PB173 - Tématický vývoj aplikací v C/C++ (jaro 2016)

Skupina: Aplikovaná kryptografie a bezpečné programování

Petr Švenda svenda@fi.muni.cz

Konzultace: A.406, Pondělí 15-15:50

CROCS
Centre for Research on Cryptography and Security

OPTIMIZING CRYPTO

Optimizing crypto

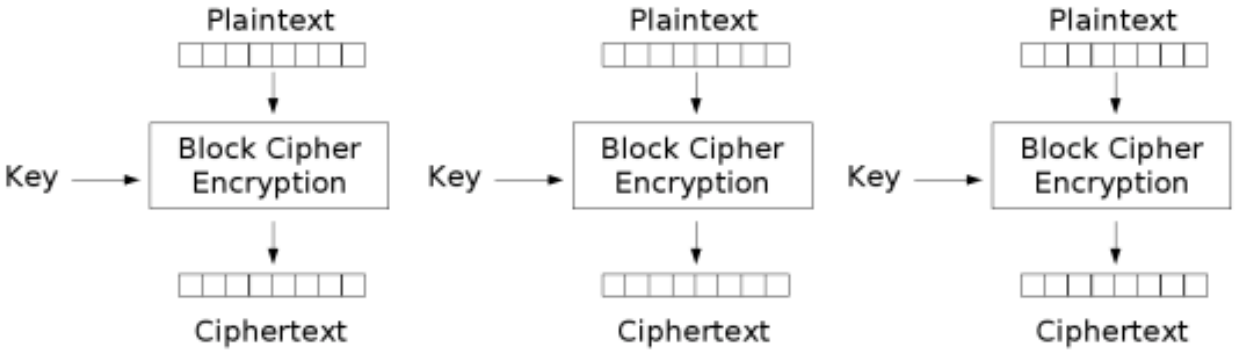
- Clever tricks both on design and implementation
  - optimization of both algorithm and mode used
  - see aestab.h and aesopt.h for example
  - https://github.com/BrianGladman/AES
- Possibility for pre-computation
  - code itself: macros, templates, static arrays
  - pre-computed tables
    - AES optimized with large tables
    - table lookup only implementation (AES/DES)
    - see http://cr.yp.to/aes-speed.html (Bernstein, Schwabe)
  - pre-computed key stream (if mode supports)
    - key stream in advance, then simple xor

Parallelization of operations

- Speedup by parallel execution
- Purpose build hardware
  - cryptographic coprocessors
  - e.g., fast modulo exponentiation
- Using multiple CPU cores
  - multiple threads running
  - Worker thread
    - AfxBeginThread(mywork), http://msdn.microsoft.com/en-us/library/69644x60%28v=VS.80%29.aspx
    - QThread, https://doc.qt.io/qt-5/qthread.html
  - When working with threads, don't forget to synchronize
    - CreateMutex, Qmutex…

Mode of cipher usage - ECB

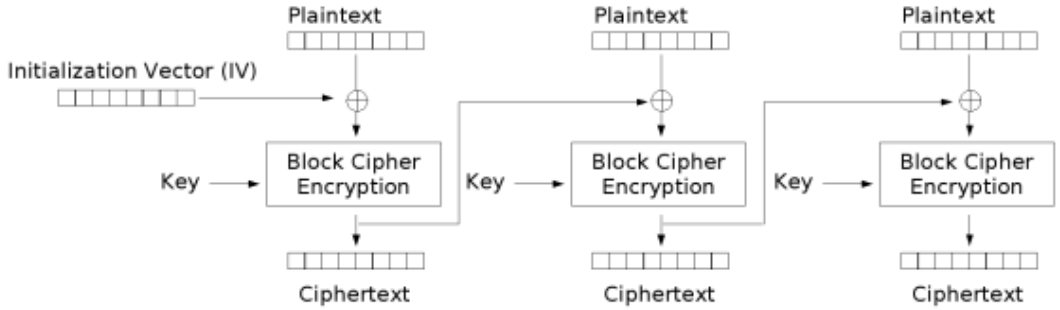
- ECB (Electronic Code Book mode)
  - used for block ciphers
  - processing of one block does not influence others
- Main problem
  - same data with same key result in same ciphertext
  - attacker c



Electronic Codebook (ECB) mode encryption

Mode of cipher usage - CBC

- CBC (Cipher Back Chaining mode)
  - used for block ciphers
  - previous cipher block is xored with next plaintext
  - prefer before ECB
- Problem
  - Initialization vector must be somehow shared
  - (random first block)



Cipher Block Chaining (CBC) mode encryption

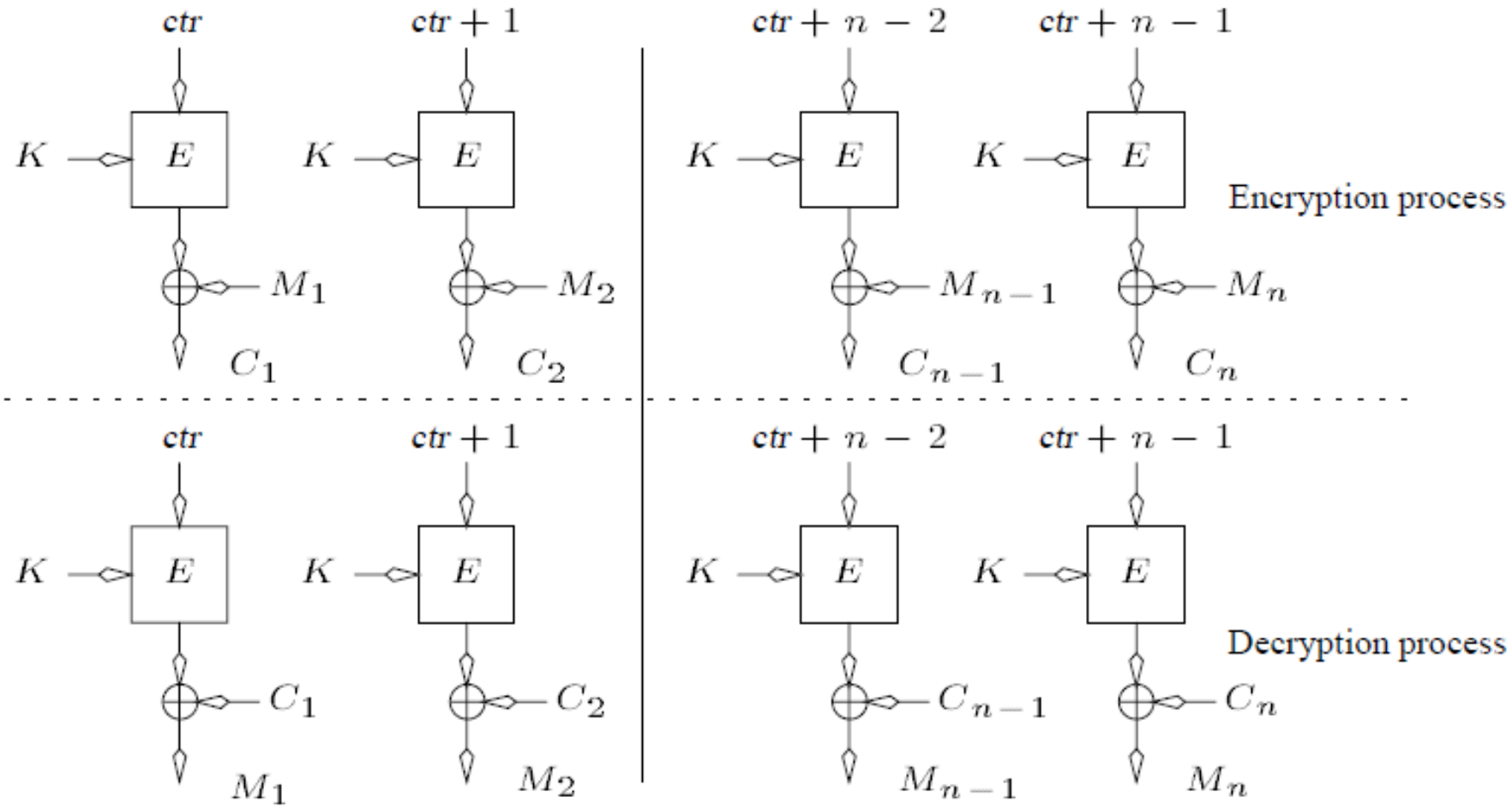
Parallelization of modes

- Assume that algorithm itself is sufficiently optimized
- Algorithm is used in some mode
  - e.g., block encryption modes (ECB, CBC…)
- We need parallelizable modes!
  - CBC encryption is not parallelizable
    - (decryption is – why?)
- Counter (CTR) mode

Counter (CTR) mode for encryption

- Mode approved by NIST (US standardization)
  - http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf
- Designed for confidentiality with parallelization and pre-computation in mind
- Key stream is produced by iterated encryption of the incremental counter
  - counter is incremented for each new block
  - key stream is then xored to message
  - **key stream(== counter) must not repeat with same key**

Counter (CTR) mode for encryption



http://www.mindspring.com/~dmcgrew/ctr-security.pdf

Practical assignment

- Speed-up encryption of data packets between two clients with CTR mode
  - Divide packet into multiple parts
  - Use parallel threads to protect parts of data packet
    - number of available cores is parameter for function
    - (at least one thread required ;))
- Document performance gains
  - speed before and after the optimization (can you increase speed linearly?)
  - What is length of packet for which multiple threads brings speedup benefit? (overhead with running threads)

Submissions, deadlines

- Upload application source codes as single zip file into IS Homework vault (Crypto - 8. homework (Threads))
- DEADLINE 25.4. 12:00
  - 0-10 points assigned