestado dizendo qual é a chave pública de Chico

Modelo PKI: certificado contém chave pública de Chico assinada por uma **autoridade certificadora (AC)**

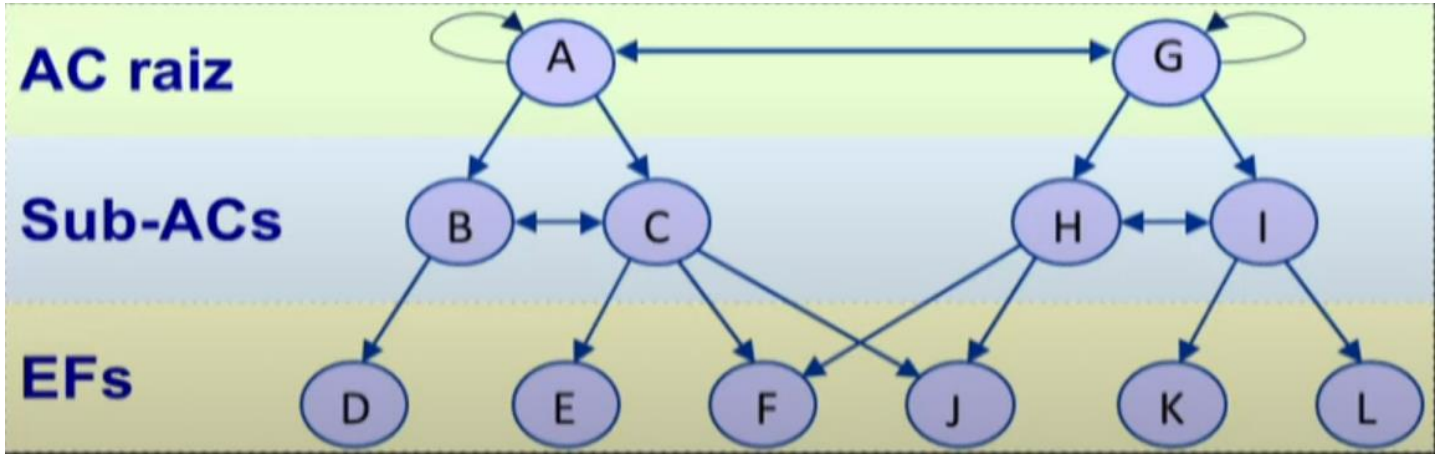
- **Premissa:** chave pública AC é amplamente conhecida.
- Na prática, **certificados das ACs são pré-estabelecidos** em sistemas computacionais, como navegadores web.
  - Também podem ser instalados por usuário



Certificados digitais: ICP

Modelo ICP (Infraestrutura de Chaves Públicas), ou PKI (*Public Key Infrastructure*): **cadeias de certificação**.

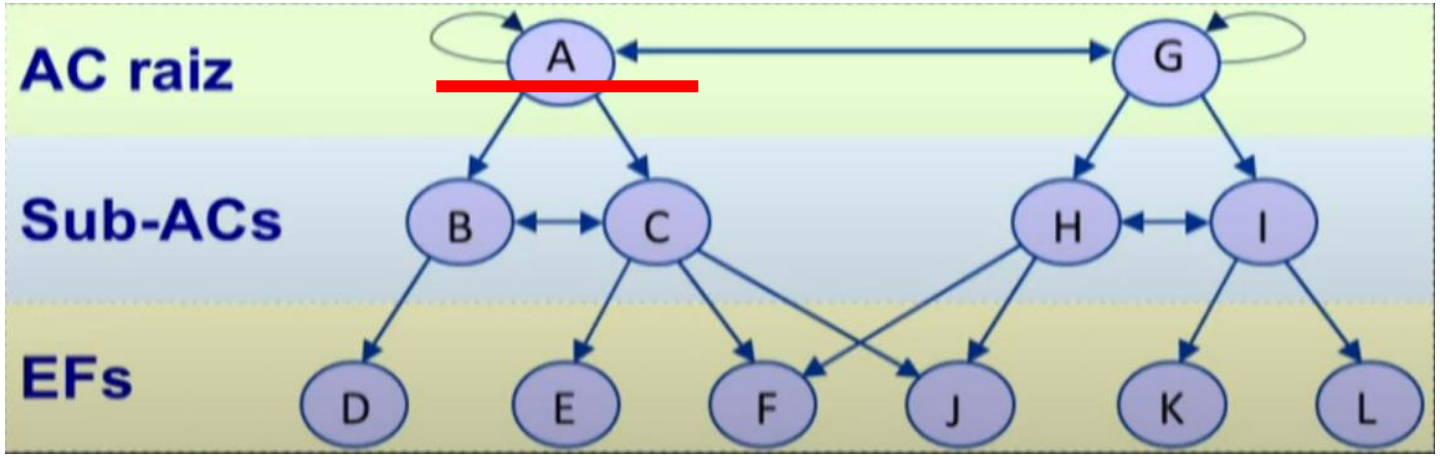
- Usa chave no certificado da AC raiz (autoassinado) para assinar outras chaves na cadeia, até entidades finais (EFs)
- Proteção das chaves mais críticas (mais próximas da raiz)
- ACs dedicadas a **vários fins**



Certificados digitais: ICP

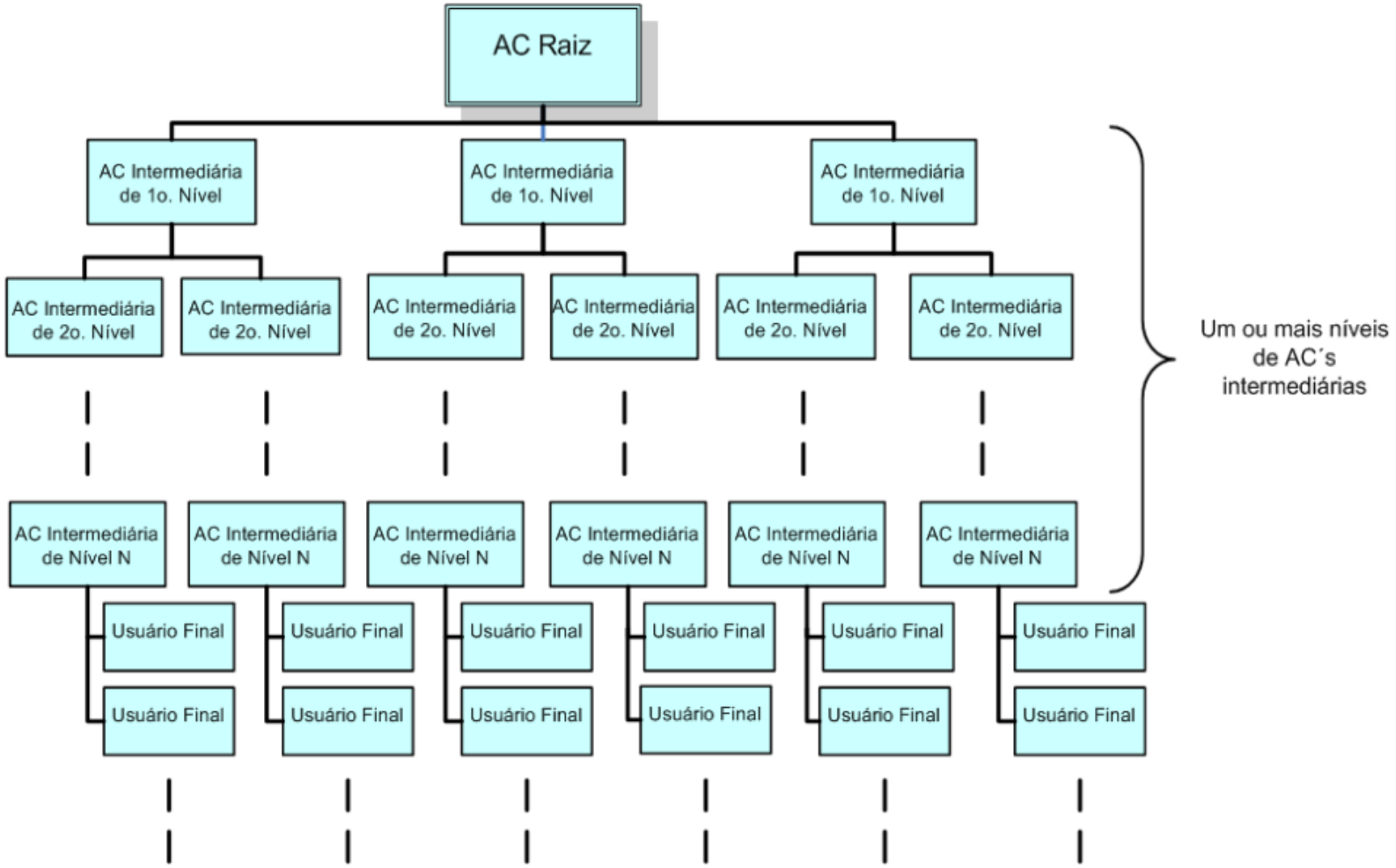
Modelo ICP (Infraestrutura de Chaves Públicas), ou PKI (*Public Key Infrastructure*): **cadeias de certificação**.

- Usa chave no certificado da AC raiz (autoassinado) para assinar outras chaves na cadeia, até entidades finais (EFs)
- Proteção das chaves mais críticas (mais próximas da raiz)
- ACs dedicadas a **vários fins**

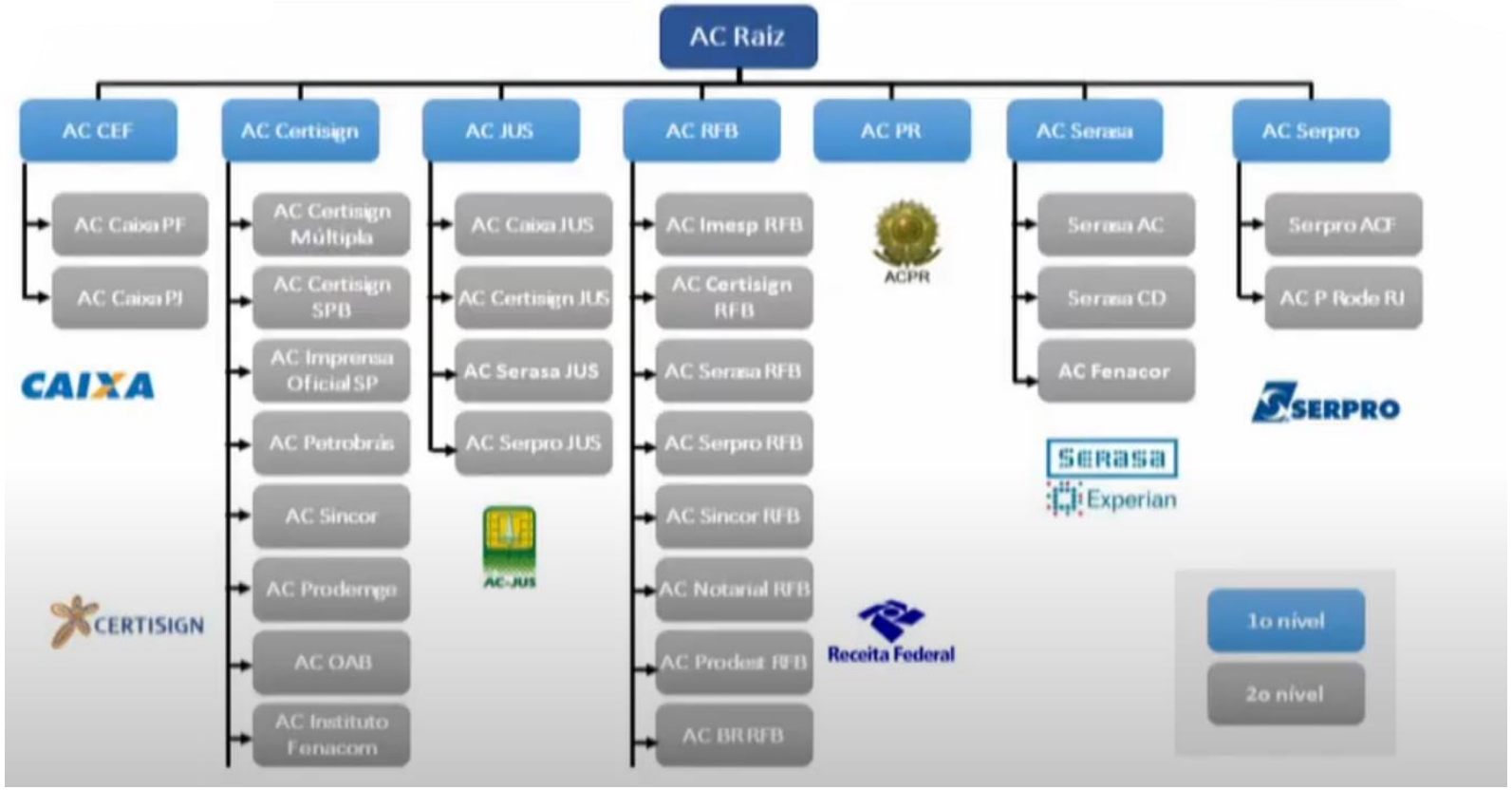


INSTITUTO FEDERAL
São Paulo
Campus Campinas

Certificados digitais: ICP



Certificados digitais: ICP



ICP BRASIL – Disponível em: https://www.gov.br/iti/pt-br/assuntos/icp-brasil
Ecossistema: https://www.gov.br/iti/pt-br/assuntos/icp-brasil/EcossistemaICPBrasil_240822.pdf
Distribuição digital – Disponível em: https://numeros.iti.gov.br/
Manual de condutas: https://www.gov.br/iti/pt-br/central-de-conteudo/lea-mct-11-voli-v1-0-pdf

Processo de certificação

A **verificação de identidade** é comumente delegada pelas ACs a entidades confiáveis: **autoridades de registro (RAs)**

- As **entidades finais (EFs)** apresentam suas chaves públicas e demonstram sua identidade, por meios legais extra criptográficos, às ARs.
- Dados a serem comprovados: dependem de legislação cabível e políticas da AC.
  - Ex.: sites web → provar que é dono do domínio.

Serviço descentralizado: por exemplo, distribuído geograficamente.

Processo de certificação

Autoridade de Registro (RA)

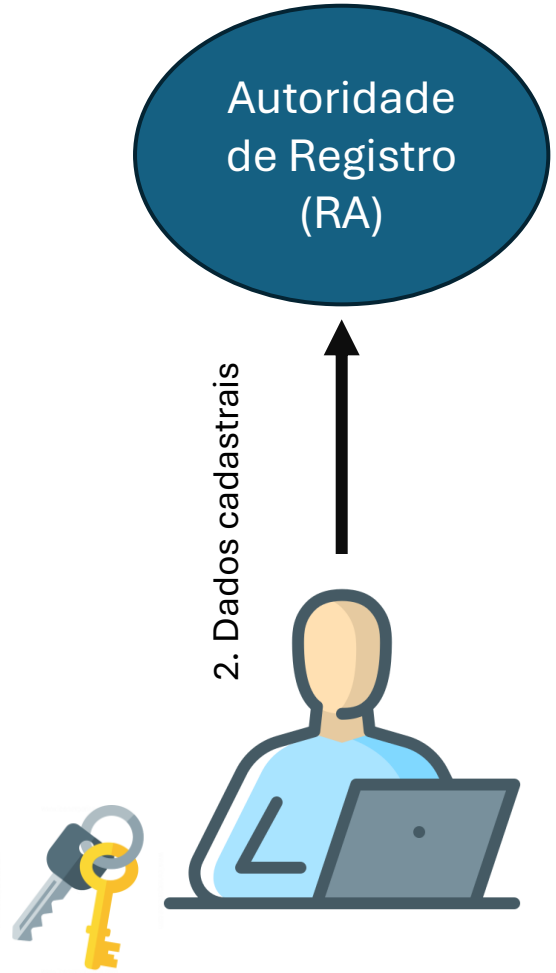
Ex.: Verisign, ICP-BRASIL

Autoridade Certificadora (CA)

Diretório

1. Chaves públicas e privadas

Processo de certificação



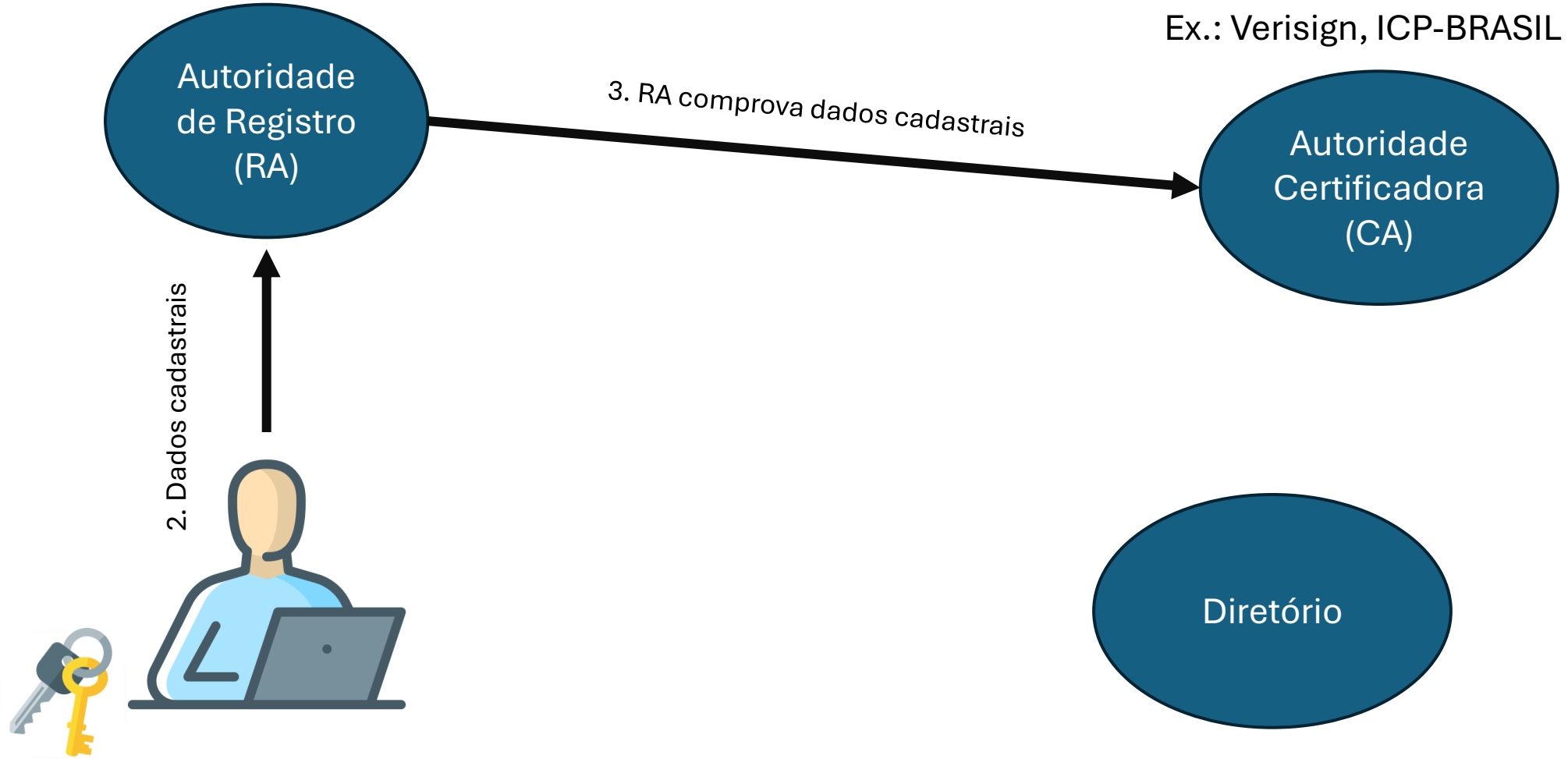
Ex.: Verisign, ICP-BRASIL

Autoridade Certificadora (CA)

Diretório

1. Chaves públicas e privadas

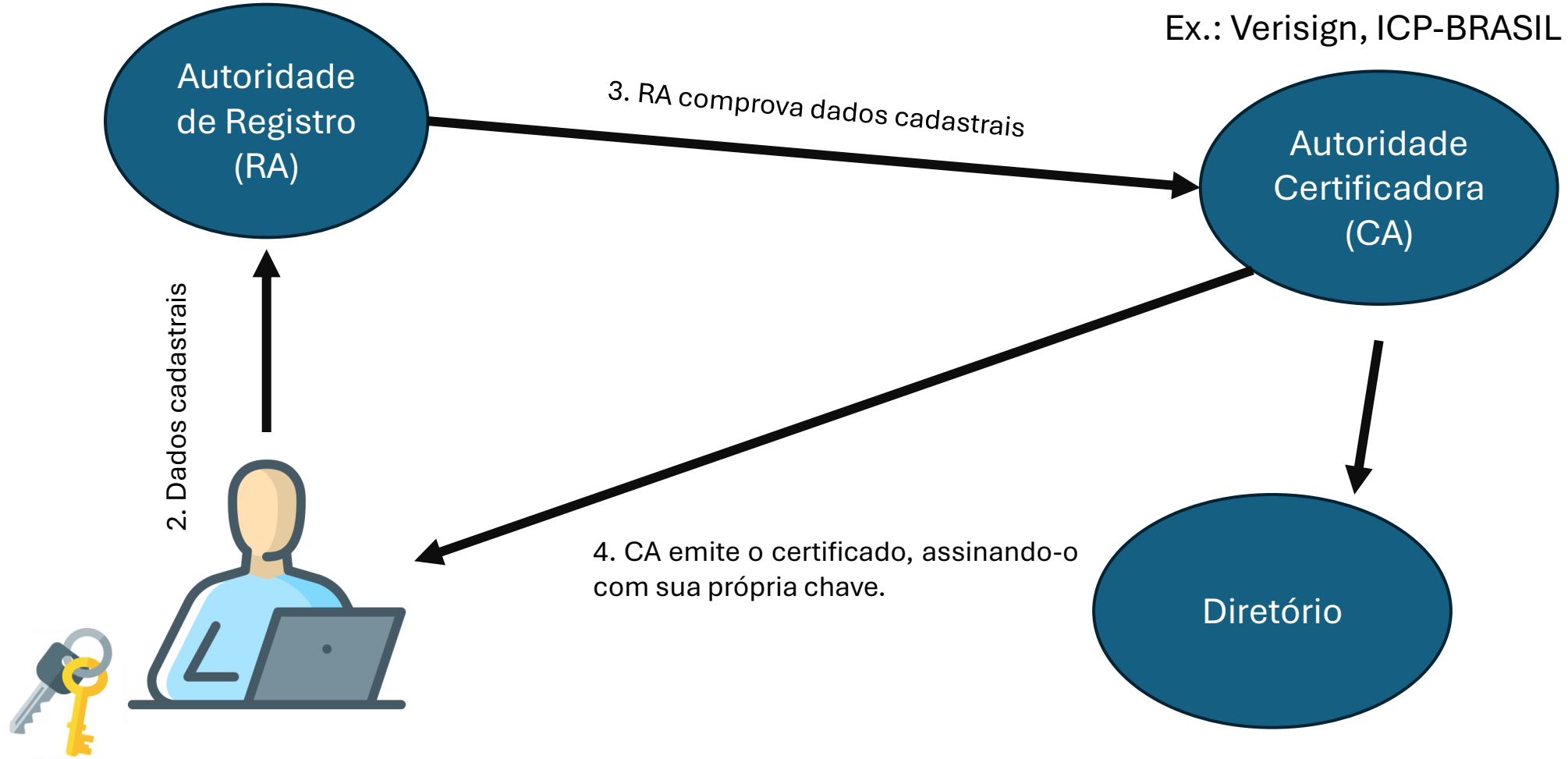
Processo de certificação



1. Chaves públicas e privadas

Processo de certificação

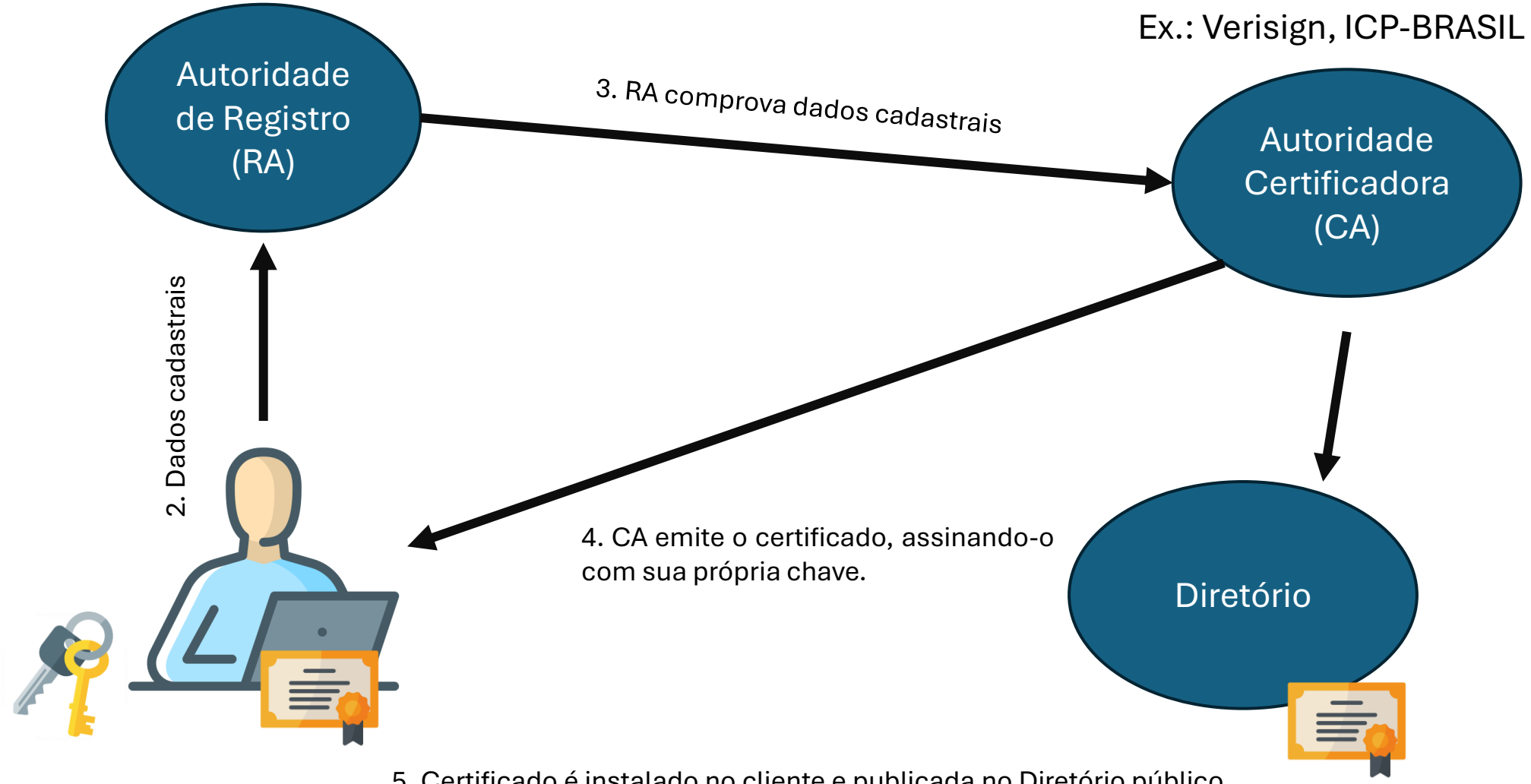
Ex.: Verisign, ICP-BRASIL



1. Chaves públicas e privadas

Processo de certificação

Ex.: Verisign, ICP-BRASIL



1. Chaves públicas e privadas

5. Certificado é instalado no cliente e publicada no Diretório público

Certificação: revogada

Comprometimento da chave privada do usuário (ou, em um caso extremo, da AC).

Alternativas

- **Offline**: *Certificate Revocation List (CRL)*
  - "Lista proibida" emitida e assinada por AC, distribuída periodicamente.
  - S.O possui essa lista em suas atualizações (Windows).
  - Enumera identificadores (#serial) de certificados revogados não expirados e datas de revogação.

- **Online**: *Online Certificate Status Protocol (OSCP)*
  - Protocolo web para consulta do status de certificados
  - Resposta assinada: "good", "revoked" ou "unkown"

INSTITUTO FEDERAL
São Paulo
Campus Campinas

Um exemplo prática de criptografia assimétrica:

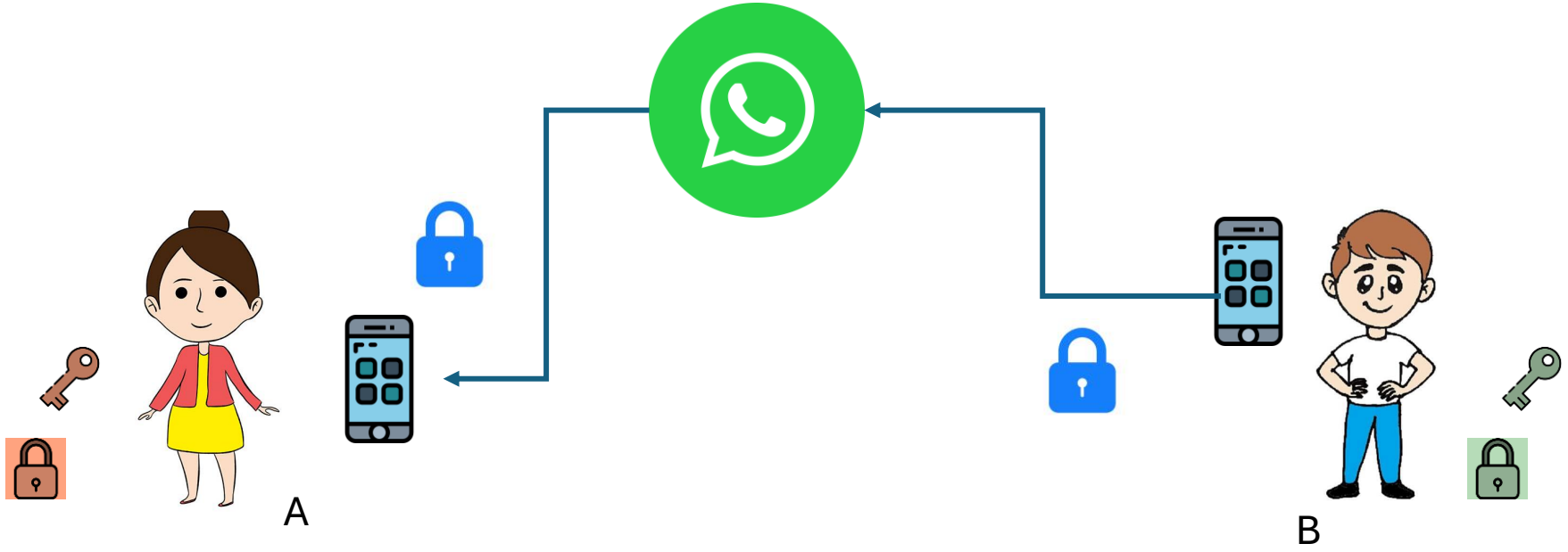
WhatsApp

Como funciona no WhatsApp (protocolo Signal):

- **Criptografia assimétrica (chaves públicas/privadas)**
  - Usada no início da comunicação, para troca de chaves de forma segura entre os dispositivos.
  - Cada usuário tem um par de chaves (pública e privada).
- **Criptografia simétrica (mestra do tráfego)**
  - Depois que as chaves foram trocadas, a comunicação em si (mensagens, áudios, vídeos) é criptografada com chaves simétricas temporárias (chaves de sessão).
  - Essas chaves mudam constantemente (ratchet), o que aumenta a segurança.

Um exemplo prática de criptografia assimétrica:

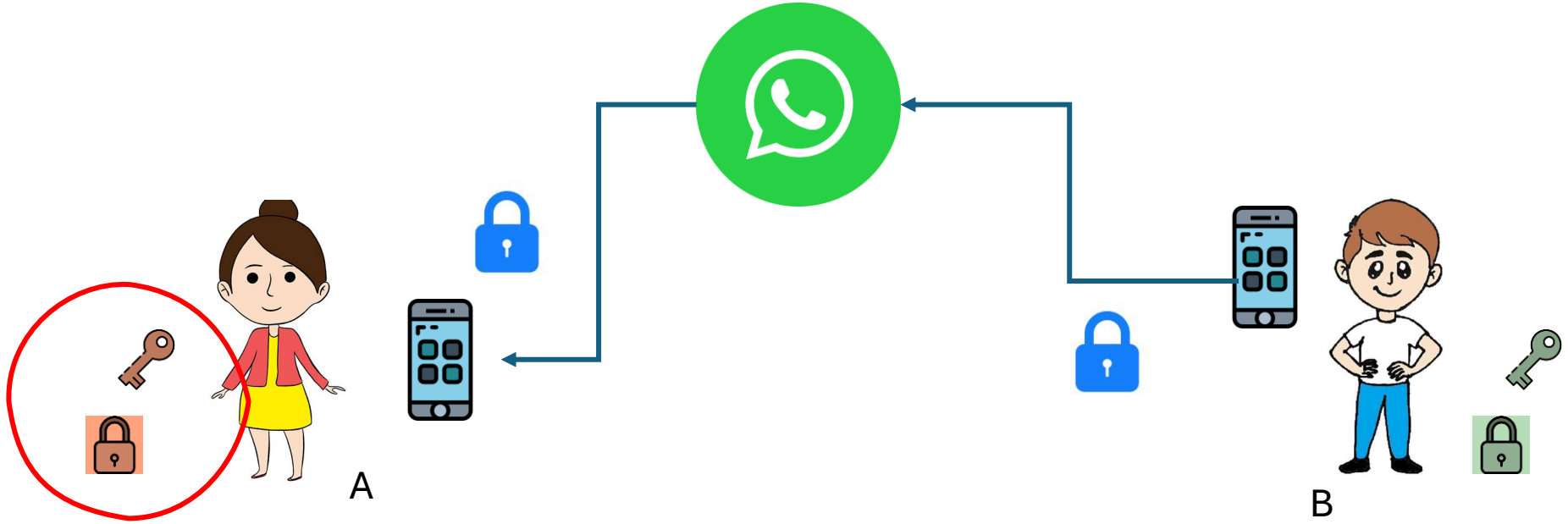
WhatsApp



- **Assimétrica:** estabelecer a confiança e trocar chaves.
- **Simétrica:** usada no tráfego real das mensagens (pois é mais rápida e eficiente).

Um exemplo prática de criptografia assimétrica:

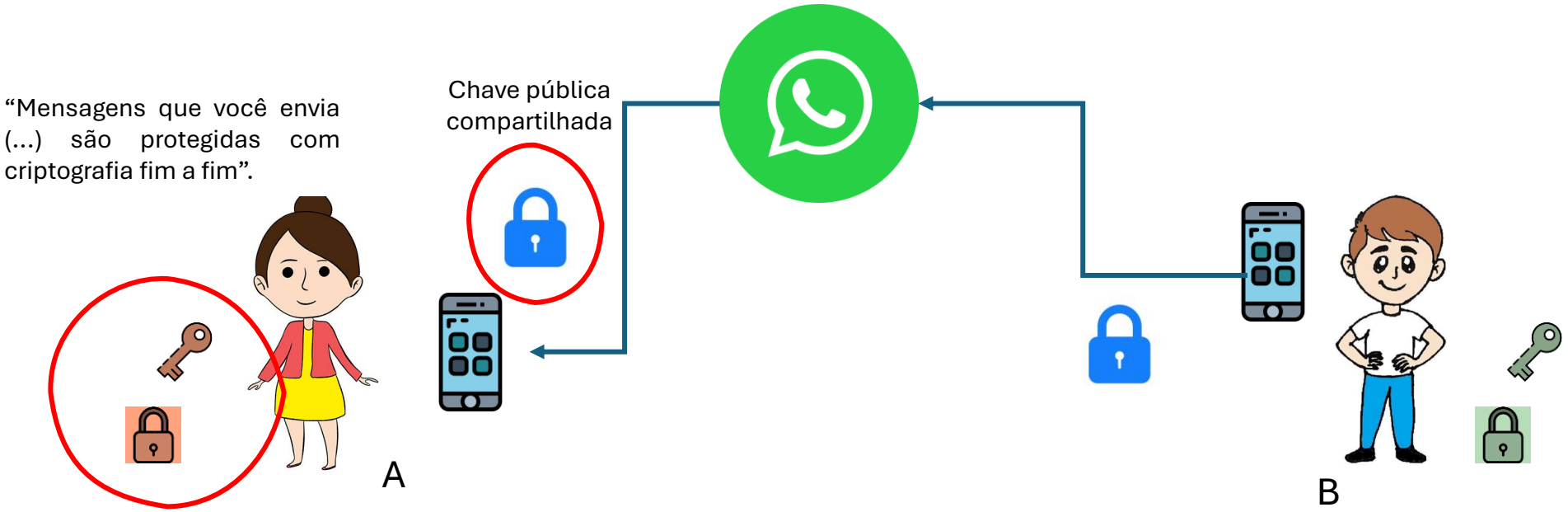
WhatsApp



- **Assimétrica:** estabelecer a confiança e trocar chaves.
- **Simétrica:** usada no tráfego real das mensagens (pois é mais rápida e eficiente).

Um exemplo prática de criptografia assimétrica:

WhatsApp



- **Assimétrica:** estabelecer a confiança e trocar chaves.
- **Simétrica:** usada no tráfego real das mensagens (pois é mais rápida e eficiente).

Um exemplo prática de criptografia assimétrica:

WhatsApp

Novidade? Serviços similares existem há décadas

- Ex.: SIMME, para e-mails, é de 1999

INSTITUTO FEDERAL
São Paulo
Campus Campinas

Dúvidas?

INSTITUTO FEDERAL São Paulo Campus Campinas

Referências Bibliográficas

SÊMOLA, Marcos. *Gestão da segurança da informação: uma visão executiva*. 2. ed., 8. tiragem. Rio de Janeiro: [s.n.], 2018.

SILVA, Pedro Tavares; CARVALHO, Hugo; TORRES, Catarina Botelho. *Segurança dos sistemas de informação: gestão estratégica da segurança empresarial*. 1. ed. Lisboa; V. N. Famalicão: Centro Atlântico, 2003. ISBN 972-8426-66-6.

STALLINGS, William; BROWN, Lawrie. Segurança de computadores: princípios e práticas. 2. ed. Tradução Arlete Simille Marques. Rio de Janeiro: Elsevier, 2014. ISBN 978-85-352-6449-4.

STALLINGS, William. Criptografia e segurança de redes: princípios e práticas. Tradução de Daniel Vieira; revisão técnica de Paulo Sérgio Licciardi Messeder Barreto e Rafael Misoczki. 6. ed. São Paulo: Pearson Education do Brasil, 2015. ISBN 978-85-430-1450-0.