Public Key Cryptography and RSA

ITC 3093 Principles of Computer Security

Based on Cryptography and Network Security by William Stallings
and Lecture slides by Lawrie Brown
and Introduction to Cryptography and Security Mechanisms by Dr Keith Martin

Every Egyptian received two names, which were known respectively as the true name and the good name, or the great name and the little name; and while the good or little name was made public, the true or great name appears to have been carefully concealed.

—*The Golden Bough, Sir James George Frazer*

Private-Key Cryptography

- traditional **private/secret/single key** cryptography uses **one** key
- shared by both sender and receiver
- if this key is disclosed communications are compromised
- also is **symmetric**, parties are equal
- hence does not protect sender from receiver forging a message & claiming is sent by sender

Public-Key Cryptography

- probably most significant advance in the 3000 year history of cryptography
- uses **two** keys – a public & a private key
- **asymmetric** since parties are **not** equal
- uses clever application of number theoretic concepts to function
- complements **rather than** replaces private key crypto

Why Public-Key Cryptography?

- developed to address two key issues:
  - **key distribution** – how to have secure communications in general without having to trust a KDC with your key
  - **digital signatures** – how to verify a message comes intact from the claimed sender
- public invention due to Whitfield Diffie & Martin Hellman at Stanford Uni in 1976
  - known earlier in classified community

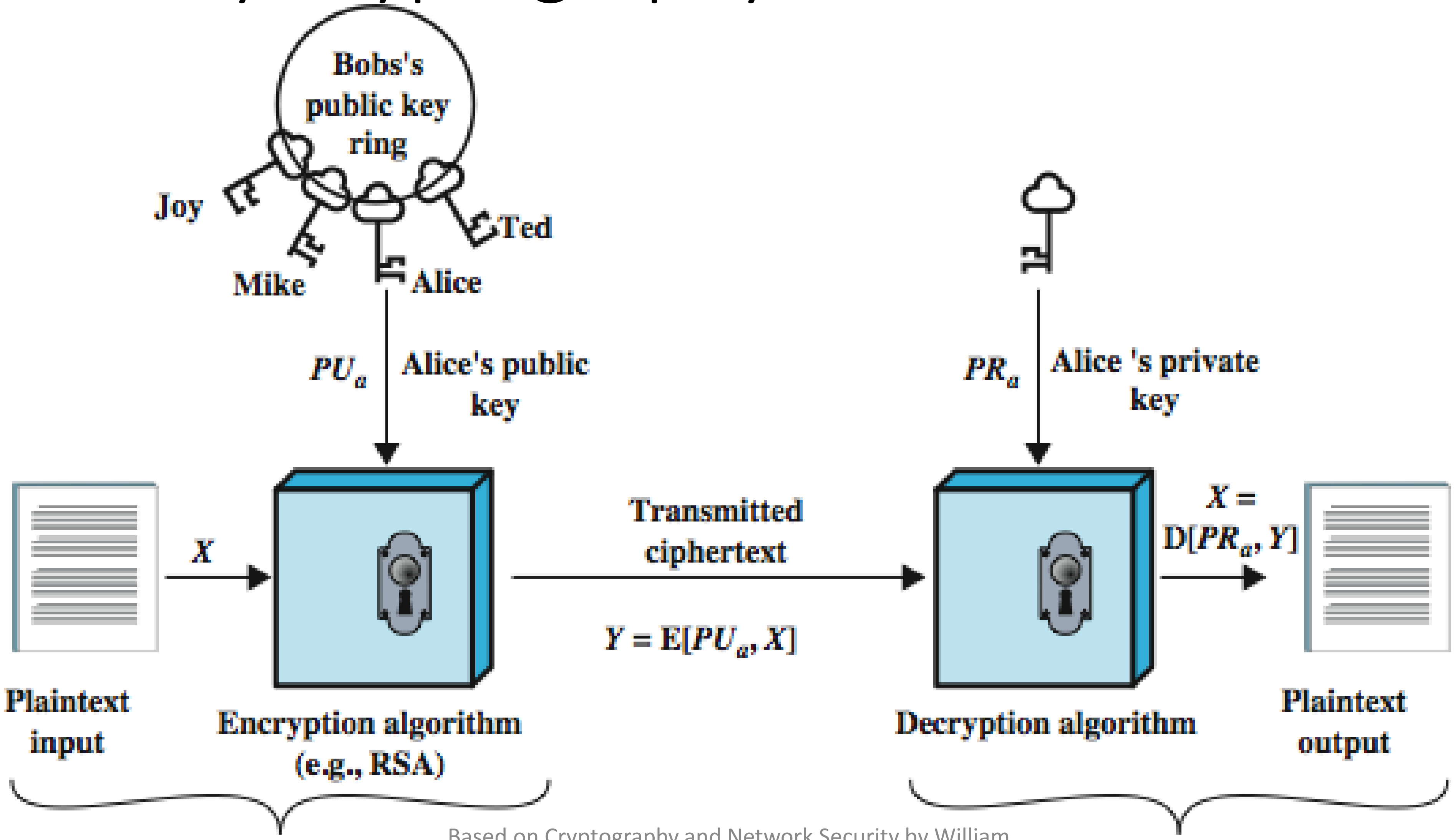
Public-Key Cryptography

- **public-key/two-key/asymmetric** cryptography involves the use of **two** keys:
  - a **public-key**, which may be known by anybody, and can be used to **encrypt messages**, and **verify signatures**
  - a related **private-key**, known only to the recipient, used to **decrypt messages**, and **sign** (create) **signatures**
- **infeasible to determine private key from public**
- is **asymmetric** because
  - those who encrypt messages or verify signatures **cannot** decrypt messages or create signatures

Public-Key Cryptography

- Consider the following analogy using padlocked boxes:
- Traditional schemes involve the sender putting a message in a box and locking it, sending that to the receiver, and somehow securely also sending them the key to unlock the box.
- The radical advance in public key schemes was to turn this around, the receiver sends an **unlocked box** (their public key) to the sender.
- Sender puts the message in the box and locks it (easy - and having locked it cannot get at the message), and sends the locked box to the receiver who can unlock it (also easy), having the (private) key.
- An attacker would have to pick the lock on the box (hard).

Public-Key Cryptography

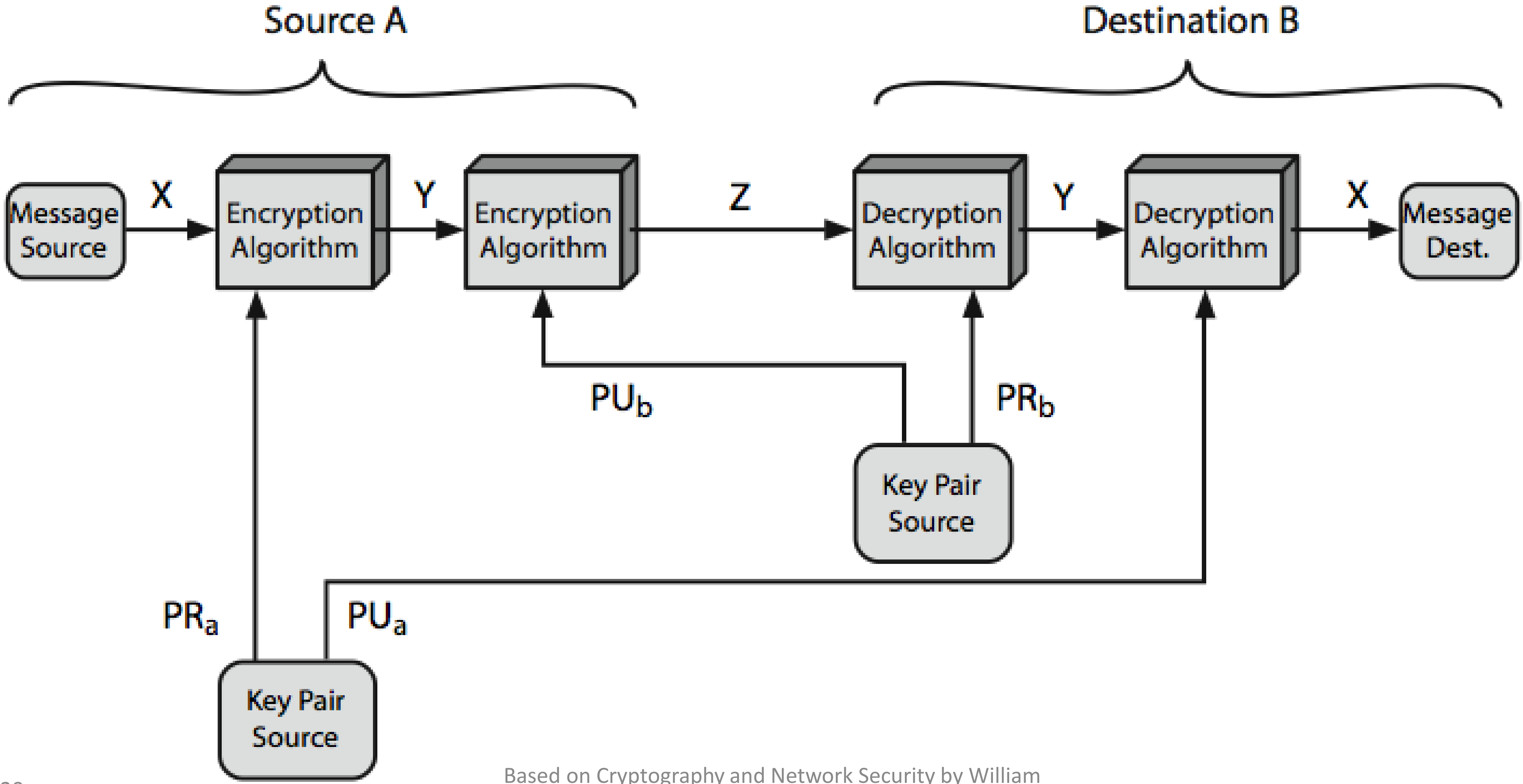


(a) Encryption with public key

Symmetric vs Public-Key

Conventional Encryption	Public-Key Encryption
Needed to Work: 1. The same algorithm with the same key is used for encryption and decryption. 2. The sender and receiver must share the algorithm and the key. *Needed for Security:* 1. The key must be kept secret. 2. It must be impossible or at least impractical to decipher a message if no other information is available. 3. Knowledge of the algorithm plus samples of ciphertext must be insufficient to determine the key.	*Needed to Work:* 1. One algorithm is used for encryption and decryption with a pair of keys, one for encryption and one for decryption. 2. The sender and receiver must each have one of the matched pair of keys (not the same one). *Needed for Security:* 1. One of the two keys must be kept secret. 2. It must be impossible or at least impractical to decipher a message if no other information is available. 3. Knowledge of the algorithm plus one of the keys plus samples of ciphertext must be insufficient to determine the other key.

Public-Key Cryptosystems



Public-Key Applications

- can classify uses into 3 categories:
  - **encryption/decryption** (provide secrecy)
  - **digital signatures** (provide authentication)
  - **key exchange** (of session keys)
- some algorithms are suitable for all uses, others are specific to one

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

Public-Key Requirements

- Public-Key algorithms rely on two keys where:
  - it is computationally infeasible to find decryption key knowing only algorithm & encryption key
  - it is computationally easy to en/decrypt messages when the relevant (en/decrypt) key is known
  - either of the two related keys can be used for encryption, with the other used for decryption (for some algorithms)
- these are formidable requirements which only a few algorithms have satisfied

Public-Key Requirements

- need a trapdoor one-way function
- one-way function has
  - $Y = f(X)$ easy
  - $X = f^{-1}(Y)$ infeasible
- a trap-door one-way function has
  - $Y = f_k(X)$ easy, if k and X are known
  - $X = f_k^{-1}(Y)$ easy, if k and Y are known
  - $X = f_k^{-1}(Y)$ infeasible, if Y known but k not known
- a practical public-key scheme depends on a suitable trap-door one-way function

Security of Public Key Schemes

- like private key schemes brute force **exhaustive search** attack is always theoretically possible
- but keys used are too large (>512bits)
- security relies on a **large enough** difference in difficulty between **easy** (en/decrypt) and **hard** (cryptanalyse) problems
- more generally the **hard** problem is known, but is made hard enough to be impractical to break
- requires the use of **very large numbers**
- hence is **slow** compared to private key schemes

One way functions

a function that is “easy” to compute and “difficult” to reverse.

OWF: Multiplying two primes

- It is easy to take two prime numbers and multiply them together.
- If they are fairly small we can do this in our heads, on a piece of paper, or on a calculator.
- As they get bigger and bigger it is fairly easy to write a computer program to compute the product.
- Multiplication runs in polynomial time.
- Multiplication of two primes is easy.

OWF: Multiplying two primes

To factor:	Comments
15	
143	
6887	
31897	
A 600 digit number	
A 600 digit even number	

OWF: Multiplying two primes

- Multiplication of two prime numbers is **believed** to be a one-way function.
- We say **believed** because nobody has been able to **prove** that it is hard to factorise.
- Maybe one day someone will find a way of factorising efficiently.
- What will happen if someone does find an efficient way of factorising?

OWF: Modular exponentiation

- The process of **exponentiation** just means raising numbers to a power.
- Raising **a** to the power **b**, normally denoted a^b just means multiplying **a** by itself **b** times. In other words:

$$a^b = a \times a \times a \times \ldots \times a$$

- **Modular exponentiation** means computing a^b modulo some other number **n**. We tend to write this as

$$a^b \bmod n.$$

- Modular exponentiation is “easy”.

OWF: Modular exponentiation

- However, given **a**, **b**, and **a^b mod n** (when **n** is prime), calculating **b** is regarded by mathematicians as a hard problem.
- This difficult problem is often referred to as the **discrete logarithm problem**.
- In other words, given a number **a** and a prime number **n**, the function

$$f(b) = a^b \bmod n$$

- is believed to be a one-way function.

OWF: Modular square roots

- What is the square root of 1369?
  - Propose a technique for finding the square root of 1369 that will generalise to any integer.

- What is the square root of 56 module 101?
  - Let's try 40…
  - Let's try 30…

Suitable OWFs

- We have seen that the encryption process of a public key cipher system requires a one way function.
- Is every one way function suitable for implementation as the encryption process of a public key cipher system?

RSA

RSA

- by Rivest, Shamir & Adleman of MIT in 1977
- best known & widely used public-key scheme
- based on exponentiation in a finite (Galois) field over integers modulo a prime
    - nb. exponentiation takes $O((\log n)^3)$ operations (easy)
- uses large integers (eg. 1024 bits)
- security due to cost of factoring large numbers
    - nb. factorization takes $O(e^{\log n \log \log n})$ operations (hard)

RSA En/decryption

- to encrypt a message M the sender:
  - obtains **public key** of recipient $PU=\{e,n\}$
  - computes: $C = M^e \bmod n$, where $0 \leq M < n$
- to decrypt the ciphertext C the owner:
  - uses their private key $PR=\{d,n\}$
  - computes: $M = C^d \bmod n$
- both sender and receiver must know the value of n.
- sender knows the value of e, and only the receiver knows the value of d.

RSA Example - En/Decryption

- sample RSA encryption/decryption is:
- given message $M = 88$ (nb. $88<187$)
- encryption:
  - $C = 88^7 \bmod 187 = 11$
- decryption:
  - $M = 11^{23} \bmod 187 = 88$

RSA Key Generation

- users of RSA must:
  - determine two primes at random - `p, q`
  - select either `e` or `d` and compute the other
- primes `p,q` must not be easily derived from modulus `n=p.q`
  - means must be sufficiently large
  - typically guess and use probabilistic test
- exponents `e`, `d` are inverses, so use inverse algorithm to compute the other

RSA Security

- possible approaches to attacking RSA are:
  - brute force key search - infeasible given size of numbers
  - mathematical attacks - based on difficulty of computing $\phi(n)$, by factoring modulus n
  - timing attacks - on running of decryption
  - chosen ciphertext attacks - given properties of RSA

Factoring Problem

- mathematical approach takes 3 forms:
  - factor n=p.q, hence compute ø(n) and then d
  - determine ø(n) directly and compute d
  - find d directly
- currently believe all equivalent to factoring
  - have seen slow improvements over the years
    - as of May-05 best is 200 decimal digits (663) bit with LS
  - biggest improvement comes from improved algorithm
    - cf QS to GHFS to LS
  - currently assume 1024-2048 bit RSA is secure
    - ensure p, q of similar size and matching other constraints

Summary

- Public key systems replace the problem of distributing symmetric keys with one of authenticating public keys
- Public key encryption algorithms need to be trapdoor one-way functions
- RSA is a public key encryption algorithm whose security is believed to be based on the problem of factoring large numbers