



# Computer Security and Cryptography

**CS381**

来学嘉

计算机科学与工程系 电院3-423室

34205440 1356 4100825 laix@sjtu.edu.cn

2016-04



## Organization

- Week 1 to week 16 (2016-02-24 to 2016-06-08)
- 东上院502
- Monday 3-4节; week 9-16
- Wednesday 3-4节; week 1-16
- lecture 10 + exercise 40 + **random tests** 40 + other 10
- Ask questions **in** class – counted as points
- Turn ON your mobile phone (after lecture)
- Slides and papers:
  - <http://202.120.38.185/CS381>
  - **computer-security**
  - <http://202.120.38.185/references>
- TA: '薛伟佳' icelikejia@qq.com, '黄格仕' <huang.ge.shi@foxmail.com>
- Send homework to: [laix@sjtu.edu.cn](mailto:laix@sjtu.edu.cn) and to TAs

**Rule: do not disturb others!**



## Contents



- **Introduction** -- What is security?
- **Cryptography**
  - Classical ciphers
  - Today's ciphers
  - Public-key cryptography
  - Hash functions/MAC
  - **Authentication protocols**
- **Applications**
  - Digital certificates
  - Secure email
  - Internet security, e-banking
- Network security**
  - SSL
  - IPSEC
  - Firewall
  - VPN
- Computer security**
  - Access control
  - Malware
  - DDos
  - Intrusion
- Examples**
  - Bitcoin
  - Hardware
  - Wireless

3



## Authentication



- **Authentication**
  - The provision of assurance of the claimed identity of an entity. [ISO]
- One of 2 main goals of cryptography:
  - Authenticity: "who **wrote** the data"
  - Confidentiality: "who can **read** the data"

4

4



# Components of Authentication



system: set of users, protocols

1. Claim identity: Alice
2. Submit authentication data by A
  - $A \rightarrow B: M$
3. Verification by B
  - $M \in \{ M_A, \dots \} ?$
4. Conclusion of B
  - accept, reject

5

5



# Authentic message



- Set of system users:  $U=\{A,B,\dots\}$
- **Authentic messages:**  $\{M_A, A \in U\}$ 
  - Only legitimate users can have generated the message
  - $M_A = (f_A(X), X)$ ,
    - $f_A$  : keyed 1-way function with A's secret key, e.g., MAC, cipher, signature.
- Verification: check the correctness of  $f_A(X)$ .
- Conclusion: after B verifying  $M \in \{ M_A, A \in U \}$  ,
  - If  $f$  is cipher or MAC, then  $U=\{A,B\}$ , B accepts A because B didn't produce M.
  - If  $f$  is signature,  $U=\{A\}$ .
  - B accepts A:
    - A produced the message (authentic)
    - A has sent the message (freshness) ??

6

6



# Authentic message: MAC



- **MAC** - shared secret key  $k$ 
  - Send:  $M, C_k(M)$  //
  - verify computed  $C_k(M) = \text{received } C_k(M)$
- **Security** of MAC:
  - If the key  $k$  is unknown, it is difficult to find a new message with a valid MAC, even if many valid  $(M, C_k(M))$  are known.
- Only users knowing the key can generate and verify the MAC. (symmetric)



# digital signature



- **RSA**
    - Parameters  $PK=\{e,n\}$  ,  $SK=\{d,p,q\}$
- Alice**  
 $S \equiv H(M)^{d_A} \pmod{n_A}$

$(M, S)$

**Bob**  
 $H(M) ? \equiv S^{e_A} \pmod{n_A}$
- only Alice can generate  $S$  (asymmetric)
  - **ElGamal** Signature
    - Alice: pri-key  $x_a$ ; pub-key  $y_a=g^{x_a}$
    - Bob: pri-key  $x_b$ ; pub-key  $y_b=g^{x_b}$
    - **Signing**
      - Alice random  $r$ ,  $\gcd(r, p-1)=1$ , and gets  $R=g^r$
    - Send:  $(m, R=g^r, S=r^{-1}(m-x_aR) \pmod{p-1})$
    - **Verification**:  $g^m=y_a^R R^S \pmod{p}$



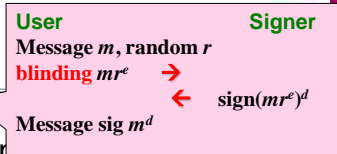
# Digital Signature Algorithm (DSA)



- NIST Digital Signature Standard (**DSS**), FIPS 186 (1991)
- **320-bit** signature; with 512-1024 bit security
- signature only, variant of ElGamal & **Schnorr** schemes
- **system** public key (p,q,g):
  - large prime **p** (512-1024 bits) ; Small prime **q** (160 bits),  $q \mid (p-1)$
  - $g = h^{(p-1)/q}$ ,  $1 < h < p-1$ ,  $h^{(p-1)/q} \bmod p > 1$
  - Users: private key  $x < q$ , public key:  $y = g^x \bmod p$
  - Sign**: one-time random signature key  $k$ ,  $k < q$   
 $r = (g^k \bmod p) \bmod q$   
 $s = [k^{-1}(H(M) + xr)] \bmod q$
- **Send**: (M, r, s)
- **verification**  
 $u1 = [H(M) s^{-1}] \bmod q$  ;  $u2 = (r s^{-1}) \bmod q$   
**verify**  $r = [(g^{u1} y^{u2}) \bmod p] \bmod q$



# different signatures



- **Blind** signature : content of a message is unknown to the signer. publicly verifiable.
  - Untraceable ----voting systems and digital cash
- **Undeniable** signatures: signer can choose who is allowed to verify
- **Group** signature: a member of a group to sign a message on behalf of the group anonymously.
  - **Ring** signature: without group manager
- **Threshold** signature: Need >t members to sign.
- **Proxy** signature : signer can delegate the signing power to a proxy (short period)
- **Attribute** signature –signing power varies according to identity-role.....



# Authentication protocols



•**Protocol**: A series of specified actions taken by specified 2 or more entities.

A protocol specifies how to use cryptographic primitives (encryption, signature...) to provide security services (ex. authentication)



# Security



|                        |   |
|------------------------|---|
| Name                   | example   |
| applications           | Email, payment, PGP, VPN,   |
| services               | Confidentiality, authenticity, integrity, non-repudiation, access control |
| Protocols              | DH, SSL, SSH, IPSEC, Kerberos, secret-sharing, ID-based..,                |
| Mechanisms (standards) | Encryption, signature, authentication, key-exchange, non-repudiation      |
| Primitives             | Encryption, signature, hash, MAC, RNG,                                    |
| algorithms             | DES, AES, RSA, DH, MD5, SHA, ElGamal,                                     |
| theory                 | Math, IT, Number theory, cryptography, complexity                         |



## Example 1 - password



- Password
  - $(A \rightarrow B)$ : Id=Alice
  - $(B \rightarrow A)$ : proof?
  - $(A \rightarrow B)$ : (password)
  - B: check (password)=stored password ?  
If yes, accept A as Alice.
- Attack by **replay**
  - If enemy intercepted the password, he can reuse it to pretend to be Alice

13

13

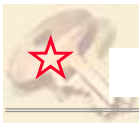


## Freshness mechanisms

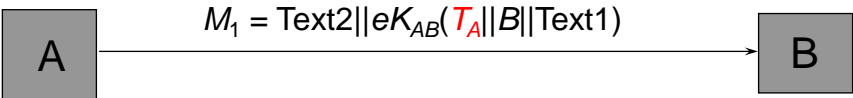


- Authenticity checking is not enough - also need means of checking '**freshness**' of **authentic messages**, to protect against **replays**.
- Two main methods:
  - use of time-stamps (**clock**-based or '**logical**' time-stamps),
  - use of 'nonces' or challenges (as in challenge-response protocols).

14



## Example 2. use time-stamp & encryption)



Clause 5.1.1 of ISO/IEC 9798-2.

- use time-stamps  $T_A$  for **freshness**
- $eK_{AB}$  encryption with shared key  $K_{AB}$  for origin and integrity checking.
- provides *unilateral authentication* ( $B$  can check  $A$ 's identity, but not vice versa).
- Requires **securely** synchronised clocks; Non-trivial to provide such clocks
- need time acceptance 'window' because of clock variations and delays.
- Acceptance window allows for undetectable replays - hence need to store a log of recently received messages.

15



## Logical time - counter



- $A$  authenticate to  $B$ :
  - $A$  maintains counter  $N_A$ , and  $B$  has  $N_B$ ,
- $A$  sends  $B$ :  $f(N)$ , ( $N > N_A$ ) and set  $N_A = N$ .
- $B$  check
  - $f(N)$  is authentic; and:
  - if  $N > N_B$  then  $B$  accept, and set  $N_B = N$ ,
  - if  $N \leq N_B$  then the message is rejected.

16





### Example 3: e-banking



User input:  
acc. number  
Password  
list number

Bank check  
acc. number  
Password  
the numbers stored

Then remove the  
number from the list

•require synchronization,  
thus only suitable in well-  
managed systems.

Karte gültig ab 17.10.2006

|         |         |          |         |          |
|---------|---------|----------|---------|----------|
| 51 8QBN | 61 YBHB | 71 TRPB  | 81 KCBF | 91 HEQL  |
| 62 8TD3 | 62 AE97 | 72 L6CQ  | 82 M747 | 92 VWTP  |
| 53 TXN  | 63 5CWB | 73 UZMK  | 83 XU23 | 93 SXBE  |
| 64 CGJT | 64 X5U2 | 74 PERX  | 84 4SKB | 94 VNY7  |
| 55 ARMM | 65 AYQE | 75 2KWA  | 85 V9HG | 95 6QXU  |
| 56 ASZV | 66 7X9P | 76 BSY5  | 86 B7KV | 96 6XJG  |
| 57 PSTB | 67 S5V5 | 77 J6G5  | 87 9VBA | 97 UNH2  |
| 58 RNXP | 68 V96J | 78 EWA4B | 88 F6CR | 98 LGHN  |
| 59 X4WU | 69 5RNU | 79 JRSE  | 89 Y7NS | 99 3SP8  |
| 60 VKCW | 70 Q2G9 | 80 27CN  | 90 ZPUX | 100 4HKZ |

17

17



### 电子银行口令卡



- use 2 numbers each time (A1,C8)
- 80X79/4 choices

图1 中国工商银行的电子银行口令卡

中国工商银行、中国建设银行的电子口令卡的使用次数、支付限额

|            | 是否有<br>口令卡 | 使用次<br>数 | 借记卡支付限<br>额                | 信用卡支付限额  |
|------------|------------|----------|----------------------------|--|
| 中国工<br>商银行 | ✓          | 1000次    | 单 笔：1000元<br>日累计：5000<br>元 | 单 笔：1000元与信用卡本身限额<br>相比低者<br>日累计：5000元与信用卡本身限<br>额相比低者 |



## Example 4: time – secureID



One-time password, change every 60 sec.



Online Banking with Direct Net

Login

|                    |                                     |
|--------------------|-------------------------------------|
| User ID            | <input type="text" value="123456"/> |
| Password           | <input type="password"/>            |
| SecureID/trike Net | <input type="text" value="147462"/> |

[Direct Net Info](#) [Demo](#)

[Approved Incoming](#)

[Reset](#) [Login](#)

Who you are  
What you know  
What you have

User supply:  
Acc. number  
Password  
**SecureID number**

Bank check  
acc. Number  
Password  
**the numbers  
computed from  
local time**

- $SID=h(\text{userID}, \text{key}, T_0)$
- $T_0 \in [T_0-a, T_0+b]$



## Example 4: nonces – secureID



One-time password, change every 60 sec.



Online Banking with Direct Net

Login

|                    |                                     |
|--------------------|-------------------------------------|
| User ID            | <input type="text" value="123456"/> |
| Password           | <input type="password"/>            |
| SecureID/trike Net | <input type="text" value="147462"/> |

[Direct Net Info](#) [Demo](#)

[Approved Incoming](#)

[Reset](#) [Login](#)

Who you are  
What you know  
What you have

User supply:  
Acc. number  
Password  
SecureID number

Bank check  
acc. Number  
Password  
the numbers stored

- $SID=h(\text{userID}, \text{key}, N) \ N>N_0$
- Hash, AES



## Example 4: nonces–challenge/response



- Who you are --- name/account number
- What you know --- password
- What you have --- device generating valid response

21



## 2 basic elements in authentication protocols



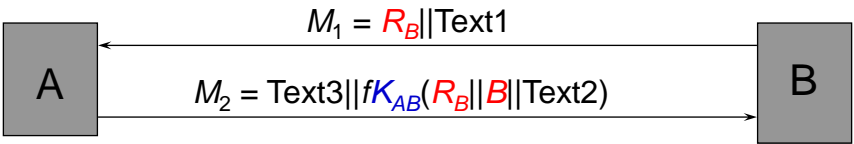
- **Authentic message**
  - a message that the receiver can verify that it can only be originated by the sender.
- **Freshness** of the authentic message:
  - To prevent “replay” attack by using the previously used authentic message.

22

22



### Example 5 (nonce & integrity mechanism)



clause 5.1.2 of ISO/IEC 9798-4.

- use of nonces  $R_B$  (for freshness) and MAC for origin and integrity checking.

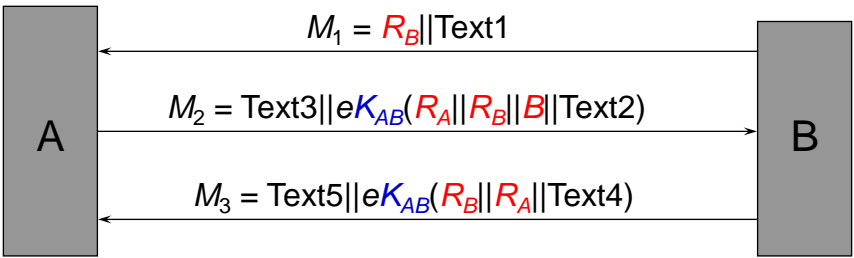
It provides *unilateral authentication* ( $B$  can check  $A$ 's identity)

$fK_{AB}$  denotes a cryptographic check (MAC) function with shared key  $K_{AB}$

This is a **challenge-response** protocol



### Example 6 (nonce & encryption)



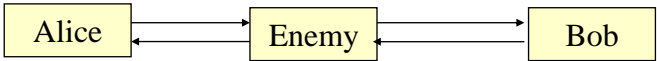
clause 5.2.2 of ISO/IEC 9798-2.

use nonces (for freshness) and encryption (for origin and integrity checking).

It provides **mutual authentication**



# Model



Model for authentication.

- 3 parties: Alice, Bob and Enemy
- All communication between A and B are under the control of Enemy (read, relay, modify, insert)
- Assumption: crypto-algorithms (cipher, MAC, hash..) used in the protocols are secure, so we concentrate on protocol.
- **Protocol:** A series of specified actions taken by specified 2 or more entities.

25

25



# Examples



- **Password.**  $(A \rightarrow B): (Alice, \text{password})$ 
  - Enemy can **replay** the message.
- **Timestamp.**  $((A \rightarrow B)\text{-authentic message})_{time}$ 
  - require universal clock
- **Serial number.** n-th message is  $((A \rightarrow B)\text{-authentic message})_n$ 
  - require synchronization
- **Random number** (nonces)
  - challenge  $B \rightarrow A: C$
  - response  $A \rightarrow B: f(C)$

26

26



## Key-Exchange protocol



- In most cases, only authentication is not enough.
- it is often used to establish a shared key ("session key")
- this session key is used to protect the real application.
- Security requirements

1. **Authenticity**: they both know who the other party is

2. **Secrecy**: only they know the resultant shared key

Also crucial (yet easy to overlook):

3. **Consistency**: if two honest parties establish a common session key then both have a consistent view of who the peers to the session are

$$A: (B, K) \text{ and } B: (x, K) \rightarrow x=A$$

*One description of secure key exchange protocol [Krawczyk]*

27



## Key management standards



- ISO SC27 generic Key management standard: 11770.
- US banking community - ANSI X9.17, X9.24, 9.28, X9.30, X9.31.
- ISO TC68, banking standards committee for ISO, leading to ISO 8732 ( $\approx$  X9.17), ISO 11568, ISO 11649 ( $\approx$  X9.28) and ISO 11166 ( $\approx$  X9.30/9.31).
- IEEE P1363.2 (Specifications for Password-based Public Key Cryptographic Techniques, used in ISO 11770-4 )
- **Note: Key management is the most difficult part in use of cryptography**

28



# Diffie-Hellman Key Agreement



W.Diffie and M.E.Hellman, "New Directions in Cryptography", IEEE Transaction on Information Theory, V.IT-22.No.6, Nov 1976, PP.644-654



Parameters:  $p, g$

**Alice**

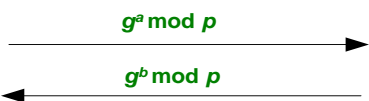
**Bob**

Choose  $a$

Choose  $b$

Compute  $g^a \bmod p$

Compute  $g^b \bmod p$



Compute  $g^{ab} \bmod p$

Compute  $g^{ab} \bmod p$

$g^{ab}$  is the secret key shared by Alice and Bob

29



# Man-in-the middle attack



Parameters:  $p, g$

**Alice**

**Bob**

Choose  $a$

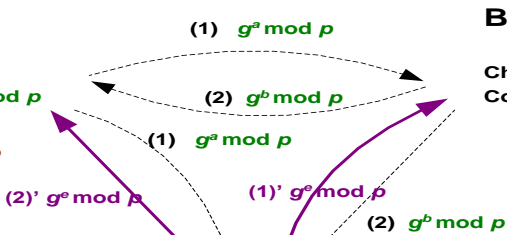
Choose  $b$

Compute  $g^a \bmod p$

Compute  $g^b \bmod p$

Compute  $g^{ae} \bmod p$

Compute  $g^{be} \bmod p$



**Eve**

Choose  $e$

Compute  $g^e \bmod p$

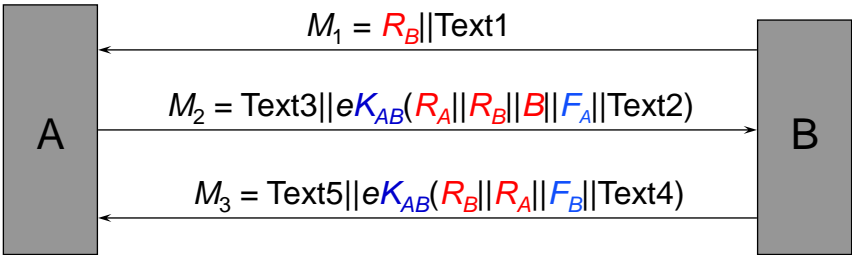
Compute  $g^{ae} \bmod p, g^{be} \bmod p,$

DH provide no authentication,  
is also called anonymous key agreement

30



## ISO 11770-2 mechanism 6



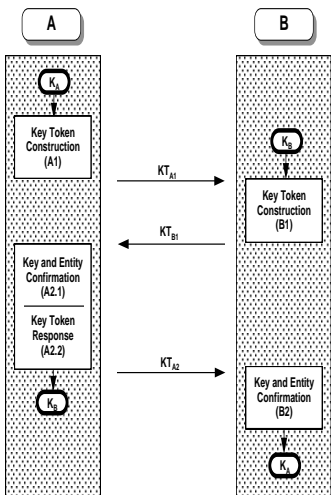
- $A, B$  share  $K_{AB}$  (master key)
- $R_A$  and  $R_B$  denote nonces, and  $F_A$  and  $F_B$  are keying material.
- The key  $K$  established between  $A$  and  $B$  is a non-invertible function of  $F_A$  and  $F_B$ .

clause 5.2.2 of ISO/IEC 9798-2. It provides mutual authentication

31



## ISO 11770-3: Key transport mechanism 6



$$KT_{A1} = E_B(A || K_A || r_A || \text{Text1}) || \text{Text2}$$
$$KT_{B1} = E_A(B || K_B || r_A || r_B || \text{Text3}) || \text{Text4}$$
$$KT_{A2} = r_B || \text{Text5}.$$

- Use public-key
- mutual authentication and implicit key authentication
- mutual key confirmation
- known as **COMSET**
- based on zero-knowledge techniques (clause 9.1 in 9798-5).





## Properties of ZK Proofs



Properties of ZK Proofs:

- completeness
  - prover** who knows the secret convinces the **verifier** with overwhelming probability (always accept)
- soundness (is a proof of knowledge)
  - no one who doesn't know the secret can convince the verifier with non-negligible probability (random guess,  $p=2^{-t}$ )
- zero knowledge
  - the proof does not leak any additional information (verifier can simulate the protocol)



## Fiat-Shamir ZK protocol



Fiat-Shamir ID protocol (ZK Proof of knowledge of square root modulo  $n$ )

- System parameter:  $n=pq$ ,
- Private authenticator:  $s$
- Public identity:  $v = s^2 \bmod n$
- Protocol (**repeat  $t$  times**)
  1. A: picks random  $r$  in  $Z_n^*$ , sends  $x=r^2 \bmod n$  to B
  2. B checks  $x \neq 0$  and sends random  $c$  in  $\{0,1\}$  to A
  3. A sends  $y$  to B, where If  $c=0$ ,  $y=r$ , else  $y=rs \bmod n$ .
  4. B accept if  $y^2 \equiv xv^c \bmod n$



## Properties of ZK Proofs



- completeness  
honest prover who knows the secret convinces the verifier with overwhelming probability (always accept)
- soundness (is a proof of knowledge)  
no one who doesn't know the secret can convince the verifier with non-negligible probability (random guess,  $p=2^{-t}$ ).  
Correct answers to both 0 and 1 implies knowing s.
- zero knowledge  
the proof does not leak any additional information (verifier can simulate the protocol):
  - Repeat the following: pick random  $c \in \{0,1\}$ ,
    - if  $c=0$ , pick random  $r$  and outputs  $(r^2, 0, r)$
    - if  $c=1$ , pick random  $y$ , and outputs  $(y^2v^{-1}, 1, y)$



## ZK Proofs



probability of forgery:  $1/2^t$

soundness (proof of knowledge):

- if A can successfully answer two challenges  $d_1$  and  $d_2$ , i.e., A can output  $D_1$  and  $D_2$  such that  $W=g^{D_1}G^{d_1}=g^{D_2}G^{d_2}$ , then  $g^{D_1-D_2}=G^{d_2-d_1}$  and thus the secret  $Q=(D_1-D_2)(d_2-d_1)^{-1} \bmod q$

zero knowledge (the proof does not leak any additional information):

Pick a random  $d$ , random  $D$ , let  $W=G^d g^D$ ,  
Outputs  $(W, d, D)$



## Key management with a trusted third party



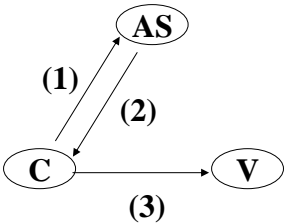
- Beside the 2-party protocols, we can use a trusted third party (**TTP**) to exchange keys
- Ex. a trusted Key Distribution Center (KDC)
  - each party shares own master key with KDC
  - KDC generates session keys used for connections between parties
  - master keys used to distribute these to them



## Denning AS Protocol



- (1)  $C \rightarrow AS: ID_C \parallel P_C \parallel ID_V$
- (2)  $AS \rightarrow C: Ticket$
- (3)  $C \rightarrow V : ID_C \parallel Ticket$



$Ticket = E_{K_V}[ID_C \parallel AD_C \parallel ID_V]$

- |   |  |
|---|--|
| <b>C</b> : client                               | <b>ID<sub>V</sub></b> : identifier of V                          |
| <b>AS</b> : Authentication Server               | <b>P<sub>C</sub></b> : password of user on C                     |
| <b>V</b> : server                               | <b>AD<sub>C</sub></b> : network address of C                     |
| <b>ID<sub>C</sub></b> : identifier of user on C | <b>K<sub>V</sub></b> : secret key shared between AS and server V |



## Key management and password



- Cryptographic keys are formed as binary digits
  - Symmetric: 128-bit
  - RSA,DL: 1024, 2048,..., bits
  - Elliptic curve: 256, 512,...,bits
- Human uses memorized password
  - 4-digit numbers
  - Text password
  - Pass phrases
- Vulnerable to brute-force attacks (guess, dictionary attack)
- Protection methods: policy, slow hash, restrict verification trials, CAPTCHA,...

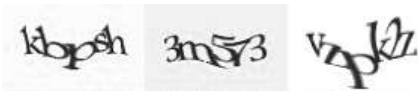


## CAPTCHA



- CAPTCHA (Completely Automated Public Turing Test to Tell Computers and Humans Apart)
  - a type of challenge-response test used in computing to ensure that the response is not generated by a computer.
  - A common type of CAPTCHA requires that the user type the letters or digits of a distorted image that appears on the screen.

- 验证码





## Secure use of password



- A: Password  $\pi$ , verifier B knows  $k=H(\pi)$
- A sends  $e_k(\text{data})$  to B, B check  $e_k(\text{data})$  .
  - Brute-force attack: guess  $\pi'$ , check  $e_{k'}(\text{data})$
  - Could be easier than breaking the cipher.
- Solution
  - B generates a public key  $p_B$ , send to A.
  - A send  $e_{p_B}(\pi, \text{nonce})$  to B
  - Brute-force attack becomes difficult (need to break the public-key cipher)
- ISO 11770-4, IEEE P1363.2



## Summary



- Authentication protocols
  - Authentic messages
    - MAC
    - signatures Math
  - Freshness mechanisms
    - Time / counter / Challenge-response
- Key-management
  - Protocols
  - password
- Next lecture: Kerberos, PKI



## Exercise 11 – authentication



1. 电子银行口令卡 : it has 8X10 numbers, each time we use 2 numbers, there are  $80 \times 79/2$  different choices in total. Question: why only  $80 \times 79/4$  are used? (Hint: compute the complexity of attack)
2. Design your own scheme for password choosing. Requirement: easy to use, change, remember and hard to break.
3. What would happen if the server V in Denning AS Protocol is compromised?
4. In example 5 and 6, there is a direction indicator (B). Find a reply attack if that B is removed, i.e., if the protocol is:
  - challenge  $B \rightarrow A: C$
  - response  $A \rightarrow B: f_{KAB}(C)$
  - Hint: re-direction

Deadline: before next lecture

Format: Subject: CS381-某某某-EX.#