Addenda for the Cryptography course

Alessandro Languasco & Alessandro Zaccagnini

December 19, 2020

Commenti, critiche, passaggi oscuri, errori di stampa possono essere segnalati agli indirizzi qui sotto.

Prof. Alessandro Languasco
Indirizzo Dipartimento di Matematica,
Torre Archimede,
via Trieste 63, 35121 Padova
e-mail alessandro.languasco@unipd.it

Prof. Alessandro Zaccagnini
Indirizzo Dipartimento di Matematica e Informatica,
Parco Area delle Scienze, 53/a
Campus Universitario, 43124 Parma
e-mail alessandro.zaccagnini@unipr.it

Il testo è stato composto per mezzo di LaTeX 2ε, © American Mathematical Society.

Contents

1 Addenda **5**

1.1 Caesar method . . . 5
1.2 E.A. Poe example . . . 5
1.3 J. Verne example . . . 9
1.4 Strong pseudoprimes . . . 11
1.4.1 On Carmichael numbers (yet) . . . 11
1.4.2 Factoring RSA-challenge's numbers . . . 11
1.5 DES (Data Encryption Standard) . . . 12
1.5.1 DES key . . . 12
1.5.2 DES Encryption . . . 12
1.5.3 Permutation π . . . 13
1.5.4 Round keys . . . 13
1.5.5 Functions E, S e P . . . 13
1.5.6 DES deciphering algorithm . . . 16
1.5.7 Triple DES (TDES) . . . 16
1.6 AES (Rijndael) . . . 17
1.6.1 High level description . . . 17
1.6.2 Function S . . . 17
1.6.3 State s . . . 19
1.6.4 Function SR . . . 19
1.6.5 Function MC . . . 19
1.6.6 Function E: round keys definition . . . 19
1.6.7 AES deciphering process . . . 20

Bibliografia **20**

Chapter 1

Addenda for the Cryptography course

1.1 Caesar method

Esempio 1.1.1. *Caesar method is a* substitution *with an alphabet of N letters and a translation of* $1 < l \leq N$ *positions. The method runs as follows:*

(1) find the number $m \pmod N$ *that corresponds to a letter of the message;* $m \in \{0,1,2,...,N-1\}$*;*

(2) the enciphering *transformation is obtained by adding l to* $m \pmod N$*; i.e.* $m \rightarrow c = m+l \pmod N$

(3) the deciphering *transformation is obtained by subtracting l to* $c \pmod N$*; i.e.* $c \rightarrow m = c-l \pmod N$

If our message is the word "CIAO" and $l = 3$ *we get:*
CIAO $\rightarrow$ *2 8 0 12* $\rightarrow$ *5 11 3 15* $\rightarrow$ *FNCR* $\rightarrow$ *...* $\rightarrow$ *CIAO*

1.2 E.A. Poe example

Esempio 1.2.1. *(IN ITALIAN) Riportiamo qui un famoso esempio di crittogramma tratto dal racconto "Lo scarabeo d'oro" di E.A. Poe. In esso è descritta la ricerca di un tesoro a partire da un pezzo di pergamena che contiene il disegno di un capretto e una serie di caratteri, come quelli qui riprodotti.*

5	3	‡	‡	†	3	0	5	)	)	6	*	;	4	8	2	6	)	4	‡
.	)	4	‡	)	;	8	0	6	*	;	4	8	†	8	¶	6	0	)	)
8	5	;	1	‡	(	;	:	‡	*	8	†	8	3	(	8	8	)	5	*
†	;	4	6	(	;	8	8	*	9	6	*	?	;	8	)	*	‡	(	;
4	8	5	)	;	5	*	†	2	:	*	‡	(	;	4	9	5	6	*	2
(	5	*	–	4	)	8	¶	8	*	;	4	0	6	9	2	8	5	)	;
)	6	†	8	)	4	‡	‡	;	1	(	‡	9	;	4	8	0	8	1	;
8	:	8	‡	1	;	4	8	†	8	5	;	4	)	4	8	5	†	5	2
8	8	0	6	*	8	1	(	‡	9	;	4	8	;	(	8	8	;	4	(
‡	?	3	4	;	4	8	)	4	‡	;	1	6	1	;	:	1	8	8	;
‡	?	;																	

Nel vedere questo schema, ci sono di solito due reazioni: la prima, che possiamo descrivere con le parole di Amleto

> HAMLET*: "O dear Ophelia, I am ill at these numbers"*
> *William Shakespeare, "Hamlet," II, 2; 119*

si può definire come rassegnazione di fronte a un problema apparentemente insolubile, mentre la seconda, per la quale prendiamo in prestito le parole di Alice,

> ALICE*: "Somehow, it seems to fill my head with ideas—only I don't know exactly what they are!"*
> *Lewis Carroll, "Through the Looking Glass"*

si può definire possibilista. Come possiamo aiutare Amleto ed Alice a scoprire il significato dei simboli del crittogramma? Seguiremo abbastanza da vicino la descrizione che l'Autore stesso fa della decifrazione.

La prima cosa da fare è un'ipotesi sulla lingua del testo del crittogramma. L'Autore spiega che il manoscritto è stato rinvenuto in una zona un tempo infestata da pirati, e che uno dei piú famosi pirata si chiamava Kidd: poiché in inglese "kid" vuol dire capretto, in mancanza di altre informazioni la prima ipotesi sulla lingua del crittogramma è che questa sia l'inglese.

L'analisi di frequenza

Il secondo passo da fare si chiama analisi di frequenza, ed in effetti prescinde dalla conoscenza della lingua in cui è scritto il testo originale: contiamo quante volte compaiono i singoli caratteri.

8	33	*	13	1	8	3	4
;	26	5	12	0	6	?	3
4	19	6	11	2	5	¶	2
‡	16	(	10	9	5	.	1
)	16	†	8	:	4	–	1

È un'osservazione piuttosto banale che in ogni lingua alcune lettere sono molto piú frequenti di altre: in particolare, le vocali tendono a comparire piú spesso delle consonanti. Se la nostra congettura sulla lingua del testo originale è corretta, i dati in questa tabella suggeriscono che il simbolo "8" rappresenti con ogni probabilità la lettera "e" dato che in un normale testo inglese questa lettera da sola compare circa il 13% delle volte. Quando usiamo la parola "normale" intendiamo dire che il testo non *è stato costruito con l'intento di distorcere appositamente la normale frequenza delle lettere.*

Il fatto che il simbolo piú frequente "8" compaia 33 volte su 203 caratteri (con una frequenza relativa del 16% circa) suggerisce anche che lo spazio *fra le parole è stato probabilmente eliminato, perché in caso contrario sarebbe il carattere di gran lunga piú frequente.*

Non potendo sapere dove iniziano e finiscono le parole (cosa che darebbe una miniera di informazioni in lingue come l'italiano nelle quali le lettere finali sono quasi esclusivamente vocali) e neppure quali sono le sillabe piú frequenti, facciamo un altro passo di analisi statistica, basato sull'osservazione che il comportamento delle vocali e delle consonanti è radicalmente diverso: infatti, le vocali hanno la proprietà di poter precedere o seguire qualunque altra lettera dell'alfabeto (con qualche eccezione relativamente infrequente, come nel caso della consonante "q" che è sempre seguita dalla vocale "u" o dalla "q" stessa), mentre le consonanti tendono a precedere o seguire un numero ristretto di altre lettere. In altre parole, la maggior parte di combinazioni consonante-consonante dà luogo a sequenze impronunciabili, mentre ciò non è vero per le combinazioni di vocali e consonanti.

Facciamo dunque una nuova analisi di frequenza, contando questa volta il numero delle occorrenze dei digrafi, *cioè delle coppie di lettere adiacenti: nella tabella seguente abbiamo raccolto i risultati di questo conteggio, trascurando naturalmente i digrafi meno frequenti, cioè quelli che compaiono meno di 4 volte.*

;	4	12	8	5	5	†	8	4
4	8	8	8	8	5	(	;	4
6	*	5	4	‡	4	8	)	4
)	4	5	;	8	4			

Il risultato conferma la nostra ipotesi a proposito del simbolo "8" che compare in compagnia di molti simboli diversi, precedendoli o seguendoli, e che spesso è raddoppiato: come si sa, il dittongo "ee" è piuttosto frequente nella lingua inglese.

Prima di sostituire il simbolo "8" con la lettera "e" spingiamo la nostra analisi statistica un passo avanti, esaminando anche i trigrafi, *cioè terne di lettere adiacenti. Compiliamo dunque la tabella che segue, anche in questo caso ignorando i trigrafi meno frequenti.*

;	4	8	7	*	;	4	3
)	4	‡	4	8	†	8	3
				‡	(	;	3

Che cosa possiamo "dedurre" da questa tabella? Considerando il fatto che il simbolo "8" rappresenta probabilmente la lettera "e" e che una delle parole piú frequenti della lingua inglese è l'articolo determinativo "the," è piuttosto ragionevole supporre che il trigrafo piú frequente rappresenti per l'appunto proprio questa combinazione di lettere. Non si tratta di una vera e propria deduzione in senso matematico, ma di una ragionevole congettura. Siamo dunque pronti per la

Prima congettura: "; 48" = "the"

Useremo la convenzione di scrivere in nero i simboli di cui non abbiamo ancora stabilito il valore, in blu le lettere sostituite ai simboli già determinati, ed in rosso le lettere su cui si concentra di volta in volta l'analisi di Poe. Il crittogramma originale diventa:

5	3	‡	‡	†	3	0	5	)	)	6	*	t	h	e	2	6	)	h	‡
.	)	h	‡	)	t	e	0	6	*	t	h	e	†	e	¶	6	0	)	)
e	5	t	1	‡	(	t	:	‡	*	e	†	e	3	(	e	e	)	5	*
†	t	h	6	(	t	e	e	*	9	6	*	?	t	e	)	*	‡	(	t
h	e	5	)	t	5	*	†	2	:	*	‡	(	t	h	9	5	6	*	2
(	5	*	–	h	)	e	¶	e	*	t	h	0	6	9	2	e	5	)	t
)	6	†	e	)	h	‡	‡	t	1	(	‡	9	t	h	e	0	e	1	t
e	:	e	‡	1	t	h	e	†	e	5	t	h	)	h	e	5	†	5	2
e	e	0	6	*	e	1	(	‡	9	t	h	e	t	(	e	e	t	h	(
‡	?	3	h	t	h	e	)	h	‡	t	1	6	1	t	:	1	e	e	t
‡	?	t																	

Il narratore del racconto di Poe suggerisce di guardare la sequenza di lettere indicata in rosso: vediamo l'articolo "the" seguito da "t(eeth." L'ipotesi è che il simbolo "(" rappresenti la lettera "r" in modo che questa sequenza indichi la parola "tree" (che vuol dire albero) e sia poi seguita dall'inizio di un'altra parola. Se questa ipotesi, che è ragionevole anche in base alla frequenza relativa del simbolo corrispondente, è vicina alla verità, sostituendo la lettera "r" dovrebbero comparire altri frammenti di parole inglesi plausibili. Passiamo quindi alla

Seconda congettura: "(" = "r"

Questo è il risultato della sostituzione indicata:

5	3	‡	‡	†	3	0	5	)	)	6	*	t	h	e	2	6	)	h	‡
.	)	h	‡	)	t	e	0	6	*	t	h	e	†	e	¶	6	0	)	)
e	5	t	1	‡	r	t	:	‡	*	e	†	e	3	r	e	e	)	5	*
†	t	h	6	r	t	e	e	*	9	6	*	?	t	e	)	*	‡	r	t
h	e	5	)	t	5	*	†	2	:	*	‡	r	t	h	9	5	6	*	2
r	5	*	–	h	)	e	¶	e	*	t	h	0	6	9	2	e	5	)	t
)	6	†	e	)	h	‡	‡	t	1	r	‡	9	t	h	e	0	e	1	t
e	:	e	‡	1	t	h	e	†	e	5	t	h	)	h	e	5	†	5	2
e	e	0	6	*	e	1	r	‡	9	t	h	e	t	r	e	e	t	h	r
‡	?	3	h	t	h	e	)	h	‡	t	1	6	1	t	:	1	e	e	t
‡	?	t																	

Si può forse essere d'accordo con un altro dei personaggi di "Amleto":

> POLONIUS: *"Though this be madness,*
> *yet there is method in't"*
> *William Shakespeare, "Hamlet," II, 2, 205–206*

Guardiamo ora il frammento di testo indicato in rosso: il narratore suggerisce che si tratti della parola "through" (attraverso), seguita dall'articolo determinativo "the." Consultando la tabella delle frequenze dei vari simboli, scopriamo che "‡" compare ben 16 volte, mentre "?" e "3" compaiono rispettivamente 3 e 4 volte ciascuno. Questo è incoraggiante, perché effettivamente la vocale "o" è piuttosto frequente in inglese, mentre "u" e "g" sono lettere relativamente infrequenti. Siamo pronti per la nostra

Terza congettura: “‡” = “o” “?” = “u” “3” = “g”

Come sempre, riportiamo il risultato della sostituzione dei simboli a cui abbiamo assegnato un, seppur ipotetico, valore.

5	g	o	o	†	g	0	5	)	)	6	*	t	h	e	2	6	)	h	o
.	)	h	o	)	t	e	0	6	*	t	h	e	†	e	¶	6	0	)	)
e	5	t	1	o	r	t	:	o	*	e	†	e	g	r	e	e	)	5	*
†	t	h	6	r	t	e	e	*	9	6	*	u	t	e	)	*	o	r	t
h	e	5	)	t	5	*	†	2	:	*	o	r	t	h	9	5	6	*	2
r	5	*	–	h	)	e	¶	e	*	t	h	0	6	9	2	e	5	)	t
)	6	†	e	)	h	o	o	t	1	r	o	9	t	h	e	0	e	1	t
e	:	e	o	1	t	h	e	†	e	5	t	h	)	h	e	5	†	5	2
e	e	0	6	*	e	1	r	o	9	t	h	e	t	r	e	e	t	h	r
o	u	g	h	t	h	e	)	h	o	t	1	6	1	t	:	1	e	e	t
o	u	t																	

Per non tediare inutilmente i Lettori, condensiamo gli ultimi passi dell’analisi di Poe: per prima cosa concentriamoci sul primo frammento in rosso qui sopra. Compare quasi per intero la parola “degree” (grado) che possiamo considerare plausibile se la pergamena contiene davvero le indicazioni per trovare un tesoro. La seconda parola in rosso sembra essere “thirteen” (tredici): considerando che il simbolo “6” compare ben 11 volte e che dopo una parola come “grado” è naturale aspettarsi qualche dato di tipo numerico, ci sentiamo fiduciosi nel fare le congetture seguenti.

Quarta congettura: “†” = “d” “6” = “i” “∗” = “n”

Continuando allo stesso modo, è possibile determinare il valore dei pochi simboli ancora sconosciuti: di seguito diamo la corrispondenza fra caratteri del testo cifrato e quelli del testo in chiaro.

8	e	*	n	1	f	3	g
;	t	5	a	0	l	?	u
4	h	6	i	2	b	¶	v
‡	o	(	r	9	m	.	p
)	s	†	d	:	y	–	c

Non resta che sostituire gli ultimi simboli con le lettere corrispondenti per ottenere il testo decifrato, o, per usare il gergo dei crittografi, “in chiaro.”

Il messaggio in chiaro

a	g	o	o	d	g	l	a	s	s	i	n	t	h	e	b	i	s	h	o
p	s	h	o	s	t	e	l	i	n	t	h	e	d	e	v	i	l	s	s
e	a	t	f	o	r	t	y	o	n	e	d	e	g	r	e	e	s	a	n
d	t	h	i	r	t	e	e	n	m	i	n	u	t	e	s	n	o	r	t
h	e	a	s	t	a	n	d	b	y	n	o	r	t	h	m	a	i	n	b
r	a	n	c	h	s	e	v	e	n	t	h	l	i	m	b	e	a	s	t
s	i	d	e	s	h	o	o	t	f	r	o	m	t	h	e	l	e	f	t
e	y	e	o	f	t	h	e	d	e	a	t	h	s	h	e	a	d	a	b
e	e	l	i	n	e	f	r	o	m	t	h	e	t	r	e	e	t	h	r
o	u	g	h	t	h	e	s	h	o	t	f	i	f	t	y	f	e	e	t
o	u	t																	

Per comodità dei Lettori, riportiamo il messaggio decifrato dotandolo degli spazi e dei normali segni di interpunzione:

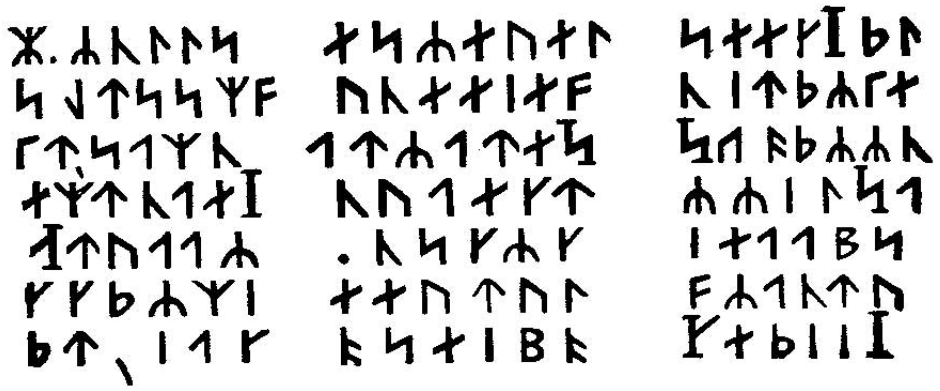

Figure 1.1 - The original Verne's cryptogram in Runic.

> *A good glass in the bishop's hostel in the devil's seat — forty-one degrees and thirteen minutes — northeast and by north — main branch seventh limb east side — shoot from the left eye of the death's-head — a bee-line from the tree through the shot fifty feet out.*

La sua traduzione in italiano è la seguente:

> *Un buon vetro nell'ostello del vescovo sulla sedia del diavolo — quarantun gradi e tredici minuti — nord-nordest — tronco principale settimo ramo lato est — cala dall'occhio sinistro del teschio — una linea retta dall'albero passando per il punto toccato lontano cinquanta piedi.*

Come si vede, si tratta della descrizione di una località dalla quale è possibile scorgere un albero, su un ramo del quale c'è un teschio. Muovendosi di cinquanta piedi dall'albero nella direzione del punto che si trova sulla verticale dell'occhio sinistro del teschio, si troverà il luogo dove è stato seppellito il tesoro. Si noti che "vetro" è un'espressione gergale per cannocchiale.
C'è un aspetto poco verosimile nella descrizione, peraltro molto accurata, di Edgar Allan Poe, e cioè che il processo di decifrazione sia unidirezionale, un passo dietro l'altro, sempre piú vicini alla soluzione corretta. In realtà è quasi certo che si commettano numerosi errori, che si finisca in vicoli ciechi, che si debba tornare sui propri passi, come in un labirinto: il filo di Arianna che guida il crittografo è rappresentato in buona parte dal suo intuito e dalla sua conoscenza delle caratteristiche della lingua in cui si suppone sia scritto il testo da decifrare, ma si basa su una solida analisi statistica del testo. Viceversa, un aspetto assolutamente verosimile riguarda la velocità con cui si decifra il messaggio una volta raggiunta una certa "massa critica" di caratteri identificati correttamente: i primi passi sono molto incerti e dubbi, mentre i passi successivi sono sempre piú sicuri e rapidi.

1.3 J. Verne example

Esempio 1.3.1. *(IN ITALIAN) Questo esempio riguarda un crittogramma tratto dal libro "Viaggio al centro della Terra" di J. Verne.*
Il luogo di accesso alla strada che conduce al centro della terra è nascosto in un crittogramma di natura completamente diversa da quello descritto qui sopra: lo riproduciamo nella Figura 1.2. Per la precisione, ci affrettiamo a ricordare che il crittogramma originale è ulteriormente protetto essendo scritto con l'alfabeto runico: per semplicità, qui abbiamo riprodotto la sua traslitterazione nell'alfabeto latino.
Questo crittogramma è del tipo "trasposizione": questo significa che le lettere del testo in chiaro sono rimescolate (potremmo dire anagrammate) seguendo una certa regola, e cioè cambiate di posto rispetto alla loro sequenza originale, ma non sono cambiate in natura. In altre parole, le "a" rimangono "a," le "b" rimangono "b" e cosí via, ma la posizione delle lettere può essere cambiata. Questo fatto può essere scoperto dal crittanalista intento alla decifrazione per mezzo di una analisi di frequenza: infatti, non essendo stata cambiata la natura delle lettere che costituiscono il testo originale, la frequenza delle lettere del testo cifrato è la stessa della frequenza delle lettere del testo in chiaro, e quindi le lettere come le vocali compariranno con una frequenza prossima a quella attesa per qualche lingua.

m . r n l l s	e s r e u e l	s e e c J d e
s g t s s m f	u n t e i e f	n i e d r k e
k t , s a m n	a t r a t e S	S a o d r r n
e m t n a e I	n u a e c t	r r i l S a
A t v a a r	. n s c r c	i e a a b s
c c d r m i	e e u t u l	f r a n t u
d t , i a c	o s e i b o	K e d i i Y

Figure 1.2 - Il crittogramma di Verne. Ci siamo presi la libertà di traslitterare il testo originario che usa l'alfabeto runico in quello latino, ed abbiamo sostituito il simbolo che sta per "mm" in questo alfabeto con "m.".

m . r n l l s
e s r e u e l
s e e c J d e
s g t s s m f
u n t e i e f
n i e d r k e
k t , s a m n
a t r a t e S
S a o d r r n
e m t n a e I
n u a e c t
r r i l S a
A t v a a r
. n s c r c
i e a a b s
c c d r m i
e e u t u l
f r a n t u
d t , i a c
o s e i b o
K e d i i Y

Figure 1.3 - La trasposizione dei blocchi..

Seguendo la trama del romanzo, il primo passo verso la decifrazione è una trasposizione dei blocchi che lo costituiscono: si veda la Figura 1.3. Il testo risultante viene poi letto in verticale:

```
messunkaSenrA.icefdoK.segnittamurtnecertserre
tte,rotaivsadua,ednecsedsadnelacartniiiluJsir
atracSarbmutabiledmekmeretarcsilucoYsleffenSnI
```

e questo viene infine letto dal fondo:

```
InSneffelsYoculiscraterемkemdelibatumbraScartaris
Juliiintracalendasdescende,audasviator,etterrestre
centrumattinges.Kodfeci.ArneSaknussem
```

Il testo in chiaro

Inserendo spazi e segni di interpunzione, possiamo finalmente ricostruire il testo originale:

> *In Sneffels Yoculis craterem kem delibat umbra Scartaris Julii intra calendas descende, audas viator, et terrestre centrum attinges. Kod feci. Arne Saknussem*

Si noti che il latino del testo lascia alquanto a desiderare: infatti, "kem" sta per "quem," "kod" per "quod," "audas" per "audax." La sua traduzione italiana è la seguente:

> *Nel cratere dello Yocul dello Sneffels che l'ombra dello Scartaris tocca alle calende di luglio discendi, audace viaggiatore, e raggiungerai il centro della terra. Ciò che io ho fatto. Arne Saknussem*

Metodi crittografici di questa natura sono usati molto di rado, perché è molto difficile accordarsi sul metodo di cifratura. In questo caso è stato utilizzato dall'autore per assicurarsi che un messaggio cosí delicato potesse essere decifrato solo da persone sufficientemente abili e determinate da poter intraprendere con successo il viaggio verso il centro della terra.

1.4 Strong pseudoprimes

Recalling the notation of §11.8.3 of [6], we denote as ψ_k the smallest composite number such that it is a strong pseudoprime to the first k prime bases. Quite recently Sorenson and Webster [10] proved that

$$\psi_{12} = 318665857834031151167461 = 399165290221 \cdot 798330580441$$

$$\psi_{13} = 3317044064679887385961981 = 2 \cdot 5 \cdot 1287836182261 \cdot 2575672364521$$

thus extending previous results.

1.4.1 On Carmichael numbers (yet)

Recently Bach and Fernando wrote a paper [1] containing some interesting remarks on pseudoprimality algorithms and Carmichael numbers.

1.4.2 Factoring RSA-challenge's numbers

On May 13th, 2016, RSA-220, an integer having 220 decimal digits which belongs to the *RSA-challenges* list, was factored by S. Bai, P. Gaudry, A. Kruppa, E. Thomé and P. Zimmermann, see [9]. Recalling

```
RSA220 = 22601385262034057849416540486101975135080389157197767183211977681094456418179666766085931213065825772506315628866769704480700018111497118630021124879281994874820660701310665866460833279828035603792053919801399464969552 61
```

its factors are:

```
p =  6863656412267566274382371499288437800130842239979164844621244993321541061441464266793821364420842019205499968 7

q = 3292907439486349812049301549212935291916455196536233952462686051169290349309465246333782486639073819176571260 3
```

On December 2nd, 2019, RSA-240, an integer having 240 decimal digits which belongs to the *RSA-challenges* list, was factored by F. Boudot, P. Gaudry, N. Heninger, E. Thomé and P. Zimmermann, see [2]. Recalling

```
RSA240 = 124620366781718784065835044608106590434820374651678805754818788883289666801188210855036039570272508747509864768438458621054865537970253930571891217684318286362846948405301614416430468066875699415246993185704183030512549594371372159029236099
```

its factors are:

```
p =  5094359522858399145550510235808437141326483820241114731866602965218212064697467006203164434788738376062523720496193345 17

q = 2446242088383181505678131390240028966538020925789314014520412213365584770951781552582188977350305906690413020459080714 47
```

On February 28th, 2020, RSA-250, an integer having 242500 decimal digits which belongs to the *RSA-challenges* list, was factored by F. Boudot, P. Gaudry, A. Guillevic, N. Heninger, E. Thomé and P. Zimmermann, see [3]. Recalling

```
RSA-250 = 21403246502407449612644230728393335630086147151447550177977549208814180234471401366433455190958046796109928518724709145876873962619215573630474547705208051190564931066876915900197594056934574522305893259766974716817380693648946998715784949759374979
37
```

its factors are:

```
p =  64135289477071580278790190170577389084825014742943447208116859632024532344630238623598752668347708737661925585694639798853367

q = 33372027594978156556226010605355114227940760344767554666784520987023841729210037080257448673296881877565718986258036932062711
```

1.5 DES (Data Encryption Standard)

It's a variation of the general Feistel system:
- length $2l = 64$ bits;
- number of rounds $r = 16$.
Developed by IBM (in the seventies) (head of the project: H. Feistel).
More detailed description can be found in Buchmann [4], Schneier [8], Menezes, van Oorschot e Vanstone [7] and Landau [5].
New operations (described below) are inserted: Initial Permutation π, Expansion E, Substitution S and Round Permutation P.

1.5.1 DES key

$\mathfrak{M} = \mathfrak{C} = \{0,1\}^{64}$.

$$\mathfrak{K} = \left\{ (b_1, \dots, b_{64}) \in \{0,1\}^{64} \text{ such that } \sum_{i=1}^{8} b_{8j+i} \equiv 1 \bmod 2,\ j = 0,1,\dots,7 \right\}.$$

Hence the length of the DES enciphering/deciphering key can be considered equal to 56 bits (just 7 bits for every byte are independent bits).

1.5.2 DES Encryption

- The text is grouped in blocks, everyone of 64 bits;
- an initial permutation π is applied;
- 16 enciphering rounds of Feistel type are applied;
- the final permutation π^{-1} is applied.
Hence the scheme is:

1) a plaintext M of length equal to 64 bits is permuted thus obtaining $\pi(M)$; then it's split in two parts (32 bits each) $\pi(M) = (L_0, R_0)$;

2) for every $1 \le i \le 16$, we consider the sequence

$$(L_i, R_i) \leftarrow (R_{i-1}, L_{i-1} \oplus P(S(E(R_{i-1}) \oplus K_i));$$

3) after the end of 16 rounds we permutate again the left and the right part; then we define $F_k(M) = \pi^{-1}(R_{16}, L_{16})$.

K_i is a round key which is built using the key k (see the Figure 1.4).

1.5.3 Permutation π

π works as in the following table of dimension 8×8. For simplicity we write the j-th bit o_j of the output just writing its position j and write the final result (64 bits) by rows:

$$\pi(b_1,\dots,b_{64}) = \begin{bmatrix} 58, & 50, & 42, & 34, & 26, & 18, & 10, & 2, \\ 60, & 52, & 44, & 36, & 28, & 20, & 12, & 4, \\ 62, & 54, & 46, & 38, & 30, & 22, & 14, & 6, \\ 64, & 56, & 48, & 40, & 32, & 24, & 16, & 8, \\ 57, & 49, & 41, & 33, & 25, & 17, & 9, & 1, \\ 59, & 51, & 43, & 35, & 27, & 19, & 11, & 3, \\ 61, & 53, & 45, & 37, & 29, & 21, & 13, & 5, \\ 63, & 55, & 47, & 39, & 31, & 23, & 15, & 7 \end{bmatrix}.$$

The inverse permutation is defined by

$$\pi^{-1}(b_1,\dots,b_{64}) = \begin{bmatrix} 40, & 8, & 48, & 16, & 56, & 24, & 64, & 32, \\ 39, & 7, & 47, & 15, & 55, & 23, & 63, & 31, \\ 38, & 6, & 46, & 14, & 54, & 22, & 62, & 30, \\ 37, & 5, & 45, & 13, & 53, & 21, & 61, & 29, \\ 36, & 4, & 44, & 12, & 52, & 20, & 60, & 28, \\ 35, & 3, & 43, & 11, & 51, & 19, & 59, & 27, \\ 34, & 2, & 42, & 10, & 50, & 18, & 58, & 26, \\ 33, & 1, & 41, & 9, & 49, & 17, & 57, & 25 \end{bmatrix}.$$

1.5.4 Round keys

From the starting key k we build a "round key K_i" whose length is 48 bits. We build K_i from the starting key k by using a different set of bits in every round. First we have to remove from k the bits used for the "parity check" (bits n. 8, 16, 24, 32, 40, 48, 56, 64) thus reducing its size to 56 bits; we will call $\widetilde{k}$ this reduced key.
Let $k_0 = \widetilde{k}$. K_i is chosen by using the two halves of k_{i-1} , $i = 1,\dots,16$, and then by rotating every part (circular shift) of one or two bits (one bit for the rounds $1,2,9,16$; two bits for the remaining rounds) on the left. The resulting news halves are then glued together and their 56 bit are denoted by b_j, $j = 1,\dots,56$. We will denote this "rotated key" as k_i.
Then we choose and sort as follows a subset of 48 bits of k_i:

$$K_i = \begin{bmatrix} 14, & 17, & 11, & 24, & 1, & 5, \\ 3, & 28, & 15, & 6, & 21, & 10, \\ 23, & 19, & 12, & 4, & 26, & 8, \\ 16, & 7, & 27, & 20, & 13, & 2, \\ 41, & 52, & 31, & 37, & 47, & 55, \\ 30, & 40, & 51, & 45, & 33, & 48, \\ 44, & 49, & 39, & 56, & 34, & 53, \\ 46, & 42, & 50, & 36, & 29, & 32 \end{bmatrix}.$$

(it means the first bits of K_i are $b_{14}, b_{17}, b_{11}, b_{24}, b_1, b_5, \dots$).
This rotation procedure is done to assure that two round keys of distict rounds are different ($K_j \neq K_i$ if $j \neq i$).

1.5.5 Functions E, S e P

The cryptosystem included in the DES has :
length is equal to 32 bits;
key length is equal to 48 bits.
Function E
The expansion function

$$E\colon \{0,1\}^{32} \to \{0,1\}^{48}$$

assures that we can apply the XOR operation at step 2).

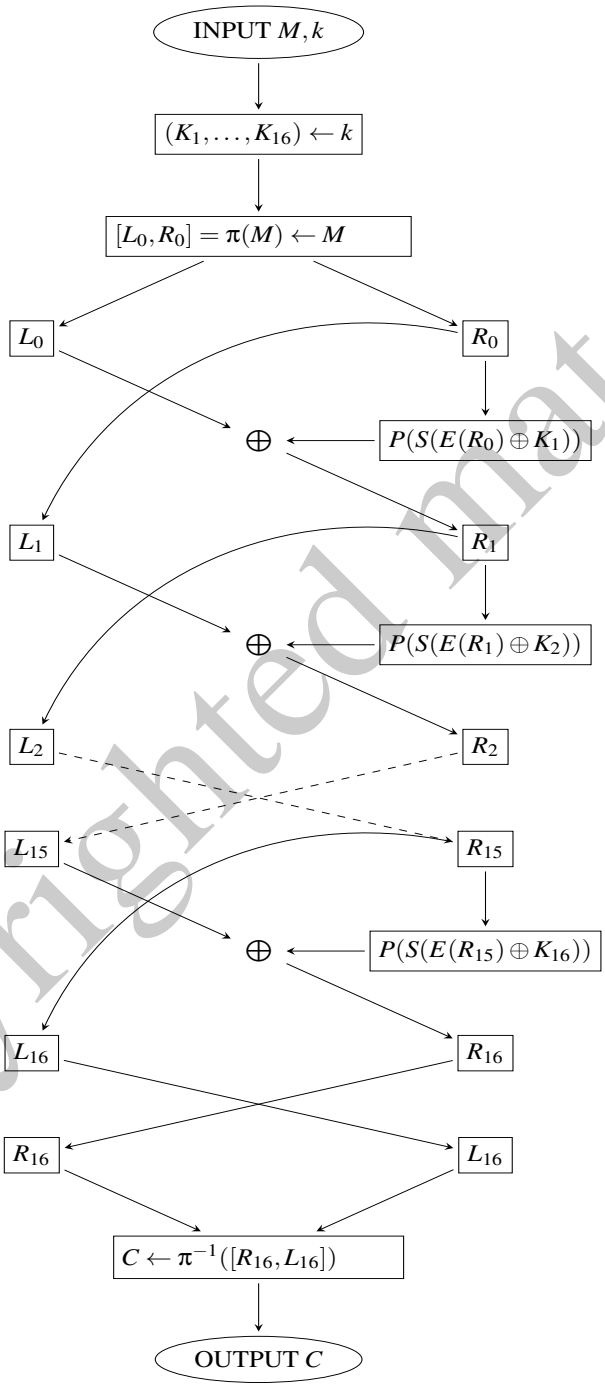

Figure 1.4 - Schema di funzionamento della codifica del DES.

It works on 6-bits blocks by copying the first four and writing twice the last two bits of every 6-bits block.

$$E(b_1,\dots,b_{32}) = \begin{bmatrix} 32, & 1, & 2, & 3, & 4, & 5, \\ 4, & 5, & 6, & 7, & 8, & 9, \\ 8, & 9, & 10, & 11, & 12, & 13, \\ 12, & 13, & 14, & 15, & 16, & 17, \\ 16, & 17, & 18, & 19, & 20, & 21, \\ 20, & 21, & 22, & 23, & 24, & 25, \\ 24, & 25, & 26, & 27, & 28, & 29, \\ 28, & 29, & 30, & 31, & 32, & 1 \end{bmatrix}.$$

E starts to work on the block $32,1,2,3,4,5$ and finishes with the block $28,29,30,31,32,1$.

Function S

S works in parallel on 8-bits blocks; from a mathematical point of view, we can consider S as the vector function defined by $(S_1,S_2,S_3,S_4,S_5,S_6,S_7,S_8)$ in which $S_j\colon \{0,1\}^6 \to \{0,1\}^4$, $j=1,\dots,8$.

S is not linear (this is the source of DES hardness).

To see how the S_j are built (they are public from 1993) see Landau [5], Buchmann [4] or Stinson [11].

S_j works by substitution in the following way: the bits b_2,b_3,b_4,b_5 of every 6-bits block are considered as inputs and the bits b_1,b_6 are the "instructions". In this way every S_j can be represented as a matrix of dimension 4×16 whose entries are in $\{0,\dots,15\}$.

For example S_3:

$$S_3(b_1,b_2,b_3,b_4,b_5,b_6) = \begin{pmatrix} 10 & 0 & 9 & 14 & 6 & 3 & 15 & 5 & 1 & 13 & 12 & 7 & 11 & 4 & 2 & 8 \\ 13 & 7 & 0 & 9 & 3 & 4 & 6 & 10 & 2 & 8 & 5 & 14 & 12 & 11 & 15 & 1 \\ 13 & 6 & 4 & 9 & 8 & 15 & 3 & 0 & 11 & 1 & 2 & 12 & 5 & 10 & 14 & 7 \\ 1 & 10 & 13 & 0 & 6 & 9 & 8 & 7 & 4 & 15 & 14 & 3 & 11 & 5 & 2 & 12 \end{pmatrix}.$$

The bits b_1,b_6 define the row of the matrix; the bits b_2,b_3,b_4,b_5 define the column. The output is the 4 digits binary expansion of the corresponding entry of the matrix.

Remark: every row of S_3 is a permutation of $\mathbb{Z}_{16}$.

The output of the function S is obtained by gluing the eight outputs of the functions S_j; hence the final length is 32 bits.

Complete description of the S-boxes:

$$S_1(b_1,b_2,b_3,b_4,b_5,b_6) = \begin{pmatrix} 14 & 4 & 13 & 1 & 2 & 15 & 11 & 8 & 3 & 10 & 6 & 12 & 5 & 9 & 0 & 7 \\ 0 & 15 & 7 & 4 & 14 & 2 & 13 & 1 & 10 & 6 & 12 & 11 & 9 & 5 & 3 & 8 \\ 4 & 1 & 14 & 8 & 13 & 6 & 2 & 11 & 15 & 12 & 9 & 7 & 3 & 10 & 5 & 0 \\ 15 & 12 & 8 & 2 & 4 & 9 & 1 & 7 & 5 & 11 & 3 & 14 & 10 & 0 & 6 & 13 \end{pmatrix}$$

$$S_2(b_1,b_2,b_3,b_4,b_5,b_6) = \begin{pmatrix} 15 & 1 & 8 & 14 & 6 & 11 & 3 & 4 & 9 & 7 & 2 & 13 & 12 & 0 & 5 & 10 \\ 3 & 13 & 4 & 7 & 15 & 2 & 8 & 14 & 12 & 0 & 1 & 10 & 6 & 9 & 11 & 5 \\ 0 & 14 & 7 & 11 & 10 & 4 & 13 & 1 & 5 & 8 & 12 & 6 & 9 & 3 & 2 & 15 \\ 13 & 8 & 10 & 1 & 3 & 15 & 4 & 2 & 11 & 6 & 7 & 12 & 0 & 5 & 14 & 9 \end{pmatrix}$$

$$S_3(b_1,b_2,b_3,b_4,b_5,b_6) = \begin{pmatrix} 10 & 0 & 9 & 14 & 6 & 3 & 15 & 5 & 1 & 13 & 12 & 7 & 11 & 4 & 2 & 8 \\ 13 & 7 & 0 & 9 & 3 & 4 & 6 & 10 & 2 & 8 & 5 & 14 & 12 & 11 & 15 & 1 \\ 13 & 6 & 4 & 9 & 8 & 15 & 3 & 0 & 11 & 1 & 2 & 12 & 5 & 10 & 14 & 7 \\ 1 & 10 & 13 & 0 & 6 & 9 & 8 & 7 & 4 & 15 & 14 & 3 & 11 & 5 & 2 & 12 \end{pmatrix}$$

$$S_4(b_1,b_2,b_3,b_4,b_5,b_6) = \begin{pmatrix} 7 & 13 & 14 & 3 & 0 & 6 & 9 & 10 & 1 & 2 & 8 & 5 & 11 & 12 & 4 & 15 \\ 13 & 8 & 11 & 5 & 6 & 15 & 0 & 3 & 4 & 7 & 2 & 12 & 1 & 10 & 14 & 9 \\ 10 & 6 & 9 & 0 & 12 & 11 & 7 & 13 & 15 & 1 & 3 & 14 & 5 & 2 & 8 & 4 \\ 3 & 15 & 0 & 6 & 10 & 1 & 13 & 8 & 9 & 4 & 5 & 11 & 12 & 7 & 2 & 14 \end{pmatrix}$$

$$S_5(b_1,b_2,b_3,b_4,b_5,b_6) =$$
$$\begin{pmatrix} 2 & 12 & 4 & 1 & 7 & 10 & 11 & 6 & 8 & 5 & 3 & 15 & 13 & 0 & 14 & 9 \\ 14 & 11 & 2 & 12 & 4 & 7 & 13 & 1 & 5 & 0 & 15 & 10 & 3 & 9 & 8 & 6 \\ 4 & 2 & 1 & 11 & 10 & 13 & 7 & 8 & 15 & 9 & 12 & 5 & 6 & 3 & 0 & 14 \\ 11 & 8 & 12 & 7 & 1 & 14 & 2 & 13 & 6 & 15 & 0 & 9 & 10 & 4 & 5 & 3 \end{pmatrix}$$

$$S_6(b_1,b_2,b_3,b_4,b_5,b_6) =$$
$$\begin{pmatrix} 12 & 1 & 10 & 15 & 9 & 2 & 6 & 8 & 0 & 13 & 3 & 4 & 14 & 7 & 5 & 11 \\ 10 & 15 & 4 & 2 & 7 & 12 & 9 & 5 & 6 & 1 & 13 & 14 & 0 & 11 & 3 & 8 \\ 9 & 14 & 15 & 5 & 2 & 8 & 12 & 3 & 7 & 0 & 4 & 10 & 1 & 13 & 11 & 6 \\ 4 & 3 & 2 & 12 & 9 & 5 & 15 & 10 & 11 & 14 & 1 & 7 & 6 & 0 & 8 & 13 \end{pmatrix}$$

$$S_7(b_1,b_2,b_3,b_4,b_5,b_6) =$$
$$\begin{pmatrix} 4 & 11 & 2 & 14 & 15 & 0 & 8 & 13 & 3 & 12 & 9 & 7 & 5 & 10 & 6 & 1 \\ 13 & 0 & 11 & 7 & 4 & 9 & 1 & 10 & 14 & 3 & 5 & 12 & 2 & 15 & 8 & 6 \\ 1 & 4 & 11 & 13 & 12 & 3 & 7 & 14 & 10 & 15 & 6 & 8 & 0 & 5 & 9 & 2 \\ 6 & 11 & 13 & 8 & 1 & 4 & 10 & 7 & 9 & 5 & 0 & 15 & 14 & 2 & 3 & 12 \end{pmatrix}$$

$$S_8(b_1,b_2,b_3,b_4,b_5,b_6) =$$
$$\begin{pmatrix} 13 & 2 & 8 & 4 & 6 & 15 & 11 & 1 & 10 & 9 & 3 & 14 & 5 & 0 & 12 & 7 \\ 1 & 15 & 13 & 8 & 10 & 3 & 7 & 4 & 12 & 5 & 6 & 11 & 0 & 14 & 9 & 2 \\ 7 & 11 & 4 & 1 & 9 & 12 & 14 & 2 & 0 & 6 & 10 & 13 & 15 & 3 & 5 & 8 \\ 2 & 1 & 14 & 7 & 4 & 10 & 8 & 13 & 15 & 12 & 9 & 0 & 3 & 5 & 6 & 11 \end{pmatrix}$$

Function P

P is the following permutation of $\{1,\dots,32\}$:

$$P(b_1,\dots,b_{32}) = \begin{bmatrix} 16, & 7, & 20, & 21, & 29, & 12, & 28, & 17, \\ 1, & 15, & 23, & 26, & 5, & 18, & 31, & 10, \\ 2, & 8, & 24, & 14, & 32, & 27, & 3, & 9, \\ 19, & 13, & 30, & 6, & 22, & 11, & 4, & 25 \end{bmatrix}.$$

P is used to distribute, in the next round, a round output to a different function S_j .

1.5.6 DES deciphering algorithm

The same operations (π, π^{-1} and the left-right exchanges) have to be applied in their inverse order (remark: the round keys has to used in their inverse order too).

For evert round i, $i = 16,15,14,\dots,1$, the fundamental relation in the deciphering algorithm is (input: $[L_i,R_i]$ output $[L_{i-1},R_{i-1}]$)

$$(R_{i-1},L_{i-1}) \leftarrow (L_i, R_i \oplus P(S(E(L_i) \oplus K_i))).$$

1.5.7 Triple DES (TDES)

This is a way to increase the length of the DES keys. Let k_1,k_2 two usual DES keys.

TDES encryption: the EDE scheme. The enciphering function of TDES runs as follows.

1) use the standard DES enciphering procedure with key k_1;

2) use the standard DES deciphering procedure with key k_2;

3) use the standard DES enciphering procedure with key k_1.

TDES decryption: the DED scheme. The deciphering function of TDES run as follows.

1) use the standard DES deciphering procedure with key k_1;

2) use the standard DES enciphering procedure with key k_2;

3) use the standard DES deciphering procedure with key k_1.

This way the length of a TDES key can be considered as 112 bits.
Why using the EDE scheme instead of an EEE one? For backward compatibility: when $k_1 = k_2$, the TDES works as an usual DES scheme with key equal to k_1.

1.6 AES (Rijndael)

Developed using an international public call started in 1998 (fifteen submitted algorithms); in 2001 the RIJNDAEL method by Vincent Rijmen and Joan Daemen was chosen as the new standard.

1.6.1 High level description

AES block length: $l = 128$ bits.
AES key length: can be 128, 192 or 256 bits.
In fact we have three different methods: AES128, AES192 and AES256.
We will see just AES128: in the following it will be named AES.
So $\mathfrak{M} = \{0,1\}^{128} = \mathfrak{C} = \mathfrak{K}$.
AES works using four functions: E (Expansion), S (Substitution), SR (Shift Rows), MC (Mix Columns).
The high-level AES structure is in Figure 1.5 and in the following pseudocode in which s is called *state* and, assuming that we want to encipher a message M, the algorithm starts letting s as M and, after the "for" loop is ended, s is equal to the ciphertext c.
The E-function E as input has a string of 128 bits; as output has a vector of 11 keys $(K_0, \dots, K_{10})$.
The other functions (S, SR, MC) are bijective functions on $\{0,1\}^{128}$.
AES has 10 rounds (in fact the general algorithm has a number of rounds depending on the length of the key: 12 if the key has 192 bits; 14 if the key has 256 bits).
The differences between two of the first nine rounds are just in the round key K_j used. Moreover, in the last round the function MC is not executed.

1.6.2 Function S

S is obtained starting from a function $\mathbb{F}_{2^8} \to \mathbb{F}_{2^8}$.
Let $\mathbb{F}_{2^8} = \mathbb{F}_2[x]/m(x)$, where $m(x) = x^8 + x^4 + x^3 + x + 1$ is an irreducible polynomial on $\mathbb{F}_2$.
A byte $\alpha = (b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0)_2$ can be considered as a polynomial $b_0 + b_1 x + b_2 x^2 + b_3 x^3 + b_4 x^4 + b_5 x^5 + b_6 x^6 + b_7 x^7 \in \mathbb{F}_2[x]$ and the operations defined on bytes can be, in fact, considered as operations on polynomials.
Remark that in $\mathbb{F}_2[x]/m(x)$ we have $x^8 = x^4 + x^3 + x + 1 = (00011011)_2 = (27)_{10} = (1B)_{16}$ (recall that in $\mathbb{F}_2$ one has $-1 \equiv 1 \bmod 2$). Moreover it is easy to compute the product of a byte with the byte $(00000010)_2 = (02)_{16}$ (i.e the polynomial x): it is sufficient to shift the eight bits of one position to the left and to insert a 0 as the less significant bit. Pay attention to the fact that the most significant bit has to be stored somewhere since if it is equal to 0 than the operation is finished; otherwise we have to add the polynomial x^8 (i.e. $(1B)_{16}$). Letting `shiftL1` the shift operation of one bit to the left, we have

$$\alpha \cdot x = \alpha \cdot (02)_{16} = \begin{cases} \texttt{shiftL1}(\alpha) & \text{if } b_7 = 0 \\ \texttt{shiftL1}(\alpha) \oplus (1B)_{16} & \text{if } b_7 = 1. \end{cases}$$

- Inverses: every non-zero element in $\mathbb{F}_{2^8}$ has a multiplicative inverse. Define the map $\texttt{inv}\colon (\mathbb{F}_{2^8})^* \to (\mathbb{F}_{2^8})^*$ as $\texttt{inv}(b) = b^{-1}$. Then we extend it to $\mathbb{F}_{2^8}$ by letting $\texttt{inv}(0) = 0$.

ALGORITMO $AES(M,k)$

1 $(K_0,\dots,K_{10}) \leftarrow E(k)$
2 $s \leftarrow M \oplus K_0$
3 **for** $r = 1$; $r \le 10$; $++r$
4 $\quad s \leftarrow S(s)$
5 $\quad s \leftarrow SR(s)$
6 $\quad$ **if** $r \le 9$
7 $\quad\quad s \leftarrow MC(s)$
8 $\quad$ **endif**
9 $\quad s \leftarrow s \oplus K_r$
10 **endfor**
11 **return**(s)

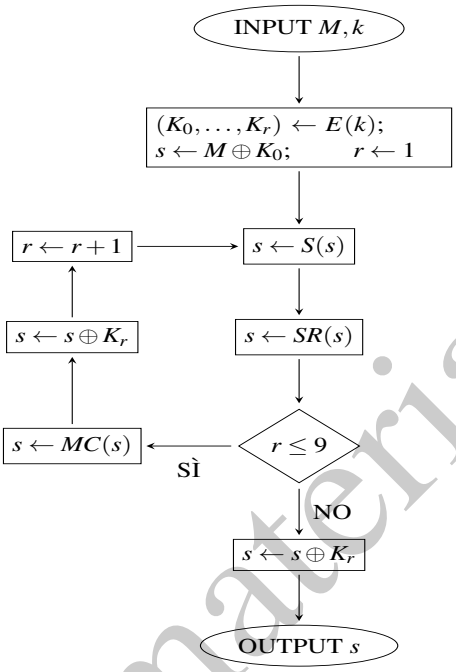

Figure 1.5 - Lo pseudocodice e lo schema di cifratura di AES. M indica il messaggio in chiaro, k la chiave di cifratura, K_r le chiavi di ciclo, s lo stato, le funzioni E, S, SR e MC sono descritte nel seguito.

- $\pi_S\colon \mathbb{F}_{2^8} \to \mathbb{F}_{2^8}$: is the composition of `inv` and the map $\sigma\colon \mathbb{F}_{2^8} \to \mathbb{F}_{2^8}$ defined as follows. Letting $\alpha = (b_7b_6b_5b_4b_3b_2b_1b_0)_2 \in \mathbb{F}_{2^8}$ then $\sigma(\alpha) = (b'_7b'_6b'_5b'_4b'_3b'_2b'_1b'_0)_2$ where

$$\begin{pmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}.$$

In fact the final form of the function $\pi_S = \sigma \circ \texttt{inv}$ can be written as follows:

$$\begin{pmatrix} 63 & 7C & 77 & 7B & F2 & 6B & 6F & C5 & 30 & 01 & 67 & 2B & FE & D7 & AB & 76 \\ CA & 82 & C9 & 7D & FA & 59 & 47 & F0 & AD & D4 & A2 & AF & 9C & A4 & 72 & C0 \\ B7 & FD & 93 & 26 & 36 & 3F & F7 & CC & 34 & A5 & E5 & F1 & 71 & D8 & 31 & 15 \\ 04 & C7 & 23 & C3 & 18 & 96 & 05 & 9A & 07 & 12 & 80 & E2 & EB & 27 & B2 & 75 \\ 09 & 83 & 2C & 1A & 1B & 6E & 5A & A0 & 52 & 3B & D6 & B3 & 29 & E3 & 2F & 84 \\ 53 & D1 & 00 & ED & 20 & FC & B1 & 5B & 6A & CB & BE & 39 & 4A & 4C & 58 & CF \\ D0 & EF & AA & FB & 43 & 4D & 33 & 85 & 45 & F9 & 02 & 7F & 50 & 3C & 9F & A8 \\ 51 & A3 & 40 & 8F & 92 & 9D & 38 & F5 & BC & B6 & DA & 21 & 10 & FF & F3 & D2 \\ CD & 0C & 13 & EC & 5F & 97 & 44 & 17 & C4 & A7 & 7E & 3D & 64 & 5D & 19 & 73 \\ 60 & 81 & 4F & DC & 22 & 2A & 90 & 88 & 46 & EE & B8 & 14 & DE & 5E & 0B & DB \\ E0 & 32 & 3A & 0A & 49 & 06 & 24 & 5C & C2 & D3 & AC & 62 & 91 & 95 & E4 & 79 \\ E7 & C8 & 37 & 6D & 8D & D5 & 4E & A9 & 6C & 56 & F4 & EA & 65 & 7A & AE & 08 \\ BA & 78 & 25 & 2E & 1C & A6 & B4 & C6 & E8 & DD & 74 & 1F & 4B & BD & 8B & 8A \\ 70 & 3E & B5 & 66 & 48 & 03 & F6 & 0E & 61 & 35 & 57 & B9 & 86 & C1 & 1D & 9E \\ E1 & F8 & 98 & 11 & 69 & D9 & 8E & 94 & 9B & 1E & 87 & E9 & CE & 55 & 28 & DF \\ 8C & A1 & 89 & 0D & BF & E6 & 42 & 68 & 41 & 99 & 2D & 0F & B0 & 54 & BB & 16 \end{pmatrix}.$$

The entries of the matrix are hexadecimal numbers (for simplicity we omitted $(\cdot)_{16}$). The table has to be read as follows: $\pi_S(00)_{16} = (63)_{16}, \pi_S(01)_{16} = (7C)_{16}, \dots, \pi_S(FF)_{16} = (16)_{16}$.
The map $S\colon \{0,1\}^{128} \to \{0,1\}^{128}$ works on the state s (16 bytes are a state s) and it is defined as a vector $(\pi_S(\alpha_0),\dots,\pi_S(\alpha_{15}))$, where $(\alpha_0,\dots,\alpha_{15}) \in \{0,1\}^{128}$ and every α_j is a byte.

1.6.3 State *s*

A state *s* is defined as a list of 16 bytes.
s can be written as a matrix in which every byte of the state is inserted as a column:

$$s = \begin{pmatrix} s_0 & s_4 & s_8 & s_{12} \\ s_1 & s_5 & s_9 & s_{13} \\ s_2 & s_6 & s_{10} & s_{14} \\ s_3 & s_7 & s_{11} & s_{15} \end{pmatrix}. \tag{1.1}$$

1.6.4 Function *SR*

Shift row: it works on a state considering every row as a ring, moving the second row to the left of one position, the third one of two positions and the fourth one of three positions (the first row is not modified). The final result is:

$$SR(s_0\, s_1\, s_2 \ldots s_{15}) = s_0\, s_5\, s_{10}\, s_{15}\, s_4\, s_9\, s_{14}\, s_3\, s_8\, s_{13}\, s_2\, s_7\, s_{12}\, s_1\, s_6\, s_{11}.$$

1.6.5 Function *MC*

Mix columns: it is defined using an auxiliary function π_{MC} which works on sets of 4 bytes: $\pi_{MC}(\alpha_0\alpha_1\alpha_2\alpha_3) = (\alpha'_0\alpha'_1\alpha'_2\alpha'_3)$ where

$$\begin{pmatrix} \alpha'_0 \\ \alpha'_1 \\ \alpha'_2 \\ \alpha'_3 \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix},$$

in which the entries of the matrix of dimension 4×4 are hexadecimal numbers. The products are performed in $\mathbb{F}_{256}$.
(REMARK: such a computation is in fact the product modulo x^4+1 of the polynomial $\alpha(x) = \alpha_3 x^3 + \alpha_2 x^2 + \alpha_1 x + \alpha_0$, $\alpha_i \in \mathbb{F}_{256}$, with the fixed polynomial $c(x) = (03)_{16}x^3 + (01)_{16}x^2 + (01)_{16}x + (02)_{16}$, where again the coefficients are elements in $\mathbb{F}_{256}$.)
The function *MC* is then defined as a vector function by combining the action of π_{MC} on every column of a state *s* (*s* described as in (1.1)).

1.6.6 Function *E*: round keys definition

k: has 128 bits.
K_i, $i = 0, \ldots, 10$: are 10 round keys plus the starting one K_0; everyone has 128 bits (i.e. 16 bytes).
word: a set of 4 bytes.
Hence every key has four words.
The function *E* acts on one word and then it repeats the operation on the next word.
We'll use some one-word constants defined by

$$C_i = [x^{i-1}|(00)_{16}|(00)_{16}|(00)_{16}].$$

In fact they are $C_1 = [01|00|00|00]$, $C_2 = [02|00|00|00]$, $C_3 = [04|00|00|00]$, $C_4 = [08|00|00|00]$, $C_5 = [10|00|00|00]$, $C_6 = [20|00|00|00]$, $C_7 = [40|00|00|00]$, $C_8 = [80|00|00|00]$, $C_9 = [1B|00|00|00]$, $C_{10} = [36|00|00|00]$, in which the numbers on the right hand sides are hexadecimal numbers.
The first step to define *E* is to set the first round key as *k*: $K_0 = k$. Then we build, word by word, $K_1 = (K_1[0], K_1[1], K_1[2], K_1[3])$. Let $K_1[i]$ be the *i*-th word of K_1. We will call `shiftL8` a circular shift of 8 bits to the left. Hence *E*, to build K_1, acts as follows:

$$\begin{aligned} K_1[0] &\leftarrow K_0[0] \oplus \pi'_S(\texttt{shiftL8}(K_0[3])) \oplus C_1 \\ K_1[1] &\leftarrow K_0[1] \oplus K_1[0] \\ K_1[2] &\leftarrow K_0[2] \oplus K_1[1] \\ K_1[3] &\leftarrow K_0[3] \oplus K_1[2], \end{aligned}$$

where $\pi'_S(\alpha_0, \alpha_1, \alpha_2, \alpha_3) = (\pi_S(\alpha_0), \pi_S(\alpha_1), \pi_S(\alpha_2), \pi_S(\alpha_3))$.

Then we compute K_2 and so on. The pseudocode of the expansion function is the following:

AES EXPANSION FUNCTION E

1 $K_0 \leftarrow k$
2 **for** $j = 1$**;** $j \le 10$**;** $++\,j$
3 $\quad K_j[0] \leftarrow K_{j-1}[0] \oplus \pi'_S(\texttt{shiftL8}(K_{j-1}[3])) \oplus C_j$
4 $\quad K_j[1] \leftarrow K_{j-1}[1] \oplus K_j[0]$
5 $\quad K_j[2] \leftarrow K_{j-1}[2] \oplus K_j[1]$
6 $\quad K_j[3] \leftarrow K_{j-1}[3] \oplus K_j[2]$
7 **endfor**
8 **return**$(K_0, \dots, K_{10})$

1.6.7 AES deciphering process

Every function used in the enciphering process is in fact an invertible function (S is a permutation and the matrix used in MC is an invertible one). S^{-1} is obtained with right circular shifts (of respectively one, two, three positions for rows n. $2,3,4$ of the state). The matrix inverse of the matrix used in π_{MC} is

$$\begin{pmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{pmatrix}$$

as it can be verified via some tedious, but trivial, computation.
The function $\sigma^{-1}\colon \mathbb{F}_{256} \to \mathbb{F}_{256}$ is the inverse function of σ and it is defined as follows. Letting $\alpha = (b_7b_6b_5b_4b_3b_2b_1b_0)_2 \in \mathbb{F}_{256}$, we write $\sigma^{-1}(\alpha) = (b'_7b'_6b'_5b'_4b'_3b'_2b'_1b'_0)_2$ where

$$\begin{pmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

Even in this case the verification that $\sigma^{-1}(\sigma(\alpha)) = \alpha$ requires tedious but trivial computations.
Deciphering remarks:
- the same operations used to cipher have to be performed in their inverse order;
- the sequence of the round keys has to be used in its inverse order;
- the operations SR, S, MC has to be replaced by their inverses (remark: the sum with the round key is equal to its inverse function).

Bibliografia

[1] E. Bach and R. Fernando. Infinitely many carmichael numbers for a modified miller-rabin prime test. In *Proceedings of the ACM on International Symposium on Symbolic and Algebraic Computation*, ISSAC 2016, pages 47–54, New York, NY, USA, 2016. ACM.

[2] F. Boudot, P. Gaudry, N. Heninger, E. Thomé, and P. Zimmermann. Factorisation of RSA-240 with CADO-NFS, 2019. `https://caramba.inria.fr/dlp240-rsa240.txt`.

[3] F. Boudot, P. Gaudry, N. Heninger, E. Thomé, and P. Zimmermann. Comparing the difficulty of factorization and discrete logarithm: a 240-digit experiment, 2020. `https://caramba.inria.fr/dlp240-rsa240.txt`.

[4] J. Buchmann. *Introduction to Cryptography*. Springer-Verlag, second edition, 2004.

[5] S. Landau. Communications Security for the Twenty-First Century: the Advanced Encryption Standard. *Notices of the American Mathematical Society*, 47:450–459, 2000.

[6] A. Languasco and A. Zaccagnini. *Manuale di Crittografia*. Ulrico Hoepli Editore, 2015. `http://www.hoepli.it/libro/manuale-di-crittografia/9788820366902.html`.

[7] A. Menezes, P. van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*. CRC Press, Boca Raton, 1996. `http://www.cacr.math.uwaterloo.ca/hac`.

[8] B. Schneier. *Applied Cryptography: Protocols, Algorithms, and Source Code in C*. Wiley, second edition, 1996.

[9] B. Shi, P. Gaudy, A. Kruppa, E. Thomé, and P. Zimmermann. Factorisation of RSA-220 with CADO-NFS, 2016. `https://lists.gforge.inria.fr/pipermail/cado-nfs-discuss/2016-May/000626.html`; `https://members.loria.fr/PZimmermann/papers/rsa220.pdf`.

[10] J. P. Sorenson and J. Webster. Strong Pseudoprimes to Twelve Prime Bases. *Math. Comp.*, (86):985–1003, 2017. `http://dx.doi.org/10.1090/mcom/3134`.

[11] D. Stinson. *Cryptography: Theory and Practice*. Chapman & Hall/CRC, Boca Raton, third edition, 2006.