ΕΠΛ 674: Εργαστήριο 3

Ο αλγόριθμος ασύμμετρης κρυπτογράφησης RSA

University of Cyprus
Department of
Computer Science

Παύλος Αντωνίου

NETworks Research Laboratory

Private-Key Cryptography

University of Cyprus

- 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

University of Cyprus

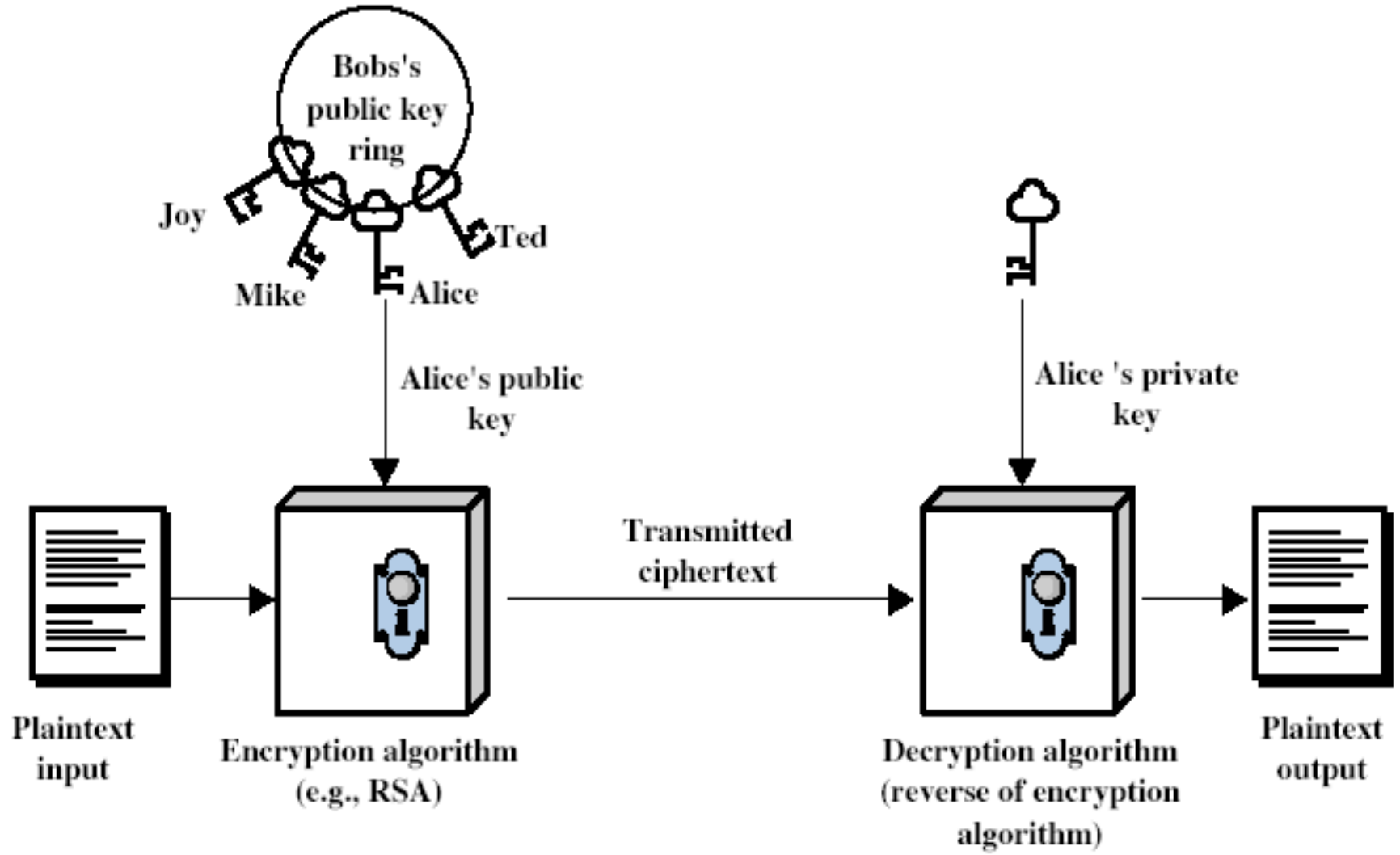
- uses **two** keys – a public & a private key
- **asymmetric** since parties are **not** equal
- uses clever application of number theoretic concepts to function
- complements **rather than** replaces private key cryptography

Public-Key Cryptography

University of Cyprus

- **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 **private-key**, known only to the recipient, used to **decrypt messages**, and **sign** (create) **signatures**
- is **asymmetric** because
  - those who encrypt messages or verify signatures **cannot** decrypt messages or create signatures

Public-Key Cryptography



Why Public-Key Cryptography?

University of Cyprus

- developed to address two key issues:
  - **key distribution** – how to have secure communications in general without having to trust a key distribution center (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 University in 1976

Public-Key Characteristics

University of Cyprus

- Public-Key algorithms rely on two keys with the characteristics that it is:
  - computationally infeasible to find decryption key knowing only algorithm & encryption key
  - 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 (in some schemes)

Public-Key Cryptosystems

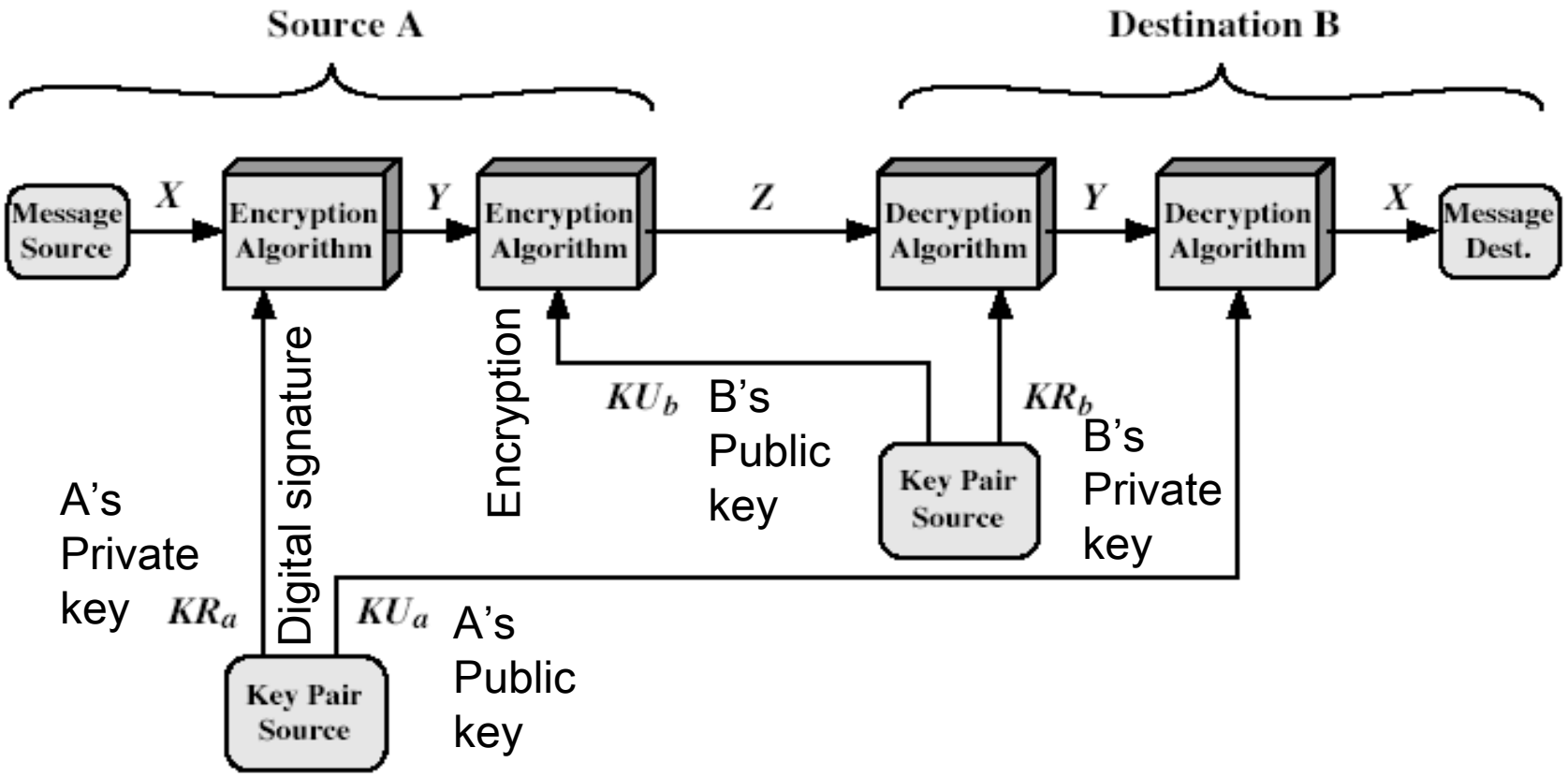


Figure 9.4 Public-Key Cryptosystem: Secrecy and Authentication

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

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, its just made too hard to do in practise
- requires the use of **very large numbers**
- hence is **slow** compared to private key schemes

RSA

- by Rivest, Shamir & Adleman of MIT in 1977
- best known & widely used public-key scheme
- security due to cost of factoring (παραγοντοποίηση) large numbers

Prime numbers

University of Cyprus

- Prime numbers only have divisors of 1 and self
  - they cannot be written as a product of other numbers
  - note: 1 is prime, but is generally not of interest
- eg. 2,3,5,7 are prime, 4,6,8,9,10 are not
- prime numbers are central to number theory
- list of prime number less than 200 is:

```
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61
67 71 73 79 83 89 97 101 103 107 109 113 127 131
137 139 149 151 157 163 167 173 179 181 191 193
197 199
```

RSA Key Setup

University of Cyprus

- Each user generates a public/private key pair by:
- selecting two large primes at random: p, q
- computing system modulus $N=p.q$
  - RSA currently recommends a modulus that's at least 768 bits long
- computing $\varphi=(p-1)(q-1)$
- selecting at random the encryption key e
  - where $1<e<\varphi$, $\gcd(e,\varphi)=1$
  - In practice, common choices for *e* are 3, 17 and 65537 (Fermat primes)
  - **Public encryption key KU={e,N}**

RSA Key Setup

University of Cyprus

- computing a private key d so that $e.d$ leaves a remainder of 1 when divided by φ.
  - We say $e.d$ is ***congruent*** to 1 ***modulo*** φ
- solving following equation to find decryption key d
  - $e.d = 1 \pmod{\varphi} \Rightarrow e.d = k.\varphi + 1$, and $0 \leq d \leq N$

    $d = e^{-1} \pmod{\varphi}$ (modular inverse → can be computed using the Extended Euclidean algorithm. More info can be found here)
  - **Private encryption key KR={d,p,q}**
- Note that d is easy to compute only if one knows the value of φ. This is essentially the same as knowing the values of p and q.

RSA – encryption/decryption

University of Cyprus

- If M is any number that is not divisible by N, then dividing $M^{e \cdot d}$ by N and taking the remainder yields the original value M.
  - This is a relatively deep mathematical theorem, which we can write as $M^{e \cdot d} \bmod N = M$
- If M is a numeric encoding of a block of plaintext, the cyphertext is $C = M^e \bmod N$.
- Then $C^d \bmod N = (M^e \bmod N)^d \bmod N = (M^e)^d \bmod N = M^{e \cdot d} \bmod N = M$. Thus, we can recover the plaintext M with the private key d.

RSA (en/de)cryption implementation

University of Cyprus

- Let's say we want to compute:
- $c = m^d \bmod n$
  Note that we don't have to calculate the full value of m to the power d here. We can make use of the fact that:
  $a = bc \bmod n = (b \bmod n).(c \bmod n) \bmod n$
  so we can break down a potentially large number into its components and combine the results of easier, smaller calculations to calculate the final value.
- For example: (m=13, d=7, n=33)
  $c = 13^7 \bmod 33 = 13^{(3+3+1)} \bmod 33 = 13^3.13^3.13 \bmod 33$
  $= (13^3 \bmod 33).(13^3 \bmod 33).(13 \bmod 33) \bmod 33$
  $= (2197 \bmod 33).(2197 \bmod 33).(13 \bmod 33) \bmod 33$
  $= 19.19.13 \bmod 33 = 4693 \bmod 33$
  $= 7.$

RSA summary

University of Cyprus

- N=p.q where p, q are distinct prime numbers
- φ, φ=(p-1).(q-1)
- e<N, gcd(e,φ)=1 & $d=e^{-1} \pmod{\varphi}$
- **public** encryption key: KU={e,N}
- **private** decryption key: KR={d,p,q}
- to encrypt a message M the sender:
  - obtains **public key** of recipient
  - computes: $C=M^e \bmod N$, where $0 \leq M < N$
- to decrypt the ciphertext C the owner:
  - uses their private key KR={d,p,q}
  - computes: $M=C^d \bmod N$
- note that the message M must be smaller than the modulus N (block if needed)

Why RSA works?

University of Cyprus

- Multiplying `p` by `q` is ***easy***: the number of operations depends on the ***number of bits*** (number of digits) in `p` and `q`.
- For example, multiplying two 384-bit numbers takes approximately
  $384^2 = 147{,}456$ bit operations

Why RSA works? (2)

University of Cyprus

- If one knows only N, finding p and q (factorization) is ***hard***: in essence, the number of operations depends on the ***value*** of N.
  - The simplest method for factoring a 768-bit number takes about $2^{384} \approx 3.94 \times 10^{115}$ trial divisions.
  - A more sophisticated methods takes about $2^{85} \approx 3.87 \times 10^{25}$ trial divisions.
  - A still more sophisticated method takes about $2^{41} \approx$ 219,000,000,000 trial divisions
- No-one has found an really quick algorithm for factoring a large number N.
- No-one has proven that such a quick algorithm doesn't exist (or even that one is unlikely to exist)

RSA example

1. Select primes: $p=17$ & $q=11$
2. Compute $n = pq = 17 \times 11 = 187$
3. Compute $\varphi = (p-1)(q-1) = 16 \times 10 = 160$
4. Select e: $\gcd(e,160)=1$; choose $e=7$
5. Determine d: $d.e=1 \pmod{160}$
   Value is $d=23$ since $23 \times 7 = 161 = 10 \times 160 + 1$
6. Publish public key $KU=\{7,187\}$
7. Keep secret private key $KR=\{23,17,11\}$

RSA example (cont)

University of Cyprus

- 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 security

University of Cyprus

- three approaches to attacking RSA:
  - brute force key search (infeasible given size of numbers)
  - mathematical attacks (based on difficulty of computing φ, by factoring modulus N)
  - timing attacks (on running of decryption)

References

[1] RSA Algorithm: http://www.di-mgt.com.au/rsa_alg.html

[2] Extended Euclidean Algorithm: http://www.di-mgt.com.au/euclidean.html#extendedeuclidean