2 Cifrario di Base a Rete di Sostituzione-Permutazione

Il cifrario che verrà utilizzato per presentare i concetti è il Substitution-Permutation Network (SPN). Il cifrario prende un blocco di input a 16 bit e lo elabora ripetendo le operazioni fondamentali di un round per quattro volte. Ogni round consiste in:

1. **Sostituzione**
2. **Permutazione**
3. **Key Mixing**

Queste operazioni di base sono simili a quelle utilizzate nei cifrari moderni.

2.1 Sostituzioni

Nel nostro cifrario, suddividiamo il blocco di dati a 16 bit in quattro sotto-blocchi da 4 bit ciascuno. Ogni sotto-blocco viene elaborato da una S-box 4×4 (una sostituzione con 4 bit di input e 4 bit di output), che può essere facilmente implementata con una tabella di lookup di sedici valori a 4 bit, indicizzati dall'intero rappresentato dai 4 bit di input.

La proprietà fondamentale di una S-box è che è una **mappatura non lineare**, ovvero i bit di output non possono essere rappresentati come un'operazione lineare sui bit di input.

- Se un sistema è **lineare**, il suo output può essere espresso come una combinazione lineare (ad esempio, XOR, addizione modulo 2, moltiplicazione di matrici) dei bit di input.
- Se è **non lineare**, non esiste una relazione lineare semplice di questo tipo.

1. Esempio di mappatura lineare

Supponiamo di avere una funzione lineare che trasforma 3 bit di input in 3 bit di output, definita come:

$$\begin{cases} y_0 = x_0 \oplus x_1 \\ y_1 = x_1 \oplus x_2 \\ y_2 = x_0 \oplus x_2 \end{cases}$$

Dove $\oplus$ è l'operazione XOR (addizione modulo 2).

Input: $(x_0, x_1, x_2) = (1, 0, 1)$

Output:

$$\begin{aligned} y_0 &= 1 \oplus 0 = 1 \\ y_1 &= 0 \oplus 1 = 1 \\ y_2 &= 1 \oplus 1 = 0 \end{aligned}$$

Risultato: $(1, 1, 0)$.

Questa è una mappatura lineare perché ogni bit di output è una combinazione lineare (XOR) dei bit di input.

2. Esempio di S-box non lineare

Una S-box è progettata per rompere la linearità. Consideriamo una semplice S-box a 3 bit (esempio semplificato, non usato in pratica):

Input (x)	Output (S[x])
000	110
001	101
010	011
011	000
100	111
101	100
110	010
111	001

Input: $(1, 0, 1)$ (5 in decimale)
Output: Cercando nella tabella, $S[5] = 100$.

Perché è non lineare?
Proviamo a verificare se esiste una combinazione lineare che descrive tutti gli output. Consideriamo la generica forma lineare:

$$\begin{cases} y_0 = a_0x_0 \oplus b_0x_1 \oplus c_0x_2 \\ y_1 = a_1x_0 \oplus b_1x_1 \oplus c_1x_2 \\ y_2 = a_2x_0 \oplus b_2x_1 \oplus c_2x_2 \end{cases}$$

Analizzando il caso input $000 \rightarrow 110$:

$$\begin{cases} 1 = 0 \oplus 0 \oplus 0 \\ 1 = 0 \oplus 0 \oplus 0 \\ 0 = 0 \oplus 0 \oplus 0 \end{cases}$$

Otteniamo $1 = 0$, che è una contraddizione. Quindi la S-box è necessariamente non lineare.

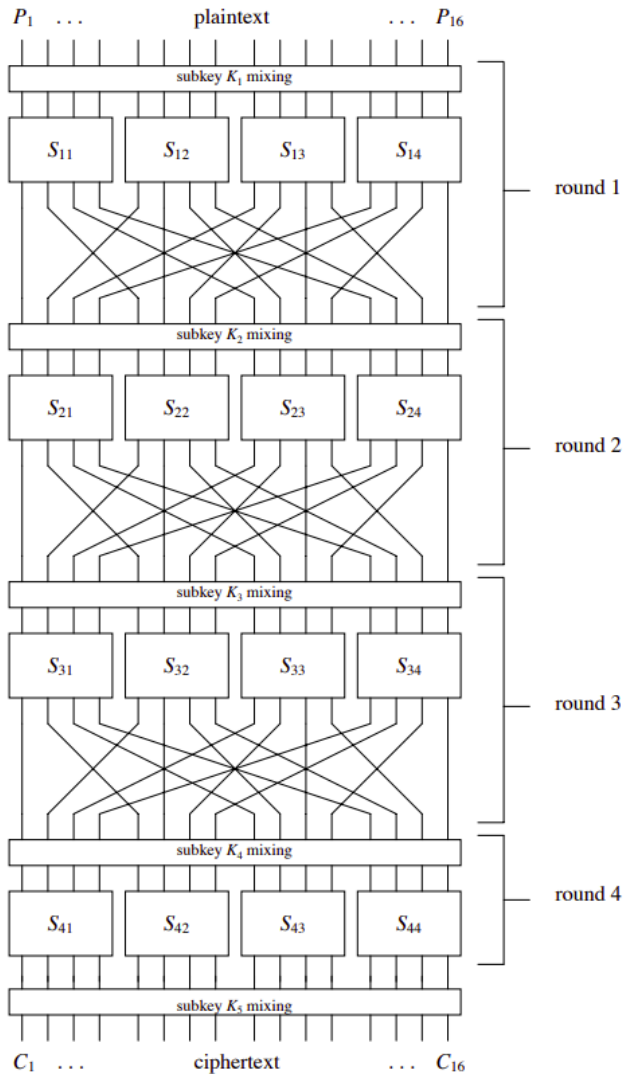
I sistemi lineari sono deboli in crittografia perché possono essere facilmente invertiti o violati utilizzando tecniche come l'eliminazione di Gauss o la crittanalisi lineare. La non linearità introduce complessità, rendendo le funzioni più difficili da analizzare e invertire, il che è essenziale per una cifratura sicura.

Per il nostro cifrario, utilizzeremo la stessa mappatura non lineare per tutte le S-box. Gli attacchi di crittanalisi lineare e differenziale si applicano allo stesso modo sia che ci sia una sola mappatura sia che tutte le S-box abbiano mappature diverse.

input	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
output	E	4	D	1	2	F	B	8	3	A	6	C	5	9	0	7

2.2 Permutazione

La parte di permutazione di un round consiste semplicemente nella trasposizione dei bit o nella permutazione delle loro posizioni. L'output i della S-box j viene collegato all'input j della S-box i. Si noti che una permutazione nell'ultimo round non avrebbe alcuno scopo e, pertanto, il nostro cifrario non la prevede.



2.3 Key Mixing

Per ottenere il mixing con la chiave, utilizziamo un semplice XOR bit-a-bit tra i bit della chiave associati a un round (chiamati sottochiave) e il blocco di dati in input a un round. Inoltre, una sottochiave viene applicata anche dopo l'ultimo round, assicurando che l'ultimo strato di sostituzione non possa essere facilmente ignorato da chi lavora all'indietro attraverso la sostituzione dell'ultimo round. Normalmente, in un cifrario, la sottochiave per un round è derivata dalla chiave principale attraverso un processo noto come schedulazione della chiave. Nel nostro cifrario, assumeremo che tutti i bit delle sottochiavi siano generati in modo indipendente e non correlati.

2.4 Decifratura

Per decifrare, i dati vengono essenzialmente elaborati all'indietro attraverso il network. Tuttavia, le mappature utilizzate nelle S-box della rete di decifratura sono l'inverso delle mappature nella rete di cifratura (ovvero, l'input diventa output e l'output diventa input). Ciò implica che, affinché una SPN permetta la decifratura, tutte le S-box devono essere biiettive, cioè una mappatura uno-a-uno con lo stesso numero di bit di input e output. Inoltre, affinché la rete decifri correttamente, le sottochiavi vengono applicate in ordine inverso e i bit delle sottochiavi devono essere riorganizzati secondo la permutazione. Si noti inoltre che l'assenza della permutazione dopo l'ultimo round garantisce che la rete di decifratura possa avere la stessa struttura della rete di cifratura. (Se ci fosse una permutazione dopo l'ultimo strato di sostituzione nella cifratura, la decifratura richiederebbe una permutazione prima del primo strato di sostituzione.)

3. Crittanalisi Lineare

3.1 Panoramica dell’Attacco

La crittanalisi lineare cerca di sfruttare le occorrenze ad alta probabilità di espressioni lineari che coinvolgono bit del plaintext, bit del "testo cifrato" (in realtà utilizzeremo bit provenienti dall’output del penultimo round) e bit delle sottochiavi. Si tratta di un attacco a plaintext noto (*known plaintext attack*), ovvero si basa sull’ipotesi che l’attaccante disponga di un insieme di testi in chiaro e dei corrispondenti testi cifrati. Tuttavia, l’attaccante non può scegliere quali testi in chiaro (e relativi testi cifrati) siano disponibili. In molte applicazioni e scenari, è ragionevole assumere che l’attaccante abbia accesso a un insieme casuale di coppie (plaintext, testo cifrato).

L’idea di base è approssimare il funzionamento di una parte del cifrario con un’espressione lineare, dove la linearità si riferisce a un’operazione bit-a-bit modulo 2 (ovvero lo XOR, indicato con "⊕"). Tale espressione ha la forma:

$$X_{i_1} \oplus X_{i_2} \oplus \cdots \oplus X_{i_u} \oplus Y_{j_1} \oplus Y_{j_2} \oplus \cdots \oplus Y_{j_v} = 0$$

dove X_i rappresenta l’i-esimo bit dell’input $X = [X_1, X_2, \ldots]$ e Y_j rappresenta il j-esimo bit dell’output $Y = [Y_1, Y_2, \ldots]$. Questa equazione rappresenta la somma XOR di u bit di input e v bit di output.

L’approccio della crittanalisi lineare consiste nell’individuare espressioni di questo tipo che abbiano una probabilità di verificarsi significativamente alta o bassa. (Se un’espressione lineare come quella sopra fosse sempre vera o falsa per ogni input e output, il cifrario sarebbe banale da violare.) Se un cifrario mostra una tendenza per cui l’equazione vale con alta probabilità (o non vale con alta probabilità), ciò è indice di una scarsa capacità di randomizzazione del cifrario.

Considera che se selezionassimo valori casuali per u + v bit e li inserissimo nell'equazione precedente, la probabilità che l'espressione sia valida sarebbe esattamente 1/2. È la deviazione o **bias** dalla probabilità di 1/2 che un'espressione sia valida a essere sfruttata nella crittanalisi lineare: più un'espressione lineare si discosta dall'avere una probabilità di 1/2, migliore sarà l'applicabilità della crittanalisi lineare. Nel resto del documento, ci riferiamo alla quantità di cui la probabilità di un'espressione lineare devia da 1/2 come **bias della probabilità lineare**.

Pertanto, se l'espressione precedente è valida con probabilità p_L per casuali testi in chiaro e corrispondenti testi cifrati, allora il bias di probabilità è $p_L - \frac{1}{2}$.
Maggiore è l'entità del bias di probabilità, $\left|p_L - \frac{1}{2}\right|$, migliore sarà l'applicabilità della crittanalisi lineare, con un minor numero di testi in chiaro noti richiesti per l'attacco.

Analizziamo la costruzione di un'approssimazione lineare che coinvolge:

- i bit del plaintext (rappresentati da X nell'equazione (1))
- l'input all'ultimo round del cifrario (o equivalentemente l'output del penultimo round), rappresentato da Y nell'equazione (1).

I bit del plaintext sono casuali e di conseguenza lo sono anche i bit di input dell'ultimo round.

L'equazione (1) potrebbe essere riformulata equivalentemente per avere al secondo membro la somma di alcuni bit della sottochiave. Tuttavia, nella forma attuale con "0" a destra, l'equazione coinvolge implicitamente bit della sottochiave: questi bit sono fissi ma ignoti (essendo determinati dalla chiave sotto attacco) e sono implicitamente assorbiti nello "0" al secondo membro dell'equazione (1) e nella probabilità p_L che l'espressione lineare sia valida.

Se la somma dei bit di sottochiave coinvolti vale "0", il bias dell'equazione (1) avrà lo stesso segno (+ o -) del bias dell'espressione contenente la somma delle sottochiavi. Se invece la somma dei bit di sottochiave coinvolti vale "1", il bias dell'equazione (1) avrà segno opposto.

L'equazione 1 in questo caso sta a indicare lo xor dei bit del plaintext con lo xor dei bit del testo cifrato = 0.

Si noti che $p_L = 1$ implica che l'espressione lineare (1) rappresenta perfettamente il comportamento del cifrario e che il cifrario presenta una vulnerabilità catastrofica. Se $p_L = 0$, allora (1) rappresenta una relazione affine nel cifrario, anch'essa indicativa di una vulnerabilità catastrofica. Per sistemi a somma mod-2, una funzione affine è semplicemente il complemento di una funzione lineare. Sia le approssimazioni lineari che quelle affini, indicate rispettivamente da $p_L > 1/2$ e $p_L < 1/2$, sono ugualmente suscettibili alla crittanalisi lineare, e useremo generalmente il termine "lineare" per riferirci sia a relazioni lineari che affini.

Si costruiscono espressioni altamente lineari e quindi sfruttabili considerando le proprietà dell'unico componente non lineare del cifrario: la S-box. Quando si enumerano le proprietà di non linearità della S-box, è possibile sviluppare approssimazioni lineari tra insiemi di bit di input e output della S-box. Di conseguenza, è possibile concatenare le approssimazioni lineari delle S-box in modo che i bit intermedi (cioè i bit di dati interni al cifrario) possano essere cancellati, ottenendo un'espressione lineare con un bias elevato che coinvolge solo bit del plaintext e bit di input dell'ultimo round.

Esempio teorico
In crittografia, un'approssimazione lineare è un'equazione del tipo:

$$\bigoplus_{i \in A} P_i \oplus \bigoplus_{j \in B} C_j = \bigoplus_{k \in K} K_k \quad (1)$$

Dove:

- P_i: bit del plaintext (*plaintext*),
- C_j: bit del testo cifrato (*ciphertext*),
- K_k: bit della sottochiave (*key bits*),

L'equazione (1) **non è esatta**, ma ha una **probabilità** $p \neq 0.5$ di essere vera. Questo "bias" ($|p - 0.5|$) è sfruttato per attaccare il cifrario.

L'equazione (1) può essere riscritta in due modi equivalenti:

1. **Forma implicita** (con "0" a destra):

$$\bigoplus_{i \in A} P_i \oplus \bigoplus_{j \in B} C_j = 0 \quad (1a)$$

   Qui i bit della sottochiave $\bigoplus_{k \in K} K_k$ sono "assorbiti" nello "0":
   - Se $\bigoplus_{k \in K} K_k = 0$, l'equazione (1a) è vera con probabilità p.
   - Se $\bigoplus_{k \in K} K_k = 1$, l'equazione (1a) è vera con probabilità $1 - p$ (il bias cambia segno).
2. **Forma esplicita** (con i bit di sottochiave):

$$\bigoplus_{i \in A} P_i \oplus \bigoplus_{j \in B} C_j = \bigoplus_{k \in K} K_k \quad (1b)$$

Nella crittanalisi lineare:

1. **Fase 1**: Troviamo un'approssimazione lineare (1a) con bias significativo ($p \neq 0.5$), **ignorando inizialmente la sottochiave**.
2. **Fase 2**: Usiamo (1a) per **indovinare** $\bigoplus_{k \in K} K_k$:
   - Se (1a) è vera più spesso del 50%, assumiamo $\bigoplus_{k \in K} K_k = 0$.
   - Se è vera meno del 50%, assumiamo $\bigoplus_{k \in K} K_k = 1$.

Esempio pratico
Supponiamo di avere:

$$P_0 \oplus P_1 \oplus C_0 = 0 \quad (\text{bias } p = 0.7)$$

- **Se** $K_0 \oplus K_1 = 0$: L'equazione è vera con probabilità 0.7.
- **Se** $K_0 \oplus K_1 = 1$: L'equazione è vera con probabilità 0.3 (cioè $1 - 0.7$).

Attacco:

1. Raccolta di N coppie (plaintext, ciphertext).
2. Contiamo quante volte $P_0 \oplus P_1 \oplus C_0 = 0$.
   - Se è vero il 70% delle volte → $K_0 \oplus K_1 = 0$.
   - Se è vero il 30% delle volte → $K_0 \oplus K_1 = 1$.

Scenario semplificato

- **Cifrario**: Un mini-cifrario a blocchi con:

- **plaintext (P)**: 2 bit (P_0, P_1).
- **Testo cifrato (C)**: 2 bit (C_0, C_1).
- **Sottochiave (K)**: 2 bit (K_0, K_1) (ignoti all'attaccante).

- **Approssimazione lineare trovata** (con bias $p = 0.75$):

$$P_0 \oplus P_1 \oplus C_0 = K_0 \oplus K_1 \quad (1)$$

  - Se $K_0 \oplus K_1 = 0$, l'equazione $P_0 \oplus P_1 \oplus C_0 = 0$ è vera con probabilità 0.75.
  - Se $K_0 \oplus K_1 = 1$, l'equazione è vera con probabilità 0.25 (cioè $1 - 0.75$).

Fase 1: Raccolta dati
Supponiamo di avere 8 coppie (plaintext, ciphertext), alcune delle quali cifrate con la stessa sottochiave K_0, K_1:

Plaintext (P)	Ciphertext (C)	$P_0 \oplus P_1 \oplus C_0$
00	10	$0 \oplus 0 \oplus 1 = 1$
01	00	$0 \oplus 1 \oplus 0 = 1$
10	11	$1 \oplus 0 \oplus 1 = 0$
11	01	$1 \oplus 1 \oplus 0 = 0$
00	11	$0 \oplus 0 \oplus 1 = 1$
01	10	$0 \oplus 1 \oplus 1 = 0$
10	00	$1 \oplus 0 \oplus 0 = 1$
11	11	$1 \oplus 1 \oplus 1 = 1$

Fase 2: Calcolo delle frequenze
Contiamo quante volte $P_0 \oplus P_1 \oplus C_0 = 0$:

- **Risultati**: `[1, 1, 0, 0, 1, 0, 1, 1]`
- **Numero di "0"**: 3 su 8 (frequenza osservata = $3/8 = 0.375$).

Fase 3: Deduzione dei bit di sottochiave

- **Se** $K_0 \oplus K_1 = 0$:
  L'equazione dovrebbe essere vera con probabilità 0.75, ma osserviamo 0.375 (vicino a 0.25).
- **Se** $K_0 \oplus K_1 = 1$:
  L'equazione dovrebbe essere vera con probabilità 0.25, e osserviamo 0.375 (più vicino a 0.25 che a 0.75).

Conclusione:
Poiché la frequenza osservata (0.375) è più vicina a 0.25 che a 0.75, deduciamo che:

$$K_0 \oplus K_1 = 1$$

Fase 4: Ricostruzione della chiave
Supponiamo di sapere (da altre approssimazioni) che:

- $K_0 = 1$.
  Allora:

$$K_1 = K_0 \oplus (K_0 \oplus K_1) = 1 \oplus 1 = 0$$

  Sottochiave recuperata: $K_0 = 1, K_1 = 0$.

Cosa succede con bias più deboli
Se avremmo avuto $p = 0.51$ (bias 0.01):

- Servirebbero $\approx \frac{1}{(0.01)^2} = 10{,}000$ coppie per distinguere il segnale dal rumore.
  Esempio con bias diverso
  Se avessi usato $p = 0.625$ (bias 0.125):
- Per $K_0 \oplus K_1 = 0$: l'equazione è vera il 62.5% delle volte.
- Per $K_0 \oplus K_1 = 1$: è vera il 37.5% delle volte.

- Servirebbero $\approx \frac{1}{(0.125)^2} = 64$ coppie per vedere l'effetto.

3.2 Piling-Up Principle

Consideriamo due variabili binarie casuali, X_1 e X_2. Iniziamo osservando le semplici relazioni:

- $X_1 \oplus X_2 = 0$ è un'espressione lineare ed equivale a $X_1 = X_2$
- $X_1 \oplus X_2 = 1$ è un'espressione affine ed equivale a $X_1 \neq X_2$

Supponiamo ora che le distribuzioni di probabilità siano date da:

$$\Pr(X_1 = i) = \begin{cases} p_1 & \text{se } i = 0 \\ 1 - p_1 & \text{se } i = 1 \end{cases}$$

e

$$\Pr(X_2 = i) = \begin{cases} p_2 & \text{se } i = 0 \\ 1 - p_2 & \text{se } i = 1 \end{cases}$$

Se le due variabili sono indipendenti, allora:

$$\Pr(X_1 = i, X_2 = j) = \begin{cases} p_1 p_2 & \text{se } i = 0, j = 0 \\ p_1(1 - p_2) & \text{se } i = 0, j = 1 \\ (1 - p_1)p_2 & \text{se } i = 1, j = 0 \\ (1 - p_1)(1 - p_2) & \text{se } i = 1, j = 1 \end{cases}$$

Si può dimostrare che:

$$\begin{aligned} \Pr(X_1 \oplus X_2 = 0) &= \Pr(X_1 = X_2) \\ &= \Pr(X_1 = 0, X_2 = 0) + \Pr(X_1 = 1, X_2 = 1) \\ &= p_1 p_2 + (1 - p_1)(1 - p_2) \end{aligned}$$

Un approccio alternativo consiste nell'esprimere:

$$p_1 = \frac{1}{2} + \epsilon_1 \quad \text{e} \quad p_2 = \frac{1}{2} + \epsilon_2$$

dove ε_1 e ε_2 rappresentano i bias di probabilità, con $-1/2 \leq \varepsilon_1, \varepsilon_2 \leq +1/2$. Ne consegue che:

$$\Pr(X_1 \oplus X_2 = 0) = \frac{1}{2} + 2\epsilon_1\epsilon_2$$

e il bias $\epsilon_{1,2}$ di $X_1 \oplus X_2 = 0$ è:

$$\epsilon_{1,2} = 2\epsilon_1\epsilon_2.$$

Questo risultato può essere esteso a più di due variabili binarie casuali, X_1 fino a X_n, con probabilità $p_1 = \frac{1}{2} + \epsilon_1$ fino a $p_n = \frac{1}{2} + \epsilon_n$. La probabilità che $X_1 \oplus \cdots \oplus X_n = 0$ valga può essere determinata dal Piling-Up Lemma, che assume che tutte le n variabili binarie casuali siano indipendenti.

Lemma di Piling-Up (Matsui)
Per n variabili binarie indipendenti e casuali $X_1, X_2, \ldots, X_n$:

$$\Pr(X_1 \oplus \cdots \oplus X_n = 0) = \frac{1}{2} + 2^{n-1} \prod_{i=1}^{n} \epsilon_i$$

o equivalentemente:

$$\epsilon_{1,2,\ldots,n} = 2^{n-1} \prod_{i=1}^{n} \epsilon_i$$

dove $\epsilon_{1,2,\ldots,n}$ rappresenta il bias di $X_1 \oplus \cdots \oplus X_n = 0$.

- Se $p_i = 0$ o 1 per ogni i, allora $\Pr(X_1 \oplus \cdots \oplus X_n = 0) = 0$ o 1.
- Se solo un $p_i = \frac{1}{2}$, allora $\Pr(X_1 \oplus \cdots \oplus X_n = 0) = \frac{1}{2}$.

Nello sviluppo dell'approssimazione lineare di un cifrario, i valori X_i rappresenteranno in realtà approssimazioni lineari delle S-box.

Consideriamo quattro variabili binarie indipendenti X_1, X_2, X_3 e X_4. Sia:

$$\Pr(X_1 \oplus X_2 = 0) = \frac{1}{2} + \epsilon_{1,2}$$

e

$$\Pr(X_2 \oplus X_3 = 0) = \frac{1}{2} + \epsilon_{2,3}$$

Consideriamo la somma $X_1 \oplus X_3$ ottenuta aggiungendo $X_1 \oplus X_2$ e $X_2 \oplus X_3$:

$$\Pr(X_1 \oplus X_3 = 0) = \Pr\big([X_1 \oplus X_2] \oplus [X_2 \oplus X_3] = 0\big)$$

Stiamo quindi combinando espressioni lineari per formare una nuova espressione lineare. Poiché possiamo considerare le variabili casuali $X_1 \oplus X_2$ e $X_2 \oplus X_3$ come indipendenti, possiamo usare il Lemma di Piling-Up per determinare:

$$\Pr(X_1 \oplus X_3 = 0) = \frac{1}{2} + 2\epsilon_{1,2}\epsilon_{2,3}$$

e di conseguenza:

$$\epsilon_{1,3} = 2\epsilon_{1,2}\epsilon_{2,3}$$

Come vedremo:

- Le espressioni $X_1 \oplus X_2 = 0$ e $X_2 \oplus X_3 = 0$ sono analoghe alle approssimazioni lineari delle S-box
- $X_1 \oplus X_3 = 0$ è analoga a un'approssimazione del cifrario dove il bit intermedio X_2 viene eliminato

Naturalmente, l'analisi reale sarà più complessa e coinvolgerà molte approssimazioni di S-box.

3.3 Analizzando i componenti del cifrario

Vediamo delel vulnerabilità lineari delle S-box. Consideriamo la rappresentazione della S-box, con input $X = [X_1X_2X_3X_4]$ e output corrispondente $Y = [Y_1Y_2Y_3Y_4]$.

Tutte le approssimazioni lineari possono essere esaminate per valutarne l'utilità calcolando il bias di probabilità per ciascuna. In particolare, esaminiamo tutte le espressioni nella forma dell'equazione (1), dove X e Y rappresentano rispettivamente l'input e l'output della S-box.

Ad esempio, per la S-box utilizzata nel nostro cifrario, consideriamo l'espressione lineare:

$$X_2 \oplus X_3 \oplus Y_1 \oplus Y_3 \oplus Y_4 = 0$$

o equivalentemente:

$$X_2 \oplus X_3 = Y_1 \oplus Y_3 \oplus Y_4$$

Analizzando tutti i 16 possibili valori di input X e i corrispondenti output Y, si osserva che l'espressione precedente risulta vera in esattamente 12 casi su 16. Pertanto, il bias di probabilità risulta: $\frac{12}{16} - \frac{1}{2} = \frac{1}{4}$

Analogamente, per l'equazione:

$$X_1 \oplus X_4 = Y_2$$

si osserva un bias di probabilità pari a 0, mentre per l'equazione:

$$X_3 \oplus X_4 = Y_1 \oplus Y_4$$

il bias di probabilità risulta: $\frac{2}{16} - \frac{1}{2} = -\frac{3}{8}$ Nell'ultimo caso, la migliore approssimazione risulta essere di tipo affine (come indicato dal segno negativo). Tuttavia, il successo dell'attacco dipende esclusivamente dal valore assoluto del bias. Come dimostreremo, le approssimazioni affini possono essere utilizzate con la stessa efficacia delle approssimazioni lineari.

X_1	X_2	X_3	X_4	Y_1	Y_2	Y_3	Y_4	$X_2 \oplus X_3$	$Y_1 \oplus Y_3 \oplus Y_4$	$X_1 \oplus X_4$	Y_2	$X_3 \oplus X_4$	$Y_1 \oplus Y_4$
0	0	0	0	1	1	1	0	0	0	0	1	0	1
0	0	0	1	0	1	0	0	0	0	1	1	1	0
0	0	1	0	1	1	0	1	1	0	0	1	1	0
0	0	1	1	0	0	0	1	1	1	1	0	0	1
0	1	0	0	0	0	1	0	1	1	0	0	0	0
0	1	0	1	1	1	1	1	1	1	1	1	1	0
0	1	1	0	1	0	1	1	0	1	0	0	1	0
0	1	1	1	1	0	0	0	0	1	1	0	0	1
1	0	0	0	0	0	1	1	0	0	1	0	0	1
1	0	0	1	1	0	1	0	0	0	0	0	1	1
1	0	1	0	0	1	1	0	1	1	1	1	1	0
1	0	1	1	1	1	0	0	1	1	0	1	0	1
1	1	0	0	0	1	0	1	1	1	1	1	0	1
1	1	0	1	1	0	0	1	1	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1	0	1	0
1	1	1	1	0	1	1	1	0	0	0	1	0	1

Una enumerazione completa di tutte le approssimazioni lineari della S-box nel nostro cifrario è fornita nella tabella sottostante (tabella delle approssimazioni lineari). Ogni elemento nella tabella rappresenta il numero di corrispondenze tra:

- Equazione lineare in input (rappresentata in esadecimale come "Input Sum")
- Somma dei bit di output (rappresentata in esadecimale come "Output Sum")

Il tutto meno 8.

Il bias di probabilità per un particolare combinazione lineare di input e output si ottiene dividendo il valore dell'elemento per 16.

Il valore esadecimale che rappresenta una somma, quando interpretato come valore binario, indica le variabili coinvolte nella somma. Per una combinazione lineare di variabili di input rappresentata come:

$$a_1 \wedge X_1 \oplus a_2 \wedge X_2 \oplus a_3 \wedge X_3 \oplus a_4 \wedge X_4$$

dove:
$a_i \in \{0, 1\}$, $\wedge$ è l'AND e il valore esadecimale rappresenta il valore binario $a_1 a_2 a_3 a_4$, dove a_1 è il bit più significativo.

Analogamente, per una combinazione lineare di bit di output:

$$b_1 \wedge Y_1 \oplus b_2 \wedge Y_2 \oplus b_3 \wedge Y_3 \oplus b_4 \wedge Y_4$$

dove $b_i \in \{0, 1\}$, il valore esadecimale rappresenta il vettore binario $b_1 b_2 b_3 b_4$.

Il bias dell'equazione lineare:

$$X_3 \oplus X_4 = Y_1 \oplus Y_4$$

- Input esadecimale: 3 (0011)
- Output esadecimale: 9 (1001)

Risultati:

- Bias: $-\frac{6}{16} = -\frac{3}{8}$
- Probabilità che l'equazione sia vera: $\frac{1}{2} + \left(-\frac{3}{8}\right) = \frac{1}{8}$

		Output Sum															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Input Sum	0	+8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	−2	−2	0	0	−2	+6	+2	+2	0	0	+2	+2	0	0
	2	0	0	−2	−2	0	0	−2	−2	0	0	+2	+2	0	0	−6	+2
	3	0	0	0	0	0	0	0	0	+2	−6	−2	−2	+2	+2	−2	−2
	4	0	+2	0	−2	−2	−4	−2	0	0	−2	0	+2	+2	−4	+2	0
	5	0	−2	−2	0	−2	0	+4	+2	−2	0	−4	+2	0	−2	−2	0
	6	0	+2	−2	+4	+2	0	0	+2	0	−2	+2	+4	−2	0	0	−2
	7	0	−2	0	+2	+2	−4	+2	0	−2	0	+2	0	+4	+2	0	+2
	8	0	0	0	0	0	0	0	0	−2	+2	+2	−2	+2	−2	−2	−6
	9	0	0	−2	−2	0	0	−2	−2	−4	0	−2	+2	0	+4	+2	−2
	A	0	+4	−2	+2	−4	0	+2	−2	+2	+2	0	0	+2	+2	0	0
	B	0	+4	0	−4	+4	0	+4	0	0	0	0	0	0	0	0	0
	C	0	−2	+4	−2	−2	0	+2	0	+2	0	+2	+4	0	+2	0	−2
	D	0	+2	+2	0	−2	+4	0	+2	−4	−2	+2	0	+2	0	0	+2
	E	0	+2	+2	0	−2	−4	0	+2	−2	0	0	−2	−4	+2	−2	0
	F	0	−2	−4	−2	−2	0	+2	0	0	−2	+4	−2	−2	0	+2	0

Alcune proprietà fondamentali della tabella delle approssimazioni lineari possono essere osservate:

1. **Combinazioni senza input**:
   - La probabilità che una somma di un sottoinsieme non vuoto di bit di output sia uguale a una somma che non coinvolge bit di input è esattamente $\frac{1}{2}$.
   - Motivo: Per una S-box biiettiva, qualsiasi combinazione lineare di bit di output deve avere un numero uguale di zeri e uni.
2. **Caso speciale (angolo in alto a sinistra)**:
   - La combinazione lineare che non coinvolge bit di output equivale sempre alla combinazione lineare senza bit di input.
   - Risultato:
     - Bias: $+\frac{1}{2}$
     - Valore nella tabella: $+8$ (posizione top-left)
3. **Struttura della tabella**:
   - La prima riga contiene tutti zeri, eccetto il valore più a sinistra (che è +8).
   - La prima colonna contiene tutti zeri, eccetto il valore in alto (sempre +8).
4. **Proprietà di somma**:
   - La somma di qualsiasi riga o colonna deve essere $+8$ o -8.

3.4 Costruire le Approssimazioni Lineari per il Cifrario Completo

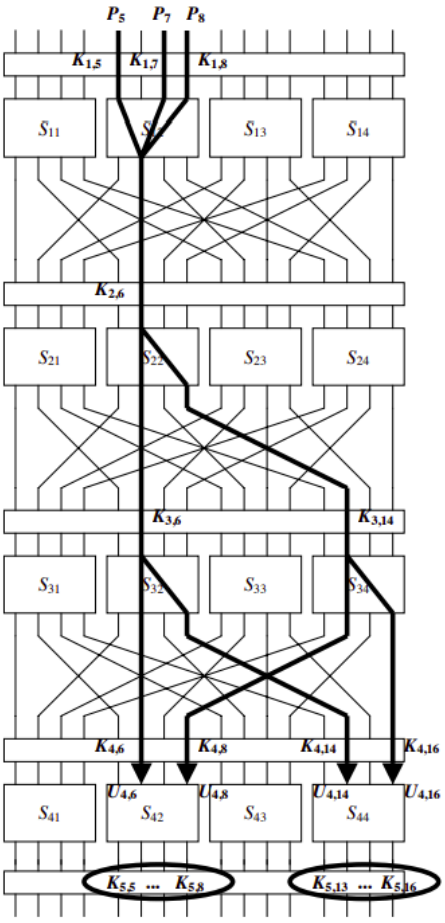
Approccio alla Crittanalisi Lineare di una SPN

Una volta compilate le informazioni di approssimazione lineare per le S-box in una SPN, abbiamo i dati necessari per determinare le approssimazioni lineari del cifrario completo nella forma dell'equazione (1). Ciò può essere ottenuto concatenando appropriate approssimazione lineari delle S-box. Costruendo un'approssimazione lineare che coinvolga i bit del plaintext e i bit dell'ouput delle S-box del penultimo round. In questo modo è possibile attaccare il cifrario recuperando un sottoinsieme dei bit della subkey utilizzata all'ultimo round.

Esempio Pratico

Consideriamo un'approssimazione che coinvolge:

S_{12}, S_{22}, S_{32}, S_{34}



Notiamo come Copre solo i primi 3 round (non tutti e 4), ciò è utile per derivare i bit della subkey finale. Usiamo le seguenti approssimazione delle S-box:

Approssimazioni Specifiche:

S-box	Approssimazione	Probabilità	Bias
S_{12}	$X_1 \oplus X_3 \oplus X_4 = Y_2$	12/16	+1/4
S_{22}	$X_2 = Y_2 \oplus Y_4$	4/16	-1/4
S_{32}	$X_2 = Y_2 \oplus Y_4$	4/16	-1/4
S_{34}	$X_2 = Y_2 \oplus Y_4$	4/16	-1/4

Notazione Tecnica

- U_i **(**V_i**)**: Blocco di 16 bit all'input (output) delle S-box del round *i*
- $U_{i,j}$ **(**$V_{i,j}$**)**: j-esimo bit del blocco U_i (V_i)
- K_i: Blocco di bit della subkey (XOR all'input del round *i*). Eccezione: K_5 è la chiave XOR all'output del round 4

Pertanto, possiamo esprimere:

$$U_1 = P \oplus K_1$$

dove P rappresenta il blocco di 16 bit del plaintext.

Utilizzando l'approssimazione lineare del 1° round, otteniamo:

$$V_{1,6} = U_{1,5} \oplus U_{1,7} \oplus U_{1,8} = (P_5 \oplus K_{1,5}) \oplus (P_7 \oplus K_{1,7}) \oplus (P_8 \oplus K_{1,8}) \quad (2)$$

con probabilità $\frac{3}{4}$.

Per l'approssimazione nel 2° round, abbiamo:

$$V_{2,6} \oplus V_{2,8} = U_{2,6}$$

con probabilità $\frac{1}{4}$. Poiché $U_{2,6} = V_{1,6} \oplus K_{2,6}$, possiamo derivare un'approssimazione della forma:

$$V_{2,6} \oplus V_{2,8} = V_{1,6} \oplus K_{2,6}$$

con probabilità $\frac{1}{4}$ e combinandola con l'equazione (2) (probabilità $\frac{3}{4}$) abbiamo che:

$$V_{2,6} \oplus V_{2,8} \oplus P_5 \oplus P_7 \oplus P_8 \oplus K_{1,5} \oplus K_{1,7} \oplus K_{1,8} \oplus K_{2,6} = 0 \quad (3)$$

con probabilità: $\frac{1}{2} + 2\left(\frac{3}{4} - \frac{1}{2}\right)\left(\frac{1}{4} - \frac{1}{2}\right) = \frac{3}{8}$ con un bias di: $-\frac{1}{8}$ data dal Pilinig-Up Lemma.

Nota: Si assume l'indipendenza delle approssimazioni delle S-box, ipotesi non rigorosamente corretta ma efficace in pratica in quanto funziona bene nella maggior parte dei cifrari.

Per l'approssimazione nel 3° round, abbiamo:

$$V_{3,6} \oplus V_{3,8} = U_{3,6}$$

con probabilità: $\frac{1}{4}$

$$V_{3,14} \oplus V_{3,16} = U_{3,14}$$

con probabilità: $\frac{1}{4}$. Poiché:
$U_{3,6} = V_{2,6} \oplus K_{3,6}$ e
$U_{3,14} = V_{2,8} \oplus K_{3,14}$ Otteniamo che:

$$V_{3,6} \oplus V_{3,8} \oplus V_{3,14} \oplus V_{3,16} \oplus V_{2,6} \oplus K_{3,6} \oplus V_{2,8} \oplus K_{3,14} = 0 \quad (4)$$

con probabilità: $\frac{1}{2} + 2\left(\frac{1}{4} - \frac{1}{2}\right)^2 = \frac{5}{8}$ e bias di: $+\frac{1}{8}$ (applicato il Piling-Up Lemma).

Ora combinando (3) e (4):

$$V_{3,6} \oplus V_{3,8} \oplus V_{3,14} \oplus V_{3,16} \oplus P_5 \oplus P_7 \oplus P_8 \oplus K_{1,5} \oplus K_{1,7} \oplus K_{1,8} \oplus K_{2,6} \oplus K_{3,6} \oplus K_{3,14} = 0$$

Sostituendo:

$$U_{4,6} = V_{3,6} \oplus K_{4,6}$$

$$U_{4,8} = V_{3,14} \oplus K_{4,8}$$

$$U_{4,14} = V_{3,8} \oplus K_{4,14}$$

$$U_{4,16} = V_{3,16} \oplus K_{4,16}$$

Formulazione finale:

$$U_{4,6} \oplus U_{4,8} \oplus U_{4,14} \oplus U_{4,16} \oplus P_5 \oplus P_7 \oplus P_8 \oplus \sum K = 0$$

dove:

$$\sum K = K_{1,5} \oplus K_{1,7} \oplus K_{1,8} \oplus K_{2,6} \oplus K_{3,6} \oplus K_{3,14} \oplus K_{4,6} \oplus K_{4,8} \oplus K_{4,14} \oplus K_{4,16}$$

con probabilità: $\frac{1}{2} + 2^3 \left(\frac{3}{4} - \frac{1}{2}\right) \left(\frac{1}{4} - \frac{1}{2}\right)^3 = \frac{15}{32}$
(e bias: $-\frac{1}{32}$)

La sommatoria è fissata a 0 o a 1 in base alla chiave del cifrario, poiché fissata notiamo che:

$$U_{4,6} \oplus U_{4,8} \oplus U_{4,14} \oplus U_{4,16} \oplus P_5 \oplus P_7 \oplus P_8 = 0 \quad (5)$$

presenta due possibili scenari probabilistici:

- Probabilità $\frac{15}{32}$ se la sommatoria è = 0
- Probabilità $(1 - \frac{15}{32}) = \frac{17}{32}$ se la sommatoria è = 1.

Abbiamo quindi un'approssimazione lineare dei primi tre roudn con un bias di magnitudo: $\frac{1}{32}$.

3.5 Trovare i Bit della chiave

Una volta scoperta un'approssimazione lineare per $R - 1$ round di un cifrario a R round con un bias di probabilità sufficientemente grande, è possibile attaccare il cifrario recuperando i bit della subkey finale. Nel caso del nostro cifrario esempio, possiamo estrarre bit dalla subkey K_5 utilizzando un'approssimazione lineare di 3 round. Chiameremo **target partial subkey** i bit da recuperare della subkey finale. Nello specifico, questi bit sono quelli della subkey finale associati alle S-box nell'ultimo round influenzate dai bit di dati coinvolti nell'approssimazione lineare.

Processo di Attacco

Il processo di attacco include una decodifica parziale dell'ultimo round del cifrario:
Per tutti i possibili valori della target partial subkey:

Si applica XOR tra i bit del ciphertext e i bit della target partial subkey e il risultato lo si esegue all'indietro attraverso le corrispondenti S-box. Ciò è eseguito per ogni chipertext/plaintext e si mantiene un contatore per ogni valore della target partial subkey.

Il contatore viene incrementato quando l'espressione lineare risulta vera per:

- I bit di input alle S-box dell'ultimo round (determinati dalla decrittazione parziale).
- I bit di plaintext noti.

Il valore della target partial subkey con il conteggio che si discosta dalla metà dei campioni di chipertext/plaintext viene assunto come corretto.

Subkey corretta: Ciò funziona perché si assume che il corretto valore della subkey parziale farà si che l'approssimazione lineare si valida con una probabilità significativamente diversa da $1/2$.

Subkey errata: Si presume che una sottochiave errata porti a un'ipotesi relativamente casuale dei bit che entrano negli S-box dell'ultimo round e, di conseguenza, l'espressione lineare sarà valida con una probabilità prossima a $1/2$.

Applicazione al Nostro Esempio

L'espressione lineare (5) coinvolge gli input delle S-box S_{42} e S_{44} nell'ultimo round. Per ogni campione plaintext/ciphertext:

1. Si testano tutti i 256 valori per la target partial subkey:

$$[K_{5,5}...K_{5,8}, K_{5,13}...K_{5,16}]$$

2. Per ogni valore della sottochiave parziale: Si incrementa il contatore quando (5) è vera dove si calcolano $[U_{4,5}...U_{4,8}, U_{4,13}...U_{4,16}]$ decrittando parzialmente attraverso S_{42} e S_{44}
3. Il valore che diverge di più dalla metà dei campioni di chipertext/plaintext si assume sia il valore corretto:
   - Se $\sum K = 0$: probabilità $< 1/2$
   - Se $\sum K = 1$: probabilità $> 1/2$

ESEMPIO DI SIMULAZIONE (16)

Simulazione dell'attacco al cifrario esempio generando 10.000 coppie plaintext/ciphertext note e seguendo il processo descritto per i valori parziali di subkey:

- $[K_{5,5}...K_{5,8}] = [0010]$ (hex 2)
- $[K_{5,13}...K_{5,16}] = [0100]$ (hex 4)

Il conteggio che si discosta maggiormente da 5.000 corrisponde al valore target `[2,4]` esadecimale. Questo conferma il successo nel derivare i bit della subkey.

partial subkey $[K_{5,5}...K_{5,8}, K_{5,13}...K_{5,16}]$	| bias |	*partial subkey* $[K_{5,5}...K_{5,8}, K_{5,13}...K_{5,16}]$	| bias |
1 C	0.0031	2 A	0.0044
1 D	0.0078	2 B	0.0186
1 E	0.0071	2 C	0.0094
1 F	0.0170	2 D	0.0053
2 0	0.0025	2 E	0.0062
2 1	0.0220	2 F	0.0133
2 2	0.0211	3 0	0.0027
2 3	0.0064	3 1	0.0050
2 4	**0.0336**	3 2	0.0075
2 5	0.0106	3 3	0.0162
2 6	0.0096	3 4	0.0218
2 7	0.0074	3 5	0.0052
2 8	0.0224	3 6	0.0056
2 9	0.0054	3 7	0.0048

Il valore del bias determinato (0.0336) risulta estremamente vicino al valore teorico atteso di $1/32 = 0.03125$.

Le discrepanze osservate tra i risultati sperimentali e le previsioni teoriche possono essere attribuite a diversi fattori:

- Le proprietà delle S-box.
- L'assunzione di indipendenza necessaria per l'applicazione del Piling-Up Lemma e l'influenza delle linear hulls.

3.6 Complessità dell'attacco

Definizione di S-box Attive

Chiamiamo **S-box attive** quelle coinvolte nell'approssimazione lineare. Nell'esempio, le quattro S-box nei round 1-3 influenzate dalle linee evidenziate sono attive. La probabilità che un'espressione lineare sia valida è legata al **Bias delle probabilità lineari** nelle S-box attive e al **Numero di S-box attive**. In generale, maggiore è l'entità della distorsione nelle S-box, maggiore è l'entità della distorsione dell'espressione complessiva. Inoltre, minore è il numero di S-box attive, maggiore è l'entità della distorsione dell'espressione lineare complessiva.

Sia ϵ il bias da $1/2$ della probabilità dell'espressione lineare valida per il cifrario completo. Il numero di plaintext noti necessari per l' attacco è proporzionale a ϵ^{-2}. È ragionevole approssimare N_L (numero di plaintext conosciuti richiesti) a:

$$N_L \approx \frac{1}{\epsilon^2}$$

In pratica, è generalmente ragionevole aspettarsi che sia richiesto un piccolo multiplo di ϵ^2 testi in chiaro noti. Sebbene, in senso stretto, la complessità della crittoanalisi potrebbe essere caratterizzata sia in termini di tempo che di spazio, faremo riferimento solo ai dati necessari per sferrare l'attacco in quanto assumiamo che se siamo in grado di acquisire N_L testi in chiaro, siamo anche in grado di elaborarli.

Poiché il bias è derivato usando il Piling-Up Lemma, dove ogni termine nel prodotto si riferisce a un'approssimazione di una S-box, è facile vedere che il bias dipende dal bias delle approssimazioni lineari delle S-box e dal numero di S-box attive coinvolte. Approcci generali per fornire sicurezza contro la crittoanalisi lineare si sono concentrati sull'ottimizzazione delle S-box (cioè, minimizzando la distorsione massima) e sulla ricerca di strutture per massimizzare il numero di S-box attive.

Realtà:

Bisogna tuttavia fare attenzione: il concetto di una "prova" di sicurezza alla crittoanalisi lineare si basa solitamente sulla non esistenza di approssimazioni lineari altamente probabili. Tuttavia, il calcolo della probabilità di tali approssimazioni lineari si fonda sull'assunzione che ogni

approssimazione di S-box sia indipendente (in modo che si possa usare il Piling-Up Lemma) e sull'assunzione che uno scenario di approssimazione lineare (cioè, un particolare insieme di S-box attive) sia sufficiente a determinare la migliore espressione lineare tra i bit del plaintext e i bit di dati in input all'ultimo round.

La realtà è che le approssimazioni delle S-box non sono indipendenti e ciò può avere un impatto significativo sul calcolo della probabilità. Inoltre, scenari di approssimazione lineare che coinvolgono gli stessi bit del plaintext e dell'input dell'ultimo round ma diversi insiemi di S-box attive possono combinarsi per dare una probabilità lineare superiore a quella prevista da un singolo insieme di S-box attive. Questo concetto è indicato come "linear hull".
Un numero di scenari di approssimazione lineare possono avere bias molto piccoli e, presi singolarmente, sembrano implicare che una cifratura potrebbe essere immune a un attacco lineare. Tuttavia, quando questi scenari vengono combinati, l'espressione lineare risultante tra i bit del plaintext e dell'ingresso dell'ultimo round potrebbe avere un bias molto elevato. Ciononostante, l'approccio usato in questo articolo tende a funzionare bene per molte cifrature perché l'assunzione di indipendenza è un'approssimazione ragionevole e quando uno scenario di approssimazione lineare di un particolare insieme di S-box attive ha un'alto bias, tende a dominare il "linear hull".

4. Crittoanalisi Differenziale

4.1 Panoramica dell'Attacco

La crittanalisi differenziale sfrutta l'alta probabilità di certe occorrenze di differenze di plaintext e differenze nell'ultimo round del cifrario. Ad esempio, consideriamo un sistema con input $X = [X_1X_2...X_n]$ e output $Y = [Y_1Y_2...Y_n]$. Siano due input del sistema X' e X'' con i corrispondenti output Y' e Y'', rispettivamente. La differenza di input è data da $\Delta X = X' \oplus X''$, quindi,

$$\Delta X = [\Delta X_1 \Delta X_2 ... \Delta X_n]$$

dove $\Delta X_i = X'_i \oplus X''_i$ con X'_i e X''_i che rappresentano l'i-esimo bit di X' e X'', rispettivamente. Analogamente, $\Delta Y = Y' \oplus Y''$ è la differenza di output e

$$\Delta Y = [\Delta Y_1 \Delta Y_2 ... \Delta Y_n]$$

dove $\Delta Y_i = Y'_i \oplus Y''_i$.

In un cifrario idealmente randomizzato, la probabilità che una particolare differenza di output ΔY si verifichi data una particolare differenza di input ΔX è $1/2^n$, dove n è il numero di bit di X. La crittanalisi differenziale cerca di sfruttare uno scenario in cui un particolare ΔY si verifica data una particolare differenza di input ΔX con una probabilità molto alta p_D (cioè molto maggiore di $1/2^n$). La coppia $(\Delta X, \Delta Y)$ è chiamata **differenziale**.

La crittanalisi differenziale è un attacco con plaintext scelto, il che significa che l'attaccante è in grado di selezionare gli input ed esaminare gli output nel tentativo di derivare la chiave. Per la crittanalisi differenziale, l'attaccante selezionerà coppie di input, X' e X'', per soddisfare un particolare ΔX, sapendo che per quel valore di ΔX, un particolare valore ΔY si verifica con alta probabilità.

La costruzione di un differenziale $(\Delta X, \Delta Y)$ che coinvolge i bit del plaintext rappresentati da X e l'input dell'ultimo round del cifrario rappresentato da Y. Si esaminano le caratteristiche differenziali altamente probabili, dove una **caratteristica differenziale** è una sequenza di differenze di input e output dei round tale che la differenza di output di un round corrisponde alla differenza di input del round successivo. Utilizzando la caratteristica differenziale altamente probabile, abbiamo l'opportunità di sfruttare le informazioni che arrivano nell'ultimo round del cifrario per derivare bit dall'ultimo livello di subkey.

Come per la crittanalisi lineare, per costruire caratteristiche differenziali altamente probabili, esaminiamo le proprietà dei singoli S-box e utilizziamo queste proprietà per determinare la caratteristica differenziale completa. Nello specifico, consideriamo le differenze di input e output degli S-box per determinare una coppia di differenze ad alta probabilità.

Combinando le coppie di differenze degli S-box da un round all'altro in modo che i bit di differenza di output non nulli di un round corrispondano ai bit di differenza di input non nulli del round successivo, possiamo trovare un differenziale ad alta probabilità costituito dalla differenza del plaintext e dalla differenza dell'input dell'ultimo round.

I bit delle subkey del cifrario finiscono per scomparire dall'espressione della differenza perché sono coinvolti in entrambi gli insiemi di dati e, quindi, considerando la loro influenza sulla differenza implica l'operazione di XOR tra bit di subkey con se stessi, il cui risultato è zero.

4.2 Analizzarndo le Componenti del Cifrario

Esaminiamo ora le coppie di differenze di un S-box. Consideriamo la rappresentazione di un S-box 4×4, con input $X = [X_1X_2X_3X_4]$ e output $Y = [Y_1Y_2Y_3Y_4]$. Tutte le coppie di differenze di un S-box, $(\Delta X, \Delta Y)$, possono essere esaminate e la probabilità di ΔY dato ΔX può essere derivata considerando coppie di input (X', X'') tali che $X' \oplus X'' = \Delta X$. Poiché l'ordine della coppia non è rilevante, per un S-box 4×4 è sufficiente considerare tutti i 16 valori possibili per X', dopodiché il valore di ΔX vincola il valore di X'' a essere $X'' = X' \oplus \Delta X$.

Considerando l'S-box del nostro cifrario, possiamo derivare i valori risultanti di ΔY per ogni coppia di input $(X', X'' = X' \oplus \Delta X)$. Ad esempio, i valori binari di X, Y e i corrispondenti valori di ΔY per le coppie di input $(X, X \oplus \Delta X)$ sono presentati nella Tabella per valori di ΔX pari a 1011

(esa B), 1000 (esa 8) e 0100 (esa 4).

X	Y	ΔY		
		$\Delta X = 1011$	$\Delta X = 1000$	$\Delta X = 0100$
0000	1110	0010	1101	1100
0001	0100	0010	1110	1011
0010	1101	0111	0101	0110
0011	0001	0010	1011	1001
0100	0010	0101	0111	1100
0101	1111	1111	0110	1011
0110	1011	0010	1011	0110
0111	1000	1101	1111	1001
1000	0011	0010	1101	0110
1001	1010	0111	1110	0011
1010	0110	0010	0101	0110
1011	1100	0010	1011	1011
1100	0101	1101	0111	0110
1101	1001	0010	0110	0011
1110	0000	1111	1011	0110
1111	0111	0101	1111	1011

Le ultime tre colonne della tabella rappresentano i valori di ΔY per il valore di X (indicato nella riga) e il particolare valore di ΔX di ogni colonna. Dalla tabella, possiamo osservare che il numero di occorrenze di $\Delta Y = 0010$ per $\Delta X = 1011$ è 8 su 16 valori possibili (cioè una probabilità di $8/16$); il numero di occorrenze di $\Delta Y = 1011$ dato $\Delta X = 1000$ è 4 su 16; il numero di occorrenze di $\Delta Y = 1010$ dato $\Delta X = 0100$ è 0 su 16. Se l'S-box fosse "ideale", il numero di occorrenze dei valori delle coppie di differenze sarebbe sempre 1, dando una probabilità di $1/16$ che si verifichi un particolare valore di ΔY dato ΔX. (Una S-box "ideale" non è matematicamente possibile.)

Possiamo tabulare i dati completi di un S-box in una **tabella di distribuzione delle differenze**, in cui le righe rappresentano i valori di ΔX e le colonne rappresentano i valori di ΔY (in esadeciamel). La tabella di distribuzione delle differenze per l'S-box della usata nel capitolo 2 è fornita nella Tabella sottostante.

		Output Difference															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Input Difference	0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	0	2	0	0	0	2	0	2	4	0	4	2	0	0
	2	0	0	0	2	0	6	2	2	0	2	0	0	0	0	2	0
	3	0	0	2	0	2	0	0	0	0	4	2	0	2	0	0	4
	4	0	0	0	2	0	0	6	0	0	2	0	4	2	0	0	0
	5	0	4	0	0	0	2	2	0	0	0	4	0	2	0	0	2
	6	0	0	0	4	0	4	0	0	0	0	0	0	2	2	2	2
	7	0	0	2	2	2	0	2	0	0	2	2	0	0	0	0	4
	8	0	0	0	0	0	0	2	2	0	0	0	4	0	4	2	2
	9	0	2	0	0	2	0	0	4	2	0	2	2	2	0	0	0
	A	0	2	2	0	0	0	0	0	6	0	0	2	0	0	4	0
	B	0	0	8	0	0	2	0	2	0	0	0	0	0	2	0	2
	C	0	2	0	0	2	2	2	0	0	0	0	2	0	6	0	0
	D	0	4	0	0	0	0	0	4	2	0	2	0	2	0	2	0
	E	0	0	2	4	2	0	0	0	6	0	0	0	0	0	2	0
	F	0	2	0	0	6	0	0	0	0	4	0	2	0	0	2	0

Ogni elemento della tabella rappresenta il numero di occorrenze del corrispondente valore di differenza di output ΔY dato il valore di differenza di input ΔX. Si noti che, a parte il caso speciale $(\Delta X = 0, \Delta Y = 0)$, il valore più grande nella tabella è 8, corrispondente a $\Delta X = \mathrm{B}$ e $\Delta Y = 2$. Pertanto, la probabilità che $\Delta Y = 2$ dato una coppia arbitraria di valori di input che soddisfano $\Delta X = \mathrm{B}$ è $8/16$. Il valore più piccolo nella tabella è 0 e si verifica per molte coppie di differenze. In questo caso, la probabilità che il valore ΔY si verifichi dato il valore ΔX è 0.

Esistono diverse proprietà generali della tabella di distribuzione delle differenze che è importante menzionare.

1. **Somma degli elementi**:
   La somma di tutti gli elementi in una riga è $2^n = 16$; allo stesso modo, la somma di qualsiasi colonna è $2^n = 16$.
2. **Elementi pari**:
   Tutti i valori nella tabella sono pari. Questo deriva dal fatto che una coppia di valori di input (o output) rappresentata come (X', X'') ha lo stesso valore ΔX della coppia (X'', X'), poiché:

$$\Delta X = X' \oplus X'' = X'' \oplus X'.$$

3. **Caso speciale $\Delta X = 0$**:
   Una differenza di input $\Delta X = 0$ deve necessariamente portare a una differenza di output $\Delta Y = 0$ a causa della mappatura biunivoca dell'S-box. Di conseguenza:
   - L'angolo in alto a destra della tabella ha un valore di $2^n = 16$ (tutte le occorrenze).
   - Tutti gli altri valori nella prima riga e nella prima colonna sono 0.
4. **S-box ideale**:
   Se fosse possibile costruire un S-box "ideale" che non fornisca alcuna informazione differenziale sull'output dato l'input, tutti gli elementi della tabella sarebbero uguali a 1, e la probabilità che un particolare valore ΔY si verifichi dato un valore ΔX sarebbe:

$$\frac{1}{2^n} = \frac{1}{16}.$$

   Tuttavia, a causa delle proprietà discusse sopra, ciò non è realizzabile.

Influenza della chiave sul differenziale dell'S-box

Prima di procedere a combinare le coppie di differenze degli S-box per derivare una caratteristica differenziale, dobbiamo considerare l'effetto della chiave sul comportamento differenziale dell'S-box.

- **Input "senza chiave" vs. "con chiave"**:
  Nell'S-box "senza chiave" (Figura sottostante), l'input è X e l'output è Y. Tuttavia, nella struttura del cifrario, dobbiamo tenere conto delle sottochiavi applicate all'input di ogni S-box.

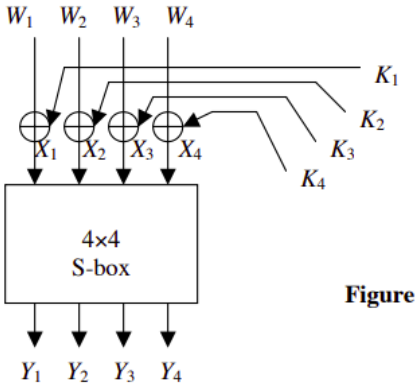


Figure 4. Keyed S-box

- **Input con chiave**:
  Se indichiamo l'input all'S-box "con chiave" come $W = [W_1 W_2 W_3 W_4]$, la differenza di input per l'S-box con chiave sarà:

$$\Delta W = [\Delta W_1 \Delta W_2 ... \Delta W_n],$$

  dove:

$$\Delta W_i = W_i' \oplus W_i'',$$

  e:

$$W' = [W_1' W_2' ... W_n'], \quad W'' = [W_1'' W_2'' ... W_n'']$$

  rappresentano i due valori di input.
  Poiché i bit della chiave rimangono gli stessi sia per W' che per W'', abbiamo:

$$\begin{aligned} \Delta W_i &= W_i' \oplus W_i'' \\ &= (X_i' \oplus K_i) \oplus (X_i'' \oplus K_i) \\ &= X_i' \oplus X_i'' \oplus K_i \oplus K_i \\ &= X_i' \oplus X_i'' \quad (\text{poiché } K_i \oplus K_i = 0) \\ &= \Delta X_i \end{aligned}$$

Pertanto, i bit della chiave non influenzano il valore della differenza di input e possono essere ignorati. In altre parole, l'S-box con chiave ha la stessa tabella di distribuzione delle differenze dell'S-box senza chiave.

4.3 Costruire le Caratteristiche Differenziali

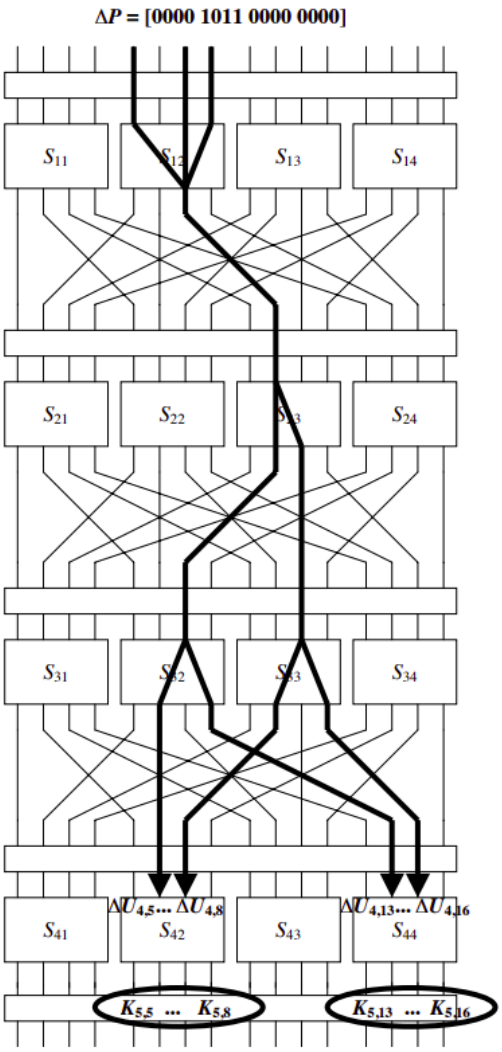
Una volta compilate le informazioni differenziali per gli S-box in una rete SPN, disponiamo dei dati necessari per determinare una caratteristica differenziale utile del cifrario nel suo complesso. Questo si ottiene concatenando opportune coppie di differenze degli S-box.

Costruzione di una caratteristica differenziale

Costruendo una caratteristica differenziale che coinvolga specifiche coppie di differenze degli S-box in ogni round - in modo che il differenziale includa bit del plaintext e bit in input agli S-box dell'ultimo round - è possibile attaccare il cifrario recuperando un sottoinsieme dei bit della sottochiave successiva all'ultimo round. Illustriamo questa costruzione con un esempio.

Consideriamo una caratteristica differenziale che coinvolge gli S-box S_{12}, S_{23}, S_{32} e S_{33}. Come nel caso della crittanalisi lineare, è utile visualizzare la caratteristica differenziale attraverso in diagramma, che mostra:

- L'influenza delle differenze non nulle nei bit mentre attraversano la rete
- Gli S-box "attivi" (cioè quelli con differenza non nulla)



Nota: questa caratteristica copre solo i primi 3 round del cifrario, non tutti i 4 round. Ciò sarà utile per derivare bit dell'ultima sottochiave.

Coppie di differenze utilizzate

Usiamo le seguenti coppie di differenze:

- S_{12}: $\Delta X = \mathrm{B} \rightarrow \Delta Y = 2$ con probabilità $8/16$
- S_{23}: $\Delta X = 4 \rightarrow \Delta Y = 6$ con probabilità $6/16$
- S_{32}: $\Delta X = 2 \rightarrow \Delta Y = 5$ con probabilità $6/16$
- S_{33}: $\Delta X = 2 \rightarrow \Delta Y = 5$ con probabilità $6/16$

Tutti gli altri S-box avranno differenza di input zero e quindi differenza di output zero.

Differenze attraverso i round

La differenza di input al cifrario equivale alla differenza di input del primo round ed è data da:

$$\Delta P = \Delta U_1 = [0000\ 1011\ 0000\ 0000]$$

dove:

- U_i per rappresentare l'input agli S-box dell'i-esimo round
- V_i per rappresentare l'output degli S-box dell'i-esimo round

Pertanto, ΔU_i e ΔV_i rappresentano le corrispondenti differenze. Di conseguenza:

$$\Delta V_1 = [0000\ 0010\ 0000\ 0000]$$

considerando la coppia di differenze per S_{12} elencata sopra e applicando la permutazione del primo round:

$$[0000\ 0000\ 0100\ 0000]$$

con una probabilità di $8/16 = 1/2$ data la differenza di plaintext ΔP.

Ora, il differenziale del secondo round utilizzando la coppia di differenze per S_{23} produce:

$$\Delta V_2 = [0000\ 0000\ 0110\ 0000]$$

e la permutazione del secondo round dà:

$$\Delta U_3 = [0000\ 0010\ 0010\ 0000]$$

con probabilità $6/16$ dato ΔU_2, e una probabilità complessiva di $8/16 \times 6/16 = 3/16$ dato ΔP. Nel determinare questa probabilità, abbiamo assunto che:

1. Il differenziale del primo round sia indipendente da quello del secondo round
2. La probabilità combinata sia quindi il prodotto delle probabilità singole

Proseguendo, possiamo usare le differenze per gli S-box del terzo round (S_{32} e S_{33}) e la permutazione del terzo round per ottenere:

$$\Delta V_3 = [0000\ 0101\ 0101\ 0000]$$

e infine:

$$\Delta U_4 = [0000\ 0110\ 0000\ 0110] \quad (6)$$

con probabilità $(6/16)^2$ dato ΔU_3, e quindi una probabilità complessiva di:
$\frac{8}{16} \times \frac{6}{16} \times \left(\frac{6}{16}\right)^2 = \frac{27}{1024}$.
Data la differenza di plaintext ΔP, dove abbiamo nuovamente assunto l'indipendenza tra le coppie di differenze degli S-box in tutti i round.

Analisi crittografica

Durante il processo di crittanalisi:

- Verranno cifrate molte coppie di plaintext con $\Delta P = [0000\ 1011\ 0000\ 0000]$
- Con alta probabilità ($27/1024$) si verificherà la caratteristica differenziale descritta

Definiamo:

- **Right pairs**: Coppie per cui la caratteristica differenziale si verifica
- **Wrong pairs**: Coppie per cui la caratteristica non si verifica

4.4 Trovare i Bit della chiave

Quando si scopre una caratteristica differenziale di $R - 1$ round per un cifrario a R round con probabilità sufficientemente alta, diventa possibile attaccare il cifrario recuperando bit dalla sottochiave finale. Nel caso del nostro cifrario d'esempio, è possibile estrarre bit dalla sottochiave K_5.

Processo di attacco

Il metodo prevede:

1. **Decrittazione parziale** dell'ultimo round
2. **Analisi dell'input** all'ultimo round per identificare probabili *right pairs*

Chiameremo **target partial subkey** (sottochiave parziale target) i bit della sottochiave che seguono l'ultimo round nell'output delle S-box dell'utlimo round e che sono influenzati da differenze non nulle nell'output differenziale.

Implementazione pratica

La decrittazione parziale dell'ultimo round richiede, per tutte le S-boxes nell'ultimo round influenzate da differenze non nulle:

1. Operazioni di **XOR** tra il ciphertext e i bit della sottochiave target
2. Esecuzione **all'indietro** attraverso gli S-box dove vengono testati **tutti i valori possibili** per la target subkey.

Per ogni coppia di ciphertext corrispondente alle coppie di plaintext utilizzate per generare la differenza di input ΔP, viene eseguita una decrittazione parziale per tutti i possibili valori della partial target subkey. Per ogni valore della partial target subkey viene mantenuto un contatore. Il contatore viene incrementato quando la differenza per l'input dell'ultimo round corrisponde al valore atteso dalla caratteristica differenziale. Si assume che il valore della sottochiave parziale con il conteggio più alto indichi i valori corretti dei bit della sottochiave.

Questo metodo funziona perché si assume che il valore corretto della sottochiave parziale porterà frequentemente alla differenza attesa nell'ultimo round (cioè al verificarsi di un *right pair*), dato che la caratteristica ha un'alta probabilità di occorrenza. (Quando si verifica un *wrong pair*, anche con la decrittazione parziale usando la sottochiave corretta, è probabile che il contatore per la sottochiave corretta non venga incrementato.) Si assume invece che una sottochiave incorretta produca un risultato casuale per i bit in input agli S-box dell'ultimo round, e di conseguenza la differenza corrisponderà a quella attesa dalla caratteristica con una probabilità molto bassa.

Considerando l'attacco al nostro cifrario d'esempio, la caratteristica differenziale influenza gli input degli S-box S_{42} e S_{44} nell'ultimo round. Per ogni coppia di ciphertext, proveremo tutti i 256 valori per $[K_{5,5}...K_{5,8}, K_{5,13}...K_{5,16}]$. Per ogni valore della sottochiave parziale, incrementeremo il contatore ogni volta che la differenza di input al round finale determinata dalla decrittazione parziale è uguale a (6), dove determiniamo il valore di $[\Delta U_{4,5}...\Delta U_{4,8}, \Delta U_{4,13}...\Delta U_{4,16}]$ facendo elaborare i dati all'indietro attraverso la sottochiave parziale e gli S-box S_{42} e S_{44}. Per ogni valore della sottochiave parziale, il contatore rappresenta il numero di occorrenze di differenze consistenti con *right pairs* (assumendo che il valore della sottochiave parziale sia corretto). Il conteggio più alto viene considerato come valore corretto, poiché assumiamo di osservare l'occorrenza ad alta probabilità del *right pair*.

Si noti che non è necessario eseguire la decrittazione parziale per ogni coppia di ciphertext. Poiché la differenza di input all'ultimo round influenza solo 2 S-box, quando la caratteristica si verifica (cioè per i *right pairs*), le differenze nei bit di ciphertext corrispondenti agli S-box S_{41} e S_{43} devono essere zero. Pertanto, possiamo filtrare molti *wrong pairs* scartando le coppie di ciphertext per cui non compaiono zeri nei sotto-blocchi appropriati della differenza di ciphertext. In questi casi, poiché la coppia di ciphertext non può corrispondere a un *right pair*, non è necessario esaminare $[\Delta U_{4,5}...\Delta U_{4,8}, \Delta U_{4,13}...\Delta U_{4,16}]$.

Simulazione dell'attacco

Simulazione di attacco al nostro cifrario di base, cifrato usando sottochiavi generate casualmente, creando 5000 coppie plaintext/ciphertext scelte (cioè 10000 cifrature con coppie di plaintext che soddisfano $\Delta P = [0000\ 1011\ 0000\ 0000]$) e seguendo il processo descritto sopra.

Il valore corretto della sottochiave parziale target era:

$$[K_{5,5}...K_{5,8}, K_{5,13}...K_{5,16}] = [0010, 0100] = [2, 4]_{hex}$$

Come previsto, il conteggio più alto è stato osservato per il valore di sottochiave parziale $[2, 4]_{hex}$, confermando che l'attacco ha derivato con successo i bit della sottochiave.

partial subkey $[K_{5,5}...K_{5,8}, K_{5,13}...K_{5,16}]$	prob	*partial subkey* $[K_{5,5}...K_{5,8}, K_{5,13}...K_{5,16}]$	prob
1 C	0.0000	2 A	0.0032
1 D	0.0000	2 B	0.0022
1 E	0.0000	2 C	0.0000
1 F	0.0000	2 D	0.0000
2 0	0.0000	2 E	0.0000
2 1	0.0136	2 F	0.0000
2 2	0.0068	3 0	0.0004
2 3	0.0068	3 1	0.0000
2 4	**0.0244**	3 2	0.0004
2 5	0.0000	3 3	0.0004
2 6	0.0068	3 4	0.0000
2 7	0.0068	3 5	0.0004
2 8	0.0030	3 6	0.0000
2 9	0.0024	3 7	0.0008

La Tabella 8
evidenzia un riepilogo parziale dei dati derivati dai conteggi delle sottochiavi. (I dati completi
contengono 256 voci di dati, una per ciascun valore parziale di sottochiave.)

I valori nella tabella indicano la probabilità stimata di occorrenza di *right pairs* per la sottochiave parziale candidata, calcolata come:

$$\text{prob} = \frac{\text{count}}{5000}$$

dove "count" è il conteggio corrispondente al specifico valore di sottochiave parziale.

Dai risultati campione mostrati in tabella, si osserva che:

- La probabilità massima si verifica per $[K_{5,5}...K_{5,8}, K_{5,13}...K_{5,16}] = [2, 4]_{hex}$
- Questa osservazione è risultata vera per l'intero set di valori di sottochiave parziale

Nel nostro esempio:

- La probabilità teorica attesa di *right pairs* era $p_D = \frac{27}{1024} = 0.0264$
- Sperimentalmente abbiamo trovato $p_D = 0.0244$ per la sottochiave corretta $[2, 4]$

Si noti che:

1. Occasionalmente si riscontrano conteggi elevati anche per partial target subkey errate
2. Ciò indica che l'esame di partial target subkey non corrette non equivale esattamente al comparare differenze casuali con il valore differenziale atteso

Vari fattori influenzano i conteggi rispetto alle aspettative teoriche:

- Proprietà degli S-box che influenzano la decrittazione parziale
- Imprecisione nell'assunzione di indipendenza necessaria per determinare la probabilità della caratteristica
- Il concetto che i differenziali sono composti da multiple caratteristiche differenziali.

4.5 Complessità dell'attacco

Nella crittanalisi differenziale, gli S-box coinvolti in una caratteristica che presentano una differenza di input non nulla (e quindi una differenza di output non nulla) vengono definiti **S-box attivi**. In generale:

1. **Probabilità differenziali**:
   Maggiori sono le probabilità differenziali degli S-box attivi, maggiore sarà la probabilità caratteristica del cifrario completo.
2. **Numero di S-box attivi**:
   Minore è il numero di S-box attivi, maggiore sarà la probabilità della caratteristica.

Complessità della crittanalisi

Analogamente alla crittanalisi lineare, quando consideriamo la complessità dell'attacco ci riferiamo ai dati necessari per condurlo. Assumiamo cioè che se siamo in grado di acquisire N_D plaintext, siamo in grado di elaborarli.

Stima del numero di coppie di plaintext

Determinare esattamente il numero di coppie di plaintext scelti necessarie per l'attacco è generalmente molto complesso. Tuttavia, si può dimostrare che una buona approssimazione per il numero di coppie richieste N_D è:

$$N_D \approx \frac{c}{p_D} \quad (7)$$

dove:

- p_D è la probabilità della caratteristica differenziale per $R - 1$ round di un cifrario a R round
- c è una costante piccola

Calcolo della probabilità della caratteristica

Assumendo che le occorrenze delle coppie di differenze in ogni S-box attivo siano indipendenti, la probabilità della caratteristica differenziale è data da:

$$p_D = \prod_{i=1}^{\gamma} \beta_i \quad (8)$$

dove:

- γ rappresenta il numero di S-box attivi
- β_i rappresenta la probabilità della specifica coppia di differenze nell'i-esimo S-box attivo della caratteristica.

Non è difficile comprendere perché l'equazione (7) sia valida. Essa indica semplicemente che sono sufficienti poche occorrenze di *right pair* per ottenere un conteggio significativamente più alto per il valore corretto della partial target subkey rispetto ai conteggi per valori errati. Poiché ci si aspetta che un *right pair* si verifichi circa ogni $1/p_D$ coppie esaminate, in pratica è generalmente ragionevole utilizzare un piccolo multiplo di $1/p_D$ coppie di plaintext scelti per condurre con successo l'attacco.

Strategie di resistenza alla crittanalisi differenziale

Gli approcci per garantire resistenza alla crittanalisi differenziale si sono concentrati su:

1. **Proprietà degli S-box**:
   - Minimizzare la probabilità delle coppie di differenze negli S-box
2. **Strutture del cifrario**:
   - Massimizzare il numero di S-box attivi

Un esempio eccellente è Rijndael (AES), progettato specificamente per offrire alta resistenza alla crittanalisi differenziale.

Avvertenze nell'analisi di sicurezza

Come per la crittanalisi lineare, è necessario esercitare cautela nel "dimostrare" l'immunità alla crittanalisi differenziale:

Il calcolo di p_D si basa sull'**assunzione di indipendenza** tra gli S-box coinvolti ma nei cifrari reali esiste una **dipendenza** tra i dati in input a diversi S-box, di conseguenza, p_D rappresenta solo una **stima**.
Tuttavia, nella pratica si è osservato che per molti cifrari questa stima risulta ragionevolmente accurata.

Importanza dei differenziali multipli

Fattore cruciale: diversi **differential characteristics** con la stessa differenza di input e output (cioè lo stesso differenziale) possono combinarsi, producendo una probabilità complessiva maggiore di quella del singolo differential characteristic. (Questo concetto è analogo agli *linear hulls*).

Per garantire sicurezza contro la crittanalisi differenziale è necessario:

- Dimostrare che la probabilità di **tutti i differenziali** sia al di sotto di una soglia accettabile
- Non basta che le probabilità delle singole caratteristiche differenziali siano sotto soglia.

Tuttavia, in generale, quando una caratteristica differenziale ha alta probabilità, domina l'occorrenza del differenziale complessivo e la sua probabilità fornisce una buona approssimazione della probabilità del differenziale.