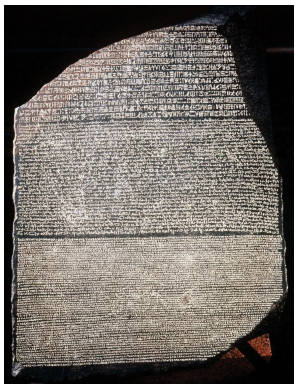
Crypto 5

Note: this material is not intended to replace the live lecture for students.

Contents



5.1 Block Ciphers: Operations Modes.

A block cipher is much more than just an encryption algorithm. It can be used as a versatile building block with which a diverse set of cryptographic mechanisms can be realized.

[Paar10, pag. 123]

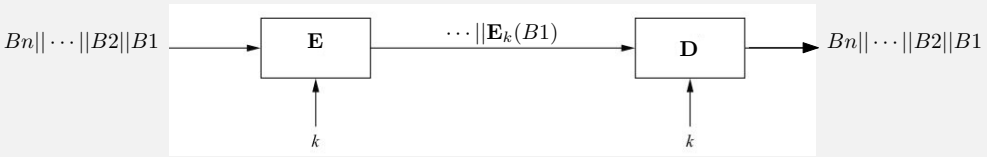
5.1.1 ECB, Electronic Codebook

The message P is divided into blocks:

$$\mathrm{P} = Bn \cdots ||B2||B1$$

and each block is encrypted separately with the same key k.

Figure 5.1.1: ECB Mode



Advantages

Synchronization is not necessary: blocks can be decipher independently of each other.

ECB encryption can be do in parallel.

Errors remains localized e.g. transmission errors.

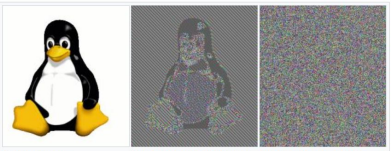
Disadvantages ☹

There are not integrity protection: the adversary can invert or substitute blocks.

If the key is not replaced it is possible to detect blocks sent twice by just watching the cipher block: Traffic analysis.

The disadvantage of this method is a lack of diffusion. Because ECB encrypts identical plaintext blocks into identical ciphertext blocks, it does not hide data patterns well. In some senses, it doesn't provide serious message confidentiality, and it is not recommended for use in cryptographic protocols at all.

A striking example of the degree to which ECB can leave plaintext data patterns in the ciphertext can be seen when ECB mode is used to encrypt a bitmap image which uses large areas of uniform color. While the color of each individual pixel is encrypted, the overall image may still be discerned as the pattern of identically colored pixels in the original remains in the encrypted version.



Original image | Encrypted using ECB mode | Modes other than ECB result in pseudo-randomness

The adjacent image is how the image might appear encrypted with CBC, CTR or any of the other more secure modes—indistinguishable from random noise. Note that the random appearance of the adjacent image does not ensure that the image has been securely encrypted; many kinds of insecure encryption have been developed which would produce output just as "random-looking".

ECB mode can also make protocols without integrity protection even more susceptible to replay attacks, since each block gets decrypted in exactly the same way.

Thus ECB is a deterministic mode which leave on the ciphertext too much information of the plaintext. It doesn't provide serious message confidentiality, and it is not recommended for use in cryptographic protocols at all.

NOTE 5.1.2

The name Electronic Codebook comes from the fact that given a key k each clear block is cipher into a unique cipher block. So we can imagine a huge book in which each line contains the pair clear block / cipher block. Such a book can be regarded as a "code".

Exercise 5.1.3

Take a look to page 9 in document NIST SP 800-38a about ECB mode.

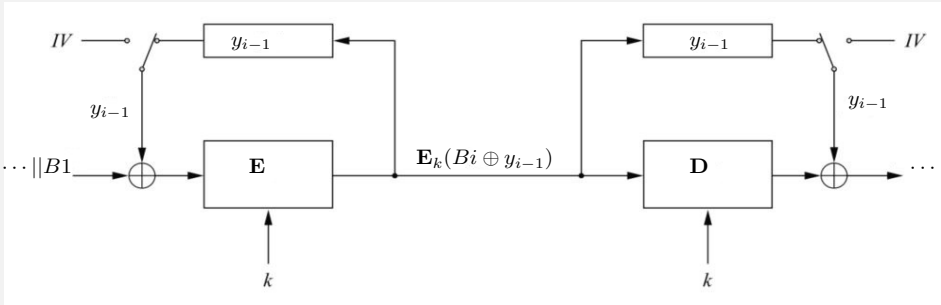
5.1.2 CBC, Cipher Block Chaining

The message P is divided into blocks:

$$\mathrm{P} = Bn \cdots ||B2||B1$$

but blocks are cipher according the following figure:

Figure 5.1.4: CBC Mode



IV is called *initializating vector* it is usually a nonce. If IV is random the CBC is not deterministic, i.e. probabilistic encryption.

5.1.5 CBC ciphering

$y_1 = \mathsf{Enc}_k(B1 \oplus IV)$
If $j > 1$ then $y_j = \mathsf{Enc}_k(Bj \oplus y_{j-1})$.

5.1.6 CBC deciphering

$B1 = \mathsf{Dec}_k(y_1) \oplus IV$
If $j > 1$ then $B_j = \mathsf{Dec}_k(y_j) \oplus y_{j-1}$.

Advantages

Allows Probabilistic Encryption.

Blocks depends upon each other as in a chain. So changing a bit from IV or from a block affects all the following cipher blocks.

Deciphering in CBC mode can be done in parallel: $Bj = \mathsf{Dec}_k(y_j) \oplus y_{j-1}$ so Bj can be recover from two consecutive enciphered blocks.

Appendix C: Generation of Initialization Vectors

The CBC, CFB, and OFB modes require an initialization vector as input, in addition to the plaintext. An IV must be generated for each execution of the encryption operation, and the same IV is necessary for the corresponding execution of the decryption operation. Therefore, the IV, or information that is sufficient to calculate the IV, must be available to each party to the communication.

The IV need not be secret, so the IV, or information sufficient to determine the IV, may be transmitted with the ciphertext.

For the CBC and CFB modes, the IVs must be unpredictable. In particular, for any given plaintext, it must not be possible to predict the IV that will be associated to the plaintext in advance of the generation of the IV.

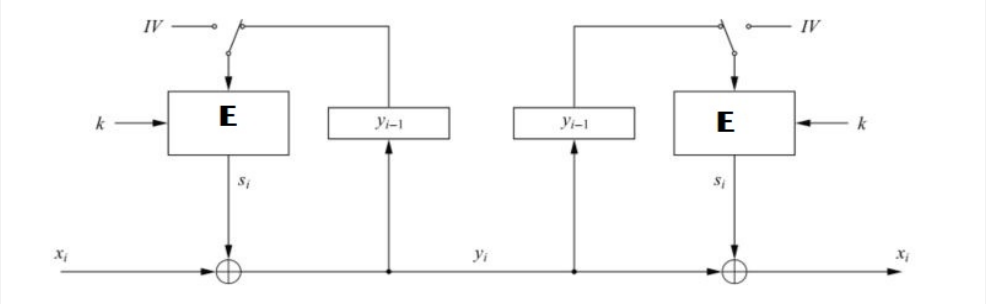
NIST 800-38a

For more about the IV in CBC mode see [Paar10, page 129].
More about IV : wikipedia IV and [KatLin15, page 95].

5.1.3 CFB, Cipher FeedBack mode (PRNG)

In this mode the block cipher is used to construct a stream cipher:

Figure 5.1.7: CFB Mode



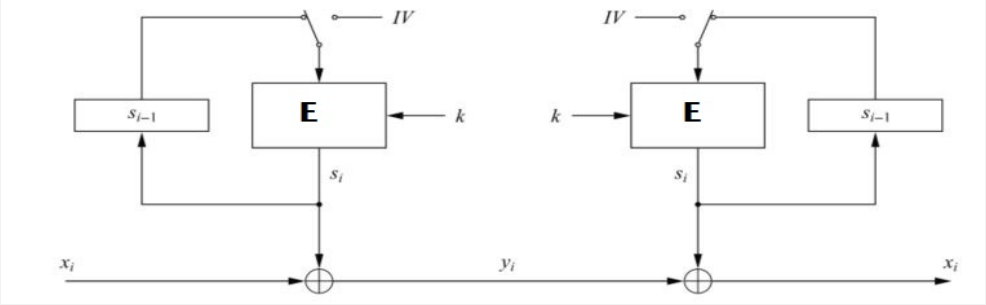
Exercise 5.1.8

Take a look to the explanation of DES in CFB mode at page 5 of the document FIPS81

5.1.4 OFB, Output FeedBack mode (PRNG)

In this mode the block cipher is used to construct a stream cipher:

Figure 5.1.9: OFB Mode



5.1.5 CTR, Counter Mode

The plaintext P is divided into N blocks $P = BN||\cdots||B1$. Then N blocks $TN, \cdots, T1$ called "counters" are created and here is how the block cipher is used;

5.1.10 CTR ciphering

$O_j = \mathsf{Enc}_k(Tj)$ for $j = 1, \cdots, N$
$C_j = Bj \oplus O_j$ for $j = 1, \cdots, N$

5.1.11 CTR deciphering

$O_j = \mathsf{Enc}_k(Tj)$ for $j = 1, \cdots, N$
$Bj = C_j \oplus O_j$ for $j = 1, \cdots, N$

Usually the counters $TN, \cdots, T1$ are obtained from the initial block $T1$ by increment or partial increment. For example, $T1 = IV||Q$, where $Q \in \mathbb{Z}_{2^m}$ and $Tj := IV||(Q+j-1)$ and IV is a nonce.

Counters can be created by a LFSR from an initial state $T1$.

NOTE 5.1.12

It is important to prevent two counters from being equal. Otherwise you risk a KPA attack: if $T1 = Tj$ and $(Ct_1, B1)$ known to the cryptanalyst then he also easily gets Bj.

To avoid two counters being the same, you should choose the size m of Q to be able to encrypt all N blocks of our message before changing the nonce IV, ie $N \leq 2^m$.

It is not necessary for the first counter $T1$ to be secret. The sender could send $T1$ together with the first encrypted block Ct_1 to the recipient.

Advantages

CRT mode is parallelizable.

5.1.6 GCM, Galois Counter Mode

Galois mode is combination of CTR mode, by using an IV, and a MAC i.e. Message Authentication Code. Namely, a **tag t** of 128 bits is produced to allow the receiver to check the authenticity of the message.

The used block cipher must have blocks of 128 bits. It is called "Galois" because blocks are regarded as numbers of the finite Galois field $\mathrm{GF}(2^{128})$. That is to say, as in our baby Galois-cipher but using the degree 128 polynomial $G(x) = x^{128} + x^7 + x^2 + x + 1$.

NOTE 5.1.13

The IVs in GCM must fulfill the following "uniqueness" requirement:

The probability that the authenticated encryption function ever will be invoked with the same IV and the same key on two (or more) distinct sets of input data shall be no greater than 2^{-32}.

Compliance with this requirement is crucial to the security of GCM. Across all instances of the authenticated encryption function with a given key, if even one IV is ever repeated, then the implementation may be vulnerable to the forgery attacks that are described in Ref [5] and summarized in Appendix A. In practice, this requirement is almost as important as the secrecy of the key.

From page 18 at https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38d.pdf
Notice that the convention to pass from blocks to polynomials is little-endian (page 12 of the above NIST document) i.e. opposed to our baby Galois-cipher.

Here is how it works. The message $\mathrm{P} = BN||\cdots||B1$ consists of blocks of 128 bits. As in CTR mode a counter is used to produce T_j and blocks are encrypted as $C_j = Bj \oplus \mathrm{O}_j$.

Here is how the tag is produced:

5.1.14 GCM tag t

$H = \mathsf{Enc}_k(0^{128})$;
$g_0 = \mathrm{AAD} \otimes H$; where $\otimes$ is the Galois multiplication in $\mathrm{GF}(2^{128})$.
$g_j = (g_{j-1} \oplus \mathrm{C}_j) \otimes H$ for $j = 1, \cdots, N$;
$\mathrm{L} = [\mathrm{len}(\mathrm{AAD})]_{64}||[\mathrm{len}(\mathrm{P})]_{64}$;
$\mathbf{t} = ((g_N \oplus \mathrm{L}) \otimes H) \oplus \mathsf{Enc}_k(T0)$.

The receiver gets

$$(\mathrm{C}_N, \cdots, \mathrm{C}_1, \mathbf{t}, \mathrm{AAD}) .$$

from the sender where AAD stands for **Additional Authentication Data**.

NOTE 5.1.15

From the mathematical point of view the tag $\mathbf{t}$ is basically the evaluation of H in a polynomial whose coefficients are the blocks of the ciphertext. Namely,

$$\mathbf{t} = C_1 H^{N+1} + C_2 H^N + \cdots + C_{N-1} H^3 + C_N H^2 + g_0 H^N + LH + \mathsf{Enc}_{\mathsf{k}}(T0)$$

The receiver decrypt as in CRT mode and by using the AAD a tag $\mathbf{t}'$ is computed via the above algorithm 5.1.14. If $\mathbf{t} = \mathbf{t}'$ then the receiver is assured that the ciphertext (and AAD) were not manipulated in transit and that only the sender could have generated the message.

In practice, the string AAD might include IP addresses and parameters in a network protocol. So the tag is used for the authentication of the AAD.

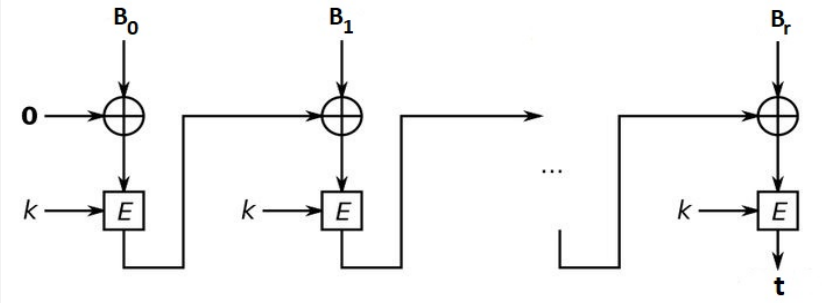
5.1.7 CCM, Counter CBC MAC

Also this mode is combination of previous CBC and CTR modes and produces a **tag t** to allow the receiver to check the integrity of the message.

This modes uses a nonce N. The message M and the nonce are encoded into blocks $B_0B_1\cdots B_r$. Namely, the blocks $B_0B_1\cdots B_r$ are constructed by using the message blocks in M and the nonce following a precise algorithm, for details see "2.2. Authentication" in https://tools.ietf.org/html/rfc3610

Then the tag **t** is produced as follows:

Figure 5.1.16: CCM tag



Counters T_j are created from the nonce N to encrypt the tag and the message (CRT mode).

NOTE 5.1.17

The tag constructed as in Figure 5.1.16 by using just the blocks of the message M is known as CBC MAC.

The nonce being non-repeating implies that all CTR counters and all CBC-MAC used during the life time of a key are distinct.

5.2 Attacks

5.2.1 to ECB: Block Replay

Here I follow two examples in [Schneier15, page 191] Cf. https://en.wikipedia.org/wiki/Replay_attack

Figure 5.2.1: Block Replay, version 1

To illustrate the problem, consider a money transfer system that moves money between accounts in different banks. To make life easier for the bank's computer systems, banks agree on a standard message format for money transfer that looks like this:

Bank One: Sending	1.5 blocks
Bank Two: Receiving	1.5 blocks
Depositor's Name	6 blocks
Depositor's Account	2 blocks
Amount of Deposit	1 block

Figure 5.2.2: Block Replay, version 2

At first glance, the banks could easily prevent this by adding a timestamp to their messages.

Date/Time Stamp:	1 block
Bank One: Sending	1.5 blocks
Bank Two: Receiving	1.5 blocks
Depositor's Name	6 blocks
Depositor's Account	2 blocks
Amount of Deposit	1 block

5.2.2 Why MACs and tags?

Block replay attack under CBC mode can cause serious problems e.g. the amount can be a random redirected to a random account which is obviously highly undesirable for banks...

Thus MACs and tags are produced to avoid such manipulations and protect the integrity of the message.

5.3 Appendixes:

5.3.1 Isomorphism of Finite Galois Field: an explicit example.

Our first week we construct $GF(2^5)$ by using as the reduction modulus the polynomial $G(x) = x^5 + x^2 + 1$. If $GF(2^5)$ is constructed by using another reduction modulus polynomial $H(x)$ then there is an isomorphism between both constructions. Such isomorphism can be explicitelly written by using a matrix ρ. Below an example of the isomorphism ρ when $H(x) = x^5 + x^4 + x^2 + 1$. To avoid confusions we are going to use the letter u for the variable of $G(u)$ and v for the variable of $H(v)$. Namely, $G(u) = u^5 + u^2 + 1$ and $H(v) = v^5 + v^4 + v^2 + 1$. Then the map ρ works as follows:

$$\rho(x_4u^4 + x_3u^3 + x_2u^2 + x_1u + x_0) = y_4v^4 + y_3v^3 + y_2v^2 + y_1v + y_0$$

where $[y_4y_3y_2y_1y_0]$ is going to be computed using a 5×5 matrix (also denoted by ρ):

$$[x_4x_3x_2x_1x_0] \cdot \rho = [y_4y_3y_2y_1y_0]\,.$$

Here is how to construct the matrix ρ.

First step. Find $\alpha(v)$ such that $G(\alpha(v)) = 0 \pmod{H(v)}$. It is a general fact (a Theorem of Galois Theory) that such polynomial $\alpha(v)$ do exist.
In our case $\alpha(v) = v + 1$. Why we need such $\alpha(v)$? We need such $\alpha(v)$ because $\rho(u) = \alpha(v)$. Indeed since ρ is going to be an isomorphism then

$$G(v) = 0 \pmod{G(v)} \Rightarrow \rho(G(v)) = G(\rho(v)) = 0 \pmod{G(v)}$$

Second step. Construction of the rows of the matrix ρ.
Observe that product $[00001] \cdot \rho$ is the last row of ρ. But such product must be [00001] since it represents the neutral element, the one 1, of the multiplication of the field $GF(2^5)$. So this gives the last row of the matrix ρ.
Now the product $[00010] \cdot \rho$ represents both $\rho(u)$ and the 4th row of ρ. But we pick $\rho(u) = \alpha(v)$ so $[00010] \cdot \rho = [00011]$. Thus we have

$$\rho = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

To get the third row of the matrix, namely $[00100] \cdot \rho$ we need to compute $\alpha(v)^2$ which equals $v^2 + 1$. So we get

$$\rho = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Now it is clear that for the first and second row we need to compute $\alpha(v)^4$ and $\alpha(v)^3$ respectivaly. Here is the matrix :

$$\rho = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

5.3.2 AEAD ciphers

wiki AEAD

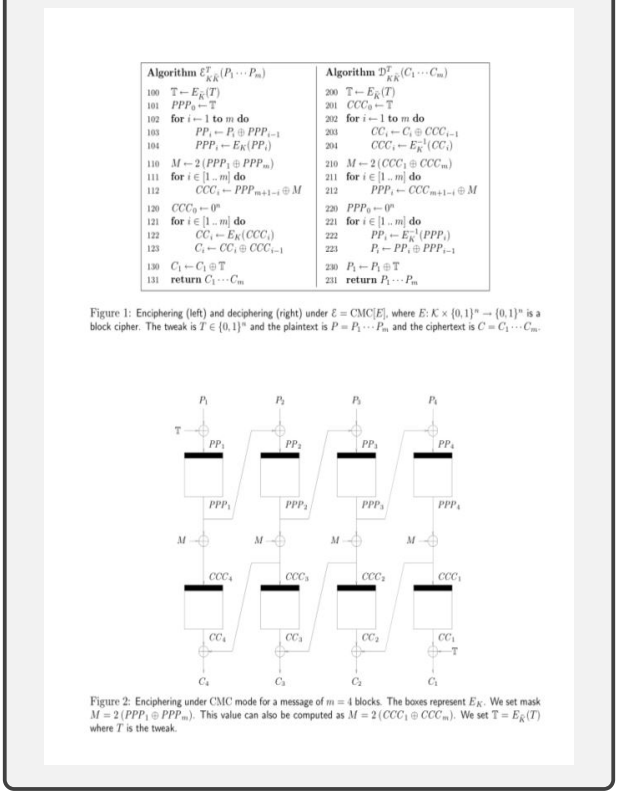
5.3.3 CMC & Tweaks

CMC stands for CBC-Mask-CBC https://en.wikipedia.org/wiki/Disk_encryption_theory

> ENCIPHERING SCHEMES. Suppose you want to encrypt the contents of a disk, but the encryption is to be performed by a low-level device, such as a disk controller, that knows nothing of higher-level concepts like files and directories. The disk is partitioned into fixed-length sectors and the encrypting device is given one sector at a time, in arbitrary order, to encrypt or decrypt. The device needs to operate on sectors as they arrive, independently of the rest. Each ciphertext must have the same length as its plaintext, typically 512 bytes. When the plaintext disk sector P is put to the disk media at location T what is stored on the media should be a ciphertext $C = \mathcal{E}_K^T(P)$ that depends not only on the plaintext P and the key K, but also on the location T, which we call the *tweak*. Including the dependency on T allows that identical plaintext sectors stored at different places on the disk will have computationally unrelated ciphertexts.

[HaRo03, page 3]

Figure 5.3.1: CMC mode



Algorithm $\mathcal{E}_{K,K}^T(P_1 \cdots P_m)$

100 $\mathbb{T} \leftarrow E_{\hat{K}}(T)$
101 $PPP_0 \leftarrow \mathbb{T}$
102 **for** $i \leftarrow 1$ **to** m **do**
103 $\quad PP_i \leftarrow P_i \oplus PPP_{i-1}$
104 $\quad PPP_i \leftarrow E_K(PP_i)$
110 $M \leftarrow 2\,(PPP_1 \oplus PPP_m)$
111 **for** $i \in [1\,..\,m]$ **do**
112 $\quad CCC_i \leftarrow PPP_{m+1-i} \oplus M$
120 $CCC_0 \leftarrow 0^n$
121 **for** $i \in [1\,..\,m]$ **do**
122 $\quad CC_i \leftarrow E_K(CCC_i)$
123 $\quad C_i \leftarrow CC_i \oplus CCC_{i-1}$
130 $C_1 \leftarrow C_1 \oplus \mathbb{T}$
131 **return** $C_1 \cdots C_m$

Algorithm $\mathcal{D}_{K,K}^T(C_1 \cdots C_m)$

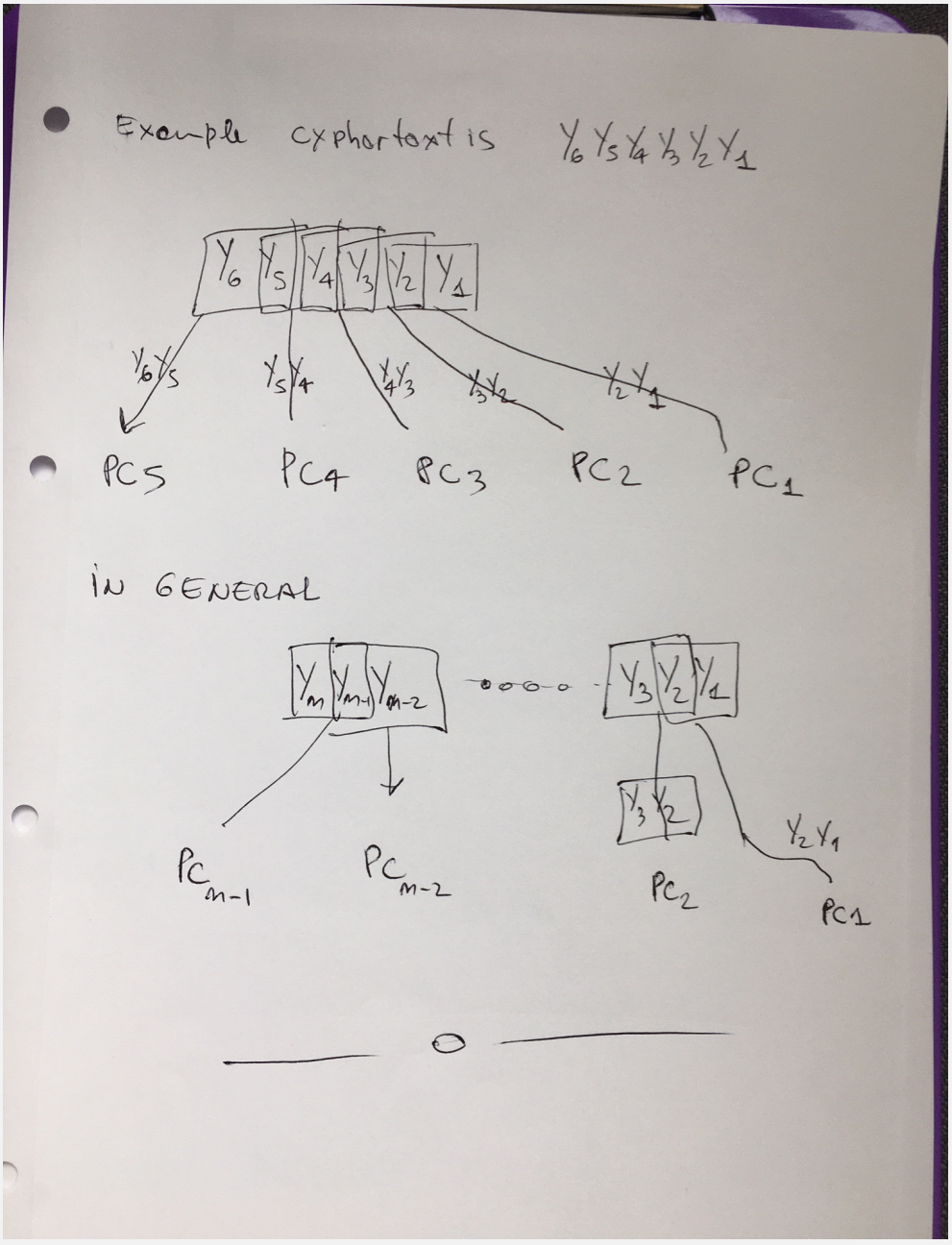
200 $\mathbb{T} \leftarrow E_{\hat{K}}(T)$
201 $CCC_0 \leftarrow \mathbb{T}$
202 **for** $i \leftarrow 1$ **to** m **do**
203 $\quad CC_i \leftarrow C_i \oplus CCC_{i-1}$
204 $\quad CCC_i \leftarrow E_K^{-1}(CC_i)$
210 $M \leftarrow 2\,(CCC_1 \oplus CCC_m)$
211 **for** $i \in [1\,..\,m]$ **do**
212 $\quad PPP_i \leftarrow CCC_{m+1-i} \oplus M$
220 $PPP_0 \leftarrow 0^n$
221 **for** $i \in [1\,..\,m]$ **do**
222 $\quad PP_i \leftarrow E_K^{-1}(PPP_i)$
223 $\quad P_i \leftarrow PP_i \oplus PPP_{i-1}$
230 $P_1 \leftarrow P_1 \oplus \mathbb{T}$
231 **return** $P_1 \cdots P_m$

Figure 1: Enciphering (left) and deciphering (right) under $\mathcal{E} = \mathrm{CMC}[E]$, where $E\colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ is a block cipher. The tweak is $T \in \{0,1\}^n$ and the plaintext is $P = P_1 \cdots P_m$ and the ciphertext is $C = C_1 \cdots C_m$.

Figure 2: Enciphering under CMC mode for a message of $m = 4$ blocks. The boxes represent E_K. We set mask $M = 2\,(PPP_1 \oplus PPP_m)$. This value can also be computed as $M = 2\,(CCC_1 \oplus CCC_m)$. We set $\mathbb{T} = E_{\hat{K}}(T)$ where T is the tweak.

5.3.4 doing CBC in parallel

5.3.2 Here a better explanation of how CBC can be done in parallel



5.4 Bibliography

Books I used to prepare this note:

[Aumasson18] Jean-Philippe Aumasson, *Serious Cryptography: A Practical Introduction to Modern Encryption*, No Starch Press, 2018.

[KatLin15] Jonathan Katz; Yehuda Lindell, *Introduction to Modern Cryptography* Second Edition, Chapman & Hall/CRC, Taylor & Francis Group, 2015.

[Paar10] Paar, Christof, Pelzl, Jan, *Understanding Cryptography, A Textbook for Students and Practitioners*, Springer-Verlag, 2010.

[Schneier15] Bruce Schneier, *Applied Cryptography: Protocols, Algorithms and Source Code in C* ,Wiley; 20th Anniversary edition,2015.

Ecco l'elenco degli articoli

[HaRo03] Shai, Halevi and Rogaway, Phillip; *A Tweakable Enciphering Mode*, Advancesin Cryptology—CRYPTO'03, Lecture Notes in Computer Science,vol.2729, D.Boneh,ed.,Springer-Verlag,2003 https://eprint.iacr.org/2003/148.pdf

[DH76] Diffie, W.; Hellman, M. *New directions in cryptography*, (1976). IEEE Transactions on Information Theory. 22 (6): 644654.

and some interesting links:

more details about CCM.

Operation Modes