COMPUTER SYSTEMS SECURITY

Cryptography: some models

Cryptographic models

Definition

- cryptographic model
  - mostly, formal description of the security properties and assumptions of a cryptographic system
  - should define: adversarial capabilities; security goals[1]; security assumptions (environmental and operational details[2])...
  - so, "includes" *attack models* (see ahead)
  - really, there is little consensus on its exact definition...

1 e.g. confidentiality
2 e.g. computing resources

Types of Cryptographic models

- Standard
- Random Oracle
- Ideal Cipher
- Universal Composability
- ...

Standard model

- uses "real" (not ideal) crypto primitives, operated by real entities
- security is open, based on hard mathematical problems, not on obscurity

Random oracle[1] model

(next page)

1 An ideal (computational) oracle is a "black box" that is able to produce a (true) solution for any instance of a given computational problem. For instance, for a decision problem, it gives the correct, trustworthy answer.

...Cryptographic models: types...

Random oracle

- uses crypto operations identical to those performed by *ideal oracle* that
  - for each input, outputs a unique and (truly) random value, uniformly distributed in the (infinite) co-domain
  - is deterministic, outputting the same value every time the same input is submitted to it

Ideal cipher model

- focus on block ciphers that behave like *perfect random permutations*
- each cipher key defines a completely random bijection (*P* <--> *C*)

Universal composability model

- focus on protocols and their inter-operation in order to assure aggregate security

Examples

→ Annex: Cryptographic Models applied to a simple situation

Attack models[1]

Definition

- specification of the assumptions attributed to cryptanalysts targeting[2] a cryptographic system
- cryptographer's defense varies with specific assumptions

Perpsectives

- cryptanalist's
- cryptographer's

1 or: classification of attacks
2 attempting to break

Attack models: cryptanalyst perspective

- goals
- knowledge
- capabilities

Goals of cryptanalyst:

- capture the keys
  - break the system, as cryptographic protection failed!
- capture plaintexts
  - partial break of confidentiality protection
- forge (or replay) plaintexts
  - partial break of integrity protection
- deny services (or communication)
  - break of availability protection

Knowledge of cryptanalyst:

- knows almost nothing of system's details (black-box, closed system)
  - in principle, great attack difficulty
- knows some system's details (grey-box system)
  - before attack, additional information gathering is needed (e.g. with social engineering)
- knows all system's details (white-box, open system)
  - in principle, least difficult to attack (unless strength of system relies in its inner robustness...)

Capabilities of cryptanalyst:

- standard
  - limitations are just the amount of time and computational power available
    - so, knowledge is no obstacle!
- passive (mostly)
  - basic
    - has access to ciphertexts only (that is not able to choose)[1]
  - known plaintext
    - some (plaintext, ciphertext) pairs are available[2]
- active interaction (next page)

1 could launch Ciphertext-Only Attack (COA)
2 could launch Known-Plaintext Attack (KPA)

...Attack models: capabilities of cryptanalyst...

- active interaction
  - basic
    - can query and interact with target system but not of much use
  - chosen plaintext
    - is able to prepare plaintexts and obtain their ciphertexts[1] [2]
  - adaptive chosen plaintext (real time interaction?...)
    - is able to iteratively query the system with a succession of plaintexts, after receiving corresponding ciphertexts[3]
  - chosen ciphertext
    - is able to prepare ciphertexts and obtain their deciphered counterparts[4][5]
    - is able to prepare ciphertexts that will decipher to predictable plaintexts
  - adaptive chosen ciphertext (next page)

1 e.g. from an encryption oracle. Trivial with public key cryptography! Why?
2 could launch Chosen-Plaintext Attack (CPA)
3 could launch Adaptive Plaintext Attack (CPA2)
4 e.g. from a "decryption" oracle (readily available in digital signatures' attacks, as public key is used for "deciphering" signed docs)
5 could launch Chosen-Ciphertext Attack (CCA)

...Attack models: capabilities of cryptanalyst, active interaction...

- ...active interaction (continued)
  - adaptive chosen ciphertext
    - is able to iteratively query the system with a succession of ciphertexts, after receiving corresponding plaintexts[1]
- side-channel
  - is able to gather marginal information unexpectedly related to the cryptographic operations: electronic noise, sound, elapsed time...
- social engineering
  - is able to trick some humans to give away partial or essential secrets

1 could launch Adaptive Ciphertext Attack (CCA2) (more on this below)

Defense models: cryptographer perspective

- defense will depend on previous attack perspectives
- can be guided by more or less formal approaches, included in the mentioned cryptographic *models*
- otherwise, or in parallel, a multitude of tests can be conducted on system, including:
  - testing for info leakage, such as:
    - padding oracle (error handling)
    - timing analyses (difference in handling of correct and malformed texts[1])

1 either plain or cipher

Example of defense, hardening property: *indistinguishability*

- some defense tests have a game format
- they challenge the attacker to distinguish between two texts (plain or cipher) in certain circunstances; if the attacker fails, the system is secure under the type of test. Some examples:
  - IND-CPA game (non adaptive):
    - the attacker chooses 2 plaintexts and asks the system to encipher them
    - the attacker fails if he is not able (beyond mere chance) to say whose ciphertext belongs to which plaintext
    - **failure proves that, with this cryptographic system:**
      - "*An attacker not only cannot decrypt a message, but also cannot tell which ciphertext maps to which plaintext*!"
  - IND-CCA2 game (adaptive[1]): (next page)

1 the 2 in CCA2 means adaptive!

...Attack models: cryptographer perspective: indistinguishability...

- IND-CCA2 game (adaptive[1]):
  - besides an encryption oracle, a "decryption" one is also available to be used interactively by attacker
  - the attacker chooses 2 plaintexts and asks the system to encipher them
  - the attacker fails if he is not able (beyond mere chance) to say whose ciphertext belongs to which plaintext...
    - of course...without using the decryption oracle on the ciphertexts!
  - **failure proves that, with this cryptographic system:**
    - "*An attacker not only cannot decrypt a message, but also cannot tell which ciphertext maps to which plaintext, even by training as much as he wants*!"

- Of course, a system that passes the last test (IND-CCA2) is much more secure than if it just passes the first (IND-CPA)!

1 the 2 in CCA2 means adaptive!

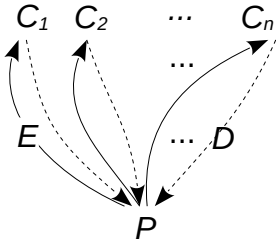
General enciphering schemes

Definition

- sets of algorithms and protocols used to transform plaintext (clear data) into ciphertext (concealed data) in such a way that unauthorized users cannot reverse the transformation.

Types

- deterministic encipherment
  - the same ciphertext is always produced for a given plaintext and key
- probabilistic encipherment [FIG]
  - different ciphertexts are, in general, produced for a given plaintext and key[1]
- format-preserving encipherment
  - ciphertext is produced in the same format[2] as the plaintext



1 An example is ElGamal's encryption system.
2 The meaning of "format" varies: only letters from English alphabet are used; *n*-bit block cipher (only *n*-bit numbers are accepted and produced), etc.

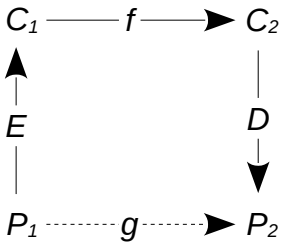
...General enciphering schemes: types...

- perfect secrecy encipherment
  - the ciphertext reveals no information at all about the plaintext
  - ideal goal: works even with an all-mighty cryptanalyst
- semantic security encipherment
  - the ciphertext possible informations about the plaintext, cannot be feasibly extracted
  - realistic goal: protects even with adaptive chosen plaintext attacks
- indistinguishable encipherment
  - a ciphertext does not reveal information to allow distinguishing which plaintext produced it from a group of chosen plaintexts[1]
- malleable encipherment
  - the ciphertext produced from a given plaintext can be modified in a way that the deciphered new plaintext is predictably related to the first[2]
  - dangerous: does not protect against (adaptive) chosen ciphertext attacks

1 or the distinction is no better then that of random guessing
2 Ex: one time pad is malleable. Show it with P="Bob", P'="Eve", by finding Q so that C(P)⊕Q deciphers to P', whatever the key.

...General enciphering schemes...

- homomorphic encipherment [mini-project?!...]
  - the ciphertexts are able to suffer computations that, when deciphered, are identical to related computations on the corresponding plaintexts
  - useful with cloud computing, as cloud server will not need to know clients' deciphering keys
  - Ex.: RSA is homomorphic![1]



- (perfect) forward secrecy encipherment[2]
  - the capture of a session key (and so being able to decipher the session) will not allow the decipherment of previous sessions
  - Also, knowledge of a long-term key does not allow the decipherment of past sessions.)[3]

1 In RSA, if n is the modulus and e the encryption exponent: $C = E(P) = P^e \bmod n$. The homomorphic property is immediate: $E(P1) * E(P2) = E(P1*P2)$.

2 This has to do more with key exchange schemes than with the encipherment operations by themselves

3 However, the breaking of the encipherment *algorithm*, in the sense of being able to operate it without a cryptographic key, might allow the decipherment of past sessions.

Cipher modes of operation[1]

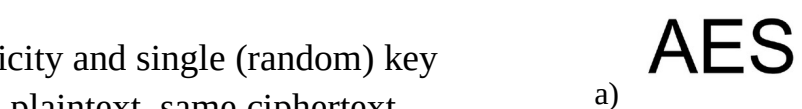
Base method

- $P = P_1\ P_2$ ... parts (blocks) of equal size
  - block size: 1 b, 1 B, 8 B (typical), 16 B (typical)...
- enciphering methods:
  - stream
    - $K = K_1 K_2 \ldots : C = E_{K1}(P_1)\ E_{K2}(P_2) \ldots =^2 K_1(P_1)\ K_2(P_2) \ldots$
  - block
    - $K : C = K(P_1)\ K(P_2) \ldots$
  - "mix" of previous
    - $K, k_1, k_2 \ldots^3 : C = E_K(P_1\ , k_1)\ E_K(P_2\ , k_2) \ldots = K_{k1}(P_1)\ K_{k2}(P_2) \ldots$

1 Necessary for the symmetric encipherment of "long" texts. But, in practice, almost any text is "long"!...
2 for simplicity
3 real single key with additional (and different) information per block: overall, looks like a different "virtual" key per block

Rationale for "operation modes"[1]

- stream
  - Pro: most secure[2]
  - Con: long, one-time usable, (random) key
- block
  - Pro: simplicity and single (random) key
  - Con: same plaintext, same ciphertext
    - if $P_1 = P_2$, then $C_1 = C_2$ [FIG]
- mixed
  - Pro: single (random) key
  - Con: added complexity
    - several possibilities

a) 

b)

Fig. a) original picture;
b) enciphered with AES 256b, ECB mode

1 Goal is *confidentiality* protection; *integrity* protection is not guaranteed: with some modes, even the "mixed", modifications of ciphertext might go undetected; for confidentiality and integrity protection, authenticated encipherment is used.
2 even *provable* secure with *One-time pad*

Pictures' notation

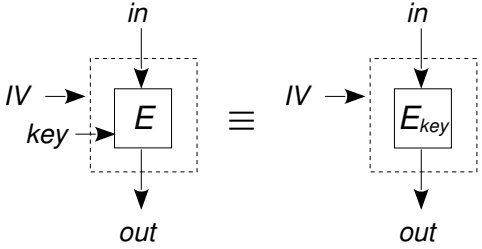


Fig. *IV* is Initialization Value (or Vector), public value that, as a rule, should be random.

Some operation modes

Stream method

- Some properties:
  - usually, $E = D =$ XOR[1] ($\oplus$)
  - no padding of last block
  - parallelizable en/deciphering
  - ultimate security: K_i random, one-time value
- Formulas:
  - $C_i = E_{ki}\ (P_i)$, $i>0$
  - usually, $P_i = E_{ki}\ (C_i)$
- Error propagation:[2]
  - exercise!

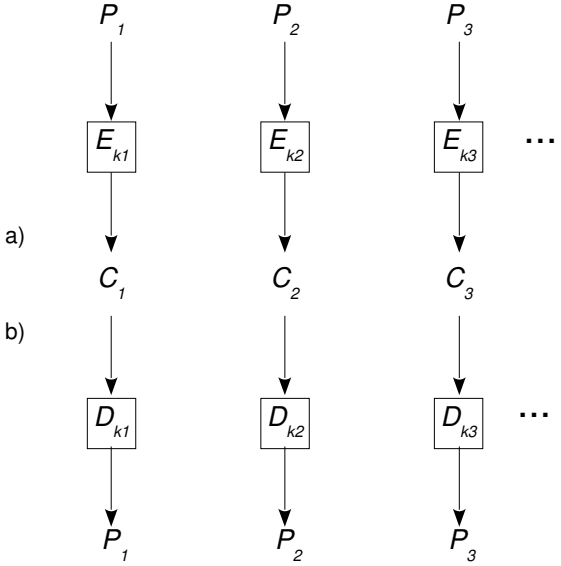


Fig. Use of plain stream method: a) enciphering; b) deciphering

1 bitwise

2 When at least one bit/byte of C_i is garbled, how that is reflected in following blocks.

Block method

- *ECB, Electronic Code Book*
- Some properties:
  - padding of last block
  - parallelizable en/deciphering
- Formulas:
  - $C_i = E_k (P_i)$, $i>0$
  - Write the decipherment formula. :-)
- Error propagation:
  - exercise!

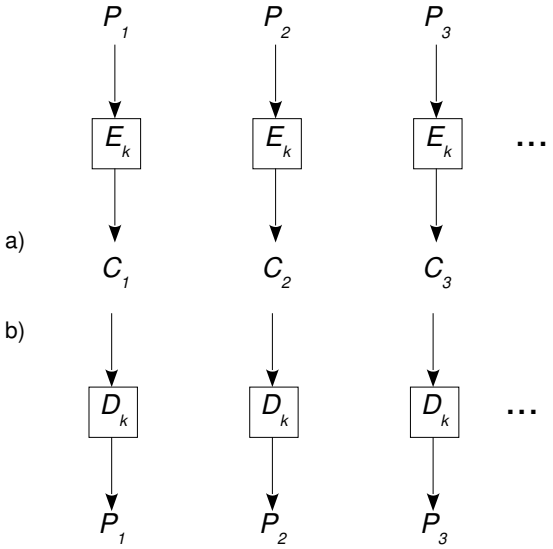


Fig. Use of (plain) block method: a) enciphering; b) deciphering.

“Mix” method: CTR

- *CTR, Counter Mode*
- Some properties:
  - *IV*[1] (random + counter)
  - no padding
  - parallelizable en/deciphering
- Formulas:
  - Write the en/decipherment formulas.
- Error propagation:
  - exercise!

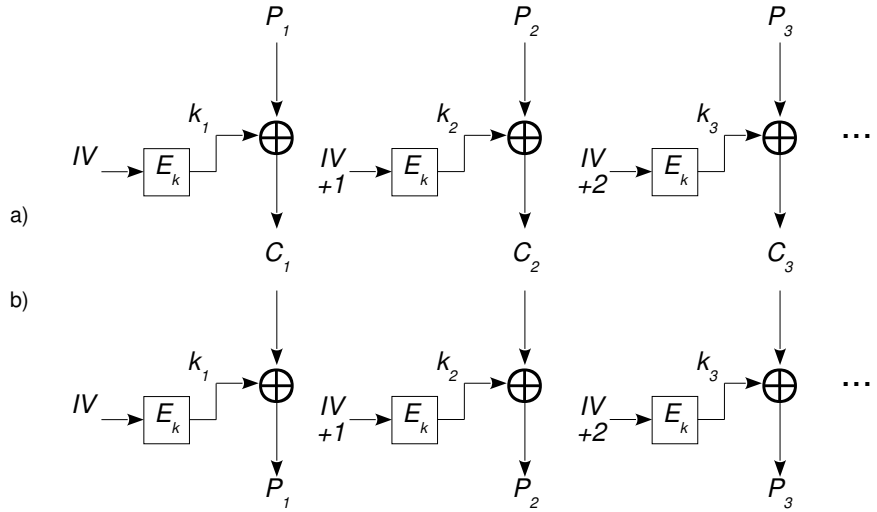


Fig. Use of “mixed” method CTR: a) enciphering; b) deciphering. (Notice the virtual keys k_i.)

1 public value that, as a rule, should be random

“Mix” method: CFB

- *CFB, Cipher FeedBack*
- Some properties:
  - *IV* (random)
  - no padding
  - not parallelizable enciphering; parallelizable deciphering
- Formulas:
  - $C_0 = IV$;
    $C_i = P_i \oplus E_k\,(C_{i-1})$, $i>0$
  - Write the decipherment formula.
- Error propagation:
  - exercise!

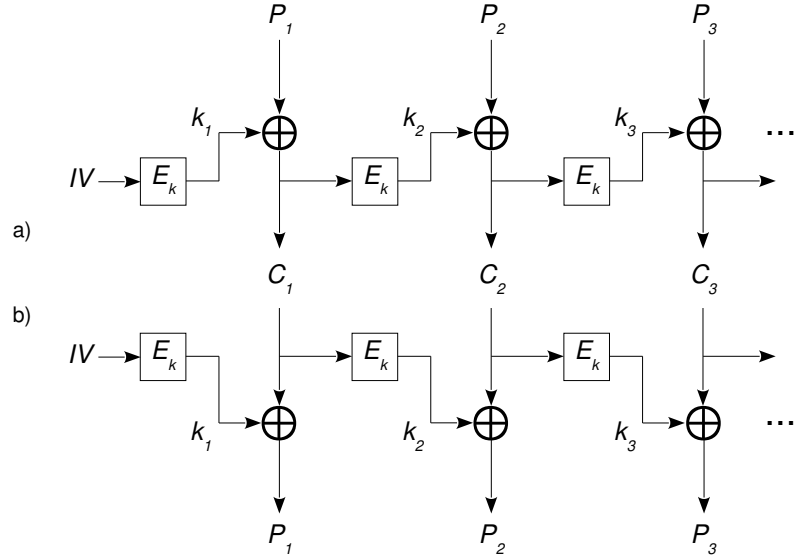


Fig. Use of “mixed” method CFB: a) enciphering; b) deciphering. (Notice the virtual keys k_i.)

"Mix" method: OFB

- *OFB, Output FeedBack*
- Some properties:
  - *IV* (random)
  - no padding
  - not parallelizable en/deciphering, but successive $E_k^i(IV)$ can be done in advance
- Formulas:
  - $C_i = P_i \oplus E_k^i\,(IV)$, $i \geq 0$
  - Write the decipherment formula.
- Error propagation:
  - exercise!

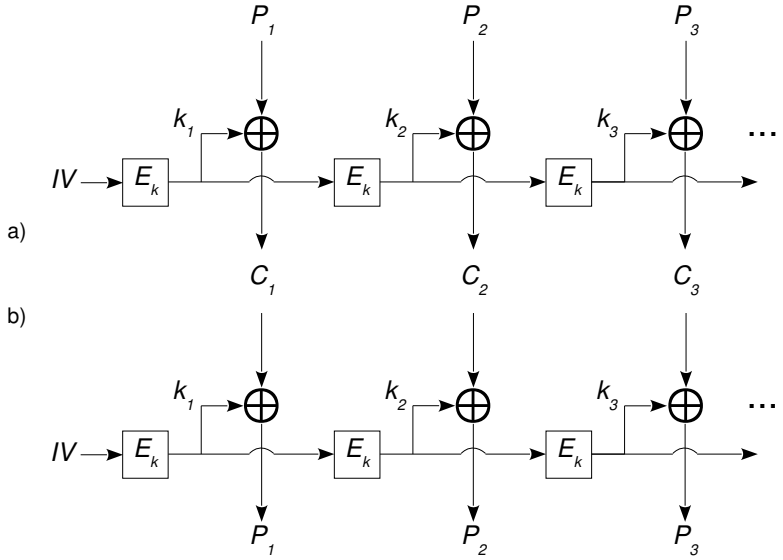


Fig. Use of "mixed" method OFB: a) enciphering; b) deciphering. (Notice the virtual keys k_i.)

“Mix” method: CBC

- *CBC, Cipher Block Chaining*
- Some properties:
  - *IV* (random) or explicit initialization by (phony) 1st block!
  - padding
  - not parallelizable enciphering; parallelizable deciphering
- Formulas:
  - $C_0 = IV$; $C_i = E_k (P_i \oplus C_{i-1})$ $i>0$
  - Write the decipherment formula.
- Error propagation:
  - exercise!

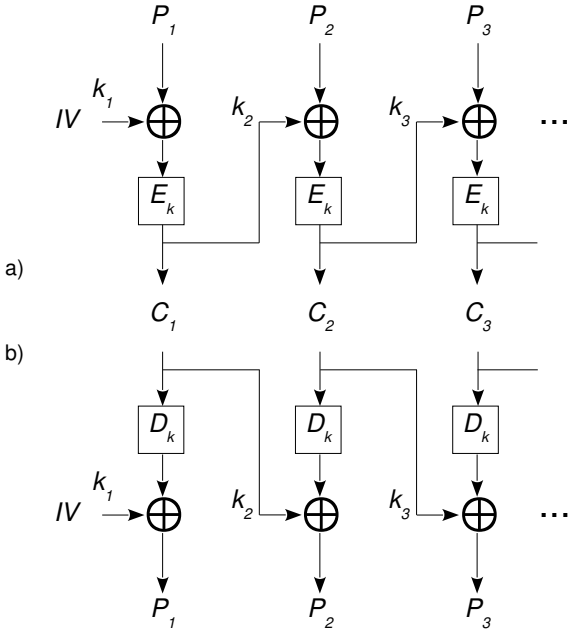


Fig. Use of “mixed” method CBC: a) enciphering; b) deciphering
(Notice the virtual keys k_i.)

Another view of some operation modes

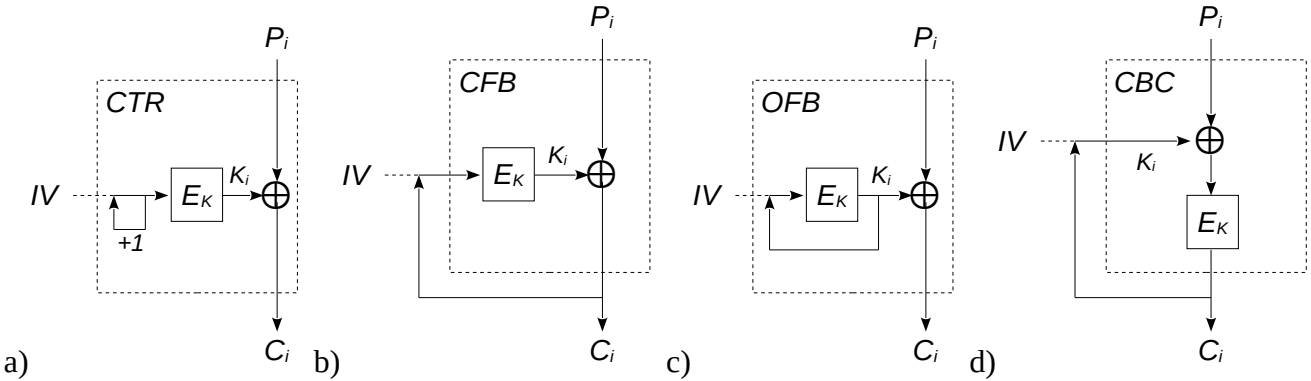


Fig. The software-view of some of the operation modes ($i>0$). In b) and c) the reason for the modes' names is apparent...

Padding

Need

- size of plaintext varies (just hardly ever is multiple of block size)
  - so, final block might need[1] padding!
  - but, "casual" padding might open an attack path (*see ahead*)!
- harden message deciphering and traffic analysis[2]
  - by obscuring the size (and content) of ciphertext
    - e.g. avoiding short messages' attack on RSA[3]
    - e.g. avoiding deterministic ciphering's attack[4]

1 Why?... Also, some "modes of operation" do not need padding... why?
2 interception and examination of communications (ciphered or not) to deduce information (e.g. from patterns)
3 when P is < n. See asecuritysite.com/encryption/crackrsa2
4 As same plaintext always produces same ciphertext, a cryptanalyst may build a collection of plaintext/ciphertext pairs and look for cipher matches in communication media; it is specially feasible with "public-key cryptography" (why?)!

Padding schemes

- several schemes (bit padding or, more usually, byte padding)
  - shared-key cryptography
    - e.g. PKCS[1] #5[2], #7[3] (enciphering) [Fig. ShKey]
  - one-way cryptography
    - e.g. RFC 6234 (SHA-1, SHA-256) [Fig. OneWay a)]
    - e.g. SHA3 (sponge) [Fig. OneWay b)]
  - public-key cryptography
    - e.g. PKCS #1 v2 (RFC 8017)
      - RSA's PKCS1-v1_5 [Fig. PKCS1]
      - RSA's OAEP, Optimal Asymmetric Encryption Padding [Fig. OAEP]
        - Exercise (after analyzing picture): what about deciphering?... does receiver need *seed* and *L*?...

1 Public Key Cryptography Standards, devised and published by RSA Security LLC since the 1990s
2 PKCS #5: Password-Based Cryptography - from a password, generate a (symmetric) key for a following symmetric encipherment.
3 #7 padding just extends 8B block #5 padding to 16B (128b) blocks

...Cipher modes of operation: padding examples (figs)...

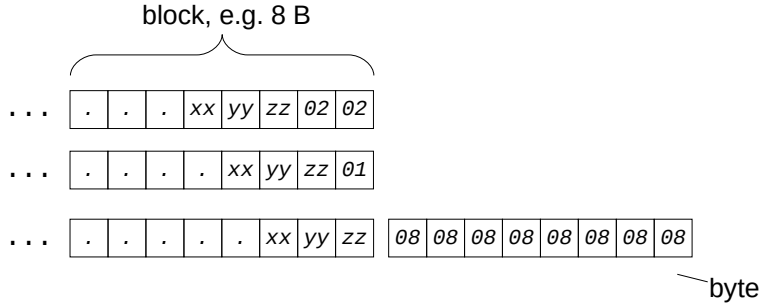


Fig. ShKey: Shared-key cryptography padding: examples for PKCS #5 (8B blocks); #7 will be similar, but appropriate to 16B blocks.
Algorithm: add (`block_size - P_length mod block_size`) bytes; all with value equal to number of added bytes: e.g. if 3 bytes are needed to complete last block, each added byte's value is 3.

...Cipher modes of operation: padding examples (figs)...

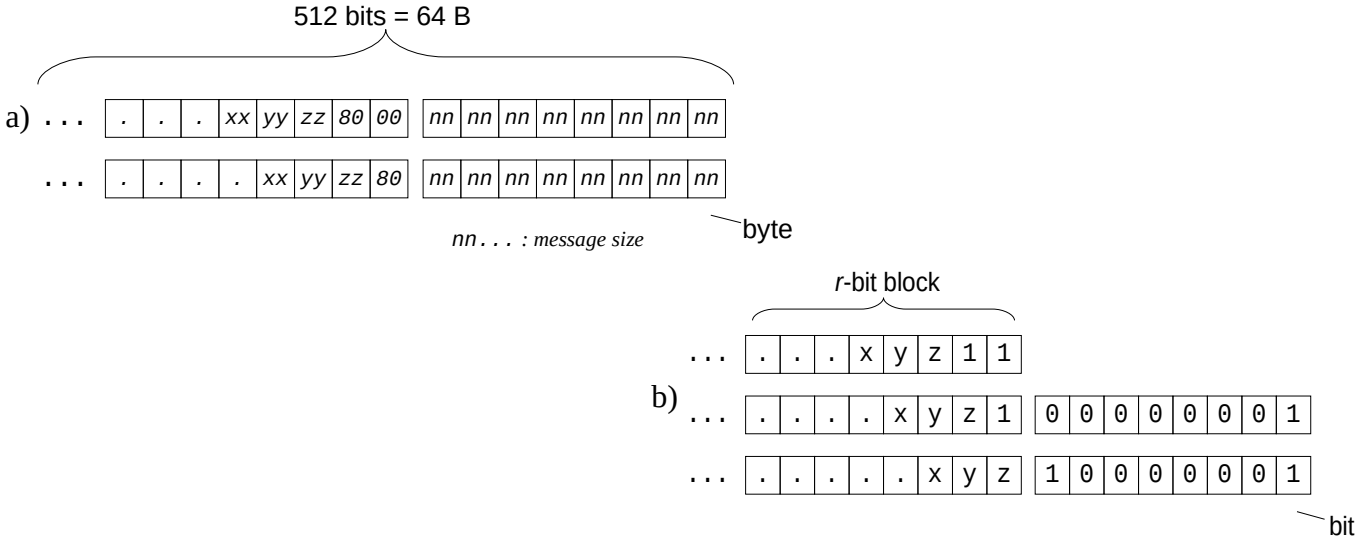


Fig. OneWay: Instances of one-way cryptography padding:
a) RFC 6234 padding: (SHA1, SHA256...) - sequence of *nn*s is message size;
b) Sponge *multirate* padding: `10*1` (*r* is the number of bits of input block).

...Cipher modes of operation: padding examples (figs)...

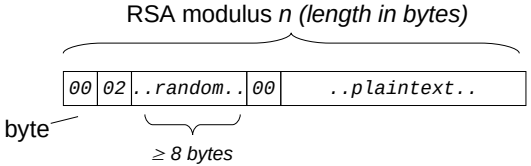


Fig. PKCS1. RSA padding: PKCS1-v1_5.

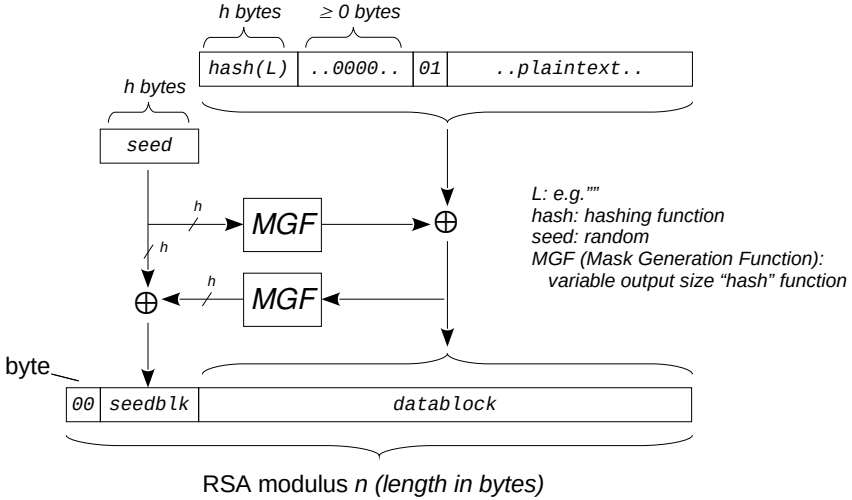


Fig. OAEP. RSA padding: OAEP, Opt*imal Asymmetric Encryption Padding.*
After padding, RSA enciphering proceeds with final data being treated as of *n*-byte hex number.

Attack examples

- length extension: one-way cryptography, MAC (if = $h(K||P)$)
  - if $hash(P1) = hash(IV, P1) = hash(hash(IV), P1)$
    $hash(P1||P2) = hash(P1, P2) = hash(hash(P1), P2)$
  - SEED Lab!
- padding oracle: two-way cryptography, CBC mode
  - if attacker can keep testing decipherment with crafted ciphertext
  - if deciphering error code says explicitly "*invalid padding*" instead of a general "*decryption failed*"
  - CBC: $P_i = D_k(C_i) \oplus C_{i-1} \quad i>0$
    - a byte/bit change in C_{i-1} affects corresponding byte/bit in P_i
    - starting from last C_i block (where padding is), keep changing last byte of previous block until padding is valid; then repeat for previous bytes
    - see [FIG] (PKCS #5, #7 padding)

...”Long” texts' encipherment: Padding...

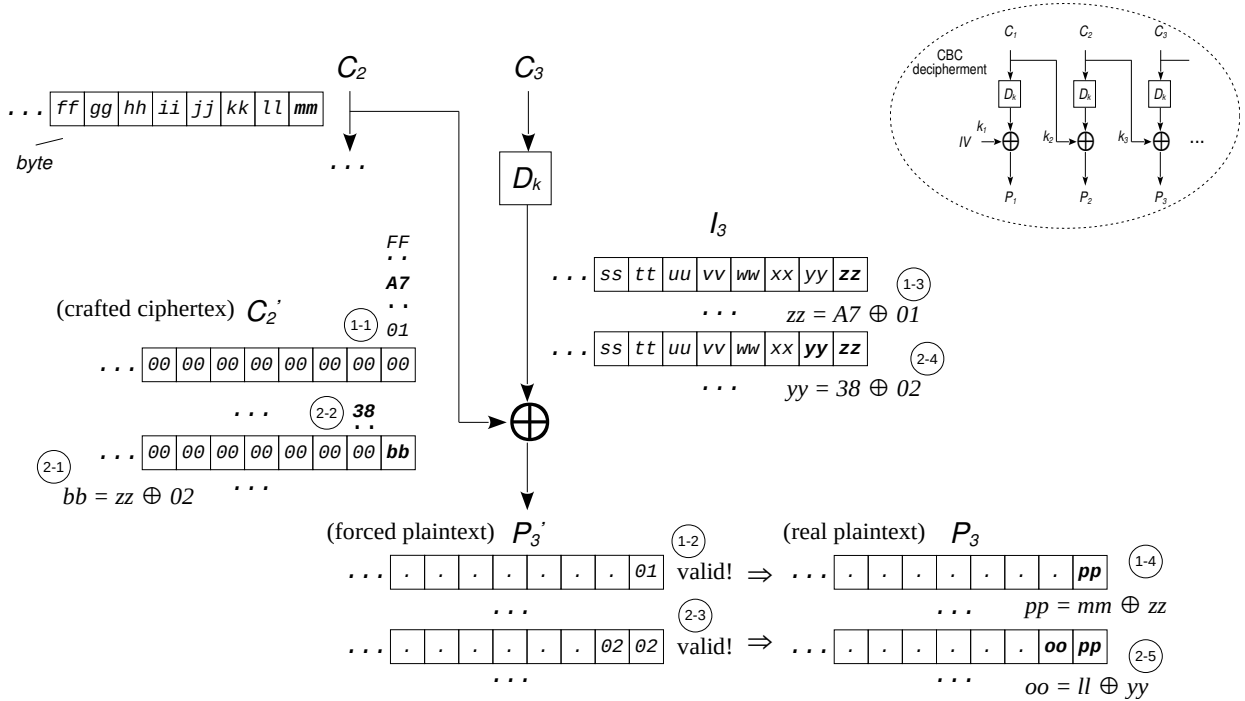


Fig. Padding oracle attack procedure for PKCS #5, #7 padding (CBC mode). C_3 is last cipher block.

Real need for padding?

- avoidance:
  - ciphertext stealing [FIG in Wikipedia]
  - residual block termination [FIG]
- will it be worth the trouble?...