Técnicas criptográficas modernas algoritmos e protocolos

Ricardo Dahab e Julio López

Instituto de Computação - UNICAMP

Roteiro

Introdução

Técnicas criptográficas

Protocolos criptográficos

Outros paradigmas

Introdução

Introdução

Sobre este curso

- A Criptografia moderna se ocupa muito menos de sigilo do que há meros trinta anos atrás, quando justificava plenamente a etimologia da palavra criptografia, cuja origem grega significa *escrita oculta*.

Sobre este curso

- A Criptografia moderna se ocupa muito menos de sigilo do que há meros trinta anos atrás, quando justificava plenamente a etimologia da palavra criptografia, cuja origem grega significa *escrita oculta*.
- Hoje, técnicas criptográficas são maciçamente empregadas na prevenção de incidentes de segurança.

Sobre este curso

- A Criptografia moderna se ocupa muito menos de sigilo do que há meros trinta anos atrás, quando justificava plenamente a etimologia da palavra criptografia, cuja origem grega significa *escrita oculta*.
- Hoje, técnicas criptográficas são maciçamente empregadas na prevenção de incidentes de segurança.
- Aplicações e sistemas que tenham requisitos como sigilo, autenticação, integridade, não-repúdio e anonimato empregam técnicas criptográficas em algum nível de sua arquitetura.

Sobre este curso

O objetivo fundamentais da segurança da informação, é dar uma visão panorâmica das técnicas criptográficas atuais mais importantes para a consecução de requisitos como sigilo, autenticação e integridade, dos quais dependem, outros requisitos de segurança.

Sobre este curso

O objetivo fundamentais da segurança da informação, é dar uma visão panorâmica das técnicas criptográficas atuais mais importantes para a consecução de requisitos como sigilo, autenticação e integridade, dos quais dependem, outros requisitos de segurança.

- O **sigilo** de mensagens, ou de identidades, pode ser necessário a uma aplicação ou auxiliar a consecução de outro requisito, como a autenticação.

Sobre este curso

O objetivo fundamentais da segurança da informação, é dar uma visão panorâmica das técnicas criptográficas atuais mais importantes para a consecução de requisitos como sigilo, autenticação e integridade, dos quais dependem, outros requisitos de segurança.

- O **sigilo** de mensagens, ou de identidades, pode ser necessário a uma aplicação ou auxiliar a consecução de outro requisito, como a autenticação.
- A **autenticação** de propriedades de mensagens (e.g. sua integridade e origem) e de entidades (e.g. sua identidade) pode ser um fim em si ou pode ancorar a obtenção de outros fins como o estabelecimento de uma chave criptográfica.

Modelo de segurança

Modelo de segurança

- O nosso modelo básico de comunicação supõe duas entidades, Alice e Beto, trocando mensagens transmitidas num canal inseguro; isto é, um canal passível de leitura e escrita por um intruso, Ivo.

Modelo de segurança

- O nosso modelo básico de comunicação supõe duas entidades, Alice e Beto, trocando mensagens transmitidas num canal inseguro; isto é, um canal passível de leitura e escrita por um intruso, Ivo.
- Os métodos de Ivo podem ser a simples escuta, um "grampo", que chamamos de ataque passivo, ou até a modificação, repetição e injeção de mensagens com objetivos variados como, por exemplo, passar-se por Alice ou Beto para obter acesso a serviços não autorizados; esses são os chamados ataques ativos. Passivos ou ativos, esses ataques representam ameaça aos requisitos de segurança que discutimos acima.

Modelo de segurança

- O nosso modelo básico de comunicação supõe duas entidades, Alice e Beto, trocando mensagens transmitidas num canal inseguro; isto é, um canal passível de leitura e escrita por um intruso, Ivo.
- Os métodos de Ivo podem ser a simples escuta, um "grampo", que chamamos de ataque passivo, ou até a modificação, repetição e injeção de mensagens com objetivos variados como, por exemplo, passar-se por Alice ou Beto para obter acesso a serviços não autorizados; esses são os chamados ataques ativos. Passivos ou ativos, esses ataques representam ameaça aos requisitos de segurança que discutimos acima.
- As técnicas criptográficas para prevenir tais ataques vêm de duas vertentes, a simétrica e a assimétrica, usadas isoladamente ou em conjunto.

Criptografia simétrica

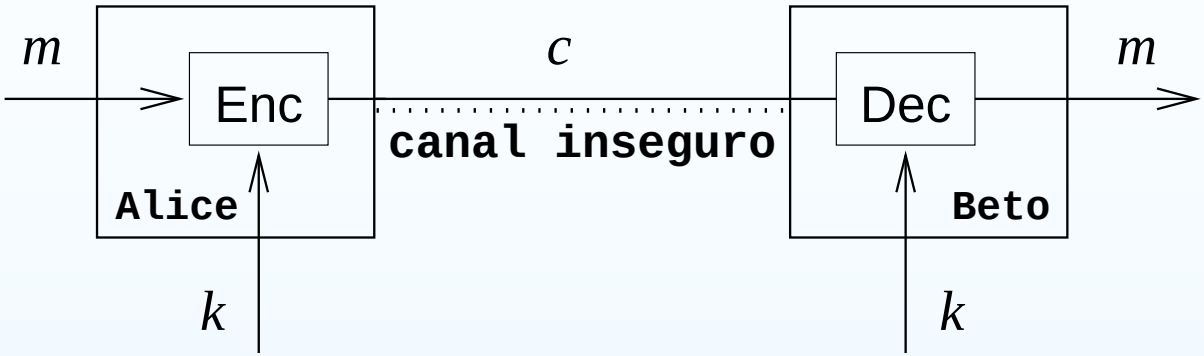


Figura 3.1 Modelo Simétrico

Criptografia simétrica

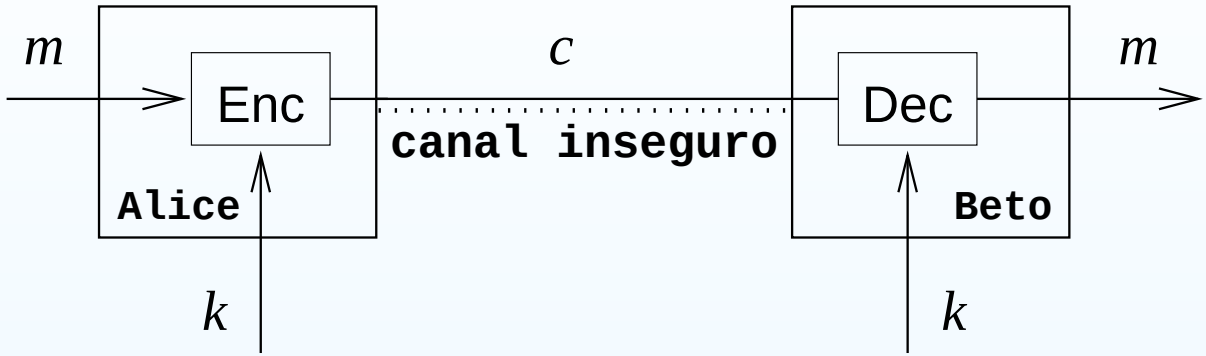


Figura 3.1 Modelo Simétrico

- Alice e Beto desejam trocar mensagens m (*texto claro*) em sigilo;

Criptografia simétrica

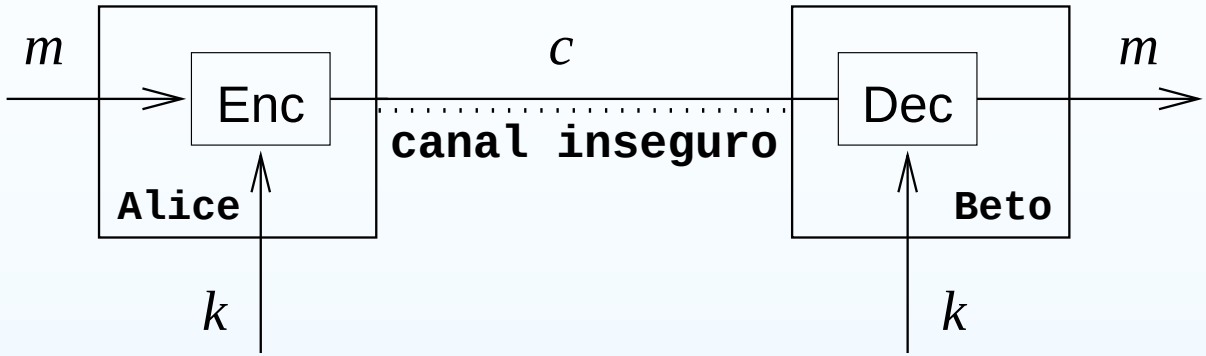


Figura 3.1 Modelo Simétrico

- Alice e Beto desejam trocar mensagens m (*texto claro*) em sigilo;
- Alice aplica uma *função (ou algoritmo) de encriptação* $\text{ENC}_k(m)$, que transforma m numa *mensagem encriptada* ou *texto encriptado* c, sob a ação da *chave* k.

Criptografia simétrica

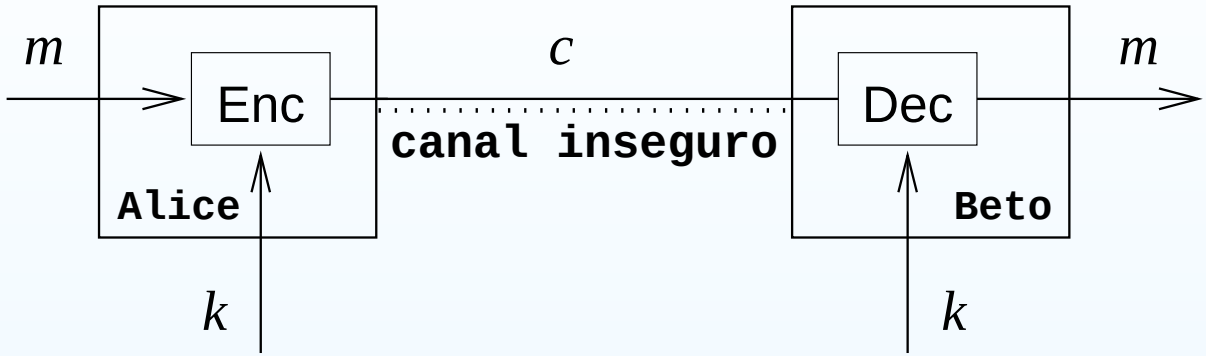


Figura 3.1 Modelo Simétrico

- Alice e Beto desejam trocar mensagens m (*texto claro*) em sigilo;
- Alice aplica uma *função (ou algoritmo) de encriptação* $\textsc{Enc}_k(m)$, que transforma m numa *mensagem encriptada* ou *texto encriptado* c, sob a ação da *chave* k.

- Ao receber c, Beto aplica a função de *decriptação* $\textsc{Dec}_k(c)$, recuperando m.

Criptografia simétrica

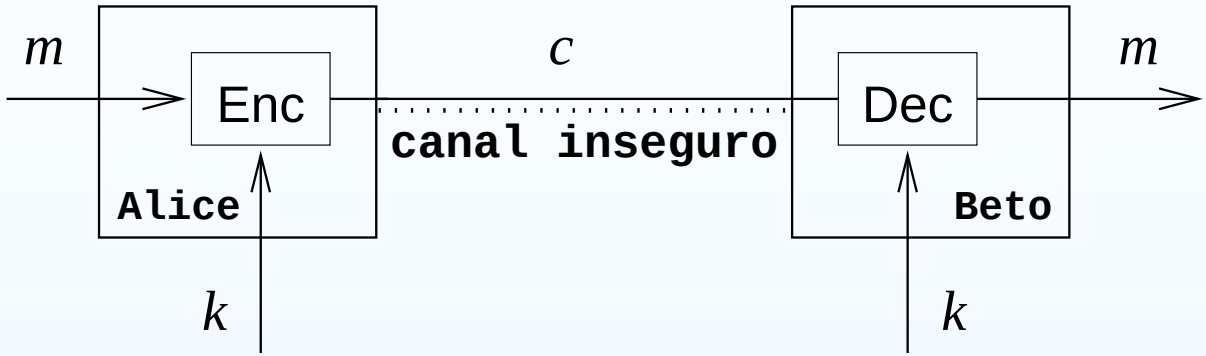


Figura 3.1 Modelo Simétrico

- O objetivo é produzir um texto c que não guarde relação alguma com m.

Criptografia simétrica

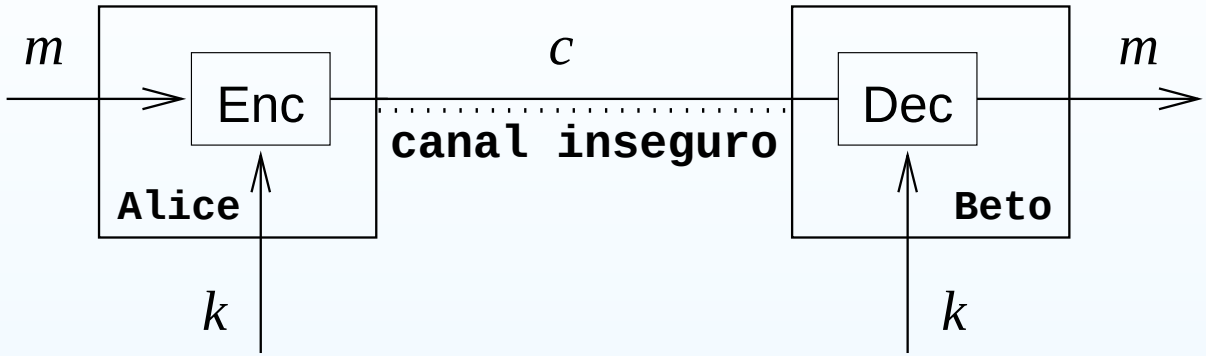


Figura 3.1 Modelo Simétrico

- O objetivo é produzir um texto c que não guarde relação alguma com m.
- A inclusão da chave k no processo tem o objetivo de dar o poder de transformar c em m apenas a quem conhece k; isto é, prover sigilo na transmissão de m.

Criptografia simétrica - exemplo

Criptografia simétrica - exemplo

- Um exemplo simples de encriptação simétrica consiste em substituir cada letra de um texto pela letra k posições à frente no alfabeto (supomos que após 'z' vem 'a').

Criptografia simétrica - exemplo

- Um exemplo simples de encriptação simétrica consiste em substituir cada letra de um texto pela letra k posições à frente no alfabeto (supomos que após 'z' vem 'a').
- Para $k = 5$, a palavra `alabastro` se transforma em `fqfgfxywt`.

Criptografia simétrica - exemplo

- Um exemplo simples de encriptação simétrica consiste em substituir cada letra de um texto pela letra k posições à frente no alfabeto (supomos que após 'z' vem 'a').
- Para $k = 5$, a palavra `alabastro` se transforma em `fqfgfxywt`.
- A chave, neste caso, é k. Esse é o chamado *método da substituição monoalfabética*.

Criptografia simétrica - exemplo

- Um exemplo simples de encriptação simétrica consiste em substituir cada letra de um texto pela letra k posições à frente no alfabeto (supomos que após 'z' vem 'a').
- Para $k = 5$, a palavra `alabastro` se transforma em `fqfgfxywt`.
- A chave, neste caso, é k. Esse é o chamado *método da substituição monoalfabética*.
- Em vez de uma só letra substituindo outra, podemos ter uma lista de letras usadas em seqüência. Essa é a *substituição polialfabética*.

Criptografia simétrica - premissas

Criptografia simétrica - premissas

- $\text{ENC}_{\cdot}(.)$ deve ser projetada de forma que seja muito difícil para Ivo calcular m a partir de c sem conhecimento de k, ainda que $\text{ENC}_{\cdot}(.)$ seja pública e Ivo use computadores.

Criptografia simétrica - premissas

- $\text{ENC}_{\cdot}(.)$ deve ser projetada de forma que seja muito difícil para Ivo calcular m a partir de c sem conhecimento de k, ainda que $\text{ENC}_{\cdot}(.)$ seja pública e Ivo use computadores.
- Dizemos que $\text{ENC}_k(.)$ deve ser uma função *unidirecional* para cada valor fixo de k; isto é, que $\text{ENC}_k(.)$ seja fácil de calcular, mas $\text{ENC}_k(.)^{-1}$, ou seja, $\text{DEC}_k(.)$, seja muito difícil de calcular sem o conhecimento da chave k.

Criptografia simétrica - premissas

- $\text{ENC}_{\cdot}(.)$ deve ser projetada de forma que seja muito difícil para Ivo calcular m a partir de c sem conhecimento de k, ainda que $\text{ENC}_{\cdot}(.)$ seja pública e Ivo use computadores.
- Dizemos que $\text{ENC}_k(.)$ deve ser uma função *unidirecional* para cada valor fixo de k; isto é, que $\text{ENC}_k(.)$ seja fácil de calcular, mas $\text{ENC}_k(.)^{-1}$, ou seja, $\text{DEC}_k(.)$, seja muito difícil de calcular sem o conhecimento da chave k.
-

Criptografia simétrica - premissas

- $\text{ENC}_{\cdot}(.)$ deve ser projetada de forma que seja muito difícil para Ivo calcular m a partir de c sem conhecimento de k, ainda que $\text{ENC}_{\cdot}(.)$ seja pública e Ivo use computadores.
- Dizemos que $\text{ENC}_k(.)$ deve ser uma função *unidirecional* para cada valor fixo de k; isto é, que $\text{ENC}_k(.)$ seja fácil de calcular, mas $\text{ENC}_k(.)^{-1}$, ou seja, $\text{DEC}_k(.)$, seja muito difícil de calcular sem o conhecimento da chave k.
- A quantidade de chaves possíveis deve ser muito grande, para evitar uma *busca exaustiva* de k.

Criptografia simétrica - premissas

- $\text{ENC}_{\cdot}(.)$ deve ser projetada de forma que seja muito difícil para Ivo calcular m a partir de c sem conhecimento de k, ainda que $\text{ENC}_{\cdot}(.)$ seja pública e Ivo use computadores.
- Dizemos que $\text{ENC}_k(.)$ deve ser uma função *unidirecional* para cada valor fixo de k; isto é, que $\text{ENC}_k(.)$ seja fácil de calcular, mas $\text{ENC}_k(.)^{-1}$, ou seja, $\text{DEC}_k(.)$, seja muito difícil de calcular sem o conhecimento da chave k.
- A quantidade de chaves possíveis deve ser muito grande, para evitar uma *busca exaustiva* de k.
- Alice e Beto têm que estabelecer a chave k em sigilo antes do seu uso. Essa dificuldade é recorrente. Veremos como essa dificuldade pode ser contornada.

Criptografia simétrica - ataques

Criptografia simétrica - ataques

- Como Ivo conhece a mensagem m' na busca exaustiva, o ataque ao modelo é chamado de *ataque do texto claro conhecido*.

Criptografia simétrica - ataques

- Como Ivo conhece a mensagem m' na busca exaustiva, o ataque ao modelo é chamado de *ataque do texto claro conhecido*.
- Se somente c fosse conhecido, o ataque seria de *texto encriptado somente*.

Criptografia simétrica - ataques

- Como Ivo conhece a mensagem m' na busca exaustiva, o ataque ao modelo é chamado de *ataque do texto claro conhecido*.
- Se somente c fosse conhecido, o ataque seria de *texto encriptado somente*.
- Se Ivo tiver acesso à função $\text{ENC}_k(.)$, por exemplo embutida em algum dispositivo, e puder produzir pares (m', c') à sua escolha, o ataque é de *texto claro escolhido*.

Criptografia simétrica - ataques

- Como Ivo conhece a mensagem m' na busca exaustiva, o ataque ao modelo é chamado de *ataque do texto claro conhecido*.
- Se somente c fosse conhecido, o ataque seria de *texto encriptado somente*.
- Se Ivo tiver acesso à função $\textsc{Enc}_k(.)$, por exemplo embutida em algum dispositivo, e puder produzir pares (m', c') à sua escolha, o ataque é de *texto claro escolhido*.
- Finalmente, se Ivo tiver acesso à função $\textsc{Dec}_k(.)$ e puder produzir pares (m', c') à sua escolha, o ataque é de *texto encriptado escolhido*.

Criptografia simétrica - ataques

- Como Ivo conhece a mensagem m' na busca exaustiva, o ataque ao modelo é chamado de *ataque do texto claro conhecido*.
- Se somente c fosse conhecido, o ataque seria de *texto encriptado somente*.
- Se Ivo tiver acesso à função $\textsc{Enc}_k(.)$, por exemplo embutida em algum dispositivo, e puder produzir pares (m', c') à sua escolha, o ataque é de *texto claro escolhido*.
- Finalmente, se Ivo tiver acesso à função $\textsc{Dec}_k(.)$ e puder produzir pares (m', c') à sua escolha, o ataque é de *texto encriptado escolhido*.

A ciência que se dedica a analisar algoritmos criptográficos em busca de falhas, ou de ”quebrar” tais algoritmos, é a **Criptoanálise**.

Criptografia simétrica - simetria

Criptografia simétrica - simetria

- O adjetivo simétrico é bastante adequado. Tudo que um puder encriptar ou decriptar o outro também pode. Um benefício dessa simetria é a confiança que Alice e Beto têm de que estão trocando mensagens sigilosas um com o outro, e não com Ivo. Por outro lado, não é possível atribuir a um ou a outro a autoria de uma mensagem sem a ajuda de uma terceira parte confiável.

Criptografia simétrica - simetria

- O adjetivo simétrico é bastante adequado. Tudo que um puder encriptar ou decriptar o outro também pode. Um benefício dessa simetria é a confiança que Alice e Beto têm de que estão trocando mensagens sigilosas um com o outro, e não com Ivo. Por outro lado, não é possível atribuir a um ou a outro a autoria de uma mensagem sem a ajuda de uma terceira parte confiável.
- Outras denominações dos sistemas simétricos são *sistemas de chaves secretas* e *sistemas de chaves simétricas*.

Alguns algoritmos simétricos modernos

Alguns algoritmos simétricos modernos

- Data Encryption Standard (DES), 1977.

Alguns algoritmos simétricos modernos

- Data Encryption Standard (DES), 1977.
- Advanced Encryption Standard (AES), 2000.

Alguns algoritmos simétricos modernos

- Data Encryption Standard (DES), 1977.
- Advanced Encryption Standard (AES), 2000.
- NIST (1997-1999): MARS, RC6, Serpent, Twofish.

Alguns algoritmos simétricos modernos

- Data Encryption Standard (DES), 1977.
- Advanced Encryption Standard (AES), 2000.
- NIST (1997-1999): MARS, RC6, Serpent, Twofish.
- NESSIE (2003): MYSTY1, AES, Camellia (ISO 2005).

Alguns algoritmos simétricos modernos

- Data Encryption Standard (DES), 1977.
- Advanced Encryption Standard (AES), 2000.
- NIST (1997-1999): MARS, RC6, Serpent, Twofish.
- NESSIE (2003): MYSTY1, AES, Camellia (ISO 2005).

- CRYPTREC (2002): Camellia, SC2000, Hierocrypt-3, CIPHERUNICORN-A.

Alguns algoritmos simétricos modernos

- Data Encryption Standard (DES), 1977.
- Advanced Encryption Standard (AES), 2000.
- NIST (1997-1999): MARS, RC6, Serpent, Twofish.
- NESSIE (2003): MYSTY1, AES, Camellia (ISO 2005).

- CRYPTREC (2002): Camellia, SC2000, Hierocrypt-3, CIPHERUNICORN-A.
- ECRYPT (European Network of Excellence for Cryptology) 2004-2008: Kasumi (64-128), AES.

Criptografia assimétrica

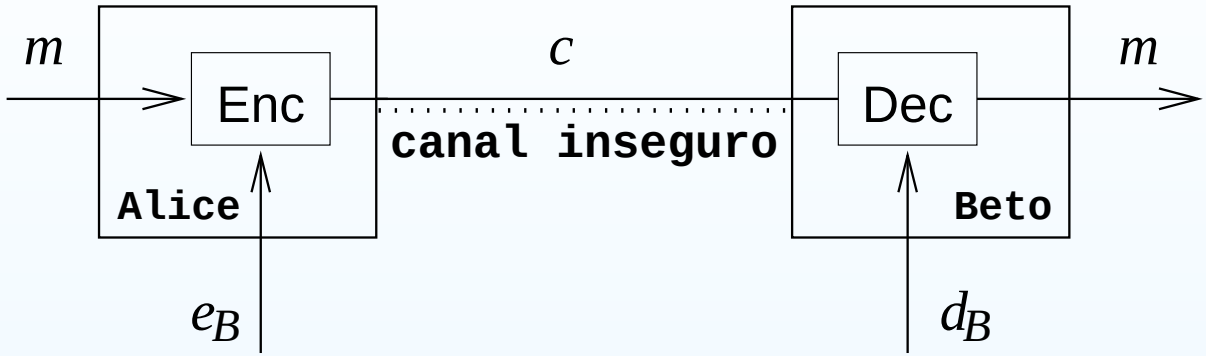


Figura 3.2 Modelo Assimétrico

Criptografia assimétrica

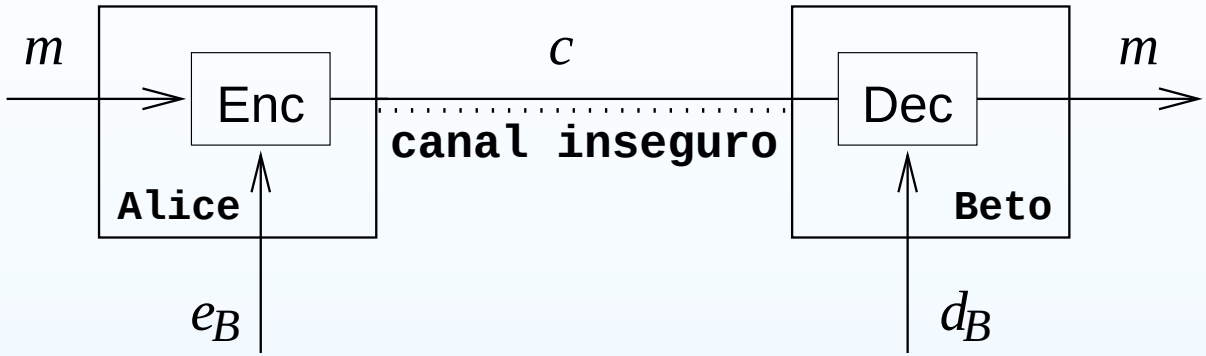


Figura 3.2 Modelo Assimétrico

- Neste modelo, Alice aplica uma função de encriptação $\text{ENC}_{e_B}(m)$, que transforma m numa mensagem encriptada c, sob a ação da chave e_B. Ao receber c, Beto aplica a função de decriptação $\text{DEC}_{d_B}(c)$, recuperando m. No caso de mensagens de Beto para Alice, as chaves usadas são: e_A para encriptação e d_A para decriptação.

Criptografia Assimétrica

As chaves e_B, d_B são ambas de Beto. A primeira, e_B, é a chave pública de Beto, distribuída e utilizada livremente. A segunda, d_B, é de conhecimento exclusivo de Beto, sua chave privada. A chave e_B é utilizada para encriptação de mensagens para Beto e d_B para decriptação dessas mensagens. O mesmo se aplica para Alice em relação a e_A e d_A.

Conseqüências do modelo assimétrico

Conseqüências do modelo assimétrico

- Não é mais necessário um acordo prévio de chaves, já que cada usuário deve necessariamente gerar o seu próprio par de chaves.

Conseqüências do modelo assimétrico

- Não é mais necessário um acordo prévio de chaves, já que cada usuário deve necessariamente gerar o seu próprio par de chaves.
- Além dessa melhora qualitativa, a redução do número de chaves também impressiona: $n \times (n-1)/2$ chaves no caso simétrico e $2n$ no caso assimétrico.

Conseqüências do modelo assimétrico

- Não é mais necessário um acordo prévio de chaves, já que cada usuário deve necessariamente gerar o seu próprio par de chaves.
- Além dessa melhora qualitativa, a redução do número de chaves também impressiona: $n \times (n-1)/2$ chaves no caso simétrico e $2n$ no caso assimétrico.
- Embora não tenha mais que se preocupar com o sigilo da chave de encriptação, Alice deve agora ter a confiança de que a chave de encriptação é, de fato, a chave e_B de Beto. Tais chaves em pouco diferem de longuíssimas seqüências aleatórias de bits e, portanto, sua identificação é difícil. Claro que o uso constante da mesma chave com Beto traz essa confiança a Alice; o problema é o primeiro uso.

Premissas do modelo assimétrico

Premissas do modelo assimétrico

- $\text{ENC}_{\cdot}(.)$ deve ser unidirecional para cada chave e_X, a menos que se conheça a chave d_X de decriptação; supomos, como antes, que $\text{ENC}_{\cdot}(.)$ e $\text{DEC}_{\cdot}(.)$ sejam públicas;

Premissas do modelo assimétrico

- $\text{ENC}_{\cdot}(.)$ deve ser unidirecional para cada chave e_X, a menos que se conheça a chave d_X de decriptação; supomos, como antes, que $\text{ENC}_{\cdot}(.)$ e $\text{DEC}_{\cdot}(.)$ sejam públicas;
- obviamente, e_B e d_B são relacionadas, mas não deve ser possível calcular d_B a partir do conhecimento de e_B, em tempo hábil. Uma condição necessária para isso é que o número de possibilidades para d_B seja muito alto. Em alguns sistemas atuais, d_B chega a ter milhares de bits!

Criptografia assimétrica - assimetria

Criptografia assimétrica - assimetria

- A assimetria deste modelo é evidente: o poder na transmissão de mensagens de Alice para Beto é de Beto, o destinatário. Tanto Alice como qualquer outra entidade podem encriptar mensagens para Beto usando e_B, mas só Beto consegue decriptá-las, usando sua chave privada d_B.

Criptografia assimétrica - assimetria

- A assimetria deste modelo é evidente: o poder na transmissão de mensagens de Alice para Beto é de Beto, o destinatário. Tanto Alice como qualquer outra entidade podem encriptar mensagens para Beto usando e_B, mas só Beto consegue decriptá-las, usando sua chave privada d_B.
- O outro lado dessa moeda é a possibilidade de que Beto possa *assinar* mensagens enviadas a Alice e outros: se existirem funções $\textsc{Sign}_{d_B}(.))$ e $\textsc{Ver}_{e_B}(\cdot)$, com a propriedade de que $\textsc{Ver}_{e_B}(m, s)$ retorna 1 quando $s = \textsc{Sign}_{d_B}(m)$, e 0 caso contrário, teremos em s o equivalente de uma assinatura digital de Beto em m. Assinaturas digitais são o que possibilitam a irretratabilidade das mensagens assinadas por Beto.

Criptografia assimétrica - assimetria

- A assimetria deste modelo é evidente: o poder na transmissão de mensagens de Alice para Beto é de Beto, o destinatário. Tanto Alice como qualquer outra entidade podem encriptar mensagens para Beto usando e_B, mas só Beto consegue decriptá-las, usando sua chave privada d_B.
- O outro lado dessa moeda é a possibilidade de que Beto possa *assinar* mensagens enviadas a Alice e outros: se existirem funções $\textsc{Sign}_{d_B}(.))$ e $\textsc{Ver}_{e_B}(\cdot)$, com a propriedade de que $\textsc{Ver}_{e_B}(m, s)$ retorna 1 quando $s = \textsc{Sign}_{d_B}(m)$, e 0 caso contrário, teremos em s o equivalente de uma assinatura digital de Beto em m. Assinaturas digitais são o que possibilitam a irretratabilidade das mensagens assinadas por Beto.
- Criptografia assimétrica = de chave pública.

Alguns sistemas de chave pública

Alguns sistemas de chave pública

- RSA, 1978.

Alguns sistemas de chave pública

- RSA, 1978.
- ElGamal, 1984-1985.

Alguns sistemas de chave pública

- RSA, 1978.
- ElGamal, 1984-1985.
- Rabin, 1979.

Alguns sistemas de chave pública

- RSA, 1978.
- ElGamal, 1984-1985.
- Rabin, 1979.
- Curvas Elípticas, Miller e Koblitz, 1985.

Alguns sistemas de chave pública

- RSA, 1978.
- ElGamal, 1984-1985.
- Rabin, 1979.
- Curvas Elípticas, Miller e Koblitz, 1985.
- DSA, 1991. Proposto por NIST.

Alguns sistemas de chave pública

- RSA, 1978.
- ElGamal, 1984-1985.
- Rabin, 1979.
- Curvas Elípticas, Miller e Koblitz, 1985.
- DSA, 1991. Proposto por NIST.
- ECDSA, NIST (1999).

Alguns sistemas de chave pública

- RSA, 1978.
- ElGamal, 1984-1985.
- Rabin, 1979.
- Curvas Elípticas, Miller e Koblitz, 1985.
- DSA, 1991. Proposto por NIST.
- ECDSA, NIST (1999).
- IBE, 2000 (Criptografia Baseada em Identidades).

Breve histórico

Breve histórico

- Conceito introduzido por Whitfield Diffie e Martin Hellman em 1976: Protocolo de acordo de chaves DH. (Logaritmo discreto módulo p).

Breve histórico

- Conceito introduzido por Whitfield Diffie e Martin Hellman em 1976: Protocolo de acordo de chaves DH. (Logaritmo discreto módulo p).
- RSA: Rivest, Shamir Adleman, 1978. (Fatoração).

Breve histórico

- Conceito introduzido por Whitfield Diffie e Martin Hellman em 1976: Protocolo de acordo de chaves DH. (Logaritmo discreto módulo p).
- RSA: Rivest, Shamir Adleman, 1978. (Fatoração).
- ElGamal: (Assinatura digital), 1984-1985.(Logaritmo discreto módulo p.)

Breve histórico

- Conceito introduzido por Whitfield Diffie e Martin Hellman em 1976: Protocolo de acordo de chaves DH. (Logaritmo discreto módulo p).
- RSA: Rivest, Shamir Adleman, 1978. (Fatoração).
- ElGamal: (Assinatura digital), 1984-1985.(Logaritmo discreto módulo p.)
- Goverment Communications Headquarters (UK). (James Ellis, Clifford Cocks, Malcolm Williamson - primeiros inventores do RSA e do Protocolo DH, 1969-1974).

Fundamentos matemáticos

Fundamentos matemáticos

- $\mathbb{Z}$ é o conjunto dos números inteiros.

Fundamentos matemáticos

- $\mathbb{Z}$ é o conjunto dos números inteiros.
- a é *divisível por* b se existe inteiro q tal que $a = qb$. Nesse caso, b é divisor de a

Fundamentos matemáticos

- $\mathbb{Z}$ é o conjunto dos números inteiros.
- a é *divisível por* b se existe inteiro q tal que $a = qb$.
  Nesse caso, b é divisor de a
  $3|21$, mas $4 \not| 15$.

Fundamentos matemáticos

- $\mathbb{Z}$ é o conjunto dos números inteiros.
- a é *divisível por* b se existe inteiro q tal que $a = qb$. Nesse caso, b é divisor de a

  $3|21$, mas $4 \not| 15$.
- Um número inteiro $p \geq 2$ é *primo* se é divisível somente por 1 e por ele mesmo.

Fundamentos matemáticos

- $\mathbb{Z}$ é o conjunto dos números inteiros.
- a é *divisível por* b se existe inteiro q tal que $a = qb$. Nesse caso, b é divisor de a

  $3|21$, mas $4 \not| 15$.
- Um número inteiro $p \geq 2$ é *primo* se é divisível somente por 1 e por ele mesmo.
- Todo inteiro $n \geq 2$ pode ser escrito como um produto de potências de números primos; a *fatoração* de n.

O máximo divisor comum (m.d.c)

O máximo divisor comum (m.d.c)

- Considere a, b dois números inteiros, não ambos nulos.

O máximo divisor comum (m.d.c)

- Considere a, b dois números inteiros, não ambos nulos.
- O *máximo divisor comum* de a e b, ($\mathrm{mdc}(a, b)$), é o maior inteiro d que divide ambos a e b.

O máximo divisor comum (m.d.c)

- Considere a, b dois números inteiros, não ambos nulos.
- O *máximo divisor comum* de a e b, ($\mathrm{mdc}(a, b)$), é o maior inteiro d que divide ambos a e b.

$$\mathrm{mdc}(20, 8) = 4$$

O máximo divisor comum (m.d.c)

- Considere a, b dois números inteiros, não ambos nulos.
- O *máximo divisor comum* de a e b, ($\mathrm{mdc}(a, b)$), é o maior inteiro d que divide ambos a e b.

$$\mathrm{mdc}(20, 8) = 4$$
$$\mathrm{mdc}(20, 0) = 20$$

O máximo divisor comum (m.d.c)

- Considere a, b dois números inteiros, não ambos nulos.
- O *máximo divisor comum* de a e b, ($\mathrm{mdc}(a, b)$), é o maior inteiro d que divide ambos a e b.

$$\mathrm{mdc}(20, 8) = 4$$
$$\mathrm{mdc}(20, 0) = 20$$
$$\mathrm{mdc}(20, 7) = 1$$

O máximo divisor comum (m.d.c)

- Considere a, b dois números inteiros, não ambos nulos.
- O *máximo divisor comum* de a e b, ($\mathrm{mdc}(a, b)$), é o maior inteiro d que divide ambos a e b.

  $$\mathrm{mdc}(20, 8) = 4$$
  $$\mathrm{mdc}(20, 0) = 20$$
  $$\mathrm{mdc}(20, 7) = 1$$
- Quando $\mathrm{mdc}(a, b) = 1$, dizemos que a e b são *primos entre si* ou *coprimos*.

O máximo divisor comum (m.d.c)

- Considere a, b dois números inteiros, não ambos nulos.
- O *máximo divisor comum* de a e b, ($\mathrm{mdc}(a, b)$), é o maior inteiro d que divide ambos a e b.

$$\mathrm{mdc}(20, 8) = 4$$
$$\mathrm{mdc}(20, 0) = 20$$
$$\mathrm{mdc}(20, 7) = 1$$

- Quando $\mathrm{mdc}(a, b) = 1$, dizemos que a e b são *primos entre si* ou *coprimos*.

O Algoritmo de Euclides, é o método mais popular para o cálculo do mdc.

O Algoritmo de Euclides

Entrada: inteiros a, b, com $a > 0,\ b \geq 0$.
Saída: inteiro d, onde $d = \mathrm{mdc}(a, b)$.
1. **se** $b = 0$ **então** retorne (a);
2. **enquanto** $b > 0$ **faça**
   2.1 $q \leftarrow a \text{ div } b;\ r \leftarrow a - q * b;$
   2.2 $a \leftarrow b;\ b \leftarrow r;$
3. $d \leftarrow a;$
4. retorne (d);

Aritmética modular

Aritmética modular

Definição 3 *Sejam a, n números inteiros com $n > 0$. O* resto *ou* resíduo *da divisão de a por n é o único inteiro r, com $0 \leq r \leq n - 1$, tal que $a = qn + r$ para algum inteiro q, o* quociente *da divisão.*

Aritmética modular

Definição 5 *Sejam a, n números inteiros com $n > 0$. O* resto *ou* resíduo *da divisão de a por n é o único inteiro r, com $0 \leq r \leq n - 1$, tal que $a = qn + r$ para algum inteiro q, o* quociente *da divisão.*

Por essa definição o resto da divisão de 7 por 3 é 1 (com quociente 2), e o resto da divisão de -7 por 3 é 2 (com quociente -3).

Aritmética modular

Definição 7 *Sejam a, n números inteiros com $n > 0$. O* resto *ou* resíduo *da divisão de a por n é o único inteiro r, com $0 \leq r \leq n - 1$, tal que $a = qn + r$ para algum inteiro q, o* quociente *da divisão.*

Por essa definição o resto da divisão de 7 por 3 é 1 (com quociente 2), e o resto da divisão de -7 por 3 é 2 (com quociente -3).

Definição 8 *Para a, n números inteiros com $n > 0$, a expressão $a \bmod n$ é a* redução de a módulo n, *definida como o resto da divisão de a por n.*

Aritmética modular

Definição 9 *Sejam a, n números inteiros com $n > 0$. O* resto *ou* resíduo *da divisão de a por n é o único inteiro r, com $0 \leq r \leq n - 1$, tal que $a = qn + r$ para algum inteiro q, o* quociente *da divisão.*

Por essa definição o resto da divisão de 7 por 3 é 1 (com quociente 2), e o resto da divisão de -7 por 3 é 2 (com quociente -3).

Definição 10 *Para a, n números inteiros com $n > 0$, a expressão $a \bmod n$ é a* redução de a módulo n, *definida como o resto da divisão de a por n.*

Portanto, $0 \bmod 5 = 0$ e $(3 - 8) \bmod 4 = -1 \bmod 4 = 3$.

Aritmética modular

Aritmética modular

Definição 13 *Dado um inteiro $n \geq 1$, denotamos por $\mathbb{Z}_n$ ao conjunto $\{0, 1, \ldots, n-1\}$ de resíduos módulo n, isto é dos restos possíveis de divisões de números inteiros por n.*

Aritmética modular

Definição 15 *Dado um inteiro $n \geq 1$, denotamos por $\mathbb{Z}_n$ ao conjunto $\{0, 1, \ldots, n-1\}$ de resíduos módulo n, isto é dos restos possíveis de divisões de números inteiros por n.*

Como todo número inteiro produz um resto ao ser dividido por n, $\mathbb{Z}_n$ tem em si um representante para cada número inteiro. A próxima definição captura essa idéia.

Aritmética modular

Definição 17 *Dado um inteiro $n \geq 1$, denotamos por $\mathbb{Z}_n$ ao conjunto $\{0, 1, \ldots, n-1\}$ de resíduos módulo n, isto é dos restos possíveis de divisões de números inteiros por n.*

Como todo número inteiro produz um resto ao ser dividido por n, $\mathbb{Z}_n$ tem em si um representante para cada número inteiro. A próxima definição captura essa idéia.

Definição 18 *Para a, b, n números inteiros com $n > 0$, escrevemos $a \equiv b \pmod{n}$, quando $a \bmod n = b \bmod n$. Dizemos que a e b são congruentes módulo n.*

Aritmética modular

Definição 19 *Dado um inteiro $n \geq 1$, denotamos por $\mathbb{Z}_n$ ao conjunto $\{0, 1, \ldots, n-1\}$ de resíduos módulo n, isto é dos restos possíveis de divisões de números inteiros por n.*

Como todo número inteiro produz um resto ao ser dividido por n, $\mathbb{Z}_n$ tem em si um representante para cada número inteiro. A próxima definição captura essa idéia.

Definição 20 *Para a, b, n números inteiros com $n > 0$, escrevemos $a \equiv b \pmod{n}$, quando $a \bmod n = b \bmod n$. Dizemos que a e b são congruentes módulo n.*

Assim, $0 \equiv 3 \pmod{3}$ e $43 \equiv 1 \pmod{2}$.

Inversos

Inversos

Definição 22 *Para a, n números inteiros com $n > 0$, o* inverso aditivo de a *módulo n é o inteiro $b = -a \bmod n$.; ou seja $a + b \equiv 0 \pmod{n}$.*
O inverso multiplicativo de a *módulo n, se existir, é o único inteiro b, $1 \leq b \leq n - 1$, tal que $ab \equiv 1 \pmod{n}$.*
Denotamos o inverso multiplicativo de a módulo n por $a^{-1} \bmod n$.

Inversos

Definição 23 *Para a, n números inteiros com $n > 0$, o* inverso aditivo de a *módulo n é o inteiro $b = -a \bmod n$.; ou seja $a + b \equiv 0 \pmod{n}$.*
O inverso multiplicativo de a *módulo n, se existir, é o único inteiro b, $1 \leq b \leq n - 1$, tal que $ab \equiv 1 \pmod{n}$.*
Denotamos o inverso multiplicativo de a módulo n por $a^{-1} \bmod n$.

- O inverso aditivo de 4 módulo 7 é 3.

Inversos

Definição 24 *Para a, n números inteiros com $n > 0$, o* inverso aditivo de a *módulo n é o inteiro $b = -a \bmod n$.; ou seja $a + b \equiv 0 \pmod{n}$.*
O inverso multiplicativo de a *módulo n, se existir, é o único inteiro b, $1 \leq b \leq n - 1$, tal que $ab \equiv 1 \pmod{n}$.*
Denotamos o inverso multiplicativo de a módulo n por $a^{-1} \bmod n$.

- O inverso aditivo de 4 módulo 7 é 3.
- O inverso multiplicativo de 2 módulo 5 é 3.

Inversos

Definição 25 *Para a, n números inteiros com $n > 0$, o* inverso aditivo de a *módulo n é o inteiro $b = -a \bmod n$.; ou seja $a + b \equiv 0 \pmod{n}$.*
O inverso multiplicativo de a *módulo n, se existir, é o único inteiro b, $1 \leq b \leq n - 1$, tal que $ab \equiv 1 \pmod{n}$.*
Denotamos o inverso multiplicativo de a módulo n por $a^{-1} \bmod n$.

- O inverso aditivo de 4 módulo 7 é 3.
- O inverso multiplicativo de 2 módulo 5 é 3.
- O inverso multiplicativo de 2 módulo 6 não existe.

Inversos

Definição 26 *Para a, n números inteiros com $n > 0$, o* inverso aditivo de a *módulo n é o inteiro $b = -a \bmod n$.; ou seja $a + b \equiv 0 \pmod{n}$.*
O inverso multiplicativo de a *módulo n, se existir, é o único inteiro b, $1 \leq b \leq n - 1$, tal que $ab \equiv 1 \pmod{n}$.*
Denotamos o inverso multiplicativo de a módulo n por $a^{-1} \bmod n$.

- O inverso aditivo de 4 módulo 7 é 3.
- O inverso multiplicativo de 2 módulo 5 é 3.
- O inverso multiplicativo de 2 módulo 6 não existe.

Teorema 6 *Para a, n números inteiros com $n > 0$, o inverso multiplicativo de a módulo n existe se e somente se $\mathrm{mdc}(a, n) = 1$.*

Extensão do Algoritmo de Euclides

Dados a, n, a extensão do Algoritmo de Euclides, retorna inteiros (d, s, t) onde $d = \mathrm{mdc}(a, n)$ e $d = sa + tn$. Isto é, $sa \equiv d \pmod{n}$. Assim, quando $\mathrm{mdc}(a, n) = 1$, o inteiro s é $a^{-1} \bmod n$.

Entrada: inteiros a, b.
Saída: inteiros d, s, t, onde $d = \mathrm{mdc}(a, b) = sa + tb$.

1. **se** $b = 0$ **então** retorne $(a, 1, 0)$;
2. $x_2 \leftarrow 1$; $x_1 \leftarrow 0$; $y_2 \leftarrow 0$; $y_1 \leftarrow 1$;
3. **enquanto** $b > 0$ **faça**
   - 3.1 $q \leftarrow a \operatorname{div} b$; $r \leftarrow a - q * b$;
   - 3.2 $s \leftarrow x_2 - q * x_1$; $t \leftarrow y_2 - q * y_1$;
   - 3.3 $a \leftarrow b$; $b \leftarrow r$; $x_2 \leftarrow x_1$; $x_1 \leftarrow s$;
   - 3.4 $y_2 \leftarrow y_1$; $y_1 \leftarrow t$;
4. $d \leftarrow a$; $s \leftarrow x_2$; $t \leftarrow y_2$;
5. retorne (d, s, t);

Aritmética modular

Definição 27 *Dado um inteiro $n \geq 2$, denotamos por $\mathbb{Z}_n^*$ ao conjunto $\{a | \mathrm{mdc}(a, n) = 1, 1 \leq a \leq n-1\}$. O tamanho de $\mathbb{Z}_n^*$ é representado por $\phi(n)$, a função de Euler.*

Aritmética modular

Definição 28 *Dado um inteiro $n \geq 2$, denotamos por $\mathbb{Z}_n^*$ ao conjunto $\{a | \mathrm{mdc}(a, n) = 1, 1 \leq a \leq n-1\}$. O tamanho de $\mathbb{Z}_n^*$ é representado por $\phi(n)$, a função de Euler.*

Assim, $\mathbb{Z}_{10}^* = \{1, 3, 7, 9\}$ e $\phi(10) = 4$. Também, $\phi(n) = n - 1$ sempre que n for primo.

Aritmética modular

Definição 29 *Dado um inteiro $n \geq 2$, denotamos por $\mathbb{Z}_n^*$ ao conjunto $\{a | \text{mdc}(a, n) = 1, 1 \leq a \leq n-1\}$. O tamanho de $\mathbb{Z}_n^*$ é representado por $\phi(n)$, a função de Euler.*

Assim, $\mathbb{Z}_{10}^* = \{1, 3, 7, 9\}$ e $\phi(10) = 4$. Também, $\phi(n) = n - 1$ sempre que n for primo.

- É fácil verificar que as operações de soma, subtração e multiplicação modular são as mesmas da aritmética usual mas com o resultado reduzido módulo n. A divisão é a única exceção: $(a/b) \bmod n$ é sempre escrita e intepretada como $ab^{-1} \bmod n$.

Grupos

Um *grupo* é formado por um conjunto $\mathbb{G}$ e uma operação $+$, com as seguintes propriedades, para todos $a, b, c \in \mathbb{G}$:

Grupos

Um *grupo* é formado por um conjunto $\mathbb{G}$ e uma operação $+$, com as seguintes propriedades, para todos $a, b, c \in \mathbb{G}$:

1. (fechamento) $a + b \in \mathbb{G}$;

Grupos

Um *grupo* é formado por um conjunto $\mathbb{G}$ e uma operação $+$, com as seguintes propriedades, para todos $a, b, c \in \mathbb{G}$:

1. (fechamento) $a + b \in \mathbb{G}$;
2. (associatividade) $(a + b) + c = a + (b + c)$;

Grupos

Um *grupo* é formado por um conjunto $\mathbb{G}$ e uma operação $+$, com as seguintes propriedades, para todos $a, b, c \in \mathbb{G}$:

1. (fechamento) $a + b \in \mathbb{G}$;
2. (associatividade) $(a + b) + c = a + (b + c)$;
3. (existência de elemento neutro ou *identidade*) existe um elemento em $\mathbb{G}$, denotado 0, tal que $a + 0 = a$;

Grupos

Um *grupo* é formado por um conjunto $\mathbb{G}$ e uma operação $+$, com as seguintes propriedades, para todos $a, b, c \in \mathbb{G}$:

1. (fechamento) $a + b \in \mathbb{G}$;
2. (associatividade) $(a + b) + c = a + (b + c)$;
3. (existência de elemento neutro ou *identidade*) existe um elemento em $\mathbb{G}$, denotado 0, tal que $a + 0 = a$;
4. (existência de inversos) para todo $a \in \mathbb{G}$, existe em $\mathbb{G}$ um elemento denotado $-a$, tal que $a + (-a) = 0$.

Um grupo é *abeliano* se $a + b = b + a$ para todos $a, b \in \mathbb{G}$.
Exemplos de grupos são:

Grupos

Um *grupo* é formado por um conjunto $\mathbb{G}$ e uma operação $+$, com as seguintes propriedades, para todos $a, b, c \in \mathbb{G}$:

1. (fechamento) $a + b \in \mathbb{G}$;
2. (associatividade) $(a + b) + c = a + (b + c)$;
3. (existência de elemento neutro ou *identidade*) existe um elemento em $\mathbb{G}$, denotado 0, tal que $a + 0 = a$;
4. (existência de inversos) para todo $a \in \mathbb{G}$, existe em $\mathbb{G}$ um elemento denotado $-a$, tal que $a + (-a) = 0$.

Um grupo é *abeliano* se $a + b = b + a$ para todos $a, b \in \mathbb{G}$.
Exemplos de grupos são:

- números inteiros, racionais e reais com a soma usual;
- os elementos de $\mathbb{Z}_n = \{0, 1, 2, \ldots, p - 1\}$, $n > 0$, com a operação de soma módulo n;
- os elementos de $\mathbb{Z}_p^* = \{1, 2, \ldots, n - 1\}$, $p > 1$, primo, com a operação de multiplicação módulo p.

Grupos aditivos e multiplicativos

Grupos aditivos e multiplicativos

- Denotaremos o grupo definido acima por $(\mathbb{G}, +)$, ou simplesmente $\mathbb{G}$, quando a operação $+$ estiver subentendida no texto.

Grupos aditivos e multiplicativos

- Denotaremos o grupo definido acima por $(\mathbb{G}, +)$, ou simplesmente $\mathbb{G}$, quando a operação $+$ estiver subentendida no texto.
- Essa definição usa a notação aditiva, isto é, $a + a + a + a$ é denotado por $4a$, 0 é a identidade, e $0.a = 0$.

Grupos aditivos e multiplicativos

- Denotaremos o grupo definido acima por $(\mathbb{G}, +)$, ou simplesmente $\mathbb{G}$, quando a operação $+$ estiver subentendida no texto.
- Essa definição usa a notação aditiva, isto é, $a + a + a + a$ é denotado por $4a$, 0 é a identidade, e $0.a = 0$.
- Poderíamos ter usado uma notação multiplicativa, onde a operação do grupo seria denotada '.'. Assim, $a.a.a$ (ou aaa) seria denotado por a^3, o elemento identidade seria 1, e $a^0 = 1$.

Como já ficou claro, essas não são necessariamente as operações usuais de soma e multiplicação.

Problema do logaritmo discreto

Problema do logaritmo discreto

- O número de elementos de $\mathbb{G}$ é a sua *ordem*. Se a ordem é finita, então $\mathbb{G}$ é um *grupo finito*.
- A *ordem de um elemento* $a \in \mathbb{G}$ é o menor inteiro positivo t tal que $ta = 0$. É um fato bem conhecido que a ordem de um elemento divide a ordem do grupo.

Problema do logaritmo discreto

- O número de elementos de $\mathbb{G}$ é a sua *ordem*. Se a ordem é finita, então $\mathbb{G}$ é um *grupo finito*.
- A *ordem de um elemento* $a \in \mathbb{G}$ é o menor inteiro positivo t tal que $ta = 0$. É um fato bem conhecido que a ordem de um elemento divide a ordem do grupo.
- Quando, para um grupo finito $\mathbb{G}$ de ordem n, existe um elemento α de ordem n, dizemos que $\mathbb{G}$ é *cíclico* e que α é um *gerador* de $\mathbb{G}$.

Problema do logaritmo discreto

- O número de elementos de $\mathbb{G}$ é a sua *ordem*. Se a ordem é finita, então $\mathbb{G}$ é um *grupo finito*.
- A *ordem de um elemento* $a \in \mathbb{G}$ é o menor inteiro positivo t tal que $ta = 0$. É um fato bem conhecido que a ordem de um elemento divide a ordem do grupo.
- Quando, para um grupo finito $\mathbb{G}$ de ordem n, existe um elemento α de ordem n, dizemos que $\mathbb{G}$ é *cíclico* e que α é um *gerador* de $\mathbb{G}$.

Definição 33 *(Problema do logaritmo discreto) Dados elementos a, b de um grupo $(G, .)$, tais que $b = a^l$, o* problema do logaritmo discreto *é o de encontrar l conhecendo a e b apenas.*

Corpos finitos

Definição 34 *Um* corpo *é formado por um conjunto* $\mathbb{F}$ *e duas operações, '*$+$*' e '*$.$*', satisfazendo as seguintes propriedades:*

- $(\mathbb{F}, +)$ *é um grupo abeliano com identidade 0;*

Corpos finitos

Definição 35 *Um* corpo *é formado por um conjunto* $\mathbb{F}$ *e duas operações, '*$+$*' e '*$.$*', satisfazendo as seguintes propriedades:*

- $(\mathbb{F}, +)$ *é um grupo abeliano com identidade 0;*
- $(\mathbb{F} \setminus \{0\}, .)$ *é um grupo abeliano com identidade 1; e*

Corpos finitos

Definição 36 *Um* corpo *é formado por um conjunto* $\mathbb{F}$ *e duas operações, '*$+$*' e '*$.$*', satisfazendo as seguintes propriedades:*

- $(\mathbb{F}, +)$ *é um grupo abeliano com identidade 0;*
- $(\mathbb{F} \setminus \{0\}, .)$ *é um grupo abeliano com identidade 1; e*
- *a operação* $.$ *é distributiva sobre a operação* $+$*, isto é,* $a.(b + c) = a.b + a.c$*, para todos* $a, b, c \in \mathbb{F}$*.*

Números racionais, reais e complexos são exemplos de corpos infinitos.
A *ordem* de um corpo finito é o número de elementos em $\mathbb{F}$. Quando a ordem é finita dizemos que o corpo é *finito*.

Corpos finitos - existência

Sejam a, b dois elementos de um corpo, finito ou não. Então

- $a - b$ é equivalente a $a + (-b)$, onde $-b$ é o (único) inverso aditivo de b;

Corpos finitos - existência

Sejam a, b dois elementos de um corpo, finito ou não. Então

- $a - b$ é equivalente a $a + (-b)$, onde $-b$ é o (único) inverso aditivo de b;
- a/b é equivalente a $a.(b^{-1})$, onde b^{-1} é o (único) inverso multiplicativo de b;

Corpos finitos - existência

Sejam a, b dois elementos de um corpo, finito ou não. Então

- $a - b$ é equivalente a $a + (-b)$, onde $-b$ é o (único) inverso aditivo de b;
- a/b é equivalente a $a.(b^{-1})$, onde b^{-1} é o (único) inverso multiplicativo de b;
- ka denota a adição de k parcelas iguais a a;

Corpos finitos - existência

Sejam a, b dois elementos de um corpo, finito ou não. Então

- $a - b$ é equivalente a $a + (-b)$, onde $-b$ é o (único) inverso aditivo de b;
- a/b é equivalente a $a.(b^{-1})$, onde b^{-1} é o (único) inverso multiplicativo de b;
- ka denota a adição de k parcelas iguais a a;
- a^k denota a multiplicação de k parcelas iguais a a,

Corpos finitos - existência

Sejam a, b dois elementos de um corpo, finito ou não. Então

- $a - b$ é equivalente a $a + (-b)$, onde $-b$ é o (único) inverso aditivo de b;
- a/b é equivalente a $a.(b^{-1})$, onde b^{-1} é o (único) inverso multiplicativo de b;
- ka denota a adição de k parcelas iguais a a;
- a^k denota a multiplicação de k parcelas iguais a a, onde $a^0 = 1$.

- Existe um corpo finito de ordem q, se e somente se $q = p^m$ para algum primo p e inteiro $m > 0$.

Corpos finitos - existência

Sejam a, b dois elementos de um corpo, finito ou não. Então

- $a - b$ é equivalente a $a + (-b)$, onde $-b$ é o (único) inverso aditivo de b;
- a/b é equivalente a $a.(b^{-1})$, onde b^{-1} é o (único) inverso multiplicativo de b;
- ka denota a adição de k parcelas iguais a a;
- a^k denota a multiplicação de k parcelas iguais a a, onde $a^0 = 1$.

- Existe um corpo finito de ordem q, se e somente se $q = p^m$ para algum primo p e inteiro $m > 0$.
- O primo p é a *característica* de $\mathbb{F}$.

Corpos finitos - existência

Sejam a, b dois elementos de um corpo, finito ou não. Então

- $a - b$ é equivalente a $a + (-b)$, onde $-b$ é o (único) inverso aditivo de b;
- a/b é equivalente a $a.(b^{-1})$, onde b^{-1} é o (único) inverso multiplicativo de b;
- ka denota a adição de k parcelas iguais a a;
- a^k denota a multiplicação de k parcelas iguais a a, onde $a^0 = 1$.

- Existe um corpo finito de ordem q, se e somente se $q = p^m$ para algum primo p e inteiro $m > 0$.
- O primo p é a *característica* de $\mathbb{F}$.
- Quando q é primo, i.e. $m = 1$, dizemos que o corpo é *primo*. Quando $m > 1$, o corpo é *de extensão*.

Corpos finitos - existência

Sejam a, b dois elementos de um corpo, finito ou não. Então

- $a - b$ é equivalente a $a + (-b)$, onde $-b$ é o (único) inverso aditivo de b;
- a/b é equivalente a $a.(b^{-1})$, onde b^{-1} é o (único) inverso multiplicativo de b;
- ka denota a adição de k parcelas iguais a a;
- a^k denota a multiplicação de k parcelas iguais a a, onde $a^0 = 1$.

- Existe um corpo finito de ordem q, se e somente se $q = p^m$ para algum primo p e inteiro $m > 0$.
- O primo p é a *característica* de $\mathbb{F}$.
- Quando q é primo, i.e. $m = 1$, dizemos que o corpo é *primo*. Quando $m > 1$, o corpo é *de extensão*.
- Denotamos o corpo finito de ordem q por $\mathbb{F}_q$.

Corpos finitos - exemplos

- O conjunto $\mathbb{Z}_p$, p primo, com as operações de soma e multiplicação *módulo* p formam o corpo primo $\mathbb{F}_p$ de ordem p.

Corpos finitos - exemplos

- O conjunto $\mathbb{Z}_p$, p primo, com as operações de soma e multiplicação *módulo* p formam o corpo primo $\mathbb{F}_p$ de ordem p.
- O corpo *binário* $\mathbb{F}_{2^m}$ é formado pelos polinômios em uma variável z de grau máximo $m-1$, cujos coeficientes são 0 ou 1. As duas operações associadas são as de soma e multiplicação de polinômios, com as seguintes restrições:
  - os coeficientes do polinômio resultante são reduzidos módulo 2;

Corpos finitos - exemplos

- O conjunto $\mathbb{Z}_p$, p primo, com as operações de soma e multiplicação *módulo* p formam o corpo primo $\mathbb{F}_p$ de ordem p.
- O corpo *binário* $\mathbb{F}_{2^m}$ é formado pelos polinômios em uma variável z de grau máximo $m-1$, cujos coeficientes são 0 ou 1. As duas operações associadas são as de soma e multiplicação de polinômios, com as seguintes restrições:
  - os coeficientes do polinômio resultante são reduzidos módulo 2;
  - o resultado da multiplicação de dois polinômios deve ser tomado módulo um polinômio *irredutível* $f(z)$ de grau m. Isto é, $f(z)$ não é o produto de dois polinômios binários de grau menor que m.

Corpos finitos - exemplos

Corpos finitos - exemplos

- Corpo Finito:

$$\mathbb{F}_{2^m} = \{ \sum_{i=0}^{m-1} a_i x^i \mid a_i \in \mathbb{Z}_2 \},$$

Corpos finitos - exemplos

- Corpo Finito:

$$\mathbb{F}_{2^m} = \{ \sum_{i=0}^{m-1} a_i x^i \mid a_i \in \mathbb{Z}_2 \},$$

$f(x) = x^m + r(x)$ polinômio irredutível

Corpos finitos - exemplos

- Corpo Finito:

$$\mathbb{F}_{2^m} = \{ \sum_{i=0}^{m-1} a_i x^i \mid a_i \in \mathbb{Z}_2 \},$$

$$f(x) = x^m + r(x) \text{ polinômio irredutível}$$

- Exemplo: $m = 3$, $\mathbb{F}_{2^3}$, $f(x) = x^3 + x + 1$

$$\begin{aligned} \mathbb{F}_{2^3} &= \{a_2x^2 + a_1x + a_0 \mid a_i \in \{0,1\}\} \\ &= \{0, 1, x, x+1, x^2, x^2+1, x^2+x, x^2+x+1\} \\ &= \{000, 001, 010, 011, 100, 101, 110, 111\} \end{aligned}$$

Corpos finitos - exemplos

Corpos finitos - exemplos

- Os elementos não nulos de um corpo finito $\mathbb{F}_q$, juntamente com a multiplicação do corpo, formam um grupo cíclico, denotado por $\mathbb{F}_q{}^*$.

Corpos finitos - exemplos

- Os elementos não nulos de um corpo finito $\mathbb{F}_q$, juntamente com a multiplicação do corpo, formam um grupo cíclico, denotado por $\mathbb{F}_q{}^*$.
- Portanto, existe para esse grupo pelo menos um gerador α, isto é,

$$\mathbb{F}_{q^*} = \{\alpha^i : 0 \leq i \leq q-2\}.$$

Corpos finitos - exemplos

- Os elementos não nulos de um corpo finito $\mathbb{F}_q$, juntamente com a multiplicação do corpo, formam um grupo cíclico, denotado por $\mathbb{F}_q{}^*$.
- Portanto, existe para esse grupo pelo menos um gerador α, isto é,

$$\mathbb{F}_{q^*} = \{\alpha^i : 0 \leq i \leq q-2\}.$$

- Os métodos conhecidos para o cálculo do logaritmo discreto em $\mathbb{F}_{q^*}$ são todos muito ineficientes quando q é muito grande, da ordem de centenas de dígitos. Para outros grupos o cálculo é muito fácil, por exemplo, em $(Z_n, +)$.

Técnicas criptográficas

Técnicas criptográficas

Algoritmos simétricos

Resumos criptográficos

Algoritmos assimétricos

Algoritmos simétricos

- DES (Data Encryption Standard), 1977
- AES (Advanced Encryption Standard), 2000

O DES

O DES

- Publicado em 1977: (The United States's National Bureau of Standards)

O DES

- Publicado em 1977: (The United States's National Bureau of Standards)
- FIPS PUB 46-3 (Federal Information Processing Standard Publication), 1999

O DES

- Publicado em 1977: (The United States's National Bureau of Standards)
- FIPS PUB 46-3 (Federal Information Processing Standard Publication), 1999
- Cifrador de blocos

O DES

- Publicado em 1977: (The United States's National Bureau of Standards)
- FIPS PUB 46-3 (Federal Information Processing Standard Publication), 1999
- Cifrador de blocos
- Tamanho dos blocos: 64 bits

O DES

- Publicado em 1977: (The United States's National Bureau of Standards)
- FIPS PUB 46-3 (Federal Information Processing Standard Publication), 1999
- Cifrador de blocos
- Tamanho dos blocos: 64 bits
- Tamanho das chaves: 64 bits (utiliza 56 bits)

O DES

- Publicado em 1977: (The United States's National Bureau of Standards)
- FIPS PUB 46-3 (Federal Information Processing Standard Publication), 1999
- Cifrador de blocos
- Tamanho dos blocos: 64 bits
- Tamanho das chaves: 64 bits (utiliza 56 bits)
- TDES (Triple DES): chaves de 112 bits

Descrição do DES

Entrada: texto claro: $M = m_1 ... m_{64}$, chave: $k = k_1 ... k_{56}$
Saída: texto cifrado $C = c_1 ... c_{64}$
1. Calcule 16 subchaves K_i de 48 bits da chave k
2. $(L_0, R_0) \leftarrow IP(M)$

Descrição do DES

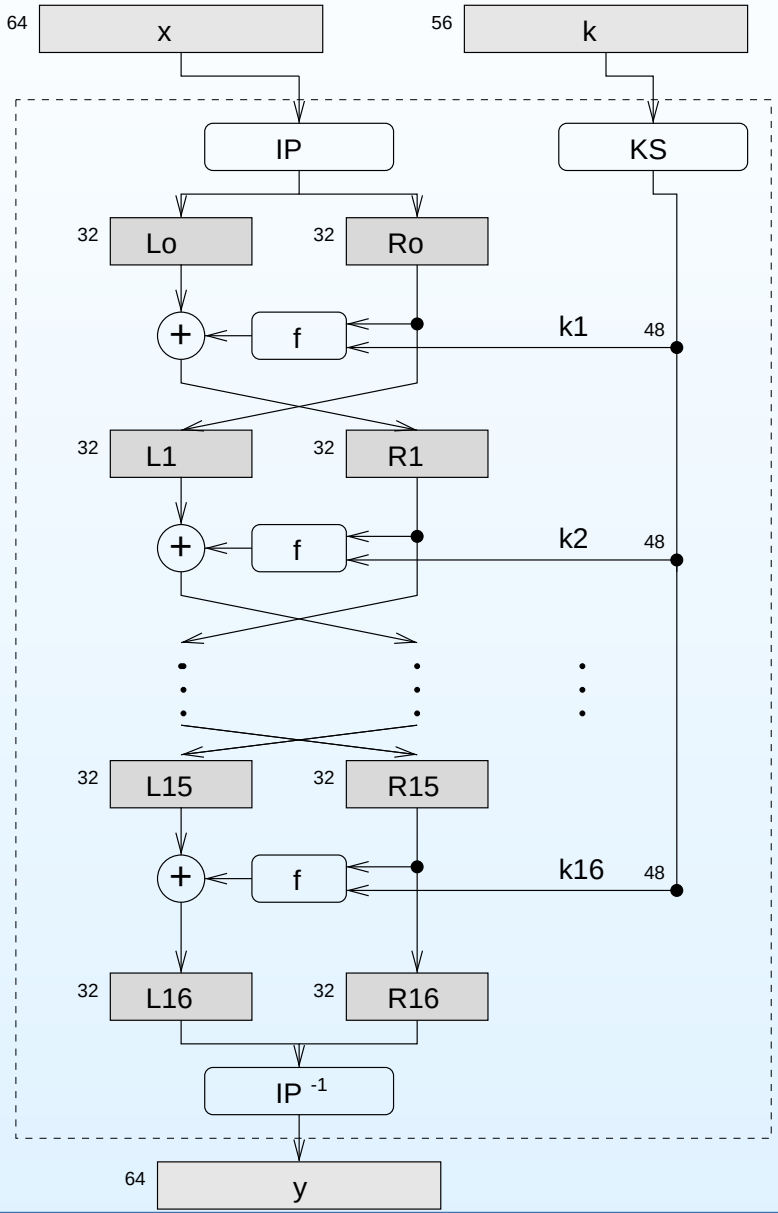
Entrada: texto claro: $M = m_1 ... m_{64}$, chave: $k = k_1 ... k_{56}$
Saída: texto cifrado $C = c_1 ... c_{64}$
1. Calcule 16 subchaves K_i de 48 bits da chave k
2. $(L_0, R_0) \leftarrow IP(M)$
3. **for** $i = 1$ **to** 16 **do**
   2.1 $L_i \leftarrow R_{i-1}$
   2.2 $R_i \leftarrow L_{i-1} \oplus f(R_{i-1}, K_i)$

Descrição do DES

Entrada: texto claro: $M = m_1...m_{64}$, chave: $k = k_1...k_{56}$
Saída: texto cifrado $C = c_1...c_{64}$
1. Calcule 16 subchaves K_i de 48 bits da chave k
2. $(L_0, R_0) \leftarrow IP(M)$
3. **for** $i = 1$ **to** 16 **do**
   2.1 $L_i \leftarrow R_{i-1}$
   2.2 $R_i \leftarrow L_{i-1} \oplus f(R_{i-1}, K_i)$
4. $b_1 b_2 ... b_{64} \leftarrow (R_{16}, L_{16})$
5. $C \leftarrow IP^{-1}(b_1 b_2 ... b_{64})$

$$f(R_{i-1}, K_i) = P(S(E(R_{i-1}) \oplus K_i))$$

Descrição do DES



O DES

- Nunca foi “quebrado”.

O DES

- Nunca foi “quebrado”.
- Busca exaustiva da chave tornou-se possível no final dos anos 90.

O DES

- Nunca foi “quebrado”.
- Busca exaustiva da chave tornou-se possível no final dos anos 90.
- Ainda muito usado como TripleDES.

O DES

- Nunca foi “quebrado”.
- Busca exaustiva da chave tornou-se possível no final dos anos 90.
- Ainda muito usado como TripleDES.

O AES

O AES

- Advanced Encryption Standard (FIPS PUB 197)

O AES

- Advanced Encryption Standard (FIPS PUB 197)
- Cifrador de blocos

O AES

- Advanced Encryption Standard (FIPS PUB 197)
- Cifrador de blocos
- Tamanho em bits dos blocos : 128

O AES

- Advanced Encryption Standard (FIPS PUB 197)
- Cifrador de blocos
- Tamanho em bits dos blocos : 128
- Tamanho em bits das chaves criptográficas: 128, 192, 256

O AES

- Advanced Encryption Standard (FIPS PUB 197)
- Cifrador de blocos
- Tamanho em bits dos blocos : 128
- Tamanho em bits das chaves criptográficas: 128, 192, 256
- Rijndael (Rijmen e Daemen), 1998.

Aritmética de Bytes

- Um byte: $(b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0),\ b_i \in \{0, 1\}$
  (00001111), (1010 0011), (1100 0001), etc.

Aritmética de Bytes

- Um byte: $(b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0),\ b_i \in \{0, 1\}$ (00001111), (1010 0011), (1100 0001), etc.
- Em base hexadecimal (16), um byte é representado por dois dígitos: 1000 0011 = `0x83`, 0111 1100 = `0x7c`, etc.

Aritmética de Bytes

- Um byte: $(b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0),\ b_i \in \{0, 1\}$ (00001111), (1010 0011), (1100 0001), etc.
- Em base hexadecimal (16), um byte é representado por dois dígitos: 1000 0011 = `0x83`, 0111 1100 = `0x7c`, etc.
- Um byte pode ser interpretado como um polinômio binário de grau máximo 7:
  $(10001111) = x^7 + x^3 + x^2 + x + 1$,
  $(00111100) = x^5 + x^4 + x^3 + x^2$.

Aritmética de Bytes

- Um byte: $(b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0),\ b_i \in \{0, 1\}$
  (00001111), (1010 0011), (1100 0001), etc.
- Em base hexadecimal (16), um byte é representado por dois dígitos: 1000 0011 = `0x83`, 0111 1100 = `0x7c`, etc.
- Um byte pode ser interpretado como um polinômio binário de grau máximo 7:
  $(10001111) = x^7 + x^3 + x^2 + x + 1$,
  $(00111100) = x^5 + x^4 + x^3 + x^2$.
- Soma e multiplicação de bytes?
  soma e multiplicação de polinômios binários (com coeficientes 0 e 1).

Aritmética de Bytes

$$GF(2^8) = \{\sum_{i=0}^{7} a_i x^i,\ a_i \in \{0,1\}\}$$

Aritmética de Bytes

$$GF(2^8) = \{\sum_{i=0}^{7} a_i x^i,\ a_i \in \{0,1\}\}$$

- Soma: $A + B = \sum_{i=0}^{7} a_i x^i + \sum_{i=0}^{7} b_i x^i = \sum_{i=0}^{7} (a_i + b_i \mod 2) = A \oplus B$ (xor).

Aritmética de Bytes

$$GF(2^8) = \{\sum_{i=0}^{7} a_i x^i,\ a_i \in \{0,1\}\}$$

- Soma: $A + B = \sum_{i=0}^{7} a_i x^i + \sum_{i=0}^{7} b_i x^i = \sum_{i=0}^{7} (a_i + b_i \mod 2) = A \oplus B$ (xor).
- Multiplicação: $A \times B = \sum_{i=0}^{14} c_i x^i \bmod \mathrm{x}^8 + \mathrm{x}^4 + \mathrm{x}^3 + \mathrm{x} + 1.$

Aritmética de Bytes

$$GF(2^8) = \{\sum_{i=0}^{7} a_i x^i,\ a_i \in \{0,1\}\}$$

- Soma: $A + B = \sum_{i=0}^{7} a_i x^i + \sum_{i=0}^{7} b_i x^i = \sum_{i=0}^{7} (a_i + b_i \mod 2) = A \oplus B$ (xor).
- Multiplicação: $A \times B = \sum_{i=0}^{14} c_i x^i \bmod \mathrm{x}^8 + \mathrm{x}^4 + \mathrm{x}^3 + \mathrm{x} + 1$.
- $(GF(2^8), +, \times)$: estrutura matemática chamada *corpo binário*.

Aritmética de Bytes

$$GF(2^8) = \{\sum_{i=0}^{7} a_i x^i,\ a_i \in \{0,1\}\}$$

- Soma: $A + B = \sum_{i=0}^{7} a_i x^i + \sum_{i=0}^{7} b_i x^i = \sum_{i=0}^{7} (a_i + b_i \mod 2) = A \oplus B$ (xor).
- Multiplicação: $A \times B = \sum_{i=0}^{14} c_i x^i \bmod \mathrm{x}^8 + \mathrm{x}^4 + \mathrm{x}^3 + \mathrm{x} + 1$.
- $(GF(2^8), +, \times)$: estrutura matemática chamada *corpo binário*.

Polinômios com Coeficientes em $GF(2^8)$

$$A = a_3x^3 + a_2x^2 + a_1x + a_0, \quad B = b_3x^3 + b_2x^2b_1x + b_0$$

Polinômios com Coeficientes em $GF(2^8)$

$$A = a_3x^3 + a_2x^2 + a_1x + a_0, \quad B = b_3x^3 + b_2x^2b_1x + b_0$$

Soma:

$$A + B = (a_3 + b_3)x^3 + (a_2 + b_2)x^2 + (a_1 + b_1)x^1 + (a_0 + b_0)$$

Multiplicação: $A \times B = ?$

$$C = A \times B = c_6x^6 + c_5x^5 + c_4x^4 + c_3x^3 + c_2x^2 + c_1x + c_0$$

Polinômios com Coeficientes em $GF(2^8)$

$$A = a_3x^3 + a_2x^2 + a_1x + a_0, \quad B = b_3x^3 + b_2x^2b_1x + b_0$$

Soma:

$$A + B = (a_3 + b_3)x^3 + (a_2 + b_2)x^2 + (a_1 + b_1)x^1 + (a_0 + b_0)$$

Multiplicação: $A \times B = ?$

$$C = A \times B = c_6x^6 + c_5x^5 + c_4x^4 + c_3x^3 + c_2x^2 + c_1x + c_0$$

$$c_0 = a_0 \bullet b_0$$
$$c_1 = a_1 \bullet b_0 \oplus a_0 \bullet b_1$$
$$c_2 = a_2 \bullet b_0 \oplus a_1 \bullet b_1 \oplus a_0 \bullet b_2$$
$$c_4 = a_3 \bullet b_1 \oplus a_2 \bullet b_2 \oplus a_1 \bullet$$
$$c_5 = a_3 \bullet b_2 \oplus a_2 \bullet b_3$$
$$c_6 = a_3 \bullet b_3$$

$$c_3 = a_3 \bullet b_0 \oplus a_2 \bullet b_1 \oplus a_1 \bullet b_2 \oplus a_0 \bullet b_3$$

Polinômios com Coeficientes em $GF(2^8)$

$$A \otimes B := C \bmod x^4 + 1 = d_3x^3 + d_2x^2 + d_1x + d_0x$$

$$x^i \bmod x^4 + 1 = x^i \mod 4$$

Polinômios com Coeficientes em $GF(2^8)$

$$A \otimes B := C \bmod x^4 + 1 = d_3x^3 + d_2x^2 + d_1x + d_0x$$

$$x^i \bmod x^4 + 1 = x^i \pmod 4$$

$$\begin{aligned}
d_0 &= (a_0 \bullet b_0) \oplus (a_3 \bullet b_1) \oplus (a_2 \bullet b_2) \oplus (a_1 \bullet b_3) \\
d_1 &= (a_1 \bullet b_0) \oplus (a_0 \bullet b_1) \oplus (a_3 \bullet b_2) \oplus (a_2 \bullet b_3) \\
d_2 &= (a_2 \bullet b_0) \oplus (a_1 \bullet b_1) \oplus (a_0 \bullet b_2) \oplus (a_3 \bullet b_3) \\
d_3 &= (a_3 \bullet b_0) \oplus (a_2 \bullet b_1) \oplus (a_1 \bullet b_2) \oplus (a_0 \bullet b_3)
\end{aligned}$$

Polinômios com Coeficientes em $GF(2^8)$

$$A \otimes B := C \bmod x^4 + 1 = d_3x^3 + d_2x^2 + d_1x + d_0x$$

$$\begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} a_0 & a_3 & a_2 & a_1 \\ a_1 & a_0 & a_3 & a_2 \\ a_2 & a_1 & a_0 & a_3 \\ a_3 & a_2 & a_1 & a_0 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Polinômios com Coeficientes em $GF(2^8)$ - Encriptação

$$D = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\} \otimes B$$

$$\begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Polinômios com Coeficientes em $GF(2^8)$ - Decriptação

$$D = \{0b\}x^3 + \{0d\}x^2 + \{09\}x + \{0e\} \otimes B$$

$$\begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 02 & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

O AES - o algoritmo

chave $K_{128} = (k_0, k_1, k_2, k_3, ..., k_{12}, k_{13}, k_{14}, k_{15})$

$$\begin{bmatrix} k_0 & k_4 & k_8 & k_{12} \\ k_1 & k_5 & k_9 & k_{13} \\ k_2 & k_6 & k_{10} & k_{14} \\ k_3 & k_7 & k_{11} & k_{15} \end{bmatrix}$$

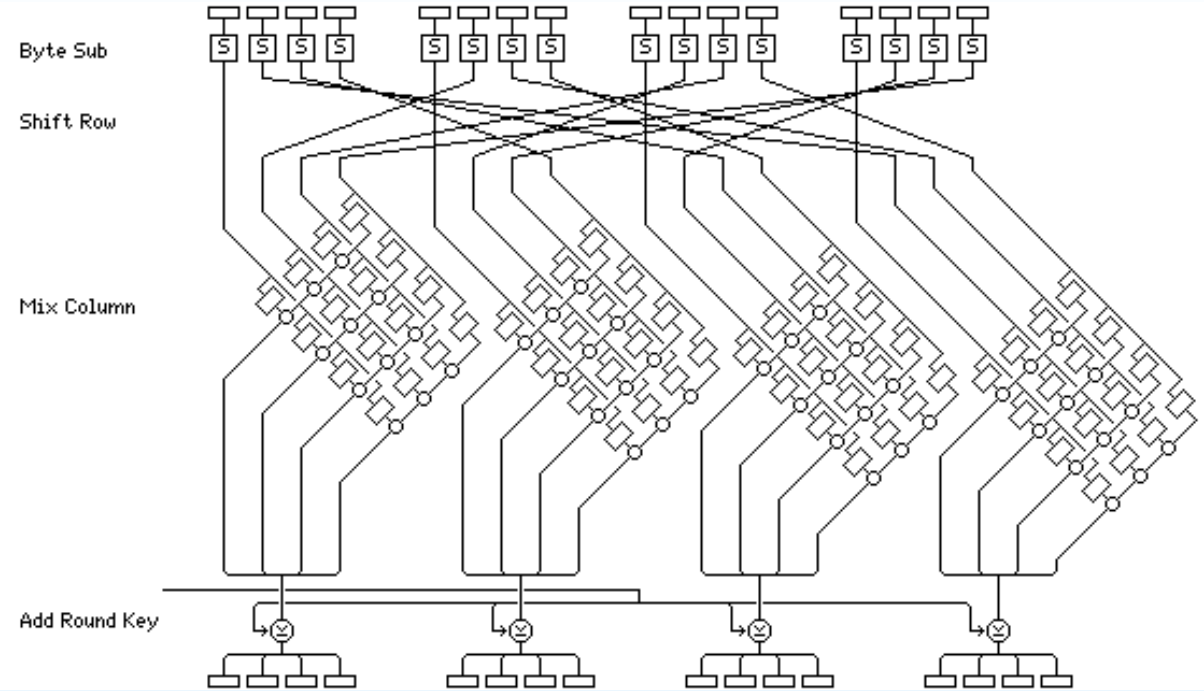
bloco $B_{128} = (b_0, b_1, b_2, b_3, ..., b_{12}, b_{13}, b_{14}, b_{15})$

$$\begin{bmatrix} b_0 & b_4 & b_8 & b_{12} \\ b_1 & b_5 & b_9 & b_{13} \\ b_2 & b_6 & b_{10} & b_{14} \\ b_3 & b_7 & b_{11} & b_{15} \end{bmatrix}$$

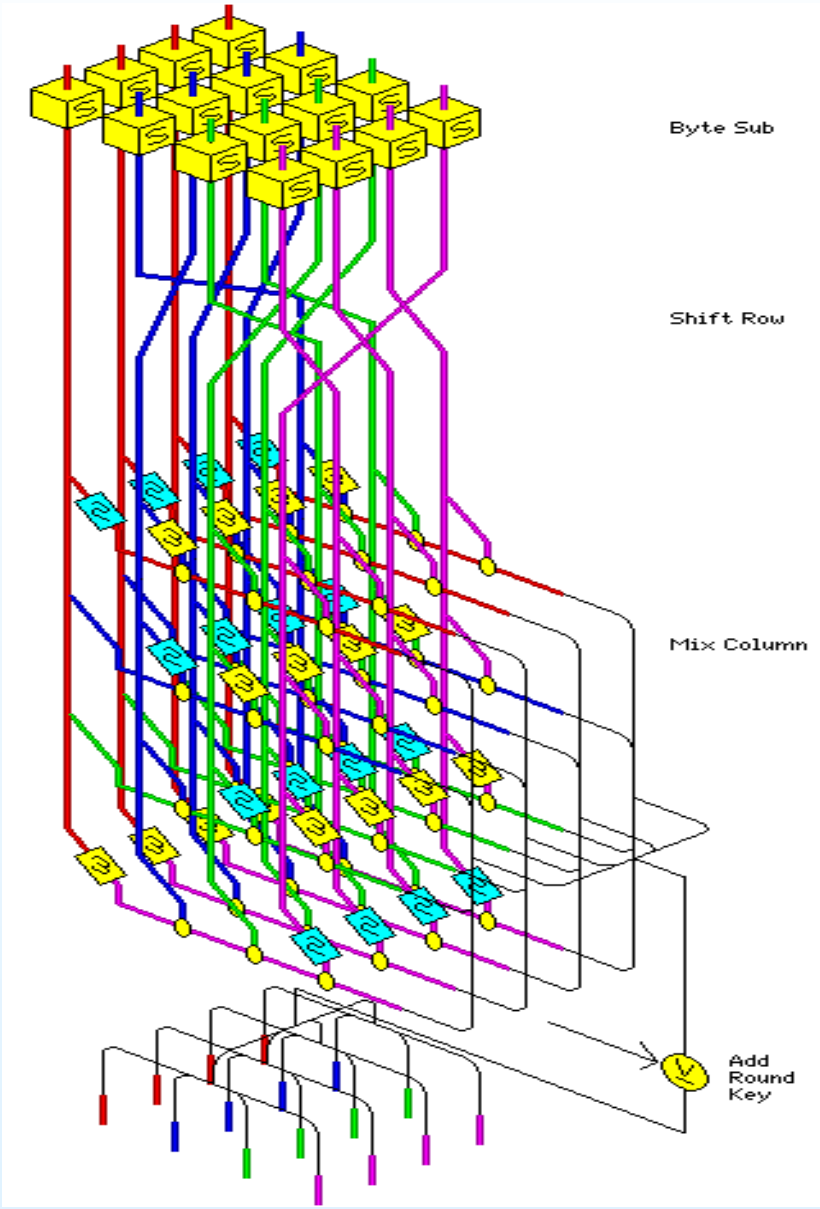
O AES - algoritmo

$State = [b_0 ... b_{15}]_{4\times4}$
$Key = [k_0 ... k_{15}]_{4\times4}$
AddRoundKey$(State, RoundKey)$;
for $i = 1$ **to** $N_r - 1$
    SubBytes$(State)$;
    ShiftRows$(State)$;
    Mixcolums$(State)$;
    AddRoundKey$(State, RoundKey)$;
SubBytes$(State)$;
ShiftRows$(State)$;
AddRoundKey$(State, RoundKey)$;

O AES



O AES em cores



SubBytes($State$)

$$S : GF(2^8) \longrightarrow GF(2^8)$$

$$S(x) := \begin{cases} Ax^{-1} + b & \text{se } x \neq 0 \\ b & \text{se } x = 0 \end{cases}$$

Modos de operação para encriptação em blocos

- Modos básicos: ECB, CBC, CFB, OFB, CTR
- Propostas para NIST: CCM (CBC-MAC), CS, CWC, GCM (2005), OMAC, ABC,...

Electronic CodeBook (ECB)

Entrada: $P_1, P_2, ..., P_m$, $|P_i| = n$ bits.
Saída: $C_1, C_2, ..., C_m$, $|C_i| = n$ bits.

for $i = 1$ **to** m **do**

$C_i \leftarrow E_K(P_i)$

O modo CBC

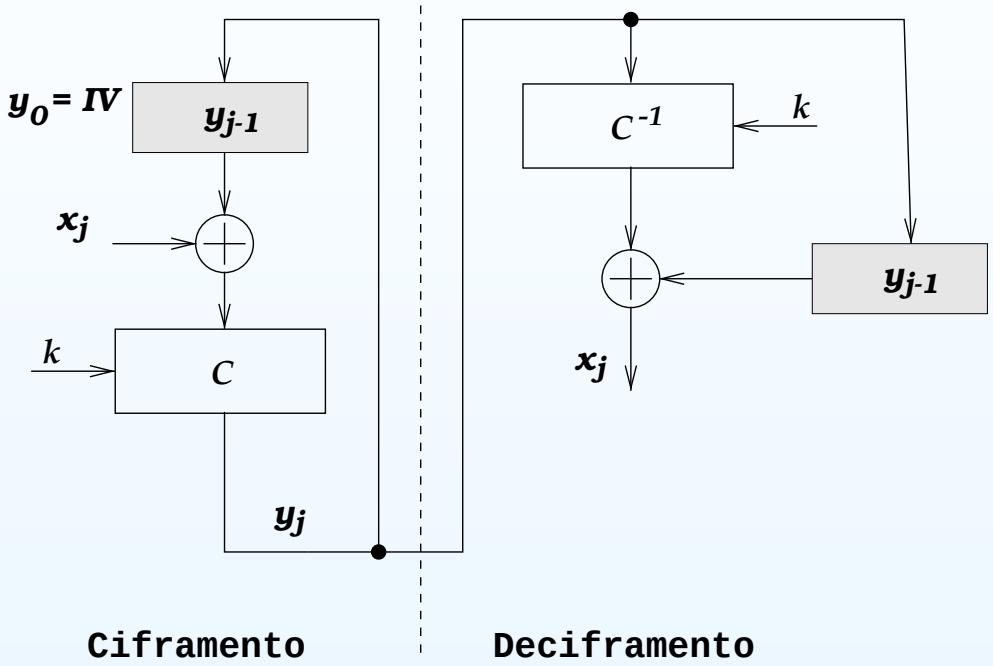
Cipher Block Chaining

Para cifrar:

Entrada: IV, $P_1, P_2, ..., P_m$, $|P_i| = n$ bits, chave K.
Saída: IV, $C_1, C_2, ..., C_m$, $|C_i| = n$ bits.

1. $C_0 \leftarrow IV$
2. **for** $i = 1$ **to** m **do**
   $C_i \leftarrow E_K(P_i \oplus C_{i-1})$

O modo CBC



O modo CBC

Cipher Block Chaining

Para decifrar:

Entrada: IV, $C_1, C_2, ..., C_m$, $|C_i| = n$ bits, chave K.
Saída: IV, $P_1, P_2, ..., P_m$, $|P_i| = n$ bits.

1. $C_0 \leftarrow IV$
2. **for** $i = 1$ **to** m **do**
   $P_i \leftarrow D_K(C_i) \oplus C_{i-1}$

O modo CTR

Counter

Para cifrar:

Entrada: Crt, $P_1, P_2, ..., P_m$, $|P_i| = n$ bits, chave K
Saída: Ctr, $C_1, C_2, ..., C_m$, $|C_i| = n$ bits.

1. **for** $i = 1$ **to** m **do**

$$C_i \leftarrow P_i \oplus E_K(Ctr + i \bmod n)$$

Contador:

$$crt \rightarrow ctr + 1 \rightarrow ctr + 2 \rightarrow \cdots ctr + m - 1$$

O modo CTR

Counter

Para decifrar:

Entrada: $crt, C_1, C_2, ..., C_m,\ |C_i| = n$ bits, chave K.
Saída: $Ctr, P_1, P_2, ..., P_m,\ |P_i| = n$ bits.

1. **for** $i = 1$ **to** m **do**
   $P_i \leftarrow C_i \oplus E_K(Ctr + i \bmod n)$

Resumos criptográficos

- Uma *função hash* ou *função de resumo* é uma função que calcula uma representação condensada de uma mensagem ou arquivo. Mais precisamente, uma função de resumo recebe como entrada uma cadeia de bits de comprimento arbitrário e devolve outra cadeia de bits de comprimento fixo, chamado resumo.
- As funções de resumo podem ser utilizadas em aplicações criptográficas, tais como autenticação de mensagems enviadas através de canais inseguros e assinaturas digitais.

Funções de resumo

Uma função de resumo criptográfico (ou função de resumo) é uma função $H : \{0,1\}^* \rightarrow \{0,1\}^n$, satisfazendo as seguintes propriedades:

- **Resistência à primeira inversão:** Dado um resumo r, é inviável encontrar uma mensagem m tal que $r = H(m)$.
- **Resistência à segunda inversão:** Dado um resumo r e uma mensagem m_1 tal que $r = H(m_1)$, é inviável encontrar uma mensagem $m_2 \neq m_1$ tal que $r = H(m_2)$.
- **Resistência a colisões:** Dado um resumo r, é inviável encontrar mensagens m_1, m_2 tais que $H(m_1) = H(m_2)$.

Algoritmos: MD5, SHA-1, SHA-2, Whirlpool.

O Algoritmo MD5

$$\mathrm{MD5}(x) : \{0,1\}^{2^{64}} \rightarrow \{0,1\}^{128}$$

- Proposto por Ronald Rivest em 1991
- MD5 é uma versão melhorada de MD4
- Ataques: 1996, 2004, 2005, 2006 (Vlastimil Klima)
- Operações: $a + b \mod 2^{32}$, $a \oplus b$, $a \wedge b$, $a \vee b$, $\neg a$, $a << s$ (ROTL$^s(a)$).

O Algoritmo SHA - The Secure Hash Algorithm

$$\mathrm{SHA}(x) : \{0,1\}^{2^{64}} \rightarrow \{0,1\}^{160}$$

- Proposto por NIST em 1993 (FIPS 180-1)
- $\mathrm{SHA}-1$ é uma versão melhorada de SHA
- Novos algoritmos (FIPS 180-2, 2002): $\mathrm{SHA}-224^{*}$, $\mathrm{SHA}-256$, $\mathrm{SHA}-384$, $\mathrm{SHA}-512$
- Operações: $a + b \bmod 2^{32}$, $a \oplus b$, $a \wedge b$, $a \vee b$, $\neg a$, $a \ll s$ ($\mathrm{ROTL}^{s}(a)$).

O Algoritmo SHA-1

Entrada: x; **Saída:** SHA(x)

1. $M^1 ... M^n \leftarrow$ Pad(x)
2. $H_0, H_1, H_2, H_3, H_4 \leftarrow$ valores inicias
3. For $i = 1$ to n do:
   - 3.1 Calcule $W_0, W_1, ..., W_{79}$ do bloco M^i
   - 3.2 $A \leftarrow H_0, B \leftarrow H_1, C \leftarrow H_2, D \leftarrow H_3, E \leftarrow H_4$
   - 3.3 For $i = 0$ to 79 do:
     $$[A, B, C, D, E] \leftarrow Th([A, B, C, D, E])$$
   - 3.4 $H_0 \leftarrow H_0 + A$, $H_1 \leftarrow H_1 + B$, $H_2 \leftarrow H_2 + C$, $H_3 \leftarrow H_3 + D$, $H_4 \leftarrow H_4 + E$
4. Return $(H_0, H_1, H_2, H_3, H_4)$

O Algoritmo Whirlpool

- O algoritmo *Whirlpool* foi desenvolvido por Vicent Rijmen e Paulo Barreto, 2003.
- Adotado como o padrão ISO/IEC 10118-3:2004
- O algoritmo Whirlpool é uma função de resumo que processa mensagens de comprimento menor do que 2^{256} bits para gerar um resumo de 512 bits.
- O algoritmo está baseado numa função de compressão W

Algoritmos assimétricos

1. Funções Unidirecionais
2. Assinatura Digital
3. Criptossistemas de Curvas Elípticas (CCE)
4. Aspectos da Implementação em Software de CCE

Funções unidirecionais: RSA

p, q: números primos
$n = p \cdot q$
$\phi = (p-1) \cdot (q-1)$
Escolha um número e, $1 < e < \phi$, tal que mdc$(e, \phi) = 1$, exemplo: $e = 2^{16} + 1$.

$$f_{\mathrm{RSA}}(x) := x^e \bmod n.$$

Como calcular $f_{\mathrm{RSA}}^{-1}(x)$? fácil se temos p e q:
aplique o algoritmo de Euclides estendido para calcular o único número d, $1 < d < \phi$, tal que $ed \equiv 1 (\bmod \phi)$. Então

$$f_{\mathrm{RSA}}^{-1}(x) = x^d \bmod n$$

Funções unidirecionais: Rabin

p, q: números primos tais que $p \equiv q \equiv 3 \pmod 4$

$n = p \cdot q$

$$f_{\text{Rabin}}(x) := x^2 \mod n.$$

Como calcular $f^{-1}_{\text{Rabin}}(x)$? fácil se temos p e q:

1. Aplique o algoritmo de Euclides estendido para calcular a e b tal que $ap + bq = 1$.
2. Calcule $r = x^{(p+1)/4} \mod p$ e $s = x^{(q+1)/4} \mod q$.
3. Calcule $t_1 = (aps + bqr) \mod n$ e $t_2 = (aps - bqr) \mod n$.

Então

$$f^{-1}_{\text{Rabin}}(x) \in \{r, s, t_1, t_2\}$$

Funções Unidirecionais: exponenciação em $\mathbb{Z}_p^*$

p: um número primo
α: um gerador de $\mathbb{Z}_p^*$

$$f_{\exp}(x) := \alpha^x \mod p.$$

Como calcular $f_{\exp}^{-1}(x)$? cálculo de "logaritmos discretos":
O problema do logaritmo discreto é o seguinte:
Dado um primo p, um gerador α de $\mathbb{Z}_p^*$, e um elemento $\beta \in \mathbb{Z}_p^*$, encontrar o inteiro x, $0 \leq x \leq p-2$, tal que $\alpha^x \equiv \beta \pmod p$.

Exemplo: cálculo de logaritmos discretos

$p = 7$: um número primo
$\alpha = 3$: um gerador de $\mathbb{Z}_7^* = \{1, 2, 3, 4, 5, 6\}$

i	$3^i \mod 7$
0	1
1	3
2	2
3	6
4	4
5	5
6	1

$$3^i \equiv 6 \pmod 7; \quad i = 3$$

RSA: encriptação

Geração de Chaves:

1. Gere dois números primos aleatórios p e q (distintos)
2. Calcule $n = p \cdot q$
3. Calcule $\phi = (p-1) \cdot (q-1)$
4. Escolha um número e, $1 < e < \phi$, tal que mdc$(e, \phi) = 1$, exemplo: $e = 2^{16} + 1$.
5. Aplique o algoritmo de Euclides estendido para calcular o único número d, $1 < d < \phi$, tal que $ed \equiv 1 \pmod{\phi}$.
6. Chave privada: d
7. Chave pública: e, n

RSA: encriptação

Algoritmo para encriptar

Para cifrar uma mensagem m para Alice, Bob faz o seguinte:

1. Obtem a chave pública (autêntica) de Alice e, n
2. Representa a mensagem m como um inteiro em $\{0, 1, 2, ...n-1\}$.
3. Calcula $c = m^e \mod n$
4. Envia o texto cifrado c para Alice.

RSA: encriptação

Algoritmo para decifrar:

Para recuperar o texto claro m a partir de c, Alice faz o seguinte:

1. Utiliza a chave privada d para calcular o texto claro m:

$$m = c^d \mod n$$

Assinaturas Digitais

The Digital Signature Standard, 1991

Parâmetros

p, q primos tais que $q|p-1$, $|p| = 1024, |q| = 160$

$g \in \mathbb{Z}_p^*$ de ordem q

$H : \{0,1\}^* \rightarrow \mathbb{Z}_q$ (H = SHA-1)

$x \in \mathbb{Z}_q$

$y = g^x \mod p$

chave Privada: x

chave Pública: p, q, g, y, H

O algoritmo de assinatura DSA

Geração(M, x):

$k \in_U \mathbb{Z}_q$

$r \leftarrow (g^k \pmod p) \pmod q,$

$s \leftarrow k^{-1}(H(M) + xr) \pmod q$

return (r, s)

O algoritmo de assinatura DSA

Verificação $(M, (r, s), p, q, g, y, H)$:

$w \leftarrow s^{-1} \pmod{q}$,

$u_1 \leftarrow H(M)w \pmod{q}$,

$u_2 \leftarrow rw \pmod{q}$,

se $(r = (g^{u_1} y^{u_2} \pmod{p}) \pmod{q})$ então devolva SIM

senão devolva NÃO.

Criptossistemas de Curvas Elípticas

- Inventados por Victor Miller e Neal Koblitz em 1985.
- Os CCE são sistemas de chave pública.
- Padrões: ANSI X9.62, IEEE P1363, FIPS 162-2, SEC 1-2, NIST
- Aplicações: cartões inteligentes, celulares, redes de sensores sem fio, etc.
- Companhias: Certicom, RSA Security, Cryptomathic, HITACHI.

Criptossistemas de Curvas Elípticas

A principal vantagem dos CCE é que utilizam chaves de comprimento menor com o mesmo nível de segurança oferecido por outros sistemas de chave pública (RSA, DSA).

CCE-160 vs RSA-1024

CCE	RSA	AES
224	2048	-
256	3072	128
384	8192	192
512	15360	256

Nível de Segurança em bits

ECRYPT - European Network of Excellence for Cryptology

ECRYPT Yearly Report on Algorithms and Keysizes (2006)

CCE	RSA	C-Bloco
160	1248	80
224	2432	112
256	3248	128
384	7936	192
512	15424	256

NÃvel de Segurança em bits

Criptossistemas de Curvas Elípticas

Aspectos Matemáticos

Curvas Elípticas sobre $\mathbb{F}_q$

Uma curva elíptica E sobre $\mathbb{F}_q$ é definida por uma equação $e_q(x, y)$ da forma:
(**se** $q = 2^m$)

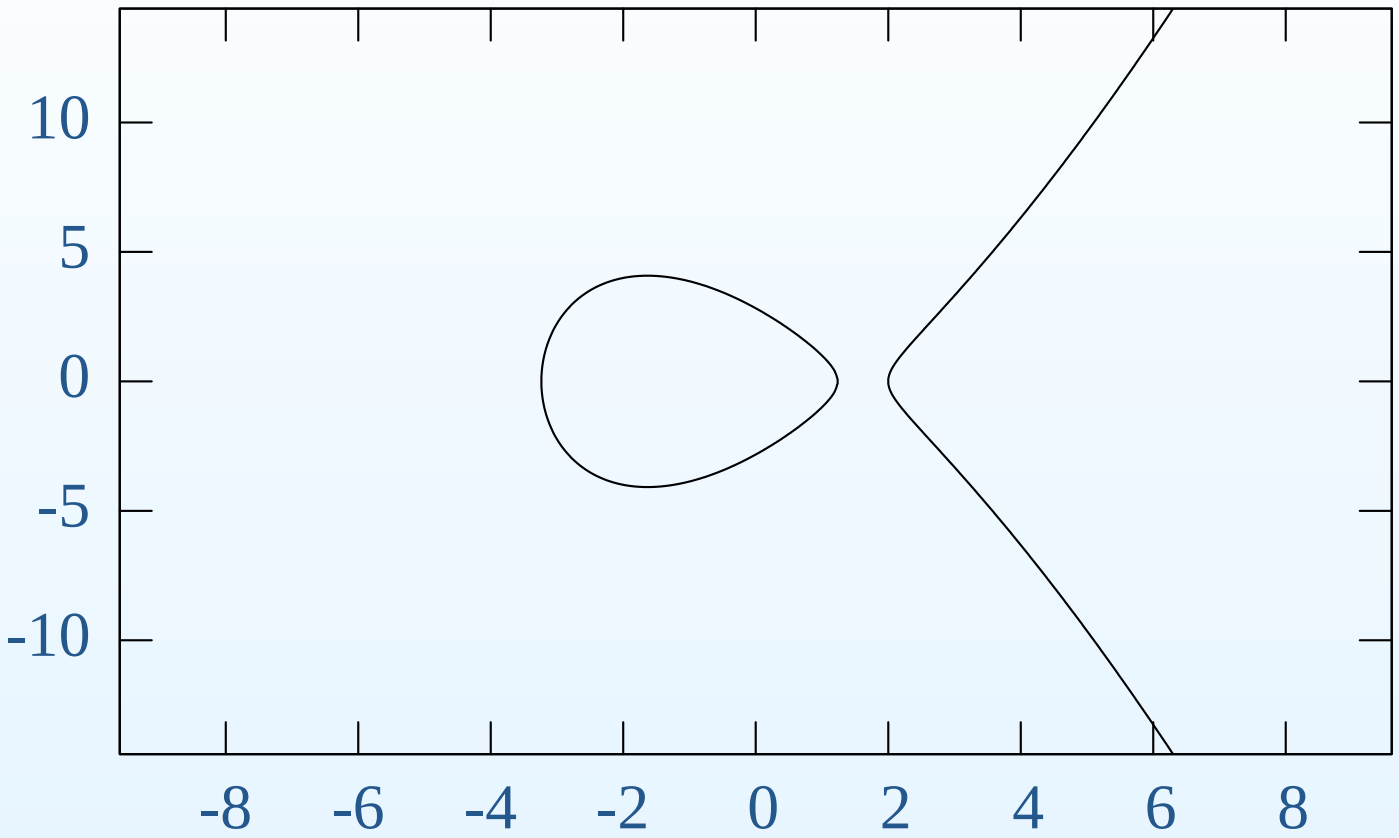
$$\boxed{y^2 + xy = x^3 + ax^2 + b}$$

onde $a, b \in \mathbb{F}_{2^m}$ e $b \neq 0$.
(**se** $q = p, p > 3$)

$$\boxed{y^2 = x^3 + ax + b}$$

onde $a, b \in \mathbb{F}_p$, e $4a^3 + 27b^2 \neq 0 \pmod{p}$.

Curvas Elípticas sobre $\mathbb{R}$: $y^2 = x^3 + ax + b$



Curvas Elípticas sobre $\mathbb{F}_{29}$

$$E : y^2 = x^3 + 4x + 20,\ p = 29$$

O conjunto de pontos de $E(\mathbb{F}_{29})$ é:

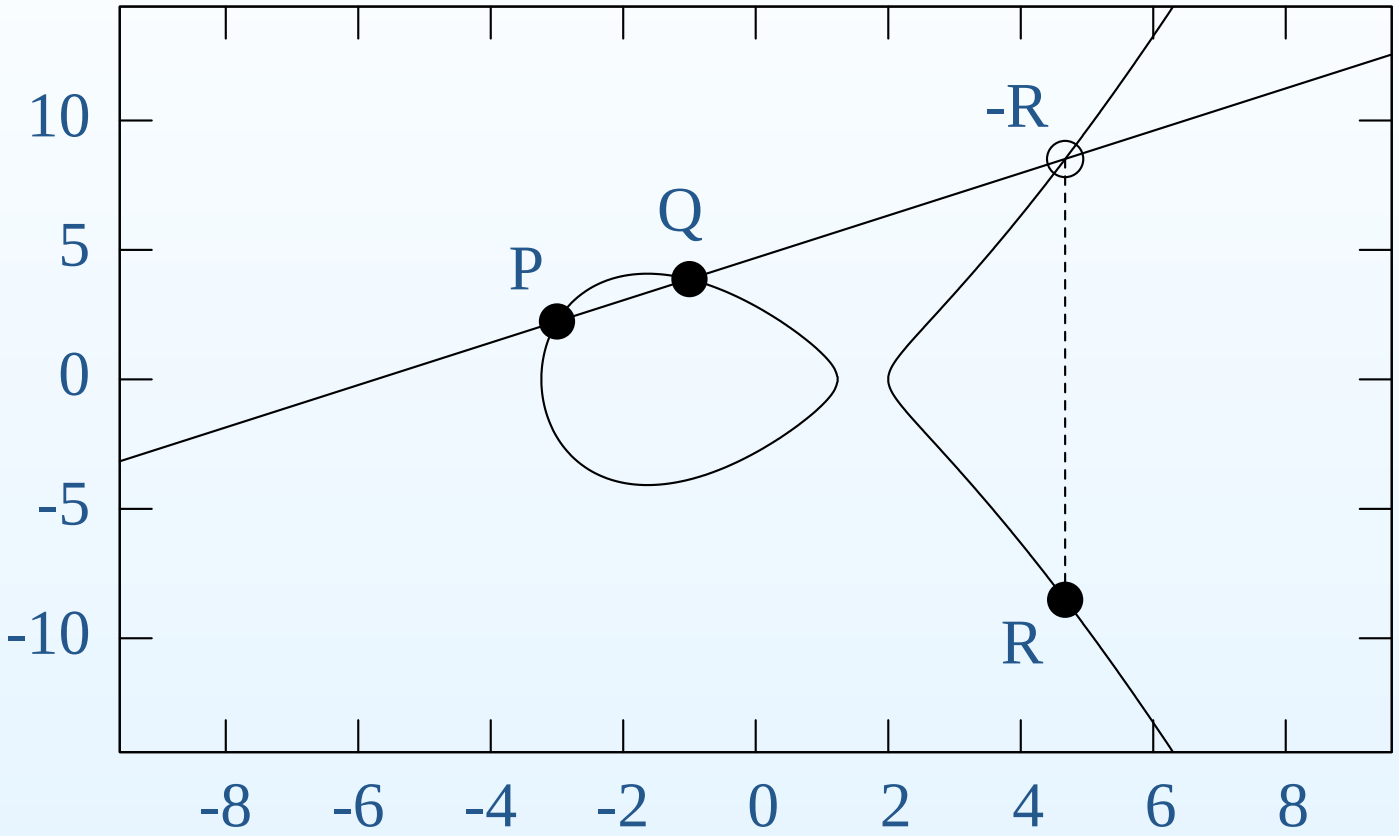
$\mathcal{O}$	$(2, 6)$	$(4, 19)$	$(8, 10)$	$(13, 23)$	$(16, 2)$	$(19, 16)$
$(0, 7)$	$(2, 23)$	$(5, 7)$	$(8, 19)$	$(14, 6)$	$(16, 27)$	$(20, 3)$
$(0, 22)$	$(3, 1)$	$(5, 22)$	$(10, 4)$	$(14, 23)$	$(17, 10)$	$(20, 26)$
$(1, 5)$	$(3, 28)$	$(6, 12)$	$(10, 25)$	$(15, 2)$	$(17, 19)$	$(24, 7)$
$(1, 24)$	$(4, 10)$	$(6, 17)$	$(13, 6)$	$(15, 27)$	$(19, 13)$	$(24, 22)$
$(27, 2)$	$(27, 27)$					

$$\#E(\mathbb{F}_{29}) = 37$$

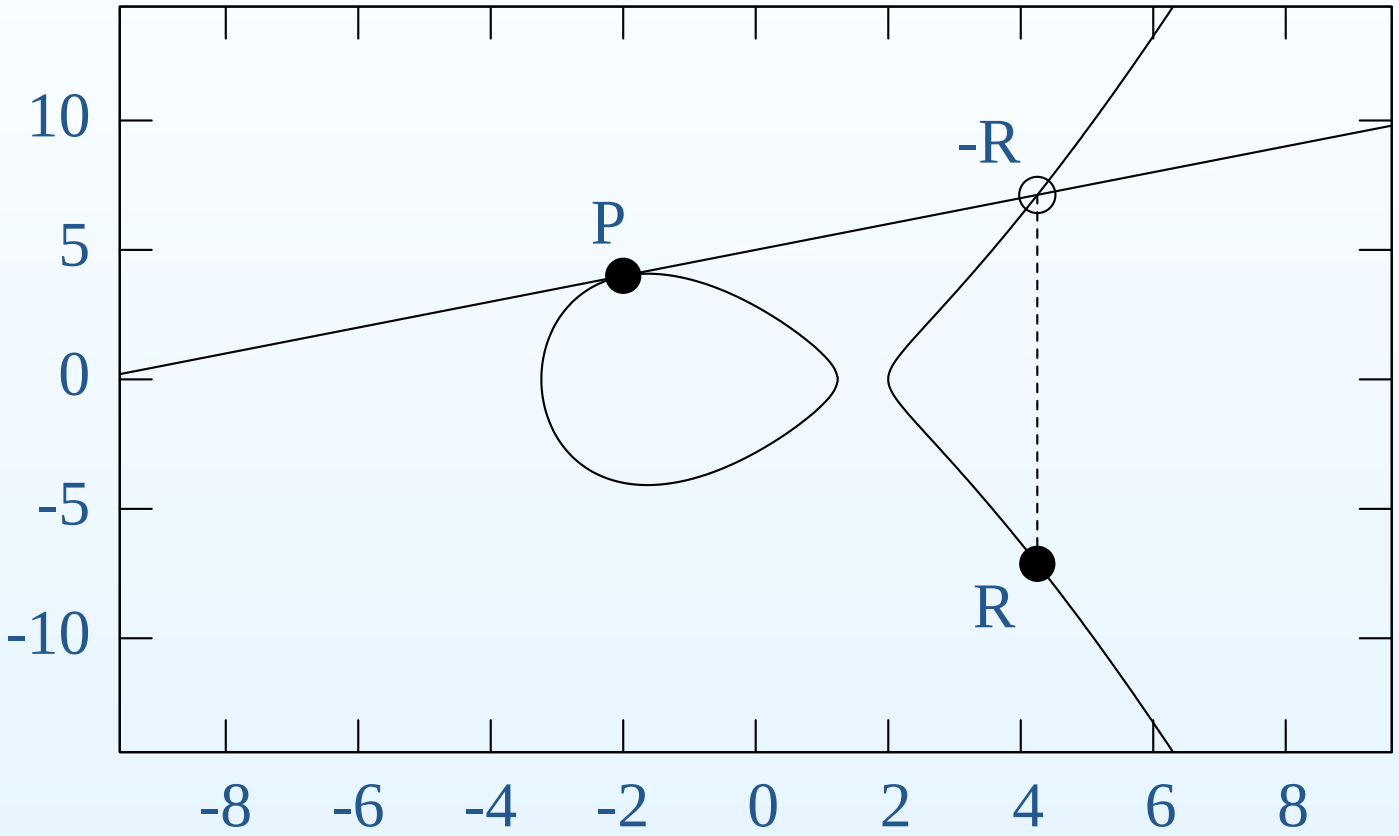
Grupo Elíptico $(E(\mathbb{F}_q), +)$

- $E(\mathbb{F}_q) := \{(x, y) \mid x \in \mathbb{F}_q, y \in \mathbb{F}_q, e_q(x, y) = 0\} \cup \{\mathcal{O}\}$
- $(E(\mathbb{F}_q), +)$ forma um grupo abeliano com identidade $\mathcal{O}$.
- Para $P = (x_1, y_1) \in E(\mathbb{F}_q)$ y $Q = (x_2, y_2) \in E(\mathbb{F}_q)$, a soma $P + Q$ é definida da seguinte forma: (secante e tangente).

Soma em $E(\mathbb{R})$: $R = P + Q$



Soma em $E(\mathbb{R})$: $R = P + P = 2P$



Grupo Elíptico $(E(\mathbb{F}_p), +)$

- $P + \mathcal{O} = P + \mathcal{O} = P$
- Se $P = (x, y)$, então $-P = (x, -y)$; $P + (-P) = \mathcal{O}$.
- Duplicação:
  Se $P = (x_1, y_1) \in E(\mathbb{F}_p)$, então
  $P + P = 2P = (x_3, y_3)$, onde

$$\begin{aligned} x_3 &= \lambda^2 - 2x_1 \\ y_3 &= \lambda \cdot (x_1 - x_3) - y_1 \\ \lambda &= \frac{3x_1^2 + a}{2y_1} \end{aligned}$$

Grupo Elíptico $(E(\mathbb{F}_p), +)$

- Soma:
  Se $P = (x_1, y_1) \in E(\mathbb{F}_p), Q = (x_2, y_2) \in E(\mathbb{F}_p)$, $P \neq \pm Q$, então $P + Q = (x_3, y_3)$, onde

$$\begin{aligned} x_3 &= \lambda^2 - x_1 - x_2 \\ y_3 &= \lambda \cdot (x_1 - x_3) - y_1 \\ \lambda &= \frac{y_2 - y_1}{x_2 - x_1} \end{aligned}$$

Aritmética Finita sobre $\mathbb{F}_p$ e $\mathbb{F}_n$ - Pentium4 2.4 GHz

Operações aritméticas	$\mathbb{F}_{p_{192}}$ IPP	$\mathbb{F}_{p_{192}}$ GMP	$\mathbb{F}_{n_{192}}$ IPP	$\mathbb{F}_{n_{192}}$ GMP
Soma	0.092	0.071	0.102	0.071
Quadrado	0.765	0.493	2.095	1.086
Multiplicação	0.758	0.489	2.092	1.086
Inversão	35.770	12.508	35.712	12.624

Tempos em microsegundos, gcc, Linux, 32 MB RAM, IPP (Intel).

$\mathbb{F}_{p_{192}}$: eficiência de kP

$\mathbb{F}_{n_{192}}$: eficiência dos protocolos criptográficos (ECDSA etc).

Aritmética Finita sobre $\mathbb{F}_p$ e $\mathbb{F}_n$ - Pentium4 2.4 GHz

Operações aritméticas	$\mathbb{F}_{p_{256}}$ IPP	$\mathbb{F}_{p_{256}}$ GMP	$\mathbb{F}_{n_{256}}$ IPP	$\mathbb{F}_{n_{256}}$ GMP
Soma	0.095	0.071	0.105	0.071
Quadrado	1.598	1.172	2.420	1.694
Multiplicação	1.625	1.184	2.444	1.698
Inversão	57.169	16.997	56.916	17.227

Tempos em microsegundos, gcc, Linux, 32 MB RAM, IPP (Intel).

Multiplicação Escalar

Dado um número inteiro k e um ponto elíptico $P \in E(\mathbb{F}_q)$

$$kP := \underbrace{P + \ldots + P}_{k-\text{vezes}}.$$

Exemplo: $14P = 8P + 6P = 2^3P + 2^2P + 2P$

3 duplicações 2 somas para calcular $14P$

Algoritmos eficientes para calcular kP:

- kP, P ponto conhecido (off-line)
- kP, P ponto arbitrário (on-line)
- $k_1P + k_2Q$

Método binário para calcular kP

$$\begin{aligned} kP &= \sum_{i=0}^{l-1} k_i 2^i P \\ &= 2(\cdots 2(2P + k_{l-2}P) + \cdots) + k_0 P \end{aligned}$$

$k =$

1	k_{l-2}	k_{l-3}	$\cdots$	$\cdots$	k_1	k_0

$Q \leftarrow P$
for $i = l - 2$ **to** 0 **do**
  $Q \leftarrow 2Q$
  if $k_i = 1$ **then** $Q \leftarrow Q + P$
endfor

O Problema do Logaritmo Discreto em Curvas Elípticas (PLDCE)

Dado um ponto $P \in E(\mathbb{F}_q)$ de ordem n ($nP = \mathcal{O}$), e um ponto $Q \in < P >$, determinar o inteiro k, $0 \leq k \leq n-1$, tal que

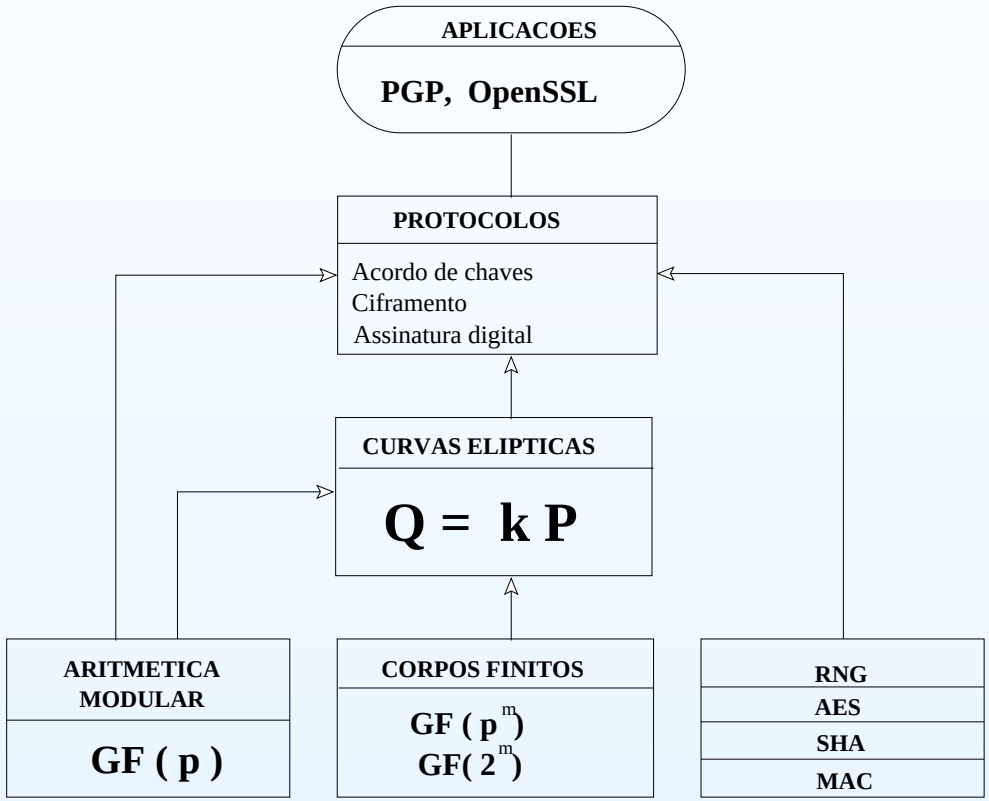
$$Q = kP.$$

A segurança dos CCE está baseada na dificultade do PLDCE.

Ataques ao PLDCE

- O melhor algoritmo conhecido para o PLDCE é o método ρ de Pollard [1978] que é de complexidade exponencial: $O(\sqrt{n})$ onde n é número de pontos da curva elíptica.
  (parâmetros pequenos)
- Desafio de Certicom [1999]: ECC2-109 (2004) solução utilizando uma versão distribuída de ρ Pollard (17 meses, 2600 computadores)
- Padrões Industriais : comprimentos das chaves 160-571 bits.
- A Agência de Segurança Nacional NSA (2005) recomendou ao governo americano chaves de 256 bits para os CCE definidos sobre corpos primos.

Estrutura dos CCE



Chaves nos CCE

$$G \in E(\mathbb{F}_q), nG = \mathcal{O}$$

- Chave privada: um número aleatório $\boxed{k}$; $0 < k < n$
- Chave pública: o ponto elíptico $\boxed{kG.}$

Componentes dos CCE

- Um corpo finito $\mathbb{F}_q$ ($q = p, q = 2^m, q = p^m$)
- Uma base para $\mathbb{F}_q$
- Uma curva $E(\mathbb{F}_q)$ ($a = ?, b = ?$)
- Um ponto $G \in E(\mathbb{F}_q)$ de ordem n
- Protocolos (ECDH, ECIES, ECDSA)

Parâmetros dos CCE

$$\boxed{(\mathbb{F}_q, FR, a, b, G, n, h)}$$

$$\#E(\mathbb{F}_q) = n \cdot h, \;\; nG = \mathcal{O}$$

- Curvas Padrão (especiais ou aleatórias)
  - National Institute of Standards and Technology (NIST)
  - Standards for Efficiency Cryptography Group (SECG)
- Curvas geradas pelo usuário
  - Dados a e b calcular $n = \#E(\mathbb{F}_q)$
  - Dado $n = \#E(\mathbb{F}_q)$,calcular a, b (o método CM)

Parâmetros NIST

- $\mathbb{F}_{2^m}$, $m \in \{163, 233, 283, 409, 571\}$.
- $\mathbb{F}_p$, $\lceil \log_2 p \rceil \in \{192, 224, 256, 384, 521\}$.
- Curvas:
  - curvas aleatórias sobre $\mathbb{F}_{2^m}$
  - curvas de Koblitz (sobre $\mathbb{F}_{2^m}$) ($a \in \{0, 1\}, b = 1$)
  - curvas aleatórias sobre $\mathbb{F}_p$

Parâmetros SECG

- $\mathbb{F}_{2^m}$: $m \in \{113, 131, 163, 233, 239, 283, 409, 571\}$.
- $\mathbb{F}_p$: $\lceil \log_2 p \rceil \in \{112, 128, 160, 192, 224, 256, 384, 521\}$
- Curvas
  - curvas aleatórias sobre $\mathbb{F}_{2^m}$
  - curvas de Koblitz (sobre $\mathbb{F}_{2^m}$)
  - curvas aleatórias sobre $\mathbb{F}_p$
  - curvas de "Koblitz" sobre $\mathbb{F}_p$

Exemplo : parâmetros NIST y SECG

Curva aleatória sobre $\mathbb{F}_p$

- $E : Y^2 = X^3 + aX + b \; (a = -3)$
- $P-192 : p = 2^{192} - 2^{64} - 1, \; h = 1, \; r \cdot b^2 \equiv a^3$
- S = `0x3045AE6F C8422F64 ED579528 D38120EA E12196D5`
- r = `0x3099D2BB BFCB2538 542DCD5F B078B6EF 5F3D6FE2 C745DE65`
- b = `0x64210519 E59C80E7 0FA7E9AB 72243049 FEB8DEEC C146B9B1`
- n = `0xFFFFFFFF FFFFFFFF FFFFFFFF 99DEF836 146BC9B1 B4D22831`
- G_x = `0x188DA80E B03090F6 7CBF20EB 43A18800 F4FF0AFD 82FF1012`
- G_y = `0x07192B95 FFC8DA78 631011ED 6B24CDD5 73F977A1 1E794811`

Implementação em Software

Aspectos Práticos

Curvas Elípticas sobre $\mathbb{F}_{2^m}$

- Corpos finitos $\mathbb{F}_{2^m}$ $m \in \{163, 233, 283\}$
- Curvas: aleatórias e Koblitz
- Protocolos: ECDSA, ECIES
- Plataformas: Pager-RIM, PalmPilot V, Pentium II-III
- Linguagem : C
- Aplicação : PGP
- 1999-2000

Exemplo 1: Pager-RIM

- Processador: Intel x86 (custom 386) 10 MHz
- Memória RAM: 304 kbytes
- Peso: 142 gramas
- Pilhas: 1 AA

Curvas Elípticas sobre $\mathbb{F}_{2^m}$ Pager RIM

Operações criptográficas	Curvas aleatórias	Curvas Koblitz	RSA-1024 $e = 2^{16} + 1$
Gerar chaves	1.08	0.75	600
Cifrar	3.13	1.76	1.24
Decifrar	2.11	1.06	15.90
Assinar	1.33	1.01	15.88
Verificar	3.24	1.82	1.01

Tempos em segundos

Curvas Elípticas sobre $F_{2^{233}}$ - Pager RIM

Operações criptográficas	Curvas aleatórias	Curvas Koblitz	RSA-2048 $e = 2^{16} + 1$
Gerar chaves	2.47	1.55	–
Cifrar	6.91	3.47	4.14
Decifrar	4.59	2.00	112.09
Assinar	3.06	1.91	111.95
Verificar	7.32	3.7	3.60

Tempos em segundos

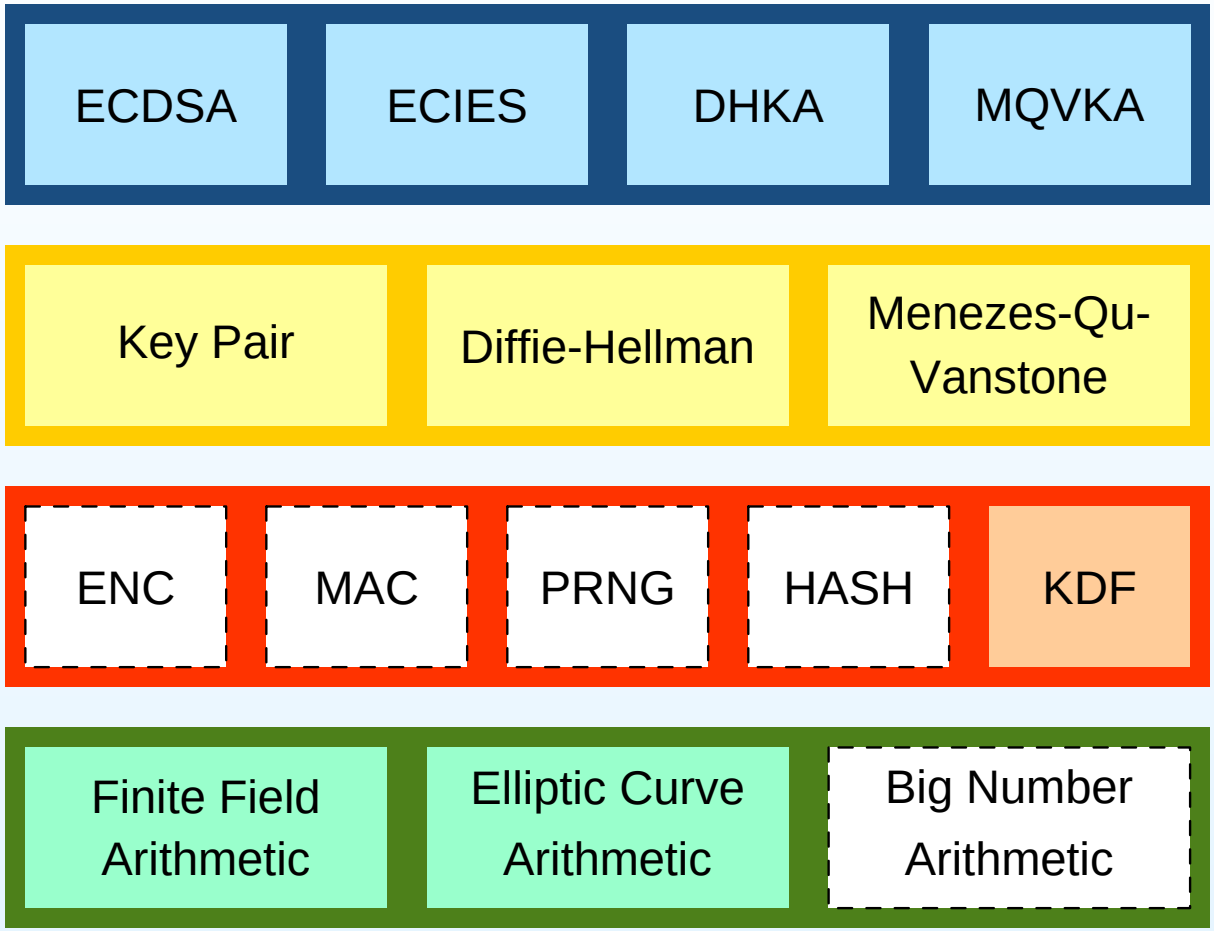
Implementações dos CCE

- Nível de segurança
- Plataforma computacional
- Restrições na plataforma computacional
- Algoritmos para as operações aritméticas em $\mathbb{F}_q$
- Algoritmos para a operação kP
- Protocolos
- Patentes
- Padrões Industriais

Curvas Elípticas sobre $\mathbb{F}_{2^m}$ e $\mathbb{F}_p$ - LCA-IC

- Biblioteca Otimizada
- Corpos finitos: NIST 160-570 bits
- Curvas: aleatórias e Koblitz
- Protocolos: ECDSA, ECIES, ECMQV
- Plataformas: XScale Intel PXA 27x, Pentium 4
- Linguagem : C
- Biblioteca Intel IPP (Integrated Performance Primitives)
- 2005-2007

Arquitetura da Biblioteca CE



Exemplo 2: PXA270- Mainstone II

Intel Xscale 520 MHz (wMMX)

Curvas Elípticas sobre $\mathbb{F}_{p_{192}}$ - XScale-PXA270- Mainstone II 520 MHz

Operações criptográficas	IPP	GMP	ASM
Gerar chaves	5.87	4.87	1.67
Assinar	6.24	5.17	1.93
Verificar	19.96	16.29	5.29

Tempos em milisegundos, gcc, Linux, 64 KB Cache, IPP (Intel).

Curvas Elípticas sobre $\mathbb{F}_{p_{256}}$ - XScale-PXA270- Mainstone II 520 MHz

Operações criptográficas	IPP gcc	GMP gcc	ASM gcc
Gerar chaves	13.53	13.94	4.02
Assinar	14.18	14.16	4.38
Verificar	46.82	45.96	13.12

Tempos em milisegundos, gcc, Linux, 64 KB Cache, IPP (Intel).

Curvas Elípticas sobre $\mathbb{F}_{p_{256}}$ - Pentium4 2.4 GHz

Operações criptográficas	IPP gcc	GMP gcc	IPP icc	GMP icc
Gerar chaves	1.78	1.22	1.72	1.20
Assinar	1.84	1.24	1.80	1.22
Verificar	5.96	4.04	5.84	3.88

Tempos em milisegundos, gcc,icc, Linux, 512 KB Cache, IPP (Intel).

Conclusões

- Os CCE oferecem a melhor alternativa para criptografia pública em diferentes plataformas computacionais.
- Segurança e eficiência dos CCE: mais pesquisa.
- Implementação em Software:
  - Tecnologia wMMX para $\mathbb{F}_q$
  - Ponto flutuante para $\mathbb{F}_p$
  - Novas arquiteturas: Tecnologia *Multi-core*
  - Implementações em Hardware/Software.

Protocolos criptográficos

Protocolos criptográficos

Gerenciamento de chaves

Identificação (autenticação de entidades)

Estabelecimento de chaves

Gerenciamento de chaves

- Atividade crítica em qualquer sistema criptográfico.

Gerenciamento de chaves

- Atividade crítica em qualquer sistema criptográfico.
- Chaves devem ser
  criteriosamente geradas e disponibilizadas aos seus legítimos donos;

Gerenciamento de chaves

- Atividade crítica em qualquer sistema criptográfico.
- Chaves devem ser
  criteriosamente geradas e disponibilizadas aos seus legítimos donos;
  cuidadosamente empregadas e armazenadas;

Gerenciamento de chaves

- Atividade crítica em qualquer sistema criptográfico.
- Chaves devem ser

  criteriosamente geradas e disponibilizadas aos seus legítimos donos;

  cuidadosamente empregadas e armazenadas;

  ao fim da sua vida útil, ser destruídas ou arquivadas seguramente.

Gerenciamento de chaves

- Atividade crítica em qualquer sistema criptográfico.
- Chaves devem ser
  criteriosamente geradas e disponibilizadas aos seus legítimos donos;
  cuidadosamente empregadas e armazenadas;
  ao fim da sua vida útil, ser destruídas ou arquivadas seguramente.

Aqui, damos atenção ao aspecto do estabelecimento (criação e disponibilização) de chaves, pois é o que depende mais fortemente de técnicas criptográficas.

Tipos de chaves

Pelo seu tempo de vida:

Tipos de chaves

Pelo seu tempo de vida:

- **Chaves de sessão.** Efêmeras, simétricas usadas durante uma sessão de comunicação e depois descartadas.

Tipos de chaves

Pelo seu tempo de vida:

- **Chaves de sessão.** Efêmeras, simétricas usadas durante uma sessão de comunicação e depois descartadas.

  Protocolos divididos em duas classes: *distribuição de chaves* e *acordo de chaves*.

Tipos de chaves

Pelo seu tempo de vida:

- **Chaves de sessão.** Efêmeras, simétricas usadas durante uma sessão de comunicação e depois descartadas.

  Protocolos divididos em duas classes: *distribuição de chaves* e *acordo de chaves*.
- **Chaves de longa vida.** Chaves não-efêmeras, simétricas ou assimétricas, usadas com certa freqüência para

Tipos de chaves

Pelo seu tempo de vida:

- **Chaves de sessão.** Efêmeras, simétricas usadas durante uma sessão de comunicação e depois descartadas.

  Protocolos divididos em duas classes: *distribuição de chaves* e *acordo de chaves*.
- **Chaves de longa vida.** Chaves não-efêmeras, simétricas ou assimétricas, usadas com certa freqüência para

  distribuição encriptada de chaves de sessão;
  confecção e verificação de assinaturas digitais;
  confecção de MACs.

Tipos de chaves

Pelo seu tempo de vida:

- **Chaves de sessão.** Efêmeras, simétricas usadas durante uma sessão de comunicação e depois descartadas.

  Protocolos divididos em duas classes: *distribuição de chaves* e *acordo de chaves*.
- **Chaves de longa vida.** Chaves não-efêmeras, simétricas ou assimétricas, usadas com certa freqüência para

  distribuição encriptada de chaves de sessão;
  confecção e verificação de assinaturas digitais;
  confecção de MACs.

Tipos de chaves

- **Chaves de longa vida (cont.)** Chaves simétricas são distribuídas com protocolos de *pré-distribuição* de chaves e armazenadas ou em hierarquias de chaves ou hardware dedicado, como *smartcards* e HSMs, ou frases-senha (*passphrases*).
  Chaves públicas devem ser distribuídas nos seus certificados.

Tipos de chaves

- **Chaves de longa vida (cont.)** Chaves simétricas são distribuídas com protocolos de *pré-distribuição* de chaves e armazenadas ou em hierarquias de chaves ou hardware dedicado, como *smartcards* e HSMs, ou frases-senha (*passphrases*).
  Chaves públicas devem ser distribuídas nos seus certificados.
- **Chaves perenes.** Chaves críticas, de vida muito longa, como chaves mestras e chaves privadas de autoridades certificadoras. Armazenamento combina HSMs e fracionamento da chave.

Certificação de chaves públicas

Certificação é a forma de vincular uma chave pública à entidade que é sua legítima dona, ou seja, de prover confiança a um sistema de chaves públicas.

Certificação de chaves públicas

Certificação é a forma de vincular uma chave pública à entidade que é sua legítima dona, ou seja, de prover confiança a um sistema de chaves públicas.
Na prática, alguém atesta (assina) a chave pública de outros. O documento de atestação é o certificado digital.

Certificação de chaves públicas

Certificação é a forma de vincular uma chave pública à entidade que é sua legítima dona, ou seja, de prover confiança a um sistema de chaves públicas.
Na prática, alguém atesta (assina) a chave pública de outros. O documento de atestação é o certificado digital.
Formas antagônicas para provimento dessa confiança:

Certificação de chaves públicas

Certificação é a forma de vincular uma chave pública à entidade que é sua legítima dona, ou seja, de prover confiança a um sistema de chaves públicas.
Na prática, alguém atesta (assina) a chave pública de outros. O documento de atestação é o certificado digital.
Formas antagônicas para provimento dessa confiança:

- **Auto-gerida.** Cada entidade mantém os certificados em que confia e troque certificados com entidades nas quais confia.

Certificação de chaves públicas

Certificação é a forma de vincular uma chave pública à entidade que é sua legítima dona, ou seja, de prover confiança a um sistema de chaves públicas.
Na prática, alguém atesta (assina) a chave pública de outros. O documento de atestação é o certificado digital.
Formas antagônicas para provimento dessa confiança:

- **Auto-gerida.** Cada entidade mantém os certificados em que confia e troque certificados com entidades nas quais confia.

  engajamento, custo baixo, difícil responsabilização.
  PGP (Phil Zimmerman).

Certificação de chaves públicas

Certificação é a forma de vincular uma chave pública à entidade que é sua legítima dona, ou seja, de prover confiança a um sistema de chaves públicas.
Na prática, alguém atesta (assina) a chave pública de outros. O documento de atestação é o certificado digital.
Formas antagônicas para provimento dessa confiança:

- **Auto-gerida.** Cada entidade mantém os certificados em que confia e troque certificados com entidades nas quais confia.
  engajamento, custo baixo, difícil responsabilização.
  PGP (Phil Zimmerman).
- **Delegação a terceiros.** Autoridades certificadoras, etc.

Certificação de chaves públicas

Certificação é a forma de vincular uma chave pública à entidade que é sua legítima dona, ou seja, de prover confiança a um sistema de chaves públicas.
Na prática, alguém atesta (assina) a chave pública de outros. O documento de atestação é o certificado digital.
Formas antagônicas para provimento dessa confiança:

- **Auto-gerida.** Cada entidade mantém os certificados em que confia e troque certificados com entidades nas quais confia.
  engajamento, custo baixo, difícil responsabilização.
  PGP (Phil Zimmerman).
- **Delegação a terceiros.** Autoridades certificadoras, etc.
  custo alto, fácil responsabilização.
  Infra-estruturas de chaves públicas (ICP/PKI).

Criptografia baseada em identidades

- Alternativa ao uso de certificados digitais.

Criptografia baseada em identidades

- Alternativa ao uso de certificados digitais.
- Proposta como esquema de assinaturas em 1984 por Adi Shamir.
- Ganhou impulso em 2001 com a primeira proposta prática de esquema de encriptação baseado em identidades, de Boneh e Franklin.

Criptografia baseada em identidades

- Alternativa ao uso de certificados digitais.
- Proposta como esquema de assinaturas em 1984 por Adi Shamir.
- Ganhou impulso em 2001 com a primeira proposta prática de esquema de encriptação baseado em identidades, de Boneh e Franklin.
- Chaves públicas auto-identificáveis: `'Esta chave pertence a alice@alice.com'`.

Criptografia baseada em identidades

- Alternativa ao uso de certificados digitais.
- Proposta como esquema de assinaturas em 1984 por Adi Shamir.
- Ganhou impulso em 2001 com a primeira proposta prática de esquema de encriptação baseado em identidades, de Boneh e Franklin.
- Chaves públicas auto-identificáveis: `'Esta chave pertence a alice@alice.com'`.
- Chaves privadas são geradas em conjunto pelo usuário e um *gerador de chaves privadas–GCP*.

Criptografia baseada em identidades

- Alternativa ao uso de certificados digitais.
- Proposta como esquema de assinaturas em 1984 por Adi Shamir.
- Ganhou impulso em 2001 com a primeira proposta prática de esquema de encriptação baseado em identidades, de Boneh e Franklin.
- Chaves públicas auto-identificáveis: `'Esta chave pertence a alice@alice.com'`.
- Chaves privadas são geradas em conjunto pelo usuário e um *gerador de chaves privadas–GCP*.

  Perda de independência das entidades.

Criptografia baseada em identidades

- Alternativa ao uso de certificados digitais.
- Proposta como esquema de assinaturas em 1984 por Adi Shamir.
- Ganhou impulso em 2001 com a primeira proposta prática de esquema de encriptação baseado em identidades, de Boneh e Franklin.
- Chaves públicas auto-identificáveis: `'Esta chave pertence a alice@alice.com'`.
- Chaves privadas são geradas em conjunto pelo usuário e um *gerador de chaves privadas–GCP*.
  Perda de independência das entidades.
- Outras informações na chave públicas (datas, etc).

Criptografia baseada em identidades

- Alternativa ao uso de certificados digitais.
- Proposta como esquema de assinaturas em 1984 por Adi Shamir.
- Ganhou impulso em 2001 com a primeira proposta prática de esquema de encriptação baseado em identidades, de Boneh e Franklin.
- Chaves públicas auto-identificáveis: `'Esta chave pertence a alice@alice.com'`.
- Chaves privadas são geradas em conjunto pelo usuário e um *gerador de chaves privadas–GCP*.
  Perda de independência das entidades.
- Outras informações na chave públicas (datas, etc).
- Propostas alternativas: *Certificateless Public Key Cryptography*), (Al-Riyami e Paterson)

Identificação (autenticação de entidades)

Aspectos importantes:

Identificação (autenticação de entidades)

Aspectos importantes:

- **domínio de validade de uma identidade:** Alice deve ter conhecimento prévio de alguma evidência da identidade de Beto (números IP, URLs, email, *user logins*, etc).

Identificação (autenticação de entidades)

Aspectos importantes:

- **domínio de validade de uma identidade:** Alice deve ter conhecimento prévio de alguma evidência da identidade de Beto (números IP, URLs, email, *user logins*, etc).
- **natureza das informações de identificação enviadas por Beto:** algo que a entidade seja, algo que a entidade possua, algo que a entidade conheça.

Identificação (autenticação de entidades)

Aspectos importantes:

- **domínio de validade de uma identidade:** Alice deve ter conhecimento prévio de alguma evidência da identidade de Beto (números IP, URLs, email, *user logins*, etc).
- **natureza das informações de identificação enviadas por Beto:** algo que a entidade seja, algo que a entidade possua, algo que a entidade conheça.
- **segurança e robustez do protocolo:** ataques pela quebra dos algoritmos criptográficos ou falhas no projeto do protocolo.

Identificação (autenticação de entidades)

Aspectos importantes:

- **domínio de validade de uma identidade:** Alice deve ter conhecimento prévio de alguma evidência da identidade de Beto (números IP, URLs, email, *user logins*, etc).
- **natureza das informações de identificação enviadas por Beto:** algo que a entidade seja, algo que a entidade possua, algo que a entidade conheça.
- **segurança e robustez do protocolo:** ataques pela quebra dos algoritmos criptográficos ou falhas no projeto do protocolo.
- **identificação fraca vs. identificação forte (com desafio-resposta).**

Identificação (autenticação de entidades)

Aspectos importantes:

- **domínio de validade de uma identidade:** Alice deve ter conhecimento prévio de alguma evidência da identidade de Beto (números IP, URLs, email, *user logins*, etc).
- **natureza das informações de identificação enviadas por Beto:** algo que a entidade seja, algo que a entidade possua, algo que a entidade conheça.
- **segurança e robustez do protocolo:** ataques pela quebra dos algoritmos criptográficos ou falhas no projeto do protocolo.
- **identificação fraca vs. identificação forte (com desafio-resposta).**
- **reciprocidade da autenticação:** identificação unilateral vs. mútua.

Identificação fraca, unilateral, com senhas

Contexto inicial:
A já possui (B, h_B), onde $h_B = \mathrm{H}(\mathrm{senha}_B)$.
Resultado: A aceita ou não a identificação de B.

1.	B:	$\rightsquigarrow A$: B.
2.	A:	$\rightsquigarrow B$: "Digite a senha".
3.	B:	$\rightsquigarrow A$: senha_B.
4.	A:	localiza (B, h_B) no seu arquivo de senhas; $y \leftarrow \mathrm{H}(\mathrm{senha}_B)$; **se** $y = h_B$, **então** aceita B, **senão** rejeita.

Identificação fraca, unilateral, com senhas descartáveis

Contexto inicial:
A já possui (B, h_B), com $h_B = \mathrm{H}^n(\mathrm{senha}_B)$.
Resultado: A aceita ou não a identificação de B.

1.	B:	$\rightsquigarrow A$: B.
2.	A:	$\rightsquigarrow B$: "Digite a senha".
3.	B:	$y \leftarrow \mathrm{H}^{n-i}(\mathrm{senha}_B)$ na i-ésima sessão.
	B:	$\rightsquigarrow A$: y.
4.	A:	localiza (B, h_B) no arquivo de senhas;
		$y' \leftarrow \mathrm{H}(y)$;
		se $y' = h_B$, **então** aceita B e $(B, h_B) \leftarrow (B, y)$;
		senão rejeita.

Identificação forte, unilateral, com técnicas simétricas

Contexto inicial: A, B compartilham uma chave secreta k_{AB}.
Resultado: A aceita ou não a identificação de B.

1.	B:	$\rightsquigarrow A$: B.
2.	A:	`sorteia`(r); $\rightsquigarrow B$: (A, r).
3.	B:	$y \leftarrow \text{MAC}_{k_{AB}}(B\|\|r)$; $\rightsquigarrow A$: y.
4.	A:	$y' \leftarrow \text{MAC}_{k_{AB}}(B\|\|r)$; **se** $y = y'$, **então** aceita B, **senão** rejeita.

Identificação forte, mútua, com técnicas simétricas

Contexto inicial: A, B compartilham uma chave secreta k_{AB} .
Resultado: A e B aceitam ou não a identificação um do outro.

1.	B:	$\texttt{sorteia}(r_B)$; $\rightsquigarrow A$: (B, r_B).
2.	A:	$\texttt{sorteia}(r_A)$; $y_1 \leftarrow \text{MAC}_{k_{AB}}(A\|\|r_B\|\|r_A)$; $\rightsquigarrow B$: (r_A, y_1).
3.	B:	$y'_1 \leftarrow \text{MAC}_{k_{AB}}(A\|\|r_B\|\|r_A)$; **se** $y_1 = y'_1$, **então** $y_2 \leftarrow \text{MAC}_{k_{AB}}(B\|\|r_A)$, **senão** rejeita A e pára; $\rightsquigarrow A$: (y_2).
4.	A:	$y'_2 \leftarrow \text{MAC}_{k_{AB}}(B\|\|r_A)$; **se** $y_2 = y'_2$, **então** aceita B, **senão** rejeita.

Identificação forte, mútua, usando chaves públicas

Contexto inicial: A, B têm certificados CERT_A, CERT_B.
Resultado: A e B aceitam ou não a identificação um do outro.

1.	B:	`sorteia`(r_B); $\rightsquigarrow A$: (CERT_B, r_B).
2.	A:	`sorteia`(r_A); $s_A \leftarrow \text{SIGN}_A(B\|\|r_B\|\|r_A)$; $\rightsquigarrow B$: $(\text{CERT}_A, r_A, s_A)$.
3.	B:	Valida CERT_A; **se** $\text{VER}_A(s_A)$, **então** $s_B \leftarrow \text{SIGN}_B(A\|\|r_A)$, **senão** rejeita A e pára; $\rightsquigarrow A$: (A, s_B).
4.	A:	valida CERT_B **se** $\text{VER}_B(s_B)$ **então** aceita B, **senão** rejeita.

Identificação forte, mútua, com técnicas simétricas

Contexto inicial: A, B compartilham uma chave secreta k_{AB} .
Resultado: A e B aceitam ou não a identificação um do outro.

1.	B:	`sorteia`(r_B); $\rightsquigarrow A$: (B, r_B).
2.	A:	`sorteia`(r_A); $y_1 \leftarrow \text{MAC}_{k_{AB}}(A\|\|r_B\|\|r_A)$; $\rightsquigarrow B$: (r_A, y_1).
3.	B:	$y'_1 \leftarrow \text{MAC}_{k_{AB}}(A\|\|r_B\|\|r_A)$; **se** $y_1 = y'_1$, **então** $y_2 \leftarrow \text{MAC}_{k_{AB}}(B\|\|r_A)$, **senão** rejeita A e pára; $\rightsquigarrow A$: (y_2).
4.	A:	$y'_2 \leftarrow \text{MAC}_{k_{AB}}(B\|\|r_A)$; **se** $y_2 = y'_2$, **então** aceita B, **senão** rejeita.

Identificação forte, mútua, usando chaves públicas

Contexto inicial: A, B têm certificados CERT_A, CERT_B.
Resultado: A e B aceitam ou não a identificação um do outro.

1.	B:	`sorteia`(r_B); $\rightsquigarrow A$: (CERT_B, r_B).
2.	A:	`sorteia`(r_A); $s_A \leftarrow \text{SIGN}_A(B\|\|r_B\|\|r_A)$; $\rightsquigarrow B$: $(\text{CERT}_A, r_A, s_A)$.
3.	B:	Valida CERT_A; **se** $\text{VER}_A(s_A)$, **então** $s_B \leftarrow \text{SIGN}_B(A\|\|r_A)$, **senão** rejeita A e pára; $\rightsquigarrow A$: (A, s_B).
4.	A:	valida CERT_B **se** $\text{VER}_B(s_B)$ **então** aceita B, **senão** rejeita.

Estabelecimento de chaves

- Protocolos para pré-distribuição de chaves não-efêmeras, utilizadas para a distribuição de chaves de sessão.
- Protocolos para distribuição de chaves de sessão.
- Esquema para fracionamento de chaves críticas, de vida muito longa.

Pré-distribuição de chaves I (Diffie-Hellman)

Contexto inicial:

- Parâmetros públicos são: grupo $(G, .)$ e $\alpha \in G$ de ordem n.

Resultado: Chave de sessão k compartilhada por A e B.

1.	A:	`sorteia`(r_A), inteiro em $\{0 \ldots n-1\}$;
		$x_A \leftarrow \alpha^{r_A}$;
		$\rightsquigarrow B$: (A, x_A);
	B:	`sorteia`(r_B), inteiro em $\{0 \ldots n-1\}$;
		$x_B \leftarrow \alpha^{r_B}$;
		$\rightsquigarrow A$: (B, x_B);
2.	A:	$k = x_B^{r_A}$;
	B:	$k' = x_A^{r_B}$;

Pré-distribuição de chaves II (Diffie-Hellman com certificados)

Contexto inicial:

- Parâmetros públicos são: grupo $(G, .)$ e α gerador de G.
- A e B têm:
  chaves privadas d_A e d_B, $0 \leq d_A, d_B \leq |G| - 1$;
  chaves públicas $e_A = \alpha^{d_A}$ e $e_B = \alpha^{d_B}$.

Resultado: Chave k compartilhada por A e B.

1:	A:	obtém e valida CERT_B;
		$k \leftarrow {e_B}^{d_A}$;
	B:	obtém e valida CERT_A;
		$k' \leftarrow {e_A}^{d_B}$.

Distribuição de chaves de sessão com técnicas simétricas (Kerberos V5 simplificado)

Contexto inicial: A, B têm k_{AT}, k_{BT} pré-combinadas com a TPC.
Resultado: Chave de sessão k compartilhada por A e B.

1.	A:	`sorteia`(r_A); $\rightsquigarrow$TPC: (A, B, r_A);
2.	TPC:	`sorteia`(k); DefineLimiteTempo(l); $y_1 \leftarrow \text{ENC}_{k_{AT}}(r_A \| B \| k \| l)$; $tkt_B \leftarrow \text{ENC}_{k_{BT}}(k \| A \| l)$; $\rightsquigarrow A$: (y_1, tkt_B);
3.	A:	$x_1 \leftarrow \text{DEC}_{k_{AT}}(y_1)$; **se** x_1 não tem os campos esperados, **então** pára; $t \leftarrow$ HoraCorrente; $y_2 \leftarrow \text{ENC}_k(A \| t)$; $\rightsquigarrow B$: (y_2, tkt_B);

Distribuição de chaves de sessão com técnicas simétricas (Kerberos V5 simplificado)

4.	B:	$tkt'_B \leftarrow \text{DEC}_{k_{BT}}(tkt_B)$; **se** tkt'_B não tem os campos esperados, **então** pára; $x_2 \leftarrow \text{DEC}_k(y_2)$; **se** x_2 não tem os campos esperados ou $t > l$, **então** pára; $y_3 \leftarrow \text{ENC}_k(t+1)$; $\rightsquigarrow A$: y_3;
5.	A:	$x_3 \leftarrow \text{DEC}_k(y_3)$; **se** $x_3 \neq t+1$, **então** rejeita y_3, **senão** aceita.

Distribuição de chaves de sessão com técnicas simétricas (Bellare-Rogaway)

Contexto inicial: A, B têm k_{AT}, k_{BT} pré-combinadas com a TPC.
Resultado: Chave de sessão k compartilhada por A e B.

1.	A:	$\texttt{sorteia}(r_A)$;
		$\rightsquigarrow B$: (A, B, r_A);
2.	B:	$\texttt{sorteia}(r_B)$;
		$\rightsquigarrow$TPC: (A, B, r_A, r_B);
3.	TPC:	$\texttt{sorteia}(k)$;
		$y_A \leftarrow \text{ENC}_{k_{AT}}(k)$; $\quad y'_A \leftarrow \text{MAC}_{k_{AT}}(B\|A\|r_A\|y_A)$;
		$y_B \leftarrow \text{ENC}_{k_{BT}}(k)$; $\quad y'_B \leftarrow \text{MAC}_{k_{BT}}(A\|B\|r_B\|y_B)$;
		$\rightsquigarrow A$: (y_A, y'_A);
		$\rightsquigarrow B$: (y_B, y'_B);
4.	A:	**se** $y'_A = \text{MAC}_{k_{AT}}(B\|A\|r_A\|y_A)$, **então** aceita k, **senão** rejeita;
5.	B:	**se** $y'_B = \text{MAC}_{k_{BT}}(A\|B\|r_B\|y_B)$, **então** aceita k, **senão** rejeita.

Distribuição de chaves de sessão usando encriptação com chaves públicas

Contexto inicial:

- A tem chaves pública e privada e_A e d_A;
- B tem chaves pública e privada e_B e d_B;

Resultado: Chave de sessão k transmitida de A para B.

1.	A:	`sorteia`(k);
		obtém e valida CERT_B;
		$y \leftarrow \text{ENC}_{e_B}(k)$; $\quad s_A \leftarrow \text{SIGN}_A(y)$
		$\rightsquigarrow B$: (A, y, s_A).
2.	B:	obtém e valida CERT_A;
		se $\text{VER}_A(s_A)$ **então** $k \leftarrow \text{DEC}_{d_B}(y)$;
		senão rejeita.

Acordo de chaves usando técnicas simétricas (Encrypted-key Exchange)

Contexto inicial:

- Parâmetros públicos são: grupo $(G, .)$ e $\alpha \in G$ de ordem n.
- A e B compartilham chave secreta k_{AB}.

Resultado: Chave de sessão k compartilhada por A e B.

1.	A:	`sorteia`(r_A), inteiro em $\{0 \ldots n-1\}$;
		$x_A \leftarrow \alpha^{r_A}$; $\quad y_A \leftarrow \text{ENC}_{k_{AB}}(x_A)$;
		$\rightsquigarrow B$: (A, y_A);
2.	B:	`sorteia`(r_B), inteiro em $\{0 \ldots n-1\}$;
		$x_B \leftarrow \alpha^{r_B}$; $\quad y_B \leftarrow \text{ENC}_{k_{AB}}(x_B)$;
		$\rightsquigarrow A$: (B, y_B);
3.	A:	$x_B \leftarrow \text{DEC}_{k_{AB}}(y_B)$; $\quad k = (x_B)^{r_A}$;
4.	B:	$x_A \leftarrow \text{DEC}_{k_{AB}}(y_A)$; $\quad k' = (x_A)^{r_B}$;

Acordo de chaves usando assinaturas digitais (*Station-to-station*)

Contexto inicial:

- Parâmetros públicos são: grupo $(G, .)$; $\alpha \in G$ de ordem n.

Resultado: Chave de sessão k, compartilhada por A e B.

1:	A:	`sorteia`(r_A), inteiro em $\{0 \ldots n-1\}$;
		$x_A \leftarrow \alpha^{r_A}$;
		$\rightsquigarrow B$: (CERT_A, x_A).
2:	B:	Valida CERT_A;
		`sorteia`(r_B), inteiro em $\{0 \ldots n-1\}$;
		$x_B \leftarrow \alpha^{r_B}$; $\quad s_B \leftarrow \text{SIGN}_B(A \| x_B \| x_A)$;
		$k \leftarrow x_A^{r_B}$;
		$\rightsquigarrow A$: $(\text{CERT}_B, x_B, s_B)$.
3:	A:	Valida CERT_B;
		se $\text{VER}_B(s_B)$, **então** aceita, **senão** rejeita e pára;
		$s_A \leftarrow \text{SIGN}_A(B \| x_A \| x_B)$; $k \leftarrow x_B^{r_A}$;
		$\rightsquigarrow B$: s_A.
4:	B:	**se** $\text{VER}_A(s_A)$, **então** aceita, **senão** rejeita e pára.

Armazenamento de chaves críticas: método de Shamir

Armazenamento de chaves críticas: método de Shamir

- Chaves críticas devem ser armazenadas sem auxílio de outras chaves.

Armazenamento de chaves críticas: método de Shamir

- Chaves críticas devem ser armazenadas sem auxílio de outras chaves.
- Uso de esquemas de limiar é ideal:

  Permite fracionar a chave entre n usuários, de forma que $t, t \leq n$ frações sejam necessárias e suficientes para recuperar a chave.

Armazenamento de chaves críticas: método de Shamir

- Chaves críticas devem ser armazenadas sem auxílio de outras chaves.
- Uso de esquemas de limiar é ideal:

  Permite fracionar a chave entre n usuários, de forma que $t, t \leq n$ frações sejam necessárias e suficientes para recuperar a chave.

  Equilíbrio dos valores de t e n é essencial.

Armazenamento de chaves críticas: método de Shamir

- Chaves críticas devem ser armazenadas sem auxílio de outras chaves.
- Uso de esquemas de limiar é ideal:

  Permite fracionar a chave entre n usuários, de forma que $t, t \leq n$ frações sejam necessárias e suficientes para recuperar a chave.

  Equilíbrio dos valores de t e n é essencial.
- Esquema de Shamir foi o pioneiro.

Esquema de limiar de Shamir

- **Inicialização.** Escolha n inteiros, $x_1, x_2, \ldots, x_n \in \mathbb{Z}_p$, não nulos, $p \geq n + 1$. Distribua x_i para a entidade E_i.

Esquema de limiar de Shamir

- **Inicialização.** Escolha n inteiros, $x_1, x_2, \ldots, x_n \in \mathbb{Z}_p$, não nulos, $p \geq n+1$. Distribua x_i para a entidade E_i.
- **Fracionamento do segredo s.**
  Seja $s \in \mathbb{Z}_p$. Sorteie, em segredo, $a_1, a_2, \ldots, a_{t-1} \in \mathbb{Z}_p$, $t \leq n$, definindo assim o polinômio

$$a(x) = \sum_{j=1}^{t-1} a_j (x_i)^j.$$

Calcule $y_i \leftarrow s + \sum_{j=1}^{t-1} a_j (x_i)^j \bmod p, i = 1 \ldots n.$

Esquema de limiar de Shamir

- **Inicialização.** Escolha n inteiros, $x_1, x_2, \ldots, x_n \in \mathbb{Z}_p$, não nulos, $p \geq n+1$. Distribua x_i para a entidade E_i.
- **Fracionamento do segredo s.**
  Seja $s \in \mathbb{Z}_p$. Sorteie, em segredo, $a_1, a_2, \ldots, a_{t-1} \in \mathbb{Z}_p$, $t \leq n$, definindo assim o polinômio

$$a(x) = \sum_{j=1}^{t-1} a_j (x_i)^j.$$

  Calcule $y_i \leftarrow s + \sum_{j=1}^{t-1} a_j (x_i)^j \bmod p, i = 1 \ldots n$.
  Distribua y_i para a entidade E_i.

Esquema de limiar de Shamir

Observe que

- $a(0) = s$.

Esquema de limiar de Shamir

Observe que

- $a(0) = s$.
- Pelo Teorema Fundamental da Álgebra, qualquer subconjunto de até $t-1$ pontos (x_i, y_i) não revela informação alguma sobre os coeficientes $a_1, a_2, \ldots, a_{t-1}$ do polinômio $a(x)$, mas um conjunto com t ou mais pontos (x_i, y_i) determina exatamente um polinômio, a saber, $a(x)$ e, portanto, o valor $a(0) = s$.

Outros paradigmas

Outros paradigmas

Outros paradigmas

- Criptografia Quântica
- Emparelhamentos Bilineares

Criptografia Quântica

- Informação não é transformada mas enviada em claro por um canal onde não pode ser lida por um intruso de forma imperceptível. Isto é, em vez de esconder a informação, coíbe o acesso a ela.

Criptografia Quântica

- Informação não é transformada mas enviada em claro por um canal onde não pode ser lida por um intruso de forma imperceptível. Isto é, em vez de esconder a informação, coíbe o acesso a ela.
- Início da década de 1970, S. Wiesner lançou idéias seminais sobre o uso de estados conjugados de partículas elementares para codificar e transmitir informação.

Criptografia Quântica

- Informação não é transformada mas enviada em claro por um canal onde não pode ser lida por um intruso de forma imperceptível. Isto é, em vez de esconder a informação, coíbe o acesso a ela.
- Início da década de 1970, S. Wiesner lançou idéias seminais sobre o uso de estados conjugados de partículas elementares para codificar e transmitir informação.
- Idéias formaram a base do trabalho de C. Bennett e G. Brassard, o primeiro a descrever um protocolo completo para a distribuição de uma chave aleatória, sem comunicação prévia entre as partes.

Criptografia Quântica

- Informação não é transformada mas enviada em claro por um canal onde não pode ser lida por um intruso de forma imperceptível. Isto é, em vez de esconder a informação, coíbe o acesso a ela.
- Início da década de 1970, S. Wiesner lançou idéias seminais sobre o uso de estados conjugados de partículas elementares para codificar e transmitir informação.
- Idéias formaram a base do trabalho de C. Bennett e G. Brassard, o primeiro a descrever um protocolo completo para a distribuição de uma chave aleatória, sem comunicação prévia entre as partes.
- a informação trocada é, necessariamente, aleatória.

Criptografia Quântica

- Informação não é transformada mas enviada em claro por um canal onde não pode ser lida por um intruso de forma imperceptível. Isto é, em vez de esconder a informação, coíbe o acesso a ela.
- Início da década de 1970, S. Wiesner lançou idéias seminais sobre o uso de estados conjugados de partículas elementares para codificar e transmitir informação.
- Idéias formaram a base do trabalho de C. Bennett e G. Brassard, o primeiro a descrever um protocolo completo para a distribuição de uma chave aleatória, sem comunicação prévia entre as partes.
- a informação trocada é, necessariamente, aleatória.

Emparelhamentos Bilineares

Definição 37 *Sejam G_1 grupo aditivo, G_2 um grupo multiplicativo, ambos de ordem prima n. Seja α um gerador de G_1. Um* emparelhamento bilinear *é um mapeamento $\hat{e} : G_1 \times G_1 \rightarrow G_2$, tal que:*

Emparelhamentos Bilineares

Definição 38 *Sejam G_1 grupo aditivo, G_2 um grupo multiplicativo, ambos de ordem prima n. Seja α um gerador de G_1. Um* emparelhamento bilinear *é um mapeamento $\hat{e} : G_1 \times G_1 \rightarrow G_2$, tal que:*

1. *(bilinearidade) Para todos $\beta, \gamma, \delta \in G_1$,*
   $\hat{e}(\beta + \gamma, \delta) = \hat{e}(\beta, \delta)\hat{e}(\gamma, \delta)$ *e*
   $\hat{e}(\beta, \gamma + \delta) = \hat{e}(\beta, \gamma)\hat{e}(\beta, \delta)$;
2. *(não-degeneração)* $\hat{e}(\alpha, \alpha) \neq 1$;

Emparelhamentos Bilineares

Definição 39 *Sejam G_1 grupo aditivo, G_2 um grupo multiplicativo, ambos de ordem prima n. Seja α um gerador de G_1. Um* emparelhamento bilinear *é um mapeamento $\hat{e} : G_1 \times G_1 \rightarrow G_2$, tal que:*

1. *(bilinearidade) Para todos $\beta, \gamma, \delta \in G_1$, $\hat{e}(\beta + \gamma, \delta) = \hat{e}(\beta, \delta)\hat{e}(\gamma, \delta)$ e $\hat{e}(\beta, \gamma + \delta) = \hat{e}(\beta, \gamma)\hat{e}(\beta, \delta)$;*
2. *(não-degeneração) $\hat{e}(\alpha, \alpha) \neq 1$;*
3. *(computabilidade) $\hat{e}$ é eficientemente computável.*

Emparelhamentos Bilineares

Definição 40 *Sejam G_1 grupo aditivo, G_2 um grupo multiplicativo, ambos de ordem prima n. Seja α um gerador de G_1. Um* emparelhamento bilinear *é um mapeamento $\hat{e} : G_1 \times G_1 \rightarrow G_2$, tal que:*

1. *(bilinearidade) Para todos $\beta, \gamma, \delta \in G_1$,*
   $\hat{e}(\beta + \gamma, \delta) = \hat{e}(\beta, \delta)\hat{e}(\gamma, \delta)$ *e*
   $\hat{e}(\beta, \gamma + \delta) = \hat{e}(\beta, \gamma)\hat{e}(\beta, \delta)$;
2. *(não-degeneração)* $\hat{e}(\alpha, \alpha) \neq 1$;
3. *(computabilidade)* $\hat{e}$ *é eficientemente computável.*

Conseqüência muito util:

$$\hat{e}(a\beta, b\gamma) = \hat{e}(\beta, \gamma)^{ab}, \text{para todos } a, b, \text{inteiros.} \qquad (1)$$

Emparelhamentos bem conhecidos: Tate e Weil.

Emparelhamentos - exemplo de uso

- Alice, Beto e Carlos (C) querem estabelecer uma chave comum k.

Emparelhamentos - exemplo de uso

- Alice, Beto e Carlos (C) querem estabelecer uma chave comum k.
- Com Diffie-Hellman clássico, duas rodadas são necessárias:
  [1.] $A \rightsquigarrow B : \alpha^{r_A}$, $B \rightsquigarrow C : \alpha^{r_B}$ e $C \rightsquigarrow A : \alpha^{r_C}$.

Emparelhamentos - exemplo de uso

- Alice, Beto e Carlos (C) querem estabelecer uma chave comum k.
- Com Diffie-Hellman clássico, duas rodadas são necessárias:
  [1.] $A \rightsquigarrow B : \alpha^{r_A}$, $B \rightsquigarrow C : \alpha^{r_B}$ e $C \rightsquigarrow A : \alpha^{r_C}$.
  [2.] $A \rightsquigarrow B : \alpha^{r_C r_A}$, $B \rightsquigarrow C : \alpha^{r_A r_B}$ e $C \rightsquigarrow A : \alpha^{r_B r_C}$.

Emparelhamentos - exemplo de uso

- Alice, Beto e Carlos (C) querem estabelecer uma chave comum k.
- Com Diffie-Hellman clássico, duas rodadas são necessárias:
  [1.] $A \rightsquigarrow B : \alpha^{r_A}$, $B \rightsquigarrow C : \alpha^{r_B}$ e $C \rightsquigarrow A : \alpha^{r_C}$.
  [2.] $A \rightsquigarrow B : \alpha^{r_C r_A}$, $B \rightsquigarrow C : \alpha^{r_A r_B}$ e $C \rightsquigarrow A : \alpha^{r_B r_C}$.
- Após essas rodadas, os três calculam $k = \alpha^{r_A r_B r_C}$.

Emparelhamentos - exemplo de uso

- Alice, Beto e Carlos (C) querem estabelecer uma chave comum k.
- Com Diffie-Hellman clássico, duas rodadas são necessárias:
  [1.] $A \rightsquigarrow B : \alpha^{r_A}$, $B \rightsquigarrow C : \alpha^{r_B}$ e $C \rightsquigarrow A : \alpha^{r_C}$.
  [2.] $A \rightsquigarrow B : \alpha^{r_C r_A}$, $B \rightsquigarrow C : \alpha^{r_A r_B}$ e $C \rightsquigarrow A : \alpha^{r_B r_C}$.
- Após essas rodadas, os três calculam $k = \alpha^{r_A r_B r_C}$.
- É possível estabelecer a chave k com apenas uma rodada de mensagens?

Emparelhamentos - exemplo de uso

- Alice, Beto e Carlos (C) querem estabelecer uma chave comum k.
- Com Diffie-Hellman clássico, duas rodadas são necessárias:
  [1.] $A \rightsquigarrow B : \alpha^{r_A}$, $B \rightsquigarrow C : \alpha^{r_B}$ e $C \rightsquigarrow A : \alpha^{r_C}$.
  [2.] $A \rightsquigarrow B : \alpha^{r_C r_A}$, $B \rightsquigarrow C : \alpha^{r_A r_B}$ e $C \rightsquigarrow A : \alpha^{r_B r_C}$.
- Após essas rodadas, os três calculam $k = \alpha^{r_A r_B r_C}$.
- É possível estabelecer a chave k com apenas uma rodada de mensagens?
- Sim, mas com emparelhamentos bilineares,

Pré-distribuição de chaves tripartite (Joux)

Contexto inicial: A, B, C usam um emparelhamento $\hat{e}$.
Resultado: Chave de sessão k compartilhada por A, B e C.

1.	A:	`sorteia`(r_A), inteiro em $[0, n-1]$;
		$x_A \leftarrow r_A\alpha$;
		$\rightsquigarrow\{B, C\}$: (A, x_A);
	B:	`sorteia`(r_B), inteiro em $[0, n-1]$;
		$x_B \leftarrow r_B\alpha$;
		$\rightsquigarrow\{A, C\}$: (B, x_B);
	C:	`sorteia`(r_C), inteiro em $[0, n-1]$;
		$x_C \leftarrow r_C\alpha$;
		$\rightsquigarrow\{A, B\}$: (C, x_C);
2.	A:	$k \leftarrow \hat{e}(x_B, x_C)^{r_A}$;
	B:	$k \leftarrow \hat{e}(x_A, x_C)^{r_B}$;
	C:	$k \leftarrow \hat{e}(x_A, x_B)^{r_C}$.