CS557: Cryptography

Public-key Cryptography-1

S. Tripathy
IIT Patna

Previous Class

- Symmetric key cryptography
  - Block cipher
  - Stream Cipher
  - Random Number Generator
  - Hash Function
  - MAC

Present Class

- Public Key Cryptography
  - Public Key Encryption
    - RSA

Private-Key Cryptography

- Traditional *private/ secret key* cryptography uses *one* key shared by both sender and receiver
- *symmetric*, Means parties are equal.
- if this key is disclosed communications are compromised

- Whats Problem?
  - Does not protect sender from receiver forging a message & claiming is sent by sender
  - Key distribution and management is a serious problem! . N users –$O(N^2)$ keys!

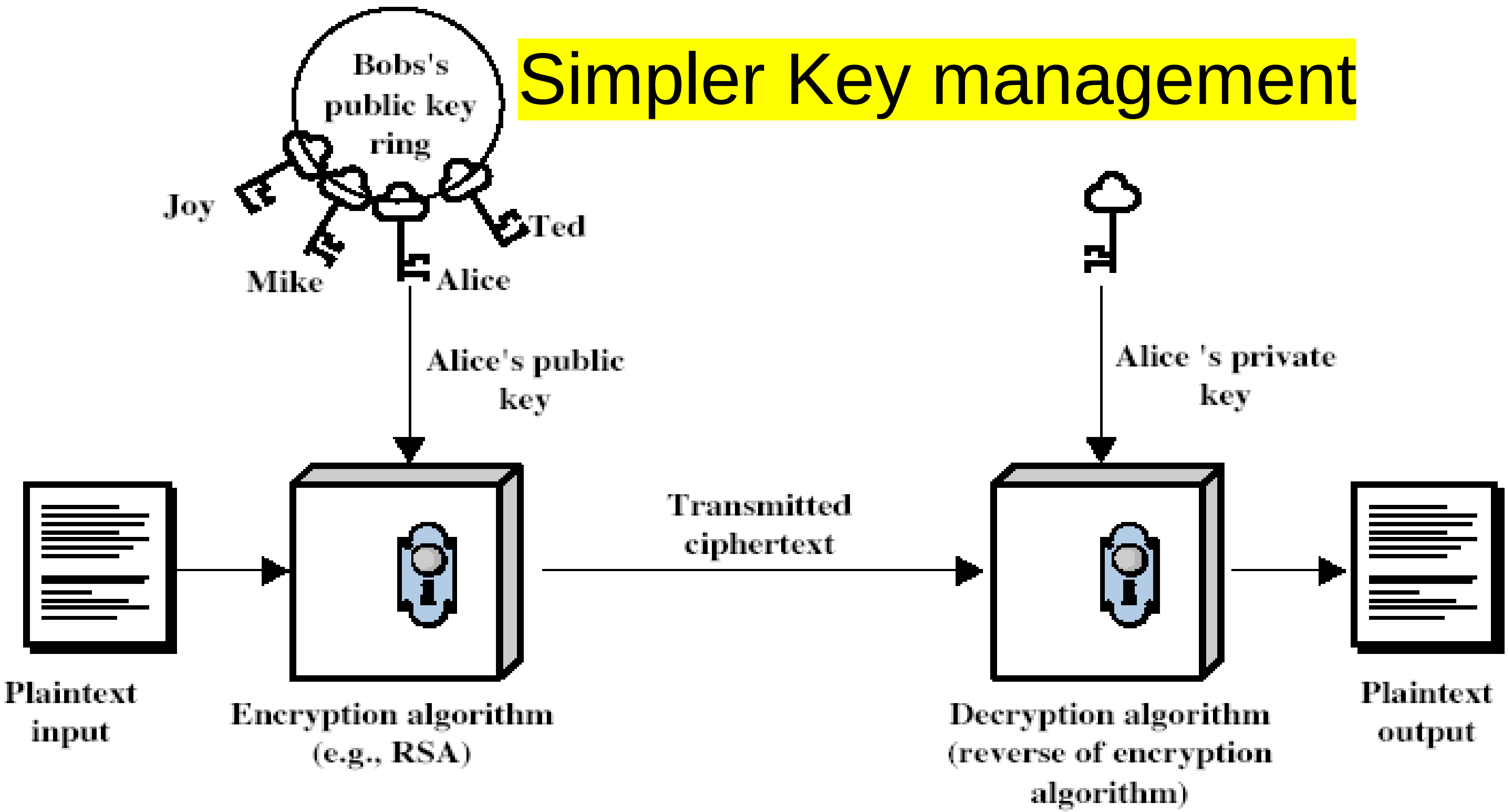
Public-Key Cryptography

- uses *two* keys – a public & a private key
- complements *rather than* replaces private key crypto
  - 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*

- *Asymmetric* because
  - those who encrypt messages or verify signatures *cannot* decrypt messages or create signatures
  - uses clever application of number theoretic concepts to function

Public Key Cryptography

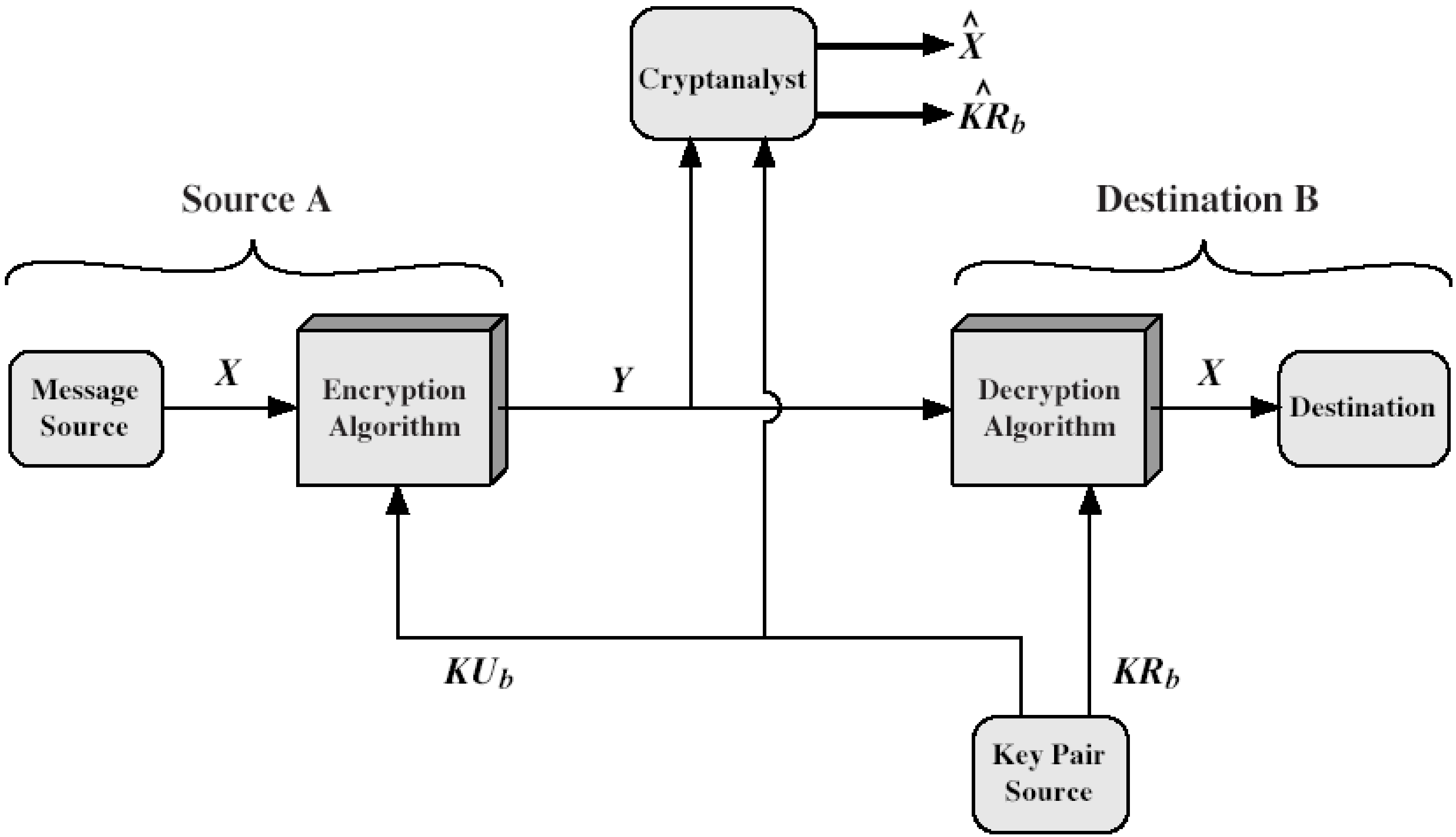
Simpler Key management



Public-Key Applications

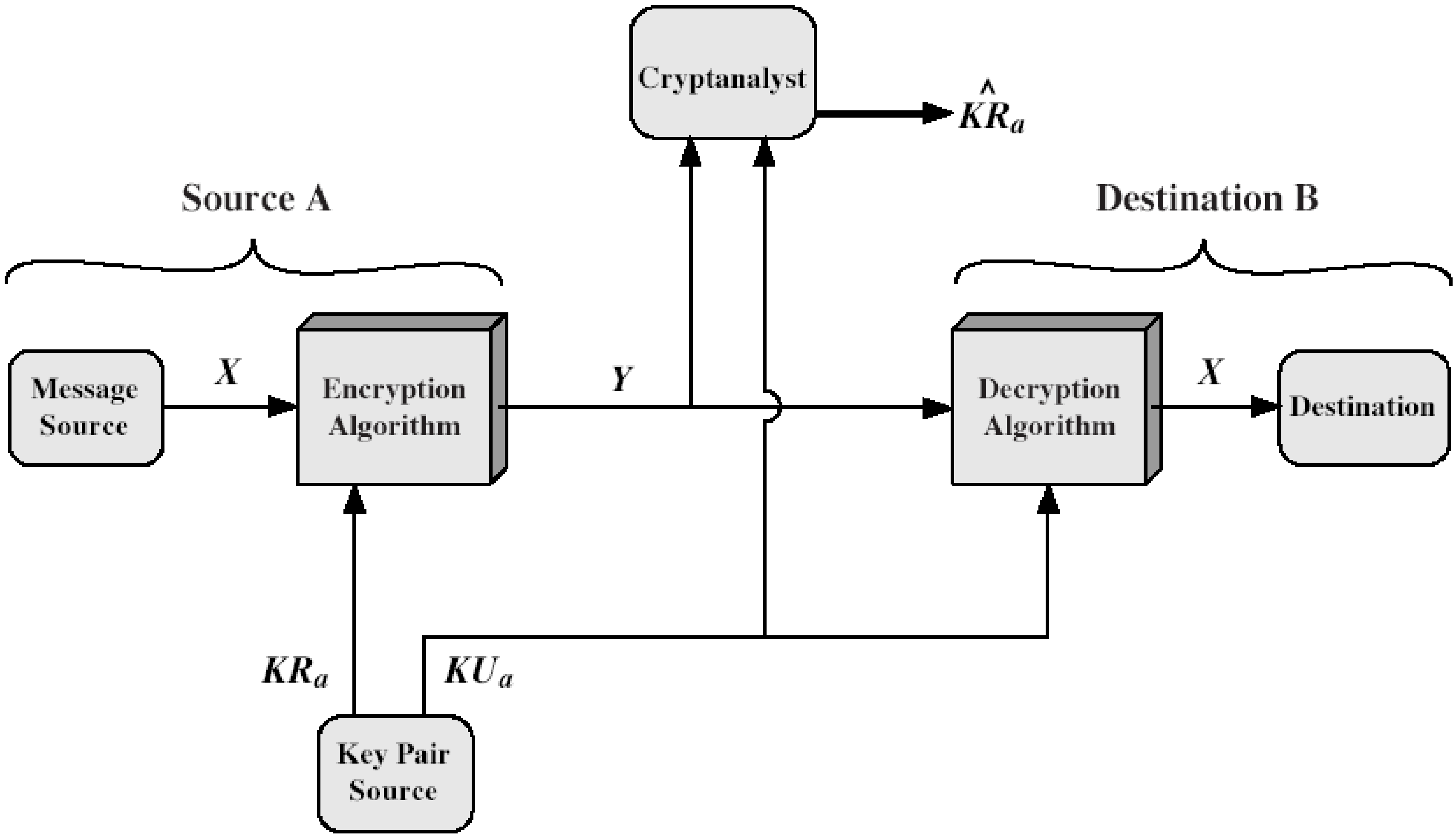
- 3 different categories of applications:
  - *encryption/decryption* (provide secrecy)
  - *digital signatures* (provide authentication)
  - *key exchange* (of session keys)
  - some algorithms are suitable for all uses, others are specific to one
- Must ensure that the public key belongs to the correct party (binding of identity to key). The public key directory may be corrupted:
  - Solution: Use a Public Key Infrastructure (PKI) to certify your keys

Public-Key Cryptosystems (confidentiality)



Public-Key Cryptosystem: Secrecy

Public-Key Cryptosystems (Authentication)



Public-Key Cryptosystem: Authentication

Public-Key Cryptosystems

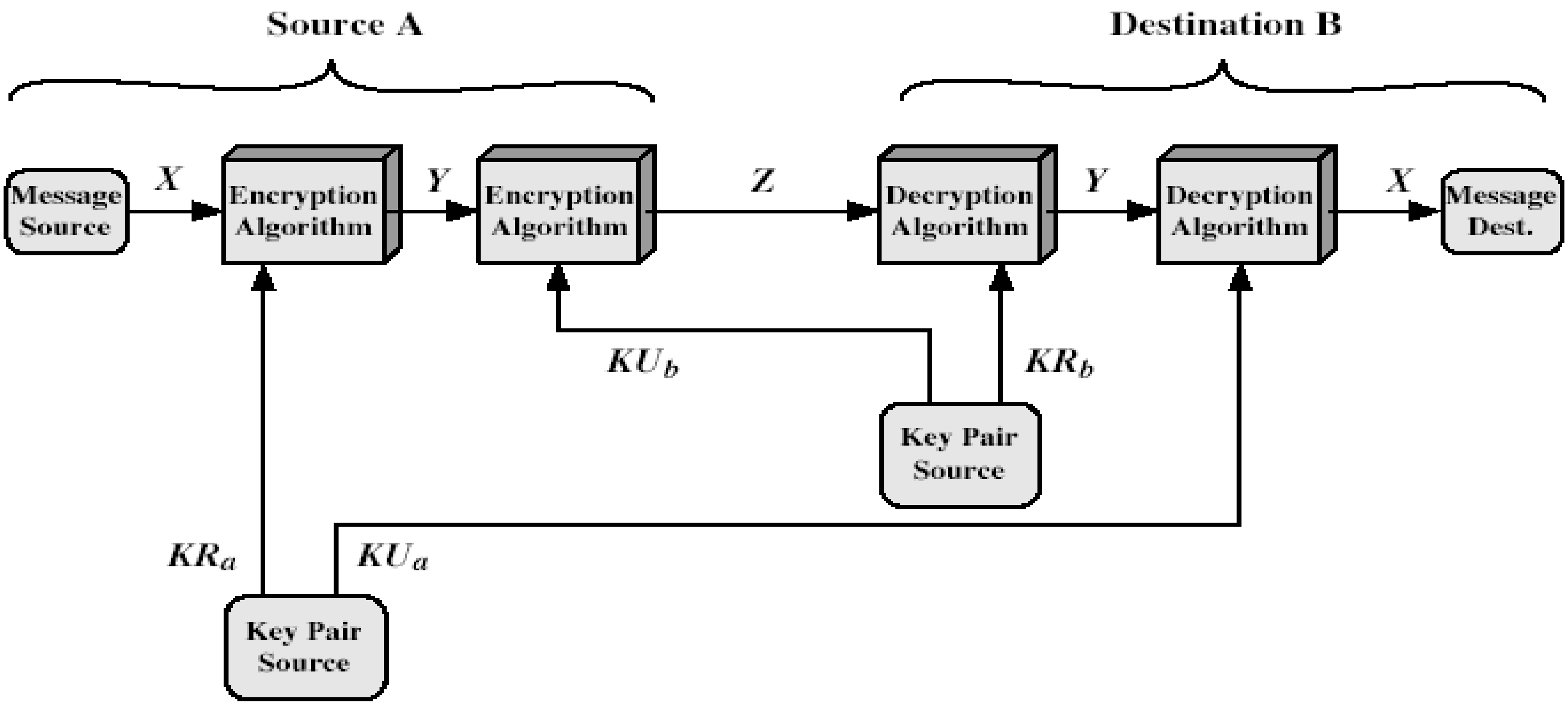


Figure 9.4 Public-Key Cryptosystem: Secrecy and Authentication

Public-Key Characteristics

- Two keys:
  - public encryption key e & private decryption key d

- Encryption is easy when e is known
- Decryption is hard when d is not known
  - d provides "trap door": decryption is easy when d is known

One-way Trapdoor function

- A function f( ) is said to be one-way if given x it is "easy" to compute $y = f(x)$, but given y it is "hard" to compute $x = f^{-1}(y)$.
- A trap-door one-way function $f_K()$ is such that to compute
- $y = f_K(x)$ **is easy if K and x are known.**
- $x = f^{-1}_K(y)$ **is easy if K and y are known.**
- $x = f^{-1}_K(y)$ **is hard if y is known but K is unknown.**
- Given a trap-door one-way function one can design a public key cryptosystem.

Security of Public Key Schemes

- like private key schemes brute force *exhaustive search* attack is always theoretically possible

  but keys used are too large

- security relies on a *large enough* difference in difficulty between easy (en/decrypt) and *hard* (cryptanalysis) 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
- based on exponentiation in a finite (Galois) field over integers modulo a prime

- uses large integers (eg. 1024 bits)
- security due to cost of factoring large numbers
  - number factorization takes $O(e^{\log n \log \log n})$ operations (hard)

RSA Key Setup

- each user generates a public/private key pair by:
  - selecting two large primes at random : p, q
  - computing their system modulus $N=p.q$
    - note $\varnothing(N)=(p-1)(q-1)$
- select at random the encryption key e
  - where $1<e<\varnothing(N)$, $\gcd(e,\varnothing(N))=1$
- solve following equation to find decryption key d
  - $e.d=1 \bmod \varnothing(N)$ and $0\le d\le N$
- publish their public encryption key: KU={e,N}
- keep secret private decryption key: KR={d,p,q}

RSA Use

- to encrypt a message M the sender:
  - obtains **public key** of recipient $KU=\{e,N\}$
  - computes: $C=M^e \bmod N$, where $0 \le M < N$
- 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)

Correctness

- RSA
  - $N=p.q$
  - $\varnothing(N)=(p-1)(q-1)$
  - carefully chosen e & d to be inverses $\bmod \varnothing(N)$
  - hence $e.d=1+k.\varnothing(N)$ for some k
- hence :

$$C^d = (M^e)^d = M^{1+k.\varnothing(N)} = M^1.(M^{k.\varnothing(N)})$$

$$C^d \bmod N = M^1.(1)^k \bmod N = M \bmod N$$

RSA Example

- Select primes: $p = 61$ and $q = 53$
- Compute $n = pq = 61 * 53 = 3233$
- Compute $\varnothing(n)=(p-1)(q-1)=60\times52=3120$

- Select e : $\gcd(e,3120)=1$; choose $e=17$
- Determine d: $d.e=1 \bmod 3120$ and $d < 3120$ Value is $d = 2753$ since $17 * 2753 = 46801 = 1 + 15 * 3120$.
- Publish public key $KU=\{n = 3233, e = 17\}$
- Keep secret private key $KR=\{(d = 2753, p=61, q=53\}$

sample RSA encryption/decryption is:

- given message $M = 123$ (number $123<3233$)
- encryption:

  $C = 123^{17} \bmod 3233 = 855$
- decryption:

  $M = 855^{2753} \bmod 3233 = 123$

  Computation over Large numbers (Multi precision integer)

Exponentiation

- can use the Square and Multiply Algorithm (Already discussed) a fast, efficient algorithm for exponentiation
- concept is based on repeatedly squaring base and multiplying in the ones that are needed to compute the result
- look at binary representation of exponent; only takes $O(\log_2 n)$ multiples for number n
  - **eg.** $7^5 = 7^4.7^1 = 3.7 = 10 \bmod 11$
  - **eg.** $3^{129} = 3^{128}.3^1 = 5.3 = 4 \bmod 11$

RSA Encryption is one-way trapdoor

- Now $D_d [E_e[x]] = x$
- $E[x]$ and $D[y]$ can be computed efficiently if keys are known

- $E^{-1}[y]$cannot be computed efficiently without knowledge of the (private) decryption key d.

- *Also*, it should be possible to select keys reasonably efficiently. Efficiency requirements are less stringent since it has not to be done too often.

- Thanks