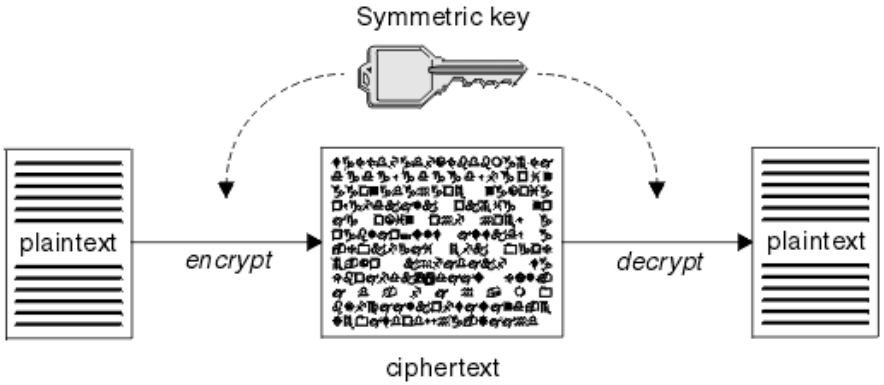


Computer Security

Symmetric Encryption

If you reveal your secrets to the wind, you should not blame the wind for revealing them to the trees.
—Kahlil Gibran

Tamer ABUHMED
Department of Computer Science & Engineering
Sungkyunkwan University

SUNG KYUN KWAN UNIVERSITY

Random Numbers

- Many uses of *random numbers* in cryptography
  - nonces in authentication protocols to prevent replay
  - session keys
  - public key generation
  - keystream for stream ciphers

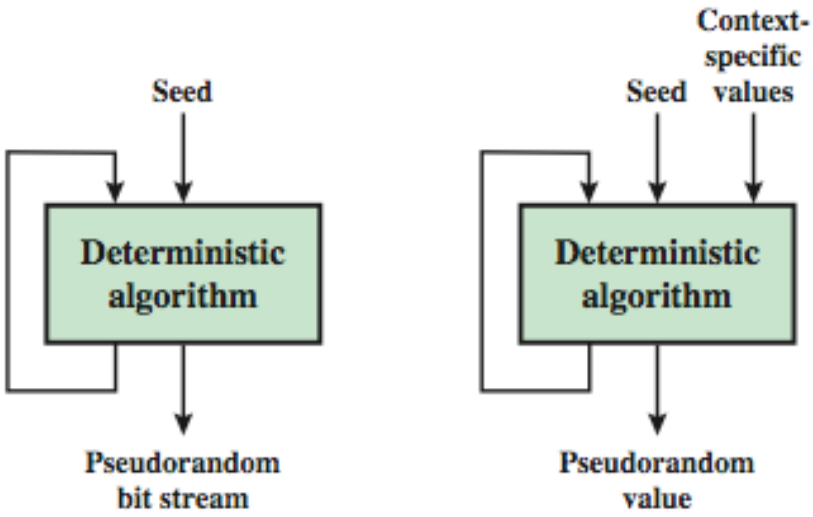
```
89206032161353150760
42991627100678658139
21768097580266432813
57585533115160214822
56368006290497803786
```

- Characteristics of random numbers
  - Statistical randomness
    - Uniform distribution of zeros and ones
    - İndependence of the bits in the sequence
  - Unpredictability of future values from previous values
- True random numbers provide these but very hard to obtain and use in practice

Pseudorandom Number Generators (PRNGs)

- often use deterministic algorithmic techniques to create "random numbers"
  - although are not truly random
  - can pass many tests of "randomness"
- known as "pseudorandom numbers"
- created by "Pseudorandom Number Generators (PRNGs)"

Pseudorandom Number Generators & Psuedorandom Functions



(b) PRNG (c) PRF

- Not much different
  - PRNG output is open-ended while PRF generates fixed size output
  - PRNG is mostly context independent while PRF is context dependent
  - Both may use feedback (there are some non-feedback ones too)
- When used in a cryptographic operation, seed must be kept secret

PRNG/PRF Requirements

- Randomness
  - Uniformity: the occurrence of zeros and ones must be equally likely
  - Scalability: any subseqeunce must pass randomness tests as well
  - Consistency: must not dependent on a particular seed value
- Unpredictability
  - forward unpredictability (next bits cannot be learned using previous bits)
  - backward unpredictability (seed cannot be learned using PRN sequence)
- There are some standard tests (total 15 of them) to check randomness and unpredictability (NIST SP800-22)
- Characteristics of the seed
  - secure
  - if known adversary can determine output
  - so must be random or pseudorandom number (there are some other standard tests for seed randomness as well)

Linear Congruential Generator

- Common iterative technique using:

  $X_{n+1} = (aX_n + c) \bmod m$

  X_0 is the seed
- Given suitable values of parameters can produce a long random-like sequence
- Suitable criteria to have are:
  - function generates a full-period (all values between 0 and m-1)
  - generated sequence should appear random
- Note that an attacker can reconstruct sequence given a small number of values
  - So, not a secure mechanism

Using Block Ciphers as PRNGs

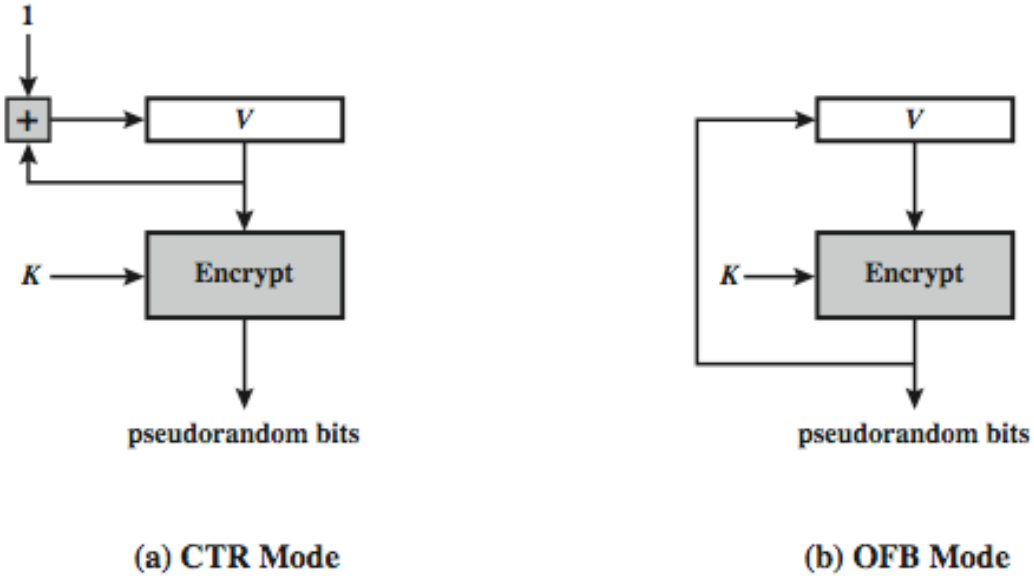
- for cryptographic applications, can use a block cipher to generate random numbers
- often for creating session keys from master key
- Standard methods
- CTR

$$X_i = E_K[V+i]$$

- OFB

$$X_i = E_K[X_{i-1}]$$

$$X_0 = E_K[V]$$

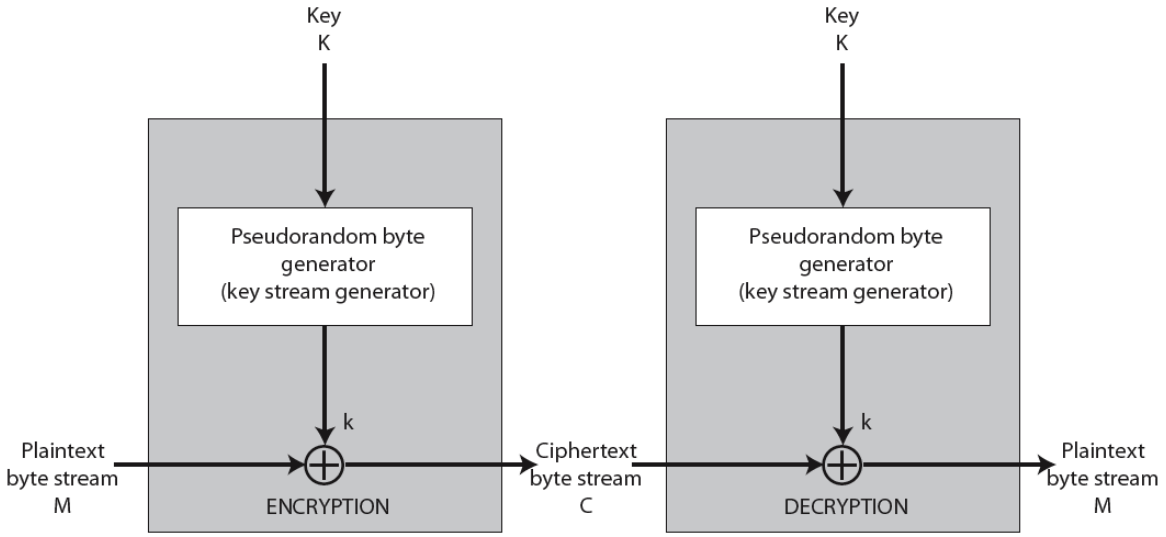


(a) CTR Mode (b) OFB Mode

(V, K) pair is the seed

Stream Ciphers

- process the message bit by bit
- Simply stating
  - a key and a Pseudo Random Number Generator (PRNG) is used to create a (pseudo) random key stream
  - keystream and the plaintext bitwise XORed to create the ciphertext
  - ciphertext is XORed with the same keystream to restore the plaintext



Some Stream Cipher Design Considerations

- A PRNG should eventually repeat
  - long period makes cryptanalysis difficult
- statistically randomness
  - e.g. approx. equal number of 0's and 1's
- large enough key (128-bit would be good to guard against brute-force attacks)

Stream Ciphers

- randomness of keystream destroys any statistical properties in the message
  - as in Vernam cipher and one-time pads
- Better than block ciphers in terms of
  - code space (implementations are simple)
  - throughput (faster per bit en/decryption)
- but must never use the same keystream more than once
  - otherwise the cryptanalyst can XOR two ciphertext streams and find out XOR of two plaintext streams
    - not so difficult to crack

Stream Ciphers

- are useful if data are transferred as a stream
  - web browser
  - voice
  - video
- actually any block cipher can be used as a stream cipher
  - CFB mode of operation (and OFB and CTR)

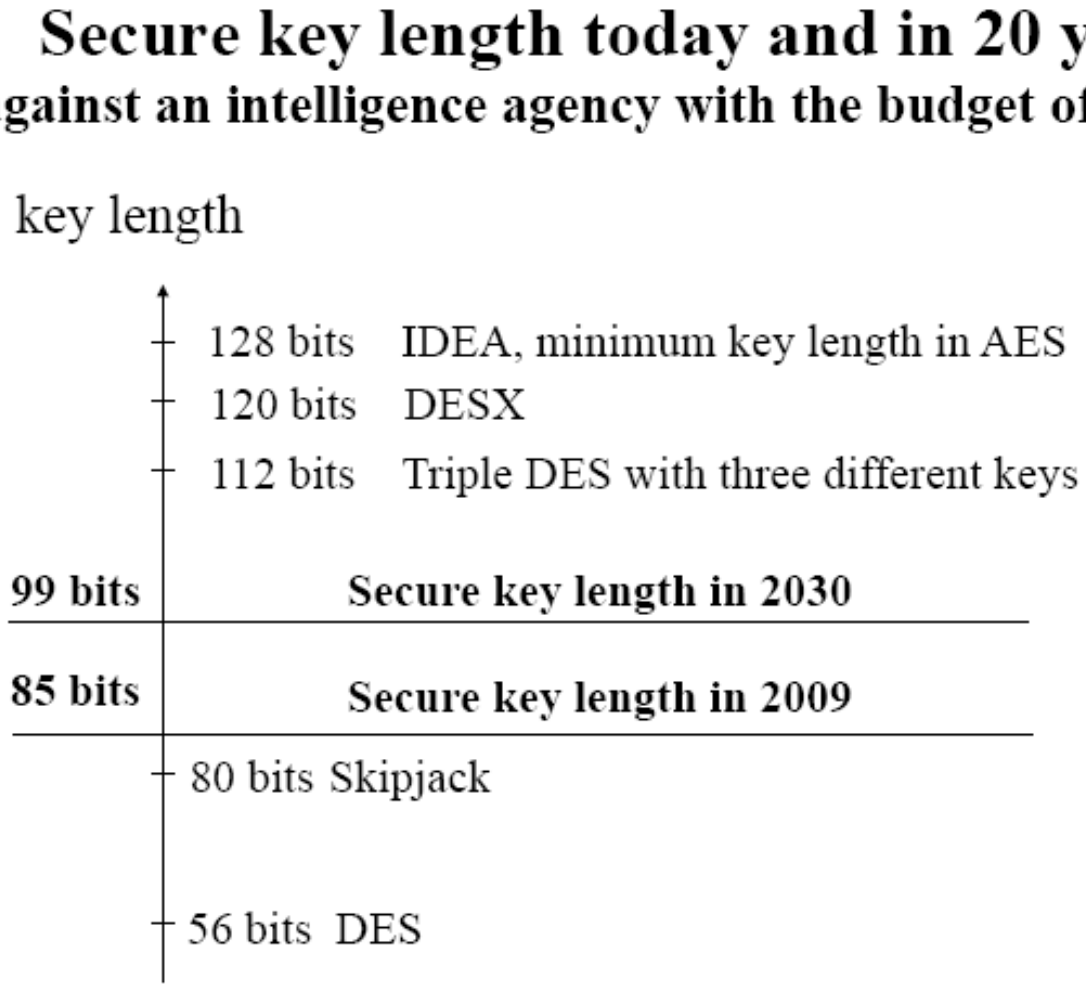
RC4

- Ron's Code 4
- Yet another cipher designed by Ron Rivest
  - owned by RSA Inc.
  - was kept as a trade secret, but in 1994 anonymously posted on the Internet
- variable key size, byte-oriented stream cipher
- simple but effective
  - 8 to 16 machine operations per output byte
- widely used (SSL/TLS, WEP/WPA)
- Some attacks reported, but not practical for key size greater than 128-bit
- However, WEP has a problem due to RC4 key generation
  - not a problem of RC4 in particular

and other symmetric ciphers

- CAST
- Skipjack
- Serpent
- Twofish
- RC6
- Mars
- SAFER+

Discussion



Courtesy of Kris Gaj

Discussion

- Assuming ~92-bit is secure enough for today and Moore's Law continues
  - 1 bit per 18 months to be added
    - 2020's: 93-bit (approx.)
    - 2040's: 107-bit (approx.)
  - with 128-bit, AES we will be secure for a long time
- unless a new efficient cryptanalysis method is found
  - known cryptanalysis methods are not practical for secure key sizes for 3DES, AES, IDEA, etc. (except DES of course)