Cryptographic Security Objectives

- Authenticity
  - verifies sends & receivers, prevents impersonation & misrepresentation
- Confidentiality
  - info exchanged is private & confidential
- Integrity
  - info remains intact and not tampered
- Non-repudiation
  - proof of txn taken place & cannot be refuted

Cryptographic Security Implementation

- Authenticity
  - implementation using challenge - response
- Confidentiality
  - implementation using data encryption
- Integrity
  - implementation using message signature
- Non-repudiation
  - implementation using message signature

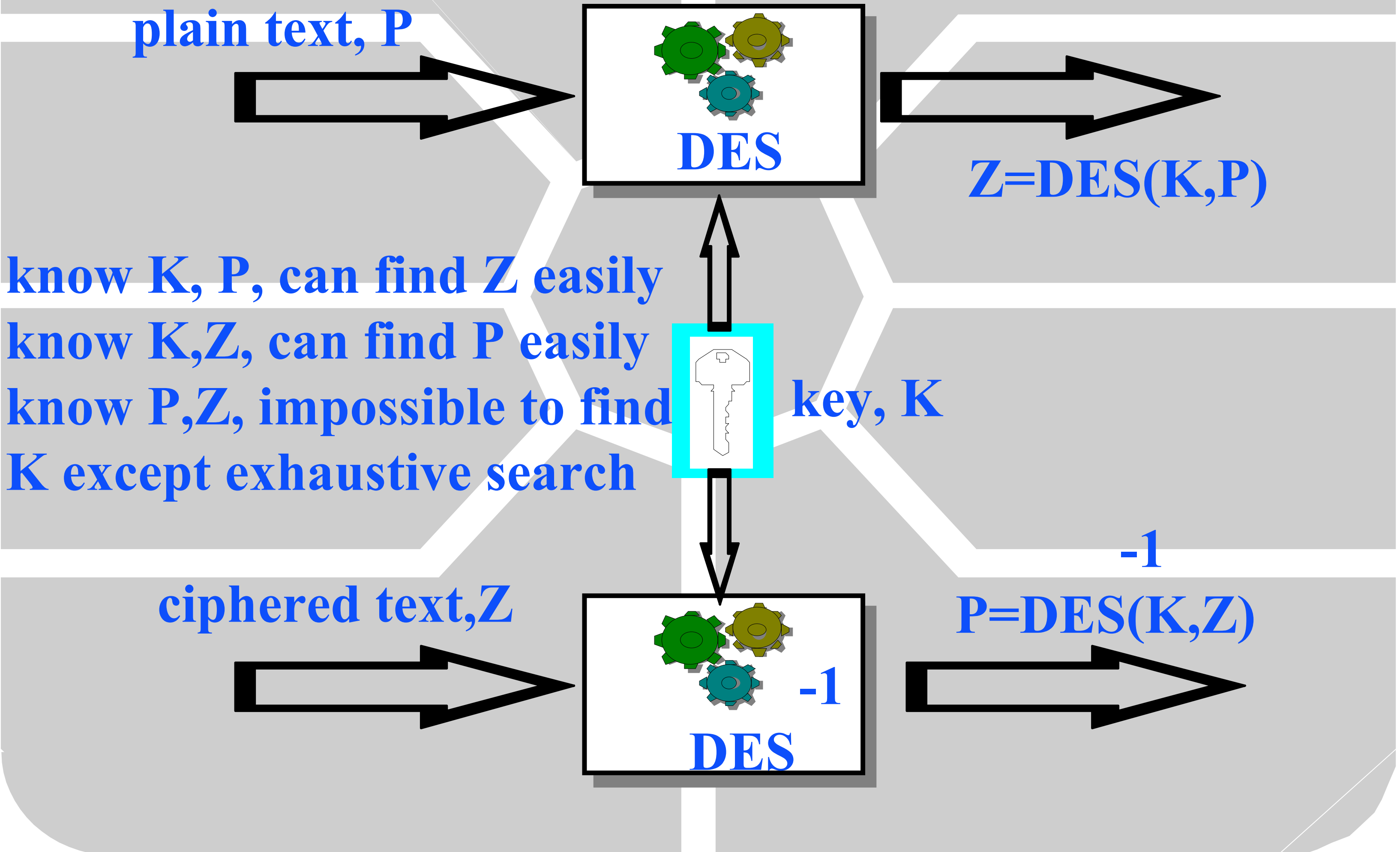
Symmetrical & Asymmetrical Algorithm

- Symmetrical eg DES (or triple DES)
  - good for many-to-one and one-to-one security for example banking
  - simple key management
- Asymmetrical (public key) eg RSA, ECC
  - good for many-to-many security for example electronic mail, electronic commerce
  - complex key management infra-structure
  - public key compliments DES, not replace DES

DES - Data Encryption Standard

- symmetrical key algorithm
- manipulate data in 8 bytes block
- only known attack is exhaustive key search, 2 to the power of 56 computations
- 2 million years for today's PC @1ms per computation or a few hours with special designed hardware, parallel processing
- security can be increased using triple DES

DES

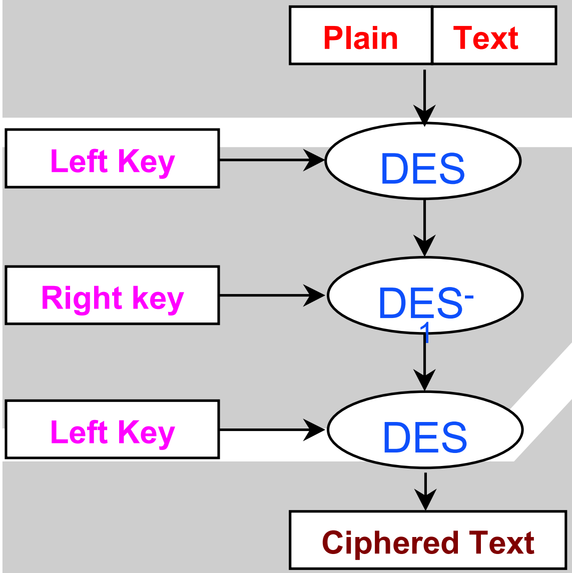

plain text, P
DES
Z=DES(K,P)
know K, P, can find Z easily
know K,Z, can find P easily
know P,Z, impossible to find
K except exhaustive search
key, K
ciphered text,Z
-1
DES
-1
P=DES(K,Z)

DES / Triple DES

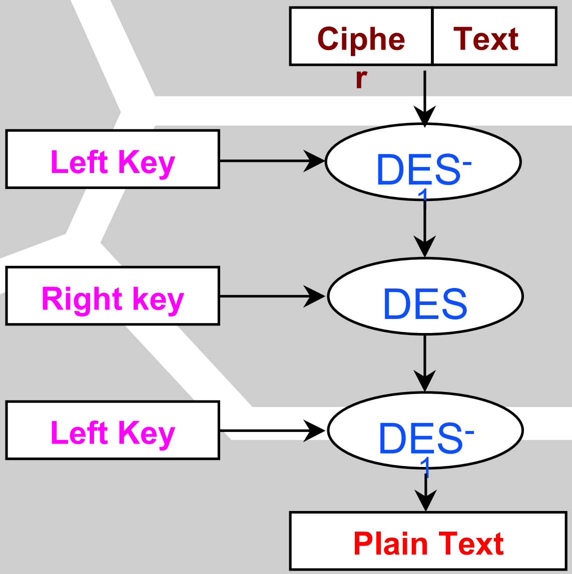
- Single DES uses single length key (8 bytes), K(8)
- 3DES uses double length key (16 bytes), K(16) = $K_L(8)$ | $K_R(8)$ or $K_A(8)$ | $K_B(8)$
- If the left and right part are the same, 3DES reduces to single DES
- Allows smooth migration from single DES to 3DES
- Least significant bit of each byte not used

Triple DES

3-DES Encryption Decryption



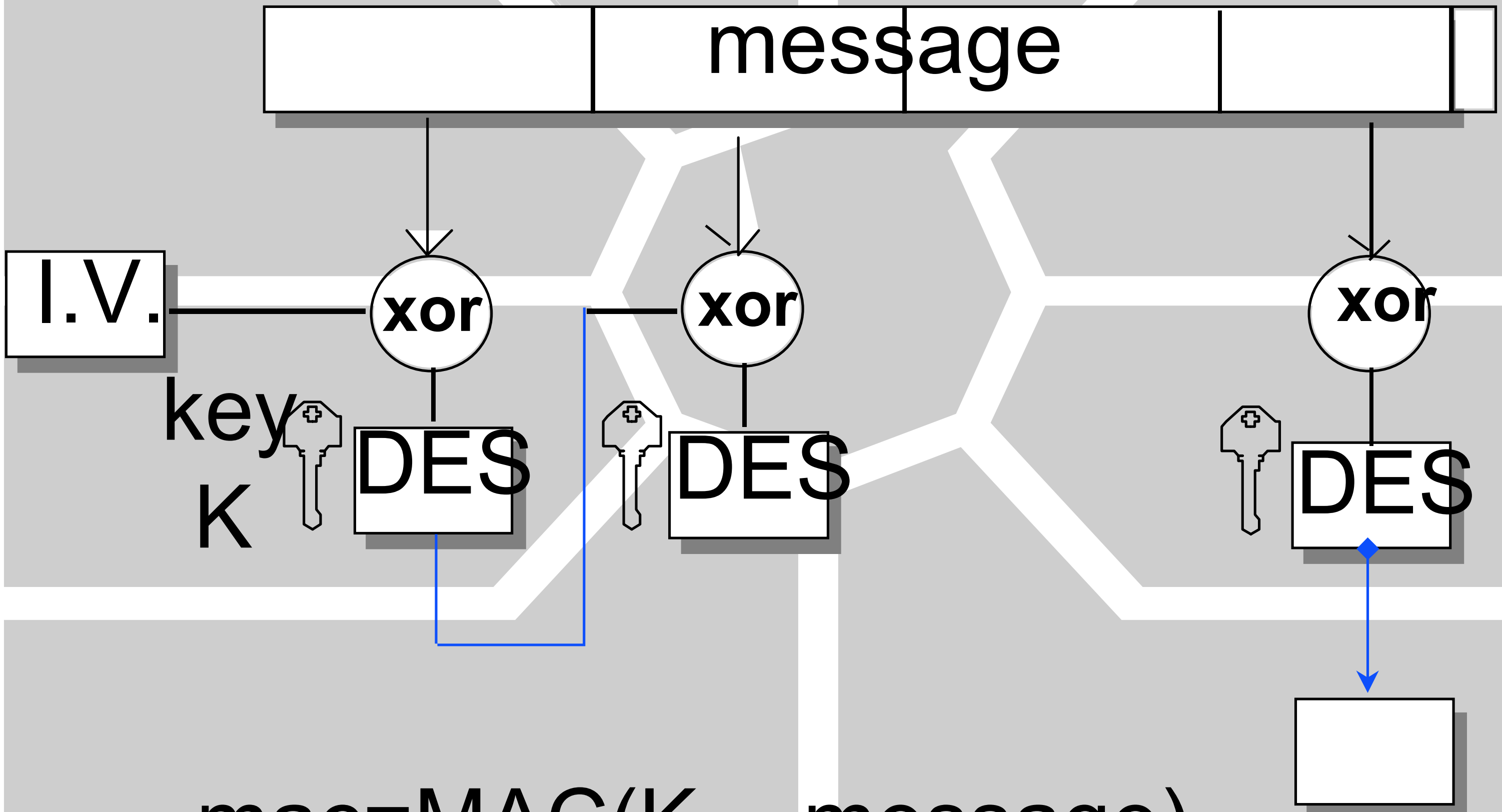
3-DES



$Z=3DES(K,P)$

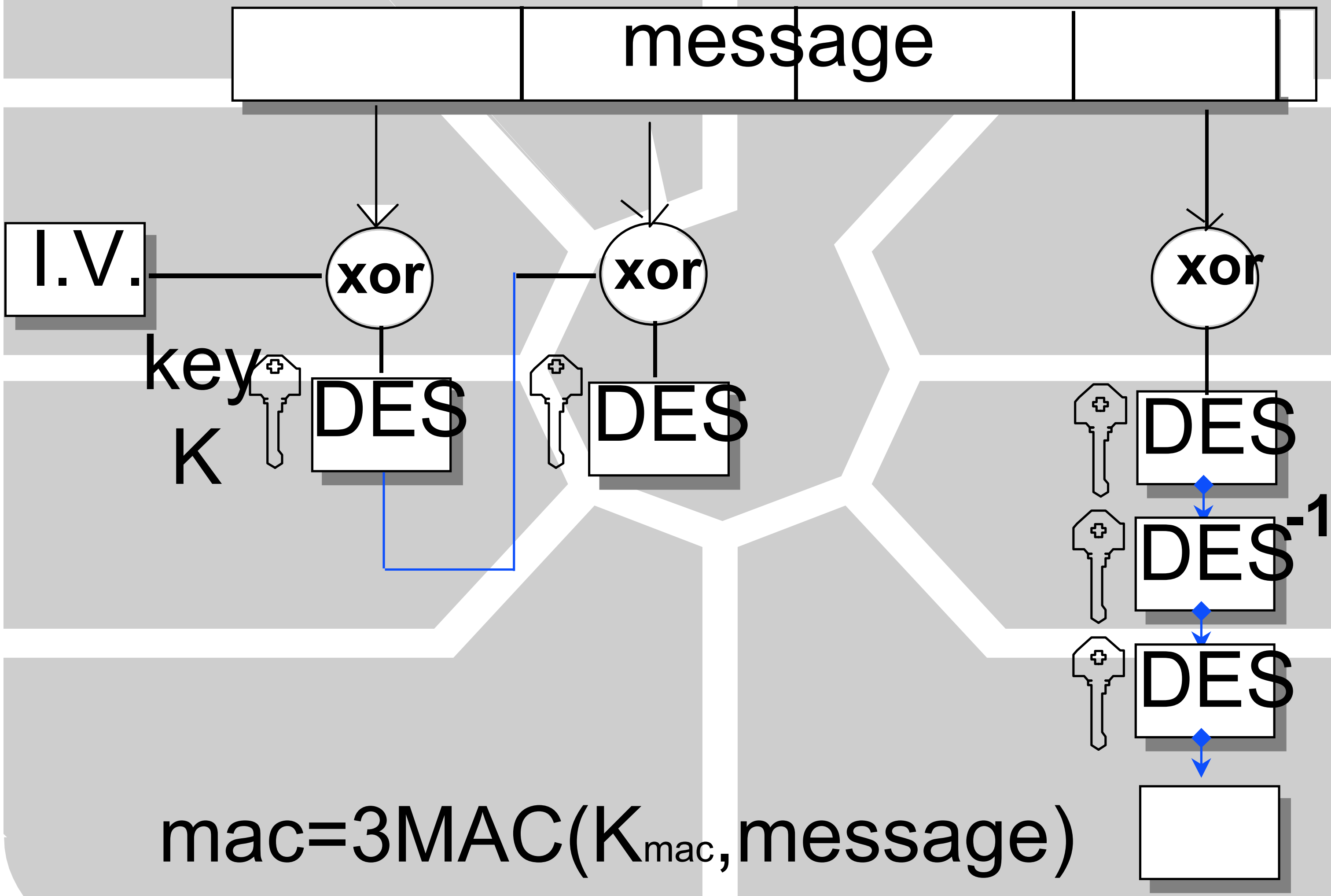
$P=3DES^{-1}(K,Z)$

MAC - Message Authentication Code
Single DES



mac=MAC(K_{mac},message)

MAC - Message Authentication Code
Triple DES



mac=3MAC(K_{mac},message)

MAC

- using a random IV may be a potential loop-hole because (IV + x) xor (block0+x) = IV xor block0
- Use IV = 0 instead
- if message is <= 8 bytes, MAC becomes a DES encryption may be a security loop hole
- padding of 80, 80 00..00 to make the message last block 8 bytes
- if message length is exactly multiple of 8, pad 8000 0000 0000 0000

Hash

- a cryptographic function
- takes a variable length message
- returns a fixed length hash value
- also known as a Message Digest function
- examples MD5(128 bits), SHA(160 bits)
- analogous to a message finger print
- no key is involved
- usage - signature on message's hash is as good as signature on the message

Public Key Algorithm

- each party gets a public key and a private (secret) key which is unique
- public key is published (free read access)
- private key is secret (known only to the party)
- public key is certified by a key certification body - key certificate
- the public key of the certification body is public read access

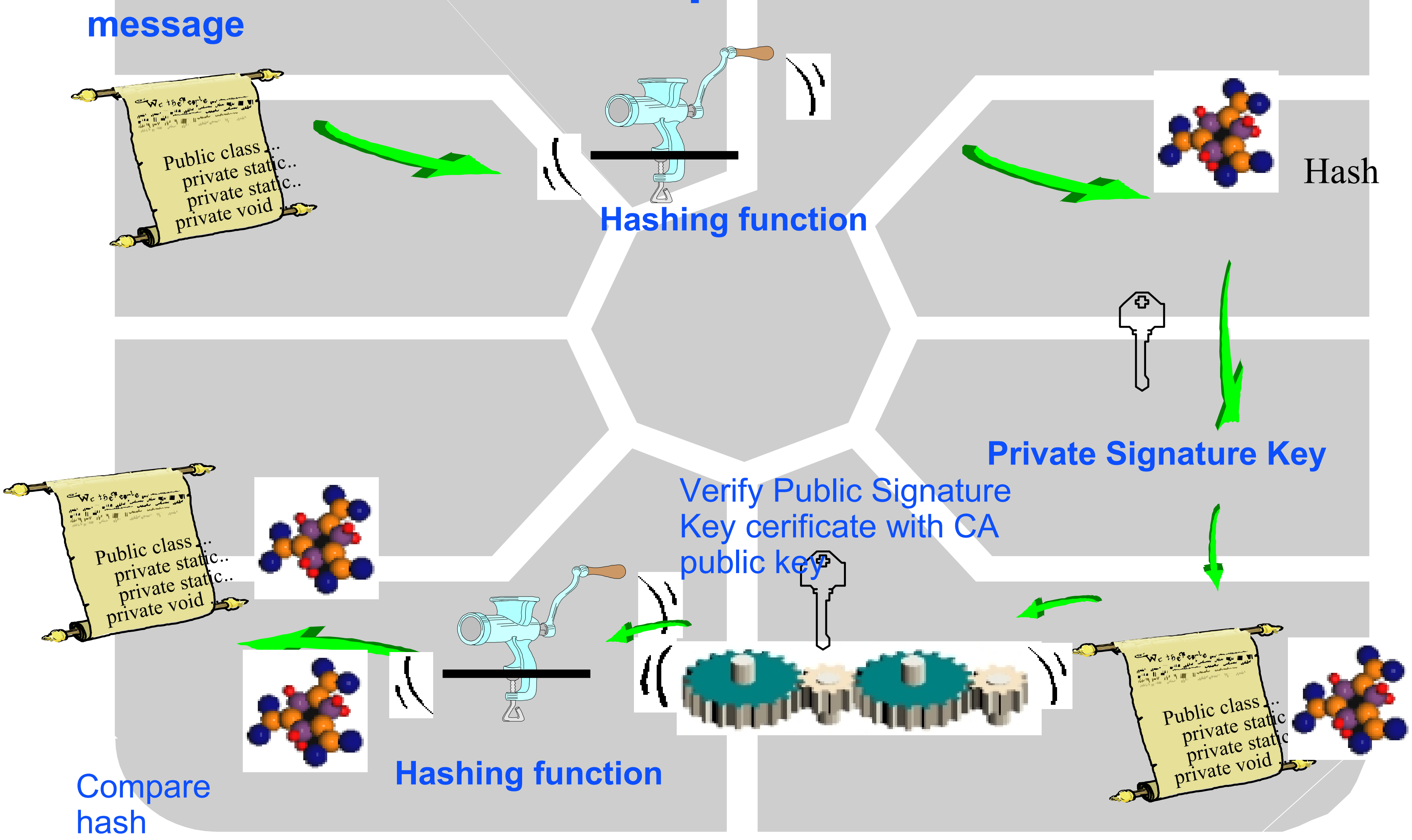
Certification Authority (CA)

- Role is to prove who you claim you are by..
- Associate a unique user to a public key by..
- Signing a public key with CA secret key to..
- Generate a key certificate containing
  - user's public key
  - relevant info about user eg name, ID number ..
  - expiry date of certificate, usage policy
  - (electronic) signature of the CA
- Other functions - certificate distribution & storage, replacement, update, revocation ..

Certificate Revocation List

- Unique certificate that is no longer trusted
  - Key Compromise - secret key lost or compromised
  - Affiliation Changes - wrong name, change company
  - Superseded - updated with a new one
  - Cessation Of Operation - no longer needed for the original purpose

Signature Verification for Integrity Non-repudiation



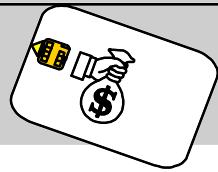
Encryption Using Public Key Algorithm

- Check receiver public key certificate with CA public key
- Check public key revocation list
- Generate random 3DES key
- Encrypt message using 3DES
- Encrypt 3DES Key using other party public key
- Append encrypted 3DES key with encrypted message

Decryption Using Public Key Algorithm

- Decrypt 3DES key using the private key
- Use decrypted 3DES key to decrypt the message

Authenticity - Card Authentication



Terminal:

1. Generate terminal random #, Rt

2. Sends Internal Authenticate command, Int_Auth(algo,@Kc,Rt)

00	88	algo	@Kc	08	Rt

3. Retrieve card cryptogram, GetResp()

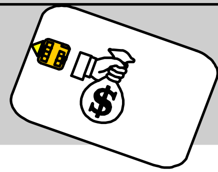
00	C0	00	00	08

Card:

1. Encrypt terminal random# with Kc
   Cc=E(Kc,Rt)
2. Prepare to return card cryptogram

Cc=E(Kc,Rt)

Authenticity - Terminal Authentication



1. Get Challenge command to get card random number, Get_Challenge()

00	84	00	00	08

2. Encrypt Rc with terminal authentication key, Kt to compute terminal response cryptogram Ct=E(Kt,Rc)
3. Issue External Authenticate command, Ext_Auth(algo,@Kt,Ct)

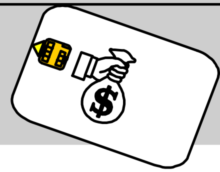
00	82	algo	@K t	08	Ct

1. Generate card random#, Rc

Rc, card random number

2. If Kt not blocked, compute Ct' where **Ct' =E(Kt,Rc)** and compare(Ct,Ct')
3. If OK, grant access right associated with Kt or increment error counter

Authenticity - Cardholder Authentication

	
1. Cardholder enter PIN 2. Terminal send PIN to card using Verify_PIN(PIN) command 00 | 20 | 20 | @PIN | 08 | PIN	1. If PIN not blocked, compare PIN inside the card and PIN from terminal 2. If OK, grant access right associated with the PIN or increment error counter

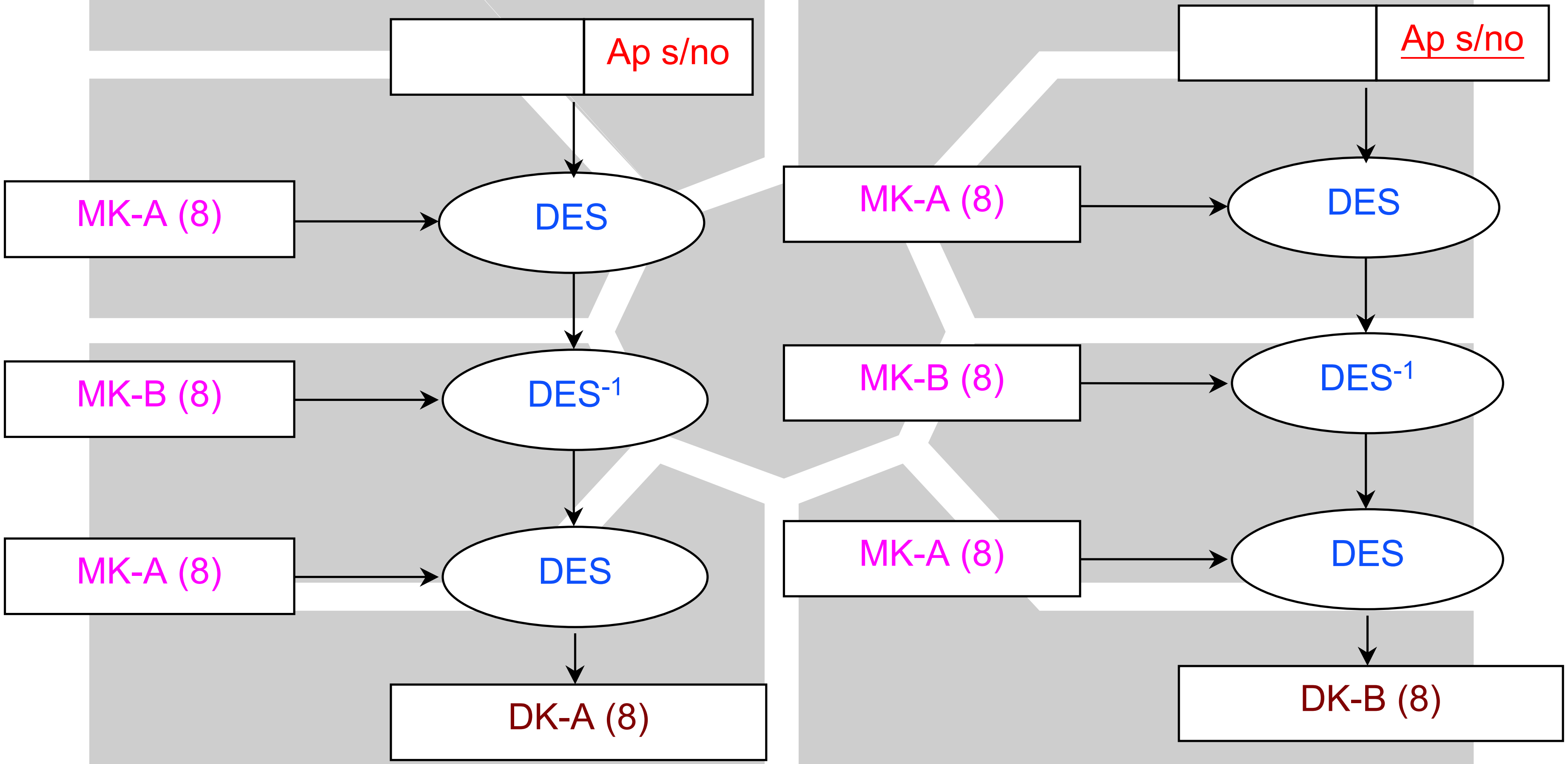
Key Diversification

- a cryptographic technique to ensure that keys in each and every card is unique
- yet allows simple key management
- uses a set a master keys e.g. Card authentication key, terminal authentication key, credit key, debit key ..
- And card unique data e.g. chip serial number, account number to generate card unique secret keys
- used in symmetric key management system

Key Diversification

- Master keys must resides in a security module eg terminal SAM, host HSM
- diversified key in the card
- master keys in devices which can be controlled and smaller quantity i.e. terminal
- diversified keys in devices which is difficult to control (=> difficult to update keys) and bigger quantity i.e. card
- card expires after some times
- back-end audit and blacklist card if necessary

3 DES Key Diversification



Ki = 3DES(Km,s/no) | 3DES(Km,s/no) where s/no is complement s/no

Key Dispersion

For a compromised diversified key, the card can be blacklisted. How about a compromised master key eg debit master key ?

- multiple groups of diversified keys in the card
- single group of master in the SAM
- terminal selects the group in the SAM to be used
- replace all SAMs if a master key is compromised

Session Key

- Valid only during the session and unique
  - function of card / terminal authentication key, card / terminal random number
  - must not be reproduce-able / replayable
- Used to enforced secured messaging
- Resulting in end-to-end security i.e. One end is the card, the other the application SAM
- Prone to loop hole if not correctly implemented

Secured Messaging

- Ensures that ISO-IN command sends to the card has not been tampered and is indeed executed by the card
- Ensures that an ISO-OUT command has not been tampered and is indeed from the card
- Enforced integrity and confidentiality
- Allows end-to-end security implementation

Secured Messaging

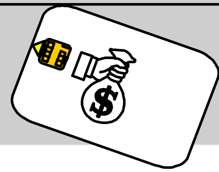
1. Compute mac of ISO-IN command
   mac=3DES(Kmac,ISO-IN-command)

CLA INS P1 P2 Lin+3 Data-in | mac0-3

2. Issue Get Response to retrieve mac7-5

00 | C0 | 00 | 00 | 03 |

3. Verify mac7-5



1. Compute mac of ISO-IN command
   mac=3DES(Kmac,ISO-IN-command)

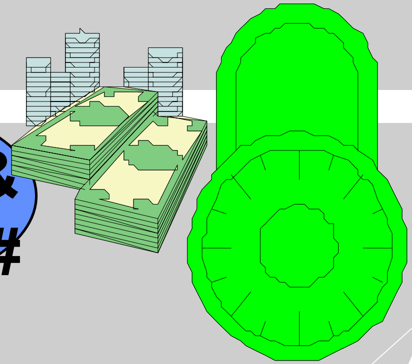
2. Verify mac. If OK execute command.

mac7-5

Transaction Certification

Tcert = MAC(K,transaction record)

Tcert



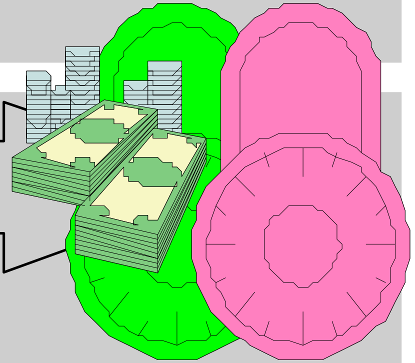
terminal ID

card ID

date & time

terminal & card txn #

Debit Certification & Verification



please debit $ as certified by Tcert

I' ve debited, the proof is DC

Tcert Debit Cert

POS verifies Debit Certificate