CMPT-335 Discrete Structures

THE INTEGERS AND DIVISION. INTRO TO CRYPTOGRAPHY

Greatest Common Divisor

- Given integers m and n, not both zero, we define the greatest common divisor of m and n to be the largest integer that divides both m and n: GCD(m, n).
- Note that if $m \neq 0$, then GCD(m, 0) = $|m|$ by definition.

Greatest Common Divisor: Euclidian Algorithm

- Determine the greatest common divisor (GCD) for two numbers.
- Euclidean algorithm: GCD (*m,n*) can be recursively found from the formula

$$GCD(m,n) = \begin{cases} m & \text{if } n=0 \\ n & \text{if } m=0 \\ GCD(n, m \bmod n) & \text{otherwise} \end{cases}$$

- Theorem. Let *m*, *n*, *r*, and *q* be integers with *n*>0. If *m*=*qn*+*r*, then GCD(*m*, *n*) = GCD(*n*, *r*).

Euclidean Algorithm. Example

$$GCD(m,n) = \begin{cases} m & \text{if } n=0 \\ n & \text{if } m=0 \\ GCD(n, m \bmod n) & \text{otherwise} \end{cases}$$

GCD(22,77):

Step	$r = m$ **mod** n	m	n
0	–	22	77

Euclidean Algorithm. Example

$$GCD(m,n)=\begin{cases} m & \text{if } n=0 \\ n & \text{if } m=0 \\ GCD(n, m \bmod n) & \text{otherwise} \end{cases}$$

GCD(22,77):

Step	$r = m$ **mod** n	m	n
0	–	22	77
1	22 **mod** 77 = 22	77	22

Euclidean Algorithm. Example

$$GCD(m,n)=\begin{cases} m & \text{if } n=0 \\ n & \text{if } m=0 \\ GCD(n, m \bmod n) & \text{otherwise} \end{cases}$$

GCD(22,77):

Step	$r = m$ **mod** n	m	n
0	–	22	77
1	22 **mod** 77 = 22	77	22
2	77 **mod** 22 = 11	22	11

Euclidean Algorithm. Example

$$GCD(m,n)=\begin{cases} m & \text{if } n=0 \\ n & \text{if } m=0 \\ GCD(n, m \bmod n) & \text{otherwise} \end{cases}$$

GCD(22,77):

Step	$r = m$ **mod** n	m	n
0	–	22	77
1	22 **mod** 77 = 22	77	22
2	77 **mod** 22 = 11	22	11
3	22 **mod** 11 = 0	11	0

The Fundamental Theorem of Arithmetic

- Theorem. Every positive integer greater than 1 can be written uniquely as a prime or as the product of two or more primes where the prime factors are written in order of non-decreasing size.

$$100 = 2 \cdot 2 \cdot 5 \cdot 5 = 2^2 \cdot 5^2$$

$$17 = 17$$

$$999 = 3 \cdot 3 \cdot 3 \cdot 37 = 3^3 \cdot 37$$

$$1024 = 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 = 2^{10}$$

Applications: Simple Encryption

- Variations on the following have been used to encrypt messages for thousands of years, from the Julius Caesar time.

1. Convert a message to capitals.
2. Encode each letter by a number between 0 and 25.
3. Apply an invertible modular encryption function (mod 26) to each number.
4. Convert back to letters.

Letter ←→ Number Conversion Table

A	B	C	D	E	F	G	H	I	J	K	L	M
0	1	2	3	4	5	6	7	8	9	10	11	12

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
13	14	15	16	17	18	19	20	21	22	23	24	25

Encryption example

- Let the encryption function ("secret key") be

$$r = f(p) = (p + 15) \bmod 26$$

- Encrypt "Stop Thief"

1. STOP THIEF (capitals)
2. 18,19,14, 15 19, 7, 8, 4, 5
3. 7, 8, 3, 4 8, 22,23,19,20
4. Cipher obtained: HIDE IWXTU

Decryption example

- Decryption works the same, except that we apply the inverse key function

$$p = g(r) = (r - 15) \bmod 26$$

- Decrypt "Hide Iwxtu"

1. HIDE IWXTU (capitals)
2. 7, 8, 3, 4 8, 22,23,19,20
3. 18,19,14, 15 19, 7, 8, 4, 5
4. Message obtained: STOP THIEF

Applications: Simple Encryption

- To ensure more reliable encryption, the encryption function (encrypting key) in the simple encryption method should have a form

$$r = f(p) = (ap + b) \bmod 26$$

- Respectively, the decryption function (decrypting key) has a form

$$p = g(r) = f^{-1}(r) = a^{-1}(r - b) \bmod 26$$

Encryption example

- Let the encryption function ("secret key") be

$$r = f(p) = (3p + 9) \bmod 26$$

- Encrypt "Stop Thief"

1. STOP THIEF (capitals)
2. 18,19,14, 15 19, 7, 8, 4, 5
3. 11,14,25, 2 14, 4, 7, 21, 24
4. Cipher obtained: LOZC OEHVY

Decryption example

- Decryption works the same, except that we apply the inverse key function.
- Find the inverse of

$$r = f(p) = (3p + 9) \bmod 26$$

➢ The inverse is

$$p = g(r) = 3^{-1}(r - 9) \bmod 26 = (r - 9)/3 \bmod 26$$

➢ $(1/3) \bmod 26 = 9$ because from $3x=1 \bmod 26$ it follows that $3x=26q+1$. We should find the smallest q such that $26q+1$ is divisible by 3. Then $q=1$ and $3x=26+1=27$. Hence, $x=27/3=3^{-1} \pmod{26} = 9$.

➢ Thus:

$$g(r) = 9(r - 9) \bmod 26 = (9r-81) \bmod 26 = (9r - 3) \bmod 26$$

Decryption example

- Decryption works the same, except that we apply the inverse key function

$$p = g\,(r) = (9r - 3) \bmod 26$$

- Decrypt "Lozc Oehvy"

1. LOZC OEHVY (capitals)
2. 11,14,25, 2 14, 4, 7, 21, 24
3. 18,19,14, 15 19, 7, 8, 4, 5
4. Message obtained: STOP THIEF

Open key Encryption

- Encryption methods, which are based on the “secret key” function have one, but significant disadvantage: it is necessary to distribute a key among all users and all of them must take care of keeping the key unavailable to others.
- The alternative is an “open key” encryption, which became very popular for the last 20 years. In this method, which is based on the modular arithmetic, two different keys are used for encryption and decryption. The encrypting key is a “public key”, it is open, while the decrypting key is hidden.

RSA Encryption

- In "open key" encryption methods, it is extremely hard (actually, it is impossible within some reasonable long time interval) to reverse a key, which is used for the encryption.
- One of the most popular, efficient (and simple!) modern "open key" encryption methods is the RSA method named after its inventors (Rivest, Shamir, Adleman).

RSA Encryption

- In the RSA encryption method, a message is converted into numbers (similarly to the simple modular encryption – each symbol has its fixed numerical code)
- The letters are then put together into number blocks ***b*** with each block less than *n* (the block, which is considered as a number, is compared to *n* in terms of the regular < relation on the set *Z*).
- Then each number block ***b*** is exponentiated by the exponent ***e*** in Z_n , which completes the encryption procedure: $b^e \bmod n$

Modular Exponentiation

- Modular exponentiation is a key operation in the RSA encryption. It raises a number to a power e in Z_n:

$$a \equiv b^e \pmod{n} \Rightarrow a \in \{0, 1, \ldots, n-1\}$$

- In the RSA method, both e and n are very large numbers. In the real world applications, n is an integer of about 400 decimal digits.

Modular Exponentiation

- To perform modular exponentiation, congruence properties are used.
- To reduce computations, exponent *e* should be presented as a sum of powers of 2 (any integer number should be easily presented in such a way: $e = e_1 + e_2 + \ldots + e_k; e_i = 2^s$)
- Then the following technique should be applied: $b^e \bmod n = b^{e_1 + e_2 + \ldots + e_k} \bmod n = b^{e_1} b^{e_2} \ldots b^{e_k} \bmod n \equiv$

$$\equiv b^{e_1} (\bmod n) \cdot b^{e_2} (\bmod n) \cdot \ldots \cdot b^{e_k} (\bmod n)$$

Modular Exponentiation

- For example, let $b = 90; e = 101; n = 1189$
- Then

$$e = 101 = 2^0 + 2^2 + 2^5 + 2^6 = 1 + 4 + 32 + 64$$

$$b^e = 90^{101} = 90^{1+4+32+64} \equiv 90 \cdot 90^4 \cdot 90^{32} \cdot 90^{64} (\mathrm{mod}\, 1189) \equiv$$

$$\equiv 90 (\mathrm{mod}\, 1189) \cdot (90^2)^2 (\mathrm{mod}\, 1189) \cdot (90^8)^4 (\mathrm{mod}\, 1189) \cdot (90^8)^8 (\mathrm{mod}\, 1189) \equiv$$

$$\equiv 90 \cdot 980 \cdot 1125 \cdot 529 = 88200 \cdot 1125 \cdot 529 = 240750 \cdot 529 \equiv 240750 \cdot 529 (\mathrm{mod}\, 1189) =$$

$$= 240750 (\mathrm{mod}\, 1189) \cdot 529 (\mathrm{mod}\, 1189) \equiv 572 \cdot 529 (\mathrm{mod}\, 1189) = 302588 (\mathrm{mod}\, 1189) \equiv 582$$

RSA Encryption

- In the RSA **encryption** method, ***n*** is chosen to be the product of **two secrete prime numbers** $\boldsymbol{n=pq}$.
- If $n=pq$, then $\boldsymbol{e}$ is chosen so that GCD $(e, \boldsymbol{r}) = 1$, where $\boldsymbol{r}= (p-1)(q-1)$. There are many such es (almost all positive integers satisfy the given condition)
- (n, e) is a public key generated every time on the server when any information should be encrypted and then forwarded to the server
- After it is generated, the public key is forwarded to the user (a user's computer or mobile device)

RSA Decryption

- In the RSA **decryption**, the exponent d (which is a private key) is chosen as the smallest positive solution d to the congruence $ed \equiv 1 \pmod{r}$. Hence $d = e^{-1} \pmod{r}$
- Then the **decryption procedure** is as follows:

$$\underbrace{(b^{e} \bmod n)}_{\text{Cipher}}{}^{d} \bmod n = b$$

- (p, q, d) is a secrete private key stored on the server during the current session and used for decryption

RSA Encryption

- To use the RSA encryption method, we have to provide our correspondent with the public key
(n, e) and to keep a secrete private key d hidden
- Our correspondent then can transmit an encrypted messages to us
- At any moment (for any new customer, any new online session, etc.), it is possible to generate new $n=pq$ and e such that GCD $(e, r) = 1$,
where $r = (p-1)(q-1)$ and $d=e^{-1}$ **(mod r)** accordingly
- A public key (n, e) <u>can always be transmitted via a communication channel</u>. This happens wherever we have to forward an encrypted message over the Internet (e.g., to access a bank account)

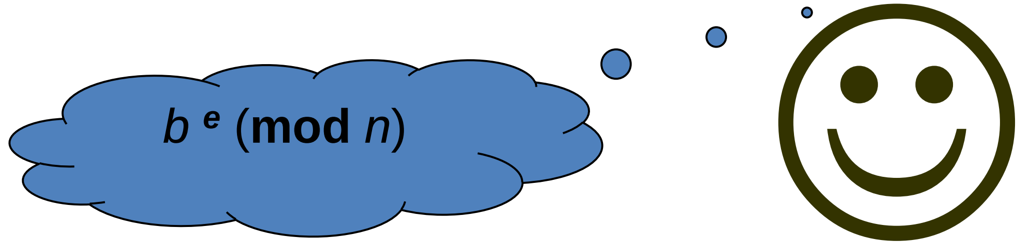
RSA Encryption: Algorithm

- A server randomly generates prime numbers p and q
- $n=pq$ and $r=(p-1)(q-1)$ are then calculated and e is chosen such that GCD $(e, r) = 1$
- d is chosen as the smallest positive solution d to the congruence
  $\boldsymbol{ed \equiv 1 \pmod{r}}$
- (n, e) is a public key, it should be then forwarded to the user, while a secrete private key $\boldsymbol{d}$ is kept hidden for decryption
- A message b can then be encrypted as $\mathbf{s} = b^{e} \bmod n$ and forwarded to the server
- A message can be decrypted on the server as
  $\mathbf{s}^{\boldsymbol{d}} \bmod n = (b^{e} \bmod n)^{\boldsymbol{d}} \bmod n = \boldsymbol{b}$

RSA Encryption

- The strength of the RSA encryption technique is based on the impossibility to fit p and q such that $pq=n$
- Even if n is known, it does not help. There is no rule, which may help to find prime p and q such that they are only factors of n (except 1 and n itself). This explains why n shall not be hidden
- When n is large enough, any attempt to find it may take years even if the most powerful supercomputer is used

RSA Encryption



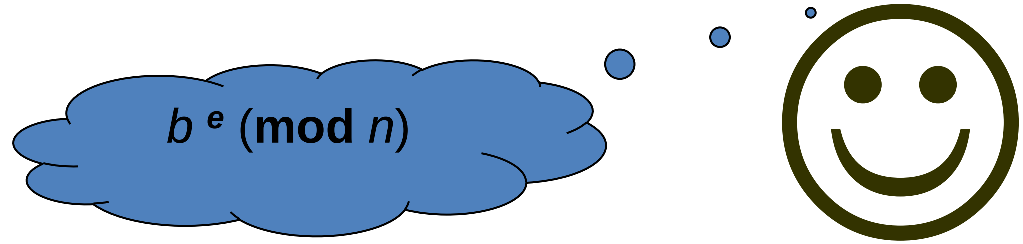
n = 4559, e = 13.

Smiley Transmits: "Last name Smiley"

n=4559=97x47; r=96x46=4416;

GCD(e,r)=1 → e=13 (for example)

RSA Encryption



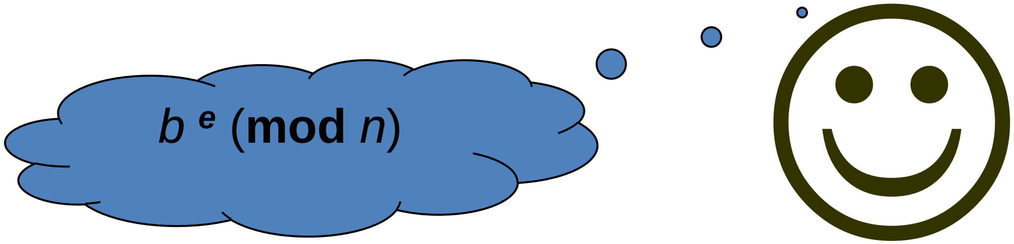
n=4559, e = 13.

Smiley Transmits: “Last name Smiley”

L A	S T	N	A M	E	S M	I L	E Y

- n=4559=97x47
- Spaces are encoded by 00, while letters are encoded by their numbers A=01, B=02, …

RSA Encryption

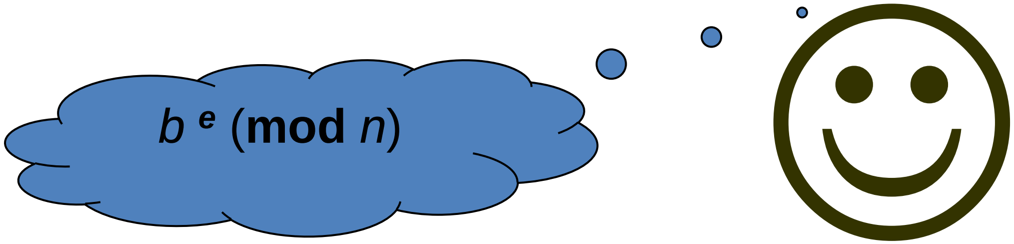


n = 4559, e = 13.

Smiley Transmits: “Last name Smiley”

◆	L A	S T	N	A M	E	S M	I L	E Y
◆	**1201**	**1920**	**0014**	**0113**	**0500**	**1913**	**0912**	**0525**

RSA Encryption



$n = 4559$, $e = 13$.

Smiley Transmits: “Last name Smiley”

L A	S T	N	A M	E	S M	I L	E Y
1201	**1920**	**0014**	**0113**	**0500**	**1913**	**0912**	**0525**

- 1201^{13} **mod** 4559, 1920^{13} **mod** 4559, …

RSA Encryption

b^{e} (**mod** n)

$n = 4559$, $e = 13$.

Smiley Transmits: “Last name Smiley”

L A	S T	N	A M	E	S M	I L	E Y
1201	**1920**	**0014**	**0113**	**0500**	**1913**	**0912**	**0525**

- 1201^{13} **mod** 4559, 1920^{13} **mod** 4559, …
- 2853 0116 1478 2150 3906 4256 1445 2462

RSA Encryption

- a private decryption exponent *d* , when applied to b^{e} **mod** *n* , recovers the original blocks *b* :

$$\left(\underbrace{\left(b^{e} \bmod n\right)}_{\text{cipher}}{}^{d} \right) \bmod n = b$$

- For *n* = 4559, *r*=4416, *e* = 13, the decryptor *d* = 3397, which is the smallest positive solution to the congruence $ed \equiv 1 \bmod r$