Criptografia Aplicada

Criptografia assimétrica - Diffie-Hellman

Sumário

- Contexto histórico
- Definições básicas
- Algoritmo de troca de chaves DH
- Segurança
- DH na prática

Criptografia Simétrica

- **Eficiente**: a base de diversos algoritmos vistos consiste em operações simples, como ou-exclusivo, permutações e substituições
- **Segura**: algoritmos como o AES são resistentes a técnicas de criptoanálise conhecidas
- **Chaves pequenas**: qualquer coisa pode ser chave
- **Desafio**: como compartilhar a chave secreta para que emissor e receptor consigam cifrar e decifrar?

Criptografia de chave pública (assimétrica)

- Uma das maiores evoluções na história da criptografia
- Baseada em funções matemáticas ao invés de substituições e permutações
- São assimétricas: utilizam duas chaves diferentes, o que tem consequências profundas nas áreas de confidencialidade, distribuição de chaves, autenticação, etc.
- **Desafio**: as técnicas existentes ainda são menos eficientes do que a criptografia simétrica.
- **Combinação**: utilização de uma técnica assimétrica para o cálculo de chaves simétricas

Diffie-Hellman (1976)

- Criptografia de chave pública foi idealizada por Diffie e Hellman (1976) em Stanford.
- Publicaram o primeiro e mais simples algoritmo de chave pública.

Troca de chaves de Diffie–Hellman:

- **Objetivo**: permitir que dois usuários consigam trocar uma chave de maneira segura para depois utilizá-la na cifragem simétrica de mensagens.
- **Segurança**: O algoritmo depende da dificuldade de calcular logaritmos discretos

Imagem: https://tinyurl.com/diffie-hellman

"what good would it do after all to develop impenetrable cryptosystems, if their users were forced to share their keys with a KDC that could be compromised by either burglary or subpoena?" - Diffie

LabSEC
Laboratório de Segurança em Computação

cpqd

Sumário

- Contexto histórico
- **Definições básicas**
- Algoritmo de troca de chaves DH
- Segurança
- DH na prática

Relembrando

- Seja p um número primo, considere o grupo multiplicativo $\mathbb{Z}_p^* = \{1, 2, 3, \ldots, p-1\}$
- Uma raiz primitiva *a* de *p* é um número tal que a sequência:

  $$a \bmod p,\ a^2 \bmod p,\ \ldots,\ a^{p-1} \bmod p$$

  gera todos os números de $\mathbb{Z}_p^*$ em alguma ordem
- Para qualquer inteiro *b* e raiz primitiva *a* de um número primo *p*, podemos encontrar um único expoente x tal que $b = a^x \bmod p$.
  - x é o **logaritmo discreto** de *b*
  - geralmente, é muito difícil calcular/encontrar x

Sumário

- Contexto histórico
- Definições básicas
- **Algoritmo de troca de chaves DH**
- Segurança
- DH na prática

Aplicações

- Podemos classificar criptossistemas de chaves públicas em três categorias:
  - Cifragem/decifragem
  - Assinatura digital
  - Troca de chaves
- Alguns algoritmos servem para os três propósitos, outros apenas para um ou dois deles.

Table 9.3 Applications for Public-Key Cryptosystems

Algorithm	Encryption/Decryption	Digital Signature	Key Exchange
RSA	Yes	Yes	Yes
Elliptic Curve	Yes	Yes	Yes
Diffie–Hellman	No	No	Yes
DSS	No	Yes	No

Imagem: W. Stallings. *Cryptography and network security*. Cap 9.3

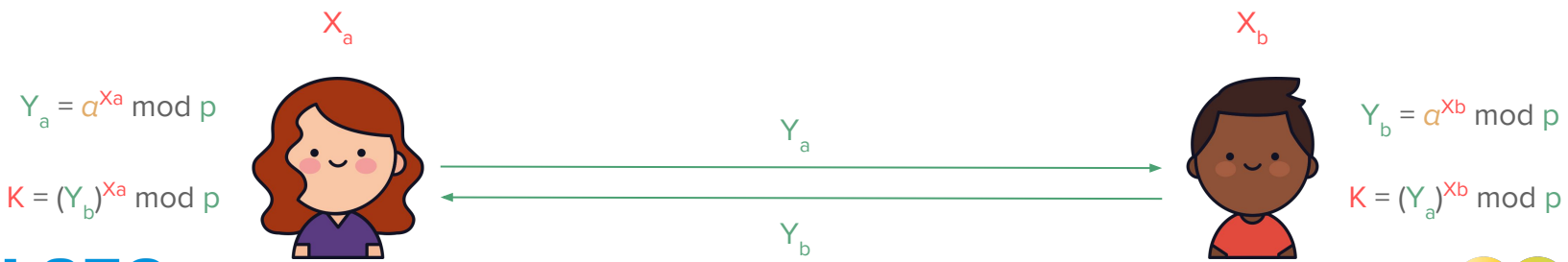
A troca de chaves Diffie-Hellman

Parâmetros

- **Parâmetros públicos:**
  - p número primo
  - α raíz primitiva de $\mathbb{Z}_p^*$
- **Parâmetros privados**:
  - X_a e X_b números aleatórios < p

Algoritmo:

- **Cálculo dos valores públicos:**
  - Alice calcula: $Y_a = \alpha^{X_a} \bmod p$
  - Bob calcula: $Y_b = \alpha^{X_b} \bmod p$
- **Cálculo do segredo:**
  - Alice calcula: $K = (Y_b)^{X_a} \bmod p$
  - Bob calcula: $K = (Y_a)^{X_b} \bmod p$



LabSEC
Laboratório de Segurança em Computação

cpqd

Exemplo

Parâmetros:

- p = 353, α = 3
- Alice calcula X_a = 97
- Bob calcula X_b = 233

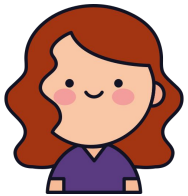
Explicação:

- Alice calcula: $Y_a = \alpha^{Xa} \bmod p$
- Bob calcula: $Y_b = \alpha^{Xb} \bmod p$
- Alice calcula: $(Y_b)^{Xa} \bmod p = (\alpha^{Xb})^{Xa} \bmod p$
- Bob calcula: $(Y_a)^{Xb} \bmod p = (\alpha^{Xa})^{Xb} \bmod p$

97

$Y_a = 3^{97} \bmod 353 = 40$

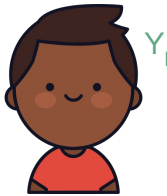
$K = (248)^{97} \bmod 353 = 160$



$Y_a = 40$

$Y_b = 248$

233



$Y_b = 3^{233} \bmod 353 = 248$

$K = (40)^{233} \bmod 353 = 160$

LabSEC
Laboratório de Segurança em Computação

cpqd

Exercício

- Vamos nos comunicar de maneira secreta!
- Assuma os valores públicos $p = 19$ e $\alpha = 2$
- **Pergunta 1:** Eles são valores válidos para iniciar o algoritmo DH?
  - 19 é primo
  - ordem de 2 (mod 19) é 18, ou seja, ele é o menor expoente tal que $2^{18} \equiv 1 \pmod{19}$
  - como $\phi(19) = 18$ = ordem de 2 módulo 19, 2 é uma *raiz primitiva módulo 19*
- A professora escolheu um valor secreto X_a e compartilhou com os alunos o seguinte valor público: $Y_a = \alpha^{X_a} \bmod p = 13$
- **Pergunta 2**: é possível descobrir o valor secreto X_a?
- **Pergunta 3**: qual o problema em descobrir o valor secreto de outro usuário?

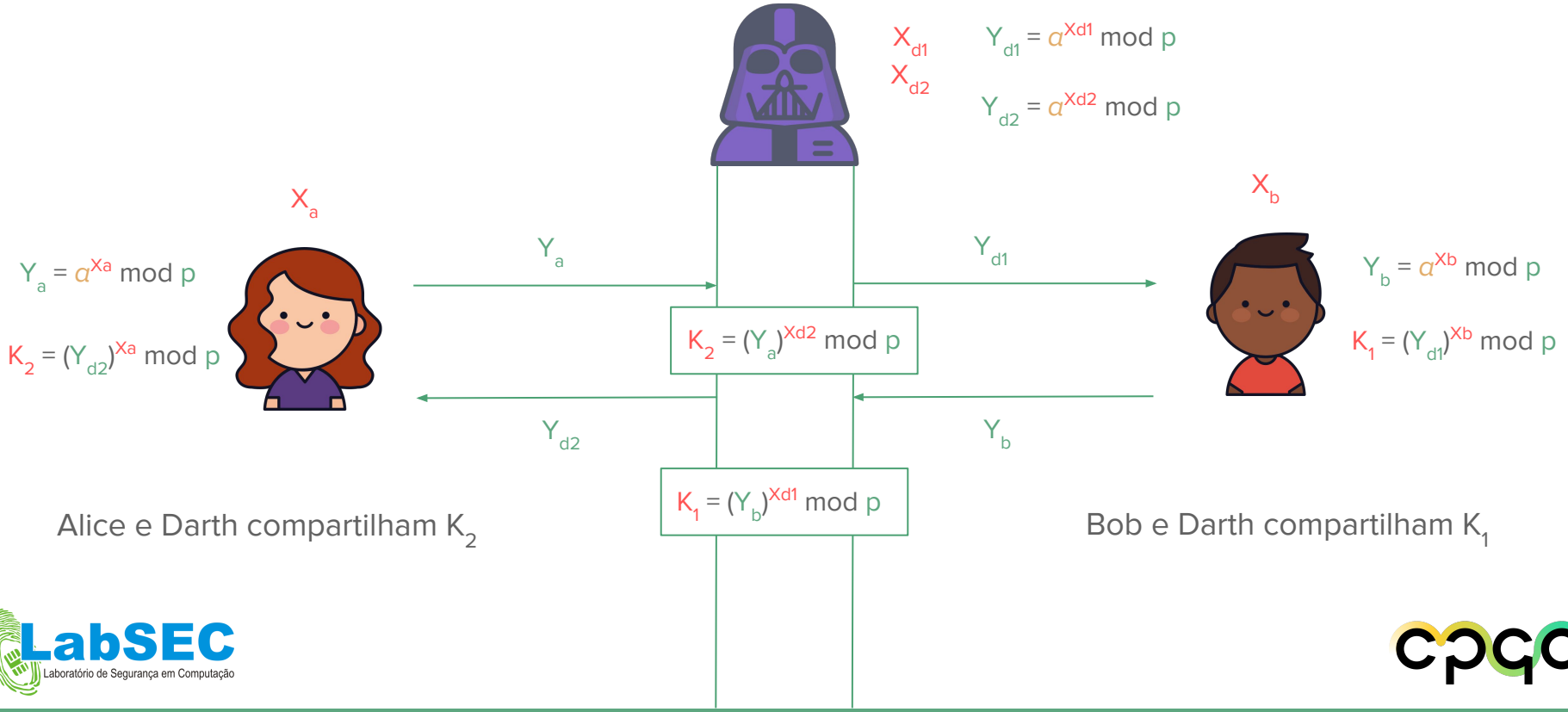
Sumário

- Contexto histórico
- Definições básicas
- Algoritmo de troca de chaves DH
- **Segurança**
- DH na prática

Segurança do algoritmo

- Apenas Alice e Bob conhecem os valores X_a e X_b, respectivamente
- é relativamente simples calcular exposições modulares
- é computacionalmente inviável resolver o **problema do logaritmo discreto** para valores grandes o suficiente
  - Ou seja, para $Y_a = \alpha^{Xa} \bmod p$ e $Y_b = \alpha^{Xb} \bmod p$
  - é inviável descobrir X_a ou X_b a partir de Y_a ou Y_b
- Ou seja, apenas Alice e Bob conseguem calcular K
  - portanto, esse valor pode ser utilizado entre eles como uma chave simétrica.
- O que significa "grande o suficiente"?
  - Atualmente, o primo *p* precisa ter, pelo menos 2048 bits

O ataque Man-in-the-Middle



LabSEC
Laboratório de Segurança em Computação

cpqd

O ataque Man-in-the-Middle

- O protocolo só é vulnerável a esse ataque pois não há autenticação dos participantes
- Essa vulnerabilidade pode ser resolvida utilizando assinaturas digitais e certificados digitais
    - a ser explorado nas próximas aulas

Sumário

- Contexto histórico
- Definições básicas
- Algoritmo de troca de chaves DH
- Segurança
- **DH na prática**

Aplicações do DH

- TLS/SSL: protocolo utilizado na proteção de tráfego web (HTTPS), frequentemente utiliza DH para estabelecer conexões seguras. Importante na proteção de dados sensíveis trafegados pela internet
- SSH: provê acesso remoto seguro à sistemas computacionais. Utiliza DH para criar uma chave entre cliente e servidor para a transmissão de dados cifrados.
- VPNs: utilizadas para estabelecer um canal de comunicação seguro na internet. Utiliza DH para estabelecer um túnel seguro entre cliente e o servidor VPN
- entre outros.

Vulnerabilidades na prática

Logjam (2015): permitiu ataques man-in-the-middle que forçam o servidor a escolher parâmetros inseguros de 512 bits. Com isso, atacantes conseguem ler e modificar qualquer dado sendo transmitido naquela conexão.

Atividade: gerando chaves DH com openssl

- Gere os parâmetros públicos do DH e armazene em um arquivo:

```
openssl dhparam -out dhparams.pem 2048
```

- Visualize os parâmetros:

```
openssl pkeyparam -in dhparams.pem -text -noout
```

- Gere os valores públicos e privados da Alice utilizando o dhparams.pem

```
openssl genpkey -paramfile dhparams.pem -out alice_keys.pem
```

- Visualize os valores da Alice:

```
openssl pkey -in alice_keys.pem -text -noout
```

Atividade: gerando chaves DH com openssl

- Gere as os valores públicos e privados do Bob utilizando o dhparams.pem

```
openssl genpkey -paramfile dhparams.pem -out bob_keys.pem
```

- Extraia e visualize a chave pública da Alice para ser enviada ao Bob:

```
openssl pkey -in alice_keys.pem -pubout -out alice_pub.pem
openssl pkey -pubin -in alice_pub.pem -text
```

- Extraia e visualize a chave pública do Bob para ser enviada a Alice:

```
openssl pkey -in bob_keys.pem -pubout -out bob_pub.pem
openssl pkey -pubin -in bob_pub.pem -text
```

Atividade: gerando chaves DH com openssl

- Gere a chave secreta da Alice usando a sua chave privada e a chave pública do Bob

```
openssl pkeyutl -derive -inkey  alice_keys.pem -peerkey bob_pub.pem -out alice_secret.bin
```

- Gere a chave secreta do Bob usando a sua chave privada e a chave pública da Alice

```
openssl pkeyutl -derive -inkey  bob_keys.pem -peerkey alice_pub.pem -out bob_secret.bin
```

- As duas chaves geradas devem ser iguais

```
cmp alice_secret.bin bob_secret.bin
```

Resumo

- A criptografia de chave pública/assimétrica foi uma das maiores evoluções na história da criptografia
- O primeiro algoritmo de chave pública proposto foi o Diffie-Hellman
- Sua segurança é baseada no problema do logaritmo discreto
- Ele é muito utilizado para fazer um acordo de chaves simétricas entre partes que desejam se comunicar utilizando criptografia simétrica
    - garantindo assim confidencialidade na comunicação

Referências

- W. Stallings. *Cryptography and network security*. 7a edição.
  - Princípios de criptossistemas de chave pública: 9.1
  - Diffie-Hellman: 10.1
- D. Stinson e M. Paterson. *Cryptography: Theory and Practice*. 4a edição.
  - Introdução a criptossistemas de chave pública: 6.1
- imagem: Flaticon.com
- Recomendações de tamanho de chaves:
  - NIST SP 800-56A
  - NIST SP 800-131A
- DH usando openssl: https://sandilands.info/sgordon/diffie-hellman-secret-key-exchange-with-openssl