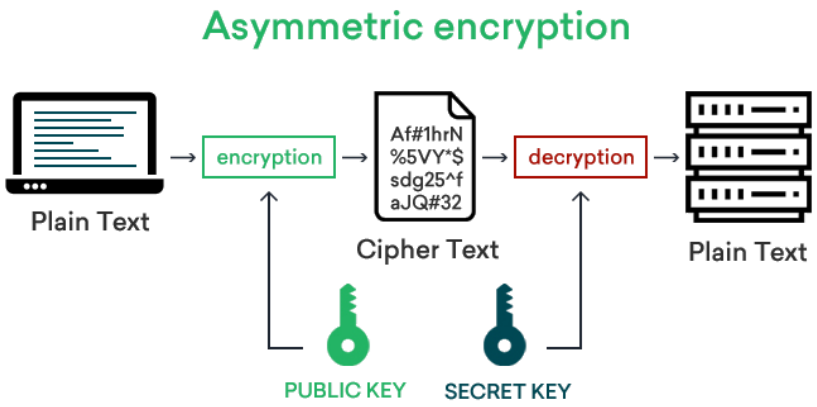


Computer Security

Public-Key Cryptography

Asymmetric cryptography

How long do you want these messages to remain secret? I want them to remain secret for as long as men are capable of evil.
— Neal Stephenson

Tamer ABUHMED
Department of Computer Science & Engineering
Sungkyunkwan University

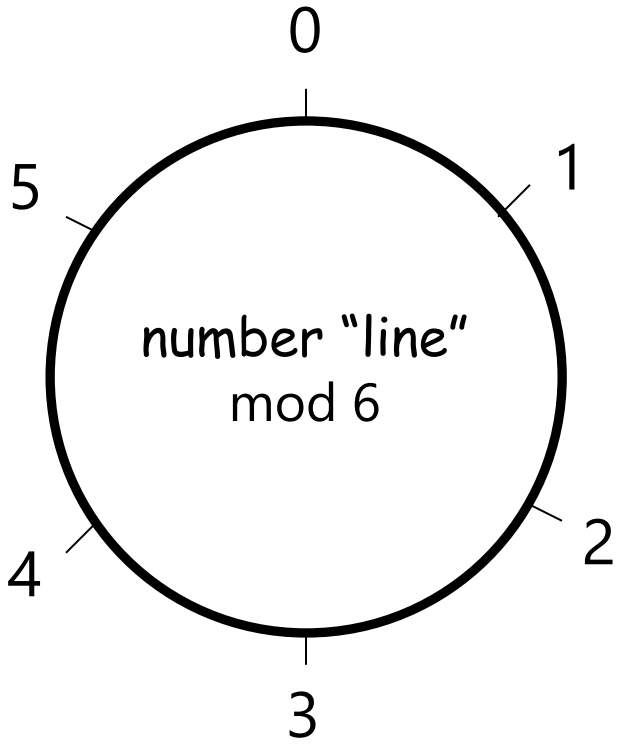
Outline

- Introduction to number theory
- Public-Key Cryptography – General Characteristics
- Public-Key Cryptography - Encryption
- Public-Key Cryptography - Authentication
- Some Issues of Public Key Schemes
- Public-Key cryptosystem : RSA
- RSA Security

Clock Arithmetic

- For integers x and n, "x mod n" is the remainder when we compute x ÷ n
  - We can also say "x modulo n"

- Examples
  - 7 mod 6 = 1
  - 33 mod 5 = 3
  - 33 mod 6 = 3
  - 51 mod 17 = 0
  - 17 mod 6 = 5



Modular Addition

- Notation and facts
  - 7 mod 6 = 1
  - 7 = 13 = 1 mod 6
  - ((a mod n) + (b mod n)) mod n = (a + b) mod n
  - ((a mod n)(b mod n)) mod n = ab mod n
- Addition Examples
  - 3 + 5 = 2 mod 6
  - 2 + 4 = 0 mod 6
  - 3 + 3 = 0 mod 6
  - (7 + 12) mod 6 = 19 mod 6 = 1 mod 6
  - (7 + 12) mod 6 = (1 + 0) mod 6 = 1 mod 6

Modular Multiplication

- Multiplication Examples
  - $3 \cdot 4 = 0 \pmod 6$
  - $2 \cdot 4 = 2 \pmod 6$
  - $5 \cdot 5 = 1 \pmod 6$
  - $(7 \cdot 4) \bmod 6 = 28 \bmod 6 = 4 \bmod 6$
  - $(7 \cdot 4) \bmod 6 = (1 \cdot 4) \bmod 6 = 4 \bmod 6$

Modular Inverses

- *Additive inverse* of x mod n, denoted $-x \bmod n$, is the number that must be added to x to get 0 mod n
  - $-2 \bmod 6 = 4$, since $2 + 4 = 0 \bmod 6$
- *Multiplicative inverse* of x mod n, denoted $x^{-1} \bmod n$, is the number that must be multiplied by x to get 1 mod n
  - $3^{-1} \bmod 7 = 5$, since $3 \cdot 5 = 1 \bmod 7$

Modular Arithmetic Quiz

- Q: What is -3 mod 6?
- A: 3
- Q: What is -1 mod 6?
- A: 5
- Q: What is 5^{-1} mod 6?
- A: 5
- Q: What is 2^{-1} mod 6?
- A: No number works!
- Multiplicative inverse might not exist

Relative Primality

- x and y are **relatively prime** if they have no common factor other than 1
- $x^{-1} \bmod y$ exists only when x and y are relatively prime
- If it exists, $x^{-1} \bmod y$ is easy to compute using Euclidean Algorithm
  - We won't do the computation here

Totient Function

- $\varphi(n)$ is "the number of numbers less than n that are relatively prime to n"
  - Here, "numbers" are positive integers
- Examples
  - $\varphi(4) = 2$ since 4 is relatively prime to 3 and 1
  - $\varphi(5) = 4$ since 5 is relatively prime to 1,2,3,4
  - $\varphi(12) = 4$
  - $\varphi(p) = p-1$ if p is prime
  - $\varphi(pq) = (p-1)(q-1)$ if p and q prime

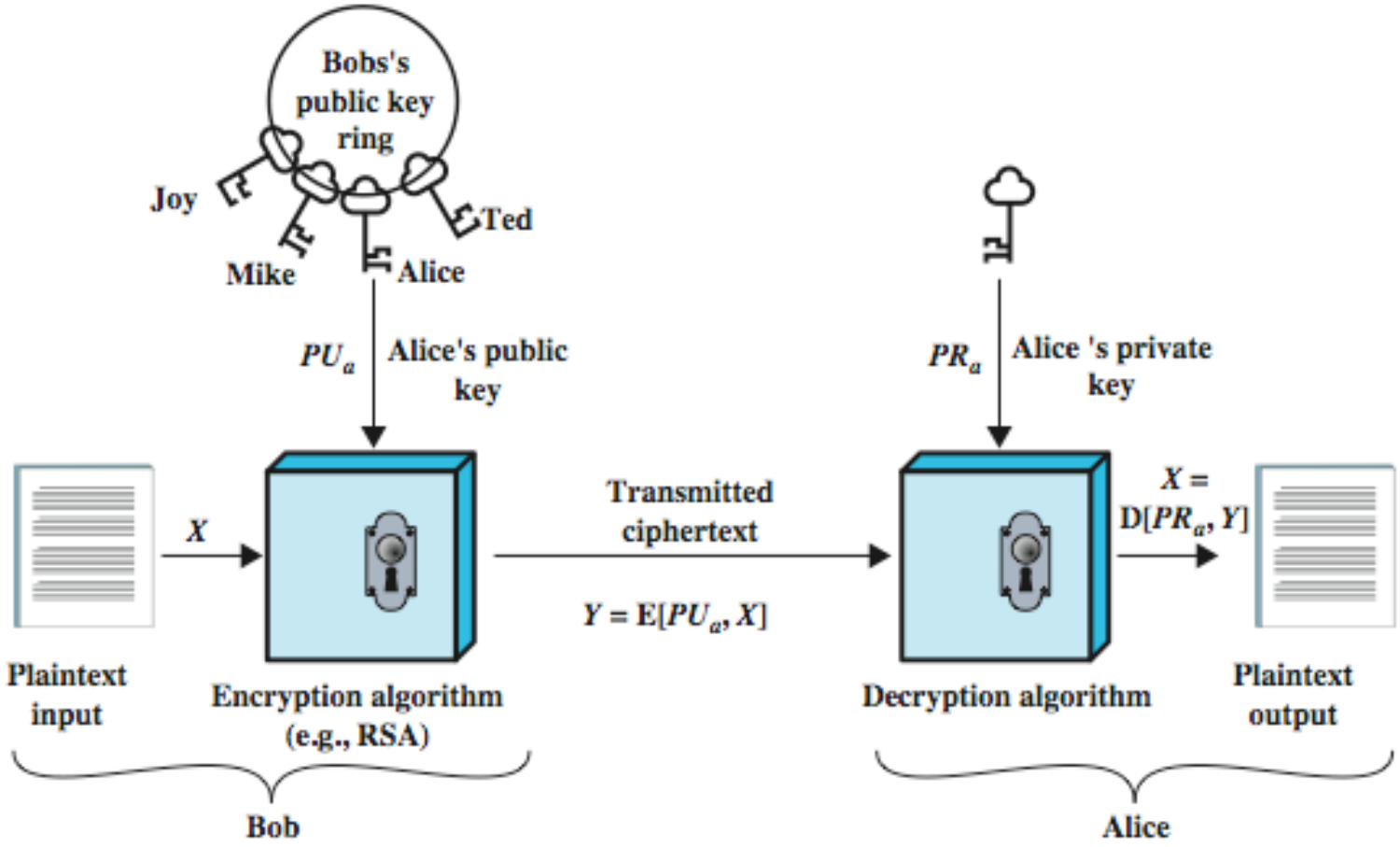
Public-Key Cryptography – General Characteristics - 1

- public-key/two-key/asymmetric cryptography
  - A concept, there are several such cryptosystems
- probably the only revolution in the 3000 years of history of cryptography
- uses 2 keys
  - public-key
    - may be known by anybody, and can be used to encrypt messages, and verify signatures
  - private-key
    - known only to the owner, used to decrypt messages, and sign (create) signatures

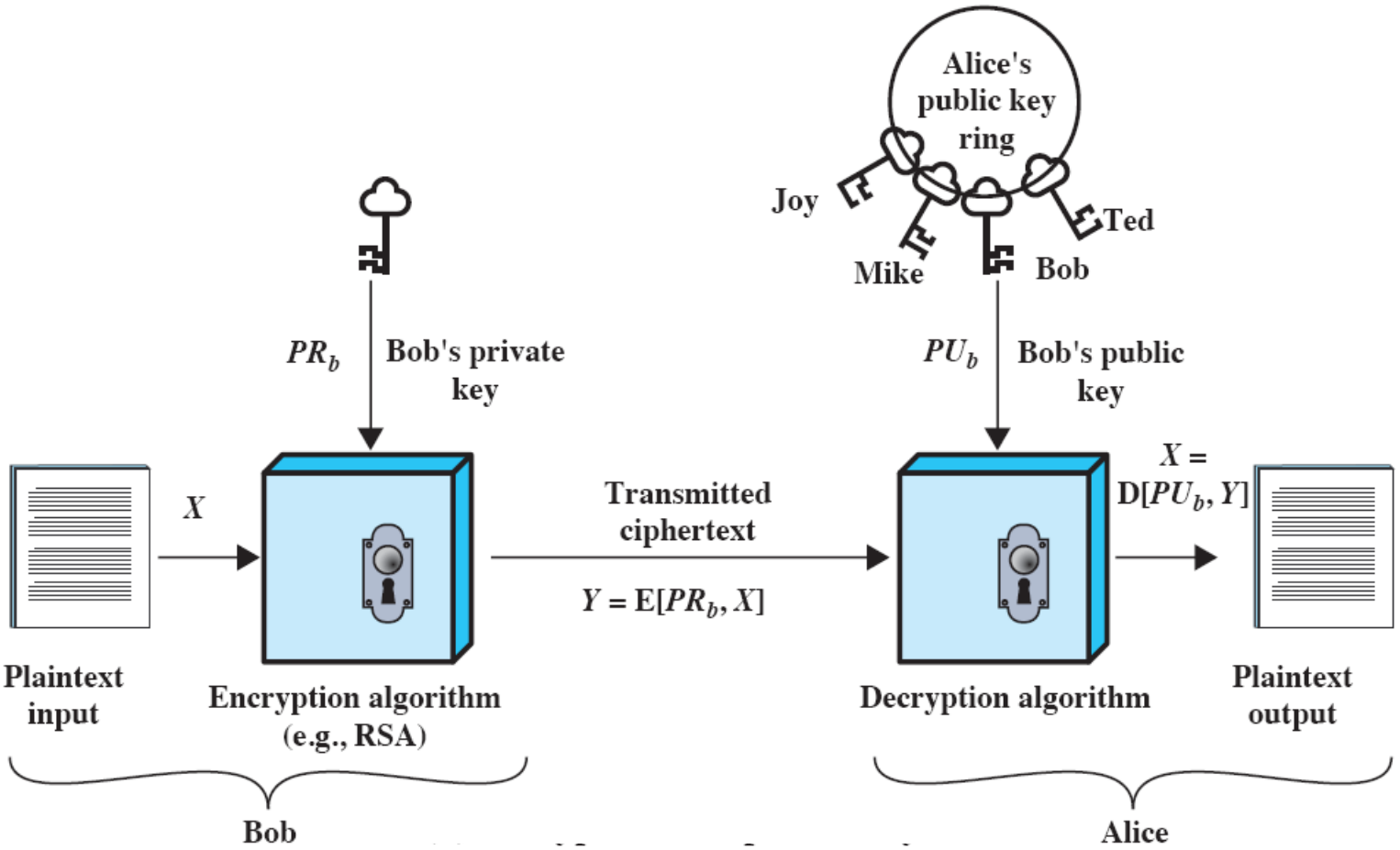
Public-Key Cryptography – General Characteristics - 2

- Keys are related to each other but it is not feasible to find out private key from the public one

Public-Key Cryptography - Encryption



Public-Key Cryptography - Authentication



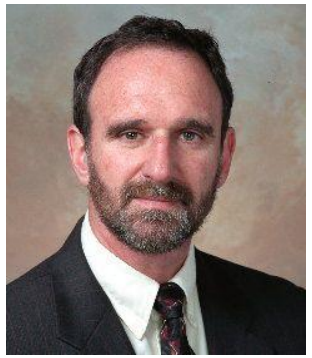
Public-Key Cryptography – General Characteristics

- based on **number theoretic hard problems**
  - rather than substitutions and permutations
- 3 misconceptions about PKC
  - it replaces symmetric crypto
    - PKC rather complements private key crypto
  - PKC is more secure
    - no evidence for that, security mostly depends on the key size in both schemes
  - key distribution is trivial in PKC since public keys are public
    - making something public is not easy. How can you make sure that a public key belongs to the intended person?
    - key distribution is easier, but not trivial

Invention of PKC

- PKC is invented by Whitfield Diffie and Martin Hellman in 1976
  - PhD student – advisor pair at Stanford Univ.
- Some gives credit to Ralph Merkle too
- NSA says that they knew PKC back in 60's
- First documented introduction of PKC is by James Ellis of UK's CESG (Communications-Electronics Security Group) in 1970
  - was a classified report
  - declassified in 1987

Whitfield Diffie

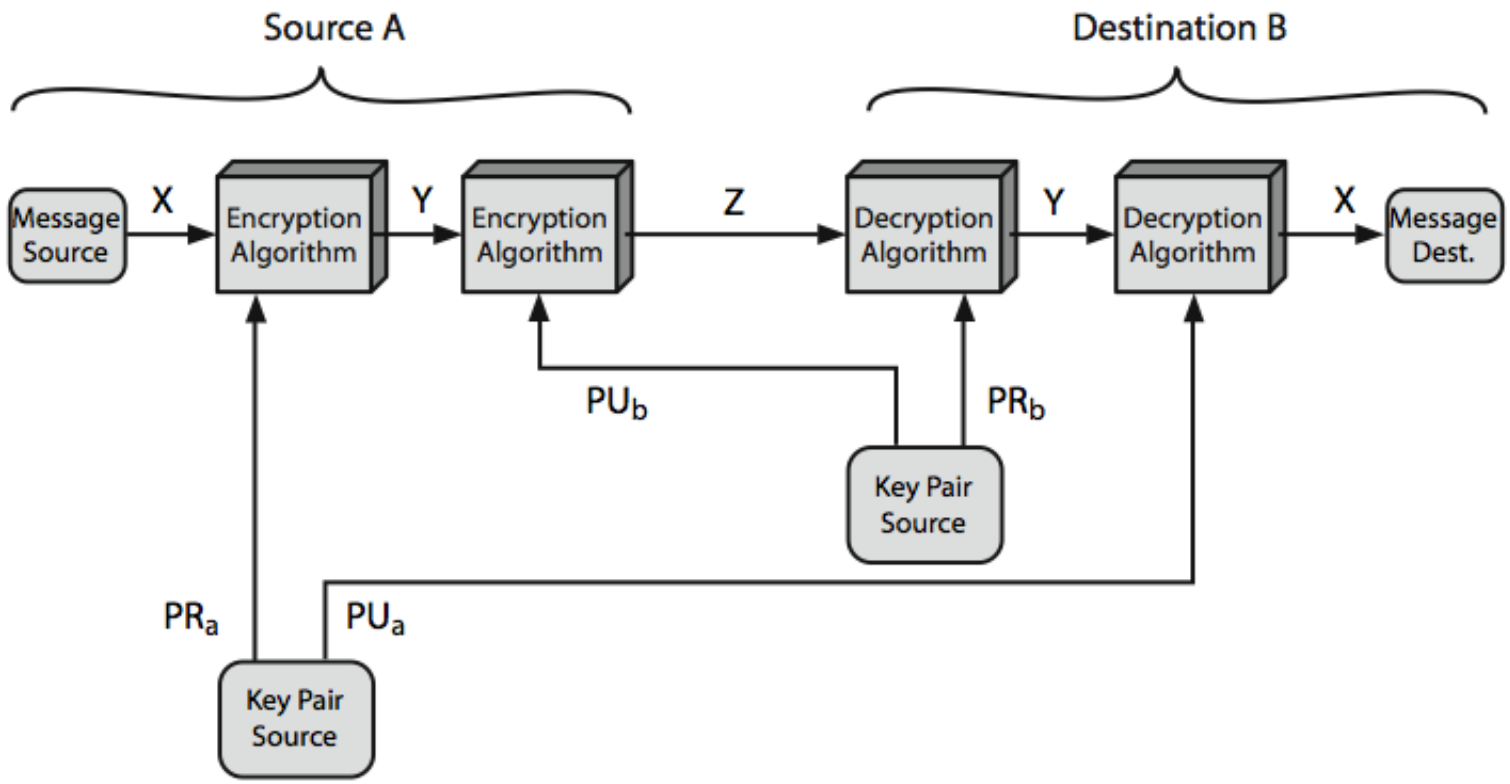


Martin Hellman

Why Public-Key Cryptography?

- Initially developed to address two key issues:
  - key distribution
    - symmetric crypto requires a trusted Key Distribution Center (KDC)
    - in PKC you do not need a KDC to distribute secret keys, but you still need trusted third parties
  - digital signatures (non-repudiation)
    - not possible with symmetric crypto

Public-Key Cryptosystems



PU_a	A's Public Key	PU_b	B's Public Key
PR_a	A's Private Key	PR_b	B's Private Key

Applications of Public- Key Cryptosystems

- 3 categories
  - encryption/decryption
    - to provide secrecy
  - digital signatures
    - to provide authentication and non-repudiation
  - key exchange
    - to agree on a session key
- 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

Some Issues of Public Key Schemes

- like private key schemes brute force attack is always theoretically possible
  - use large keys
  - consider the security vs. performance tradeoff
- due to public key / private key relationships, number of bits in the key should be much larger than symmetric crypto keys
  - to make the hard problem really hard
  - 80-bit symmetric key and 1024-bit RSA key has comparable resistance to cryptanalysis
- a consequence of use of large keys is having slower encryption and decryption as compared to private key schemes
  - thus, PKC is not a proper method for bulk encryption

RSA

- by Rivest, Shamir & Adleman of MIT in 1977
  - published in 1978
- best known and widely used public-key scheme
- was patented and patent was used by RSA Inc
  - however patent expired in 2000
- uses large integers
  - 1024+ bits
- security depends on the cost of factoring large numbers

Rivest Shamir Adleman

RSA Key Setup

Key Generation by Alice

Select p, q	p and q both prime, $p \neq q$
Calculate $n = p \times q$	
Calculate $\phi(n) = (p - 1)(q - 1)$	
Select integer e	$\gcd(\phi(n), e) = 1;\ \ 1 < e < \phi(n)$
Calculate d	$d \equiv e^{-1} \pmod{\phi(n)}$
Public key	$PU = \{e, n\}$
Private key	$PR = \{d, n\}$

e is usually a small number

RSA Use

- to encrypt a message M < n, the sender:
  - obtains public key of recipient $PU=\{e,n\}$
  - computes: $C=M^e \bmod n$, where $0 \leq M < n$

Encryption by Bob with Alice's Public Key

Plaintext:	$M < n$
Ciphertext:	$C = M^e \bmod n$

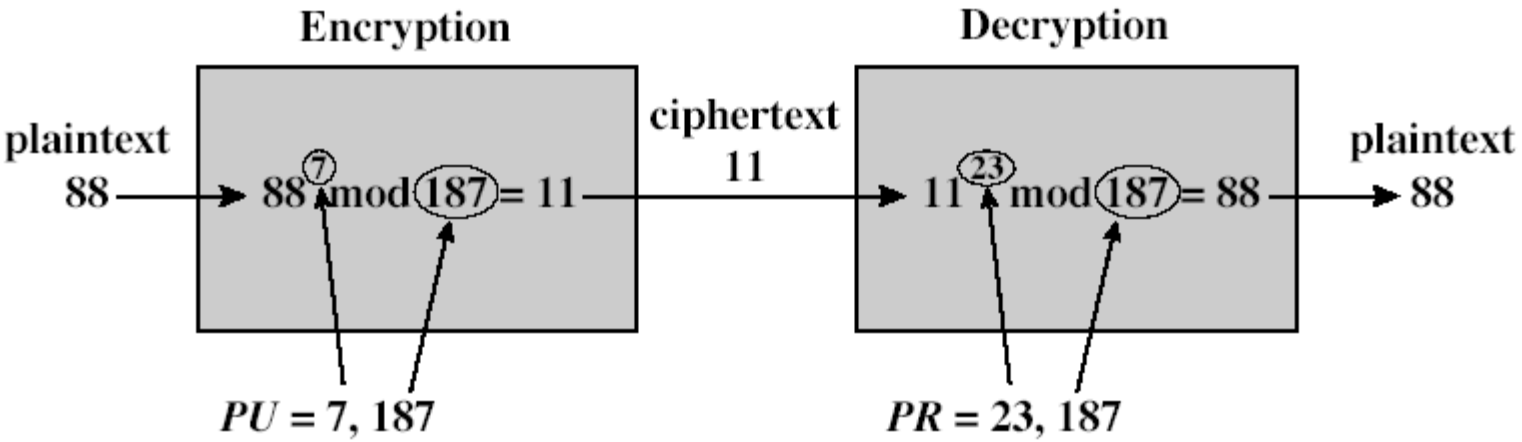
- to decrypt the ciphertext C the owner:
  - uses their private key $PR=\{d,n\}$
  - computes: $M=C^d \bmod n$

Decryption by Alice with Alice's Private Key

Ciphertext:	C
Plaintext:	$M = C^d \bmod n$

- note that the message M must be smaller than the modulus n: use several blocks if needed

RSA Example



p = 17, q = 11, n = p*q= 187

Φ(n) = 16*10 =160, pick e=7, d.e=1 mod Φ(n) ➔ d = 23

Why RSA Works

- because of Euler's Theorem:

  $a^{\phi(n)} \bmod n = 1$ where $\gcd(a,n)=1$

- in RSA have
  - $n=p.q$
  - $\phi(n)=(p-1)(q-1)$
  - carefully chose e & d to be inverses $\bmod \phi(n)$
    - i.e. $e.d = 1 \bmod \phi(n)$
  - hence $e.d=1+k.\phi(n)$ for some k
- hence

$$C^d = M^{e.d} = M^{1+k.\phi(n)} = M^1.(M^{\phi(n)})^k$$
$$= M^1.(1)^k = M^1 = M \bmod n$$

Computational Aspects

- An RSA implementation requires complex arithmetic
  - modular exponentiation for encryption and encryption
  - primality tests
  - finding inverse of e mod $\Phi(n)$
- There are acceptably fast solutions to those computational problems (see Stallings for details)

RSA Security

- 4 approaches of attacking on RSA
  - brute force key search
    - not feasible for large keys
    - actually nobody attacks on RSA in that way
  - mathematical attacks
    - based on difficulty of factorization for large numbers as we shall see in the next slide
  - side-channel attacks
    - based on running time and other implementation aspects of decryption

  - chosen-ciphertext attack
    - Some algorithmic characteristics of RSA can be exploited to get information for cryptanalysis

Factorization Problem

- 3 forms of mathematical attacks
  - factor $n=p.q$, hence find $\varnothing(n)$ and then d
  - determine $\varnothing(n)$ directly and find d
    - is equivalent of factoring n
  - find d directly
    - as difficult as factoring n
- so RSA cryptanalysis is focused on factorization of large n

Factorization Problem – RSA Challenges

Number of Decimal Digits	Approximate Number of Bits	Date Achieved	MIPS-years	Algorithm
100	332	April 1991	7	quadratic sieve
110	365	April 1992	75	quadratic sieve
120	398	June 1993	830	quadratic sieve
129	428	April 1994	5000	quadratic sieve
130	431	April 1996	1000	generalized number field sieve
140	465	February 1999	2000	generalized number field sieve
155	512	August 1999	8000	generalized number field sieve
160	530	April 2003	—	Lattice sieve
174	576	December 2003	—	Lattice sieve
200	663	May 2005	—	Lattice sieve

193	640	November 2005
232	768	December 2009
212	704	July 2012
220	729	May 2016

- RSA-129 was a challenge by RSA inventors
  - 1977, reward is $100
  - they estimated 40 quadrillion ($40*10^{15}$) years
  - solved in 1993/4 in 8 months (Atkins, Graff, Lenstra and Leyland + 600 volunteers worldwide)
  - A group of computers (1600) over the Internet used their spare time

Reasons of improvement in Factorization

- increase in computational power
- biggest improvement comes from improved algorithm
  - "Quadratic Sieve" to "Generalized Number Field Sieve"
  - Then to "Lattice Sieve"

(Latest-4) RSA challenge factored

- RSA-576 (174 decimal digits)
- Mostly German team
  - December 2003
- First of the RSA challenge numbers to be factored from the "new" challenge started in 2001
- ~13200 MIPS-years

http://mathworld.wolfram.com/news/2003-12-05/rsa/

(Latest-4) RSA challenge factored

- RSA-200
  - May 2005
  - One of the old challenges
  - Bit equivalent is 663
    - Was the largest RSA challenge number factored until December 2009
  - The team is F. Bahr, M. Boehm, J. Franke, and T. Kleinjung

https://en.wikipedia.org/wiki/RSA_Factoring_Challenge

(Latest-3) RSA challenge factored

- RSA 640
  - November 2005
  - 2nd challenge of the new set
    - Prize USD 20K
  - Same team as RSA-200
  - Smaller number than RSA 200
  - Reported computation effort is half of the RSA-200

(Latest-2) RSA challenge factored

- RSA 768
  - Largest RSA challenge factored so far
  - December 2009
  - 4th challenge of the new set
    - No prize since RSA discontinued RSA challenge (prize was $ 50,000)
    - 3rd challenge (RSA 704) was skipped (later solved)
  - A multinational and multi-institutional team led by Thorsten Kleinjung
  - Reported computational effort is 2000 2.2GHz-Opteron-CPU years (~66 times more than RSA-640)

(Latest-1) RSA challenge factored

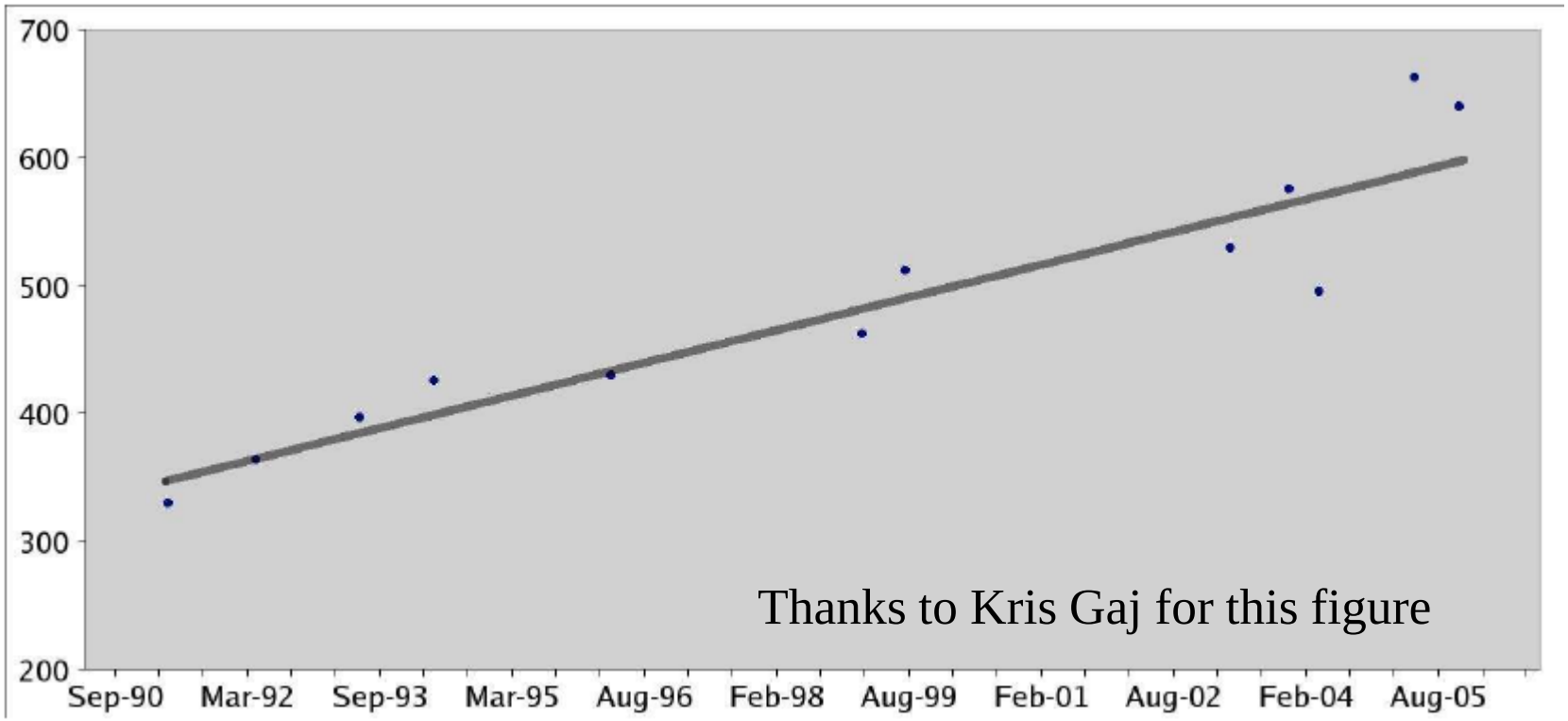
- RSA 704
  - July 2012
  - Third challenge of the new set (cash prize was $30000, but could not be received)
    - Smaller than previously solved one
  - Shi Bai, Emmanuel Thomé and Paul Zimmermann
  - Details are at http://eprint.iacr.org/2012/369.pdf

Latest RSA challenge factored

- RSA 220
  - May 2016
  - One of the old challenges; Bit equivalent is 729
  - S. Bai, P. Gaudry, A. Kruppa, E. Thomé, P. Zimmermann
  - GNFS based solution using open-source software (CADO_NFS)
  - Details are at https://members.loria.fr/PZimmermann/papers/rsa220.pdf
- Some smaller RSA challenges from the old set were solved in 2010 and beyond (not mentioned here)
- Next RSA challenge is 896-bit (prize $ 75,000)
  - RSA Labs discontinued RSA challenge in 2007, so if you factorize these numbers, you'll get no money!

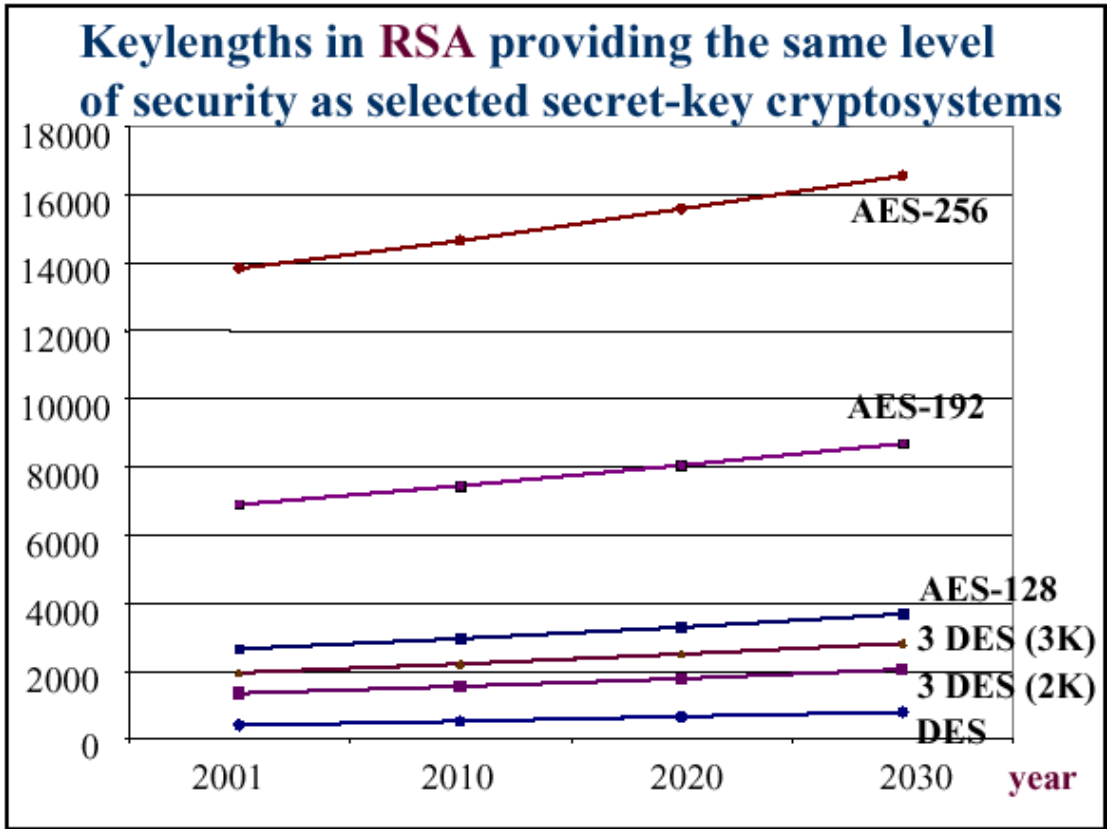
An estimate for RSA-1024 Factorization

- Linear regression-based
- Around 2028



Side Channel Attacks

- For example timing attacks
  - based on timing variations in operations
  - some operations are slow, some faster depending on the key
- In RSA, there are time variations in exponentiation during decryption
- countermeasures
  - use constant exponentiation time
  - add random delays
  - blinding (offered by RSA Inc.)
    - multiply the ciphertext by a random value so that attacker cannot know the ciptertext being decrypted
    - let’s see on the board



Thanks to Kris Gaj for this figure

Key size and Security

Validity period	Minimal RSA key length (bits)	Equivalent symmetric key length (bits)
2003-2010	1024	80
2010-2030	2048	112
2030-	3072	128

Recommendations of RSA Security Inc.
May 6, 2003

Five security levels

- NIST SP 800-56

Level	RSA / DH	ECC	Symmetric ciphers
1	1024	160	80
2	2048	224	112
3	3072	256	128
4	8192	384	192
5	15360	512	256

Summary

- Introduction to number theory
- Public-Key Cryptography – General Characteristics
- Public-Key Cryptography - Encryption
- Public-Key Cryptography - Authentication
- Some Issues of Public Key Schemes
- Public-Key cryptosystem : RSA
- RSA Security