Systems' Security | *Segurança de Sistemas*

Symmetric Cryptography – Modern Techniques

Miguel Frade

POLITÉCNICO DE LEIRIA
ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO

Overview

OVERVIEW

Learning Objectives

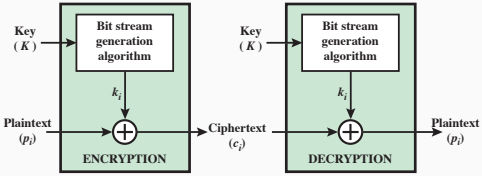
After this chapter, you should be able to:

1. Understand the distinction between stream ciphers and block ciphers
2. Present an overview of the Feistel cipher
3. Present an overview of Data Encryption Standard (DES)
4. Analyze the security of multiple encryption schemes
5. Compare and contrast ECB, CBC, CFB, OFB, and counter modes of operation

Introduction

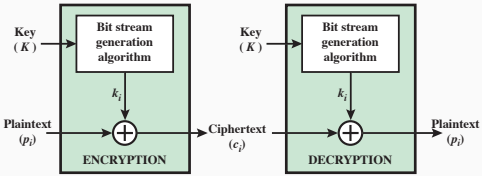
Methods of plaintext processing

stream cipher – encrypts a digital data stream one bit or one byte at a time

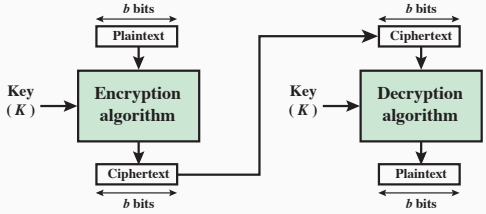


Methods of plaintext processing

stream cipher – encrypts a digital data stream one bit or one byte at a time



block cipher – a block of plaintext is treated as a whole and used to produce a ciphertext block of equal length. Typically, a block size of 64 or 128 bits



Nonsingular transformation

A block cipher operates on a plaintext block of n bits to produce a ciphertext block of n bits. A transformation is called reversible, or nonsingular, if there is a **unique** ciphertext block for each of the 2^n possible different plaintext, *i. e.* for decryption to be possible.

Nonsingular transformation

A block cipher operates on a plaintext block of n bits to produce a ciphertext block of n bits. A transformation is called reversible, or nonsingular, if there is a **unique** ciphertext block for each of the 2^n possible different plaintext, *i. e.* for decryption to be possible.

Ideal block cipher

For each 2^n possible different plaintext blocks there is an arbitrary substitution map, *i. e.* the mapping itself is the key for encryption.

- for $n = 4$ the key size is: 4 bits $\times 2^4$ possible combinations $= 64$ bits
- for $n = 64$ the key size is: 64 bits $\times 2^{64} = 2^{70}$ bits $\rightarrow$ 1 073 741 824 Tera Bytes

Feistel Cipher

Feistel Cipher Motivations

- a small block size, such as $n = 4$, then the system is equivalent to a classical substitution cipher $\rightarrow$ vulnerable to a statistical analysis
- a large block size is able to hide the statistical characteristics of the source plaintext $\rightarrow$ an arbitrary reversible substitution cipher for a large block size is not practical
- an approximation to the ideal block cipher is needed

Product Cipher

the execution of two or more simple ciphers in sequence that the final result, or product, is cryptographically stronger than any of the component ciphers.

Product Cipher

the execution of two or more simple ciphers in sequence that the final result, or product, is cryptographically stronger than any of the component ciphers.

Feistel Cipher Structure

- product cipher $\rightarrow$ approximation to the ideal block cipher
- proposes the alternate usage of **substitutions** and **transpositions** functions
- applies the concepts of **diffusion** and **confusion**

To thwart cryptanalysis

Diffusion

Diffusion seeks to make the statistical relationship between **the plaintext and ciphertext** as complex as possible:

- the statistical structure of the plaintext is dissipated into long-range statistics of the ciphertext.
- each plaintext digit affect the value of many ciphertext digits
- achieved by repeatedly performing some permutation on the data followed by applying a function to that permutation;

To thwart cryptanalysis

Diffusion

Diffusion seeks to make the statistical relationship between **the plaintext and ciphertext** as complex as possible:

- the statistical structure of the plaintext is dissipated into long-range statistics of the ciphertext.
- each plaintext digit affect the value of many ciphertext digits
- achieved by repeatedly performing some permutation on the data followed by applying a function to that permutation;

Confusion

Confusion seeks to make the relationship between the statistics of **the ciphertext and the value of the encryption key** as complex as possible

- achieved by the use of a complex substitution algorithm

Feistel Cipher depends on the choice of the following parameters

- **block size** – larger block sizes mean greater security, but reduced operation speed for a given algorithm, common size was 64 bits, now is 128 bits

Feistel Cipher depends on the choice of the following parameters

- **block size** – larger block sizes mean greater security, but reduced operation speed for a given algorithm, common size was 64 bits, now is 128 bits
- **key size** – larger key size means greater security, but may decrease performance, common size nowadays is 128 bits

Feistel Cipher depends on the choice of the following parameters

- **block size** – larger block sizes mean greater security, but reduced operation speed for a given algorithm, common size was 64 bits, now is 128 bits
- **key size** – larger key size means greater security, but may decrease performance, common size nowadays is 128 bits
- **number of rounds** – multiple rounds offer increasing security, a typical value is 16 rounds

Feistel Cipher depends on the choice of the following parameters

- **block size** – larger block sizes mean greater security, but reduced operation speed for a given algorithm, common size was 64 bits, now is 128 bits
- **key size** – larger key size means greater security, but may decrease performance, common size nowadays is 128 bits
- **number of rounds** – multiple rounds offer increasing security, a typical value is 16 rounds
- **subkey generation algorithm** – greater complexity should lead to greater difficulty of cryptanalysis

Feistel Cipher depends on the choice of the following parameters

- **block size** – larger block sizes mean greater security, but reduced operation speed for a given algorithm, common size was 64 bits, now is 128 bits
- **key size** – larger key size means greater security, but may decrease performance, common size nowadays is 128 bits
- **number of rounds** – multiple rounds offer increasing security, a typical value is 16 rounds
- **subkey generation algorithm** – greater complexity should lead to greater difficulty of cryptanalysis
- **round function F** – greater complexity generally means greater resistance to cryptanalysis

Feistel Cipher depends on the choice of the following parameters

- **block size** – larger block sizes mean greater security, but reduced operation speed for a given algorithm, common size was 64 bits, now is 128 bits
- **key size** – larger key size means greater security, but may decrease performance, common size nowadays is 128 bits
- **number of rounds** – multiple rounds offer increasing security, a typical value is 16 rounds
- **subkey generation algorithm** – greater complexity should lead to greater difficulty of cryptanalysis
- **round function F** – greater complexity generally means greater resistance to cryptanalysis
- **fast encryption/decryption** – speed of execution of the algorithm might be a concern in embedded applications

Feistel Cipher depends on the choice of the following parameters

- **block size** – larger block sizes mean greater security, but reduced operation speed for a given algorithm, common size was 64 bits, now is 128 bits
- **key size** – larger key size means greater security, but may decrease performance, common size nowadays is 128 bits
- **number of rounds** – multiple rounds offer increasing security, a typical value is 16 rounds
- **subkey generation algorithm** – greater complexity should lead to greater difficulty of cryptanalysis
- **round function F** – greater complexity generally means greater resistance to cryptanalysis
- **fast encryption/decryption** – speed of execution of the algorithm might be a concern in embedded applications
- **ease of analysis** – if the algorithm can be concisely and clearly explained, it is easier to analyze for cryptanalytic vulnerabilities and therefore develop a higher level of assurance of its strength

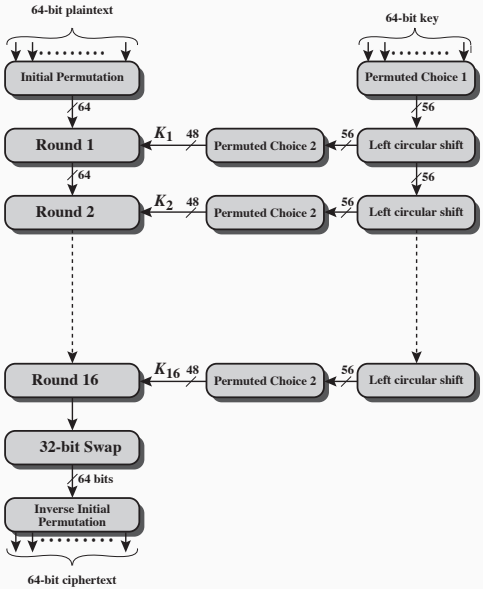
Data Encryption Standard

History of DES

- DES was issued in 1977 by the National Bureau of Standards as **FIPS PUB 46**
- the algorithm itself is referred to as the Data Encryption Algorithm (DEA)
- data are encrypted in 64-bit blocks using a 56-bit key
- In 1994, NIST reaffirmed DES for federal use for another 5 years for the protection of non-classified information
- In 1999, NIST issued a new version of its standard **FIPS PUB 46-3** that DES should be used only for legacy systems and 3DES should be used instead
- until 2001 (when AES was introduced) was the most widely used encryption algorithm

Internal scheme of DES Encryption Algorithm

- the algorithm expects a 64-bit key as input, but only 56 of these are ever used, the other 8 bits can be used as parity bits or simply set arbitrarily



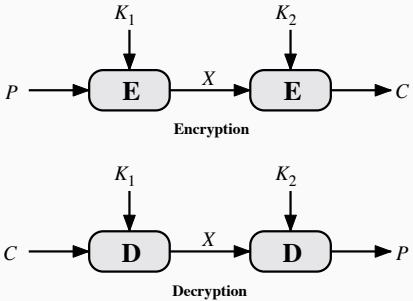
Multiple encryption

- due to its short key and the increased computational power, DES can now be brute-forced
  - in 1997 the first DES Challenge was solved in 96 days by the DESCHALL Project
  - in 1998 the DES Challenge II-1 was solved in 39 days by distributed.net
  - still in 1998 Deep Crack decrypted a message after only 56 hours
  - in 1999 the DES Challenge III was solved in 22h15m by Deep Crack

Multiple encryption

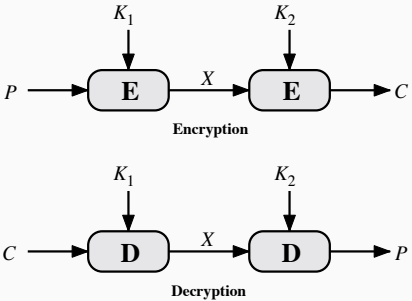
- due to its short key and the increased computational power, DES can now be brute-forced
  - in 1997 the first DES Challenge was solved in 96 days by the DESCHALL Project
  - in 1998 the DES Challenge II-1 was solved in 39 days by distributed.net
  - still in 1998 Deep Crack decrypted a message after only 56 hours
  - in 1999 the DES Challenge III was solved in 22h15m by Deep Crack
- to increase security and preserve the investment in software and equipment it is possible to use multiple encryption with DES and multiple keys
  - Double DES was created

Multiple encryption – Double DES



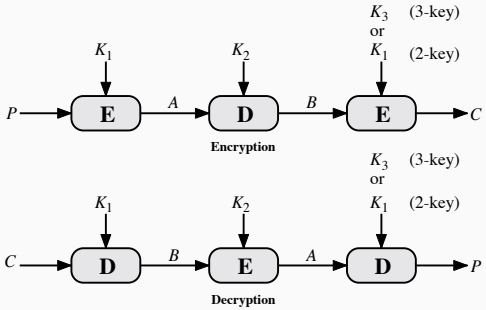
- apparently has a key strength of $2 \times 56 = 112$ bits

Multiple encryption – Double DES



- apparently has a key strength of $2 \times 56 = 112$ bits
- meet-in-the-middle attack
  - if $C = E_{k2}(E_{k1}(P))$, then $X = E_{k1}(P) = D_{k2}(C)$
  - the compuational effort to crack 2DES is 2^{56}
  - therefore, this type of double encryption is not useful

Triple DES (3DES)



- three stages of encryption solve the meet-in-the-middle attack
- Triple DES (3DES) was created with 2 versions
  - two-key (k_1, k_2) triple encryption, 112 bits key compatible with single DES (1DES) if $k_1 = k_2$
  - three-key (k_1, k_2, k_3) triple encryption, 168 bits key

1. What is the difference between **stream** and **block** ciphers?
2. What is the problem of the **Ideal Block Cipher**?
3. What is the **product chiper**?
4. What are the parameters that Feistel took into account when designing a cipher?
5. Does multiple encryption always increase the robustness of the ciphertext?
6. Write the short representation of the 3DES with 168 bits key

   $A \rightarrow B$:

Block Cipher Modes of Operation

Block Ciphers

- take a fixed-length block of text of length b bits and a key
- produces a b bits block of ciphertext
- the amount of plaintext to be encrypted can be greater than b bits
  - the plaintext must braked into b bits blocks

Block Ciphers

- take a fixed-length block of text of length b bits and a key
- produces a b bits block of ciphertext
- the amount of plaintext to be encrypted can be greater than b bits
  - the plaintext must braked into b bits blocks
- multiple blocks of plaintext are encrypted using the same key
  - several security issues might arise
  - several modes of operation have been created to address those issues

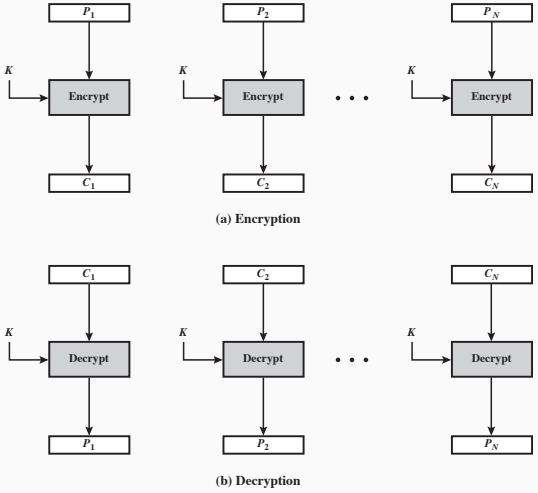
Modes of Operation for any symmetric block cipher

- Electronic Codebook (**ECB**)
- Cipher Block Chaining (**CBC**)
- Cipher Feedback (**CFB**)
- Output Feedback (**OFB**)
- Counter (**CTR**)

Mode	Description	Typical Application
ECB	Each block of plaintext bits is encoded independently using the same key.	Secure transmission of single values, *e. g.* an encryption key
CBC	The input to the encryption algorithm is the XOR of the next block of plaintext and the preceding block of ciphertext.	General-purpose block-oriented transmission. Authentication.
CFB	Input is processed s bits at a time. Preceding ciphertext is used as input to the encryption algorithm to produce pseudorandom output, which is XORed with plaintext to produce next unit of ciphertext.	General-purpose stream-oriented transmission. Authentication.
OFB	Similar to CFB, except that the input to the encryption algorithm is the preceding encryption output, and full blocks are used.	Stream-oriented transmission over noisy channel *e. g.* satellite communication
CTR	Each block of plaintext is XORed with an encrypted counter. The counter is incremented for each subsequent block.	General-purpose block-oriented transmission. Useful for high-speed requirements

Electronic Codebook (**ECB**)

- is the simplest mode
- plaintext is handled one block at a time
- each block of plaintext is encrypted using the same key
- for the same key, 2 equal plaintext blocks result in the same ciphertext
- should be used only to secure messages shorter than a single block
- ECB mode is not secure for lengthy message

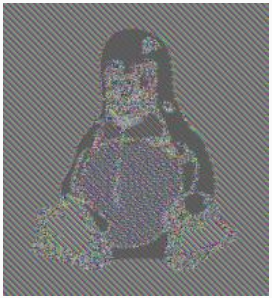


Electronic Codebook (**ECB**)

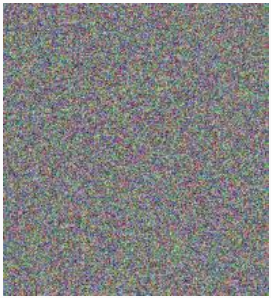
- example of pattern repetition in lengthy messages
  source `https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation`

Original image

ECB mode

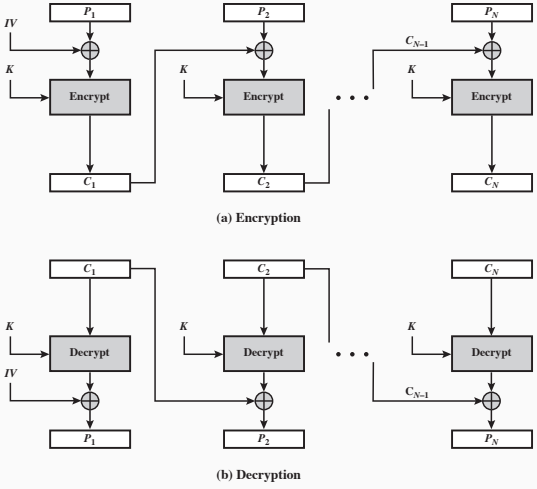


Other mode



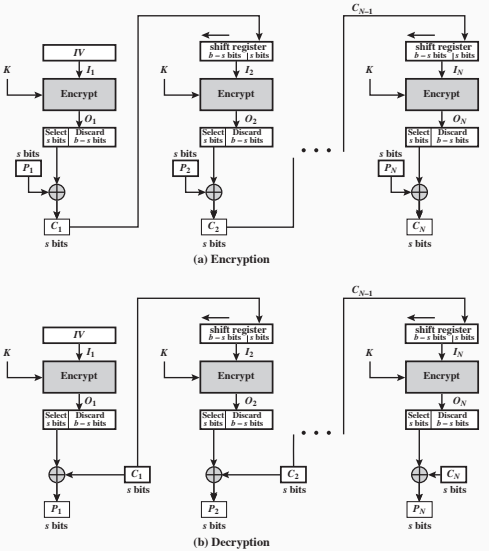
Cipher Block Chaining (**CBC**)

- for the same key, 2 equal plaintext blocks result in different ciphertext
- repeating patterns of the plaintext blocks are not exposed on the ciphertext
- an initialization vector (IV) is XORed with the first block of plaintext
- the IV must be known to both the sender and receiver, but be unpredictable by a third party
- the IV should be protected against unauthorized changes, *e. g.* using ECB encryption
- propagates eventual transmission errors



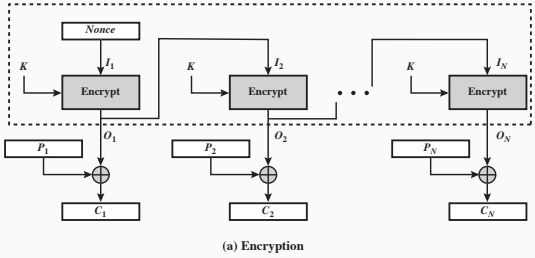
Cipher Feedback (**CFB**)

- converts a block cipher into a stream cipher
- no need to pad a message in the last block
- can operate in real time
- the ciphertext has the same length of the plaintext
- an initialization vector (IV) is needed
- ideal to encrypt character streams
- propagates eventual transmission errors
- doesn't require decryption algorithm



Output Feedback (**OFB**)

- converts a block cipher into a stream cipher
- similar structure to CFB
- the output of the encryption function is fed back as the input for the next block
- eventual errors in transmission do not propagate
- more vulnerable to a message stream modification attack than CFB
- doesn't require decryption algorithm

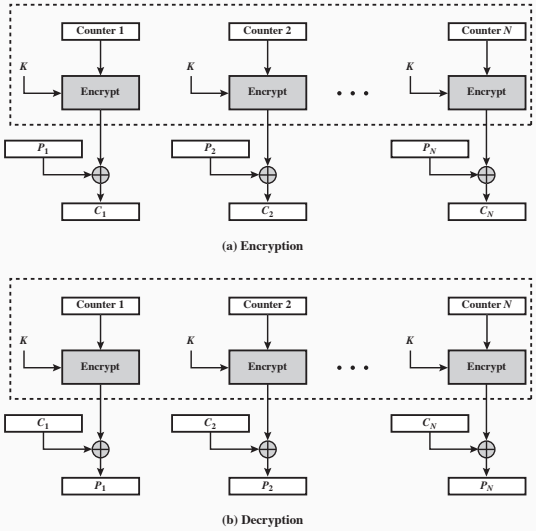


(a) Encryption

(b) Decryption

Counter (CTR)

- uses a counter equal to the plaintext block size (similar to IV)
- the counter randomly is initialized and then incremented by 1 for each subsequent block
- simpler than CBC, CFB and OFB, but at least as secure
- doesn't require decryption algorithm
- no feed back
  - does not require padding
  - does not propagate transmission errors
  - is very efficient, encryption and decryption can be done in parallel



There are many other modes of operation

- but are out of the scope of this course
- many of them were specifically developed with encryption of large amounts of data in mind, *e. g.* storage devices
- some examples are XTS, LRW, XEX, CMC, EME
- for more information visit `https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation`

1. Why do some block cipher modes of operation only use encryption while others use both encryption and decryption?

1. Why do some block cipher modes of operation only use encryption while others use both encryption and decryption?
2. In the CBC mode transmission errors propagate. How many blocks can be affected by a single transmission error?

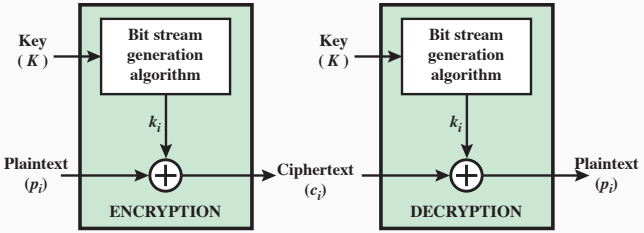
1. Why do some block cipher modes of operation only use encryption while others use both encryption and decryption?
2. In the CBC mode transmission errors propagate. How many blocks can be affected by a single transmission error?
3. Is it possible to perform encryption operations in parallel on multiple blocks of plaintext in CBC mode? How about decryption?

1. Why do some block cipher modes of operation only use encryption while others use both encryption and decryption?
2. In the CBC mode transmission errors propagate. How many blocks can be affected by a single transmission error?
3. Is it possible to perform encryption operations in parallel on multiple blocks of plaintext in CBC mode? How about decryption?
4. If a bit error occurs in the transmission of a ciphertext character in 8-bit CFB mode, how far does the error propagate?

Stream Ciphers

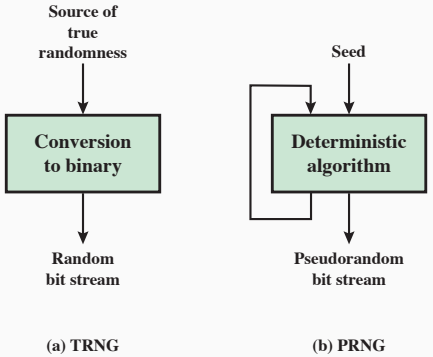
How stream ciphers work

- a key is input to a pseudorandom bit generator that produces a stream of 8-bit numbers that are apparently random
- the output of the generator, is combined with the plaintext stream using the bitwise exclusive-OR (XOR) operation



Random bits streams

- are used in key generation and encryption
- there are two classes of random bits generators
  1. **True Random Number Generators** (TRNGs) – produce bits non-deterministically using some physical source that produces random output
  2. **Pseudorandom Number Generators** (PRNGs) – computes bits deterministically using an algorithm



True Random Number Generators (TRNGs) are best fit to

- generate nonces (numbers used only once)
- generate keys (session keys and long term keys, such as RSA)

True Random Number Generators (TRNGs) are best fit to

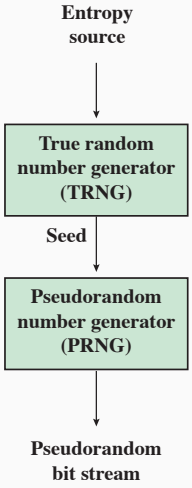
- generate nonces (numbers used only once)
- generate keys (session keys and long term keys, such as RSA)

RFC 4086 lists the following possible sources of randomness:

- sound/video input – the “input” from a sound digitizer with no source plugged in or from a camera with the lens cap on is essentially thermal noise and can provide reasonably high quality random bits
- disk drives – disk drives have small random fluctuations in their rotational speed due to chaotic air turbulence, low-level disk seek-time instructions produces a series of measurements with randomness
- to prevent bias, more than one source should be used

Pseudorandom Number Generators (PRNGs)

- are best fit to generate a predictable bit stream for symmetric stream encryption
- but the seed must be unpredictable



Comparison of PRNGs and TRNGs

	PRNGs	TRNGs
Efficiency	Very efficient	Generally inefficient
Determinism	Deterministic	Nondeterministic
Periodicity	Periodic	aperiodic

Stream ciphers versus block ciphers

- a stream cipher can be as secure as a block cipher of comparable key length
- stream ciphers are typically faster and use far less code than do block ciphers
- on block cipher keys can be reused, but not in a stream cipher
  - two plaintexts encrypted with the same key are easy to attack through cryptanalysis

Symmetric Algorithms

1. Modern **symmetric algorithms** can be dived into:

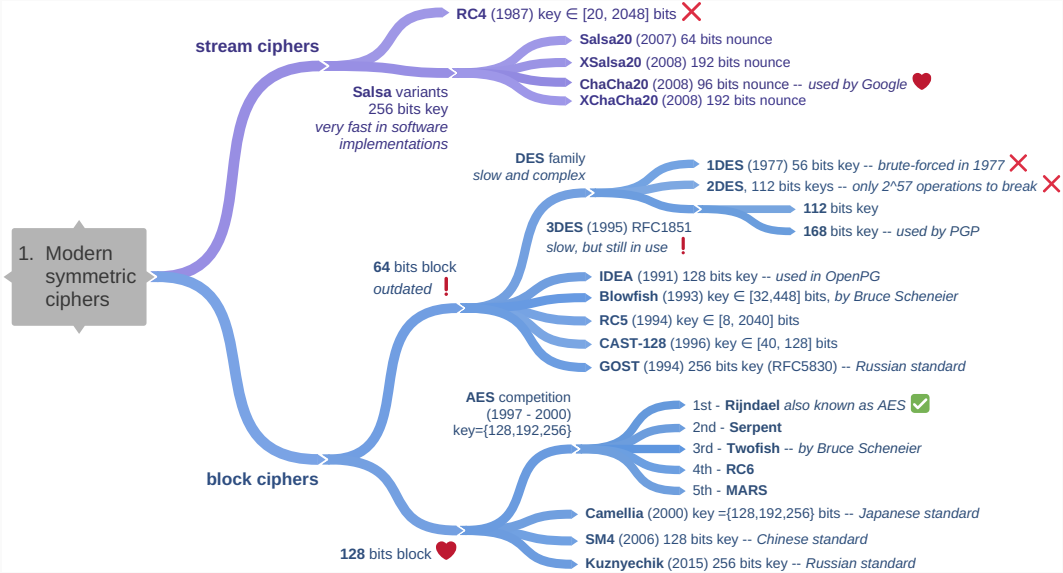
    1.1 Stream ciphers:

    - can encrypt one bit at a time
    - good for communications (HTTPs, SSH, *etc*)

1. Modern **symmetric algorithms** can be dived into:

    1.1 Stream ciphers:
    - can encrypt one bit at a time
    - good for communications (HTTPs, SSH, *etc*)

    1.2 Block ciphers:
    - encrypt a block of bits at a time
        - nowadays 128 bits block is the standard
    - good to encrypt large amounts of data, such as files
    - can be adapted to work as stream ciphers, but will be less efficient

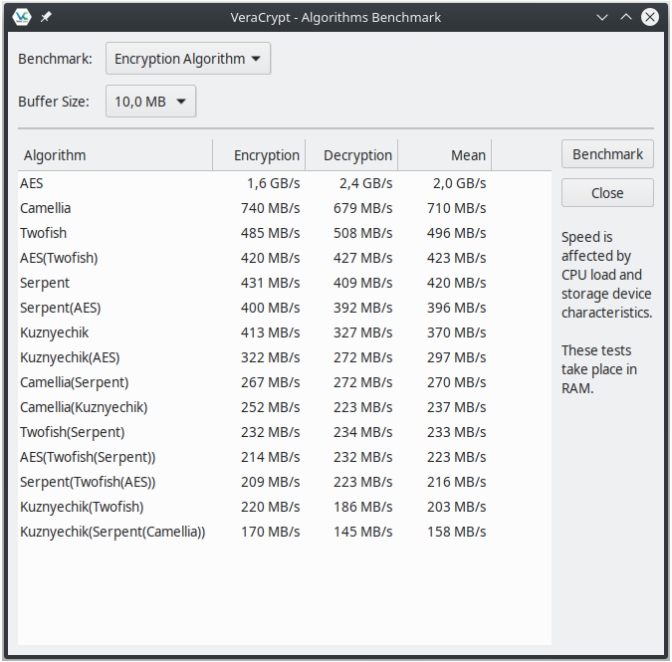

1. Modern symmetric ciphers
stream ciphers
RC4 (1987) key ∈ [20, 2048] bits
Salsa variants
256 bits key
very fast in software implementations
Salsa20 (2007) 64 bits nounce
XSalsa20 (2008) 192 bits nounce
ChaCha20 (2008) 96 bits nounce -- used by Google
XChaCha20 (2008) 192 bits nounce
block ciphers
64 bits block
outdated
DES family
slow and complex
1DES (1977) 56 bits key -- brute-forced in 1977
2DES, 112 bits keys -- only 2^57 operations to break
3DES (1995) RFC1851
slow, but still in use
112 bits key
168 bits key -- used by PGP
IDEA (1991) 128 bits key -- used in OpenPG
Blowfish (1993) key ∈ [32,448] bits, by Bruce Scheneier
RC5 (1994) key ∈ [8, 2040] bits
CAST-128 (1996) key ∈ [40, 128] bits
GOST (1994) 256 bits key (RFC5830) -- Russian standard
128 bits block
AES competition
(1997 - 2000)
key={128,192,256}
1st - Rijndael also known as AES
2nd - Serpent
3rd - Twofish -- by Bruce Scheneier
4th - RC6
5th - MARS
Camellia (2000) key ={128,192,256} bits -- Japanese standard
SM4 (2006) 128 bits key -- Chinese standard
Kuznyechik (2015) 256 bits key -- Russian standard

Some tools that use symmetric encryption algorithms:

- disk-based encryption:
  - Veracrypt (Windows, MacOS, Linux)
  - BitLocker (Windows)
  - ecryptfs (Linux)
  - dm-crypt (Linux, Android 7–9)
- file-based encryption:
  - EFS (Windows)
  - 7-zip (Windows, MacOS, Linux)
  - dm-crypt (Android 10–?)
- password managers:
  - keepass (Windows, MacOS, Linux, Android, iOS)

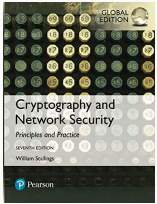
Veracrypt benchmark

hardware acceleration for AES has a big impact on performance



Algorithm	Encryption	Decryption	Mean
AES	1,6 GB/s	2,4 GB/s	2,0 GB/s
Camellia	740 MB/s	679 MB/s	710 MB/s
Twofish	485 MB/s	508 MB/s	496 MB/s
AES(Twofish)	420 MB/s	427 MB/s	423 MB/s
Serpent	431 MB/s	409 MB/s	420 MB/s
Serpent(AES)	400 MB/s	392 MB/s	396 MB/s
Kuznyechik	413 MB/s	327 MB/s	370 MB/s
Kuznyechik(AES)	322 MB/s	272 MB/s	297 MB/s
Camellia(Serpent)	267 MB/s	272 MB/s	270 MB/s
Camellia(Kuznyechik)	252 MB/s	223 MB/s	237 MB/s
Twofish(Serpent)	232 MB/s	234 MB/s	233 MB/s
AES(Twofish(Serpent))	214 MB/s	232 MB/s	223 MB/s
Serpent(Twofish(AES))	209 MB/s	223 MB/s	216 MB/s
Kuznyechik(Twofish)	220 MB/s	186 MB/s	203 MB/s
Kuznyechik(Serpent(Camellia))	170 MB/s	145 MB/s	158 MB/s

Questions?



Chapters 4, 7 and 8 of
William Stallings, Cryptography and Network Security: Principles and Practice, Global Edition, 2016