Certificateless Public-Key Cryptography

Mohsen Toorani

Department of Informatics

University of Bergen

Norsk Kryptoseminar

November 9, 2011

Public-Key Cryptography (PKC)

- Also known as **asymmetric** cryptography.
- Each user has two keys: public and private.
- Alice's public key typically used for:
  - encryption to Alice by Bob.
  - verification of Alice's signatures by Bob.
- Alice's private key typically used for:
  - decryption by Alice.
  - signing by Alice.
- No need for Alice and Bob to share a common key before they begin secure communications!
  - Compare with **symmetric** key cryptography.

Public-Key Cryptography (PKC)

A significant problem in PKC is **verification of the authenticity of public keys**: Users must be assured that they cannot be fooled into using a false public key!
Solutions for authenticity of public keys:

1. Public-Key Infrastructure (PKI)
2. Identity-based Cryptography
3. Self-Certified Public-Key Cryptography
4. Certificate-based Public-Key Cryptography (CB-PKC)
5. Certificateless Public-Key Cryptography (CL-PKC)

1. Public-Key Infrastructure (PKI)

- PKI is a system for supporting deployment of PKC

- By the term “traditional PKI” we mean:
  - a combination of hardware, software and policies;
  - needed to deploy and manage **certificates**;
  - to produce trust in public keys;
  - used in a particular application or set of applications.

Digital Certificates

A **certificate** binds an entity with its public key.

The certificate is issued and signed by a **trusted** Certificate Authority (CA).

Certificate =

an **entity's description** (name, etc.)

+

entity's **public key**

+

expiration date, serial number, etc.

+

CA's name

+

a **signature issued by a CA**

Digital signature:

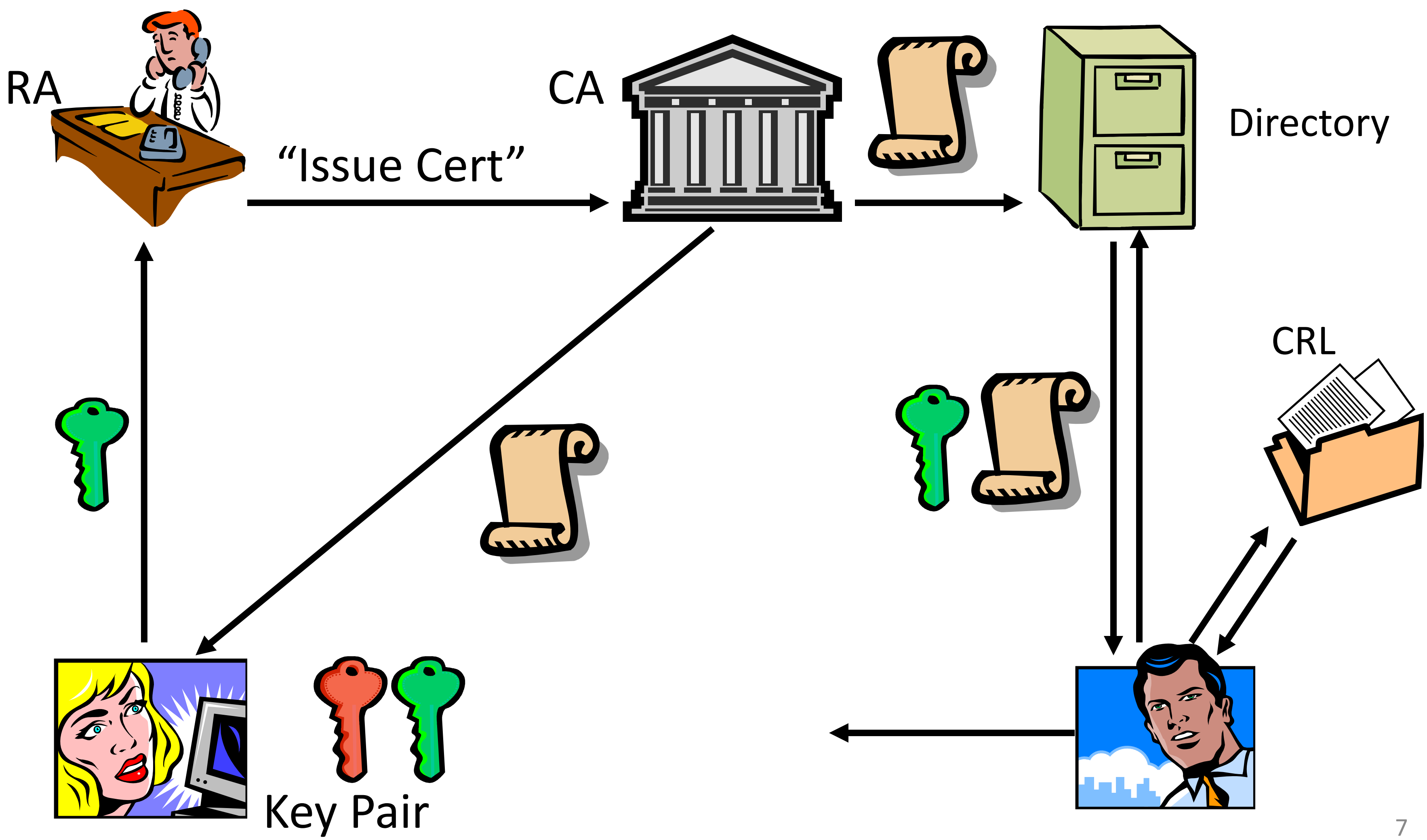
CA signature = certificate hash,

encrypted with CA's private key

PKI Components

- Registration Authority (RA)
  - Authenticates individuals/entities, optionally checks for possession of private key matching public key.
  - Passes off result to Certification Authority.
- Certification Authority (CA)
  - Issues certificates: CA issues signatures binding public keys and identities.
  - Relying parties need authentic copy of CA's public key…
- Directory Service
  - Directory of public keys/certificates.
- Revocation Service
  - May involve distribution of Certificate Revocation List (CRL) or on-line certificate status checking (OCSP).

Using PKI



Some PKI Problems

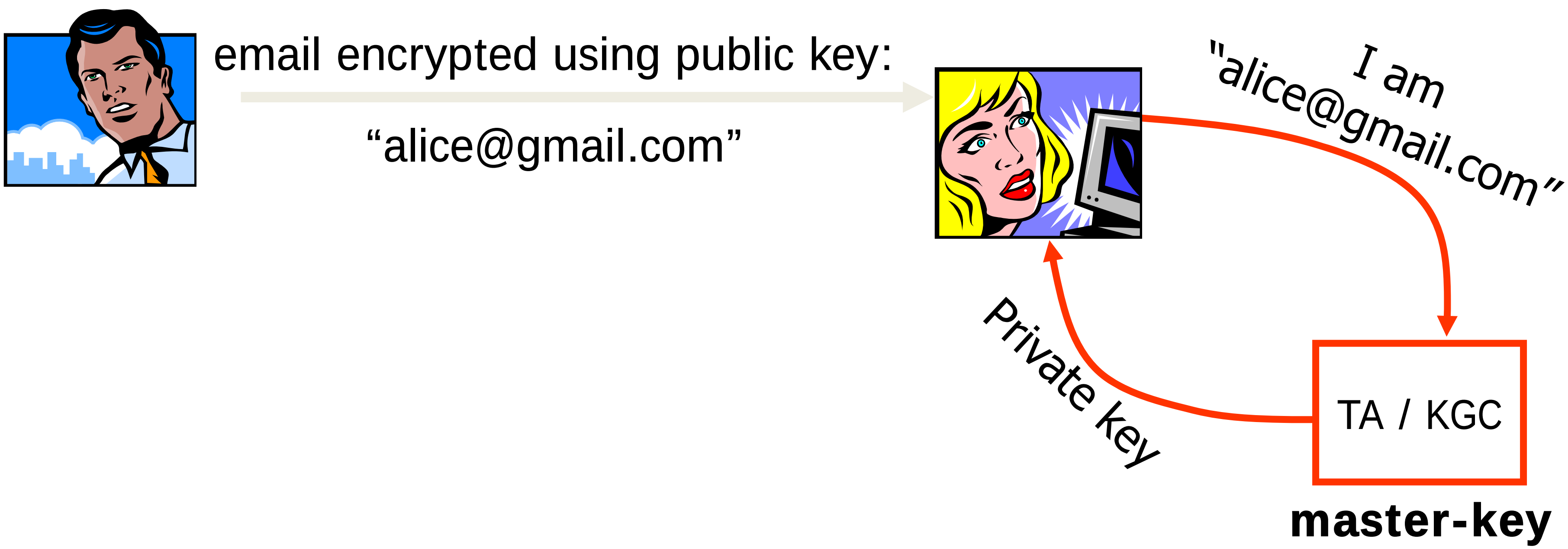
- Acute where consumers/end-user populations (humans) are involved.
- Legal and regulatory
- Interoperability and standards
- Costs and business models
- Some technical issues:
    - How is revocation to be handled?
    - How should the CA be designed and run?
    - How should keys and algorithms be managed?

Certificates and their management are the source of some problems.

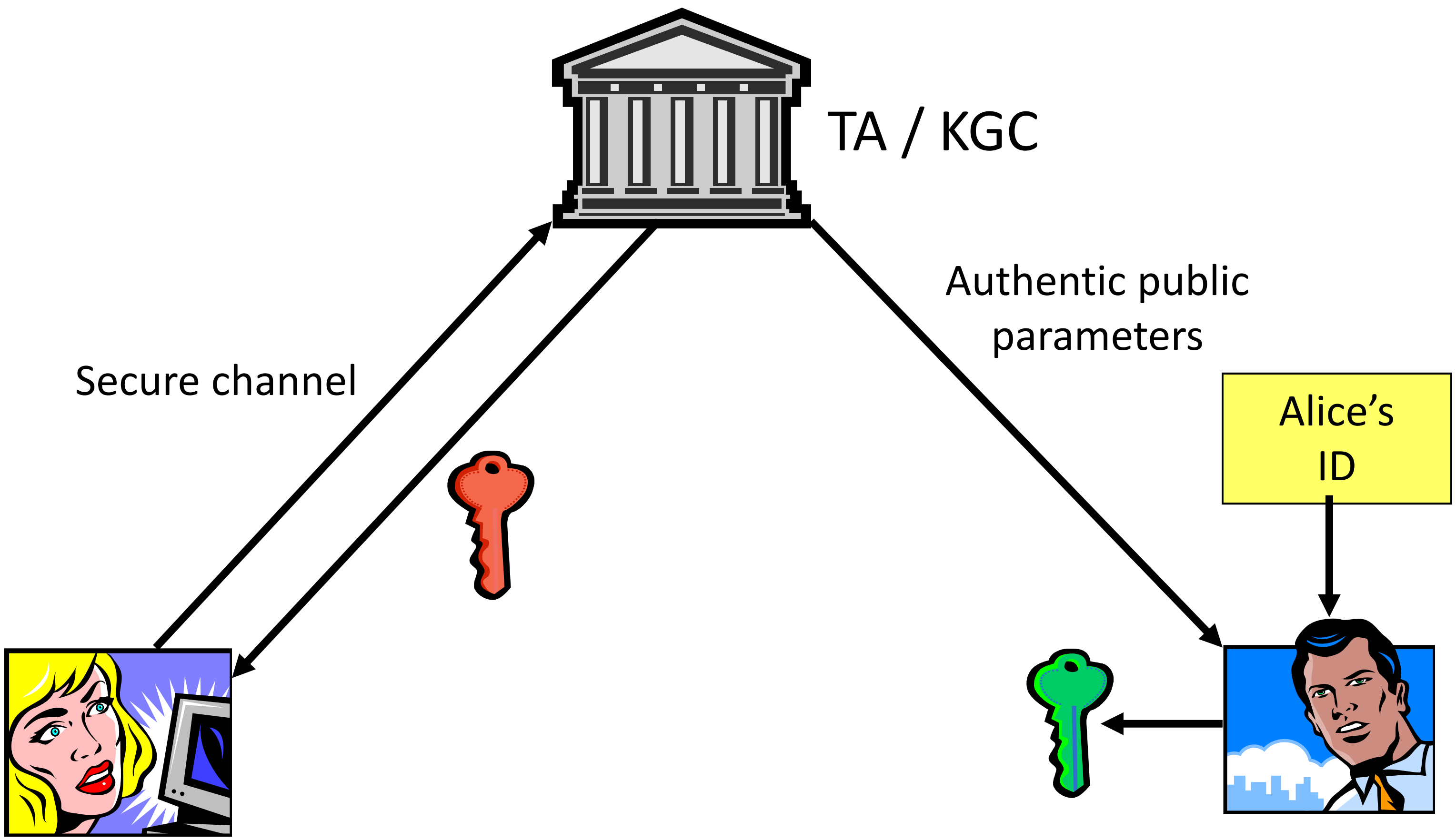
2. Identity-based Cryptography

- Public keys derived directly from system identities (e-mail address, mobile number, IP address, etc).
- The first idea due to Shamir (1984) but it was just an ID-based signature scheme.
- Construction of *practical* and *secure* ID-based encryption scheme was an open problem until 2001 when Boneh and Franklin (proposed in Crypto'01):
  - A Pairing-based IBE scheme, practical and provably secure.

2. Identity-based Cryptography



2. Identity-based Cryptography
(in Reality)



2. Advantages of ID-PKC

- **Certificate-free**
  - No production, checking, management or distribution of certificates.
- **Directory-less**
  - Bob can encrypt for Alice without looking-up Alice's public key first.
  - Alice need not have her private key when she receives Bob's encryption.
- **Automatic revocation**
  - Can extend identifier to include a validity period.
  - Alice's private key becomes useless at end of each period.
  - Alice needs to obtain private key for current period in order to decrypt new messages.
  - No need for CRLs or OCSP.
- **support for key recovery**
  - TA can calculate private key for any user.
  - May be needed e.g. when user leaves the organisation.
  - Enables applications like content scanning of e-mail at the server.

2. Disadvantages of ID-PKC

- **Effect of Catastrophic Compromise**: What is the cost of compromise of the master secret?
  - All past encrypted messages are exposed & all old signatures become worthless.
  - Potentially higher than cost of compromise of CA's signing key in PKI: CA in PKI can re-issue all certificates under new signing key without compromising clients' private keys.
- **Key Escrow**
  - TA can calculate all the private keys in the system.
  - We need to trust TA not to abuse this privilege.
  - PKI is more flexible in this respect.
- **Inability to Provide Non-repudiation**
  - Another consequence of key escrow.
  - TA *could* forge signatures if an ID-based signature were adopted.
    - So need to trust TA not to do that.
  - EU electronic signature legislation requires private key to be under "sole control" of signer in order for signatures to be fully recognised.
    - So It is incompatible with some legislative regimes.

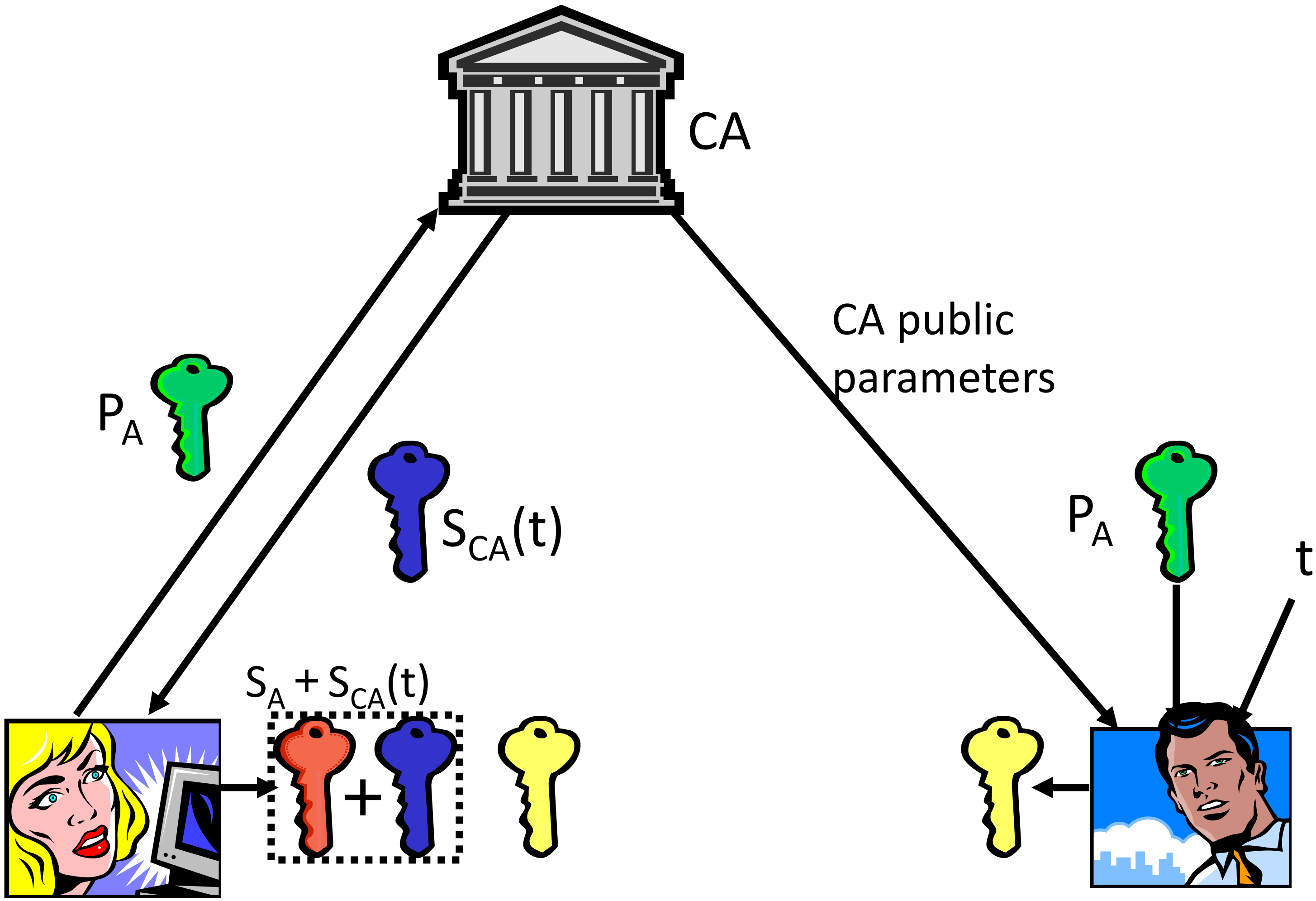
3. Self-Certified PKC

- Introduced by Girault (Crypto'91) to reduce storage and computation costs:
  - No key escrow
  - No need for hash functions in computing public keys
  - No need for a secure channel between CA and user.
- Users are associated with a 3-tuple (ID, s, P): (User's identity, User-chosen private key, the public key that doubles as a certificate).
- CA issues a certificate on ID, which is then used as the public key. (different from traditional PKI, where users have separate certificate validating their public keys.
- P cannot be immediately derived from ID (varies from ID-based schemes)

4. Certificate-based Public-Key Cryptography (CB-PKC)

- Introduced by Gentry (Eurocrypt 2003).
- Simplifies revocation in traditional PKIs.
- Alice's private key consists of two components:
  - The private part S_A of a "traditional" key pair (S_A, P_A).
  - A time-dependent certificate $S_{CA}(t)$ pushed to Alice on a regular basis by the CA, so long as Alice not revoked.
- Bob can compute a matching public key using only the CA's public parameters, time t and Alice's public component $P_{A.}$
- Bob is assured that Alice can only decrypt if the CA has issued certificate $S_{CA}(t)$ for the current time interval t.

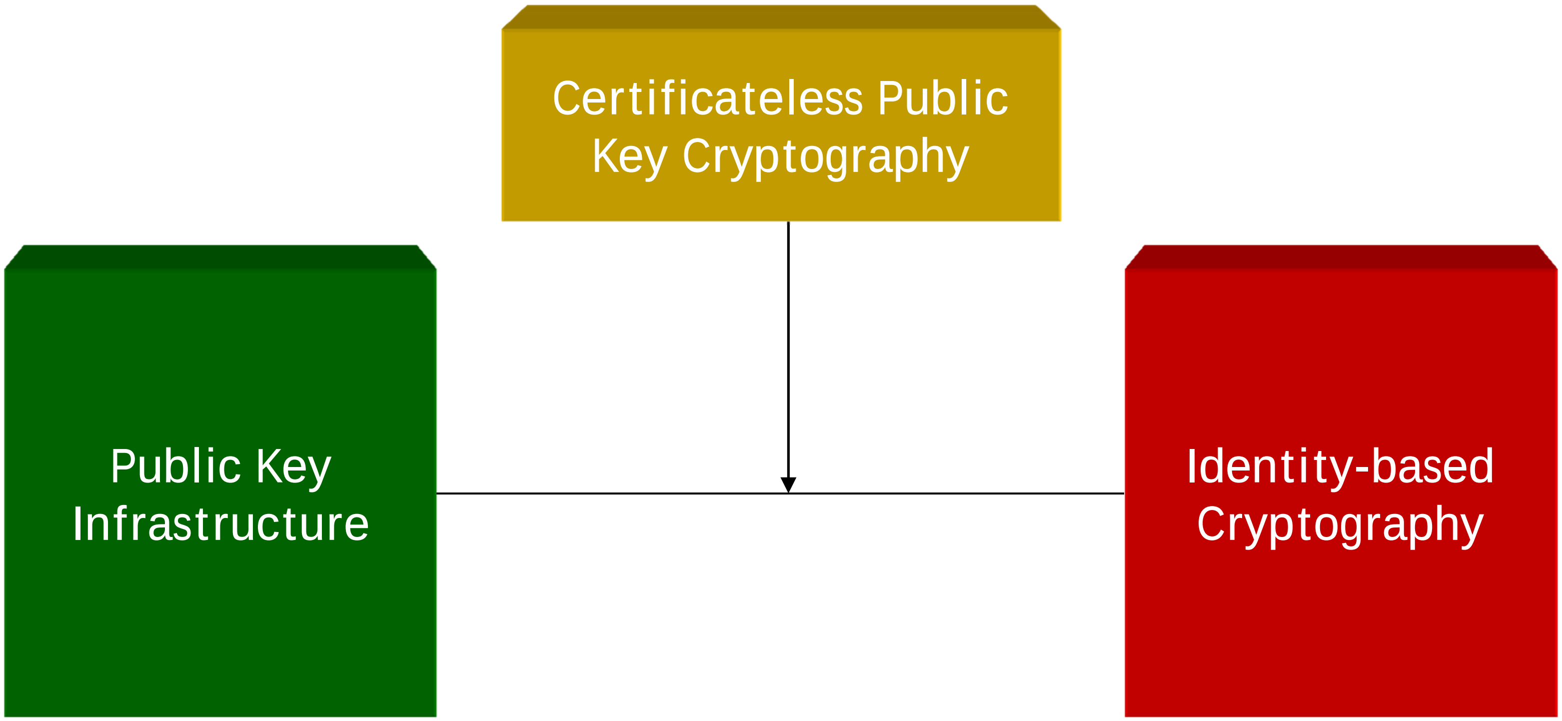
4. Certificate-based PKC (CB-PKC)



5. Certificateless Public-Key Cryptography (CL-PKC)

- Introduced by Al-Riyami and Paterson (Asiacrypt 2003).
  - A thriving sub-area of ID-PKC.
- Design objective:
  - Remove the key escrow problem of ID-PKC without introducing certificates.

CL-PKC



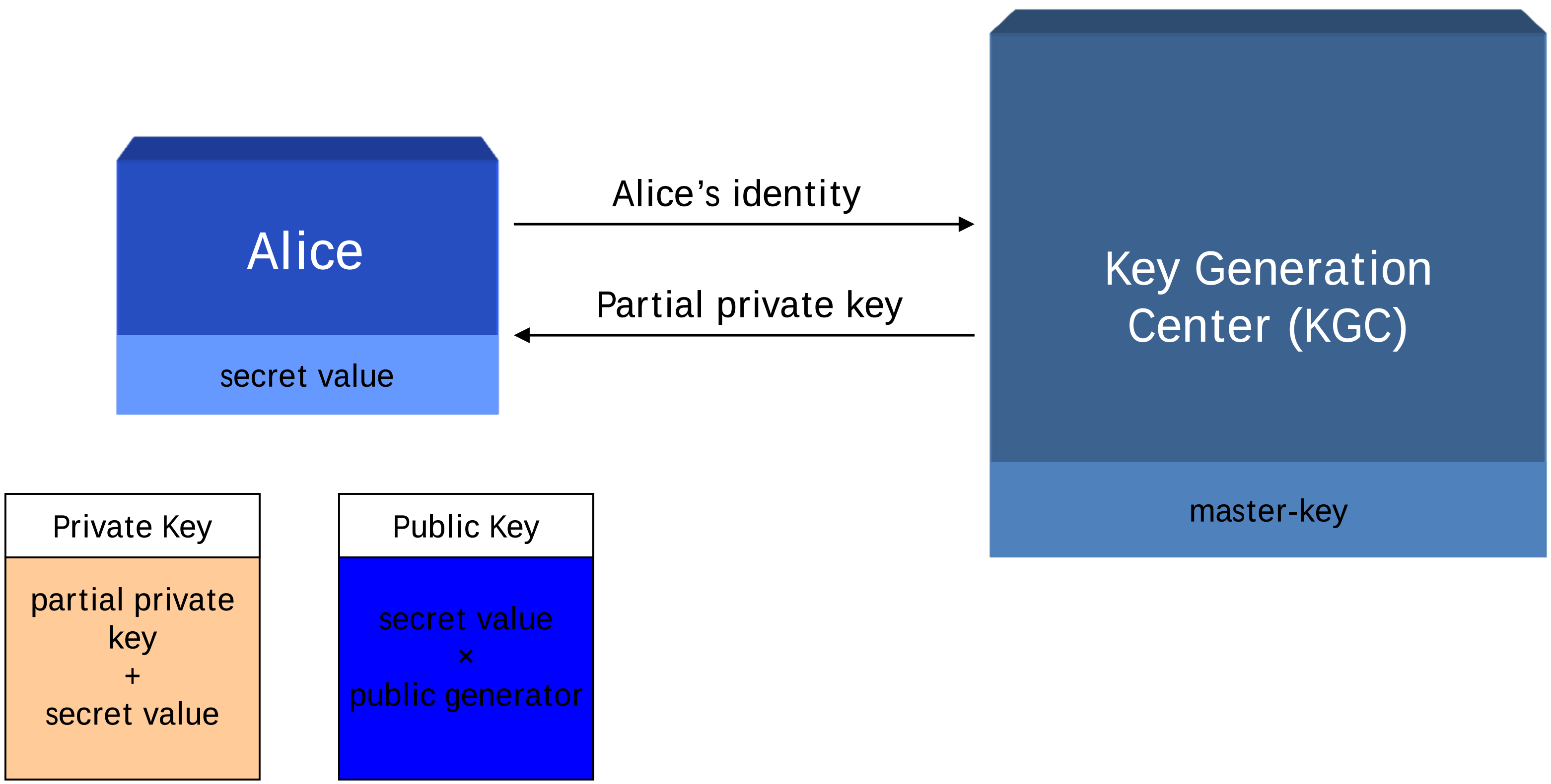
CL-PKC:

- A paradigm for generating trust in public keys.
- Lies midway between traditional PKI and ID-PKC in terms of trust model and functionality

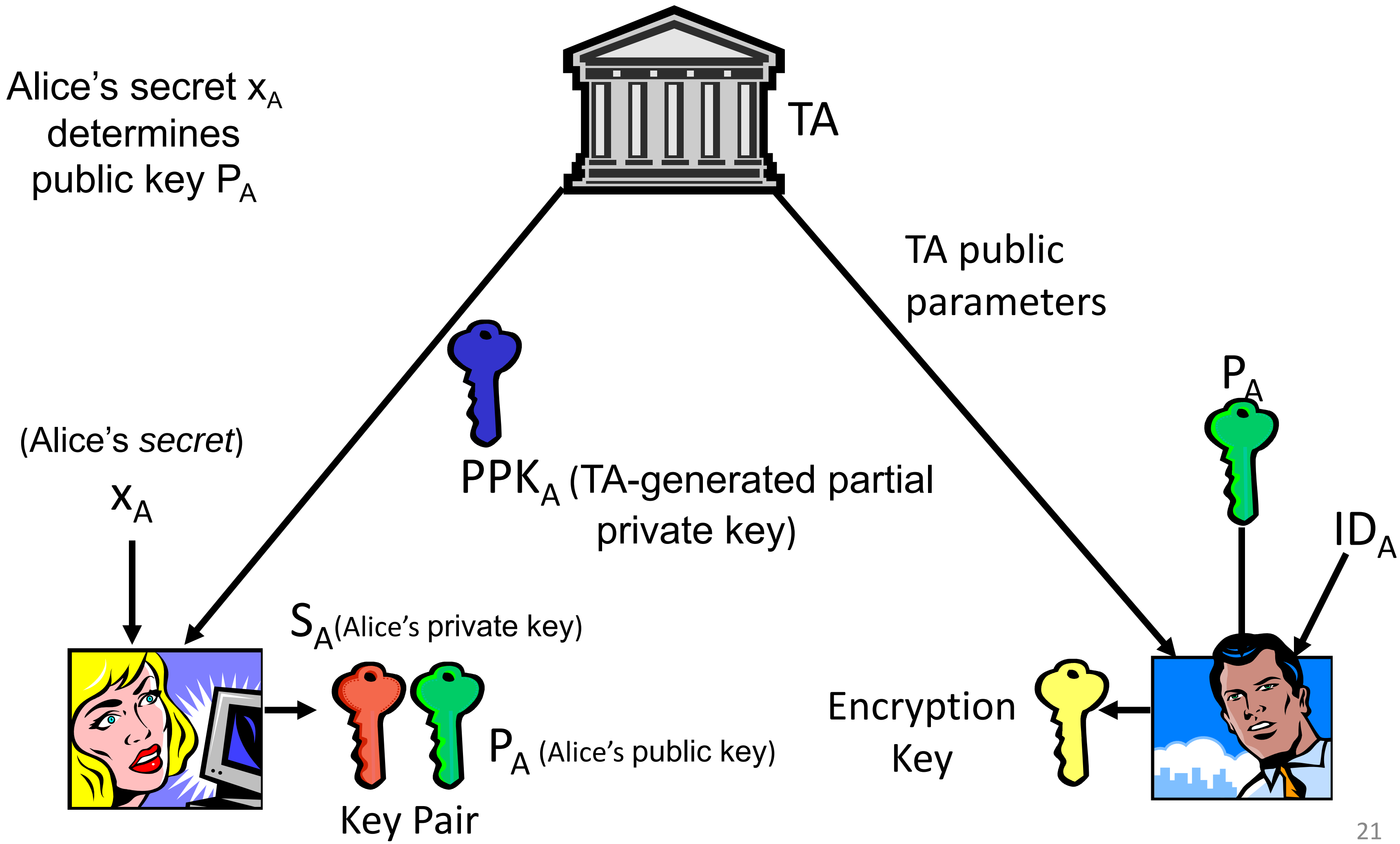
Why CL-PKC?

- No certificates used (PKI)
  - Low storage and communication bandwidth
  - No need to verify certificates (certificate chains)
  - Higher degree of privacy
- Public keys are always valid
  - No need for CRLs
- No key escrow (ID-PKC)
  - TA cannot recover session keys
  - TA cannot forge signatures

CL-PKC


Alice
secret value
Alice's identity
Partial private key
Key Generation Center (KGC)
master-key
Private Key
partial private key
+
secret value
Public Key
secret value
×
public generator

CL-PKE


Alice's secret x_A determines public key P_A
TA
TA public parameters
P_A
(Alice's *secret*)
x_A
PPK_A (TA-generated partial private key)
ID_A
S_A(Alice's private key)
Encryption Key
P_A (Alice's public key)
Key Pair

CL-PKE

- Each user generates its own public key from a randomly generated “secret value”.
- KGC provides a partial private key for a user’s identity.
- Encryption requires the user’s public key and the user’s identity.
- Decryption requires a private key based on the user’s secret value and partial private key.

CL-PKE Features

- No key escrow.
  - User-generated secret component x_A protects against eavesdropping TA.
- No explicit certification of public keys required.
  - Adversary does not know partial private key PPK_A so cannot calculate the full private key.
  - Should assume that TA is not engaged in active adversarial behavior.
- A complete suite of certificateless cryptographic primitives is available:
  - Digital Signatures
  - Key Exchange (KE) and Authenticated-Key Exchange (AKE) protocols
  - Hierarchical schemes
  - Signcryption

CL-PKC Drawbacks

- Is not purely identity-based.
  - Identifier *and* public key needed for encryption.
- Secure channel needed for delivery of partial private keys – as in ID-PKC.
- Revocation is a potential problem
- Does not attain full security of traditional PKI, since TA *might* cheat.
  - But TA must mount an active attack for replacing public keys (in ID-PKC, it could be done by a passive attack).

Al-Riyami & Paterson's Certificateless AKE (2003)

A		B
$a \in_R \mathbb{Z}_q^*;\ T_A = aP$		$b \in_R \mathbb{Z}_q^*;\ T_B = bP$
	$\xrightarrow{T_A, P_A}$	
	$\xleftarrow{T_B, P_B}$	
$e(X_B, P_{\text{KGC}}) \stackrel{?}{=} e(Y_B, P)$		$e(X_A, P_{\text{KGC}}) \stackrel{?}{=} e(Y_A, P)$
$K_A = e(Q_B, Y_B)^a e(S_A, T_B)$		$K_B = e(S_B, T_A) e(Q_A, Y_A)^b$
$h'(K_A \| aT_B)$		$h'(K_B \| bT_A)$.

KGC's master private key: s

KGC's master public key $P_{KGC} = sP$

Public parameters: $(G_1, G_T, e, q, P, P_{KGC}, h, h')$

Alice's secret value: x_A

$Q_A = h(ID_A)$

Alice's partial private key (issued by KGC): $D_A = sQ_A$

Alice's Public key: $(X_A, Y_A) = (x_i P,\ x_i P_{KGC})$

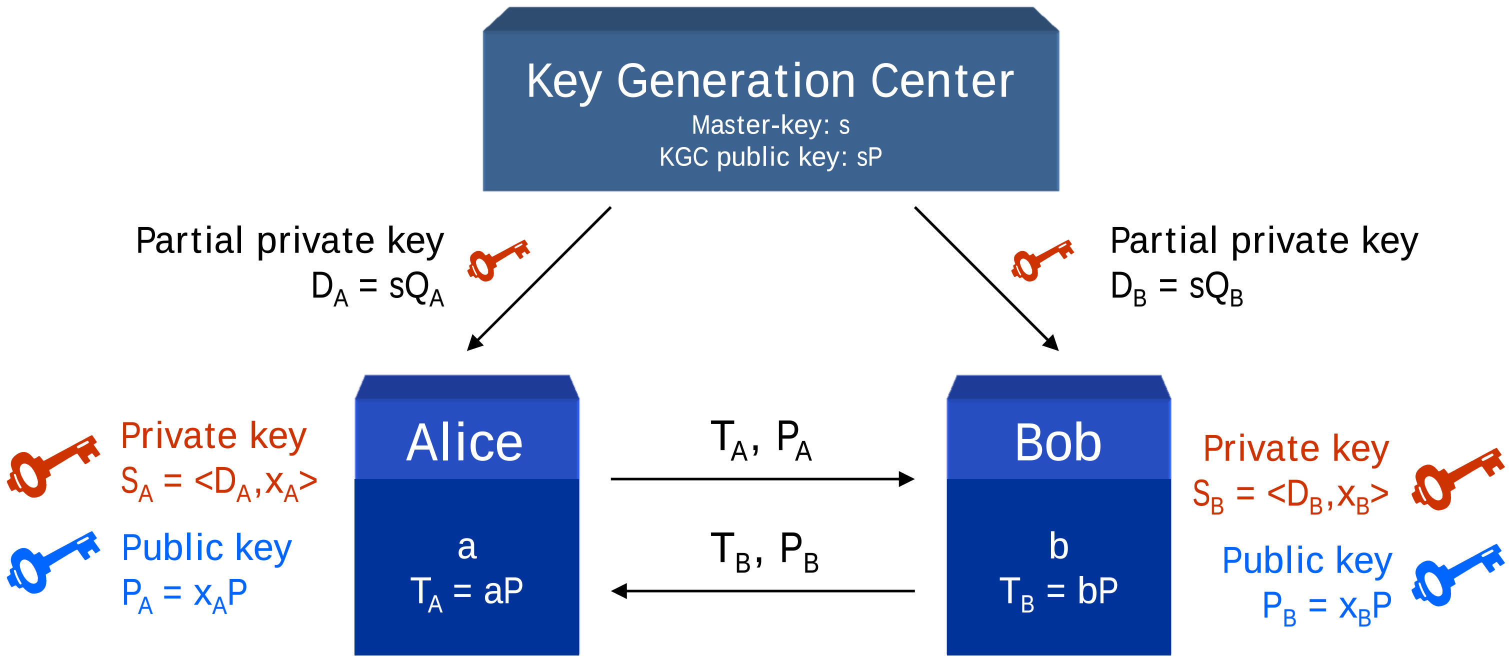
$K_A = e(Q_B, Y_B)^a\, e(S_A, T_B) =$

$e(Q_B, x_B sP)^a\, e(x_A sQ_A, bP) =$

$e(x_B sQ_B, aP)\, e(Q_A, x_A sP)^b =$

$e(S_B, T_A)\, e(Q_A, Y_A)^b = K_B$

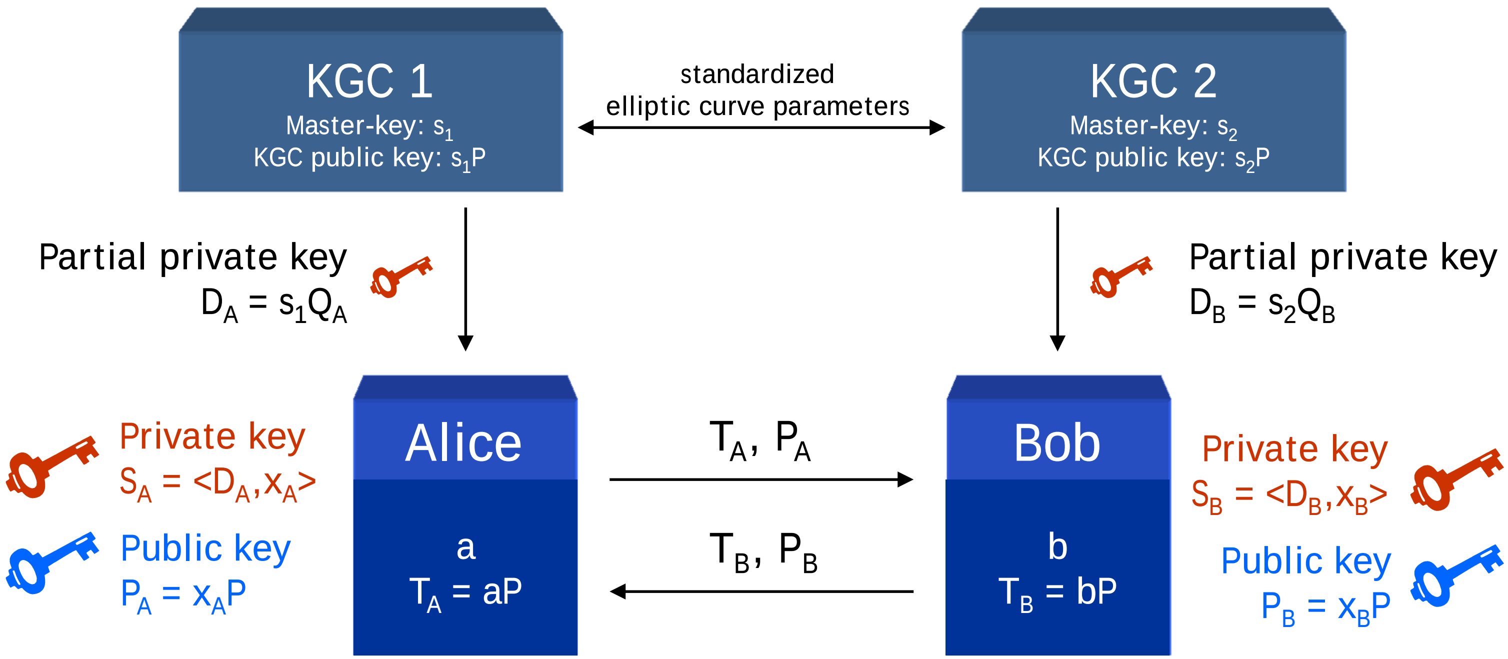
Another Certificateless AKE Protocol



$K_A = \hat{e}(Q_B, P_B + sP)^a \cdot \hat{e}(x_A Q_A + D_A, T_B)$ | $K_B = \hat{e}(Q_A, P_A + sP)^b \cdot \hat{e}(x_B Q_B + D_B, T_A)$

$$K = \hat{e}(Q_B, P)^{a(s+x_B)} \cdot \hat{e}(Q_A, P)^{b(s+x_A)}$$

Another Certificateless AKE Protocol (with multiple KGC)



$K_A = \hat{e}(Q_B, P_B + s_2P)^a \cdot \hat{e}(x_AQ_A + D_A, T_B)$ | $K_B = \hat{e}(Q_A, P_A + s_1P)^b \cdot \hat{e}(x_BQ_B + D_B, T_A)$

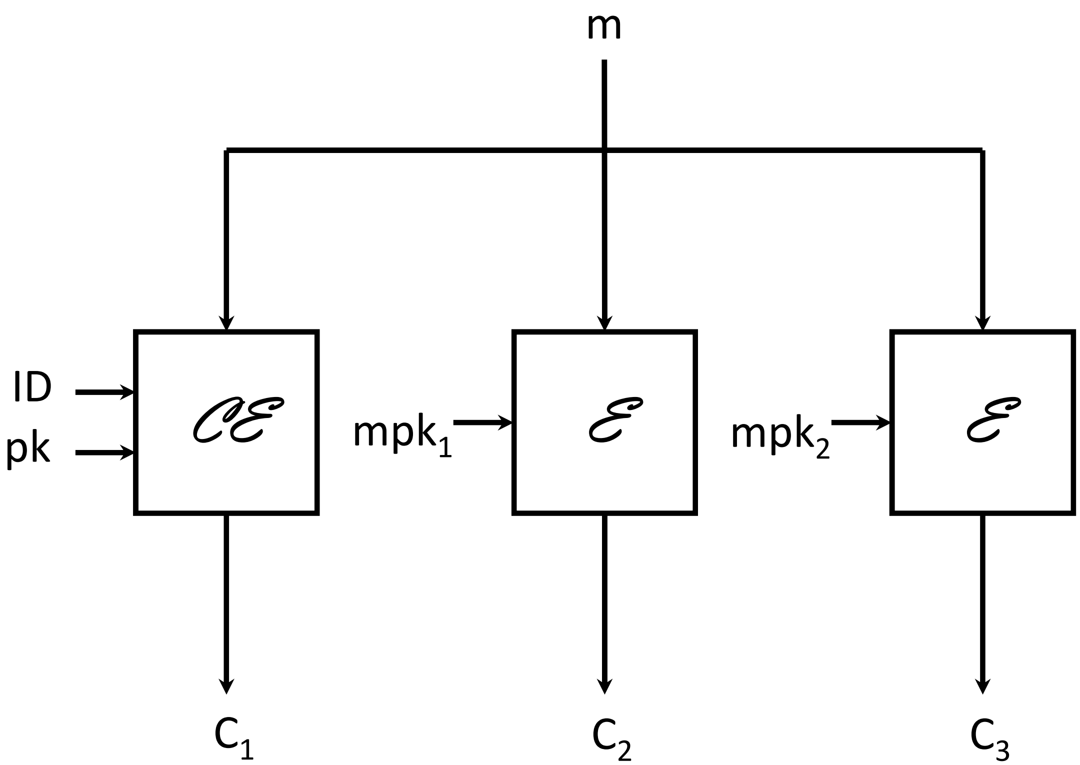
$$K = \hat{e}(Q_B, P)^{a(s_2+x_B)} \cdot \hat{e}(Q_A, P)^{b(s_1+x_A)}$$

A Certificateless AKE Protocol without bilinear pairings (He et. al, 2011)

A		B
Generate a random number a;		Generate a random number b;
$T_A = a \cdot P$;	$M_1 = \{ID_A, T_A\}$ →	$T_B = b \cdot P$;
	← $M_2 = \{ID_B, T_B\}$	
$K_{AB}^1 = (x_A + s_A)T_B + a \cdot (P_B + R_B + H_1(ID_B, R_B, P_B)P_{pub})$;		$K_{BA}^1 = (x_B + s_B)T_A + b \cdot (P_A + R_A + H_1(ID_A, R_A, P_A)P_{pub})$;
$K_{AB}^2 = a \cdot T_B$;		$K_{BA}^2 = b \cdot T_A$;
$sk = H_2(ID_A \parallel ID_B \parallel T_A \parallel T_B \parallel K_{AB}^1 \parallel K_{AB}^2)$		$sk = H_2(ID_A \parallel ID_B \parallel T_A \parallel T_B \parallel K_{BA}^1 \parallel K_{BA}^2)$

Strongly Secure Certificateless Encryption

(Dent et al., PKC'08)



\+ NIZK proof that (C_1,C_2,C_3) are all encryptions of the same message.

- ID and pk are the user's identity and public key.
- mpk_1 and mpk_2 are part of the system parameters
- Decryption process uses the certificateless encryption scheme

One passively secure certificateless encryption scheme: $\mathcal{CE}$

Two instances of a passively secure public-key encryption schemes: $\mathcal{E}$

Questions?