

(1.8) Cryptanalysis:

The science of deducing the plaintext from a ciphertext, without knowledge of the key.

(1.8.1) Attacks on encrypted messages

The objective of the following attacks is to systematically recover plaintext from ciphertext, or even more drastically, to deduce the decryption key.

1. **A ciphertext-only attack:** is one where the adversary (or cryptanalyst) tries to deduce the decryption key or plaintext by only observing ciphertext. Any encryption scheme vulnerable to this type of attack is considered to be completely insecure.
2. **A known-plaintext attack:** is one where the adversary has a quantity of plaintext and corresponding ciphertext. This type of attack is typically only marginally more difficult to mount.
3. **A chosen-plaintext attack:** is one where the adversary chooses plaintext and is then given corresponding ciphertext. Subsequently, the adversary uses any information deduced in order to recover plaintext corresponding to previously unseen ciphertext.

4. **An adaptive chosen-plaintext attack:** is a chosen-plaintext attack wherein the choice of plaintext may depend on the ciphertext received from previous requests.
5. **A chosen-ciphertext attack:** is one where the adversary selects the ciphertext and is then given the corresponding plaintext. One way to mount such an attack is for the adversary to gain access to the equipment used for decryption (but not the decryption key, which may be securely embedded in the equipment). The objective is then to be able, without access to such equipment, to deduce the plaintext from (different) ciphertext.
6. **An adaptive chosen-ciphertext attack:** is a chosen-ciphertext attack where the choice of ciphertext may depend on the plaintext received from previous requests.

Some concepts on cryptanalysis:

- ✦ **Frequency:** number of appearance of the letter in the ciphertext, where the frequencies of the ciphertext letters are compared with the frequencies in Table 1 or Figure 5.
- ✦ **Repetition:** is the similar parts in the ciphertext that have length not less than three. This helps us to find the length of the key (the number of alphabets that used to enciphering in the polyalphabetic systems).
Take the Highest Common Factor HCF between the repetitions, which represent the length of the key, this method, is called the Kasiski method.

Letter	%	Letter	%
a	8.167	n	6.749
b	1.492	o	7.507
c	2.782	p	1.929
d	4.253	q	0.095
e	12.702	r	5.987
f	2.228	s	6.327
g	2.015	t	9.056
h	6.094	u	2.758
i	6.966	v	0.978
j	0.153	w	2.360
k	0.772	x	0.150
l	4.025	y	1.974
m	2.406	z	0.074

Table 1: English letters frequencies
English Letter Frequency

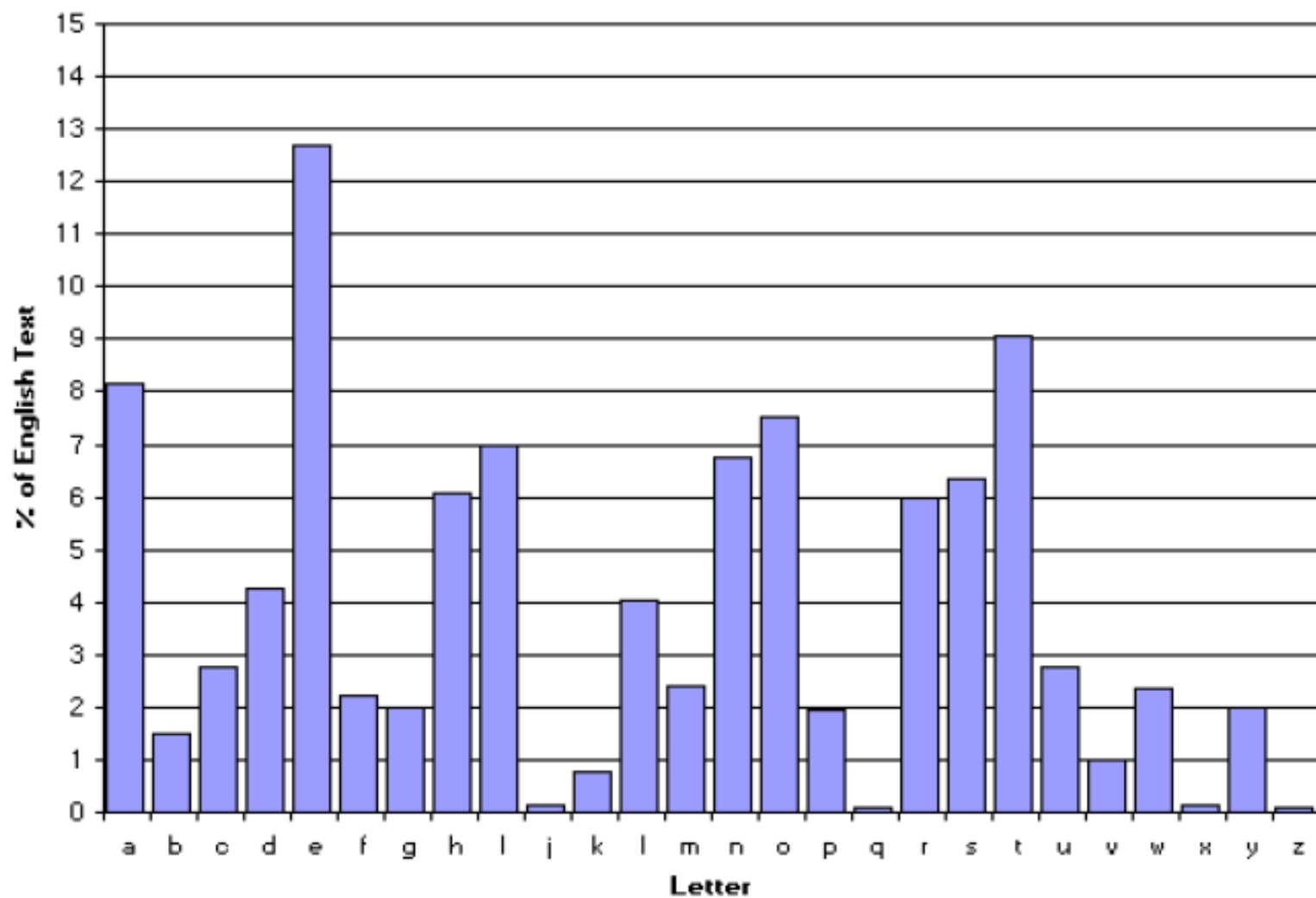


Figure 5: Histogram of English letters frequencies

- ✦ **Index of Coincidence (IC):** is the probability that two letters selected from the text are identical, we can compute the IC from the following equation:

$$IC = \frac{\sum_{\lambda} f_{\lambda}(f_{\lambda} - 1)}{n(n - 1)},$$

where f_{λ} is the frequency of the letter λ in the ciphertext and n is the length of the letter. The IC value differs from language to another. We can use the IC to discover if the message were enciphered using Monoalphabetic system or polyalphabetic system.

- ✦ **Coincidence:** is the computing of the coincidence of the ciphertexts, where two messages put one over the other, and the purpose is to discover if the two messages were enciphered using the same key. If there is 7 coincidence letters between 100 letters in the two messages then the two messages were enciphered using the same key, while if there is 4 letters coincidence between every 100 letters then they enciphered with different keys.

Cryptanalysis examples:

First of all we must specify the type of the cipher system that was used. If the frequencies of the ciphertext are the same as the frequencies of the language then, a transposition cipher system was used; otherwise a substitution cipher system was used.

1- Cryptanalysis of transposition cipher systems:

When we decide that a transposition cipher system were used, we put the cipher text in $m \times n$ matrix, m and n depends on the length of the received ciphertext, for example if the length is 500 then one of the possible sizes is 20×25 . Then we rearrange the columns to get some known patterns such as (and, the, ion, that,...) in addition to some expected word in the message.

As we know there are two types of transposition cipher system: simple and double transposition, the cryptanalysis of the last one is more complicated because we lose the ability to find the known patterns.

2- Cryptanalysis of substitution cipher systems:

If we know that a substitution cipher system was used, the next step is to determine whether a monoalphabetic system or polyalphabetic system was used, by using the IC of the language.

Example: A sample of ordinary English contains the following distribution of letters

Letter	Count	Letter	count
A	141	N	119
B	36	O	132
C	36	P	28
D	103	Q	1
E	188	R	95
F	37	S	64
G	34	T	182
H	102	U	59
I	123	V	13
J	4	W	55
K	18	X	3
L	56	Y	23
M	27	Z	0

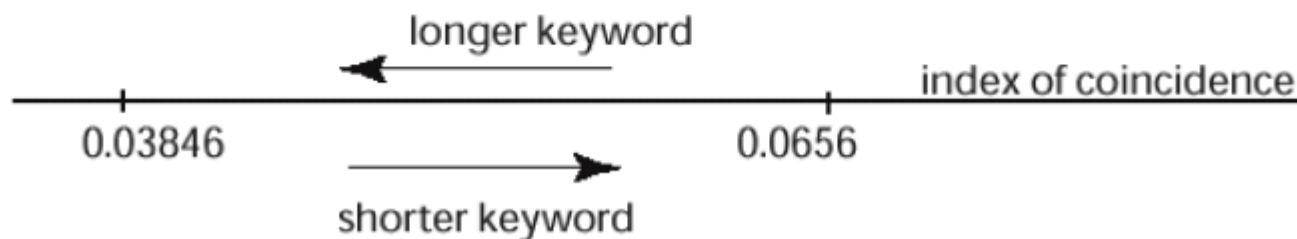
What is the probability of selecting an identical pair of letters from this collection? in other word compute the IC.

$$IC = \frac{\sum_A f_{\lambda}(f_{\lambda} - 1)}{n(n - 1)}$$

$$IC = \frac{141(141 - 1) + 36(36 - 1) + \dots + 23(23 - 1) + 0(0 - 1)}{1679(1679 - 1)} = \frac{184838}{2817362} \approx 0.0656.$$

As we see from the two examples above the index of coincidence of totally random (uniformly distributed) collection of letters is about 0.0385. Vigenere ciphertexts from longer keywords have a more uniform distribution of letters. For keyword length closer to 1, the index of coincidence will be larger, closer to 0.0656.

Polyalphabeticity Measure for English



If the length of the text is n , we can quantify the connection between index of coincidence and keyword length k , (number of alphabets), where:

$$k \approx \frac{0.0265 \cdot n}{(0.065 - IC) + n(IC - 0.0385)}$$

Example: A polyalphabetic ciphertext has the following letter counts.

Letter	Count	Letter	count
A	60	N	28
B	50	O	83
C	42	P	44
D	64	Q	69
E	51	R	13
F	63	S	29
G	19	T	66
H	48	U	87
I	56	V	63
J	67	W	19
K	23	X	43
L	45	Y	39
M	44	Z	67

Estimate the keyword length.

Solution: There are $n=1282$ letters.

$$IC = \frac{60 \cdot 59 + 50 \cdot 49 + \dots + 67 \cdot 66}{1282 \cdot 1281} = \frac{35761}{821121} \approx 0.04355.$$

$$K = \frac{0.0265 \cdot 1282}{(0.065 - 0.04355) + 1282(0.04355 - 0.03846)} = 5.1892.$$

Based only on this evidence, a reasonably likely keyword length is 5.

➤ Now, after the above tests if we conclude that a monoalphabetic cipher system was used, then:

❶ If a direct standard or reversed system were used, we compare the frequencies of the ciphertext with the frequencies of the English language, start by putting E against the letter with the higher frequency in the ciphertext, then we put the other letters sequentially.

❷ If a mixed cipher system was used (Random) then we compare the frequencies of the ciphertext with that in Table 1 and Figure 5.

For advanced analysis we can use in addition to Table 1, a table of double letter frequencies TH, HE, IN, ER, RE, ON, AN, EN,..., and triple letter frequencies THE, AND, TIO, ATI, FOR, THA, TER, RES,... and so on.

➤ If a polyalphabetic cipher system was used then we will use the Kasiski method to find the length of the key k (number of alphabets). Then we divide the ciphertext into k parts, each part will analyze as in ❷ above.

2. Euclidean Algorithm:

If have two numbers c, q that $c = q * d + r$, then $GCD(c, q) = GCD(d, r)$

Ex1: find the Greatest Common Divisor (GCD) between 132 and 55 by using Euclid's Algorithm.

$$132 = 55 * 2 + 22$$

$$55 = 22 * 2 + 11$$

$$22 = 11 * 2 + 0$$

Stopping when getting zero 0 then GCD is 11:

$$GCD(132, 55) = GCD(55, 22) = GCD(22, 11) = GCD(11, 0) = 11$$

Ex2: find the GCD (252 , 198) by using Euclid's Algorithm.

$$252 = 198 * 1 + 54$$

$$198 = 54 * 3 + 36$$

$$54 = 36 * 1 + 18$$

$$36 = 18 * 2 + 0$$

$$GCD(252, 198) = (198, 54) = (54, 36) = (36, 18) = (18, 0) = 18$$

Public Key Cryptography and the RSA Algorithm

Private-Key Cryptography

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

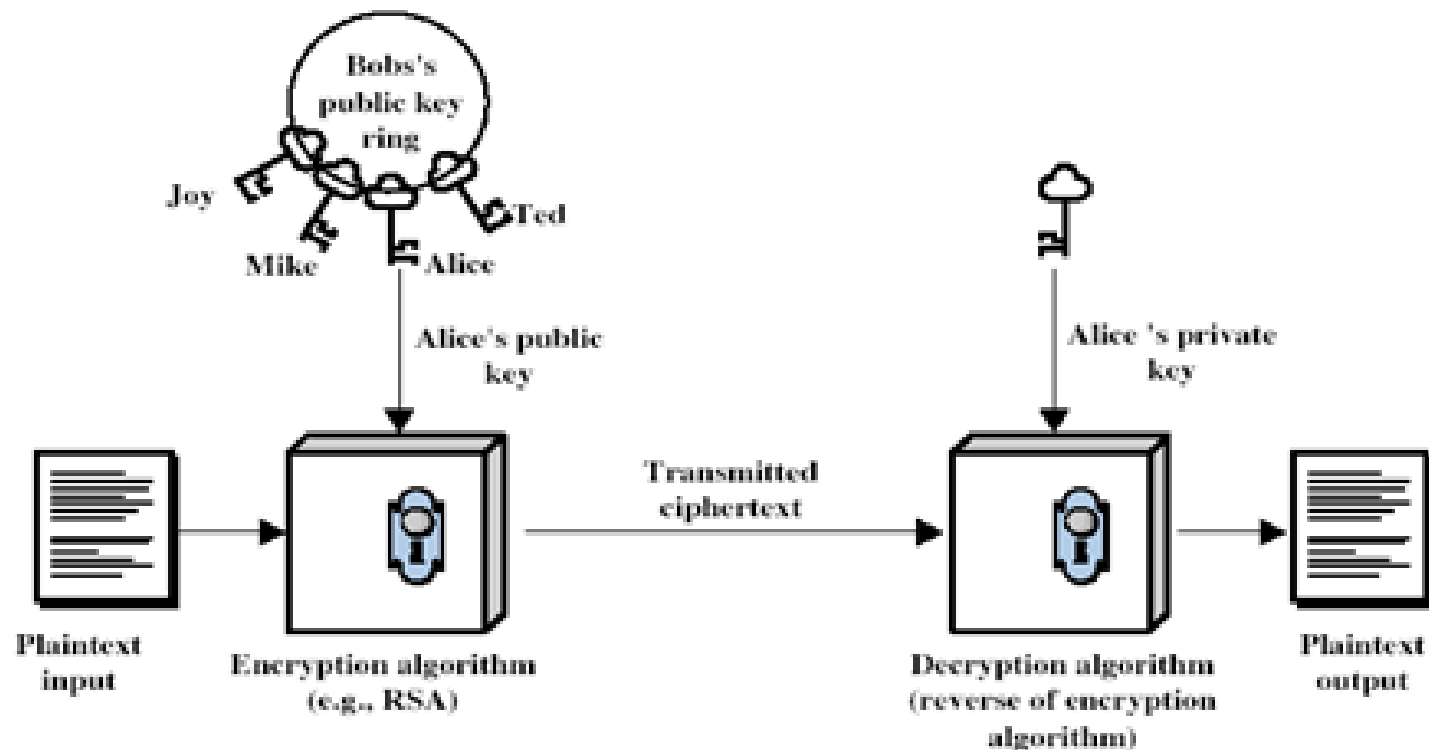
Public-Key Cryptography

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

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 **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?

- 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 U. in 1976
 - known earlier in classified community

Public-Key Characteristics

- 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)

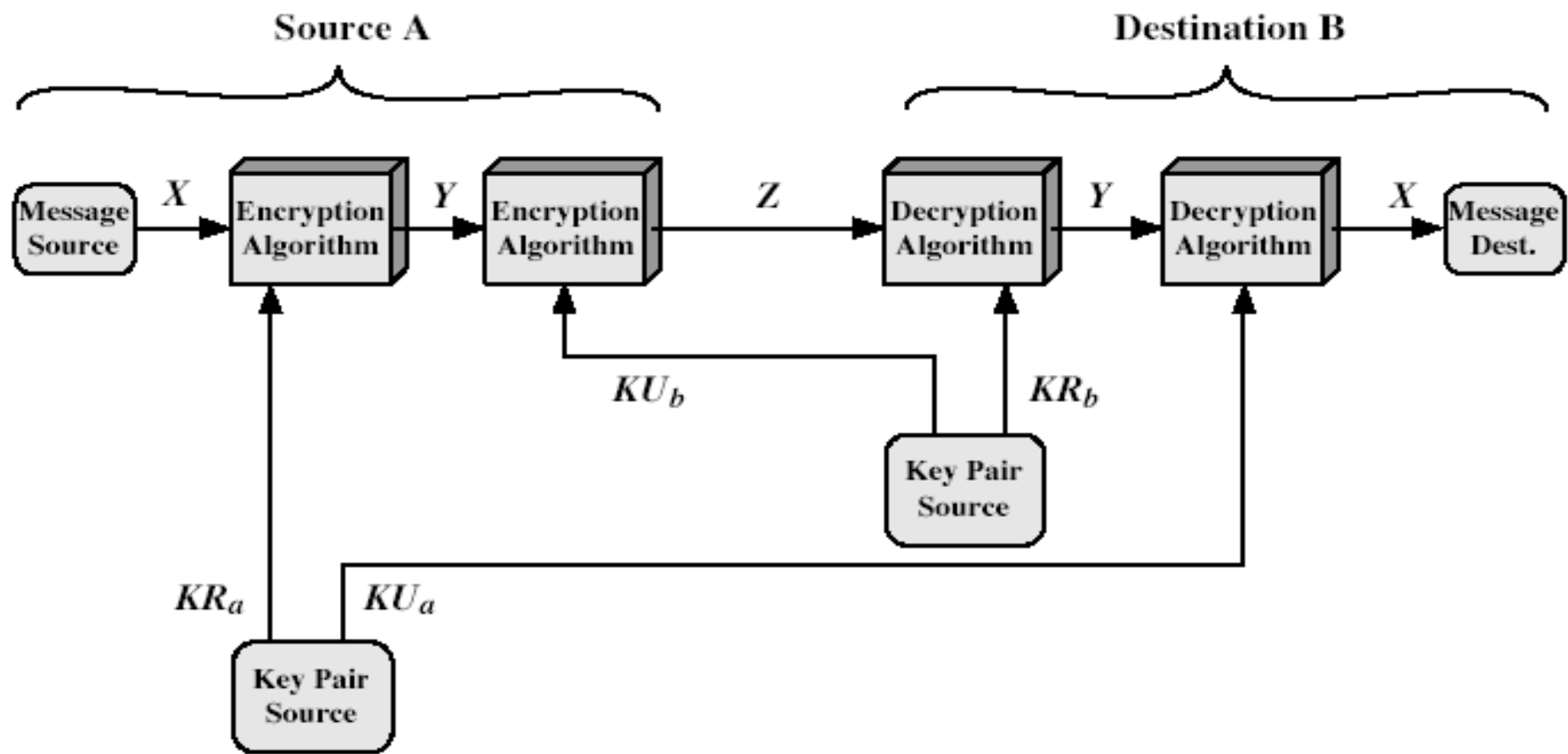


Figure 9.4 Public-Key Cryptosystem: Secrecy and Authentication

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 (>512 bits)
- 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
- 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 Key Setup

- each user generates a public/private key pair by:
- selecting two large primes at random - p, q
- computing their system modulus $N=p \cdot q$
 - note $\phi(N) = (p-1)(q-1)$
- selecting at random the encryption key e
 - where $1 < e < \phi(N)$, $\gcd(e, \phi(N)) = 1$
- solve following equation to find decryption key d
 - $e \cdot d = 1 \bmod \phi(N)$ and $0 \leq d \leq N$
- 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 $0 \leq M < N$
- 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

- because of Euler's Theorem:
- $a^{\varphi(N)} \bmod N = 1$
 - where $\gcd(a, N) = 1$
- in RSA have:
 - $N = p \cdot q$
 - $\varphi(N) = (p-1)(q-1)$
 - carefully chosen e & d to be inverses mod $\varphi(N)$
 - hence $e \cdot d = 1 + k \cdot \varphi(N)$ for some k
- hence :
$$\begin{aligned} C^d &= (M^e)^d = M^{1+k \cdot \varphi(N)} = M^1 \cdot (M^{\varphi(N)})^k = M^1 \cdot (1)^k \\ &= M^1 = M \bmod N \end{aligned}$$

RSA Example

1. Select primes: $p=17$ & $q=11$
2. Compute $n = \underline{pq} = 17 \times 11 = 187$
3. Compute $\phi(n) = (p-1)(q-1) = 16 \times 10 = 160$
4. Select e : $\underline{\gcd}(e, 160) = 1$; choose $e=7$
5. Determine d : $de = 1 \pmod{160}$ and $d < 160$
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

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

Exponentiation

- can use the Square and Multiply Algorithm
- 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. $7^5 = 7^4 \cdot 7^1 = 3 \cdot 7 = 10 \pmod{11}$
 - eg. $3^{129} = 3^{128} \cdot 3^1 = 5 \cdot 3 = 4 \pmod{11}$

$c \leftarrow 0; d \leftarrow 1$

for $i \leftarrow k$ **downto** 0

do $c \leftarrow 2 \times c$

$d \leftarrow (d \times d) \bmod n$

if $b_i = 1$

then $c \leftarrow c + 1$

$d \leftarrow (d \times a) \bmod n$

return d