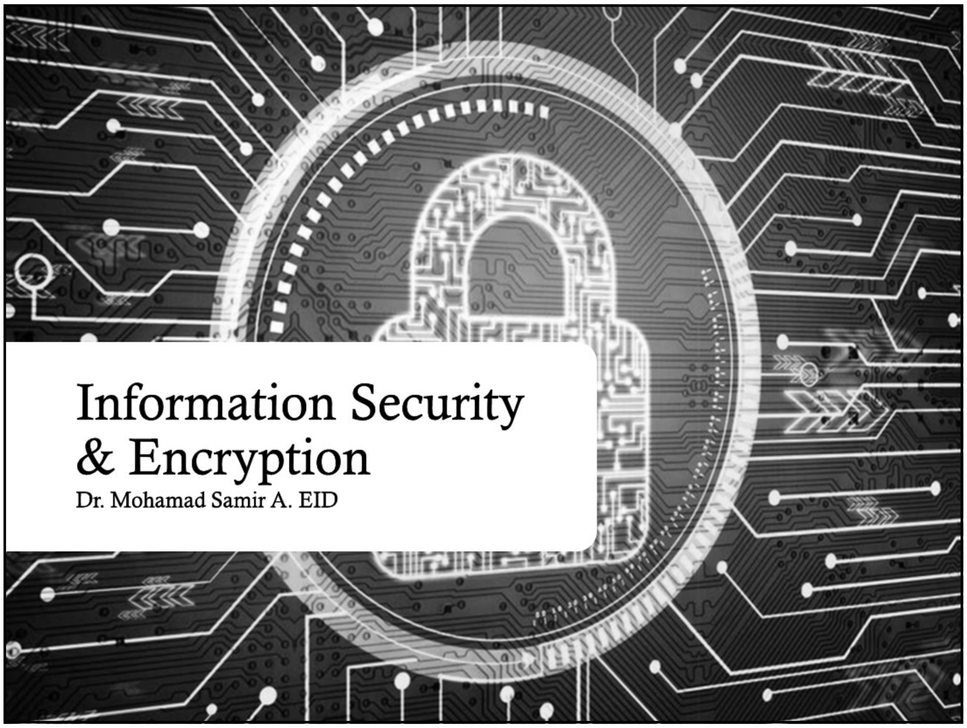


Information Security & Encryption

Dr. Mohamad Samir A. EID

1

Remarks

- Detailed syllabus has been updated (V5).
- Week 3 in the book:
  - Paar: Chapter 4 (till section 4.4.3)
  - Paar: Chapter 5 (Sections 5.1.1, 5.1.2, 5.1.4)
- Week 4 in the book:
  - Paar: Chapter 4 (Section 4.4.4 ~ end)
  - Paar: Chapter 6 (Sections 6.1~6.3)

2

Assignment 2

- Answer the following problems of Sheet 2 (1, 2, 6, 8b, 10, 13, 14).
- Due date: 27/Feb (Thurs.) by 5:30 PM.
- Submit via Google Classroom.

Last Week (Week 3)

- Modes of Operation of Symmetric Encryption
- AES (Encryption)

This Week (Week 4)

- Continue AES
- Tools for PKC

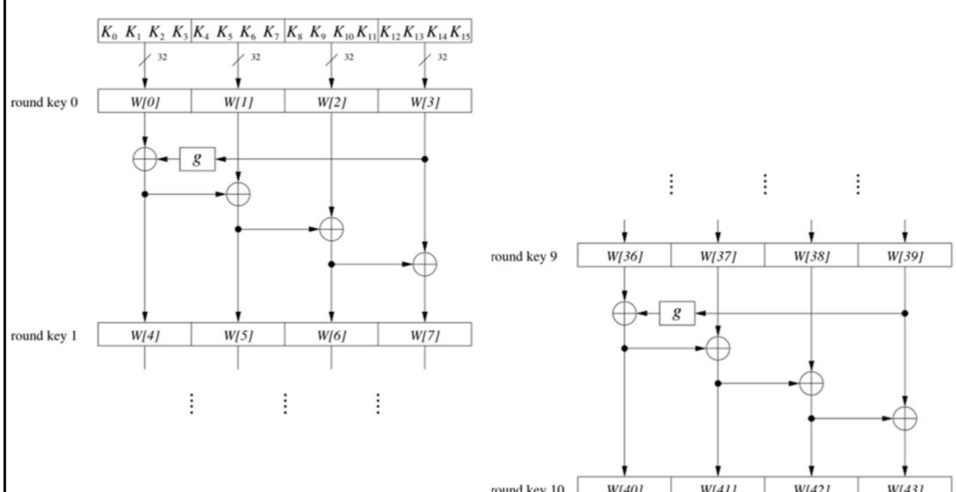
AES Key Scheduling or Expansion

- Objective: take a key vector (size: 128/192/256 bits) and generate n_r+1 round-keys for (n_r:10/12/14 rounds).
- Three different (yet similar) expansion methods.
- Subkeys are generated recursively.
- Processing is word-oriented (32 bits).
- Key expansion array W is used to store all round-keys.

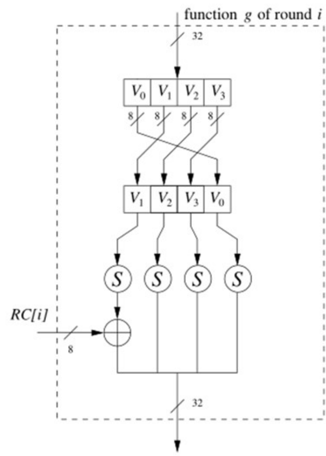
Design Criteria

- Simplicity of description and speed of processing.
- Usage of round constants to eliminate symmetries.
- Diffusion: Each key bit affects many round-key bits.
- Knowledge of a part of the cipher key or round-key does not enable calculation of many other round-key bits.

AES-128 Key Scheduling



AES-128 Key Scheduling



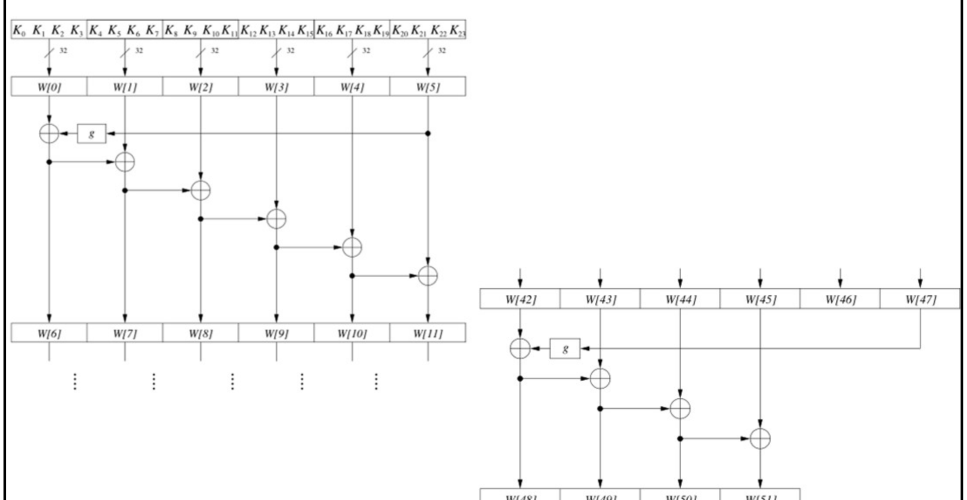
Rationale:

- ⟐ Introduce nonlinearity.
- ⟐ Remove symmetry or similarity of how different subkeys are generated.

$$
\begin{aligned}
RC[1] &= x^0 = (0000\,0001)_2,\\
RC[2] &= x^1 = (0000\,0010)_2,\\
RC[3] &= x^2 = (0000\,0100)_2,\\
&\vdots\\
RC[10] &= x^9 = (0011\,0110)_2.
\end{aligned}
$$

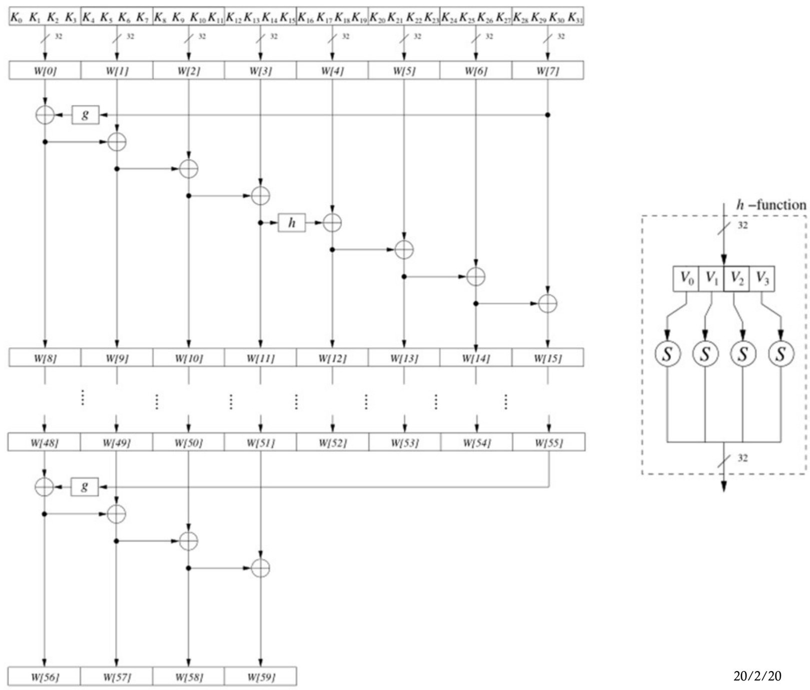
9

AES-192 Key Scheduling



10

AES-256 Key Scheduling



AES Decryption

AES decryption inverts all the encryption operations.

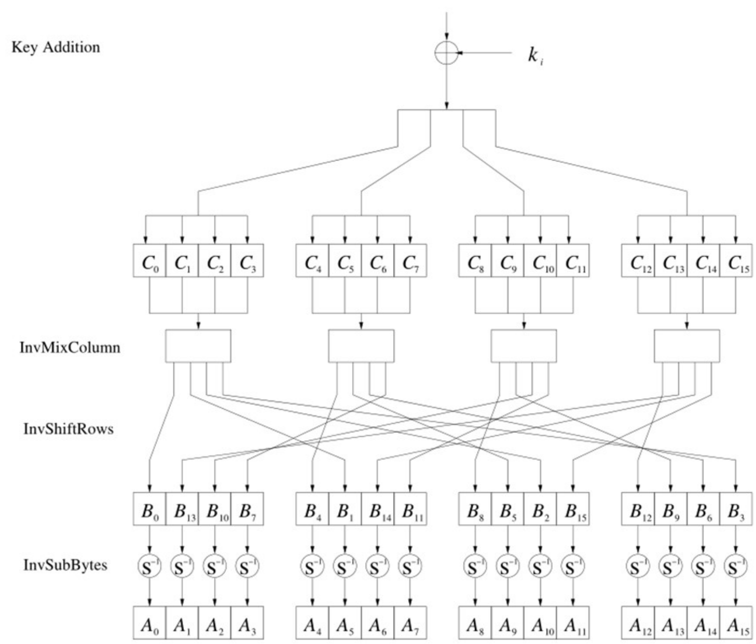
- Last round-key used for encryption is the first used in decryption.
- No MixColumn in last round in encryption and first round of decryption.

AES Decryption

“AES decryption block diagram”

13

AES Decryption



“AES decryption round”

14

AES Decryption

Inverse MixColumn Layer

$$\begin{pmatrix} B_0 \\ B_1 \\ B_2 \\ B_3 \end{pmatrix} = \begin{pmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{pmatrix} \begin{pmatrix} C_0 \\ C_1 \\ C_2 \\ C_3 \end{pmatrix}$$

Multiplication and addition are done in $GF(2^8)$

How to verify that this is the inverse of MixColumn?

AES Decryption

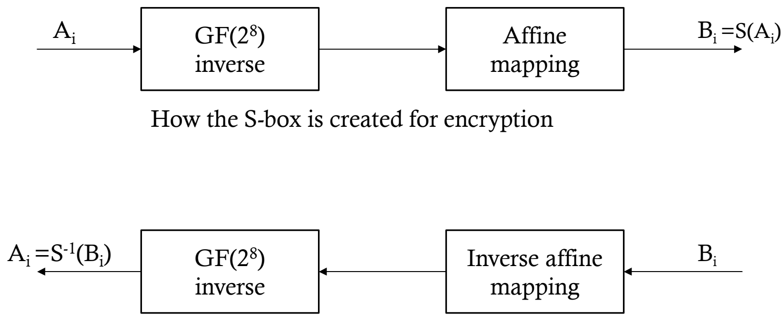
Inverse Shift Rows Layer

Opposite of the Shift Rows operation in encryption.

B_0	B_4	B_8	B_{12}	no shift
B_{13}	B_1	B_5	B_9	$\longrightarrow$ one position right shift
B_{10}	B_{14}	B_2	B_6	$\longrightarrow$ two positions right shift
B_7	B_{11}	B_{15}	B_3	$\longrightarrow$ three positions right shift

AES Decryption

Inverse Byte Substitution Layer



How the S-box is created for encryption

How the Inverse S-box is created for decryption

See table (page 114)

 20/2/20 17

17

Helpful Tools

AES (step-by-step)

https://www.cryptool.org/en/cto-highlights/aes

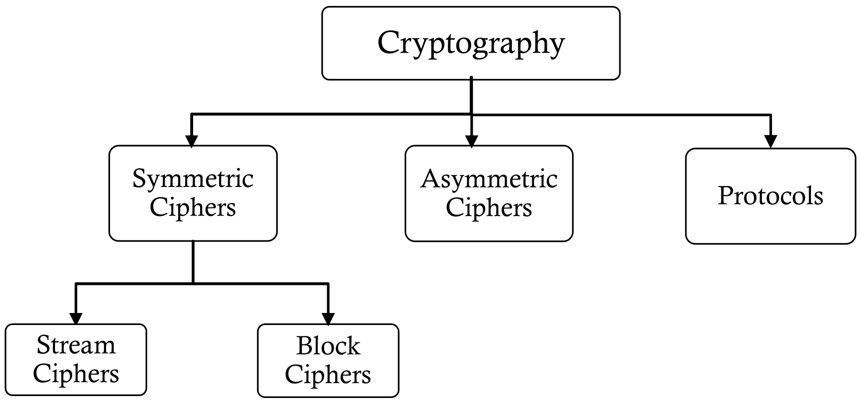
AES Internal Steps (Excel file)

https://www.nayuki.io/page/aes-cipher-internals-in-excel

 20/2/20 18

18

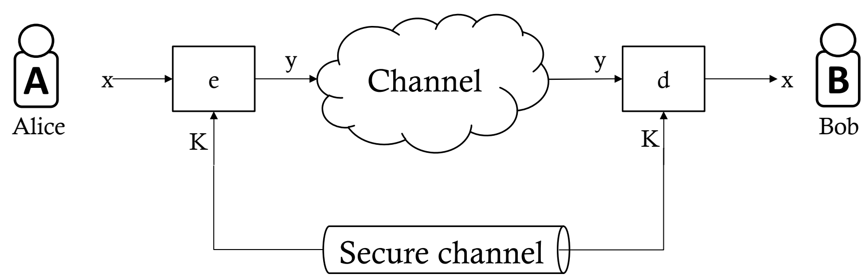
Main Areas of Cryptography



Symmetric Cryptography Revisited

Principle of Symmetric-Key encryption:

X: plaintext
Y: ciphertext
K: key

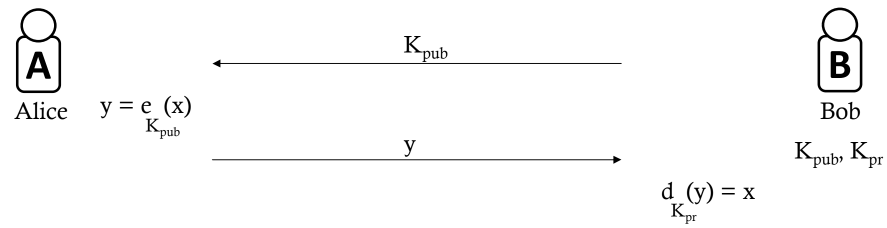


We have some problems; key distribution, number of keys, etc. (see page 150)

Intro to Public-Key Cryptography (PKC)

K_{pub}: public key
K_{pr}: private key

Basic protocol for Public-Key encryption:



To study e() and d() .. **<u>more Math is needed</u>…**

Essential Number Theory for PKC

- Topics:
  - Euclidean Algorithm (EA)
  - Extended Euclidean Algorithm (EEA)
  - Euler's Phi Function
  - Fermat's Little Theorem and Euler's Theorem

Euclidean Algorithm (EA)

Goal: computing greatest common divisor of two positive numbers r_0, r_1 ;
$\gcd(r_0, r_1)$

For small numbers, simple factoring can get the gcd. i.e., no real need for EA:

e.g., $r_0 = 84$, $r_1 = 30$

$r_0 = 84 = 2 \cdot 2 \cdot 3 \cdot 7$

$r_1 = 30 = 2 \cdot 3 \cdot 5$

gcd is the product of all common prime factors

$\gcd(84,30) = 2 \cdot 3 = 6$

Such method doesn't work with large numbers, i.e., the case of PKC. **We need the EA.**

e.g., 3^{2000}

Efficient
(faster, less complex)

Euclidean Algorithm (EA)

Basic idea: $\gcd(r_0, r_1) = \gcd(r_0 \bmod r_1, r_1)$ . . . we simply reduce the problem.

$= \gcd(r_1, r_0 \bmod r_1)$

e.g. 1) $\gcd(84, 30) = \gcd(84 \bmod 30, 30) = \gcd(\mathbf{24}, 30)$

$= \gcd(30 \bmod 24, 24) = \gcd(\mathbf{6}, 24) \boxed{= 6}$

$= \gcd(24 \bmod 6, 6) = \gcd(\mathbf{0}, 6)$

Terminate once a zero remainder is reached; gcd is the last remainder.

Same e.g., (illustrated)

$\overset{r_0}{84} = 2 \cdot \overset{r_1}{30} + \overset{r_2}{24}$

$30 = 1 \cdot 24 + \overset{r_3}{6} \longrightarrow \gcd(84, 30) = 6$

$24 = 4 \cdot 6 + 0$ . . . zero remainder reached.

Euclidean Algorithm (EA)

e.g. 2) $\gcd(\overset{r_0}{973}, \overset{r_1}{301})$

$\overset{r_0}{973} = 3\cdot\overset{r_1}{301} + \overset{r_2}{70}$

$301 = 4\cdot 70 + \overset{r_3}{21}$

$70 = 3\cdot 21 + \overset{r_4}{7} \longrightarrow \gcd(973, 301) = 7$

$21 = 3\cdot 7 + 0$ . . . zero remainder reached.

Pretty simple…

25

Extended Euclidean Algorithm (EEA)

Goal: rewrite $\gcd(r_0, r_1) = s\cdot r_0 + t\cdot r_1$

Why and **How**?

Why: To compute modular inverses of large numbers.

26

Extended Euclidean Algorithm (EEA)

How: Using regular EA for the LHS & the extension below for each step:

$\gcd(r_0, r_1)$	$r_0 = q_1\cdot r_1 + r_2$	$r_2 = s_2\cdot r_0 + t_2\cdot r_1$
$\gcd(r_1, r_2)$	$r_1 = q_2\cdot r_2 + r_3$	$r_3 = s_3\cdot r_0 + t_3\cdot r_1$
	$\vdots$	$\vdots$
$\gcd(r_{l-2}, r_{l-1})$	$r_{l-2} = q_{l-1}\cdot r_{l-1} + r_l$	$r_l = \overset{s}{s_l}\cdot r_0 + \overset{t}{t_l}\cdot r_1 = \gcd(r_0, r_1)$
$\gcd(r_{l-1}, r_l)$	$r_{l-1} = q_l\cdot r_l + 0$	

r_l is the last remainder, i.e., the result of $\gcd(r_0, r_1)$

How does that lead to modulo inverse?

To compute $a^{-1} \bmod n$:
$\gcd(n, a) = r_l = s\cdot n + t\cdot a = 1$ (condition for inverse existence)
Then $s\cdot 0 + t\cdot a \equiv 1 \bmod n$ (mod n for both sides)
$t\cdot a \equiv 1 \bmod n$
$t \equiv a^{-1} \bmod n$

Extended Euclidean Algorithm (EEA)

e.g., Compute $91^{-1} \bmod 1500$ . . . then $r_0 = \mathbf{1500}$, $r_1 = \mathbf{91}$

gcd(1500, 91)	$\overset{r_0}{1500} = 16\cdot \overset{r_1}{91} + \overset{r_2}{44}$	$\overset{r_2}{44} = \overset{s_2}{(1)}\cdot \overset{r_0}{\mathbf{1500}} + \overset{t_2}{(-16)} \cdot \overset{r_1}{\mathbf{91}}$
gcd(91, 44)	$\overset{r_1}{91} = (2)\cdot \overset{r_2}{44} + \overset{r_3}{3}$	$\overset{r_3}{3} = (1)\cdot \overset{r_1}{91} + (-2) \cdot \overset{r_2}{44}$
		$= (1)\cdot 91 + (-2)\cdot[1500 + (-16)\cdot 91] = (-2)\cdot \overset{r_0}{\mathbf{1500}} + (33)\cdot \overset{r_1}{\mathbf{91}}$
gcd(44, 3)	$\overset{r_2}{44} = (14)\cdot \overset{r_3}{3} + \overset{r_4}{2}$	$\overset{r_4}{2} = (1)\cdot \overset{r_2}{44} + (-14) \cdot \overset{r_3}{3}$
		$= (1)\cdot[1500 + (-16)\cdot 91] + (-14)\cdot[(-2)\cdot 1500 + (33)\cdot 91]$
		$= 1500 + (-16)\cdot 91 + (28)\cdot 1500 + (-462)\cdot 91$
		$= (29)\cdot \overset{r_0}{\mathbf{1500}} + (-478)\cdot \overset{r_1}{\mathbf{91}}$
gcd(3, 2)	$\overset{r_3}{3} = (1)\cdot \overset{r_4}{2} + \overset{r_5}{1}$	$\overset{r_5}{1} = (1)\cdot \overset{r_3}{3} + (-1)\cdot \overset{r_4}{2}$ . . . . . . $= \overset{s}{(-31)}\cdot \overset{r_0}{\mathbf{1500}} + \overset{t}{(511)}\cdot \overset{r_1}{\mathbf{91}}$
gcd(2, 1)	$2 = (2)\cdot 1 + 0$	

Then $\gcd(1500, 91) = r_l = r_5 = 1 = (-31)\cdot\mathbf{1500} + (511)\cdot\mathbf{91}$
Doing mod 1500 for both sides → $91^{-1} \bmod 1500 \equiv 511 \bmod 1500$

Extended Euclidean Algorithm (EEA)

How to compute the modular inverse using the Extended Euclidean Algorithm:

i	$q_{i-1} = \left\lfloor \frac{r_{i-2}}{r_{i-1}} \right\rfloor$	$s_i = s_{i-2} - q_{i-1} \cdot s_{i-1}$	$t_i = t_{i-2} - q_{i-1} \cdot t_{i-1}$	$r_i = r_{i-2} - q_{i-1} \cdot r_{i-1}$
0		$s_0 = 1$	$t_0 = 0$	r_0
1		$s_1 = 0$	$t_1 = 1$	r_1
2	$q_1 = \left\lfloor \frac{r_0}{r_1} \right\rfloor$	$s_2 = s_0 - q_1 \cdot s_1$	$t_2 = t_0 - q_1 \cdot t_1$	$r_2 = r_0 - q_1 \cdot r_1$
3	$q_2 = \left\lfloor \frac{r_1}{r_2} \right\rfloor$	$s_3 = s_1 - q_2 \cdot s_2$	$t_3 = t_1 - q_2 \cdot t_2$	$r_3 = r_1 - q_2 \cdot r_2$
⋮	⋮	⋮	⋮	⋮

For the initialization steps ($i \in \{0,1\}$), the cell values are predetermined as proven in the appendix.

For $i \geq 2$, compute the q_{i-1}, s_i, t_i, r_i columns.

For each iteration *i*, check:

If $r_i=1$ is reached, then **gcd(r_0,r_1) = r_i = 1**. Then **multiplicative inverse of <u>r_1 mod r_0</u> exists and equals t_i**. Stop.

<u>Else</u>, if $r_i=0$ is reached, **then gcd(r_0,r_1) = r_{i-1}**. Then **multiplicative inverse of <u>r_1 mod r_0</u> doesn't exist**. Stop.

Note: r_0 should be the modulus and should be > r_1.

 29

29

Euler's Phi Function

For PKC, it's important to know how many numbers in Z_m that are relatively prime to m.

Why and **how**?

Why: Will be clear later once we study actual PK cryptosystems.

How: Using Euler's Phi function simply counts these numbers.

Manually counting may work for <u>small numbers</u>.

e.g., manually count the numbers in Z_6 that are relatively prime to 6.→ $\Phi(6) = 2$

For <u>large numbers</u>, we use Euler's Phi function.

 30

30

Euler's Phi Function

How to compute $\Phi(m)$ for a large m?

Let m have the following factorization form:

$$m = p_1^{e_1} \cdot p_2^{e_2} \cdot \ldots \cdot p_n^{e_n},$$

Where p_i are distinct prime numbers and e_i are positive integers, then

$$\Phi(m) = \prod_{i=1}^{n} (p_i^{e_i} - p_i^{e_i - 1}).$$

Euler's Phi Function

e.g. 1) compute $\Phi(m)$ for m = 240

$m = 16 \cdot 15 = 2^4 \cdot 3^1 \cdot 5^1$
$= p_1^{e_1}\, p_2^{e_2}\, p_3^{e_3}$

$$\Phi(240) = \prod_{i=1}^{3} (p_i^{e_i} - p_i^{e_{i-1}}) = (2^4-2^3)(3^1-3^0)(5^1-5^0)$$
$$= 8 \cdot 2 \cdot 4 = 64$$

e.g. 2) compute $\Phi(m)$ for m = 100

Euler's Theorem

Used in public-key cryptography.

Euler's Theorem:
Let a and m be integers with gcd(a,m) = 1,
then:
$a^{\Phi(m)} \equiv 1 \bmod m$

e.g., Let's check with m = 12 and a = 5.

$$\Phi(12) = \Phi(2^2 \cdot 3) = (2^2 - 2^1)(3^1 - 3^0) = (4-2)(3-1) = 4$$

$$5^{\Phi(12)} = 5^4 = 25^2 = 625 \equiv 1 \bmod 12$$

Euler's Phi Function

Notes:
If p is prime, then $\Phi(p) = p - 1$
if p and q are prime, then $\Phi(pq) = \Phi(p) \times \Phi(q)$

How to prove that $\Phi(pq) = \Phi(p) \times \Phi(q)$?

Since Z_n has (pq-1) positive integers.

Since integers that are not relatively prime to n are {p,2p,…(q-1)p} and {q,2q,..,(p-1)q} … i.e., (p-1) elements + (q-1) elements.

Then the number of integers in Z_n that are relatively prime to n = (pq-1) – [(p-1)+(q-1)]

i.e., pq – (p+q) +1

Then $\Phi(n)$=(p-1)x(q-1) = $\Phi(p) \times \Phi(n)$

Fermat's Little Theorem

Fermat's Little Theorem:

Let a be an integer and p be a prime,

then: $a^p \equiv a \bmod p$

so, $a^{p-1} \equiv 1 \bmod p$

or, $a \cdot a^{p-2} \equiv 1 \bmod p$

so, $a^{-1} \equiv a^{p-2} \bmod p$

e.g., Let's check with p = 7 and a = 2.

$$a^{p-2} = 2^5 = 32 \equiv 4 \bmod 7$$

$$2 \cdot 4 \equiv 1 \bmod 7$$

$$\boxed{\text{Therefore, } 2^{-1} \equiv 4 \bmod 7}$$

Practice: Sheet 2

Solved in tutorial: 4, 7, 8a, 11, and 12.
Assignment 2: 1, 2, 6, 8b, 10, 13, 14

Thank You

37