Cryptography, its applications

Vašek Matyáš

PA018 –
Advanced Topics in IT Security

Crypto mechanisms

- Workstation vs. LAN/firewall granularity
- Application vs. workstation granularity
- Traffic analysis, privacy services
  - Traffic padding
- Considerations (as usual):
  - Cost
  - Security
  - Administration/Logistics requirements

End-to-end vs. Link encryption

- En-/De-cryption device at sender/recipient ends
- Packet content protected at all nodes
- Headers available to all nodes on the way
- Many services cannot be provided
- IPsec

- En-/De-cryption device at ends of each link
- Processing and message avail. at each node
- Headers can be encrypted on the link (onion routing)
- Advanced network services can be provided

Public-key cryptography

- Shared-key crypto: good security vs. problems with key management
- Authentication of data
  - Hash functions (MAC)
  - Symmetric ciphers (MAC-like)
- GCHQ (UK, 1970) – non-secret encryption
  - Principles of Diffie-Hellman (76), RSA (78)
  - More at *www.gchq.gov.uk*

Shared-key data authentication

- Use the shared key to encrypt the data image
- Only those able to decrypt such message can verify the image correctness

- Use the shared key to create a Message Authentication Code (**MAC**) representing both the data and the key
- Only those able to recalculate the MAC can verify the image correctness

Public-key management

- Yellow Pages-like directory
  - Diffie-Hellman, “phonebooks”
  - Electronic form (browsers)
  - Efforts like Global Trust Register
- Trust models of PGP vs. (?) X.509
  - Web of trust vs. (?) Certification authority
  - PGP modified to accept X.509 certificates
  - Trust model not defined by software, but by the environment (that also implies type of S/W used)

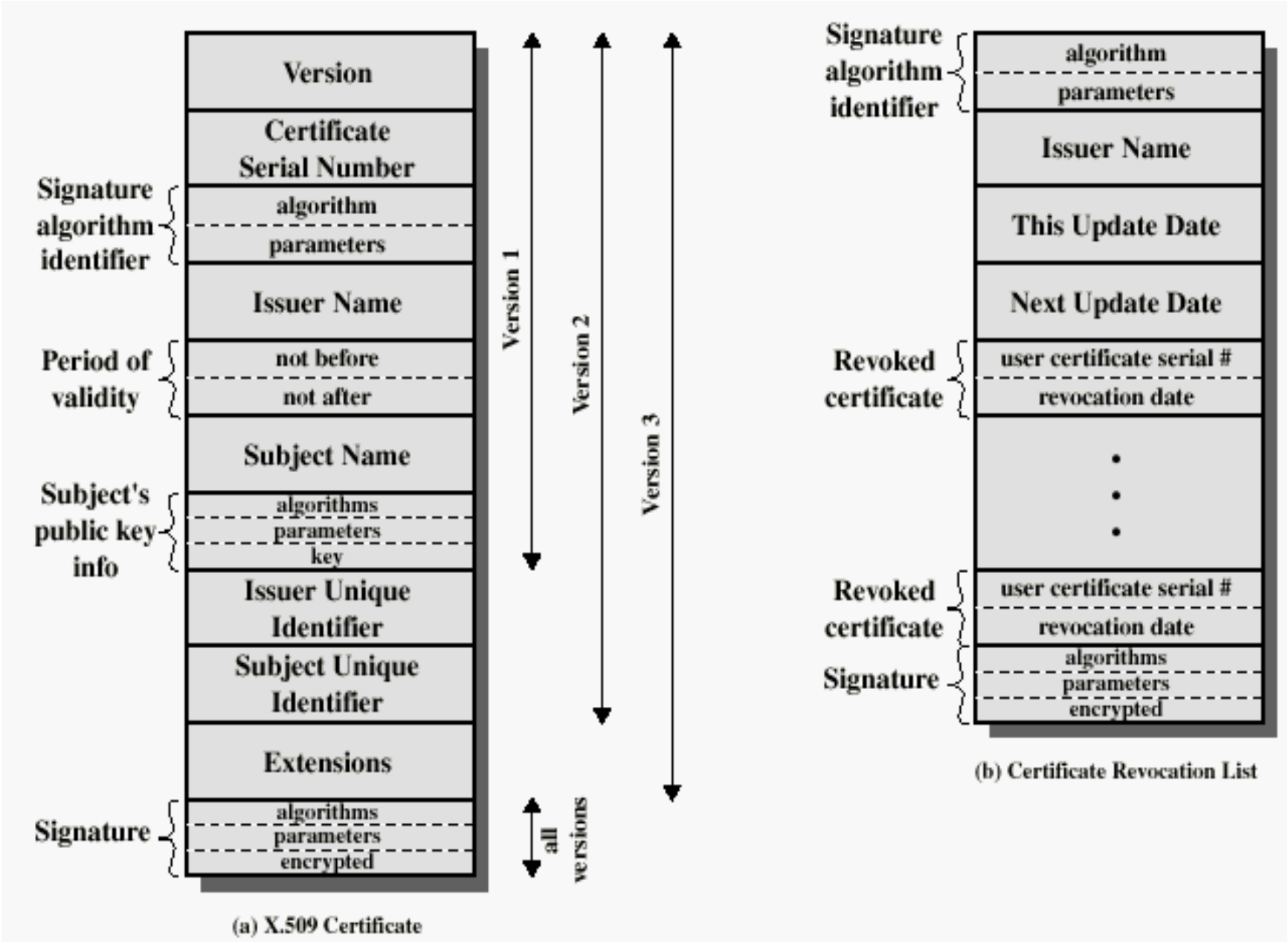
Reliance on the CA

- Anyone (with user X's certificate) can verify with X's CA that X's certificate is valid
  - That this CA created it (possibly off-line using CA's own public key)
  - That the CA still considers it valid (both off-line and on-line)
- No-one (except for the CA = owner of the CA's private key) can create/modify X's certificate

X.509 based authentication

- X.509 specifies the format for public-key certificates.
- The certificate contains the public key of a user and is signed with the private key of a Certification Authority (CA).
- Distributed environment using a database with certificate (user) information.
- Used in S/MIME, IP Security, SSL/TLS, SET.

X.509 certificate



Key/Certificate control

- **Liberal**: key/certificate is **valid** unless we are not explicitly and reliably told otherwise.
  - CRL – Certificate Revocation List.

- **Conservative**: key/certificate **invalid** unless we are explicitly and reliably told otherwise.
  - fresh confirmation, from a trusted party, and useful in case of dispute.
  - OCSP – Online Certificate Status Protocol

- **Revocation is the matter of highest importance!!!**

Certificate revocation

- Certificate revocation != key revocation
- User-lead (PGP) or CA-lead (X.509) revocation
- Reasons for certificate revocation
  - The user is no longer certified (represented) by a given CA
  - CA’s certificate or even private key misused
  - User’s private key misused

Revocation – Technical note

- PGP users can revoke their key without certifier's knowledge
- X.509 CAs can revoke user's key without her knowledge

PGP lessons

- Obviously, key servers unreliable <president@whitehouse.gov>
- Key IDs unreliable
  - should not be used for binding
- Key fingerprints better (yet not unique!!!)

PKI in use today

1) Internal systems (authentication in distributed environments)

2) With existing customers (online banking)

3) Communication with other players (partners, etc.) that have been previously known

Authenticity of documents

- Current approaches to digital signatures unsuitable to publishing, unclear liability issues, etc.

- Possible solutions:
  - Signing keys with shorter life than verification key(s)
  - Hash trees

Symmetric block ciphers (more in PV079)

- Figures and some slides used from:

  http://williamstallings.com/Crypto/Crypto4e.html

  – Some slides provided by Henric Johnson (Blekinge Inst. of Techn., Sweden) and Lawrie Brown (Australian Defence Force Academy)

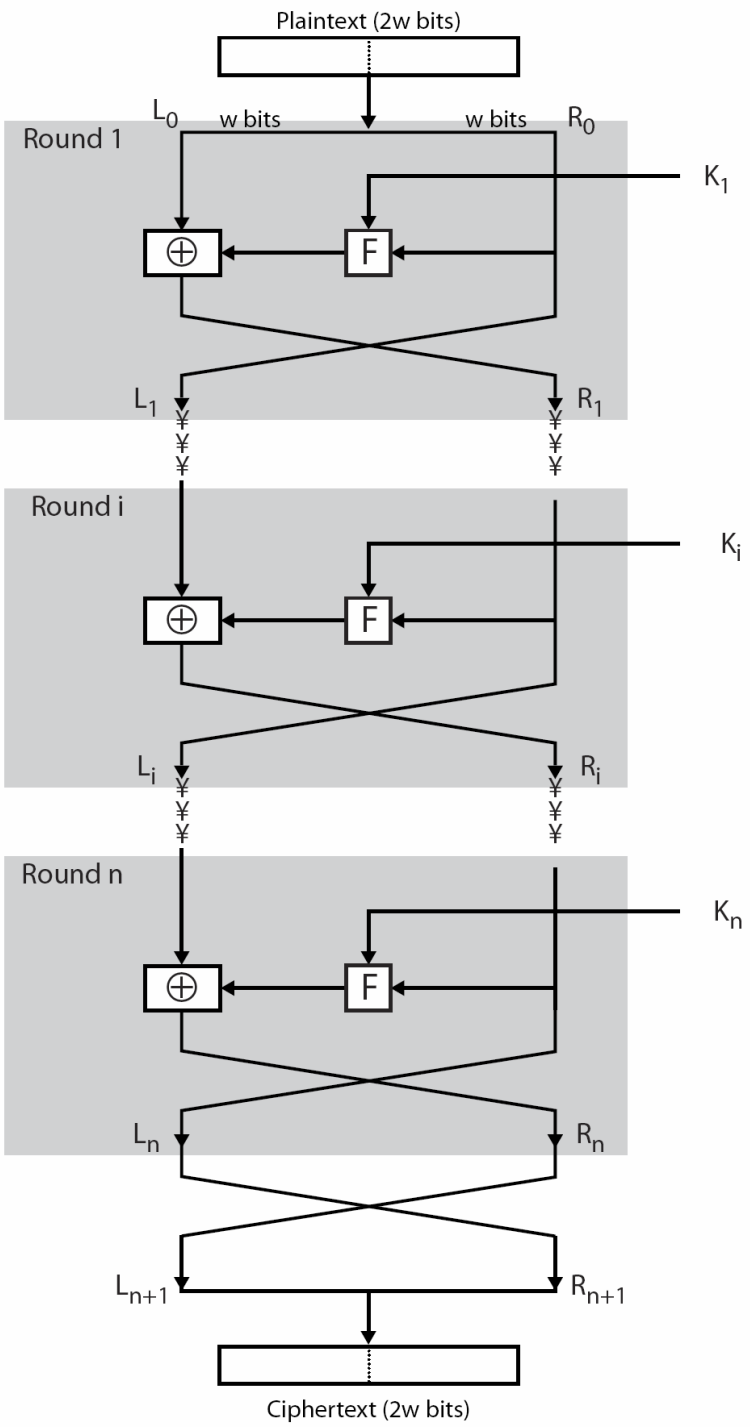
- AES standard, etc.

  http://csrc.nist.gov/CryptoToolkit/aes/rijndael/misc/nissc2.pdf

Feistel ciphers

- Block manipulation, with the block
  - Not too small – cipher would not be complicated
  - Not too big – permutations would be complicated
- Substitution performed on left half of data
  - Round function applied on the right half
  - XORing with the left half
- Permutation – exchange of the two halves

- Parameters: key size, block size, number of rounds


Plaintext (2w bits)
L_0
w bits
w bits
R_0
Round 1
K_1
F
L_1
R_1
Round i
K_i
F
L_i
R_i
Round n
K_n
F
L_n
R_n
L_{n+1}
R_{n+1}
Ciphertext (2w bits)

DES – Data Encryption Standard

- Still the most widely used encryption scheme
- Data Encryption Algorithm (DEA)
- IBM cipher LUCIPHER, modified(!)
  - LUCIPHER – H. Feistel, project for Lloyd's Bank (UK)
  - 128bit key-length reduced to 56 bits
  - Design of S-boxes classified
- US standard in 1977, last renewal in 1994
  - NBS/NIST – FIPS PUB 46
- 64 bit blocks of input/output
- 56 bit key (64 with parity bits)
- Weak keys (4): $E_k(x) = x$
- Semi-weak keys (6 pairs): $E_{k_2}(E_{k_1}(x)) = x$

Breaking DES

- 1977 Diffie & Hellman – design ($20M)
- 1993 M. Wiener – chip design
  - $10M – 21 minutes
  - $1M – 3.5 hours
  - $100k – 35 hours
- 1997 DES-breaking, 70’000 systems, 96 days
- 1998 EFF DES-breaking machine built
  - Special circuits, PC-master
  - $200’000
  - Breaking keys in single hours

DES-based ciphers

- Double DES: $E_{k_2}(E_{k_1}(x))$
- Triple DES (3-DES-3):
  - Diffie-Hellman: $E_{k_3}(E_{k_2}(E_{k_1}(x)))$
  - Merkle: $E_{k_3}(D_{k_2}(E_{k_1}(x)))$
- **Triple DES (3-DES-2): $E_{k_1}(D_{k_2}(E_{k_1}(x)))$**

Advanced Encryption Standard exercise

- Rumors from NIST in 1996
- January 1997 – Official announcement
- September 1997 – Call for Proposals
- August 1998 – 15 candidates announced
- August 1999 – 5 finalists
- 2 October 2000 – Choice of algorithm
- Late 2000, early 2001 – First implementations (PGP 7.0.3)
- Spring 2001 – Standard – FIPS

AES evaluation criteria

- Initial criteria:
  - security – effort for practical cryptanalysis
  - cost – in terms of computational efficiency
  - algorithm & implementation characteristics
- Final criteria
  - general security
  - ease of software & hardware implementation
  - implementation attacks
  - flexibility (in en/decrypt, keying, other factors)

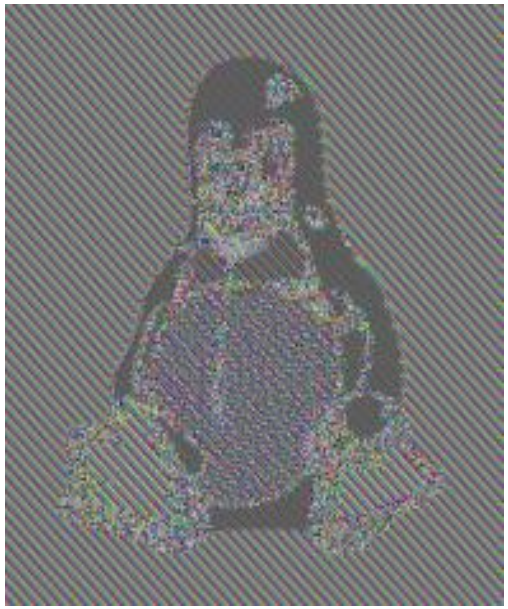
AES finalists

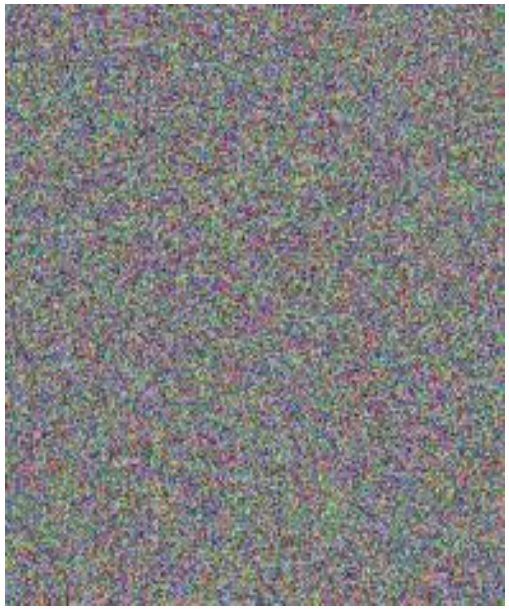
- MARS (IBM)
  - high security, large ROM req., no good HW impl.
- RC6 (RSA Labs)
  - adequate security, moderate ROM req., average HW impl.
- Rijndael (Rijmnen, Daemen – Belgium!)
  - adequate security, fast-SW, low memory req., fast-HW
- Serpent (Anderson, Biham, Knudsen)
  - high security, low memory req., slow-SW, fast-HW
- Twofish (Schneier et al.)
  - adequate security, high ROM req., average HW impl.

AES-Rijndael

- Input & Output: 128 bits
- Key: 128, 192 or 256 bits
- Processing by bytes – basic units
- Operations – addition (XOR), multiplication
- 10, 12 or 14 rounds (given by key length)
  - Initial Round Key addition
  - Last Round slightly different

(DES&AES) Modes of operation – Block modes

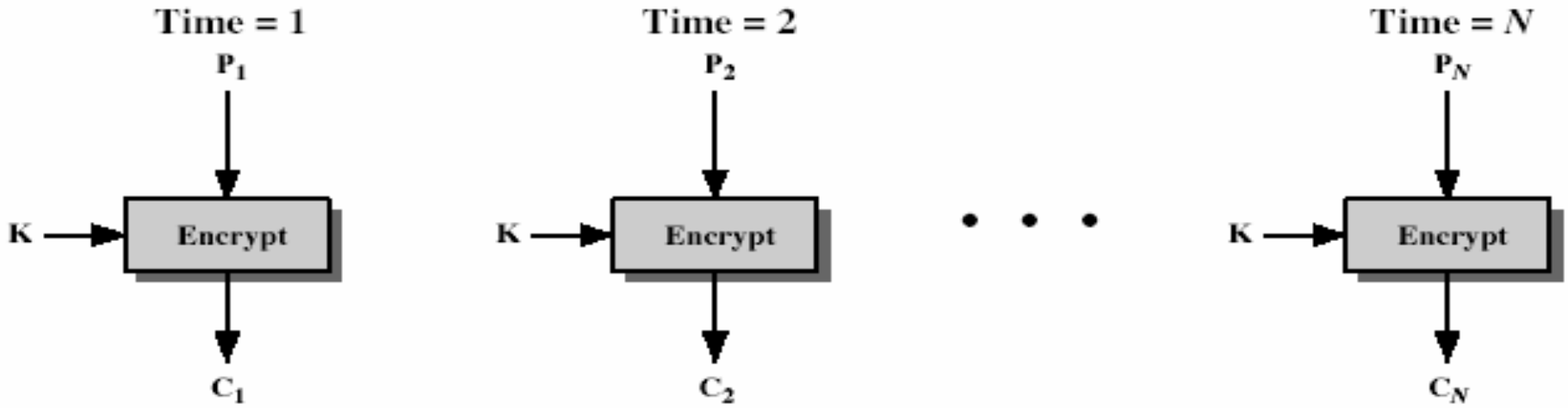




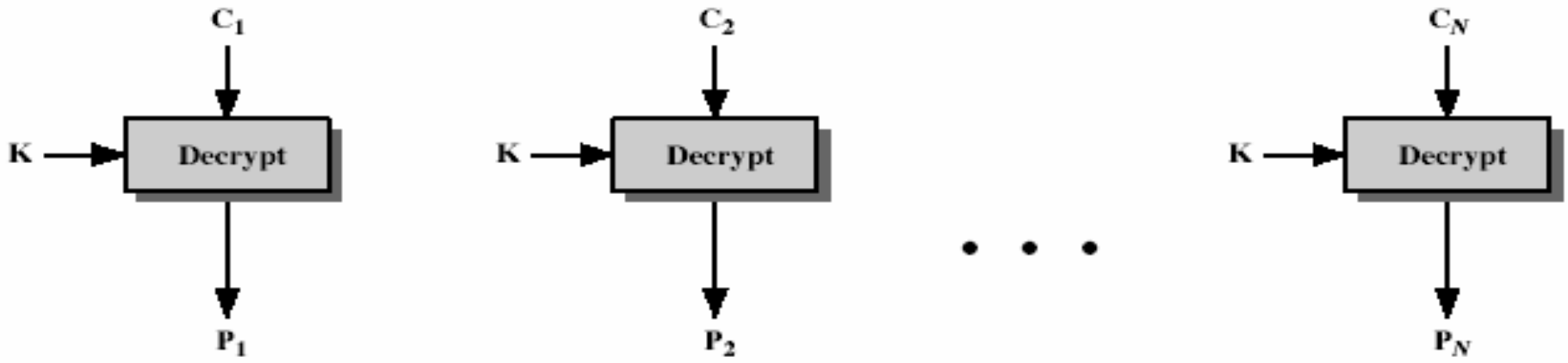
- ECB-encrypted image has observable patterns
- CTR/CBC encryption looks like random noise

Credit for pics – Eric Swankoski & Vijay Narayanan

ECB



(a) Encryption

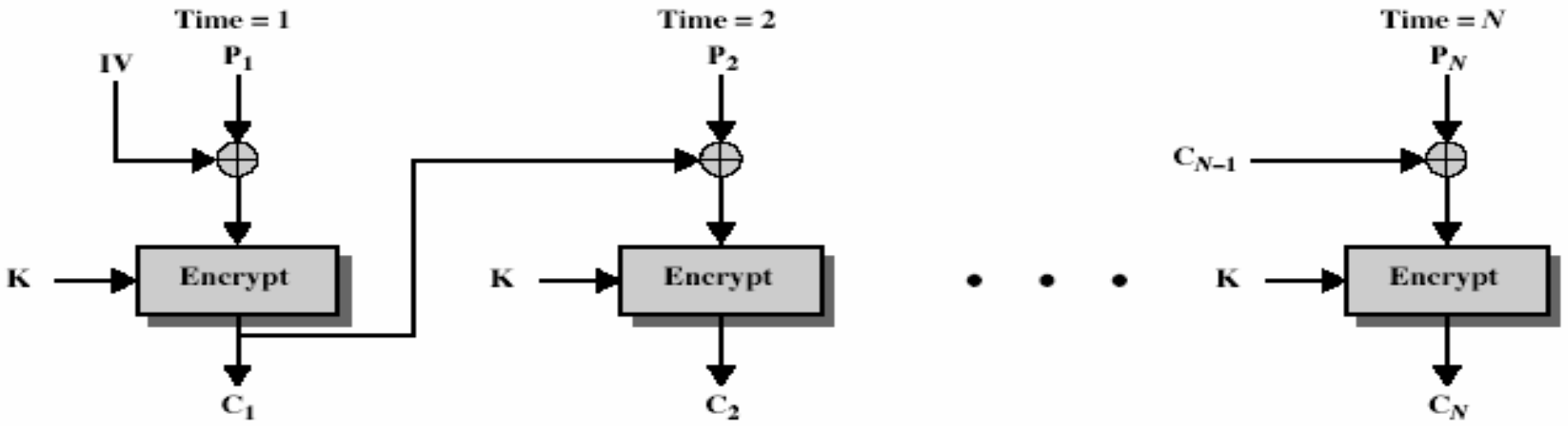


(b) Decryption

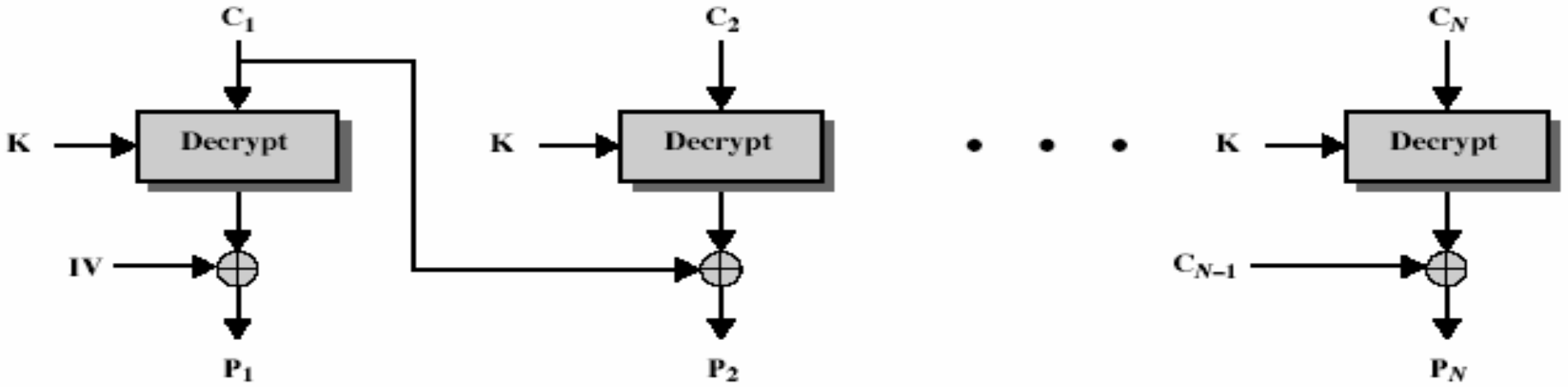
ECB issues

- Repetitions in message can be reflected in ciphertext!!!
  - E.g., with messages that change very little, which become a code-book analysis problem
    - High-redundancy formats – e.g., video, audio
- Reason – enciphered message blocks are independent of each other.
- Main use – sending a few blocks of data

CBC



(a) Encryption

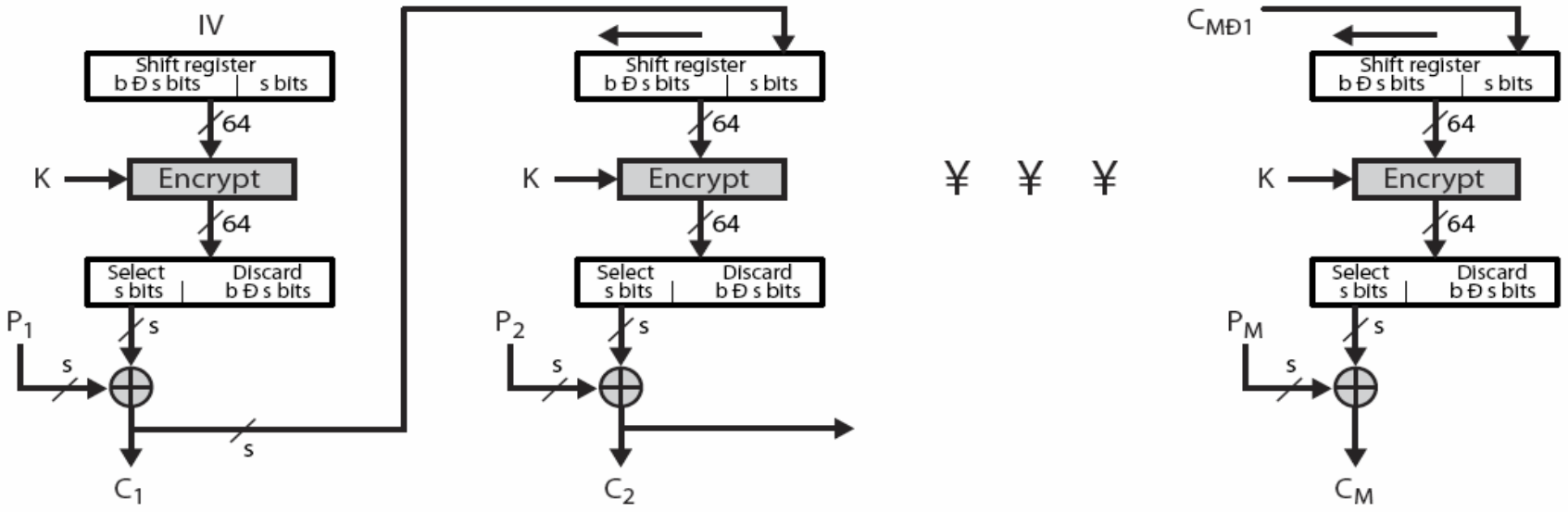


(b) Decryption

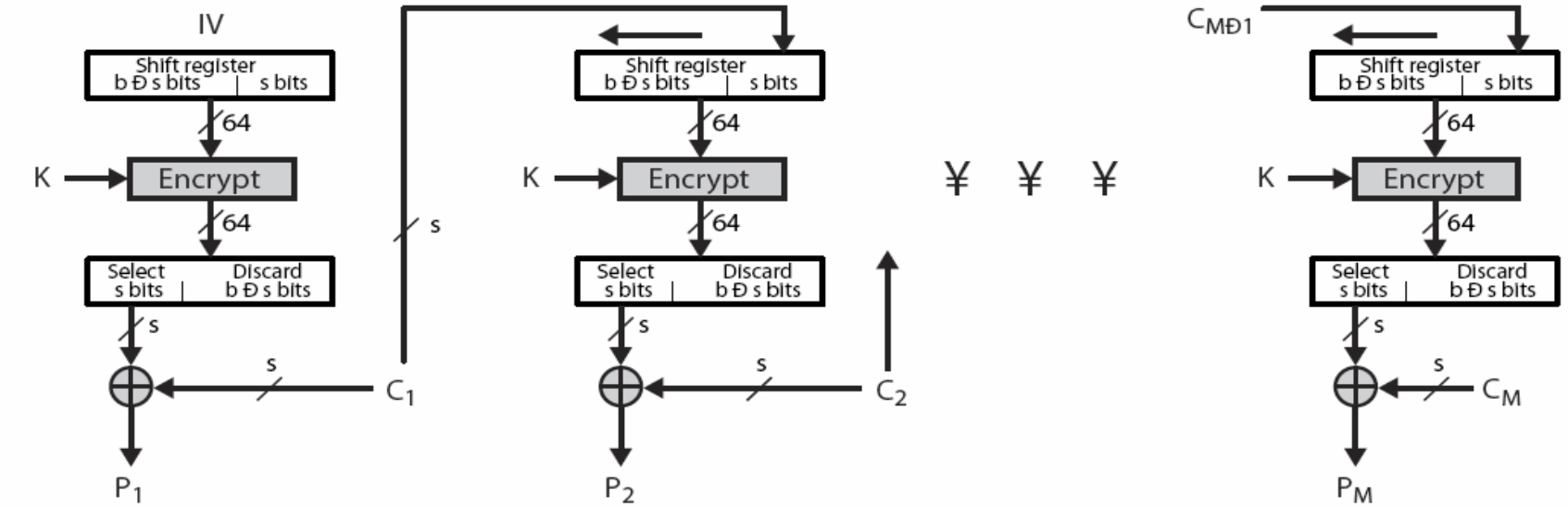
CBC issues

- Each ciphertext block is dependent on *all* message blocks before it
  - I.e., a change in the message affects the ciphertext block after the change as well as the original block.
  - Often marked out as the most secure mode
- *Initial Value* (IV) must be known by both sender and receiver!
  - IV cannot be sent in clear – must either be a fixed value or be sent encrypted in ECB mode before rest of message
- Caution – end of the message, have to handle a possible last “short” block – *padding*.

CFB



(a) Encryption

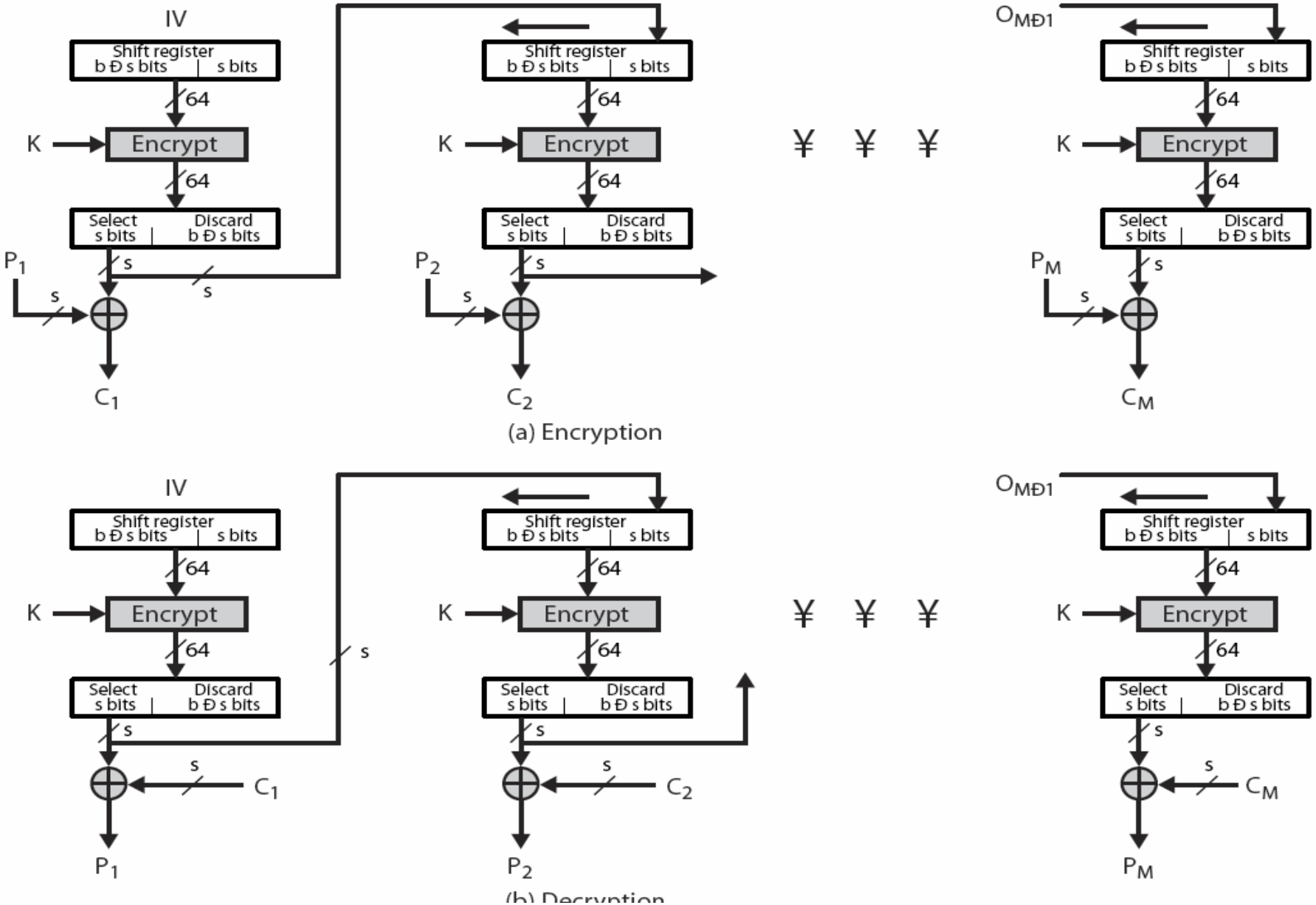


(b) Decryption

CFB issues

- Use when data is bit or byte oriented – a stream mode
  - Actually the most common stream mode
- The block cipher is use in *encryption mode at both ends*, with input being a feed-back copy of the ciphertext
- Can vary the number of bits fed back, trading off efficiency for ease of use.
- Errors also propagate for several blocks after the error (given by the size of feedback register and shift value).

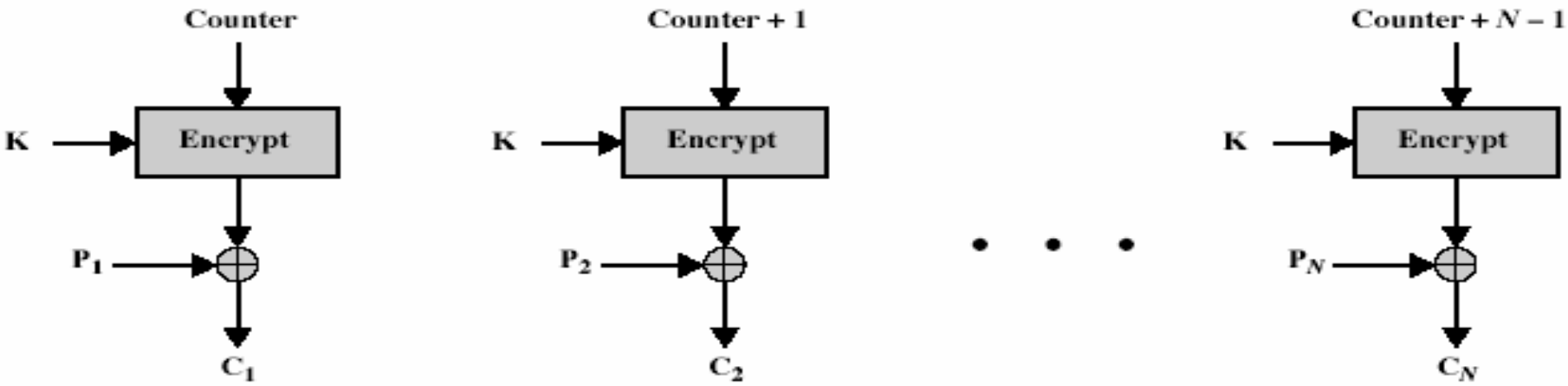
OFB


IV
Shift register
b Ð s bits
s bits
64
K
Encrypt
64
Select
s bits
Discard
b Ð s bits
s
P1
C1
P2
C2
OMÐ1
PM
CM
(a) Encryption
C1
C2
CM
P1
P2
PM

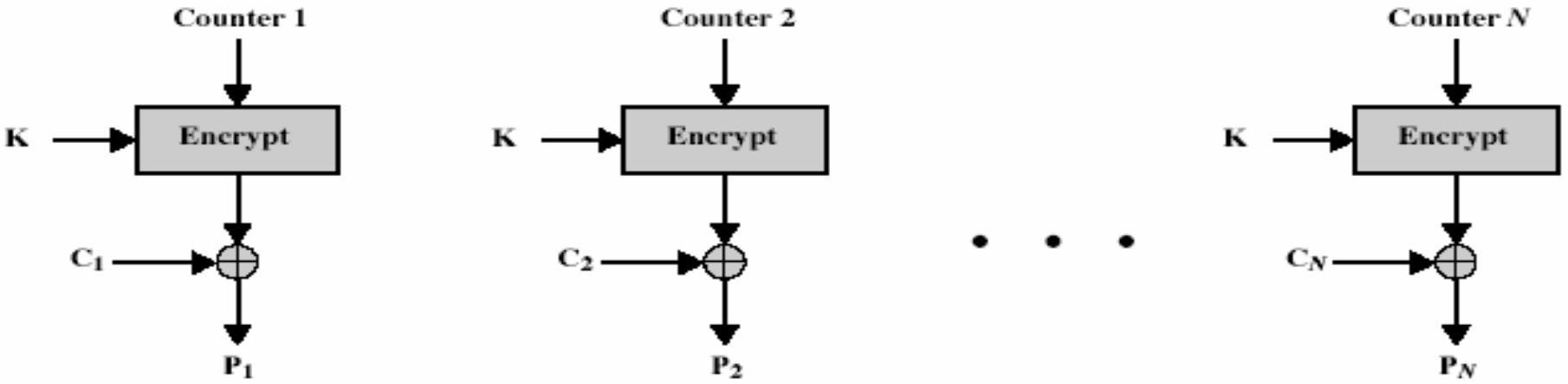
OFB issues

- Intended for use where the error feedback is a problem, or where the encryptions (expensive operations) should be done before the message is available.
- Difference from CFB: the feedback is from the output of the block cipher and is *independent of the message*, a variation of a Vernam cipher.
  - hence must **never reuse** the same sequence (key+IV)
- Again, an IV is needed; and sender and receiver must remain in synchronization, and some recovery method is needed to ensure this occurs!!!
- Originally specified with m-bit feedback
  - subsequent research has shown that only **full block feedback** (ie CFB-64 or CFB-128) should ever be used

CTR



(a) Encryption



(b) Decryption

Advantages and limitations of CTR

- Efficiency (much better than CBC)
  - can do parallel encryptions in h/w or s/w
  - can preprocess in advance
  - good for bursty high speed links
- Random access to encrypted data blocks
- Provable security (good as other modes)
- No error propagation – errors are completely isolated
- Must avoid key/counter values reuse, otherwise could break (cf OFB)

Classical fielded applications

- Symmetric crypto
- Keys at different levels (of security, time of use, etc.). Example (simplified IBM model):
  - Master key – protects terminal keys, in a highly tamper-resistant module
  - Terminal key – protects session keys, stored in a secure (tamper-evident/resistant) memory
  - Session key – protects data in transmission

Use of session (short-term) keys

- To limit volume of ciphertext (under one key) for cryptanalytic attack
- To limit the window of exposure (time and data volume) in the event of key compromise
- To avoid storing large number of distinct keys by creating keys only when actually needed
- To create independence across sessions and/or applications

Security and crypto – reduction of the cornerstone problem

- Knowledge of a secret (key) $\Rightarrow$ identity
- For shared-key crypto based on trust in the party the key is shared with
  - Ability to en-/de-crypt or MAC
- For public-key crypto based on trust in the association between the public key and other data
  - Ability to sign or decrypt messages

Course reading – week 2

- Using encryption for authentication in large networks of computers (Needham & Schroeder, 1978 in Comm. ACM)
  - Part of the first assignment

Reminder – term project report

- Approvals after March 3 with 25% penalty
  - And 50% penalty if not approved by March 17th
- Your report should be:
  - Focused on the topic, analytical in nature (your own view/comments, at least in conclusions, is critical!)
  - 9-10 pages, sharp! Single lines, equiv. Times N. R. 11 (10 if necessary)
  - Delivered on/before the deadline – May 23rd