

CP 460 - Applied Cryptography

Introduction to Public-Key Cryptography

**Department of Physics and Computer Science
Faculty of Science, Waterloo**

Abbas Yazdinejad, Ph.D.

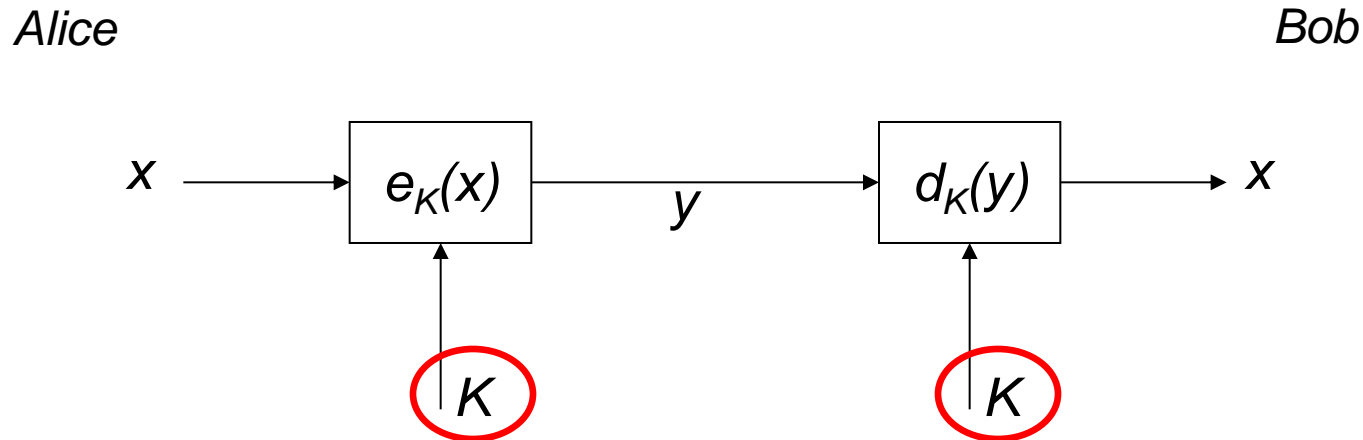
Fall 2024

Content of this Chapter

- Symmetric Cryptography Revisited
- Principles of Asymmetric Cryptography
- Practical Aspects of Public-Key Cryptography
- Important Public-Key Algorithms
- Essential Number Theory for Public-Key Algorithms

■ Symmetric Cryptography revisited

Symmetric cryptography uses the same key for both encryption (to encode data) and decryption (to decode data). This method is straightforward and relies on a single, shared secret key that both parties (e.g., Alice and Bob) must keep private.



Two properties of symmetric (secret-key) crypto-systems:

- The **same secret key K** is used for encryption and decryption
- Encryption and Decryption are very similar (or even identical) functions

Symmetric algorithms, such as AES and 3DES, are computationally efficient and fast, making them suitable for encrypting large volumes of data.

■ Symmetric Cryptography: Analogy



Safe with a strong lock, only Alice and Bob have a copy of the key

- Alice encrypts → locks message in the safe with her key
- Bob decrypts → uses his copy of the key to open the safe

■ Symmetric Cryptography: Shortcomings

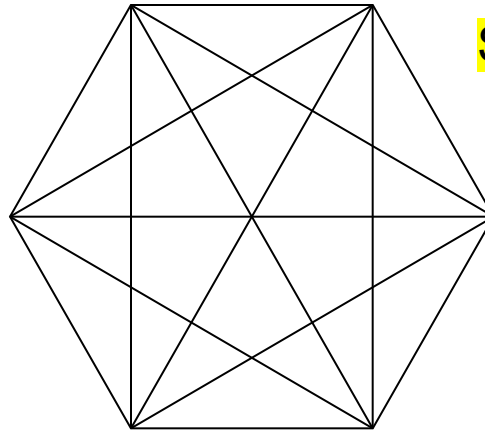
- Symmetric algorithms, e.g., AES or 3DES, are very secure, fast & widespread **but**:
- **Key distribution problem**: The secret key must be **transported securely**
- Number of keys: In a network, each pair of users requires an individual key

→ n users in the network require $\frac{n \cdot (n - 1)}{2}$ keys, each user stores $(n-1)$ keys

Example:

6 users (nodes)

$$\frac{6 \cdot 5}{2} = 15 \text{ keys (edges)}$$



Scalability in Networks

- Alice or Bob can **cheat each other**, because they have identical keys.

Example: Alice can claim that she never ordered a TV on-line from Bob (he could have fabricated her order). To prevent this: „non-repudiation“

Content of this Chapter

- Symmetric Cryptography Revisited
- **Principles of Asymmetric Cryptography**
- Practical Aspects of Public-Key Cryptography
- Important Public-Key Algorithms
- Essential Number Theory for Public-Key Algorithms

■ Idea behind Asymmetric Cryptography



New Idea:

Use the „good old mailbox“ principle:

Everyone can drop a letter

But: Only the owner has the correct key to open the box

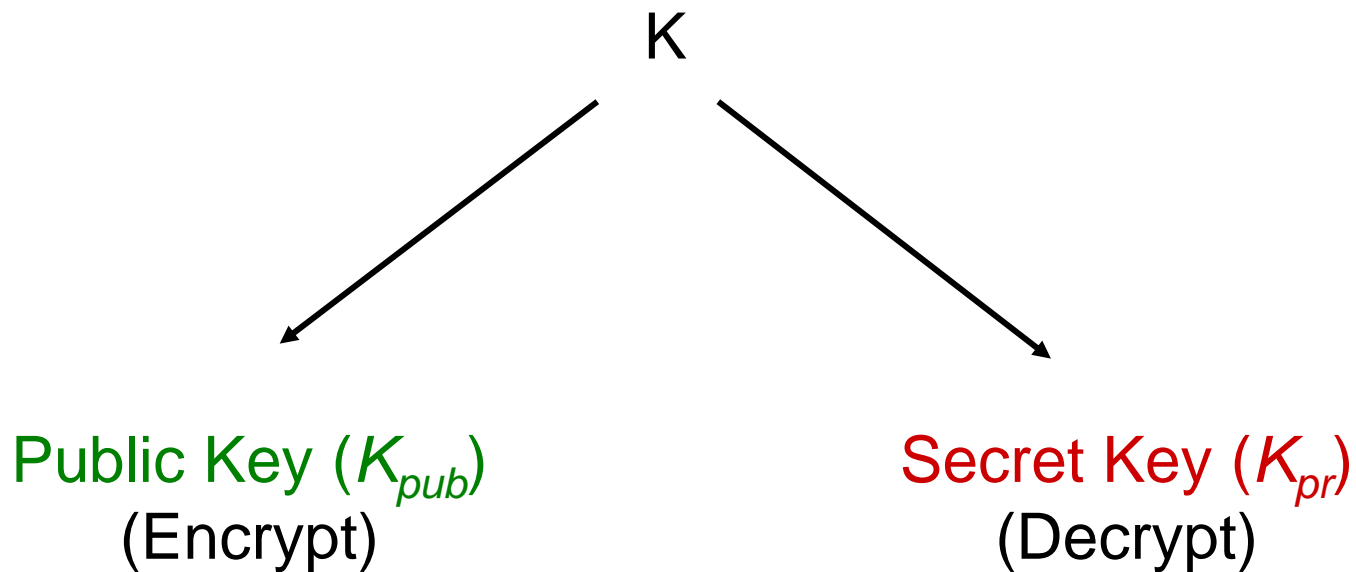


1976: first publication of such an algorithm by Whitfield Diffie and Martin Hellman, and also by Ralph Merkle.

Asymmetric cryptography, also known as public-key cryptography, is a cryptographic system that uses a pair of keys—one public and one private—offering a more flexible and secure solution to some of the limitations found in symmetric cryptography.

■ Asymmetric (Public-Key) Cryptography

Principle: “Split up” the key



→ During the key generation, a key pair K_{pub} and K_{pr} is computed

■ Asymmetric Cryptography: Analogy

Safe with public lock and private lock:



- Alice deposits (encrypts) a message with the - *not secret* - public key K_{pub}
- Only Bob has the - *secret* - private key K_{pr} to retrieve (decrypt) the message

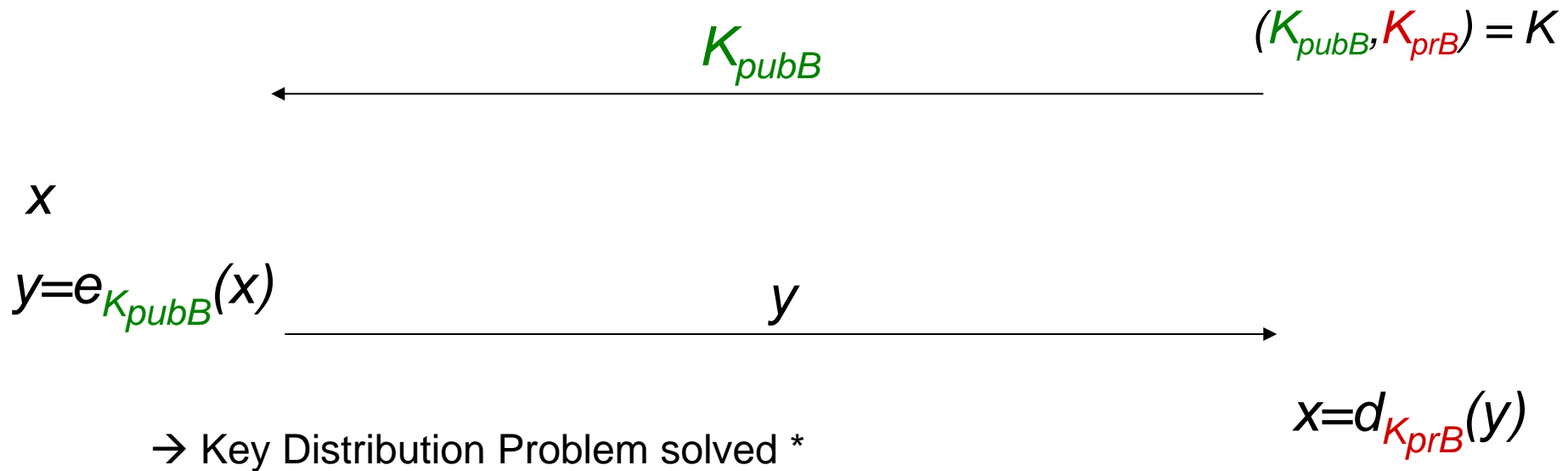
Content of this Chapter

- Symmetric Cryptography Revisited
- Principles of Asymmetric Cryptography
- **Practical Aspects of Public-Key Cryptography**
- Important Public-Key Algorithms
- Essential Number Theory for Public-Key Algorithms

■ Basic Protocol for Public-Key Encryption

Alice

Bob



*) at least for now; public keys need to be authenticated, cf. Chptr. 13 of *Understanding Cryptogr.*

The public key can be shared openly, resolving the key distribution problem seen in symmetric cryptography.

■ Security Mechanisms of Public-Key Cryptography

Here are main mechanisms that can be realized with asymmetric cryptography:

- **Key Distribution** (e.g., Diffie-Hellman key exchange, RSA) without a pre-shared secret (key)
- **Nonrepudiation and Digital Signatures** (e.g., RSA, DSA or ECDSA) to provide message integrity
- **Identification**, using challenge-response protocols with digital signatures
- **Encryption** (e.g., RSA / Elgamal)
Disadvantage: Computationally very intensive
(1000 times slower than symmetric Algorithms!)

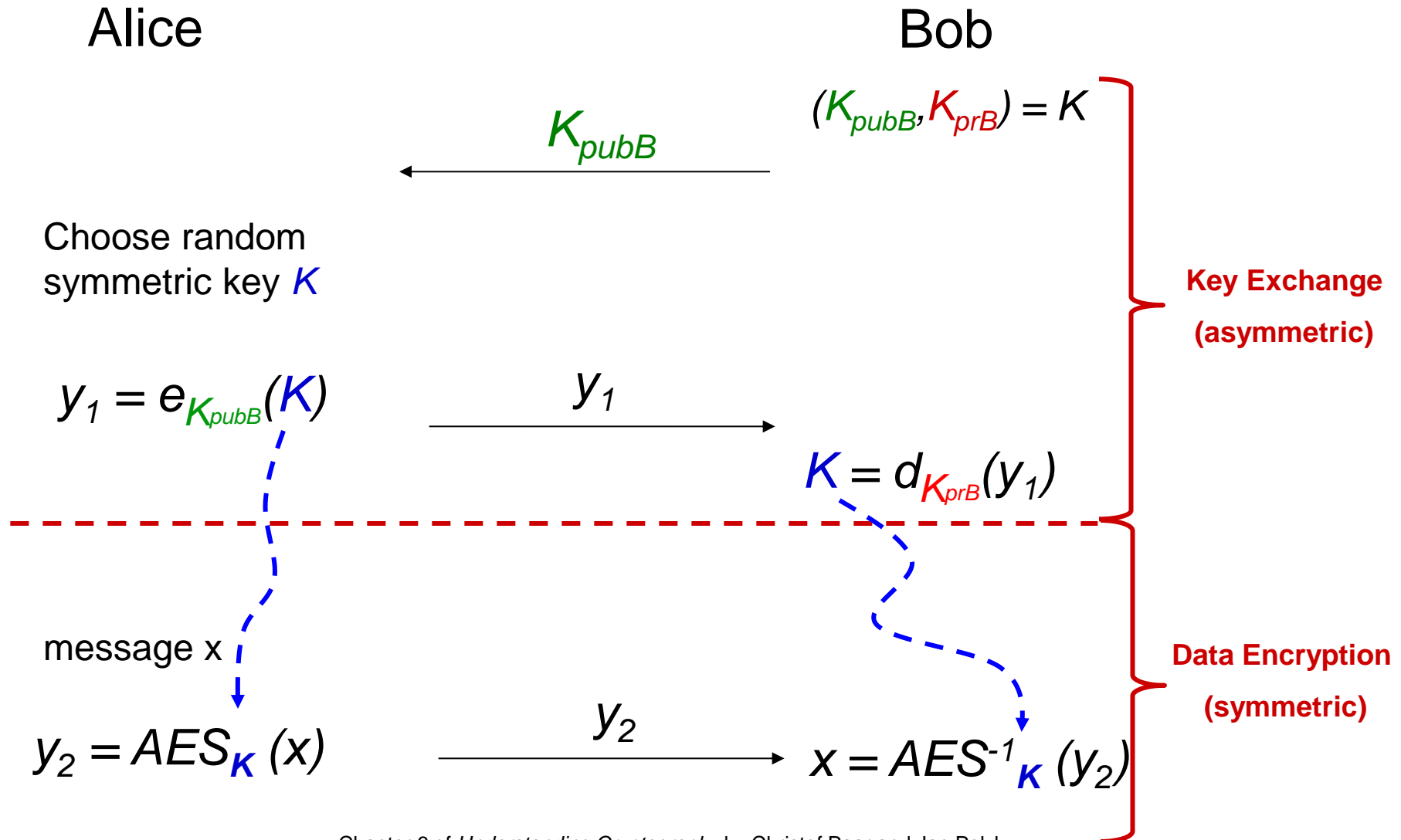
■ Basic Key Transport Protocol 1/2

In practice: **Hybrid systems**, incorporating asymmetric and symmetric algorithms

1. **Key exchange** (for symmetric schemes) and **digital signatures** are performed with (slow) **asymmetric** algorithms
2. **Encryption** of data is done using (fast) symmetric ciphers, e.g., **block ciphers** or **stream ciphers**

■ Basic Key Transport Protocol 2/2

Example: Hybrid protocol with AES as the symmetric cipher



Content of this Chapter

- Symmetric Cryptography Revisited
- Principles of Asymmetric Cryptography
- Practical Aspects of Public-Key Cryptography
- **Important Public-Key Algorithms**
- Essential Number Theory for Public-Key Algorithms

■ How to build Public-Key Algorithms

Asymmetric schemes are based on a „**one-way function**“ $f()$:

- Computing $y = f(x)$ is computationally easy
- Computing $x = f^{-1}(y)$ is computationally infeasible

One way functions are based on **mathematically hard problems**.

Three main families:

- **Factoring integers** (RSA, ...):

Given a composite integer n , find its prime factors

(Multiply two primes: easy)

- **Discrete Logarithm** (Diffie-Hellman, Elgamal, DSA, ...):

Given a , y and m , find x such that $a^x = y \bmod m$

(Exponentiation a^x : easy)

- **Elliptic Curves (EC)** (ECDH, ECDSA): Generalization of discrete logarithm

Note: The problems are considered mathematically hard, but no proof exists (so far).

■ Key Lengths and Security Levels

<i>Symmetric</i>	<i>ECC</i>	<i>RSA, DL</i>	<i>Remark</i>
64 Bit	128 Bit	≈ 700 Bit	Only short term security (a few hours or days)
80 Bit	160 Bit	≈ 1024 Bit	Medium security (except attacks from big governmental institutions etc.)
128 Bit	256 Bit	≈ 3072 Bit	Long term security (without quantum computers)

- The exact complexity of RSA (factoring) and DL (Index-Calculus) is difficult to estimate
- The existence of quantum computers would probably be the end for ECC, RSA & DL (at least 2-3 decades away, and some people doubt that QC will ever exist)

Content of this Chapter

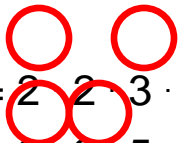
- Symmetric Cryptography Revisited
- Principles of Asymmetric Cryptography
- Practical Aspects of Public-Key Cryptography
- Important Public-Key Algorithms
- **Essential Number Theory for Public-Key Algorithms**

■ Euclidean Algorithm

is a fundamental mathematical method for finding the **greatest common divisor (GCD)** of two integers. The GCD of two numbers is the largest number that divides both without leaving a remainder.

- Compute the **greatest common divisor $\gcd(r_0, r_1)$** of two integers r_0 and r_1
- gcd is **easy for small numbers**:
 1. factor r_0 and r_1
 2. gcd = highest common factor

- Example:


$$\begin{aligned} r_0 = 84 &= 2 \cdot 2 \cdot 3 \cdot 7 \\ r_1 = 30 &= 2 \cdot 3 \cdot 5 \end{aligned}$$

→ The gcd is the product of all common prime factors:

$$2 \cdot 3 = 6 = \gcd(30, 84)$$

- **But:** Factoring is complicated (and often infeasible) for large numbers

■ Euler's Phi Function

is a number-theoretic function that calculates the number of integers less than a given integer n that are **relatively prime** to n (i.e., integers that share no common divisors with n other than 1).

- *New problem, important for public-key systems, e.g., RSA:*

Given the set of the m integers $\{0, 1, 2, \dots, m-1\}$,

How many numbers in the set are **relatively prime to m** ?

- Answer: **Euler's Phi function $\phi(m)$**

- **Example** for the sets $\{0,1,2,3,4,5\}$ ($m=6$), and $\{0,1,2,3,4\}$ ($m=5$)

$$\gcd(0, 6) = 6$$

$$\gcd(1, 6) = 1 \quad \leftarrow$$

$$\gcd(2, 6) = 2$$

$$\gcd(3, 6) = 3$$

$$\gcd(4, 6) = 2$$

$$\gcd(5, 6) = 1 \quad \leftarrow$$

$$\gcd(0, 5) = 5$$

$$\gcd(1, 5) = 1 \quad \leftarrow$$

$$\gcd(2, 5) = 1 \quad \leftarrow$$

$$\gcd(3, 5) = 1 \quad \leftarrow$$

$$\gcd(4, 5) = 1 \quad \leftarrow$$

→ 1 and 5 relatively prime to $m=6$,

hence **$\phi(6) = 2$**

→ **$\phi(5) = 4$**

- Testing one gcd per number in the set is **extremely slow for large m** .

■ Fermat's Little Theorem

is a fundamental theorem in number theory that describes a property of integers with respect to prime numbers. It's especially useful in cryptography, particularly in algorithms like RSA, as it underpins many calculations involving modular arithmetic.

- Given a **prime** p and an **integer** a : $a^p \equiv a \pmod{p}$
- Can be rewritten as $a^{p-1} \equiv 1 \pmod{p}$

- Use: Find modular inverse**, if p is prime. Rewrite to $a \cdot a^{p-2} \equiv 1 \pmod{p}$
- Comparing with definition of the modular inverse $a \cdot a^{-1} \equiv 1 \pmod{m}$
 $\rightarrow a^{-1} \equiv a^{p-2} \pmod{p}$ is the modular inverse modulo a prime p

Example: $a = 2, p = 7$

$$a^{p-2} = 2^5 = 32 \equiv 4 \pmod{7}$$

$$\text{verify: } 2 \cdot 4 \equiv 1 \pmod{7} \checkmark$$

- Fermat's Little Theorem works only **modulo a prime** p

■ Euler's Theorem

is a generalization of **Fermat's Little Theorem** and applies to any two **relatively prime integers** rather than only primes. It's especially important in cryptography, as it underpins the RSA encryption algorithm and other systems involving modular exponentiation.

- Generalization of Fermat's little theorem to **any integer modulus**
- Given two **relatively prime integers** **a** and **m** : $a^{\Phi(m)} \equiv 1 \pmod{m}$
- **Example:** $m=12$, $a=5$

1. Calculate Euler's Phi Function

$$\Phi(12) = \Phi(2^2 \cdot 3) = (2^2 - 2^1)(3^1 - 3^0) = (4 - 2)(3 - 1) = 4$$

2. Verify Euler's Theorem

$$5^{\Phi(12)} = 5^4 = 25^2 = 625 \equiv 1 \pmod{12}$$

- Fermat's little theorem = special case of Euler's Theorem
- for a prime **p** : $\Phi(p) = (p^1 - p^0) = p - 1$
→ Fermat: $a^{\Phi(p)} = a^{p-1} \equiv 1 \pmod{p}$

Practical Networking

■ Lessons Learned

- Public-key algorithms have **capabilities that symmetric ciphers don't have**, in particular digital signature and key establishment functions.
- Public-key algorithms are **computationally intensive** (a nice way of saying that they are *slow*), and hence are poorly suited for bulk data encryption.
- Only **three families of public-key schemes** are widely used. This is considerably fewer than in the case of symmetric algorithms.
- The **extended Euclidean algorithm** allows us to compute **modular inverses** quickly, which is important for almost all public-key schemes.
- **Euler's phi function** gives us the number of elements smaller than an integer n that are relatively prime to n . This is important for the RSA crypto scheme.

**Thank
You**

