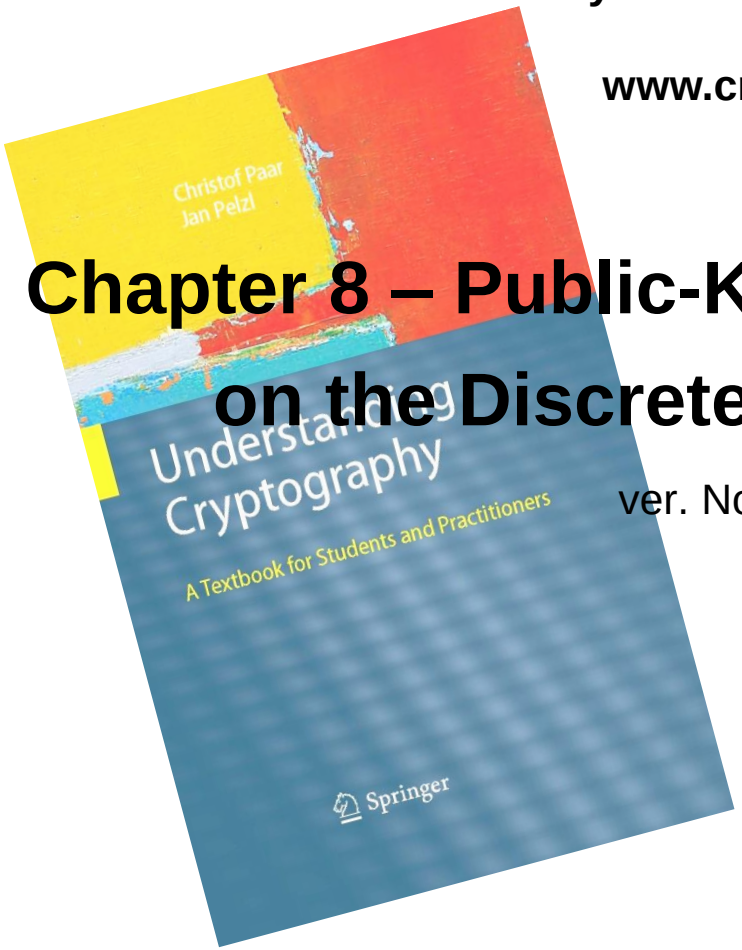
Understanding Cryptography

by Christof Paar and Jan Pelzl

www.crypto-textbook.com



Chapter 8 – Public-Key Cryptosystems Based on the Discrete Logarithm Problem

ver. November 3, 2024

These slides were originally prepared by Christof Paar and Jan Pelzl. Later, they were modified by Tomas Fabsic for purposes of teaching I-ZKRY at FEI STU.

Homework till 4.11.

- Read Sections 7.7. – 7.12. from Chapter 7.
- Read Section 8.2 together with „Pomocný materiál o grupách“.
- Solve problems from the exercise set no. 6 and submit them to AIS by **4.11.2024 23:59**.
- You do not need to read the rest of this presentation. We will cover the material in the presentation next week.

Homework till 11.11.

- Read Chapter 8 (you can skip Section 8.3.3).
- Solve problems from the exercise set no. 7 and submit them to AIS by **11.11.2024 23:59**.

Some legal stuff (sorry): Terms of Use

- The slides can used free of charge. All copyrights for the slides remain with Christof Paar and Jan Pelzl.
- The title of the accompanying book “Understanding Cryptography” by Springer and the author’s names must remain on each slide.
- If the slides are modified, appropriate credits to the book authors and the book title must remain within the slides.
- It is not permitted to reproduce parts or all of the slides in printed form whatsoever without written consent by the authors.

Content of this Chapter

- The Discrete Logarithm Problem
- Diffie–Hellman Key Exchange
- The Elgamal Encryption Scheme

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 $5^x \equiv 41 \bmod 47$

Remark: For the coverage of groups and cyclic groups, we refer to Chapter 8 of *Understanding Cryptography*

The Generalized Discrete Logarithm Problem

- Given is 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:

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

The Generalized Discrete Logarithm Problem

The following discrete logarithm problems have been proposed for use in cryptography

1. The multiplicative group of the prime field Z_p *or a subgroup of it. For instance,* the classical DHKE uses this group (cf. previous slides), but also Elgamal encryption or the Digital Signature Algorithm (DSA).
2. The cyclic group formed by an elliptic curve (see Chapter 9)
3. The multiplicative group of a Galois field $GF(2^m)$ or a subgroup of it. *S*chemes such as the DHKE can be realized with them.
4. Hyperelliptic curves or algebraic varieties, which can be viewed as generalization of elliptic curves.

Remark: The groups 1. and 2. are most often used in practice.

Attacks against the Discrete Logarithm Problem

- Security of many asymmetric primitives is based on the difficulty of computing the DLP in cyclic groups, i.e.,

  Compute x for a given α and β such that $\beta = \alpha \circ \alpha \circ \alpha \circ \ldots \circ \alpha = \alpha^x$

- 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
- Remark: Elliptic curves can only be attacked with generic algorithms which are weaker than non-generic algorithms. Hence, elliptic curves are secure with shorter key lengths than the DLP in prime fields Z_p

Attacks against the Discrete Logarithm Problem

- In order to prevent attacks that compute the DLP, it is recommended to use primes with a length of at least 2048 bits for schemes such as Diffie-Hellman in Z_p^*

Diffie–Hellman Key Exchange: Overview

- Proposed in 1976 by **Whitfield Diffie and Martin Hellman**
- **Widely used** (e.g. in TLS)
- 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

1. Choose a large prime *p.*
2. Choose a generator $\alpha \in Z_p^*$.
3. Publish *p and* α.

Diffie–Hellman Key Exchange

Alice		Bob
Choose random private key $k_{prA}=a \in \{2,\ldots,p-2\}$		Choose random private key $k_{prB}=b \in \{2,\ldots,p-2\}$
Compute corresponding public key $k_{pubA} = A = \alpha^a \bmod p$	A →	
	← 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

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

Diffie–Hellman Key Exchange: Example

Domain parameters p=29, α=2

Alice		Bob
Choose random private key $k_{prA} = a = 5$		Choose random private key $k_{prB} = b = 12$
Compute corresponding public key $k_{pubA} = A = 2^5 = 3 \bmod 29$	A →	
	← B	Compute correspondig public key $k_{pubB} = B = 2^{12} = 7 \bmod 29$
Compute common secret $k_{AB} = B^a = 7^5 = 16 \bmod 29$		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} !

Ephemeral keys in DHKE

- *(a,A)* and *(b,B)* are ephemeral key pairs (*dočasné (efemérne) páry kľúčov*)
- When Alice and Bob run DHKE again, they will generate new values of *(a,A)* and *(b,B).*
- This is different than in RSA where *(d,(n,e))* is a long-term key-pair.

Security of the classical Diffie–Hellman Key Exchange

- Which information does Oscar have?
  - α, p
  - $A = \alpha^a \bmod p$
  - $B = \alpha^b \bmod p$
- Which information does Oscar want to have?
  - $k_{AB} = \alpha^{ba} = \alpha^{ab} \bmod p$
  - This is known as **Diffie-Hellman Problem (DHP)**

Diffie-Hellman Problem

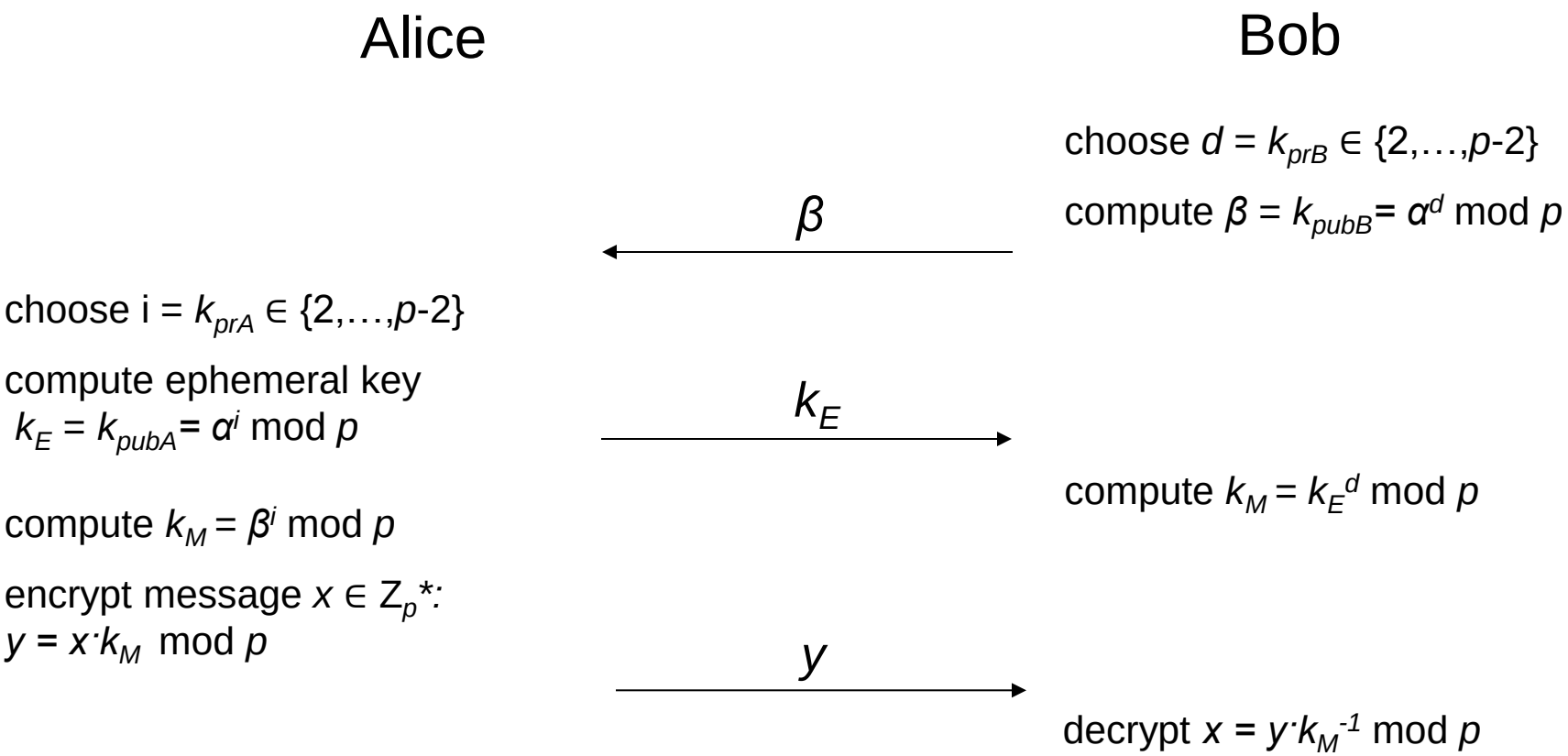
Definition 8.4.1 Generalized Diffie–Hellman Problem (DHP)
Given is a finite cyclic group G of order n, a primitive element $\alpha \in G$ and two elements $A = \alpha^a$ and $B = \alpha^b$ in G. The Diffie–Hellman problem is to find the group element α^{ab}.

- The only known way to solve the DHP is to solve the DLP, i.e.
  1. Compute $a = log_\alpha A$
  2. *Compute* $k_{AB} = B^a = \alpha^{ba}$
- It is, however, possible that there exists another method for solving the DHP without computing the discrete logarithm.
- It is conjectured that there is no easier method to solve the DHP than solving the DLP (i.e. DHP and DLP are conjectured to be equivalently hard).

The Elgamal Encryption Scheme: Overview

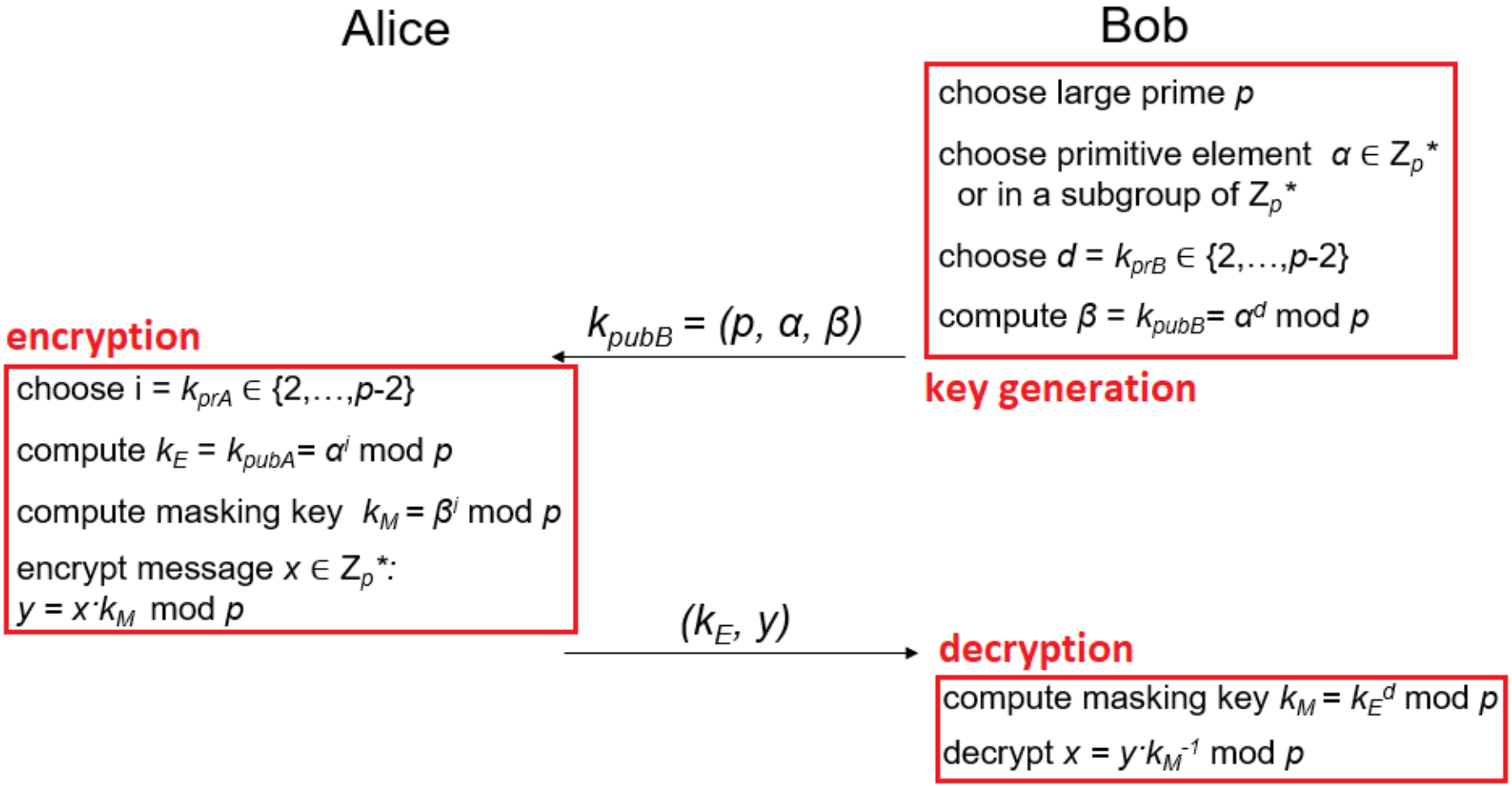
- Proposed by Taher Elgamal in 1985
- Can be viewed as an extension of the DHKE protocol
- Based on the intractability of the discrete logarithm problem and the Diffie–Hellman problem

The Elgamal Encryption Scheme: Principle



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



Long-term and ephemeral keys in Elgamal

- *(d,β)* is a long-tem key pair (like *(d,(n,e))* in RSA)
- Values *i*, k_E and k_M are ephemeral.
  - For every new plaintext *x*, new *i* has to be generated and new k_E and k_M have to be computed.

Computational Aspects

- Key Generation
  - Generation of prime p of size of at least 2048 bits
  - cf. Section 7.6 in *Understanding Cryptography* for prime-finding algorithms
- Encryption
  - Encryption is probabilistic! (unlike in schoolbook RSA)
  - The ciphertext has twice as many bits as plaintext (in RSA bit-lengths of plaintext and ciphertext are the same)
  - Requires two modular exponentiations and a modular multiplication
  - All operands have a bitlength of $\log_2 p$
  - Efficient execution requires methods such as the square-and-multiply algorithm
- Decryption
  - Requires one modular exponentiation and one modular inversion
  - As shown *in Understanding Cryptography*, the inversion can be computed from the ephemeral key

Security

- Passive attacks
  - Attacker eavesdrops p, α, $\beta = \alpha^d$, $k_E = \alpha^i$, $y = x \cdot \beta^i$ and wants to recover x
  - Problem relies on the DHP

- Active attacks
  - If the public keys are not authentic, an attacker could send an incorrect public key (cf. Chapter 14)
  - An Attack is also possible if the secret exponent i is being used more than once (cf. *Understanding Cryptography* for more details on the attack)
  - Like RSA, schoolbook Elgamal is malleable. In practice, it is therefore used with padding.

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 2048 bits* long.
- 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.