# Chapters 10

# Other Public-Key Cryptosystems

Dr. Shin-Ming Cheng

NTUST

CONNECTIVITY LAB

CS4003701

# Discrete Logarithm Problem

› Diffie–Hellman Key Exchange

› The Discrete Logarithm Problem

› Security of the Diffie–Hellman Key Exchange

› The Elgamal Encryption Scheme

# Diffie–Hellman Key Exchange: Overview

- Proposed in 1976 by Whitfield Diffie and Martin Hellman
  - Invented public-key cryptography
  - W. Diffie and M. E. Hellman, “New Directions in Cryptography”, *IEEE Transactions on Information Theory,* vol. IT-22, Nov. 1976, pp. 644-654
  - Widely used, in Secure Shell (SSH), Transport Layer Security (TLS), and Internet Protocol Security (IPSec)
- The Diffie–Hellman Key Exchange (DHKE) is a key exchange protocol and not used for encryption
  - For the purpose of encryption based on the DHKE, ElGamal can be used.

# Diffie–Hellman Key Exchange: Set-up

- Choose a large prime $p$
- Choose an integer $\alpha \in \{2,3,\ldots,p-2\}$
- Publish $p$ and $\alpha$

- Discrete Logarithm/離散對數 Problem
  - Given $p$, $\alpha$, and $A(=\alpha^a)$, how to compute $a$
  - It's hard when $p$ is large
- Diffie-Hellman Problem
  - Given $p$, $\alpha$, $A(=\alpha^a)$, and $B(=\alpha^b)$, how to compute $\alpha^{ab}$

# Diffie–Hellman Key Exchange

| Alice | | Bob |
|---|---|---|
| Choose random private key $k_{prA} = a \in \{1, 2, \ldots, p-1\}$ | | Choose random private key $k_{prB} = b \in \{1, 2, \ldots, p-1\}$ |
| Compute corresponding public key $k_{pubA} = A = \alpha^a \bmod p$ | $\xrightarrow{A}$ | |
| | $\xleftarrow{B}$ | Compute correspondig public key $k_{pubB} = B = \alpha^b \bmod p$ |
| Compute common secret $k_{AB} = B^a = (\alpha^a)^b \bmod p$ | | Compute common secret $k_{AB} = A^b = (\alpha^b)^a \bmod p$ |

---

We can now use the joint key $k_{AB}$ for encryption, e.g., with AES

$$y = AES_{k_{AB}}(x) \xrightarrow{\quad y \quad} x = AES^{-1}_{k_{AB}}(y)$$

# Diffie–Hellman Key Exchange: Example

Domain parameters $p$ = 29, $\alpha$ = 2

**Alice**

Choose random private key
$k_{prA} = a = 5$

Compute corresponding public key
$k_{pubA} = A = 2^5 = 3 \bmod 29$

$\xrightarrow{A}$

Compute common secret
$k_{AB} = B^a = 7^5 = 16 \bmod 29$

**Bob**

Choose random private key
$k_{prB} = b = 12$

Compute correspondig public key
$k_{pubB} = B = 2^{12} = 7 \bmod 29$

$\xleftarrow{B}$

Compute common secret
$k_{AB} = A^b = 3^{12} = 16 \bmod 29$

Proof of correctness:

Alice computes: $B^a = (\alpha^b)^a \bmod p$
Bob computes: $A^b = (\alpha^a)^b \bmod p$

i.e., Alice and Bob compute the same key $k_{AB}$!

# The Discrete Logarithm Problem

## Discrete Logarithm Problem (DLP) in $Z_p^*$

- Given is the finite cyclic group $Z_p^*$ of order $p-1$ and a primitive element $\alpha \in Z_p^*$ and another element $\beta \in Z_p^*$
- The DLP is the problem of determining the integer $1 \leq x \leq p-1$ such that $\alpha^x \equiv \beta \bmod p$
- This computation is called the discrete logarithm problem (DLP)

$$x = \log_\alpha \beta \bmod p$$

- Example: Compute $x$ for $5^x \equiv 41 \bmod 47$

# The Generalized Discrete Logarithm Problem

- Given a finite cyclic group $G$ with the group operation $\circ$ and cardinality $n$
- We consider a primitive element $\alpha \in G$ and another element $\beta \in G$
- The discrete logarithm problem is finding the integer $x$, where $1 \leq x \leq n$, such that:

$$\beta = \underbrace{\alpha \circ \alpha \circ \alpha \circ \ \dots \circ \alpha}_{x \text{ times}} = \alpha^x$$

# The Generalized Discrete Logarithm Problem

- The difficulty of this problem depends on the group $G$
  - Very easy: polynomial time algorithm
    - $(Z_{\mathrm{N}}, +)$
  - Rather hard: sub-exponential time algorithm
    - $(Z_P, \times)$
  - Very hard: exponential time algorithm
    - Elliptic Curve groups
- The multiplicative group of a Galois field $GF(2^m)$ or a subgroup of it. Schemes such as the DHKE can be realized with them
- Hyperelliptic curves or algebraic varieties, which can be viewed as generalization of elliptic curves

# Attacks against the Discrete Logarithm Problem

- The following algorithms for computing discrete logarithms exist
  - Generic algorithms: Work in any cyclic group
    - Brute-Force Search
    - Shanks' Baby-Step-Giant-Step Method
    - Pollard's Rho Method
    - Pohlig-Hellman Method
  - Non-generic Algorithms: Work only in specific groups, in particular in $Z_p$
    - The Index Calculus Method

# Attacks against the Discrete Logarithm Problem

- Summary of records for computing discrete logarithms in $Z_p^*$

| Decimal digits | Bit length | Date |
|---|---|---|
| 58 | 193 | 1991 |
| 68 | 216 | 1996 |
| 85 | 282 | 1998 |
| 100 | 332 | 1999 |
| 120 | 399 | 2001 |
| 135 | 448 | 2006 |
| 160 | 532 | 2007 |

  - In order to prevent attacks that compute the DLP, it is recommended to use primes with a length of at least $1024$ bits for schemes such as Diffie-Hellman in $Z_p^*$

# Security of the classical Diffie–Hellman Key Exchange

- Which information does Oscar have?
  - $\alpha, p$
  - $k_{pubA} = A = \alpha^a \bmod p$
  - $k_{pubB} = B = \alpha^b \bmod p$
- Which information does Oscar want to have?
  - $k_{AB} = \alpha^{ba} = \alpha^{ab} \bmod p$
  - Diffie-Hellman Problem (DHP)
- The only known way to solve the DHP is to solve the DLP, i.e.
  - Compute $a = \log_\alpha A \bmod p$
  - Compute $k_{AB} = B^a = \alpha^{ba} \bmod p$
    - It is conjectured that the DHP and the DLP are equivalent, i.e., solving the DHP implies solving the DLP.
- To prevent that the DLP is solved, choose $p > 2^{1024}$

# The ElGamal Encryption Scheme: Overview

- Proposed by Taher ElGamal in 1985
  - A public key cryptosystem and a signature scheme based on discrete logarithms
- Can be viewed as an extension of the DHKE protocol
- Based on the intractability of the discrete logarithm problem and the Diffie–Hellman problem

# The ElGamal Encryption Scheme: Principle

| Alice | | Bob |
|---|---|---|
| | | choose $d = k_{prB} \in \{2, \ldots, p-2\}$ |
| | $\xleftarrow{\beta}$ | compute $\beta = k_{pubB} = \alpha^d \bmod p$ |
| choose $i = k_{prA} \in \{2, \ldots, p-2\}$ | | |
| compute ephemeral key $k_E = k_{pubA} = \alpha^i \bmod p$ | $\xrightarrow{k_E}$ | |
| | | compute $k_M = k_E^{\,d} \bmod p$ |
| compute $k_M = \beta^i \bmod p$ | | |
| encrypt message $x \in Z_p^*$: $y = x \cdot k_M \bmod p$ | $\xrightarrow{y}$ | |
| | | decrypt $x = y \cdot k_M^{-1} \bmod p$ |

This looks very similar to the DHKE! The actual Elgamal protocol re-orders the computations which helps to save one communication (cf. next slide)

# The ElGamal Encryption Protocol

| Alice | | Bob |
|---|---|---|
| | | choose large prime $p$ |
| | | choose primitive element $\alpha \in Z_p^*$ or in a subgroup of $Z_p^*$ |
| | | choose $d = k_{prB} \in \{2, \ldots, p-2\}$ |
| | $\xleftarrow{k_{pubB} = (p, \alpha, \beta)}$ | compute $\beta = k_{pubB} = \alpha^d \bmod p$ |
| choose $i = k_{prA} \in \{2, \ldots, p-2\}$ | | |
| compute $k_E = k_{pubA} = \alpha^i \bmod p$ | | |
| compute masking key $k_M = \beta^i \bmod p$ | | |
| encrypt message $x \in Z_p^*$: $y = x \cdot k_M \bmod p$ | $\xrightarrow{(k_E, y)}$ | compute masking key $k_M = k_E^d \bmod p$ |
| | | decrypt $x = y \cdot k_M^{-1} \bmod p$ |

# Computational Aspects

- Key Generation
  - Generation of prime $p$
    - $p$ has to of size of at least $1024$ bits
- Encryption
  - two modular exponentiations and a modular multiplictation
  - All operands have a bitlength of $\log_2 p$
  - Efficient execution requires methods such as the square-and-multiply algorithm
- Decryption
  - one modular exponentiation and one modulare inversion
  - the inversion can be computed from the ephemeral key
    - $k_{\mathrm{M}}^{-1} = k_{\mathrm{E}}^{-d} = \alpha^{-di} = k_{\mathrm{E}}^{p-1-d} \bmod p$
      - $p-1-d>0, -di<0$ requires inverse

# Security

- Passive attacks
  - Attacker eavesdrops $p, \alpha, \beta = \alpha^d, k_E = \alpha^i, y = x \cdot \beta^i$ and wants to recover $x$
  - Problem relies on the DLP
- Active attacks
  - If the public keys are not authentic, an attacker could send an incorrect public key
  - An attack is also possible if the secret exponent $i$ is being used more than once

# Man-in-the-Middle Attack

- Should be careful whom you are agreeing a key with
  - Alice agrees a key with Eve, thinking it is Bob
  - Bob agrees a key with Eve, thinking it is Alice
  - Eve can now examine communications as they pass through (acting as a router)

| Alice | | Eve | | Bob |
|---|---|---|---|---|
| $a$ | $\rightarrow$ | $g^a$ | | |
| $g^m$ | $\leftarrow$ | $m$ | | |
| $g^{am}$ | | $g^{am}$ | | |
| | | $n$ | $\rightarrow$ | $g^n$ |
| | | $g^b$ | $\leftarrow$ | $b$ |
| | | $g^{bn}$ | | $g^{bn}$ |

# Lessons Learned

- The Diffie–Hellman protocol is a widely used method for key exchange. It is based on cyclic groups
- The discrete logarithm problem is one of the most important one-way functions in modern asymmetric cryptography. Many public-key algorithms are based on it
- For the Diffie–Hellman protocol in $Z_p^*$, the prime p should be at least $1024$ bits long. This provides a security roughly equivalent to an $80$-bit symmetric cipher
- For a better long-term security, a prime of length $2048$ bits should be chosen
- The Elgamal scheme is an extension of the DHKE where the derived session key is used as a multiplicative masked to encrypt a message
- Elgamal is a probabilistic encryption scheme, i.e., encrypting two identical messages does not yield two identical ciphertexts

# (NIST) SP 800-57 Part1

| Bits of security | Symmetric key algorithm | Finite Field Cryptography (FFC e.g., DSA, D-H) | Integer Factorization Cryptography (IFC, e.g., RSA) | Elliptic Curve Cryptogrpahy (ECC, e.g., ECDSA) |
|---|---|---|---|---|
| 80 | 2TDEA | $L$=1024, $N$=160 | $k$=1024 | $f$=160-223 |
| 112 | 3TDEA | $L$=2048, $N$=224 | $k$=2048 | $f$=224-255 |
| 128 | AES-128 | $L$=3072, $N$=256 | $k$=3072 | $f$=256-383 |
| 192 | AES-192 | $L$=7680, $N$=384 | $k$=7680 | $f$=384-511 |
| 256 | AES-256 | $L$=15360, $N$=512 | $k$=15360 | $f$=512+ |