Lecture Notes

Network Security – Part 4
Public Key Cryptography

Mohammad Sayad
Department of Information Technology
University of Tehran

1

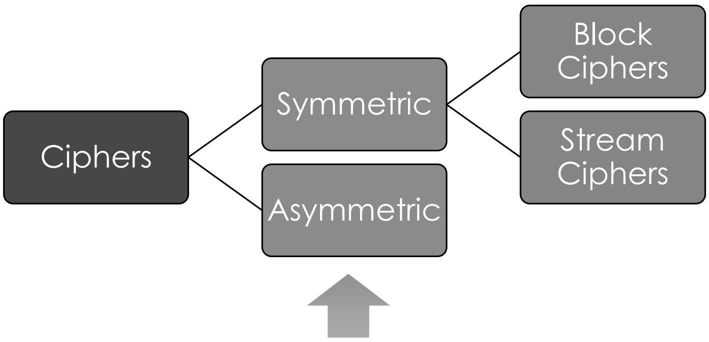
Asymmetric Cryptography
(Public Key Cryptography)

2

Lecture Notes-Sayad

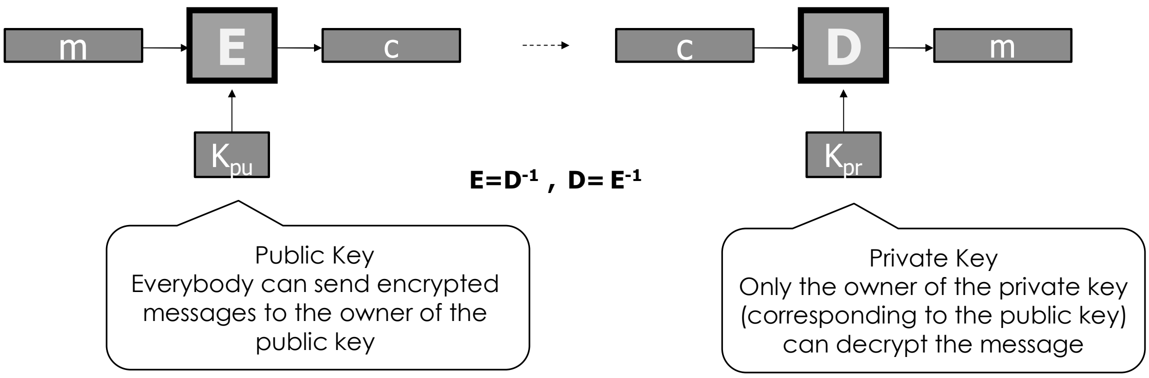
Encryption Methods

Asymmetric Cryptography (Public Key Cryptography)



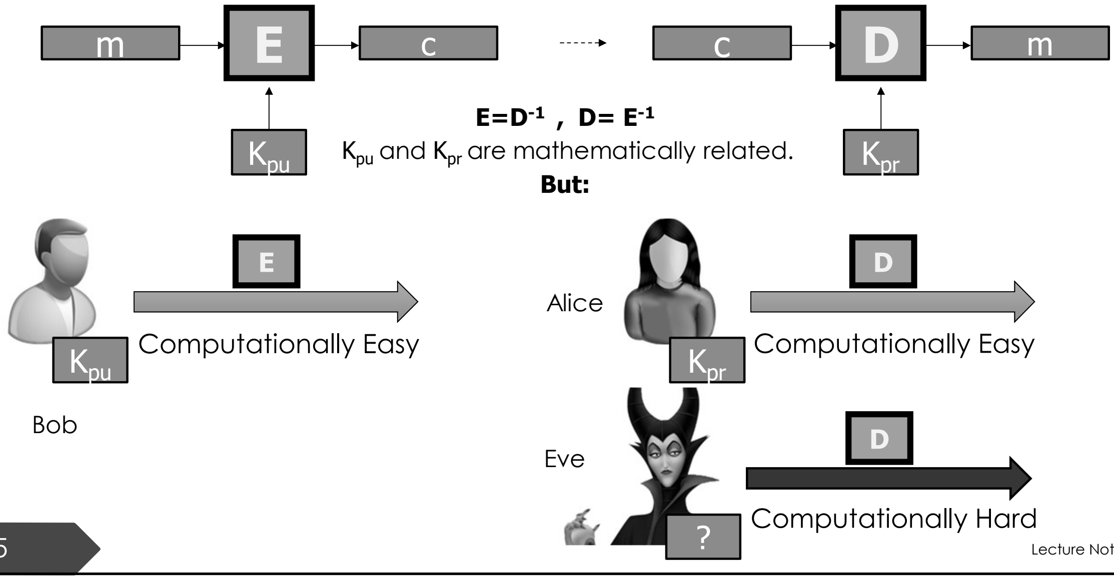
General Form of Asymmetric Cryptography

Asymmetric Encryption (Public Key Encryption)



Samples: RSA ،Elgamal ،Elliptic Curve Cryptos, …

Security of Asymmetric Algorithms



Public Key Systems

- Merkle-Hellman knapsack
- Diffie-Hellman key exchange
- RSA
- Rabin cipher
- NTRU cipher
- ElGamal
- …

Public Key Crypto

- Some public key systems provide it all: encryption, digital signatures, etc.
  - For example: RSA
- Some are only for key exchange
  - For example: Diffie-Hellman
- Some are used for signatures more
  - For example: ElGamal
- **All of these are public-key systems !**

Modular Arithmetic

"**mod**" gives the residue of a division operation:

example :

8 mod 4 = 0

6 mod 4 = 2

1 mod 4 = 1

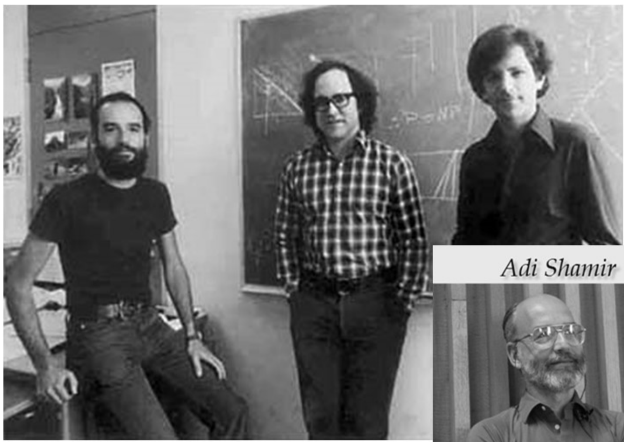
13 mod 4 =1

"**a**" and "**b**" are called **congruent modulo n** if they have the same residue in division over "**n**".

$$a \overset{n}{\equiv} b \quad \text{or} \quad a \equiv b \bmod n$$

RSA Asymmetric Encryption Algorithm

➨Introduced by Rivest, Shamir & Adleman at MIT in 1978.



RSA Asymmetric Encryption Algorithm

➨How do we make an RSA cryptosystem? :

1- choose two large prime numbers p & q.

2- calculate n=p*q

3- calculate Euler's Phi function as $\varphi(n) = (p-1)(q-1)$

$\varphi(n)$ is an arithmetic function that counts the positive integers less than or equal to *n* that are relatively prime to n (i.e. their GCD with *n* is 1).

RSA is called a block cipher in Stallings' book, while it is not

RSA (cont'd)

4- choose an encryption key ("**e**") so that it's relatively prime to $\varphi(n)$.

5- calculate its inverse congruent modulo $\varphi(n)$ and call it "**d**". $e.d \overset{\varphi(n)}{\equiv} 1$

- This is done by the Extended Euclidean Algorithm which we will see later

Public Parameters : PU={e, n}

Private Parameters : PR={d}

Encryption :

Plaintext:	$M < n$
Ciphertext:	$C = M^e \pmod{n}$

Decryption :

Ciphertext:	C
Plaintext:	$M = C^d \pmod{n}$

gcd(M,n) must be 1?

No, the only condition is that **M<n**

Why does RSA work?

Encryption: $C \overset{n}{\equiv} M^e$

Decryption: $C^d \overset{n}{\equiv} (M^e)^d \overset{n}{\equiv} M^{ed}$

Euler's Theorem (Fermat's Little Theorem): $a^{\varphi(n)} \overset{n}{\equiv} 1$ if $(a,n)=1$

Assignment #1 $\longrightarrow M^{ed} \overset{n}{\equiv} M^{ed \bmod \varphi(n)} \overset{n}{\equiv} M^1$

Even if gcd(M,n)≠ 1 , it's possible to prove that $M^{ed} \equiv M \bmod n$

RSA (cont'd)

Factoring is known to be a hard mathematical problem:

28=2*2*7

Based on this, it has been proven that knowing ***n** & **e**, and* without knowing **p & q,** calculation of **d** is mathematically hard and is equivalent to factoring ***n*** (which is a big number!).

While, if one has **p & q**, he can easily compute $\varphi(n)$ and find the inverse of **e** (i.e. **d**) .

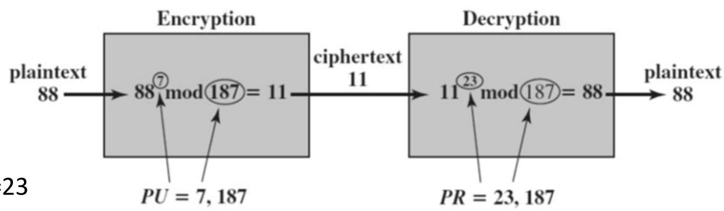
Example:

p=11, q=17

n=11*17=187

$\varphi(n)$=(p-1)(q-1)=160

e=7 (notice that gcd(e, $\varphi(n)$)=1 => d=23



Encryption

$$88^7 \bmod 187 = [(88^4 \bmod 187) \times (88^2 \bmod 187) \times (88^1 \bmod 187)] \bmod 187$$
$$88^1 \bmod 187 = 88$$
$$88^2 \bmod 187 = 7744 \bmod 187 = 77$$
$$88^4 \bmod 187 = 59{,}969{,}536 \bmod 187 = 132$$
$$88^7 \bmod 187 = (88 \times 77 \times 132) \bmod 187 = 894{,}432 \bmod 187 = 11$$

Decryption

$$11^{23} \bmod 187 = [(11^1 \bmod 187) \times (11^2 \bmod 187) \times (11^4 \bmod 187) \times (11^8 \bmod 187) \times (11^8 \bmod 187)] \bmod 187$$

$$11^1 \bmod 187 = 11$$

$$11^2 \bmod 187 = 121$$

$$11^4 \bmod 187 = 14{,}641 \bmod 187 = 55$$

$$11^8 \bmod 187 = 214{,}358{,}881 \bmod 187 = 33$$

$$11^{23} \bmod 187 = (11 \times 121 \times 55 \times 33 \times 33) \bmod 187 = 79{,}720{,}245 \bmod 187 = 88$$

Euclidean Algorithm

- How do we find the inverse of a number modulo $\varphi(n)$:

$e*d \bmod \varphi(n) = 1$

$7*d \bmod 40 = 1$

Step 1: Euclidean algorithm

$40x + 7y = 1$

$40 = 5(7) + 5$

$7 = 1(5) + 2$

$5 = 2(2) + 1$

Step 2: Back substitution

$1 = 5 - 2(2)$

$1 = 5 - 2(7 - 1(5))$

$1 = 3(5) - 2(7)$

$1 = 3(40 - 5(7)) - 2(7)$

$1 = 3(40) - 17(7)$

$d = 40 - 17$

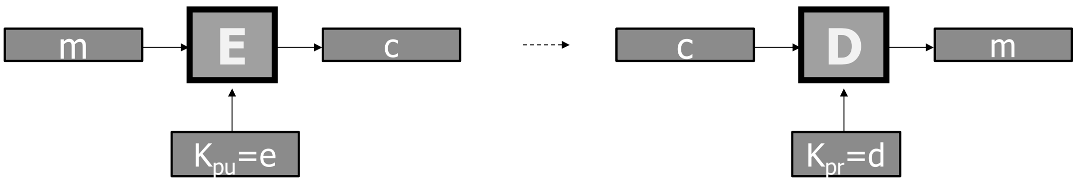
$= 23$

$p = 11$
$q = 5$
$n = 55$
$\varphi(n) = 40$
$e = 7$
$d =$

The End of RSA

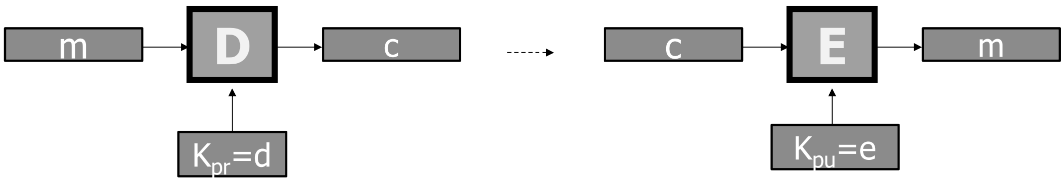
Recall

Asymmetric Cryptosystems:



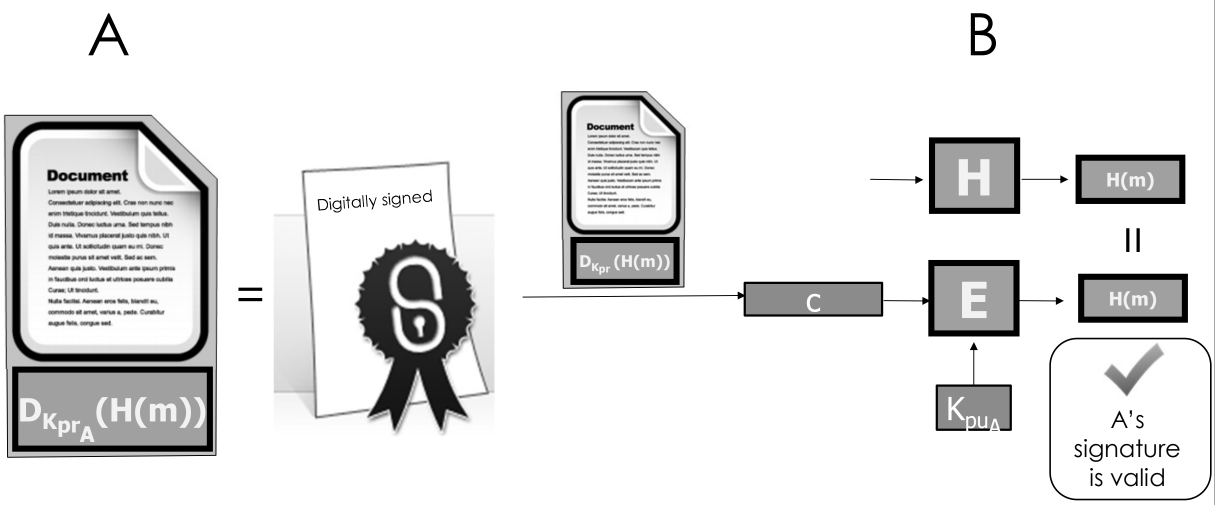
Encryption and Decryption in the Reverse Order

$$E=D^{-1}\ ,\ D=E^{-1}$$



We mentioned that this order is usually used in digital signatures.

Digital Signature



Example: DSS algorithm (very much like RSA)

Major Points about Digital Signature

- Digital signatures (DSs) provide "integrity", "authentication" and "non-repudiation" services.
- Forging a DS is not possible, since nobody has the private key but **A**.
  - Having the public key does not give a clue to what the private key is (Remember, it is computationally hard!).
- Any manipulation of the message on its way will cause mismatches of the hashes (i.e. H(m)) at the final stage and hence, will be detected.
- Since nobody else has **A**'s private key, whatever he signs cannot be denied later on. This is called non-repudiation. Everybody is free to check what **A** has signed.

Diffie-Hellman (DH) Key Exchange

Diffie-Hellman (DH) Key Exchange Protocol

The goal is that two independent parties, A and B, who have had no contacts previously, can make a symmetric key using a common channel without sending the key over it.

- RSA's security was based on the difficulty of the factoring problem.
- DH's security is based on the difficulty of another mathematical problem which is so called Discrete Logarithm problem.

Discrete Logarithm

- Consider the prime number p. Among 1,…,p-1, some are called the "primitive roots" since they create the whole set of 1,…,p-1 numbers by being powered to different numbers modulo p.

p=5, a=2

a=2,

a^2 mod p = 4,

a^3 mod p = 3,

a^4 mod p = 1

Easy

i is not modulo p

For any $1 \leq b \leq p-1$ $\Rightarrow$ $b = a^i \ mod \ p$

$\Downarrow$

$i = dlog_a \mathrm{b}$

Hard

Diffie-Hellman Protocol

Global Public Elements	
q	prime number
α	$\alpha < q$ and α a primitive root of q

User A Key Generation	
Select private X_A	$X_A < q$
Calculate public Y_A	$Y_A = \alpha^{X_A} \bmod q$

User B Key Generation	
Select private X_B	$X_B < q$
Calculate public Y_B	$Y_B = \alpha^{X_B} \bmod q$

Generation of Secret Key by User A

$K = (Y_B)^{X_A} \bmod q$

Generation of Secret Key by User B

$K = (Y_A)^{X_B} \bmod q$

25

Lecture Notes-Sayad

Diffie-Hellman Key Agreement

Private key part of user A

Private key part of user B

X_A

X_B

$Y_A = \alpha^{X_A} \bmod q, \quad \{\alpha, q\}$

$Y_B = \alpha^{X_B} \bmod q$

$K = (\alpha^{X_A})^{X_B} \bmod q = \alpha^{X_A X_B} \bmod q$

$K = (\alpha^{X_B})^{X_A} \bmod q = \alpha^{X_A X_B} \bmod q$

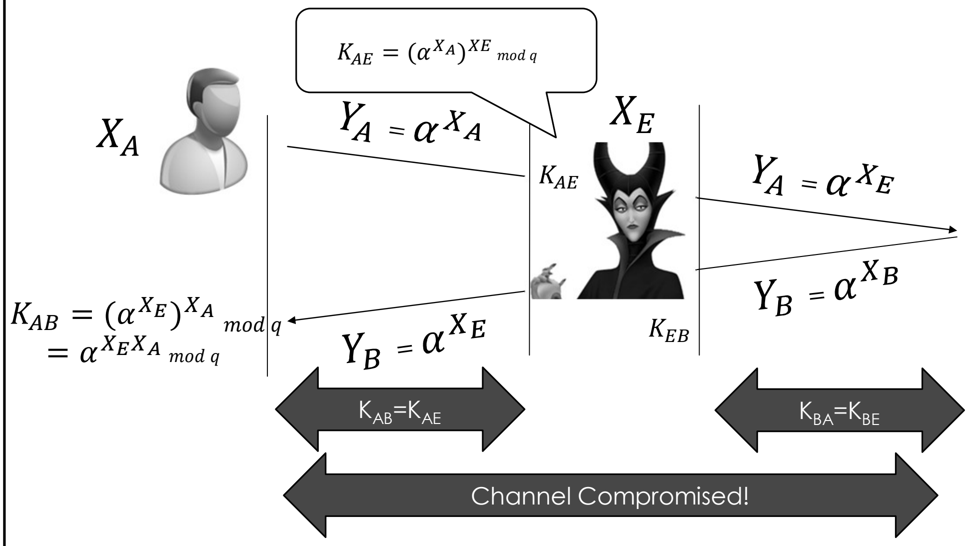
Secured Channel with the Key K

q & α are both public values and can be used by any adversary

26

Lecture Notes-Sayad

Man in the Middle Attack



Each of the parties thinks that it has established a secure connection with the other. While the line is intercepted and the information is decrypted and re-encrypted at the middle.

27

Lecture Notes-Sayad

End of Part 1

28

Lecture Notes-Sayad

Elgamal (Asymmetric) Cryptosystem

Elgamal Public-key Encryption Scheme

Arabic: طاهر الجمل

- ElGamal is a public-key cryptosystem
  - was designed by Dr. Taher Elgamal
- Mostly known for his digital signatures

Taher A. Elgamal

Taher A. Elgamal (2010)

Born	18 August 1955 (age 60) Cairo, Egypt
Residence	United States
Nationality	Egyptian, United States

Two Important yet Similar Theories

- Fermat's Little Theorem

$$a^{p-1} \overset{p}{\equiv} 1 \quad \text{(p is prime)}$$

- Euler Theorem

$$a^{\phi(n)} \overset{n}{\equiv} 1 \quad \text{if gcd(a,n)=1}$$

Elgamal Cryptography

Key Generation

1. Chooses a prime p
2. Chooses a generator a
3. Chooses an integer x $(x \leq p-2)$ as his secret
4. Calculates $d = a^x \ mod \ p$
5. Public key = $\{p, a, d\}$, Private key=$\{x\}$

Encryption

6. Gets $\{p, a, d\}$
7. Chooses an integer k $(1 < \mathrm{k} \leq \mathrm{p} - 2)$
8. Take the plain-text message m $(1 < \mathrm{m} \leq \mathrm{p} - 1)$
9. Computes $\mathrm{y} = a^k \ mod \ p$
10. Computes $\mathrm{z} = d^k \times m \ mod \ p$
11. Cipher-text : C= (y, z)

Decryption

12. Computes $\mathrm{r} = \mathrm{y}^{p-1-x} \ mod \ p$
13. Plain-text : $\mathrm{m} = (\mathrm{r} \times z) \ mod \ p$

Why does it work?

$$
\begin{aligned}
(\mathrm{r} \times z) \quad mod\ p &= \mathrm{y}^{p-1-x} \times \mathrm{d}^{\mathrm{k}} \times \mathrm{m} \ \ \mathrm{mod\ p} \\
&= \left(\mathrm{a}^{k}\right)^{p-1-x} \times \mathrm{d}^{\mathrm{k}} \times \mathrm{m} \ \ \mathrm{mod\ p} \\
&= \left(\mathrm{a}^{k}\right)^{p-1-x} \times (\mathrm{a}^{x})^{k} \times \mathrm{m} \ \ \mathrm{mod\ p} \\
&= a^{k(p-1-x)+kx} \times \mathrm{m} \ \ \mathrm{mod\ p} \\
&= a^{k(p-1)} \times \mathrm{m} \ \ \mathrm{mod\ p} \\
&= \mathrm{m} \ \ \mathrm{mod\ p}
\end{aligned}
$$

The original message

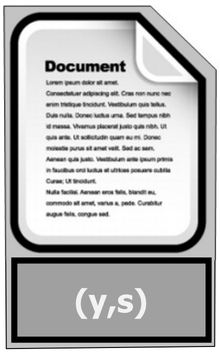
33

Lecture Notes-Sayad

Elgamal's Signature

Singing:

- ➨Choose a random k $(1 < k < p-1)$ & $\gcd(k, p-1) = 1$
- ➨ Compute $y = a^k \ \ mod\ p$ & $\mathrm{d} = a^x \ \mathrm{mod\ p}$
- ➨ Compute $s = (H(m) - xy) \times k^{-1} \ \ mod\ p - 1$
- ➨ The pair of (y,s) is the signature of m



Verification:

- ➨ checks whether $a^{H(m)} \overset{p}{\equiv} \mathrm{d}^{y} y^{s}$

Test it yourself using Fermat's Little Theorem

34

Lecture Notes-Sayad

Major Points about Elgamal's Scheme

- The encryption process requires two modular exponentiations (extra time).
- A disadvantage of ElGamal encryption is that there is message expansion by a factor of 2. That is, the cipher-text is twice as long as the corresponding plain-text.
- It's security relies on Discrete Logarithm problem (just like DH)
- Some reductions in space makes it less secure

Major Points about Elgamal's Scheme

- The signer must be careful to choose a different *k* randomly for each signature and to be certain that *k* is not leaked.
- Otherwise, an attacker may be able to deduce the secret key *x* with reduced difficulty, perhaps enough to allow a practical attack.
- In particular, if two messages are sent using the same value of *k* and the same key, then an attacker can compute *x* directly.*

- *Taher ElGamal (1985). "A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms", IEEE Transactions on Information Theory.* ***31*** *(4): 469–472*

Elliptic Curve Cryptography

Elliptic Curve Cryptography

- **Elliptic curve cryptography** (**ECC**) is an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite fields. ECC requires smaller keys compared to non-EC cryptography (based on plain Galois fields) to provide equivalent security.

- Elliptic curves are applicable for encryption, digital signatures, pseudo-random number generation and other tasks.

Elliptic Curve Cryptography

- For almost every **public-key cryptosystem**, there is an alternative based on ECC. So ECC is another domain for implementation of public-key schemes.

- The ECC schemes are usually faster or more secure with the same key size. Consequently, ECC is particularly appropriate for resource-limited embedded devices (e.g. smart cards).
- It has its own mathematics: ADDITION, MULTIPLICATION, …

39

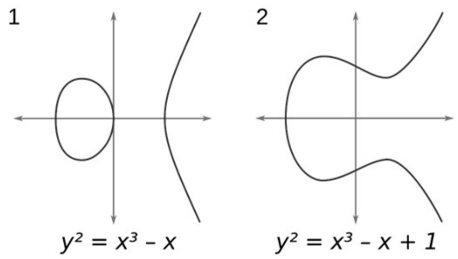
Lecture Notes-Sayad

What's an Elliptic Curve

➨**EC** is the set of points described by the equation:

$$y^2 = x^3 + ax + b$$

Where $4a^3 + 27b^2 \neq 0$. (this is required to exclude singular curves).



Non-singularity

➨Non-singularity means that the graph has no cusps, self-intersections, or isolated points.

On the left, a curve with a cusp (*y*^2=*x*^3). On the right, a curve with a self-intersection (*y*^2=*x*^3−3*x*+2). None of them is a valid EC for cryptography.

A bit of mathematics:

We can define a <u>group</u> over elliptic curves. Specifically:

- The elements of the group are the points of an elliptic curve;
- The identity element is the point at infinity 0; ($e.g.\ y = \infty$)
- The inverse of a point $P = (x, y)$ is the one symmetric about the x-axis, that is $-P = (x, -y)$;

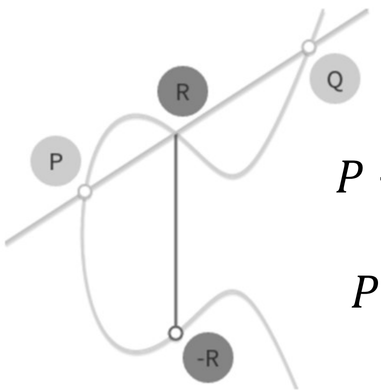
These two minuses are different !

- Addition is given by the following rule: given three aligned, non-zero points P, Q and R, their sum is P+Q+R=0.

43

Lecture Notes-Sayad

"+" operation on ECs



$$P + Q + R \triangleq 0$$
$$\rightarrow$$
$$P + Q \triangleq -R$$

Draw the line through P and Q. The line intersects a third point R. The point symmetric to it, $-R$, is the result of $P + Q$.

Similarly we can imagine where 0 lies from P+0=P → y=inf

(Andrea Corbellini)

44

Lecture Notes-Sayad

Addition on ECs:

- If P and Q are distinct, the line through them has the **slope**:

$$m = \frac{y_P - y_Q}{x_P - x_Q}$$

- If P=Q, the tangent line will have the **slope**:

$$m = \frac{3x_P^2 + a}{2y_P}$$

- The intersection of this line with the curve is $R = (x_R, y_R)$

$$\begin{aligned} x_R &= m^2 - x_P - x_Q \\ y_R &= y_P + m(x_R - x_P) \end{aligned}$$

$P + Q = -R$ → $(x_P, y_P) + (x_Q, y_Q) = (x_R, -y_R)$

Example

$$y^2 = x^3 - 7x + 10$$

P=(1,2) *Q*=(3,4) : P+Q=-R=?

$$\begin{aligned} m &= \tfrac{y_P - y_Q}{x_P - x_Q} = \tfrac{2-4}{1-3} = 1 \\ x_R &= m^2 - x_P - x_Q = 1^2 - 1 - 3 = -3 \\ y_R &= y_P + m(x_R - x_P) = 2 + 1 \cdot (-3 - 1) = -2 \end{aligned}$$

➔ $P + Q = -R = (-3, 2)$

"×" operation on ECs

$$nP = \underbrace{P + P + \cdots + P}_{n \text{ times}}$$

n is a natural number.

Written in that form, it may seem that computing nP requires n additions. If n has k binary digits, then this algorithm would be O(2^k), which is not really good. But there exist faster algorithms. One of them is the **double and add** algorithm -> O(log n).

O(2^k)=O(n)

Double and Add Algorithm

- It is of O(log n)
- Or O(k)

the **double and add** algorithm. Its principle of operation can be better explained with an example. Take $n = 151$. Its binary representation is 10010111_2. This binary representation can be turned into a sum of powers of two:

$$\begin{aligned} 151 &= 1 \cdot 2^7 + 0 \cdot 2^6 + 0 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 \\ &= 2^7 + 2^4 + 2^2 + 2^1 + 2^0 \end{aligned}$$

(We have taken each binary digit of n and multiplied it by a power of two.)

In view of this, we can write:

$$151 \cdot P = 2^7 P + 2^4 P + 2^2 P + 2^1 P + 2^0 P$$

What the double and add algorithm tells us to do is:

- Take P.
- *Double* it, so that we get $2P$.
- *Add* $2P$ to P (in order to get the result of $2^1 P + 2^0 P$).
- *Double* $2P$, so that we get $2^2 P$.
- *Add* it to our result (so that we get $2^2 P + 2^1 P + 2^0 P$).
- *Double* $2^2 P$ to get $2^3 P$.
- Don't perform any addition involving $2^3 P$.
- *Double* $2^3 P$ to get $2^4 P$.
- *Add* it to our result (so that we get $2^4 P + 2^2 P + 2^1 P + 2^0 P$).
- ...

Why is it complex to find *n* from *nP* ? *(see the video)*

Elliptic curves in $\mathcal{F}_p$

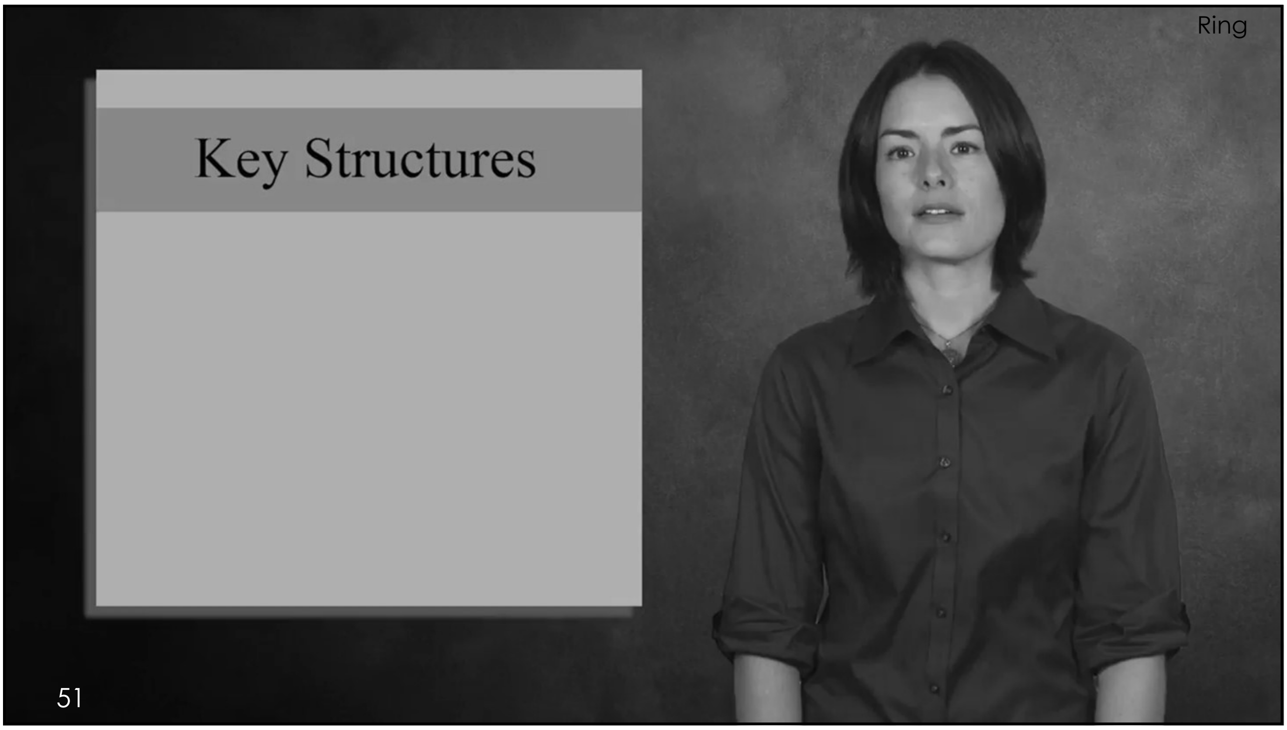
➨Usually we restrict elliptic curves over $\mathcal{F}_p$. Originally the set of points on EC are:

$$\{(x, y) \in \mathbb{R}^2 \quad | \quad y^2 = x^3 + ax + b, \\ 4a^3 + 27b^2 \neq 0\} \ \cup \ \{0\}$$

which is restricted to $\mathcal{F}_p$ as:

$$\{(x, y) \in (\mathbb{F}_p)^2 \quad | \quad y^2 \equiv x^3 + ax + b \pmod{p}, \\ 4a^3 + 27b^2 \not\equiv 0 \pmod{p}\} \ \cup \ \{0\}$$

where 0 is the identity element of + operation, and a and b are two integers in $\mathcal{F}_p$.


Ring
Key Structures
51

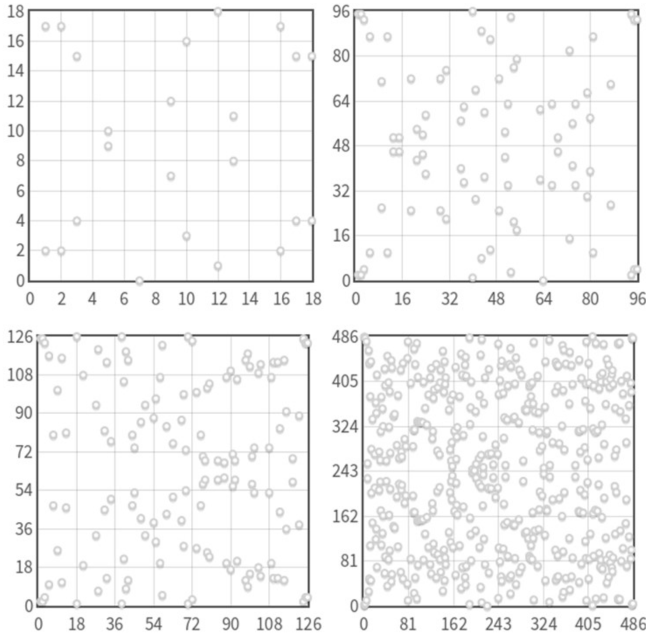

Field
52

Elliptic curves in $\mathcal{F}_p$

The set of EC points in $\mathcal{F}_p$ with $p = 19, 97, 127, 487$

The EC equation is:

$$y^2 = x^3 - 7x + 10$$



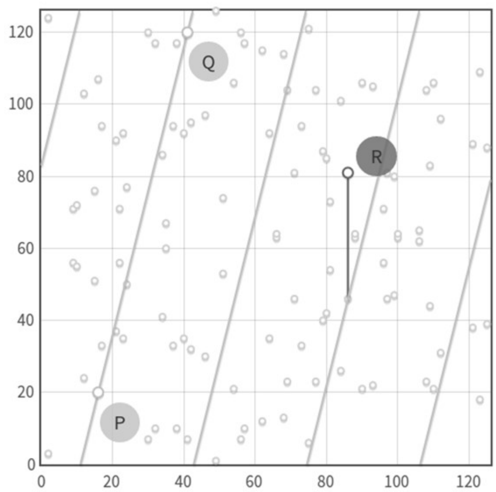
53

(Andrea Corbellini)

Lecture Notes-Sayad

Elliptic curves in $\mathcal{F}_p$

Clearly, we need to change a bit our definition of addition in order to make it work in $\mathcal{F}_p$. With reals, we said that the sum of three aligned points was zero. We can keep this definition. Now, we can say that three points are aligned if there's a line that connects all of them. Of course, lines in $\mathcal{F}_p$ are not the same as lines in R. We can say, informally, that a line in $\mathcal{F}_p$ is the set of points that satisfy the equation $ax + by + c \equiv 0 \; mod \; p$ (this is the standard line equation, with the addition of "$mod \; p$").



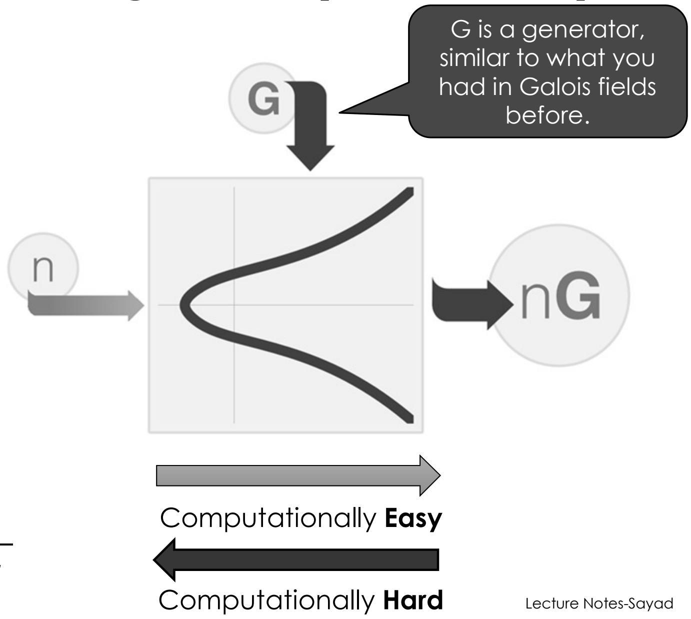
54

Lecture Notes-Sayad

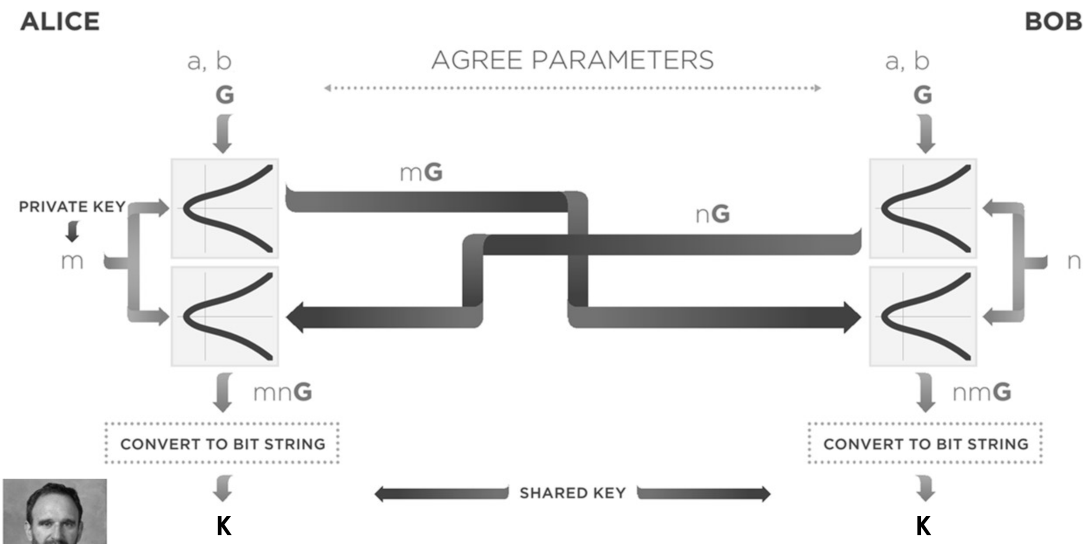
Elliptive Curves Discrete Logarithm (one-way F)

- Given the point **G** and the EC, calculating **nG** is easy.
- But if one gives you the point **W=nG**, it's hard to find **n**, even if you have **G**.

GF		ECC
g^n	$\approx$	nG



Elliptic Curve Diffie-Hellman Key Agreement (ECDH)



(Entrust Datacard)

Elliptic Curve Digital Signature

For Alice to sign a message m, she follows these steps:

1. Calculate $e = \mathrm{HASH}(m)$. (Here HASH is a cryptographic hash function, such as SHA-2, with the output converted to an integer.)
2. Let z be the L_n leftmost bits of e, where L_n is the bit length of the group order n. (Note that z can be *greater* than n but not *longer*.[1])
3. Select a **cryptographically secure random** integer k from $[1, n-1]$.
4. Calculate the curve point $(x_1, y_1) = k \times G$.
5. Calculate $r = x_1 \mod n$. If $r = 0$, go back to step 3.
6. Calculate $s = k^{-1}(z + rd_A) \mod n$. If $s = 0$, go back to step 3.
7. The signature is the pair (r, s). (And $(r, -s \mod n)$ is also a valid signature.)

Elliptic Curve Digital Signature

After that, Bob follows these steps:

1. Verify that r and s are integers in $[1, n-1]$. If not, the signature is invalid.
2. Calculate $e = \mathrm{HASH}(m)$, where HASH is the same function used in the signature generation.
3. Let z be the L_n leftmost bits of e.
4. Calculate $u_1 = zs^{-1} \mod n$ and $u_2 = rs^{-1} \mod n$.
5. Calculate the curve point $(x_1, y_1) = u_1 \times G + u_2 \times Q_A$. If $(x_1, y_1) = O$ then the signature is invalid.
6. The signature is valid if $r \equiv x_1 \pmod{n}$, invalid otherwise.

Attacks on Public-key Cryptosystems

Some Notes

- Factoring, Discrete Logarithm, Knapsack, … are all hard mathematical problems.
  - We build our public-key cryptosystems based on these.
- These are called NP (non-deterministic polynomial) problems, since they cannot be solved by any algorithm whose run-time complexity is of the order of a polynomial.
  - Polynomial example: if the problem space is as big as **n**, the number of operations required to solve the problem is something like **n^3+5n**.
  - If it's like **2^n** , then it's non-polynomial.

Some Notes

- It's been proven that NP (complete) problems can be converted to each other.
  - So if one hard problem is solved, the others are also solved!

Interested? → read "Theory of Complexity“

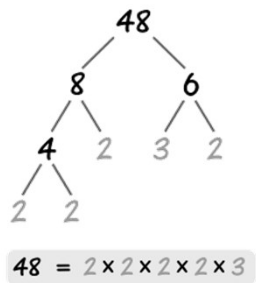
We only focus on RSA and factoring problem ....

61

Lecture Notes-Sayad

RSA Cryptanalysis

- **Brute-force (Trial Division)**
  - We know n=pq
  - Try every prime number smaller than $\sqrt{n}$ to find p or q



62

Lecture Notes-Sayad

RSA Cryptanalysis

For RSA, never choose p or q whose p-1 or q-1 have small prime factors

➨ Pollard's p-1 method

We know $a^{\varphi(n)} \overset{n}{\equiv} 1$, right? (of course if gcd(a,n)=1)

$$n = p \;\rightarrow\; \varphi(n) = p-1 \;\rightarrow\; a^{p-1} \overset{n}{\equiv} 1 \;\rightarrow\; a^{p-1} - 1 \bmod n = 0$$

RSA: $n = pq$

If p-1 (or q-1) has small prime factors (e.g. smaller than B), then:

$F = 2^{E_2} \times 3^{E_3} \times 5^{E_5} \times 7^{E_7} \times \cdots \times {q_n}^{E_n}$ $(q_i^{E_i} \le B)$

If we choose big enough E_i values (but not too big to exceed B), then $a^F - 1 \bmod p = 0$. Now, if we calculate $\gcd(n, a^F - 1)$, we have found p !!! (more precisely, we check whether $\gcd(n, a^F - 1)$=1 or not (>1) which is a P-time decision problem using the Euclidean algorithm).

RSA Cryptanalysis

Other RSA attacks:

- ➨**Pollard's ρ method**
- ➨**Elliptic Curve Method**
- ➨**Quadratic Sieve and Number Field Sieve Methods**
- ➨**Low Private Exponent Attack**
- ➨**Partial Key Exposure Attack**
- ➨**Broadcast and Related Message Attack**
- ➨**Short Pad Attack**
- ➨**Side Channel Analysis (power, timing, fault)**

cryptanalysis-rsa-survey-1006.pdf

reference

End of Part 2

Assignment 1

65

Lecture Notes-Sayad

The End of Cryptography

More might be coming, subject to time availability

66

Lecture Notes-Sayad