Estensioni di DES

Limiti di DES

- DES è vulnerabile ad attacchi a forza bruta (lunghezza della chiave: 56 bit):
  - nel 1977 si riteneva che già esistesse la tecnologia per violare DES in 10 ore (costo 20 milioni USD)
  - nel 1998 la *Electronic Frontier Foundation* costruisce una macchina 'DES cracker' in grado di violare DES in tre giorni (costo 250 000 USD)

NOTA: l'attacco a forza bruta richiede che l'analista sia in grado di riconoscere il testo in chiaro decifrato!

Soluzioni

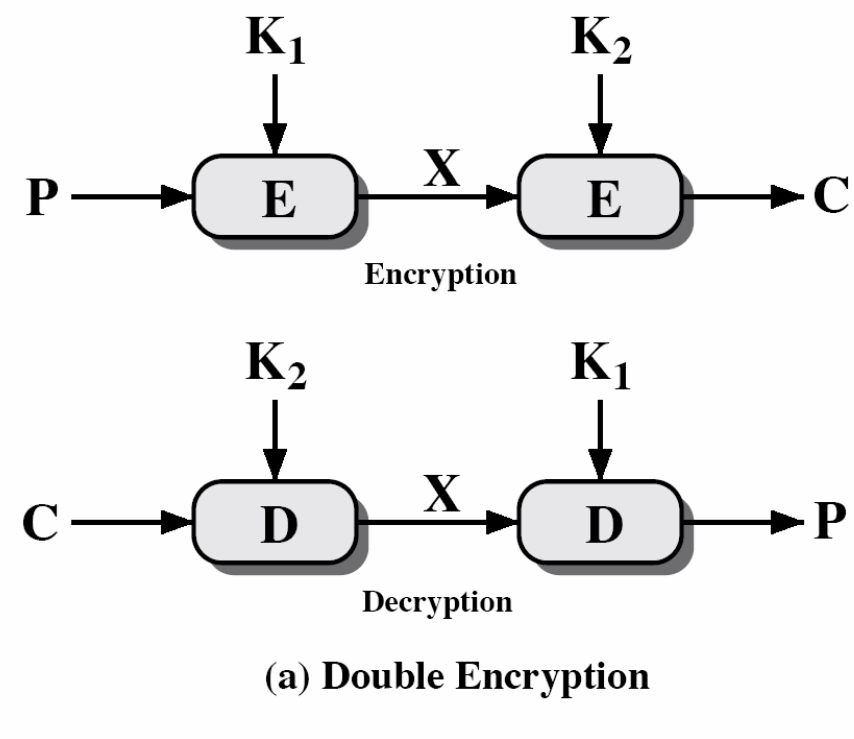
- Progettare un algoritmo di crittografia completamente nuovo e più resistente agli attacchi a forza bruta (i.e., AES).
- Utilizzare una crittografia DES multipla con più chiavi:
  - Double DES
  - Triple DES a due chiavi
  - Triple DES a tre chiavi

Double DES

- Due fasi di crittografia in cascata che utilizzano due chiavi distinte K_1 e K_2:

$$C = E_{K_2}[E_{K_1}[P]]$$

$$P = E_{K_1}[E_{K_2}[C]]$$



(a) Double Encryption

Double DES

- Apparentemente Double DES usa una chiave di 56x2=112 bit.
- Domanda: date due chiavi K_1 e K_2, è possibile trovare una chiave K_3 tale che

$$\mathrm{E}_{\mathrm{K}_2}[\mathrm{E}_{\mathrm{K}_1}[P]] = \mathrm{E}_{\mathrm{K}_3}[P]\,?$$

- Risposta: in generale, applicando DES due volte si ottiene un mappaggio che non può essere ottenuto da una sola applicazione di DES.

Attacco Meet-in-the-Middle

$$C = E_{K_2}[E_{K_1}[P]] \quad \Rightarrow \quad X = E_{K_1}[P] = D_{K_2}[C]$$

- Data una coppia nota (P,C), l'attacco procede come segue:
  - Si esegue la crittografia di P per tutti i 2^{56} valori di K_1
  - Si esegue la decrittografia di C per tutti i 2^{56} valori di K_2
  - Quando si trova una corrispondenza, si esegue il test delle due chiavi risultanti contro una nuova coppia (P_a,C_a). Se le due chiavi producono il testo cifrato corretto, si tratta delle chiavi corrette.

Attacco Meet-in-the-Middle

- Per ogni testo in chiaro P, Double DES produce uno fra 2^{64} possibili valori cifrati.
- Poiché Double DES usa chiavi di 112 bit, ci sono in media $2^{112}/2^{64}= 2^{48}$ chiavi che producono lo stesso testo cifrato C.
- L'attaco meet-in-the-middle produce in media 2^{48} falsi allarmi sulla coppia (P,C).
- Utilizzando una seconda coppia (P_a,C_a), i falsi allarmi si riducono a $2^{48-64}= 2^{-16}$, ovvero la probabilità di individuare le chiavi corrette è $1- 2^{-16}$.

Attacco Meet-in-the-Middle

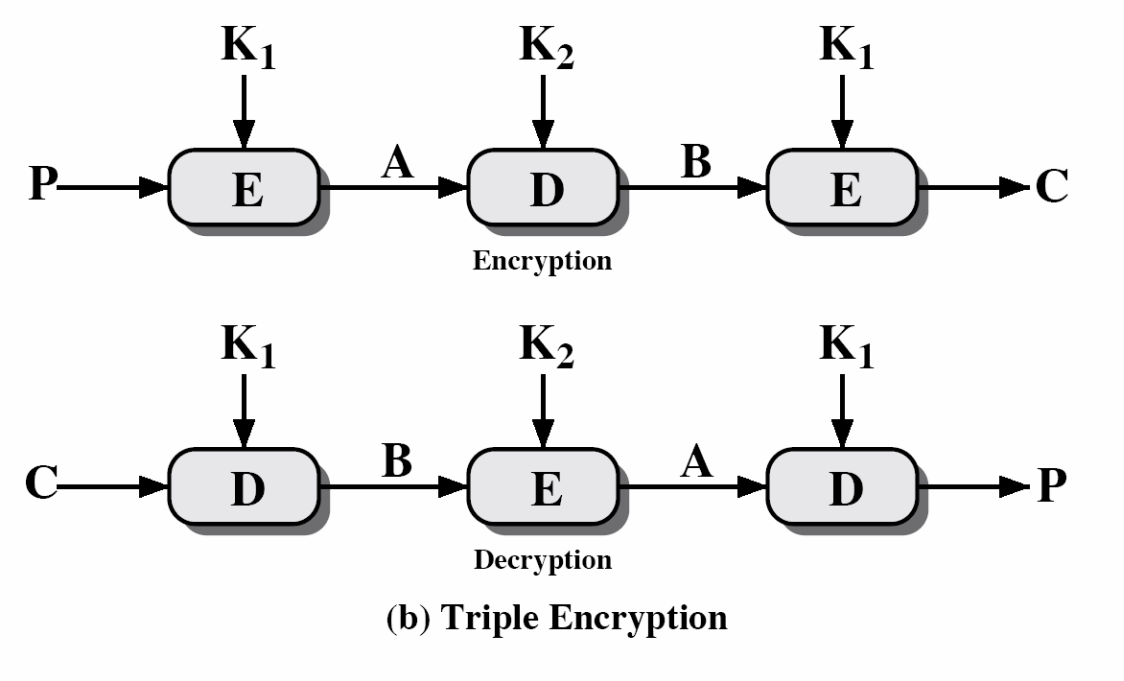
- L'attacco meet-in-the-middle ha una complessità computazionale pari 2^{56}.

- Di conseguenza, pur utilizzando chiavi di 112 bit, Double DES può essere violato in maniera relativamente semplice con un attacco a testo in chiaro noto.

NOTA: Si ricordi che DES può essere violato con un attacco a testo in chiaro noto con un impegno pari a 2^{55} .

Triple DES con due chiavi

- Una contromisura contro l'attacco meet-in-the-middle è l'uso di tre fasi di crittografia:



(b) Triple Encryption

Triple DES con due chiavi

- Vengono ancora utilizzate due chiavi crittografiche da 56 bit ciascuna.
- La presenza di una decrittografia nella seconda fase consente agli utenti Triple DES di decrittografare i dati crittografati dagli utenti DES (basta porre $K_1=K_2$).
- L'attacco meet-in-the-middle a testo in chiaro noto avrebbe una complessità 2^{112}.

Attacchi a Triple DES con due chiavi

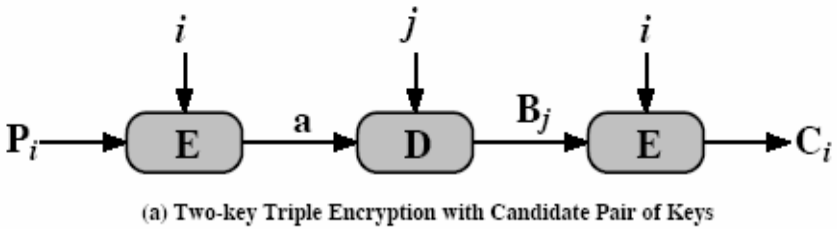
- Se è nota la coppia (A,C), il problema si riduce ad un attacco a Double DES.

- In pratica, se le due chiavi K_1 e K_2 sono sconosciute, A non è noto all'analista. In tal caso si procede fissando una valore potenziale di A e si tenta di trovare una coppia nota (P,C) che produce A.

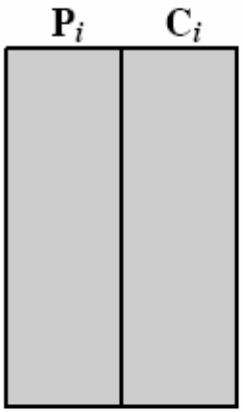
Schema di attacco (1/3)

- Ottenere n coppie (P,C) e inserirle nella tabella (b) ordinate per valori di P.

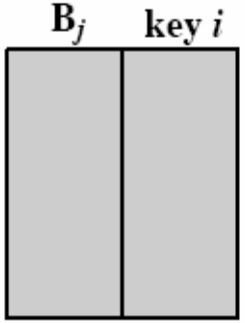
- Scegliere un valore A=a. Per tutte le 2^{56} chiavi K_i=i calcolare $P_i=D_i[a]$. Per ciascun P_i presente anche nella tabella (b) creare una voce nella tabella (c) contenente K_i e $B_i=D_i[C]$. Ordinare la tabella (c) sulla base dei valori di B.



(a) Two-key Triple Encryption with Candidate Pair of Keys



(b) Table of n known plaintext-ciphertext pairs, sorted on P

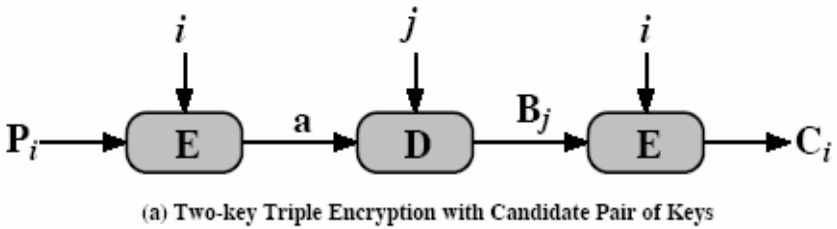


(c) Table of intermediate values and candidate keys

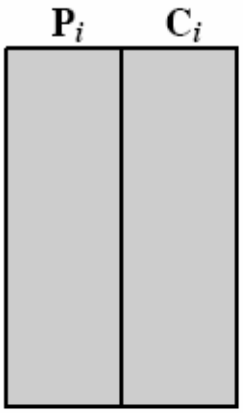
Schema di attacco (2/3)

- Per tutte le 2^{56} chiavi K_j=j calcolare $B_j=D_j[a]$. Se si trova una corrispondenza nella tabella (c) si è trovata una coppia di chiavi candidate che produce una coppia nota (P,C).

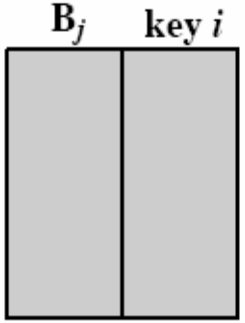
- Verificare le coppie di chiavi candidate su altre coppie testo in chiaro/testo cifrato. Se necessario, ripetere la procedura con un nuovo valore di a.



(a) Two-key Triple Encryption with Candidate Pair of Keys



(b) Table of n known plaintext-ciphertext pairs, sorted on P



(c) Table of intermediate values and candidate keys

Schema di attacco (3/3)

- Date n coppie (P,C), la probabilità di successo per un singolo valore selezionato di A=a è $n/2^{64}$.
- Dalla teoria della probabilità è noto che il numero medio di tentativi per estrarre una pallina rossa da un'urna contenete n palline rosse e (N-n) verdi è (N+1)/(n+1).
- Pertanto il tempo si esecuzione dell'algoritmo è dell'ordine di

$$2^{56}\,\frac{2^{64}+1}{n+1} \approx 2^{56}\,\frac{2^{64}}{n} = 2^{120-\log_2 n}$$

Triple DES con tre chiavi

- La sicurezza di Triple DES può essere ulteriormente incrementata usando chiavi crittografiche distinte per ciascuna fase:

$$C = E_{K_3}[D_{K_2}[E_{K_1}[P]]]$$

- La compatibilità con DES si ha ponendo $K_3=K_2$ o $K_2=K_1$.

- Molte applicazioni per internet e per la gestione della posta elettronica hanno adottato Triple DES a tre chiavi (PGP, S/MIME, etc.).

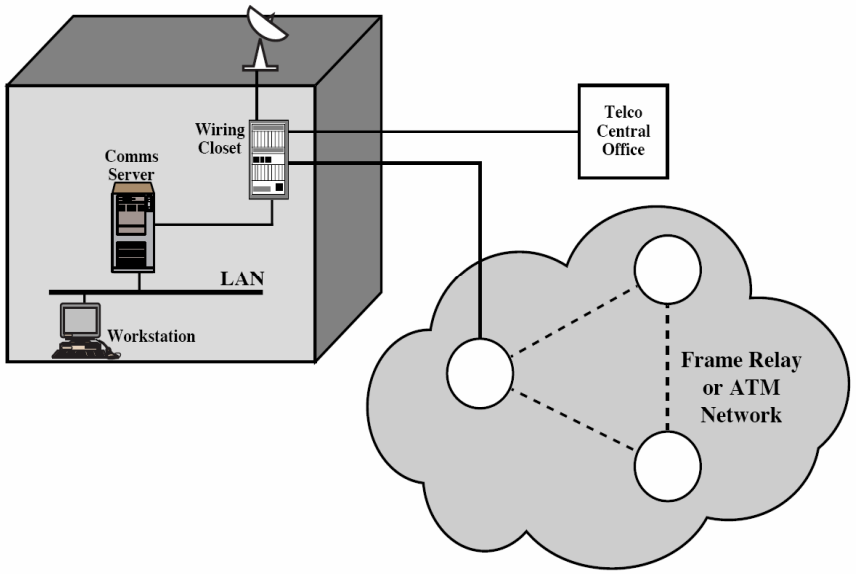
Segretezza e crittografia simmetrica

Sommario

- Potenziali punti di attacco
- Posizionamento della logica crittografica
  - Crittografia del collegamento
  - Crittografia punto-a-punto
- Mascheramento del traffico
- Distribuzione della chiave
- Cenni sulla generazione di numeri casuali

Scenario tipico

- Workstation e Server connessi fra loro medianti reti LAN;
- Reti LAN di uno stesso edificio connesse fra loro mediante switch e router;
- Reti LAN geograficamente separate interconnesse mediante reti pubbliche a commutazione di pacchetto.



Possibili attacchi alla segretezza

- Analisi del traffico all'interno della rete locale (dall'interno o dall'esterno).
- Attacco all'armadio di cablaggio e/o ad uno qualsiasi dei collegamento fisici della rete.
- Analisi e/o alterazione dei pacchetti in transito nei nodi di commutazione delle reti di interconnessione pubbliche.

Tipi di attacco

- Attacco attivo
  - L'intruso deve assumere il controllo fisico di una porzione di collegamento (in genere è necessario far uso di apparecchiature invasive).
  - L'intruso è in grado sia di ascoltare che di modificare le trasmissioni.
- Attacco passivo
  - L'intruso non ha il controllo fisico del canale, ma è in grado di ascoltare le trasmissioni (utilizzando mezzi non invasisi).

Crittografia del collegamento

- Ciascun collegamento fisico vulnerabile viene dotato di un dispositivo di crittografia.

- Vantaggi:
  - Tutto il traffico su tutti i collegamenti è sicuro
  - Ogni coppia di nodi ha una chiave univoca usata per tutti i messaggi in transito
  - Facilità nella distribuzione delle chiavi

- Svantaggi:
  - Il messaggio è decrittografato (e dunque vulnerabile) negli switch
  - L'utente non controlla la sicurezza dei nodi della rete pubblica
  - Tutti i collegamenti devono essere crittografati
  - Assenza di autenticazione

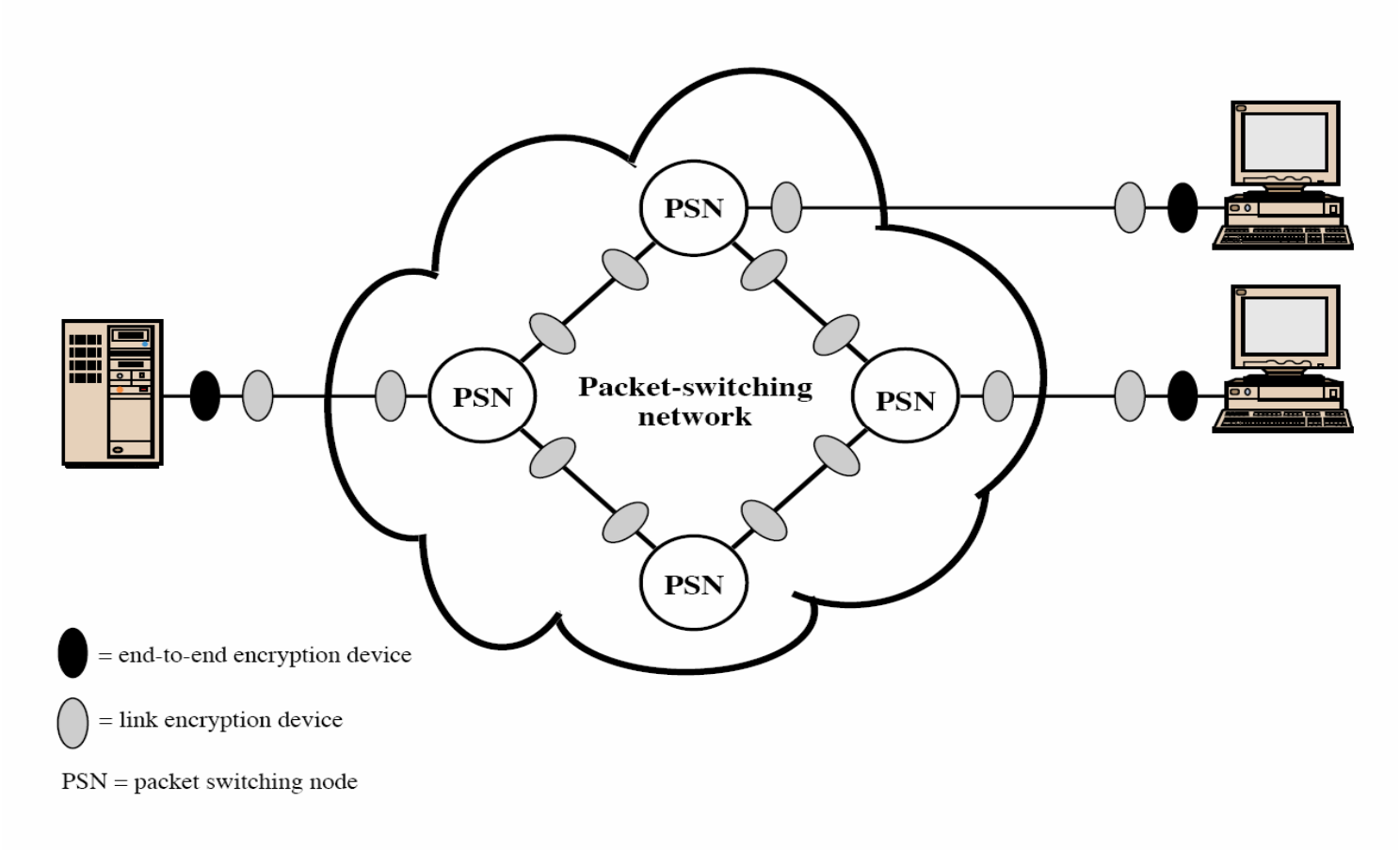
Crittografia punto-a-punto

- Il processo di crittografia viene eseguito solo dai due sistemi terminali, ed è completamente trasparente ai nodi interni della rete di interconnessione.

- Vantaggi:
  - Dati protetti contro gli attacchi ai collegamenti o agli switch
  - Garantisce l'autenticazione

- Svantaggi:
  - La distribuzione delle chiavi è più complessa
  - L'indirizzo del destinatario non può essere crittografato

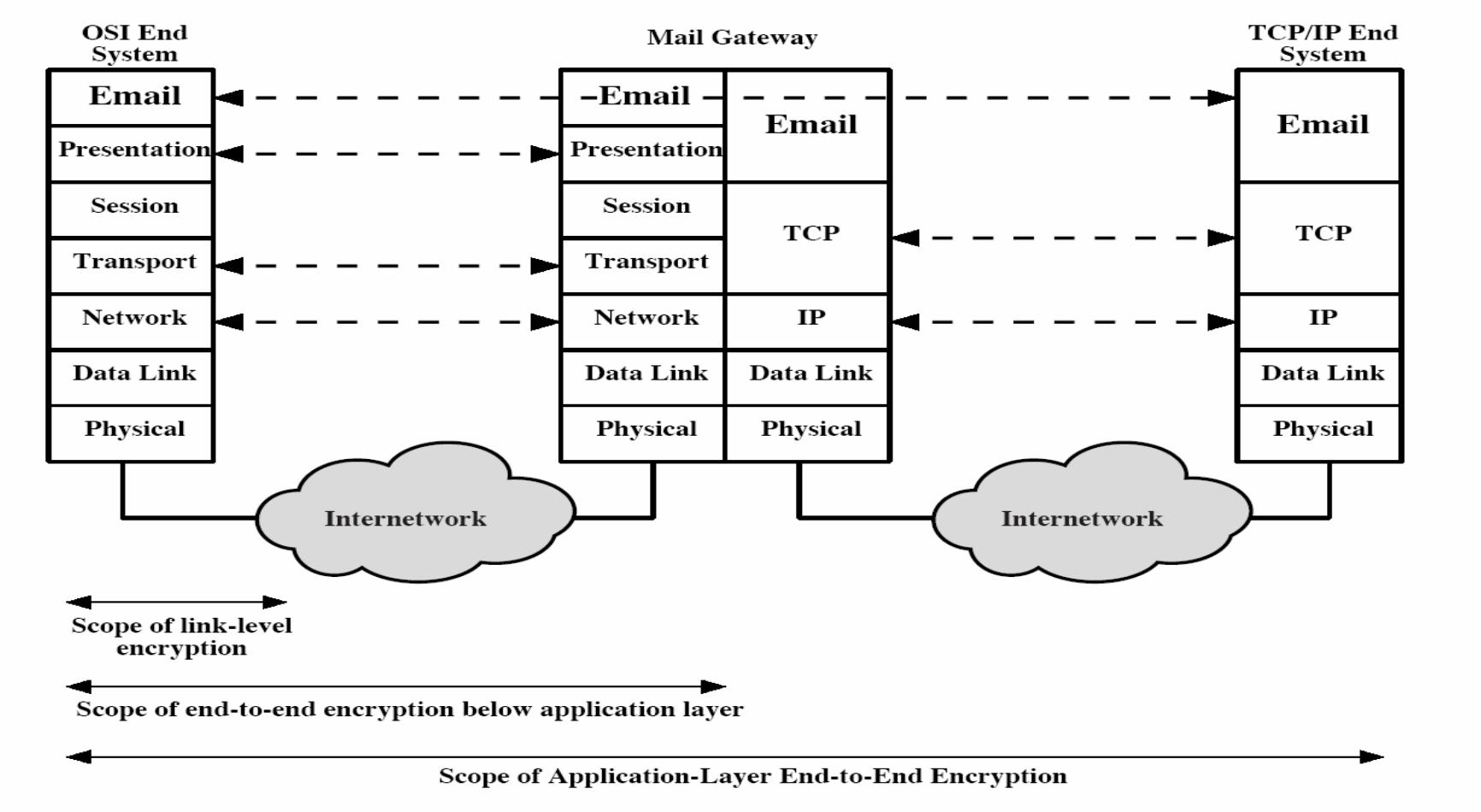
Crittografia del collegamento e punto-a-punto



Posizionamento logico della crittografia

- Crittografia del collegamento: livello fisico (1) o di collegamento (2) del modello OSI (Open System Interconnection).
- Crittografia punto-a-punto: livelli 3-7 del modello OSI.
- Spostandoci verso l'alto nella gerarchia dei livelli di comunicazione vengono crittografate meno informazioni, ma si ottiene una maggiore sicurezza sui dati. Inoltre, aumenta il numero di chiavi crittografiche da utilizzare.

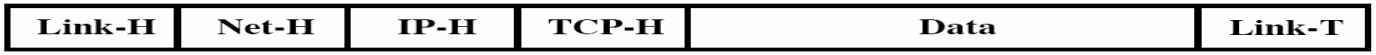
Posizionamento logico della crittografia



Posizionamento logico della crittografia

(a) Application-Level Encryption (on links and at routers and gateways)

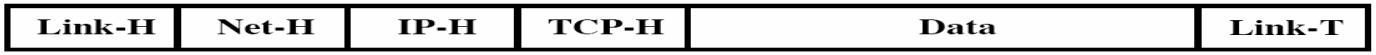
On links and at routers



In gateways

(b) TCP-Level Encryption

On links



In routers and gateways

(c) Link-Level Encryption

Shading indicates encryption.

TCP-H	=	TCP header
IP-H	=	IP header
Net-H	=	Network-level header (e.g., X.25 packet header, LLC header)
Link-H	=	Data link control protocol header
Link-T	=	Data link control protocol trailer

Utilizzo delle chiavi

- **Crittografia del collegamento:** una singola chiave è utilizzata per crittografare tutti i dati in transito nel collegamento.

- **Crittografia punto-a-punto a livello di rete:** tutti i processi utenti e le applicazioni di un dato terminale T1 condividono la stessa chiave per comunicare con un altro terminale T2.

- **Crittografia punto-a-punto a livello dell'applicazione:** ogni processo utente o applicazione di un dato terminale T1 usa una propria chiave per comunicare con il corrispondente processo utente o applicazione del terminale T2

Analisi del traffico

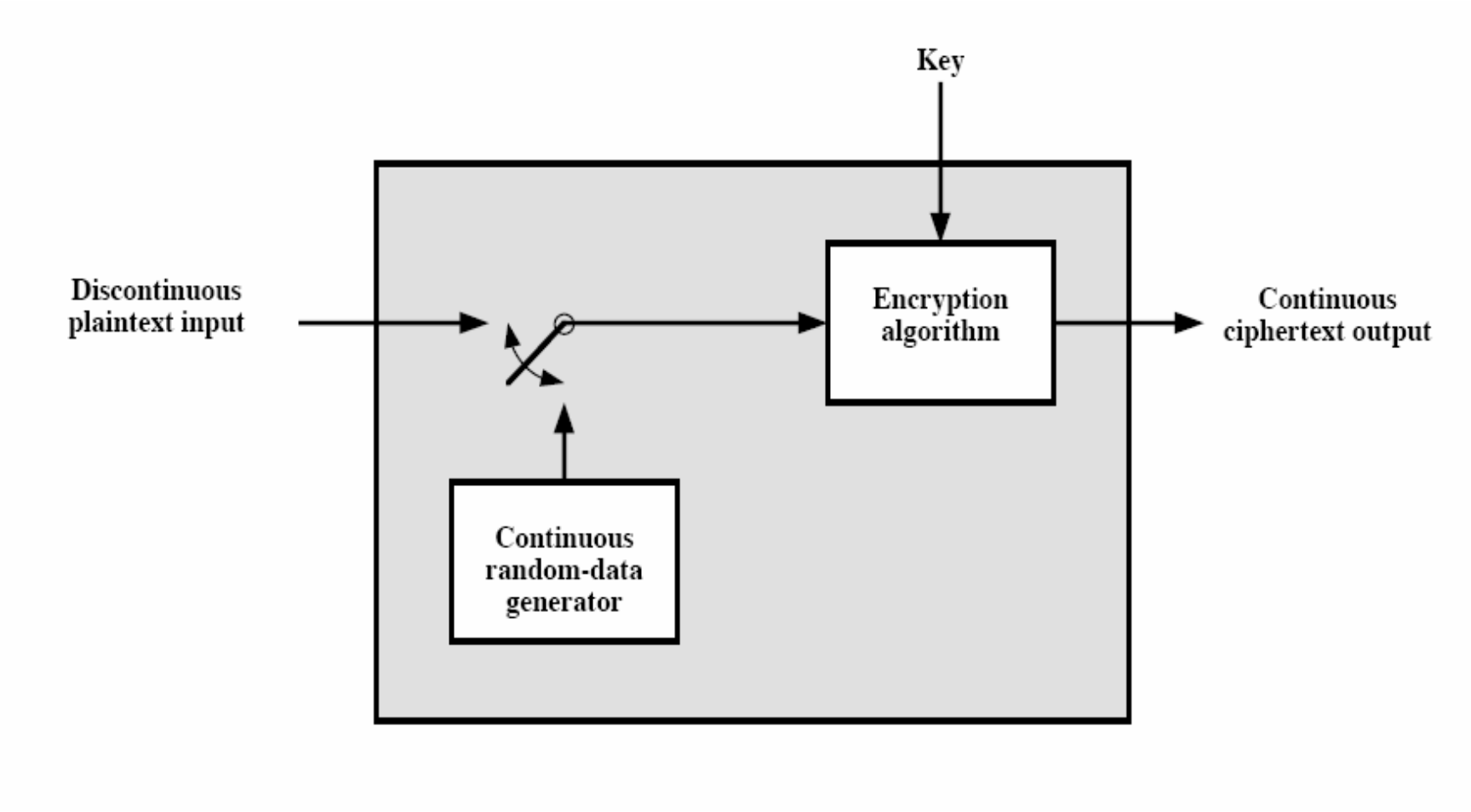
- L'analisi del traffico può fornire informazioni su:
  - Identità dei partner
  - Frequenza di comunicazione dei partner
  - Lunghezza e/o quantità dei messaggi scambiati
  - Schema di trasmissione dei messaggi
- Schemi di traffico segreti possono essere impiegati per creare canali nascosti (ad esempio, pacchetto lungo = 0, pacchetto corto = 1).
- L'analisi del traffico è usata in ambito militare e commerciale.

Segretezza del traffico

- La **crittografia punto-a-punto** è estremamente vulnerabile all'analisi del traffico (essendo le intestazione dei pacchetti in chiaro). Contromisure:
  - inserimento di pacchetti nulli
  - pacchetti in uscita di uguale dimensione (maschera il volume di dati scambiato)
  - crittografia del collegamento (maschera gli indirizzi finali)

- La **crittografia del collegamento** può essere resa più sicura utilizzando la tecnica **Traffic Padding:** in assenza di testo in chiaro viene comunque prodotto un output cifrato in maniera casuale.

Traffic Padding



Distribuzione della chiave

- Dati due terminali A e B, si possono avere varie alternative per la distribuzione delle chiavi.

1. A sceglie una chiave e la consegna fisicamente a B;
2. Una parte terza C sceglie una chiave e la consegna fisicamente ad A e B;
3. Se A e B hanno usato recentemente una chiave, una delle parti trasmette all'altra parte la nuova chiave usando la vecchia chiave;
4. Se A e B hanno una connessione crittografata sicura con C, quest'ultima può consegnare una chiave sui collegamenti crittografati ad A e B;

Distribuzione della chiave

- Le opzioni 1 e 2 richiedono la consegna manuale della chiave: sono ragionevoli solo per la crittografia di un collegamento.

- L'opzione 3 è poco sicura: se una chiave viene violata, anche le chiavi successive saranno automaticamente violate.

- Per la crittografia punto-a-punto sono general-mente adottate delle varianti dell'opzione 4.

Centro di Distribuzione delle Chiavi (KDC)

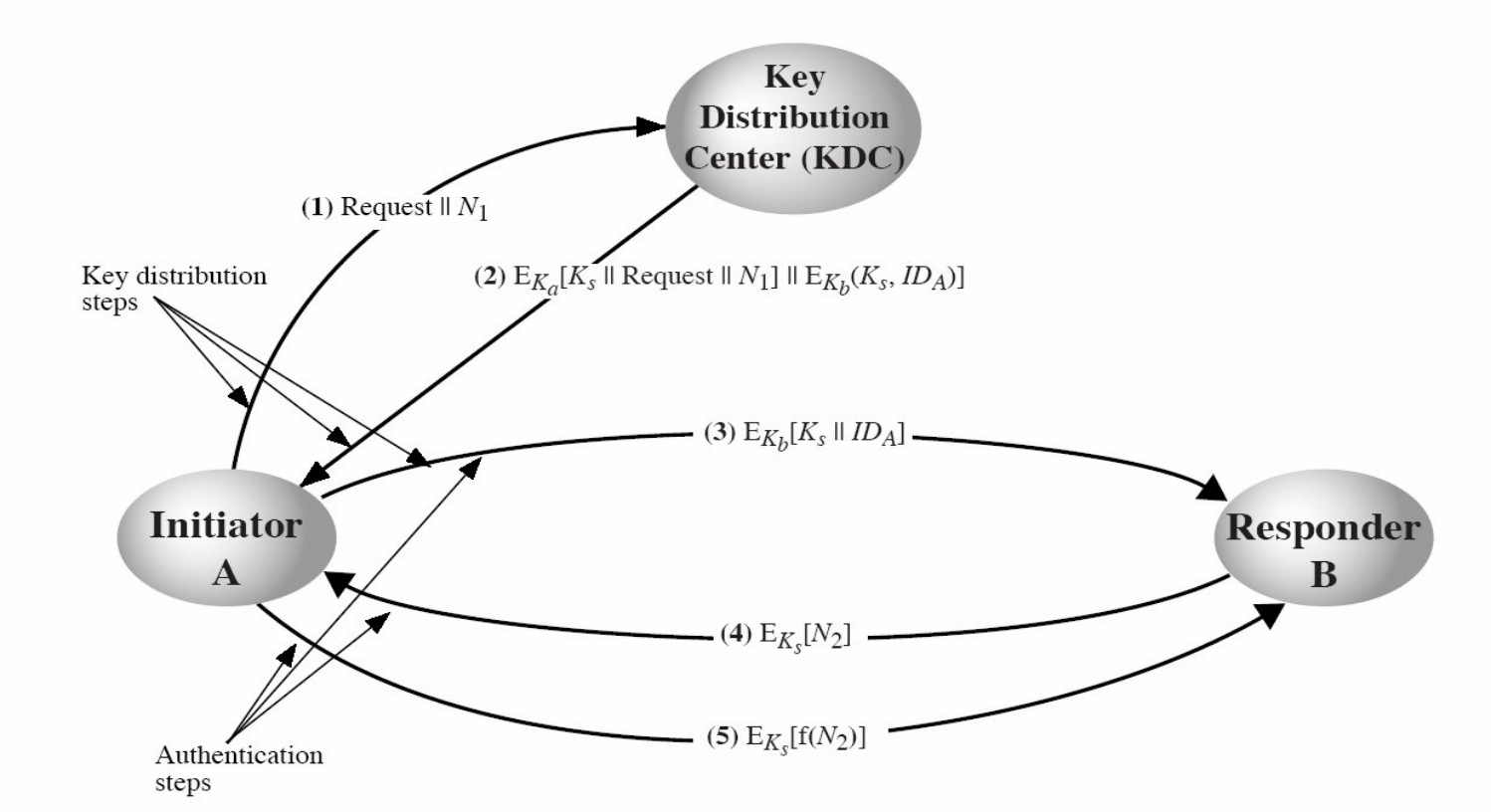
- Per la crittografia punto-a-punto si assume che ci sia un centro responsabile della distribuzione delle chiavi alle coppie di utenti in base alle richieste.

- Sono utilizzati due livelli gerarchici di chiavi
  - Chiavi master: condivise fra ciascun utente e il KDC
  - Chiavi di sessione: utilizzate per la comunicazione fra due utenti

- Le chiave di sessione vengono trasmesse alle coppie di utenti in forma crittografata usando le chiavi master.

- Il problema della distribuzione delle chiavi è semplificato: se ci sono N terminali andranno consegnate fisicamente solamente N chiavi master.

Esempio di distribuzione della chiave di sessione



- K_A → chiave master di A (nota solo ad A ed al KDC)
- K_B → chiave master di B (nota solo ad B ed al KDC)

Controllo gerarchico della chiave

- In reti di grandi dimensioni si possono definire centri gerarchici di distribuzione delle chiavi.
  - Ogni centro è responsabile di un piccolo dominio
  - Per comunicazioni fra terminali dello stesso dominio, le richieste vengono gestite dal KDC locale
  - Per comunicazione fra terminali in domini differenti, i KDC locali comunicano con un KDC globale per coordinare la scelta della chiave
- Vantaggi: maggiore robustezza ai guasti e semplicità nella distribuzione delle chiavi master.

Durata di una chiave di sessione

- Due esigenze contrastanti:
  - Maggiore è la frequenza con cui si cambiano le chiavi di sessione, maggiore sarà la sicurezza delle chiavi.
  - La distribuzione delle chiavi introduce un overhead iniziale ad ogni comunicazione, introducendo ritardi di trasmissioni. Inoltre, genera traffico nella rete.

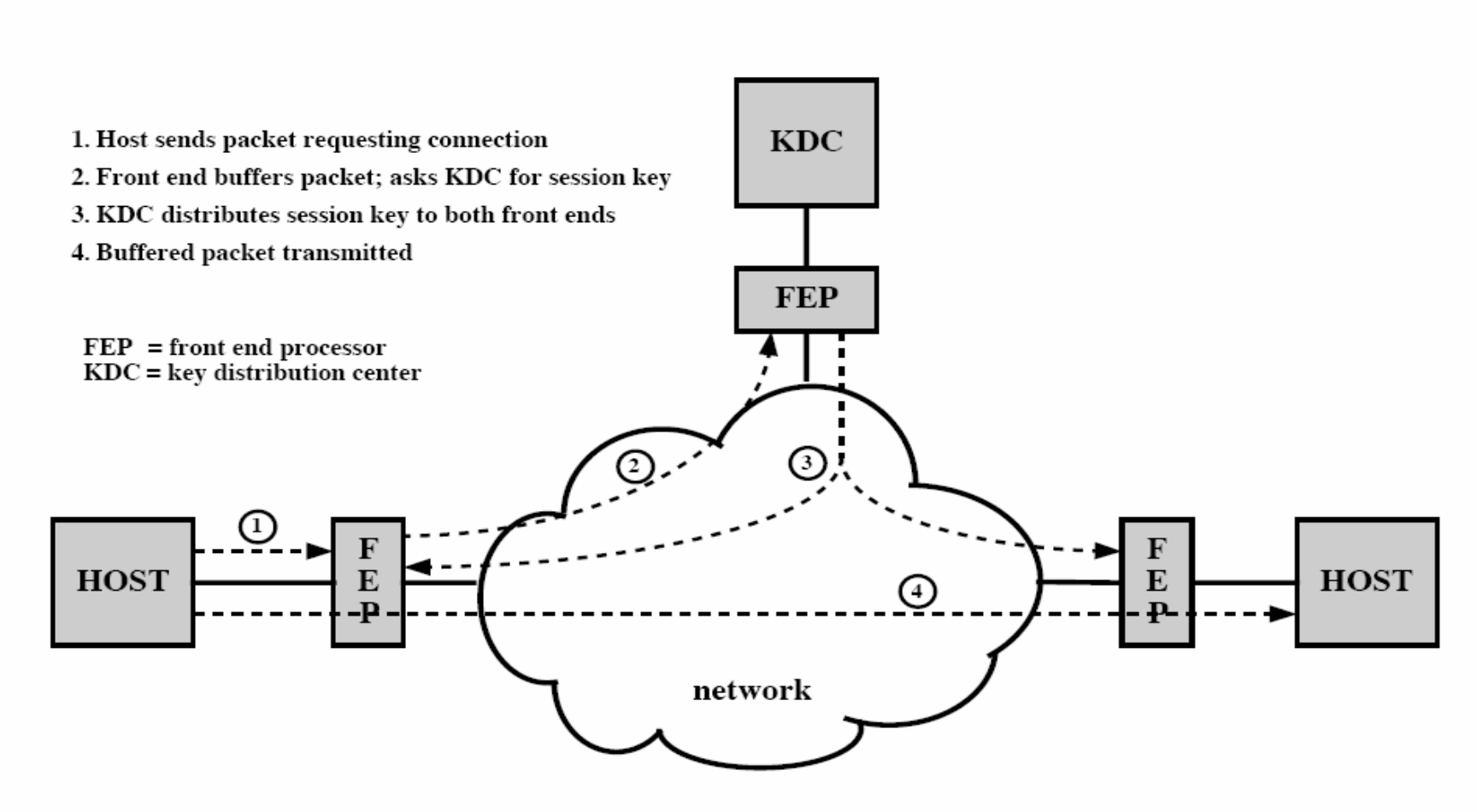
- Regola pratica: vanno stabiliti **tempi massimi** di durata e/o **numero massimo di transazioni** effettuabili con una singola chiave di sessione.

Schema a controllo trasparente della chiave

- Per crittografie punto-a-punto a livello di rete orientate alla connessione, le operazioni di richiesta della chiave di sessione e di crittografia del messaggio possono essere delegate ad un processore esterno denominato FEP (Front-End-Processor).

- Con quest'approccio l'operazione di crittografia diviene trasparente ai singoli terminali:
  - Per il terminale e la rete esterna, il processore FEP appare come un nodo di commutazione

Schema a controllo trasparente della chiave



Controllo della chiave decentralizzato

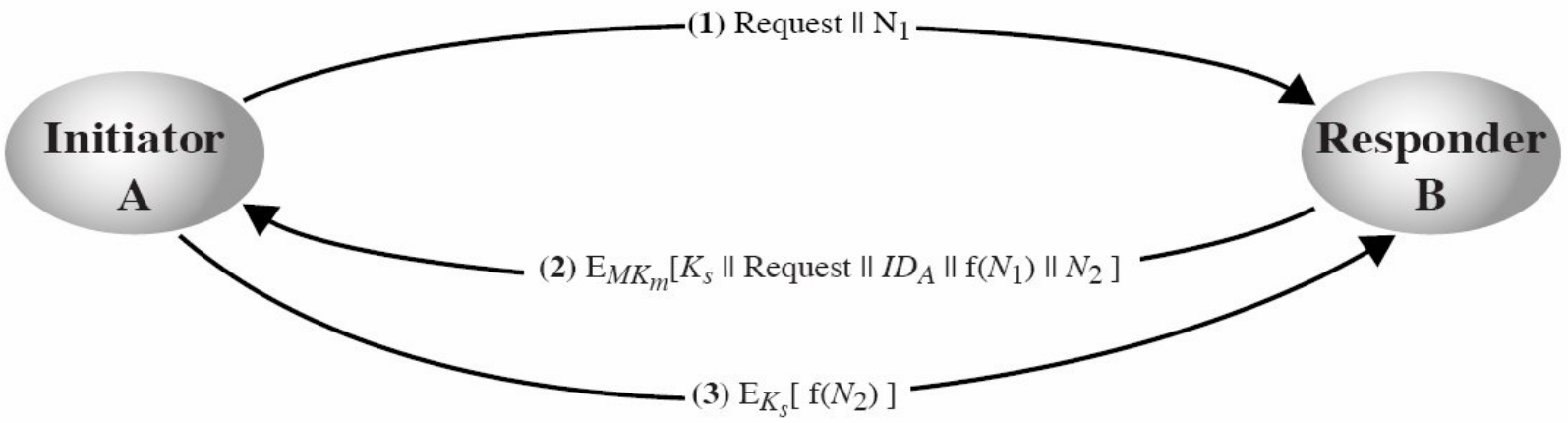
- In uno schema centralizzato, il KDC deve essere fidato e protetto contro ogni tipo di attacco.
  - I rischi derivanti da eventuali attacchi al KDC potrebbero essere eliminati se il processo di distribuzione delle chiavi di sessione fosse decentralizzato.

- Una decentralizzazione completa in reti di grandi dimensioni non è pensabile. Tuttavia, è una valida alternativa in contesti locali.

Controllo della chiave decentralizzato

- Si assume che ci siano n terminali in grado di comunicare in modo univoco fra loro.
  - Sono richieste n(n-1)/2 chiavi master (la distribuzione può avvenire di persona dato che queste chiavi non verranno più cambiate).

- Le chiavi di sessione vengono ottenute come segue:



Controllo dell'utilizzo delle chiavi

- Il concetto di gerarchia delle chiavi e l'uso di tecniche automatiche di distribuzione delle chiavi riduce notevolmente il numero di chiavi da distribuire manualmente.
- Per un corretto uso delle chiavi sessione distribuite dai KDC, può essere utile specificare la loro funzione:
  - Chiavi di sessione / Chiavi master
  - Chiavi di crittografia per dati
  - Chiavi di crittografia per codici PIN
  - Chiavi per crittografia di file
  - …

Esempio di controllo dell'utilizzo delle chiavi

- DES accetta in ingresso una chiave a 64 bit. Tuttavia, solo 56 bit vengono utilizzati per produrre le sottochiavi. I restanti 8 bit hanno il seguente significato:
  - Un bit indica se la chiave è di sessione o master
  - Un bit indica se la chiave può essere utilizzata per la crittografia
  - Un bit indica se la chiave può essere utilizzata per la decrittografia
  - I bit rimanenti sono riservati per utilizzi futuri

Generazione di numeri casuali

- I numeri casuali vengono utilizzati in numerosi algoritmi crittografici:
  - Schemi di autenticazione
  - Generazione chiave di sessione
  - Generazione chiavi per l'algoritmo di crittografia RSA a chiave pubblica (che verrà studiato in seguito)
- Idealmente gli elementi di una sequenza casuale sono statisticamente indipendenti fra loro (ovvero, imprevedibili) ed uniformemente distribuiti.

Generazione di numeri casuali

- Trovare sorgenti di numeri effettivamente casuali è difficile.
  - I generatori fisici di rumore sono buone sorgenti, ma sono poco pratici da usare.

- Un'alternativa è usare algoritmi numerici deterministici per la produzione di sequenze di numeri pseudo-casuali.
  - Se l'algoritmo è di buona qualità, le sequenza prodotte passeranno numerosi test ragionevoli di casualità

Metodo della congruenza lineare

$$X_{n+1} = (aX_n + c) \bmod m$$

m	il modulo	$m > 0$
a	il moltiplicatore	$0 < a < m$
c	l'incremento	$0 \le c \le m$
X_0	il valore iniziale o seme	$0 \le X_0 \le m$

Metodo della congruenza lineare

- La scelta dei valori di *a*, *c*, e *m* è fondamentale per ottenere buone sequenza di numeri pseudo-casuali.

- Si vorrebbe che *m* fosse molto esteso in modo da produrre una lunga sequenza di numeri casuali distinti
  - *m* è scelto pari al massimo intero non-negativo rappresentabile

- Si dimostra che una buona scelta è a=16807, c=0 e $m=2^{31}-1$
  - La sequenza che si ottiene ha un periodo pari ad *m*

Metodo della congruenza lineare

- Una volta fissati il seme ed i parametri *a* e *c*, la sequenza risultante di numeri segue in maniera deterministica.
- Nell'algoritmo non vi è nulla di casuale, se non la scelta dei parametri iniziali.
- La sequenza di numeri così generata può essere ricostruita da un estraneo che è a conoscenza dell'algoritmo e dei parametri usati.
  - Per evitare tale problema la sequenza viene periodicamente riavviata usando come seme il valore dell'orologio interno del calcolatore.

Algoritmo AES (Advanced Encryption Standard)

Origini di AES

- 3DES non rappresenta un buon candidato nel lungo periodo:
  - Esecuzione software lenta (DES era stato progettato agli inizi degli anni ’70 per un’implementazione hardware)
  - Usa blocchi di 64 bit di dati (per motivi di efficienza e sicurezza sarebbe preferibile utilizzare blocchi di lunghezza maggiore)
- Nel 1997 il NIST emette una richiesta di proposte per un nuovo standard chiamato (AES).
- Il nuovo standard dovrà progressivamente sostituire 3DES.

Origini di AES

- Requisiti minimi richiesti:
  - Doti di sicurezza uguali o maggiori a 3DES
  - Elevata efficienza computazionale
  - Blocchi dati di 128 bit / Lunghezza chiavi: 128, 192, 256 bit
  - Specifiche di progetto di pubblico dominio
  - Vita media 20-30 anni
- Inizialmente vengono accettate 15 delle 21 proposte.
- Successivamente i candidati sono ristretti a 5.
- Standard finale (FIPS PUB 197) pubblicato nel 2001. L'algoritmo selezionato è 'RIJNDAEL', sviluppato dai crittografi belgi Dr. Joan Daemen e Dr. Vincent Rijmen.

Criteri iniziali di valutazione di AES

- **Sicurezza:** Scartati gli attacchi a forza bruta, viene valutato l'impegno necessario per un analisi crittografica dell'algoritmo.

- **Costo:** algoritmo disponibile a tutti, ed utilizzabile in un ampia gamma di applicazioni; criteri di valutazione sono anche l'efficienza computazionale e i requisiti di memoria.

- **Caratteristiche dell'algoritmo e dell'implementazione:** flessibilità, possibilità di usare chiavi e blocchi di dati di varie dimensioni, possibilità di impiego in varie implementazioni hardware e software, semplicità di analisi

AES: caratteristiche

- **Sicurezza Generale:** basato su un'analisi pubblica della sicurezza condotta dalla comunità dei crittografi (durata 3 anni).
- **Implementazioni software:** velocità di esecuzione, prestazioni su piattaforme differenti, velocità in funzione della lunghezza della chiave.
- **Ambienti con spazio limitato:** occupazione di memoria
- **Implementazione hardware:** costi, velocità di esecuzione e dimensioni dei chip.
- **Attacchi alle implementazioni:** resistenza ad attacchi che misurano il tempo di esecuzione e/o la potenza consumata.

AES: Caratteristiche

- **Crittografia e Decrittografia:** differenze fra gli algoritmi di crittografia e decrittografia.
- **Agilità della chiave:** tempo richiesto per impostare una nuova chiave, calcolo delle sottochiavi
- **Flessibilità:** possibilità di variare e/o ottimizzare i parametri dell'algoritmo in base al tipo di applicazione richiesta.
- **Parallelismo:** capacità di sfruttare le funzionalità di architetture multi-processore.

Cifratura AES

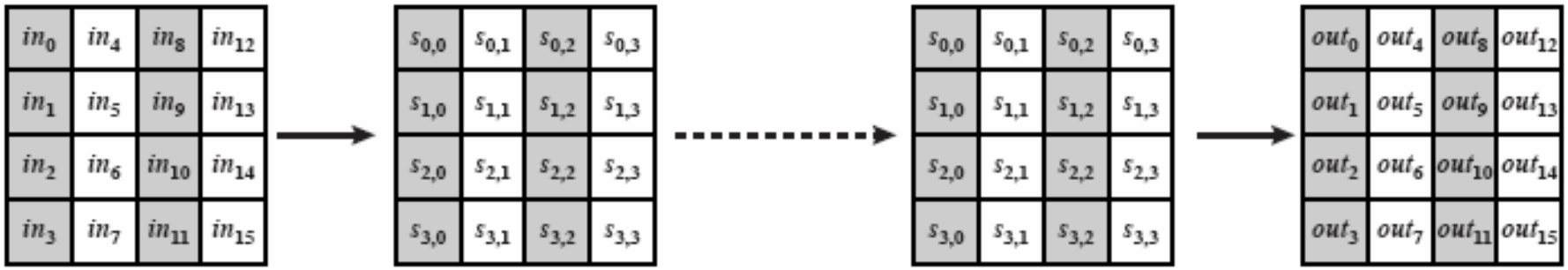
- Criteri progettuali dell'algoritmo RIJNDAEL:
  - Resistenza contro attacchi noti
  - Velocità e compattezza del codice
  - Semplicità
- Blocco dati: 128 bit / Lunghezza della chiave: qualunque multiplo di 32 bit.
  - In pratica si usano chiavi di **128**, 192 e 256 bit
- I parametri dell'algoritmo dipendono dalla lunghezza della chiave (vedi lucido seguente).
- Nel seguito si considera una chiave a 128 bit.

Parametri Algoritmo

Tabella 5.3 I parametri di AES.

Dimensioni della chiave (word/byte/bit)	4/16/128	6/24/192	8/32/256
Dimensioni del blocco di testo in chiaro (word/byte/bit)	4/16/128	4/16/128	4/16/128
Numero di fasi	10	12	14
Dimensioni della chiave di fase (word/byte/bit)	4/16/128	4/16/128	4/16/128
Dimensioni espanse della chiave (word/byte)	44/176	52/208	60/240

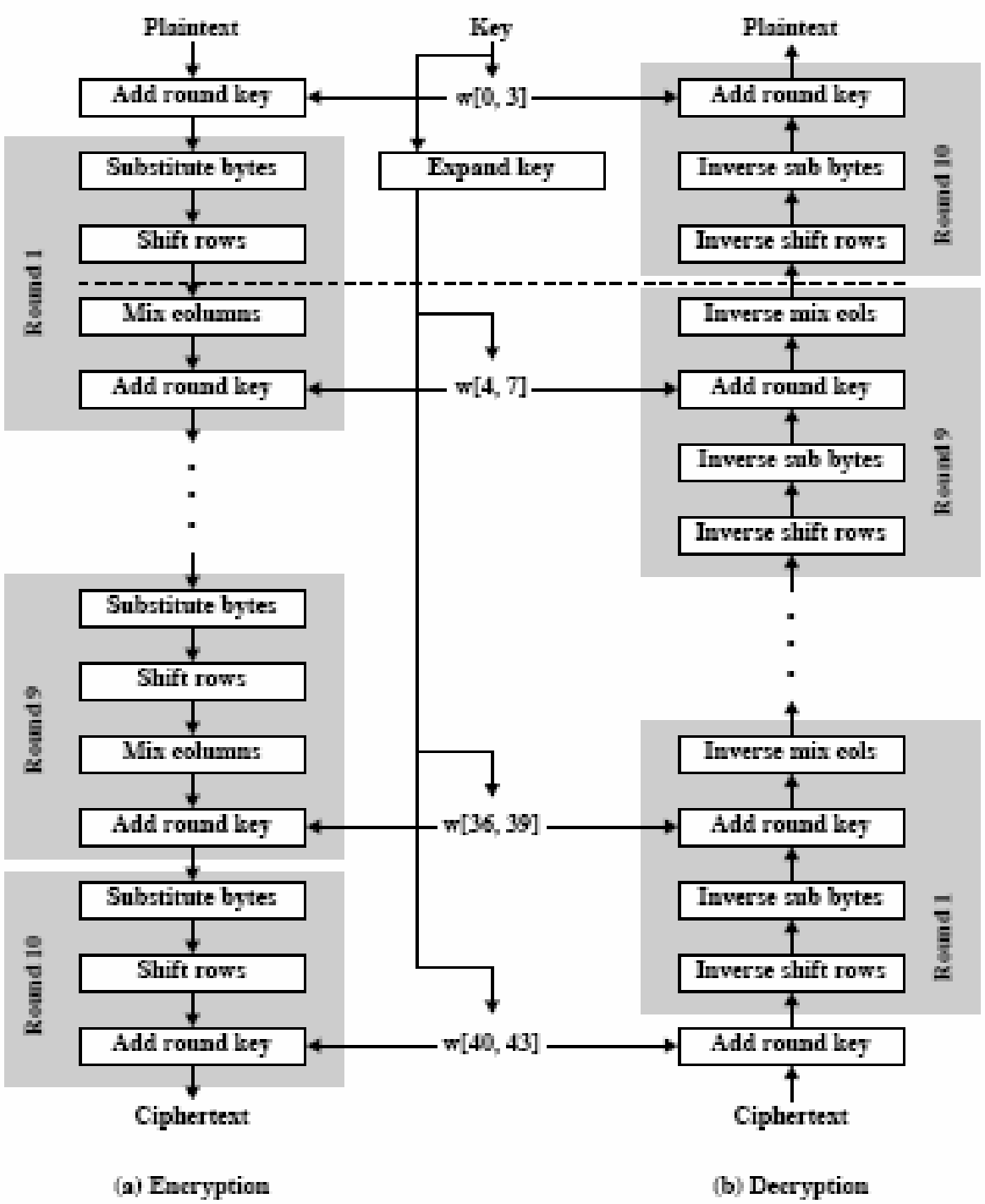
Dati Input Algoritmo AES



(a) Input, state array, and output

(b) Key and expanded key

Schema Algoritmo AES



Commenti su AES

- Non si tratta di una struttura di Feistel.
- La chiave viene espansa in 44 word da 32 bit. Ciascuna fase usa 4 word distinte.
- Operazioni effettuate
  - **Substitute Bytes:** blocchi di 8 bit sono sostituiti mediante una S-box.
  - **Shift Rows:** semplice permutazione.
  - **Mix Columns:** sostituzione che usa l'aritmetica su GF(2^8), con il polinomio irriducibile $m(x)=x^8+x^4+x^3+x+1$.
  - **Add Round Key:** XOR bit-a-bit con la sottochiave.

Commenti su AES

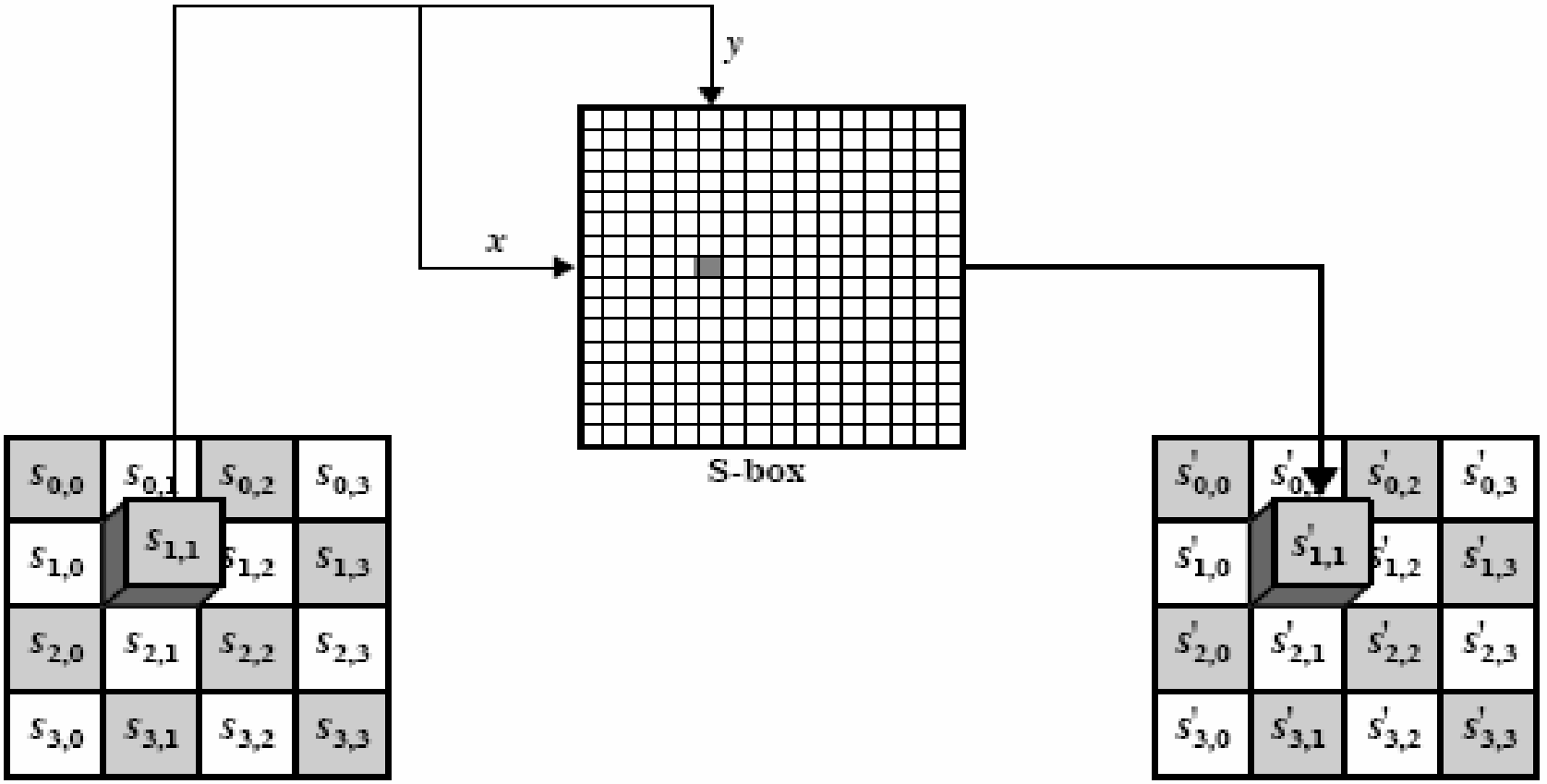
- Tutte le trasformazioni sono invertibili.
- L'algoritmo di decrittografia usa le sottochiavi in ordine inverso.
- L'algoritmo di decrittografia non è identico a quello di crittografia.
- L'ultima fase della crittografia e della decrittografia contiene solo tre trasformazioni.

Trasformazione Substitute Bytes (diretta)

- Viene definita una matrice 16x16 di byte detta S-box

- Ogni byte in ingresso viene mappato in un nuovo byte nel seguente modo:
  - I primi quattro bit indicano la riga e i rimanenti 4 bit la colonna
  - I valori di riga e colonna fungono da indici per la S-box

Trasformazione Substitute Bytes (diretta)



(a) Substitute byte transformation

Trasformazione Substitute Bytes (diretta)

A. S-Box

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

Esempio

EA	04	65	85
83	45	5D	96
5C	33	98	B0
F0	2D	AD	C5

SubBytes →

87	F2	4D	97
EC	6E	4C	90
4A	C3	46	E7
8C	D8	95	A6

$$(1110\,1010) \xrightarrow{bin/esa} EA \xrightarrow{SubBytes} 87 \xrightarrow{esa/bin} (1000\ 0111)$$

$$(0000\ 0100) \xrightarrow{bin/esa} 04 \xrightarrow{SubBytes} F2 \xrightarrow{esa/bin} (1111\ 0010)$$

$$(0110\ 0101) \xrightarrow{bin/esa} 65 \xrightarrow{SubBytes} 4D \xrightarrow{esa/bin} (0100\ 1101)$$

$$(1000\ 0101) \xrightarrow{bin/esa} 85 \xrightarrow{SubBytes} 97 \xrightarrow{esa/bin} (1001\ 0111)$$

Progetto della S-box

- Inizializzare la S-box con il valore dei byte in sequenza ascendente. La prima riga contiene {00},{01},…,{0F}.
- Associate a ciascun byte della S-box il suo inverso moltiplicativo su GF(2^8), con il polinomio irriducibile $m(x)=x^8+x^4+x^3+x+1$; {00} è mappato su se stesso.
- Considerare che ciascun byte della S-box è costituito da 8 bit ($b_7, b_6,\ldots, b_0$). Applicare la trasformazione:

$$b'_i = b_i \oplus b_{(i+4)\bmod 8} \oplus b_{(i+5)\bmod 8} \oplus b_{(i+6)\bmod 8} \oplus b_{(i+7)\bmod 8} \oplus c_i$$

$$\text{con} \quad \mathbf{c} = (c_7, c_6, c_5, c_4, c_3, c_2, c_1, c_0) = (0,1,1,0,0,0,1,1)$$

Progetto della S-box

- Progettata per resistere a tutti gli attacchi noti ad analisi crittografica.
- Bassa correlazione fra i bit in ingresso e in uscita (l'output non è una semplice funzione matematica dell'input).
- La costante additiva ***c*** è stata scelta in modo da non avere punti fissi (S-box(*a*)=*a*) e punti fissi opposti (S-box(*a*)=neg(*a)*).

Trasformazione Substitute Bytes (inversa)

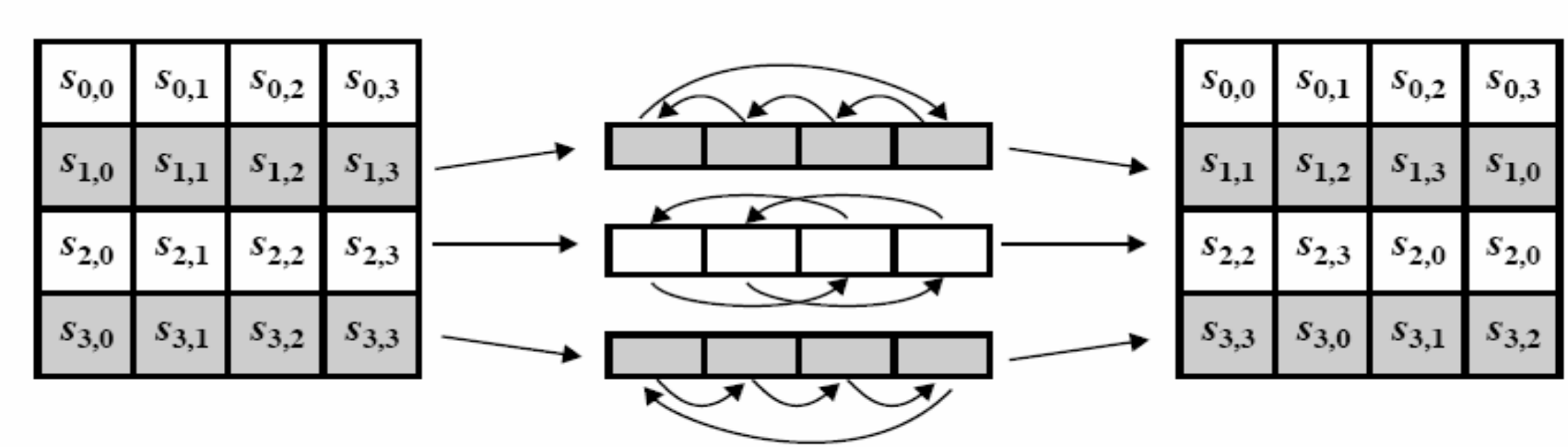
B. S-Box inversa

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

Trasformazione Shift Rows

- **Diretta:** Riga 1 non viene modificata; Riga 2 scorrimento circolare a sx di un byte; Riga 3 scorrimento circolare a sx di 2 byte; Riga 4 scorrimento circolare a sx di 3 byte.

- **Inversa:** gli scorrimenti sono effettuati verso destra.



(a) Shift row transformation

Esempio

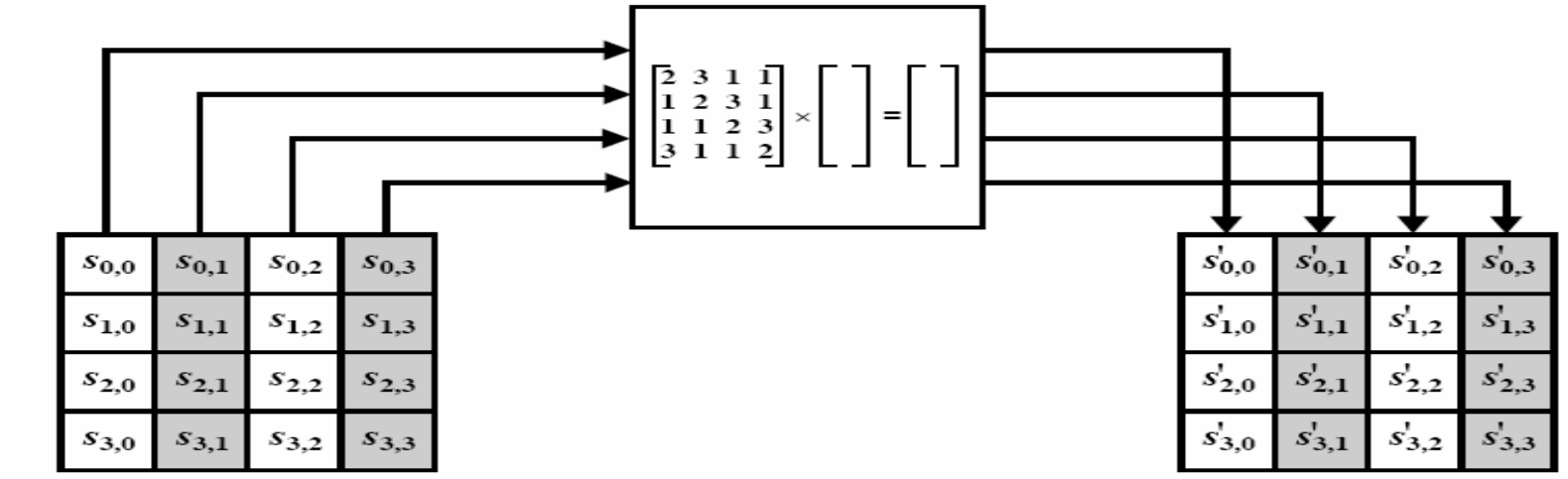
87	F2	4D	97
EC	6E	4C	90
4A	C3	46	E7
8C	D8	95	A6

Shift Rows

87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

Trasformazione Mix Columns (Diretta)

- Opera su una singola colonna: ciascun byte di una colonna viene mappato in un nuovo valore funzione dei 4 byte presenti nella colonna.
- Somme e moltiplicazioni sono eseguite secondo l'aritmetica di GF(2^8), con $m(x)=x^8+x^4+x^3+x+1$.



(b) Mix column transformation

Aritmetica su GF(2^8)

- La **somma** di due polinomi in GF(2^8) corrisponde all'operazione di XOR bit-a-bit.

ESEMPIO

notazione polinomiale :

$$(x^6 + x^4 + x^2 + x + 1) + (x^7 + x + 1) = x^7 + x^6 + x^4 + x^2$$

notazione binaria :

$$(01010111) \oplus (10000011) = (11010100)$$

notazione esadecimale :

$$\{57\} \oplus \{83\} = \{D4\}$$

Aritmetica su GF(2^8)

- La **moltiplicazione** su GF(2^8) può essere difficile da implementare. Tuttavia, utilizzando come polinomio irriducibile ***m(x)=x⁸+x⁴+x³+x+1***, *si ha*

$$x^8 \bmod m(x) = [m(x) - x^8] = (x^4 + x^3 + x + 1)$$

da cui

$$f(x) = b_7 x^7 + b_6 x^6 + b_5 x^5 + b_4 x^4 + b_3 x^3 + b_2 x^2 + b_1 x + b_0$$

$$x \cdot f(x) = (b_7 x^8 + b_6 x^7 + b_5 x^6 + b_4 x^5 + b_3 x^4 + b_2 x^3 + b_1 x^2 + b_0 x) \bmod m(x) =$$

$$= \begin{cases} b_6 x^7 + b_5 x^6 + b_4 x^5 + b_3 x^4 + b_2 x^3 + b_1 x^2 + b_0 x & \text{se } b_7 = 0 \\ (b_6 x^7 + b_5 x^6 + b_4 x^5 + b_3 x^4 + b_2 x^3 + b_1 x^2 + b_0 x) + (x^4 + x^3 + x + 1) & \text{se } b_7 = 1 \end{cases}$$

In bit :

$$x \cdot f(x) = \begin{cases} (b_6, b_5, b_4, b_3, b_2, b_1, b_0, 0) & \text{se } b_7 = 0 \\ (b_6, b_5, b_4, b_3, b_2, b_1, b_0, 0) \oplus (0,0,0,1,1,0,1,1) & \text{se } b_7 = 1 \end{cases}$$

Esempio

87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

MixColumns

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

Esempio (cont.)

$$(\{02\} \bullet \{87\}) \oplus (\{03\} \bullet \{6E\}) \oplus \{46\} \oplus \{A6\} = \{47\}$$

Infatti :

$$\{87\} \xrightarrow{esa/bin} (1000\ 0111), \quad \{6E\} \xrightarrow{esa/bin} (0110\ 1110),$$

$$\{46\} \xrightarrow{esa/bin} (0100\ 0110), \quad \{A6\} \xrightarrow{esa/bin} (1010\ 0110),$$

$$\{47\} \xrightarrow{esa/bin} (0100\ 0111)$$

$$\{02\} \bullet \{87\} \xrightarrow{esa/bin} (0000\ 1110) \oplus (0001\ 1011) = (0001\ 0101)$$

$$\{03\} \bullet \{6E\} \xrightarrow{esa/bin} \{6E\} \oplus (\{02\} \bullet \{6E\}) = (0110\ 1110) \oplus (1101\ 1100)$$

$$= (1011\ 0010)$$

$$(0001\ 0101) \oplus (1011\ 0010) \oplus (0100\ 0110) \oplus (1010\ 0110) =$$

$$(0100\ 0111) \xrightarrow{bin/esa} \{47\}$$

Trasformazione Mix Columns (Inversa)

- Si usa una matrice di trasformazione che è l'inversa moltiplicativa della matrice di trasformazione diretta:

$$\begin{pmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{pmatrix} \begin{pmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{pmatrix} = \begin{pmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{pmatrix}$$

con

$$\begin{pmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{pmatrix} \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

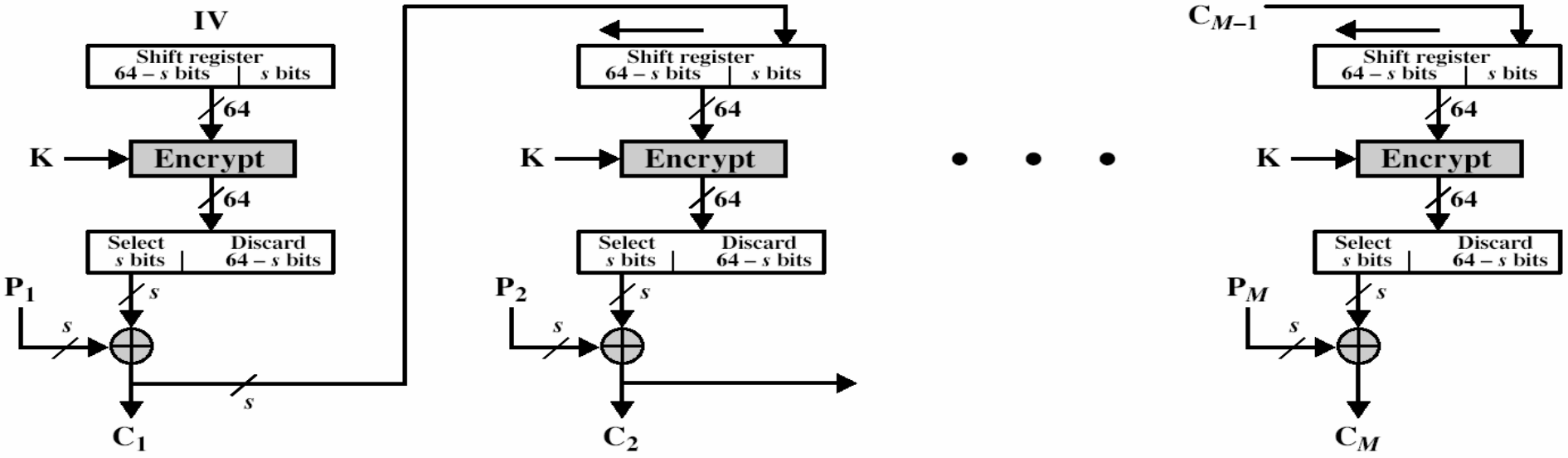
Trasformazione Mix Columns – Commenti generali

- Mix Columns combinata con Shift Rows garantisce che dopo alcune fasi tutti i bit di output dipendano da tutti i bit di input.

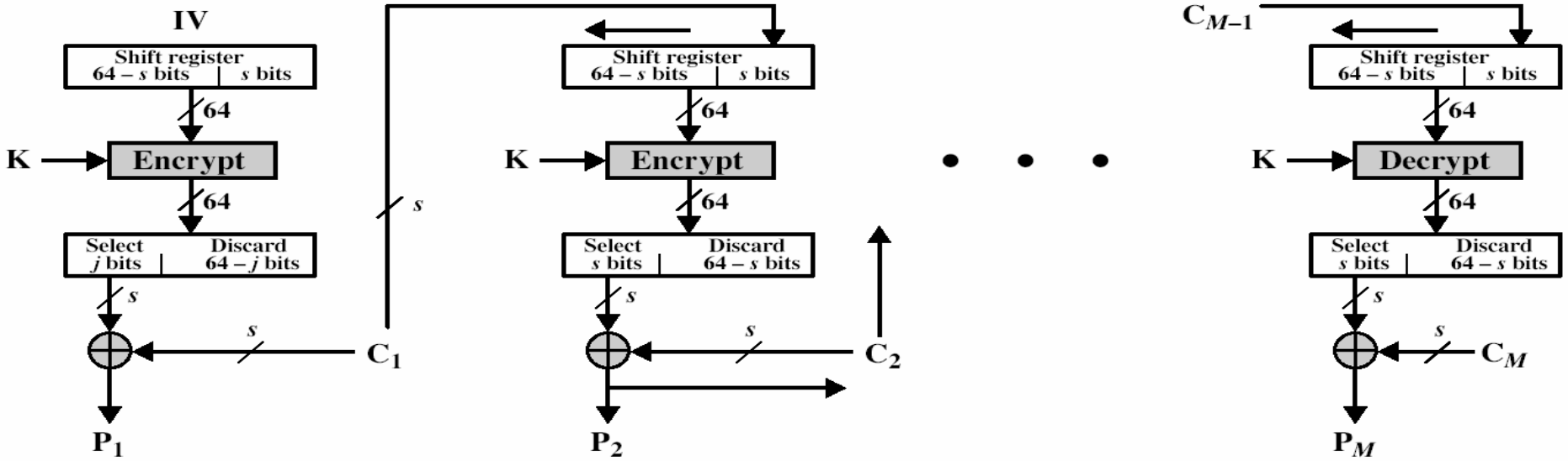
- La scelta dei coefficienti della trasformazione diretta {01}, {02} e {03} semplifica l'implementazione:
  - La moltiplicazione richiede al più uno scorrimento e uno XOR.

- La trasformazione inversa è invece più complessa da implementare.
  - Per le modalità Cipher Feedback e Output Feedback si usa solo la crittografia.

Cipher FeedBack (CFB)

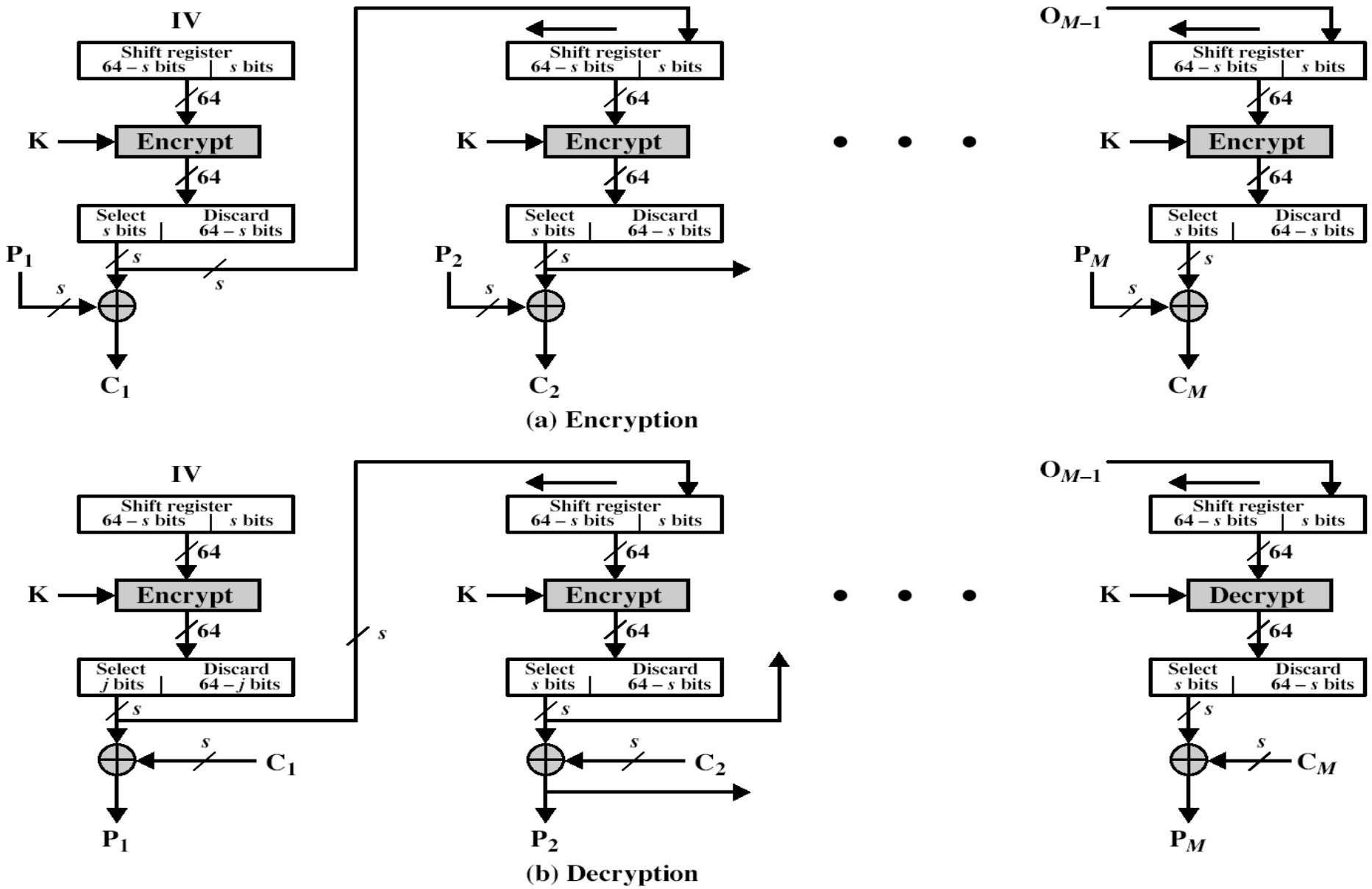


(a) Encryption



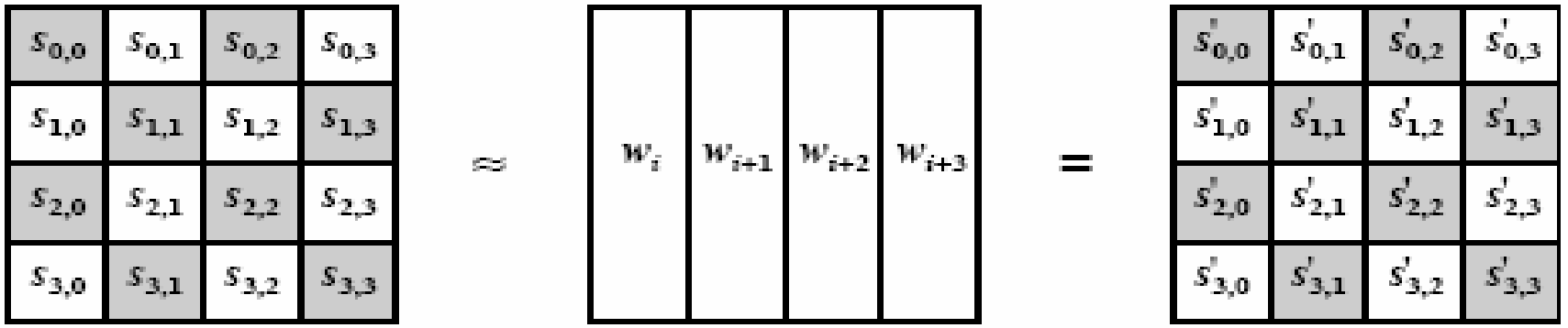
(b) Decryption

Output FeedBack (OFB)



Trasformazione Add Round Key (Diretta e Inversa)

- XOR della matrice di dati con i 128 bit della chiave di fase.
- Trasformazione molto semplice: sicurezza garantita dalla complessità dell'espansione della chiave e dalla complessità delle altre trasformazioni.



(b) Add Round Key Transformation

Esempio

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

$\oplus$

AC	19	28	57
77	FA	D1	5C
66	DC	29	00
F3	21	41	6A

$=$

EB	59	8B	1B
40	2E	A1	C3
F2	38	13	42
1E	84	E7	D2

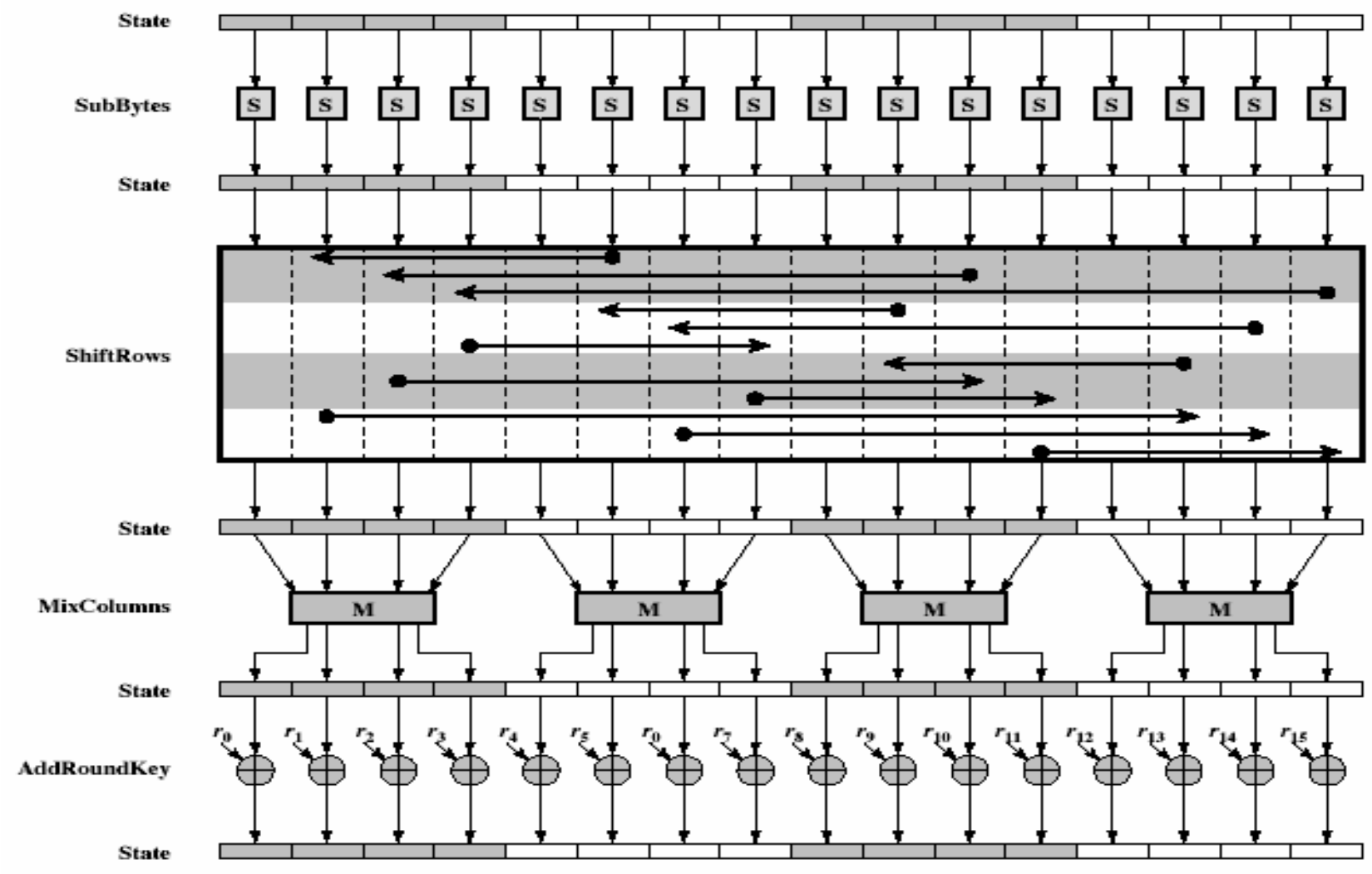
$$47 \xrightarrow{esa/bin} (0100\ 0111), \quad AC \xrightarrow{esa/bin} (1010\ 1100)$$

$$(0100\ 0111) \oplus (1010\ 1100) = (1110\ 1011) \xrightarrow{bin/esa} EB$$

$$40 \xrightarrow{esa/bin} (0100\ 0000), \quad 19 \xrightarrow{esa/bin} (0001\ 1001)$$

$$(0100\ 0000) \oplus (0001\ 1001) = (0101\ 1001) \xrightarrow{bin/esa} 59$$

Schema di una fase di AES



Espansione della chiave

- A partire da una chiave di 4 word (16 byte) viene prodotto un array di 44 word (156 byte).
- L'espansione è descritta dal seguente pseudocodice:

```
KeyEspansion (byte key[16], word w[44])
{
    word temp;
    for (i = 0; i < 4; i++),  w[i] = (key[4i], key[4i+1], key[4i+2], key[4i+3]);
    for (i = 4; i < 44; i++)
    {
        temp = w[i-1];
        if (i mod 4 = 0) temp = SubWord(RotWord(temp)) ⊕ Rcon[i/4];
        w[i] = w[i-4] ⊕ temp;
    }
}
```

Espansione della chiave

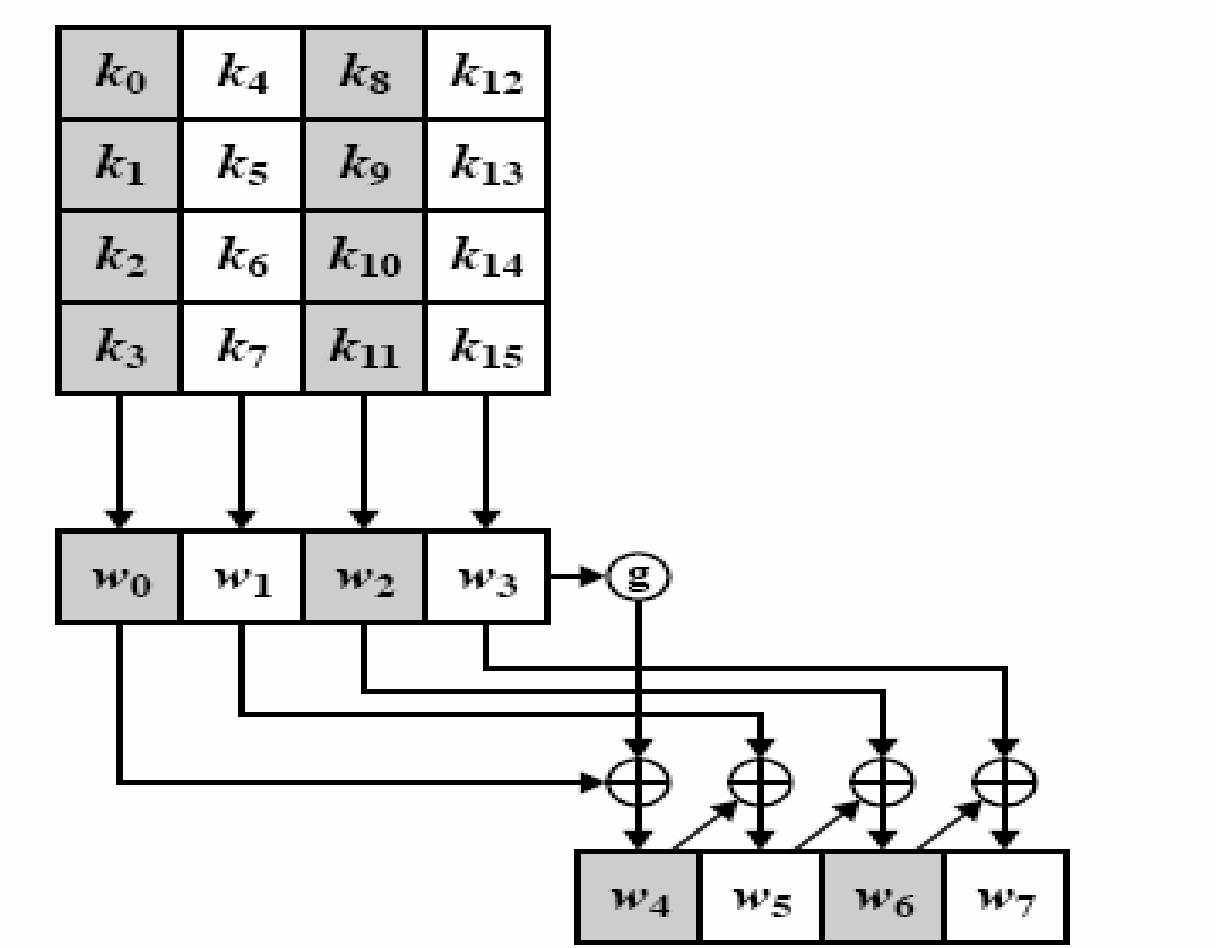
- La chiave viene copiata nelle prime 4 word della chiave espansa.
- Ciascuna word aggiuntiva w[i] dipende da w[i-1] e w[i-4]:
  - Se i non è multiplo di 4, viene usata una semplice funzione di XOR;
  - Altrimenti viene usata una funzione più complessa

Espansione della chiave

- **RotWord** → svolge uno scorrimento circolare a sinistra su una word
- **SubWord** → svolge una sostituzione su ciascun byte della word utilizzando la medesima S-box della trasformazione Substitute Bytes diretta.
- **Rcon[j]** → costante di fase. E' una word in cui i tre byte più a destra sono zero. In esadecimale si ha: Rcon[j] = (RC[j],0,0,0), con

j	1	2	3	4	5	6	7	8	9	10
RC[j]	01	02	04	08	10	20	40	80	1B	36

Schema di generazione delle prime 8 word della chiave espansa



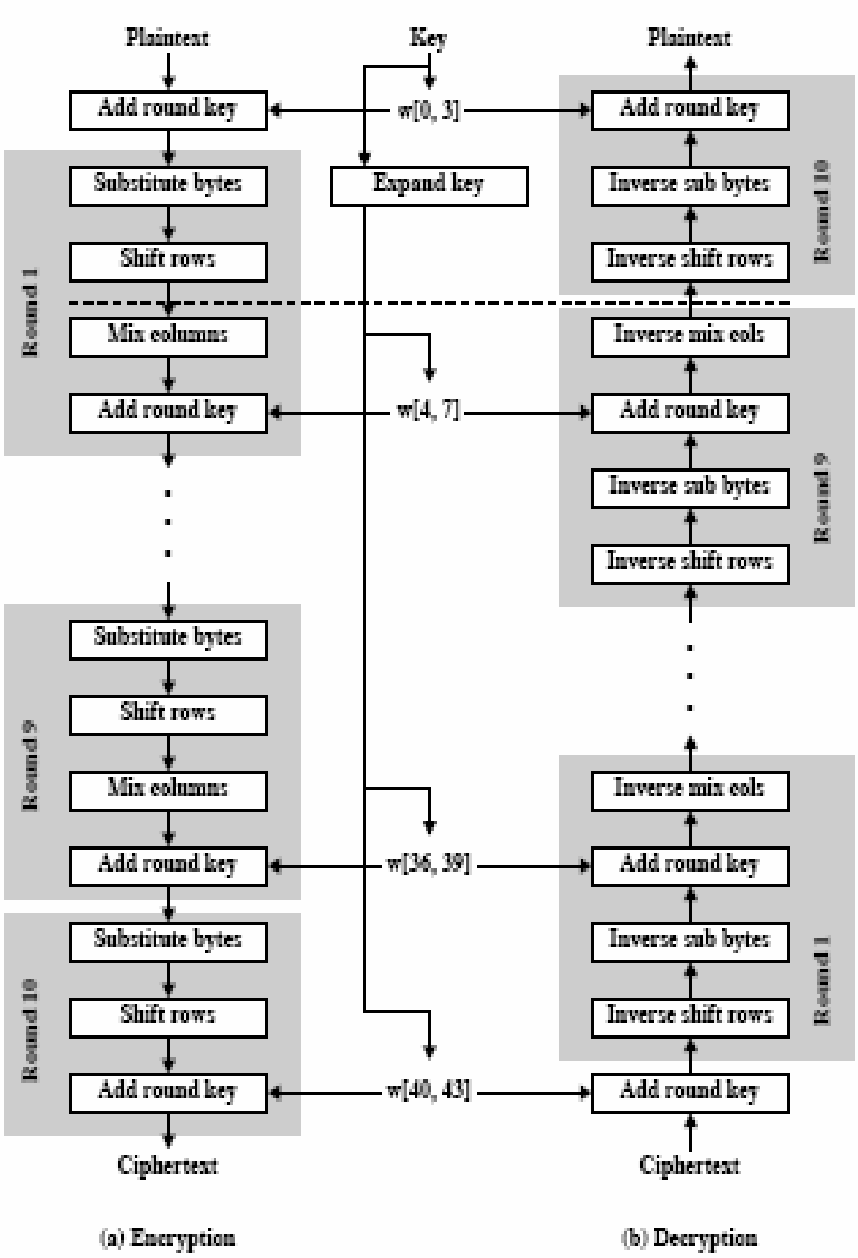
Criteri usati per la generazione della chiave espansa

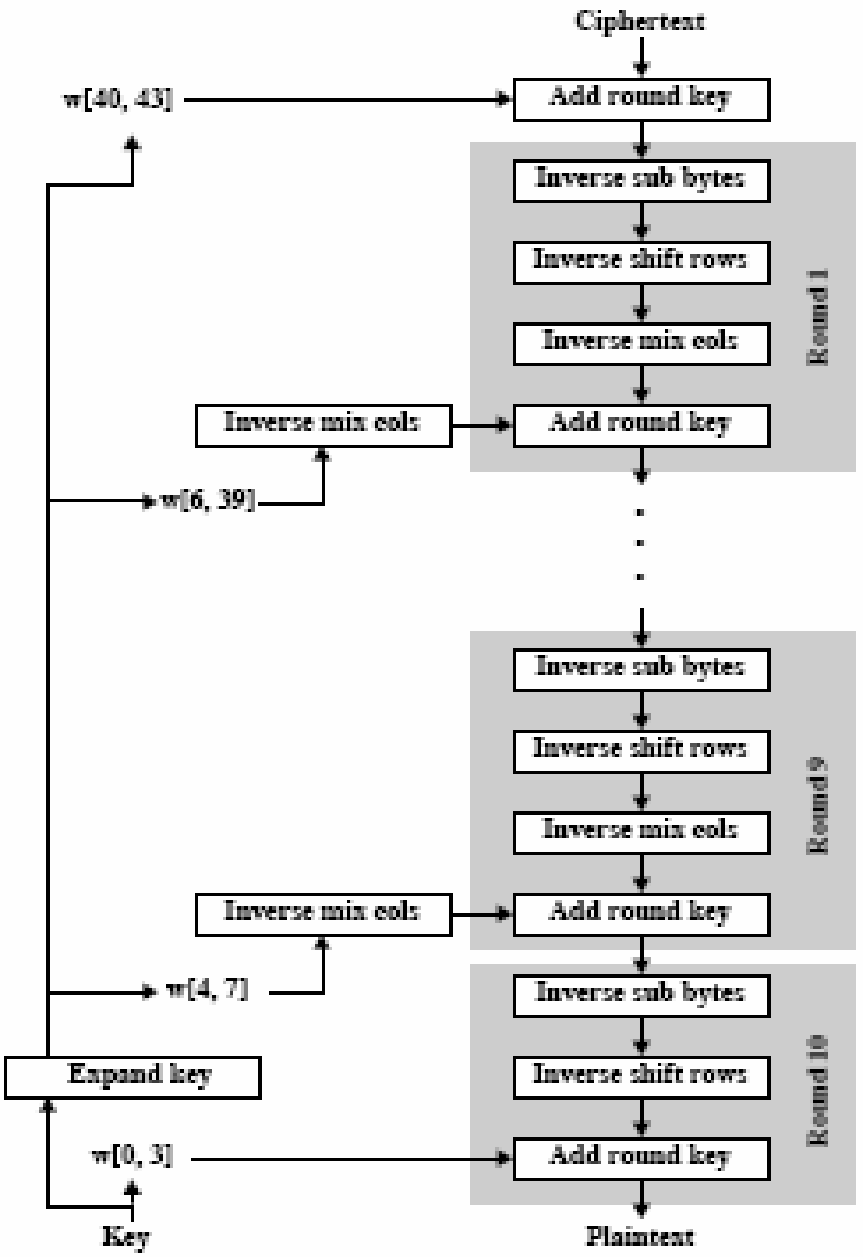
- Resistenza ad attacchi crittografici
- Trasformazione invertibile
- Alta velocità di esecuzione
- Uso di costanti di fasi per eliminare le simmetrie nelle varie fasi
- Ogni bit della chiave crittografica influenza più bit della chiave espansa
- Algoritmo di generazione non lineare.
- Facilità di descrizione

Cifratura inversa equivalente

- Per la crittografia e la decrittografia di AES, la programmazione della chiave è la stessa, ma la sequenza di operazioni da effettuare è diversa.
  - Sono richiesti due moduli software o firmware per le applicazioni che effettuano entrambe le operazioni

- Vi è una versione equivalente dell'algoritmo di decrittografia che usa la stessa sequenza di operazioni di quello di crittografia (invertendole), ma una diversa programmazione della chiave.


Plaintext
Key
Add round key
w[0, 3]
Expand key
Substitute bytes
Shift rows
Mix columns
Add round key
w[4, 7]
Substitute bytes
Shift rows
Mix columns
Add round key
w[36, 39]
Substitute bytes
Shift rows
Add round key
w[40, 43]
Round 1
Round 9
Round 10
Ciphertext
(a) Encryption
Plaintext
Add round key
Inverse sub bytes
Inverse shift rows
Inverse mix cols
Add round key
Inverse sub bytes
Inverse shift rows
Inverse mix cols
Add round key
Inverse sub bytes
Inverse shift rows
Add round key
Round 10
Round 9
Round 1
Ciphertext
(b) Decryption


Ciphertext
w[40, 43]
Add round key
Inverse sub bytes
Inverse shift rows
Inverse mix cols
Inverse mix cols
Add round key
w[6, 39]
Inverse sub bytes
Inverse shift rows
Inverse mix cols
Inverse mix cols
Add round key
w[4, 7]
Inverse sub bytes
Inverse shift rows
Expand key
w[0, 3]
Add round key
Key
Plaintext
Round 1
Round 9
Round 10

Cifratura inversa equivalente

- **Scambio InvShiftRows e InvSubBytes:** InvShiftRows altera la sequenza dei byte, ma non il loro contenuto; InvSubBytes agisce sul contenuto del byte, indipendentemente dalla sua posizione → Possono essere invertite.
- **Scambio InvMixColumns e AddRoundKey:** si ha

$$\text{InvMixColumns}(\text{S}_\text{j} \oplus w_j) = \text{InvMixColumns}(S_j) \oplus \text{InvMixColumns}(w_j)$$

infatti

$$\begin{pmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{pmatrix} \begin{pmatrix} s_0 \oplus w_0 \\ s_1 \oplus w_1 \\ s_2 \oplus w_2 \\ s_3 \oplus w_3 \end{pmatrix} = \begin{pmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{pmatrix} \begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \end{pmatrix} \oplus \begin{pmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{pmatrix} \begin{pmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{pmatrix}$$

Aspetti implementativi

- Implementazione efficiente su CPU a 8 bit (presenti nelle smart card):
  - **Add Round Key:** semplice operazione di XOR
  - **Shift Rows:** operazione di scorrimento dei byte
  - **Substitute Bytes:** opera a livello del byte e richiede una tabella di soli (16x16) 256 byte
  - **Mix Columns:** richiede solo scorrimenti e XOR
- Implementazione efficiente su CPU a 32 bit: tutte le operazioni possono essere espresse sulle word invece che sui singoli byte.

Algoritmo di crittografia
Blowfish

Caratteristiche di Blowfish

- Sviluppato da Bruce Schneier (1993)
- **Velocità:** su CPU a 32bit può crittografare dati ad una frequenza di 18 cicli di clock per byte (DES, invece, richiede 50 cicli di clock per byte).
- **Compatezza:** può operare con 5KB di memoria.
- **Semplicità:** la struttura è facile da implementare (facilità di analisi).
- **Sicurezza regolabile:** lunghezza della chiave regolabile da 32 a 448 bit.
- **Blocco dati:** 64 bit

Generazione della sottochiave e della S-box

- Una chiave da 32 a 448 bit (ovvero, da 1 a 14 word da 32 bit) viene usata per:
  - generare 18 sottochiavi a 32 bit
  - generare 4 S-box 256x32

- La chiave è conservata in una matrice K:
  $K=[K_1,K_2,\ldots,K_j]$ con $1\leq j\leq 14$.

- Le sottochiavi sono memorizzate in un array P:
  $P=[P_1,P_2,\ldots,P_{18}]$

Generazione della sottochiave e della S-box

- Ciascuna delle 4 S-Box è memorizzata in 256 voci da 32 bit:

$$S_{1,0}, S_{1,1}, \ldots, S_{1,255}$$
$$S_{2,0}, S_{2,1}, \ldots, S_{2,255}$$
$$S_{3,0}, S_{3,1}, \ldots, S_{3,255}$$
$$S_{4,0}, S_{4,1}, \ldots, S_{4,255}$$

Generazione della sottochiave e della S-box

1. Inizializzare prima l'array P e poi le 4 S-Box utilizzando i bit della parte frazionaria della costante π.

2. Svolgere uno XOR bit-a-bit dell'array P e dell'array K, riutilizzando ciclicamente le word dell'array K. Per esempio, per la chiave di lunghezza massima (14 word da 32 bit): $P_1=P_1 \oplus K_1$, $P_2=P_2 \oplus K_2$,…., $P_{14}=P_{14} \oplus K_{14}$, $P_{15}=P_{15} \oplus K_1$,…, $P_{18}=P_{18} \oplus K_4$.

3. Crittografare un blocco di 64 bit nulli utilizzando P e le 4 S-Box correnti. Sostituire P_1 e P_2 con l'output della crittografia.

4. Crittografare l'output del passo 3 utilizzando P e le 4 S-Box correnti e sostituire P_3 e P_4 con l'output della crittografia.

5. Continuare questa operazione per aggiornare tutte le word di P e tutte le word delle 4 S-Box.

Commenti

- Per produrre l'array finale P e le 4 S-Box finali sono necessarie 521 esecuzioni consecutive dell'algoritmo di crittografia.
  - Blowfish non è adatto per applicazioni in cui la chiave segreta cambia spesso
  - Per memorizzare la matrice P e le 4 S-Box sono necessari più di 4 KB di memoria
  - Resistente contro attacchi a forza bruta (fino ad ora la sicurezza di Blowfish è ritenuta inattaccabile!)

Crittografia e Decrittografia

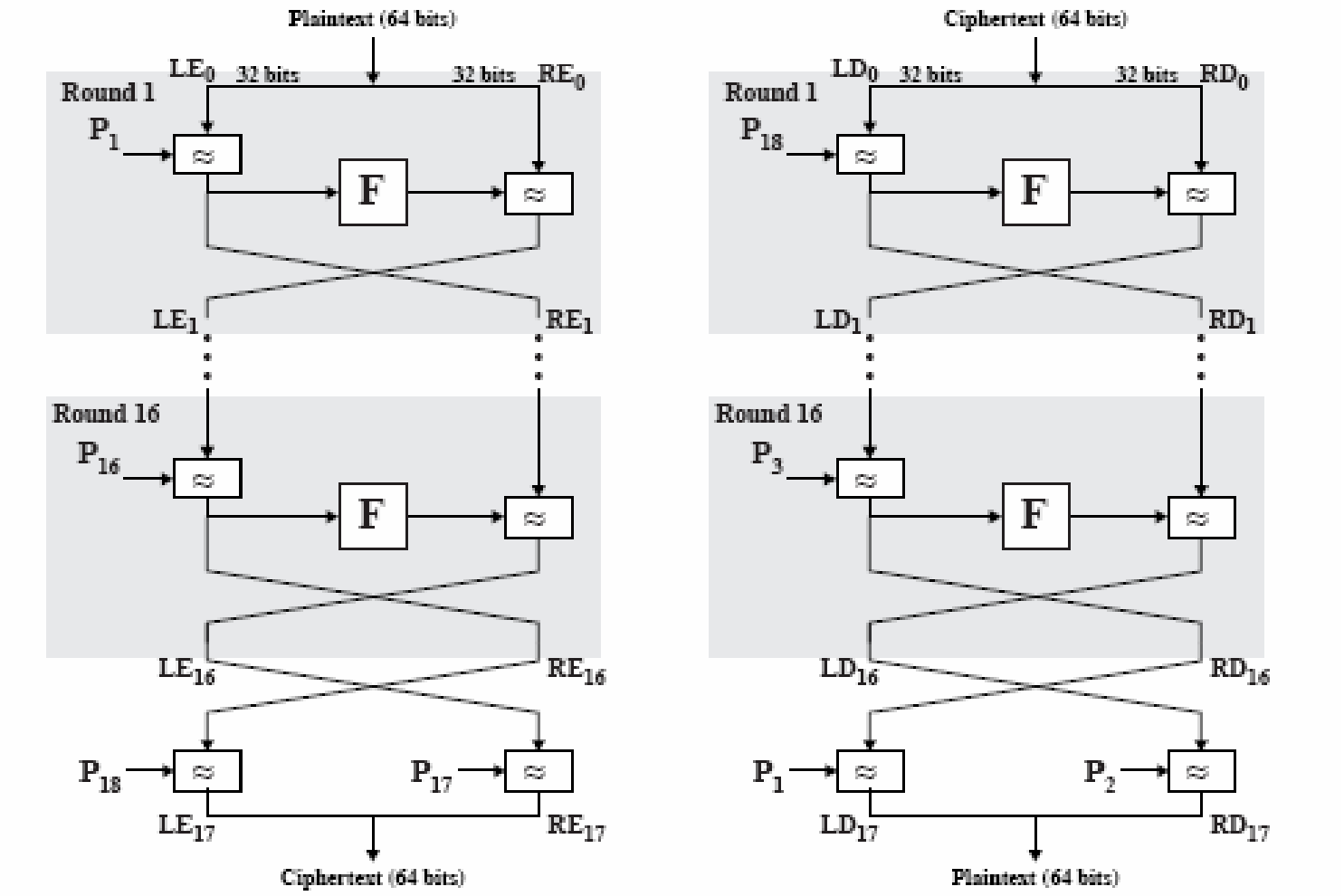


Figure 6.3 Blowfish Encryption and Decryption

Funzione F (S-Box)

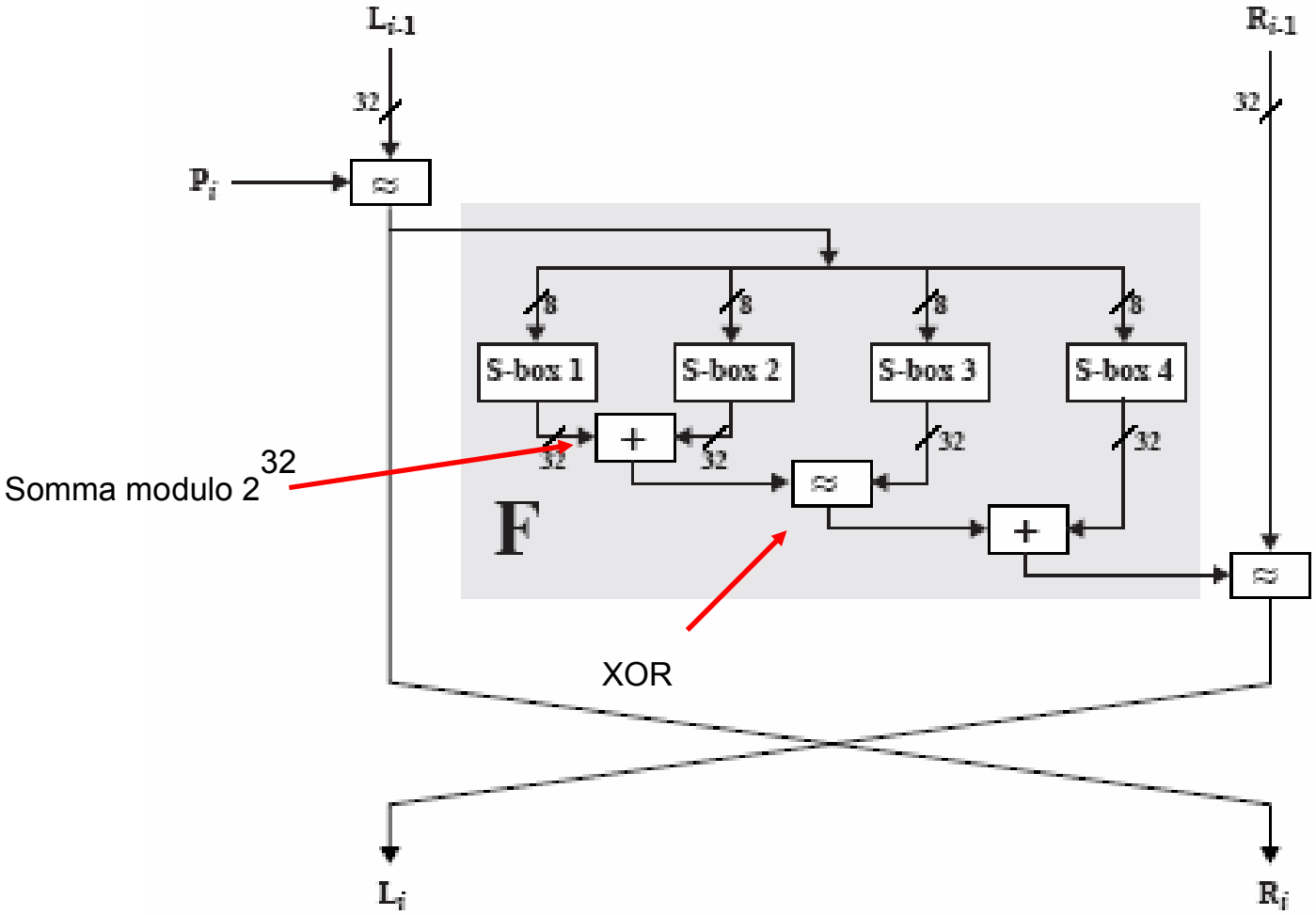


Figure 6.4 Detail of Single Blowfish Round

Funzione F (S-Box)

- Sono usate due operazioni primitive:
  - **Somma di word**, rappresentata con +, eseguita modulo 2^{32}
  - **XOR bit-a-bit**, rappresentato con $\oplus$

- L'input di 32 bit di F viene diviso in 4 byte. Se questi byte vengono indicati con *a, b, c, d*, si ha

$$\mathbf{F[}a,b,c,d\mathbf{]=((S_{1,a}+S_{2,b})\oplus S_{3,c})+S_{4,d})}$$

Blowfish: descrizione

- La generazione di una chiave richiede 522 applicazioni dell'algoritmo: difficoltà dell'attacco a forza bruta
- La funzione F introduce un forte effetto valanga
- Le S-box dipendono dalla chiave
- La funzione F non dipende dalla fase