Cifrari Simmetrici

L'impostazione simmetrica considera due parti che condividono una chiave e utilizzeranno questa chiave per conferire ai dati comunicati vari attributi di sicurezza. I principali obiettivi di sicurezza sono la privacy e l'autenticità dei dati comunicati. Qui parleremo della privacy.

Schema di cifratura simmetrica

La primitiva che considereremo è chiamata schema di cifratura. Tale schema specifica un algoritmo di cifratura, che dice al mittente come elaborare il testo in chiaro utilizzando la chiave, producendo così il testo cifrato che viene effettivamente trasmesso. Uno schema di cifratura specifica anche un algoritmo di decrittazione, che dice al destinatario come recuperare il testo in chiaro originale della trasmissione, possibilmente anche eseguendo una verifica. Infine, c'è un algoritmo di generazione della chiave, che produce una chiave che le parti devono condividere. In breve:

- L'algoritmo *KeyGen* (randomizzato) restituisce una stringa random (chiave) appartenente all'insieme K.
- L'algoritmo Enc (randomizzato o a stati) prende in input una chiave k e un messaggio m e restituisce un critto-testo c (o un simbolo speciale $\perp$).

- L'algoritmo Dec (deterministico) dall'input (k, c) produce m (o il simbolo $\perp$).

Segue la descrizione formale:

Definizione 1

Uno schema di cifratura simmetrica $SE = (KeyGen, Enc, Dec)$ è costituito da tre algoritmi, come segue:

- L'algoritmo di generazione della chiave randomizzata $KeyGen$ restituisce una stringa k. Indichiamo con $Keys(SE)$ l'insieme di tutte le stringhe che hanno la probabilità diversa da zero di essere emesse da K. I membri di questo insieme sono chiamati chiavi. Scriviamo $k \leftarrow_R K$ per l'operazione che estrae in maniera random da K una chiave k.
- L'algoritmo di cifratura Enc, che può essere randomizzato o stateful, richiede una chiave $k \in Keys(SE)$ e un testo in chiaro $m \in \{0,1\}^*$ per restituire un testo cifrato $c \in \{0,1\}^* \cup \{\perp\}$. Scriviamo $c \leftarrow Enc_k(m)$ per l'operazione che esegue Enc_k su m e ritorna c che denota il testo cifrato.
- L'algoritmo di decifratura deterministico Dec, prende una chiave $k \in Keys(SE)$ e un testo cifrato $c \in \{0,1\}^*$ per restituire $m \in \{0,1\}^* \cup \{\perp\}$. Scriviamo $m \leftarrow Dec_k(C)$ per l'operazione che esegue Dec_k su c e restituisce m che è il messaggio originale.

Si dice che lo schema fornisca una **decifratura corretta** se per ogni $k \in Keys(SE)$, qualsiasi sequenza di messaggi $M_1, \ldots, M_q \in \{0,1\}^*$ e qualsiasi sequenza di testi cifrati $C_1 \leftarrow E_k(M_1), \ldots, C_q \leftarrow E_k(M_q)$ che possono sorgere nel corso della cifratura di $M_1, \ldots, M_q$ deve valere che $D_k(M_i) = M_i, \forall C_i \neq \perp$.

L'algoritmo di generazione delle chiavi, come indica la definizione, è randomizzato. Non richiede input. Quando viene eseguito, lancia le monete internamente e le usa per selezionare una chiave k. In genere, la chiave è solo una stringa casuale di una certa lunghezza. Quando due parti vogliono utilizzare lo schema, si presume che siano in possesso di una chiave k generata tramite K.
Come siano entrati in possesso congiunto di questa chiave k in modo tale che l'avversario non abbia conosciuto k non è la nostra preoccupazione qui, e sarà affrontata in seguito. Per ora supponiamo che la chiave sia stata condivisa.
Una volta in possesso di una chiave condivisa, il mittente può eseguire l'algoritmo di cifratura con la chiave k e il messaggio di input m per ottenere una stringa che chiamiamo testo cifrato. Quest'ultimo può quindi essere trasmesso al ricevitore.
L'algoritmo di cifratura può essere randomizzato o stateful. Se randomizzato, lancia le monete e le usa per calcolare il suo output su un dato input k, m. Ogni volta che viene invocato l'algoritmo, lancia di nuovo le monete. In particolare, invocare due volte l'algoritmo sugli stessi input potrebbe non produrre la stessa risposta entrambe le volte.

Diciamo che l'algoritmo di cifratura è **stateful** se il suo funzionamento dipende da una quantità chiamata *stato* che è inizializzata in un modo prestabilito. Quando l'algoritmo di cifratura viene invocato sugli input k, m, calcola un testo cifrato basato su k, m e sullo $stato$ corrente. Quindi aggiorna lo stato e il nuovo valore di stato viene archiviato. (Il ricevitore non mantiene lo stato di corrispondenza e, in particolare, la decrittazione non richiede l'accesso a nessuna variabile globale o richiede alcuna sincronizzazione tra le parti). Di solito, quando c'è uno stato da mantenere, lo stato è solo un contatore. Se non c'è stato mantenuto dall'algoritmo di cifratura, lo schema di cifratura è detto senza stato.
L'algoritmo di cifratura potrebbe essere sia randomizzato che stateful, ma in pratica questo è raro: di solito è l'uno o l'altro ma non entrambi.

Quando parliamo di uno schema di cifratura simmetrica randomizzato intendiamo che l'algoritmo di cifratura è randomizzato. Quando parliamo di uno schema di cifratura simmetrica con stato intendiamo che l'algoritmo di cifratura è con stato.
Il destinatario, dopo aver ricevuto un testo cifrato c, eseguirà l'algoritmo di decifratura con la stessa chiave utilizzata per creare il testo cifrato, ovvero esegue $Dec_K(c)$. L'algoritmo di decifratura non è né randomizzato né stateful.

Molti schemi di cifratura limitano l'insieme di stringhe che sono disposti a cifrare, ad esempio il messaggio deve essere il multiplo di un blocco di n bit, come nel caso di AES. Questi tipi di restrizioni vengono catturati facendo in modo che l'algoritmo di cifratura restituisca il simbolo speciale $\perp$ quando viene inviato un messaggio che non soddisfa la restrizione richiesta. In uno schema senza stato, c'è tipicamente un insieme di stringhe M, chiamato **spazio del testo in chiaro**, tale che:

$$m \in M \iff Pr[k \leftarrow K; C \leftarrow E_k(m) : C \neq \perp] = 1$$

In uno schema stateful, se $Enc_k(M)$ restituisce o meno $\perp$ dipende non solo da M ma anche possibilmente dal valore della variabile di stato. Ad esempio, quando viene utilizzato un contatore, è tipico che vi sia un limite al numero di cifrature eseguite e quando il contatore raggiunge un certo valore l'algoritmo di cifratura restituisce $\perp$ indipendentemente dal messaggio che gli viene inviato.

Esempi di cifrari simmetrici

Di seguito vedremo alcuni esempi di cifrari simmetrici. In particolare ci concentreremo sui cosiddetti **modi d'operazione**. Tali meccanismi si basano su famiglie di permutazioni e stabiliscono come utilizzare le permutazioni per generare il critto-testo. Supporremo (per semplicità) che la taglia del messaggio è un multiplo della taglia del blocco.

Modo ECB (Electronic Codebook Mode)

Sia $E : K \times \{0,1\}^n \to \{0,1\}^n$ un cifrario a blocchi. ECB produce un cifrario senza stati, infatti è deterministico. L'algoritmo di generazione della chiave si limita a restituire una stringa random.

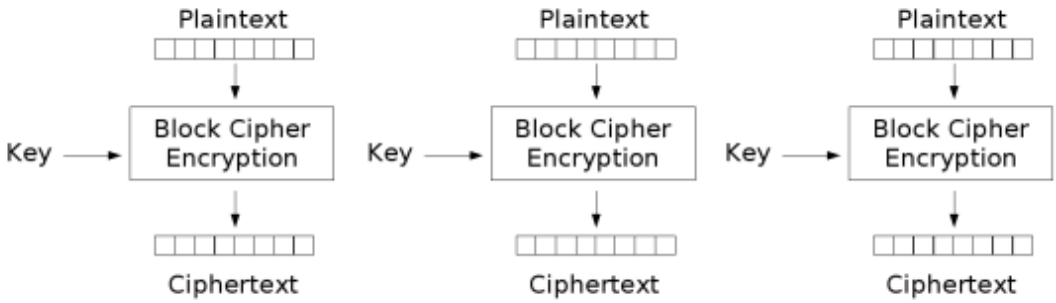
Gli algoritmi di cifratura e decifratura sono i seguenti:

algorithm $\mathcal{E}_K(M)$
  if $(|M| \bmod n \neq 0$ or $|M| = 0)$ **then return** $\perp$
  Break M into n-bit blocks $M[1] \cdots M[m]$
  for $i \leftarrow 1$ to m **do**
    $C[i] \leftarrow E_K(M[i])$
  $C \leftarrow C[1] \cdots C[m]$
  return C

---

algorithm $\mathcal{D}_K(C)$
  if $(|C| \bmod n \neq 0$ or $|C| = 0)$ **then return** $\perp$
  Break C into n-bit blocks $C[1] \cdots C[m]$
  for $i \leftarrow 1$ to m **do**
    $M[i] \leftarrow E_K^{-1}(C[i])$
  $M \leftarrow M[1] \cdots M[m]$
  return M

"Break M into n-bit blocks M[1], . . ., M[m]" significa porre $m = |M|/n$ e per $i \in \{1, \ldots, m\}$, poniamo $M[i]$ all'i-esimo blocco di n-bit in M. Allo stesso modo si spezza C. Si noti che questa volta l'algoritmo di cifratura non ha effettuato scelte casuali. Inoltre, si tratta di uno schema **not expanded**, se in ingresso diamo un messaggio di n blocchi in output si avrà un critto-testo di n blocchi.



Electronic Codebook (ECB) mode encryption

Modo CBC$ (Cipher block Chaining Mode)

Sia $E : K \times \{0,1\}^n \to \{0,1\}^n$ un cifrario a blocchi. CBC$, con vettore iniziale IV produce un cifrario senza stati e randomizzato. L'algoritmo di generazione della chiave si limita a restituire una stringa random.

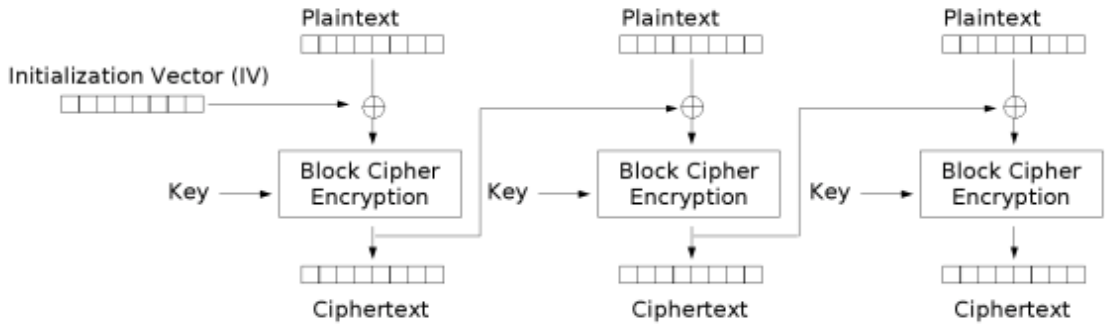
Gli algoritmi di cifratura e decifratura sono i seguenti:

algorithm $\mathcal{E}_K(M)$
if $(|M| \bmod n \neq 0$ or $|M| = 0)$ **then return** $\perp$
Break M into n-bit blocks $M[1] \cdots M[m]$
$C[0] \leftarrow \text{IV} \xleftarrow{\$} \{0,1\}^n$
for $i \leftarrow 1$ to m **do**
$C[i] \leftarrow E_K(C[i-1] \oplus M[i])$
$C \leftarrow C[1] \cdots C[m]$
return $\langle \text{IV}, C \rangle$

---

algorithm $\mathcal{D}_K(\langle \text{IV}, C \rangle)$
if $(|C| \bmod n \neq 0$ or $|M| = 0)$ **then return** $\perp$
Break C into n-bit blocks $C[1] \cdots C[m]$
$C[0] \leftarrow \text{IV}$
for $i \leftarrow 1$ to m **do**
$M[i] \leftarrow E_K^{-1}(C[i]) \oplus C[i-1])$
$M \leftarrow M[1] \cdots M[m]$
return M

IV è $C[0]$, il quale è scelto casualmente dall'algoritmo di cifratura. Questa scelta è indipendente ogni volta che l'algoritmo viene invocato.

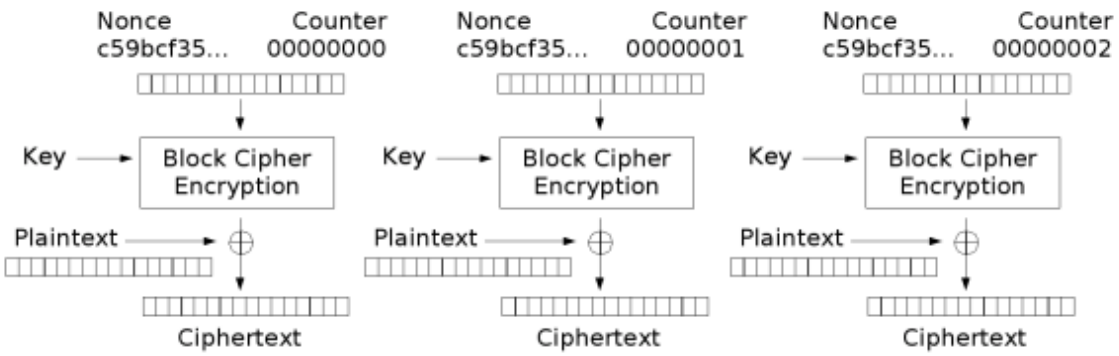


Cipher Block Chaining (CBC) mode encryption

Modo CTR (Counter)

Le modalità CTR (contatore) che seguono non sono molto utilizzate, per quanto ne sappiamo. Vedremo in seguito che hanno buone proprietà di privacy. A differenza del CBC, la procedura di cifratura è parallelizzabile, il che può essere sfruttata per velocizzare il processo in presenza di supporto hardware. È anche vero che i metodi funzionano per stringhe di lunghezze di bit

arbitrarie, senza fare nulla di "speciale" per raggiungere questo scopo. Esistono due varianti della modalità CTR, una **casuale** e l'altra **stateful**.



Counter (CTR) mode encryption

Modo CTR$ (Counter Randomizzato)

Sia $F : K \times \{0,1\}^l \to \{0,1\}^L$ una famiglia di funzioni, non necessariamente un cifrario a blocchi. CTR$ produce un cifrario senza stati e randomizzato. L'algoritmo di generazione della chiave si limita a restituire una stringa random.

Gli algoritmi di cifratura e decifratura sono i seguenti:

$\mathbf{Enc}_k(M)$

$m \leftarrow \lceil |M|/L \rceil;\ R \leftarrow_R \{0,1\}^\ell$

$Pad \leftarrow F_k(R+1)||F_k(R+2)||\ldots||F_k(R+m)$

$Pad \leftarrow$ Primi $|M|$ bits di Pad

$C' \leftarrow M \oplus Pad$

$C \leftarrow R||C'$

return C

$\mathbf{Dec}_k(C)$

if $(|C| < \ell)$ return $\perp$

Sia $C = R||C'$ $(|R| = \ell)$

$m \leftarrow \lceil |C'|/L \rceil$

$Pad \leftarrow F_k(R+1)||F_k(R+2)||\ldots||F_k(R+m)$

$Pad \leftarrow$ Primi $|C'|$ bits di Pad

$M \leftarrow C' \oplus Pad$

return M

Il punto di partenza R viene utilizzato per definire una sequenza di valori su cui viene applicato F_k per produrre uno "pseudo one-time pad" a cui viene eseguito l'XOR del testo in chiaro. Il punto di partenza R scelto dall'algoritmo di cifratura è una stringa casuale di n bit. Per aggiungere una stringa di n bit R a un intero i — quando scriviamo $F_k(R+i)$ — convertiamo la stringa di n bit R in un intero nell'intervallo $[0 \ldots 2^n - 1]$ nel solito modo, aggiungiamo questo numero in i, prendi il risultato modulo 2^n e poi riconvertilo in una stringa di n bit. Si noti che il punto di

partenza R è incluso nel testo cifrato, per consentire la decrittazione. In cifratura, Pad è inteso come la stringa vuota quando m = 0.

Modo CTRC (Counter Non Randomizzato)

Sia $F : K \times \{0,1\}^l \to \{0,1\}^L$ una famiglia di funzioni. CTRC produce un cifrario con stati e non randomizzato. Viene mantenuto un contatore globale ctr, inizialmente posto a zero. L'algoritmo di generazione della chiave si limita a restituire una stringa random.

Gli algoritmi di cifratura e decifratura sono i seguenti:

$\mathbf{Enc}_k(M)$
$m \leftarrow \lceil |M|/L \rceil$;
If $ctr+m \geq 2^\ell$ return $\perp$
$Pad \leftarrow F_k(ctr+1)||F_k(ctr+2)||\ldots||F_k(ctr+m)$
$Pad \leftarrow$ Primi $|M|$ bits di Pad
$C' \leftarrow M \oplus Pad$
$ctr \leftarrow ctr+m$
return $\langle ctr-m, C \rangle$

$\mathbf{Dec}_k(\langle i, C \rangle)$
$m \leftarrow \lceil |C|/L \rceil$;
$Pad \leftarrow F_k(i+1)||F_k(i+2)||\ldots||F_k(i+m)$
$Pad \leftarrow$ Primi $|C|$ bits di Pad
$M \leftarrow C \oplus Pad$
return M

L'indice di posizione più la dimensione del messaggio non possono superare la dimensione massima di bit in input che è pari a 2^l, in quel caso l'algoritmo di cifratura ritornerà $\perp$. L'indice di posizione è incluso nel testo cifrato per consentire la decrittazione del messaggio. L'algoritmo di cifratura aggiorna l'indice di posizione ad ogni chiamata e inizia con il valore aggiornato la volta successiva che viene richiamato.

Privacy

Il prossimo obiettivo è quello di arrivare ad una soddisfacente definizione di sicurezza.
L'avversario A vede un certo numero di crittotesti e vuole cercare di "trarre" informazione da essi. È più facile pensare all'insicurezza che alla sicurezza, perché possiamo certamente identificare azioni avversarie che indubbiamente implicano che lo schema è insicuro. Ad esempio, se l'avversario può, da alcuni testi cifrati, derivare la chiave sottostante k, può in seguito decifrare tutto ciò che vede, quindi se lo schema consente un facile recupero della chiave da alcuni testi cifrati è decisamente insicuro.
Ora, l'errore che viene spesso fatto è quello di invertire questo, dicendo che se il ripristino della chiave è difficile, allora lo schema è sicuro. Questo non è certamente vero, perché ci sono altre possibili debolezze. Ad esempio, cosa accadrebbe se, dato il testo cifrato, l'avversario potesse

facilmente recuperare il testo in chiaro M senza trovare la chiave? Certamente lo schema è insicuro anche allora. Quindi dovremmo ora dichiarare sicuro uno schema se è difficile recuperare un testo in chiaro dal testo cifrato? Molte persone direbbero di sì. Eppure anche questo sarebbe sbagliato. Una ragione è che l'avversario potrebbe essere in grado di ricavare informazioni parziali su M . Ad esempio, anche se potesse non essere in grado di recuperare M , l'avversario potrebbe, dato C, essere in grado di recuperare il primo bit di M , o la somma di tutti i bit di M . Questo non va bene, perché questi bit potrebbero contenere informazioni preziose. Per fare un esempio concreto, diciamo che sto comunicando al mio broker un messaggio che è una sequenza di decisioni di "acquisto" o "vendita" per una sequenza prestabilita di azioni. Cioè, abbiamo determinate azioni, numerate da 1 a m, e il bit i del messaggio è 1 se voglio acquistare azioni e 0 altrimenti. Il messaggio viene inviato cifrato. Ma se il primo bit viene catturato, l'avversario sa se voglio comprare o vendere il titolo 1, che potrebbe essere qualcosa che non voglio rivelare. Se la somma dei bit viene catturata, l'avversario sa quante azioni sto acquistando.

Gli utenti non dovrebbero preoccuparsi di come formattano i propri dati: li formattano come preferiscono e la cifratura dovrebbe comunque garantire la privacy. Consideriamo che è il compito del progettista del protocollo garantire che ciò sia vero.

Esiste una lista pressoché infinita di proprietà che rendono insicuro un sistema. Bisogna pensare alla sicurezza in maniera più diretta per poter sperare si riuscire a definirla. Si potrebbe voler dire qualcosa del tipo: lo schema di cifratura è sicuro se dato C, l'avversario non ha idea di cosa sia M . Questo però non può essere vero, a causa di quella che viene chiamata **informazione a priori**. Spesso si sa qualcosa del messaggio. Ad esempio, potrebbe essere un pacchetto con intestazioni note. Oppure, potrebbe essere una parola inglese. Quindi l'avversario, e tutti gli altri, hanno alcune informazioni sul messaggio anche prima che sia cifrato. Vorremo che la cifratura avesse le proprietà **ideali**, ovvero un canale perfettamente sicuro attraverso il quale mittente e destinatario possono comunicare. In particolare nessuna informazione parziale sarebbe trapelata. Però questo non è possibile, perché informazioni come la dimensione della stringa del testo in chiaro sono già note a priori.

Si consideri lo schema di crittografia ECB. Dato il testo cifrato, un avversario che sta ascoltando può capire il messaggio? È difficile capire come, dal momento che non conosce k, e se F è un cifrario a blocchi "buono", allora dovrebbe avere difficoltà a invertire F_k senza conoscere la chiave sottostante. Tuttavia questo non è un buon schema. Si consideri solo il caso $n = 1$ di un singolo blocco di messaggio. Supponiamo che un centro militare missilistico abbia solo due messaggi, 1^n per sparare e 0^n per non sparare. Continua a inviare dati, ma sempre uno di questi due. Che succede? Quando passa il primo testo cifrato C_1, l'avversario potrebbe non sapere qual è il testo in chiaro. Ma poi, diciamo che può facilmente decifrare tutti i messaggi successivi, perché se vede un testo cifrato C, il messaggio è 1^n se $C = C_1$ e 0^n se $C \neq C_1$.

In uno schema di cifratura sicuro, non dovrebbe essere possibile mettere in relazione testi cifrati di messaggi diversi della stessa lunghezza in modo tale da far trapelare informazioni.

Conseguenze

La cifratura deve essere probabilistica o dipendere dalle informazioni sullo stato. In caso contrario, si può sempre sapere se lo stesso messaggio è stato inviato due volte. Ogni cifratura deve utilizzare nuovi lanci di monete o, ad esempio, un contatore e una cifratura di un particolare messaggio può essere diversa ogni volta. In termini della nostra configurazione significa che Enc è un algoritmo **probabilistico** o **stateful**.

Una formalizzazione della privacy è quella che viene chiamata perfetta sicurezza, una nozione della teoria dell'informazione introdotta da Shannon.

> La perfetta sicurezza richiede che, indipendentemente dalla potenza di calcolo disponibile all'avversario, il testo cifrato non gli fornisce alcuna informazione sul testo in chiaro oltre alle informazioni a priori.

La perfetta sicurezza è un attributo molto forte, ma ottenerla richiede una chiave lunga quanto la quantità totale di dati crittografati, e questo di solito non è pratico. Quindi qui esaminiamo una nozione di **sicurezza computazionale**, ovvero l'avversario possiede una potenza di calcolo limitata.

Indistinguibilità

L'idea di base dietro l'indistinguibilità è di considerare un avversario (non in possesso di una chiave segreta) che sceglie due messaggi della stessa lunghezza. Uno dei due messaggi viene cifrato e il crittotesto è dato all'avversario. Lo schema è considerato sicuro se l'avversario ha difficoltà a dire quale dei due messaggi era quello cifrato.

In breve:

- A sceglie q coppie di messaggi M_0^i, M_1^i della stessa lunghezza.
- A riceve q crittotesti C^i.
- I crittotesti cifrano o solo i messaggi di destra o solo quelli di sinistra (secondo un bit segreto b).
- L'obiettivo di A è indovinare b.

L'avversario sceglie la sequenza delle coppie di messaggi tramite un attacco di testo in chiaro (**chosen-plaintext attack**). Ciò significa che l'avversario sceglie la prima coppia, poi riceve C_1, quindi sceglie la seconda coppia, riceve C_2 e così via. (A volte questo è chiamato attacco adattivo di testo in chiaro, perché l'avversario può scegliere in modo adattivo ciascuna query in un modo che risponda alle risposte precedenti).

Formalizziamo il problema. Fissiamo uno schema di cifratura $SE = (KeyGen, Enc, Dec)$. Potrebbe essere senza stato o con stato. Consideriamo un avversario A. È un programma che ha accesso a un oracolo al quale può fornire come input una qualsiasi coppia di messaggi di uguale lunghezza. L'oracolo restituirà un testo cifrato.
Considereremo due possibili modi in cui questo testo cifrato viene calcolato dall'oracolo, corrispondenti a due possibili “mondi” in cui “vive” l'avversario. Per fare ciò, bisogna definire prima l'oracolo sinistro o destro (abbreviato **lr-encryption oracle**) $E_k(LR(\cdot, \cdot, b))$. L'oracolo cifra uno dei messaggi, la cui scelta è fatta secondo il bit b. Ora i due mondi sono i seguenti:

- **Mondo 0**: L'oracolo fornito all'avversario è $Enc_k(LR(\cdot, \cdot, 0))$. Quindi, ogni volta che l'avversario fa una query (M_0, M_1) con $|M_0| = |M_1|$, l'oracolo calcola $C \leftarrow Enc_k(M_0)$ e restituisce C come risposta.
- **Mondo 1**: L'oracolo fornito all'avversario è $Enc_k(LR(\cdot, \cdot, 1))$. Quindi, ogni volta che l'avversario fa una query (M_0, M_1) con $|M_0| = |M_1|$ al suo oracolo, l'oracolo calcola $C \leftarrow Enc_k(M_1)$ e restituisce C come risposta.

$$LR(x_0, x_1, b) = \begin{cases} x_0 & \text{if } b = 0 \\ x_1 & \text{if } b = 1 \end{cases}$$

Il Mondo 0 viene anche chiamato mondo di "sinistra", il Mondo 1 viene anche chiamato mondo di "destra". Il problema per l'avversario è, dopo aver parlato a lungo con il suo oracolo, di dire quale dei due oracoli gli è stato dato.

Si pensi all'oracolo come a una subroutine a cui A ha accesso. L'avversario A può eseguire una query (M_0, M_1) chiamando la subroutine con argomenti (M_0, M_1). In un solo passaggio, viene quindi restituita la risposta. L'avversario A non ha alcun controllo su come viene calcolata la risposta, né A può vedere il funzionamento interno della subroutine, che in genere dipenderà da informazioni segrete che A non è fornita. L'avversario A ha solo un'interfaccia in comune con la subroutine: la capacità di chiamarla come una scatola nera e ottenere una risposta.

In primo luogo, supponiamo che lo schema di cifratura simmetrico SE dato sia senza stato. L'oracolo, in entrambi i mondi, è probabilistico, perché chiama l'algoritmo di cifratura. Ricordiamo che questo algoritmo è probabilistico. Sopra, quando diciamo $C \leftarrow Enc_k(M_b)$, è implicito che l'oracolo scelga le proprie monete casuali e le usi per calcolare il testo cifrato C. Le scelte casuali della funzione di cifratura sono in qualche modo rappresentate nella notazione. Se il dato schema di cifratura simmetrica SE è stateful, anche gli oracoli, in entrambi i mondi, diventano stateful.

Definizione 2

Sia $SE = \{KeyGen, Enc, Dec\}$ uno schema di cifratura simmetrica e sia A un algoritmo che ha accesso ad un oracolo. Si consideri il seguente esperimento:

$$\begin{array}{l|l} \text{Experiment } \mathbf{Exp}^{\text{ind-cpa-1}}_{\mathcal{SE}}(A) & \text{Experiment } \mathbf{Exp}^{\text{ind-cpa-0}}_{\mathcal{SE}}(A) \\ \quad K \xleftarrow{\$} \mathcal{K} & \quad K \xleftarrow{\$} \mathcal{K} \\ \quad d \xleftarrow{\$} A^{\mathcal{E}_K(\text{LR}(\cdot,\cdot,1))} & \quad d \xleftarrow{\$} A^{\mathcal{E}_K(\text{LR}(\cdot,\cdot,0))} \\ \quad \text{Return } d & \quad \text{Return } d \end{array}$$

Il vantaggio dell'avversario è:

$$Adv^{ind-cpa}_{SE}(A) = |Pr[Exp^{ind-cpa-1}_{SE}(A) = 1] - Pr[Exp^{ind-cpa-0}_{SE}(A) = 1]|$$

Un cifrario simmetrico fornisce **indistinguibilità** se il vantaggio è prossimo a zero, questo significa che A sta emettendo 1 all'incirca tanto spesso nel mondo 0 quanto nel mondo 1, il che significa che sta facendo un buon lavoro.

Informalmente, affinché lo schema di cifratura simmetrica SE sia sicuro contro l'attacco di testo in chiaro, il vantaggio **IND-CPA** di un avversario deve essere piccolo, indipendentemente dalla strategia tentata dall'avversario. Tuttavia, dobbiamo essere realistici nelle nostre aspettative, comprendendo che il vantaggio può crescere man mano che l'avversario investe più impegno nel suo attacco. La sicurezza è una misura di quanto grande potrebbe essere il vantaggio dell'avversario rispetto alle risorse dell'avversario. Ad esempio l'avversario può vincere con probabilità $1/2$, ma un tale avversario non può essere considerato una minaccia. Il vantaggio, misura quindi la capacità dell'avversario di far meglio che sparare a caso.

Consideriamo uno schema di cifratura "sicuro contro l'attacco di testo in chiaro" se un avversario è limitato all'utilizzo di una quantità "pratica" di risorse (tempo di calcolo, numero di query) non può ottenere un vantaggio "significativo". La nozione tecnica è chiamata indistinguibilità sinistra o destra sotto l'attacco di testo in chiaro, indicato con IND-CPA.

Interpretazione alternativa

Passiamo a descrivere un'interpretazione un po' diversa dell'indistinguibilità sinistra o destra. Perché $Adv^{ind-cpa}_{SE}(A)$ è chiamato il “vantaggio” dell'avversario? Possiamo vedere il compito dell'avversario come cercare di indovinare in quale mondo si trova. Un'ipotesi banale è che l'avversario restituisca un bit casuale. In tal caso, ha probabilità $1/2$ di avere ragione.

Chiaramente, non ha fatto nulla di dannoso in questo caso. Il vantaggio dell'avversario misura quanto meglio di questo riesce a indovinare in quale mondo si trova, vale a dire l'eccesso di $1/2$ della probabilità dell'avversario di indovinare correttamente.

Preposizione 1

Sia $SE = (KeyGen, Enc, Dec)$ uno schema di cifratura simmetrica e sia A un algoritmo che ha accesso a un oracolo che accetta in input una coppia di stringhe e restituisce un bit. Consideriamo il seguente esperimento:

$$\begin{array}{l}\text{Experiment } \mathbf{Exp}^{\text{ind-cpa-cg}}_{\mathcal{SE}}(A)\\ \quad b \xleftarrow{\$} \{0,1\}\ ;\ K \xleftarrow{\$} \mathcal{K}\\ \quad b' \xleftarrow{\$} A^{\mathcal{E}_K(\mathrm{LR}(\cdot,\cdot,b))}\\ \quad \textbf{if } \ b = b' \textbf{ then } \textbf{ return } 1 \textbf{ else } \textbf{ return } 0\end{array}$$

Allora:

$$Adv^{ind-cpa}_{SE}(A) = 2 \cdot Pr[Exp^{ind-cpa-cg}_{SE}(A) = 1] - 1$$

Nell'esperimento analizzato, l'avversario A viene eseguito con un oracolo per il mondo b, dove il bit b viene scelto a caso. A alla fine emette un bit b', la sua ipotesi sul valore di b. L'esperimento restituisce 1 se l'ipotesi di A è corretta. Così,

$$Pr[Exp^{ind-cpa-cg}_{SE}(A) = 1]$$

è la probabilità che A indovini correttamente in quale mondo si trova ("cg" nell'apice sta per "correct guess" (ipotesi corretta)). Questo valore è $1/2$ quando l'avversario non merita alcun vantaggio, poiché si può indovinare correttamente b con una strategia semplice come "rispondi sempre zero" o "rispondi con un bit casuale". Il “vantaggio” di A può quindi essere visto come l'eccedenza di questa probabilità su $1/2$, che, ridimensionata, è:

$$2 \cdot Pr[Exp^{ind-cpa-cg}_{SE}(A) = 1] - 1$$

La Proposizione dice che questo vantaggio ridimensionato è esattamente la stessa misura di prima.

Dimostrazione

$$
\begin{aligned}
&Pr[Exp_{SE}^{ind-cpa-cg}(A) = 1] = \\
&= Pr[b = b'] = \\
&= Pr[b = b' \wedge b = 1] + Pr[b = b' \wedge b = 0] = \\
&= Pr[b = b' | b = 1] \cdot Pr[b = 1] + Pr[b = b' | b = 0] \cdot Pr[b = 0] = \\
&= Pr[b = b' | b = 1] \cdot \frac{1}{2} + Pr[b = b' | b = 0] \cdot \frac{1}{2} = \\
&= \frac{1}{2}(Pr[b = b' | b = 1] + Pr[b = b' | b = 0]) = \\
&= \frac{1}{2}(Pr[b' = 1 | b = 1] + Pr[b' = 0 | b = 0]) = \\
&= \frac{1}{2}(Pr[b' = 1 | b = 1] + 1 - Pr[b' = 1 | b = 0]) = \\
&= \frac{1}{2}Pr[b' = 1 | b = 1] + \frac{1}{2} - \frac{1}{2}Pr[b' = 1 | b = 0] = \\
&= \frac{1}{2} + \frac{1}{2}(Pr[b' = 1 | b = 1] - Pr[b' = 1 | b = 0]) = \\
&= \frac{1}{2} + \frac{1}{2}(Pr[Exp_{SE}^{ind-cpa-1}(A) = 1] - Pr[Exp_{SE}^{ind-cpa-0}(A) = 1]) = \\
&= \frac{1}{2} + \frac{1}{2} \cdot Adv_{SE}^{ind-cpa}(A)
\end{aligned}
$$

Da questa segue che:

$$
\begin{aligned}
Pr[Exp_{SE}^{ind-cpa-cg}(A) = 1] &= \frac{1}{2} + \frac{1}{2} \cdot Adv_{SE}^{ind-cpa}(A) \\
Adv_{SE}^{ind-cpa}(A) &= 2 \cdot Pr[Exp_{SE}^{ind-cpa-cg}(A) = 1] - 1
\end{aligned}
$$

Perché questa è una buona definizione?

La nostra tesi è che dovremmo considerare uno schema di cifratura come "sicuro" se e solo se è sicuro IND-CPA, il che significa che la formalizzazione sopra cattura il nostro senso intuitivo di privacy e i requisiti di sicurezza che si potrebbero applicare a uno schema di cifratura può essere ridotto a questo.
Ma perché? Perché IND-CPA cattura la "privacy"? Questa è una domanda importante a cui rispondere. Sono state notate una serie di proprietà di sicurezza necessarie ma non sufficienti per la sicurezza. Ad esempio, dovrebbe essere computazionalmente impossibile per un avversario recuperare la chiave da alcune coppie testo in chiaro-testo cifrato, o recuperare un testo in chiaro da un testo cifrato.
Un test della nostra definizione è che implica le proprietà necessarie che abbiamo discusso e altre. Ad esempio, uno schema sicuro nel senso IND-CPA della nostra definizione dovrebbe essere, automaticamente, protetto anche contro key recovery e plaintext recovery.

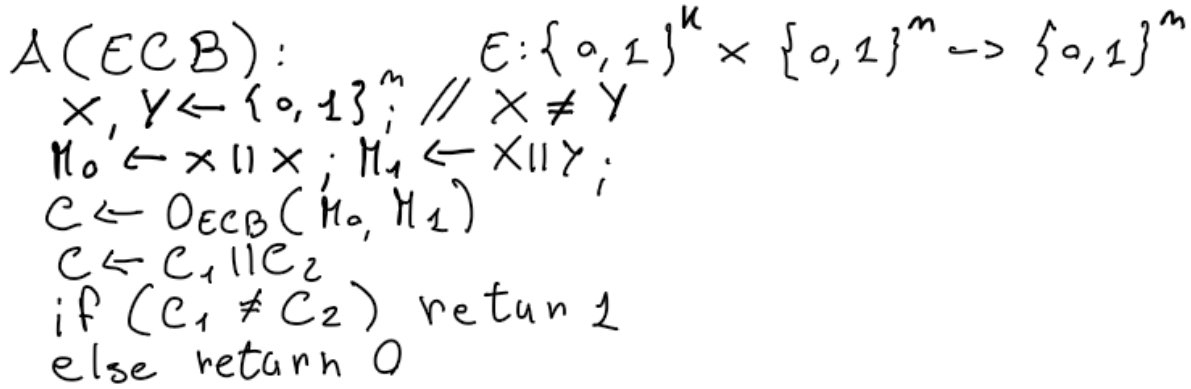
Esempi di CPA

Attacco su ECB

Illustriamo l'uso della definizione IND-CPA nella ricerca di attacchi fornendo un attacco alla modalità ECB.

Fissiamo un cifrario a blocchi $E : K \times \{0,1\}^n \to \{0,1\}^n$. Supponiamo che un avversario veda un testo cifrato $C = Enc_k(M)$ corrispondente a un testo in chiaro casuale M, cifrato con la chiave k anch'essa sconosciuta all'avversario. L'avversario può recuperare M? Non facilmente, se E è un cifrario a blocchi "buono". Ad esempio, se E è AES, sembra abbastanza irrealizzabile. Tuttavia, abbiamo già discusso come l'impossibilità di recuperare il testo in chiaro dal testo cifrato non sia un'indicazione di sicurezza. La ECB ha altri punti deboli. Nota che se due testi in chiaro M e M' concordano nel primo blocco, allora lo sono anche i corrispondenti testi cifrati. Quindi un avversario, dati i testi cifrati, può dire se i primi blocchi dei corrispondenti testi in chiaro sono o meno gli stessi. Si tratta di una perdita di informazioni parziali sui testi in chiaro e non è consentita in uno schema di cifratura sicuro.

Vogliamo mostrare che esiste un avversario che ha un alto vantaggio IND-CPA mentre usa una piccola quantità di risorse. Costruiamo ora un tale avversario A. Ricorda che ad A viene assegnato un oracolo di cifratura-lr $Enc_k(LR(\cdot,\cdot,b))$ che accetta come input una coppia di messaggi e che restituisce una cifratura del messaggio sinistro o destro della coppia, a seconda del valore del bit b. L'obiettivo di A è determinare il valore di b. Il nostro avversario funziona così:



La singola query dell'oracolo dell'avversario è la coppia di messaggi M_0, M_1. Poiché ciascuno di essi è lungo due blocchi, lo è anche il testo cifrato calcolato secondo lo schema ECB. Ora, noi affermiamo che:

$$Pr[Exp_{SE}^{ind-cpa-1}(A) = 1] = 1$$
$$Pr[Exp_{SE}^{ind-cpa-0}(A) = 1] = 0$$

Sapendo che M_0 è un messaggio costituito da due blocchi uguali, se il messaggio cifrato contiene due blocchi identici l'avversario è sicuro di trovarsi nel Mondo 1, quindi ritorna 1 e mai 0. Proprio per questo motivo il vantaggio di A è:

$$Adv^{ind-cpa}(A) = 1$$

ECB è uno schema di cifratura non sicuro anche se il cifrario a blocchi sottostante E è altamente sicuro.

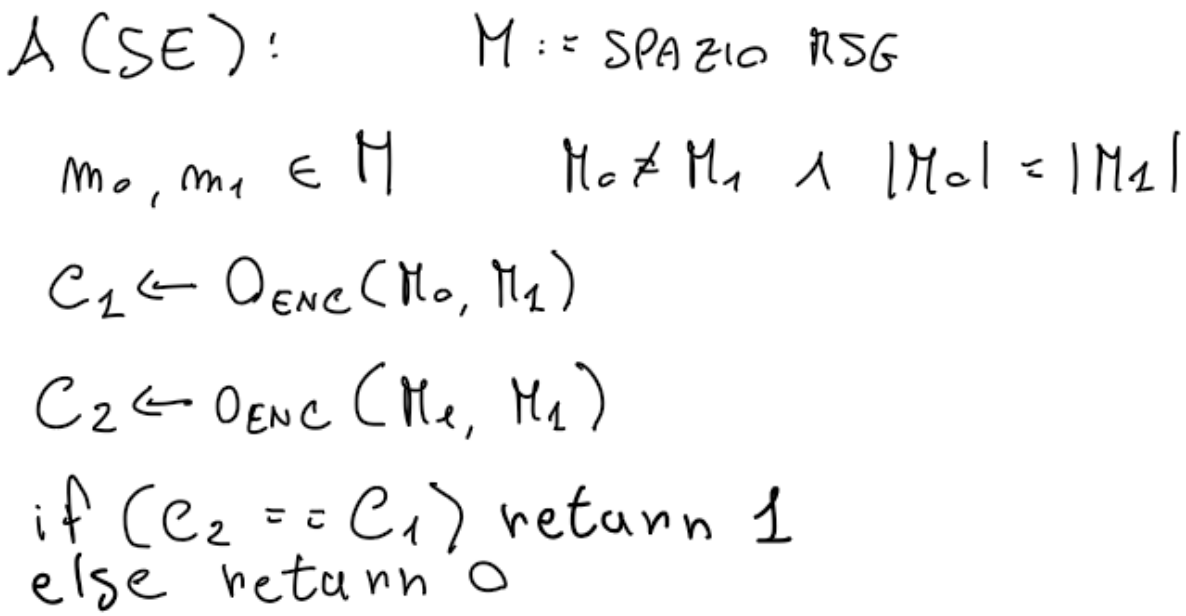
Ogni schema deterministico e senza stati è insicuro

Teorema

Sia $SE = (KeyGen, Enc, Dec)$ un cifrario simmetrico (deterministico e senza stati). Sia M lo spazio dei messaggi (contenente almeno due messaggi distinti della stessa lunghezza). Allora, esiste un avversario A, contro SE, tale che:

$$Adv^{ind-cpa}(A) = 1$$

La dimostrazione è una riscrittura dell'algoritmo precedente:



Quindi qualunque sistema deterministico non è sicuro.

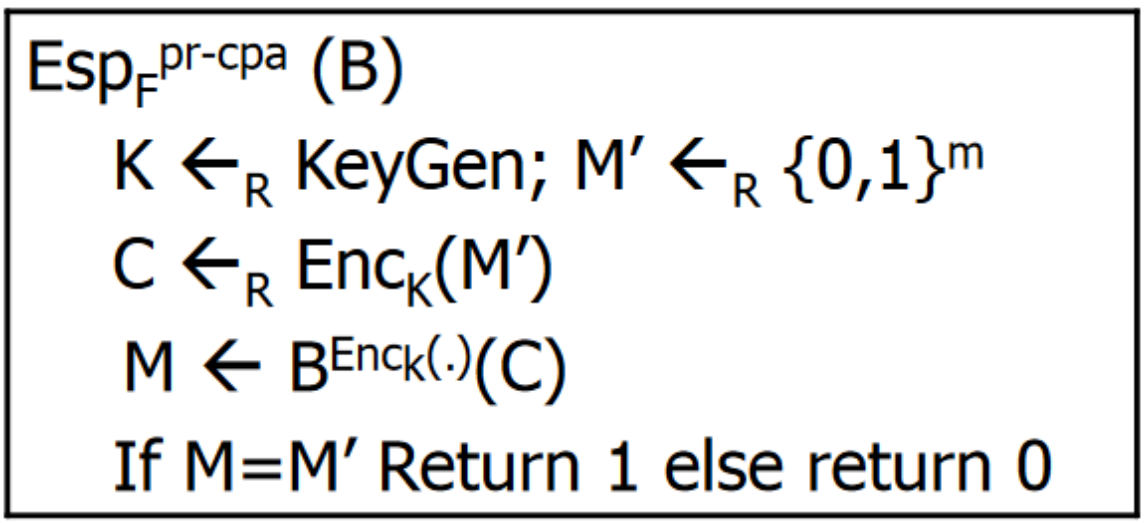
Testiamo la definizione

Esistono un certo numero di proprietà che sono necessarie per la privacy, ad esempio la resistenza ad attacchi di tipo KR o PR. Un buon test per la definizione analizzata è verificare se essa implica queste proprietà. Il ragionamento è analogo a quello fatto per le **PRF**. Uno schema sicuro in senso IND-CPA dovrebbe essere anche sicuro in senso key recovery o plaintext recovery. Mostriamo che IND-CPA implica PR-CPA.

IND-CPA => PR-CPA

Dimostriamo che dato uno schema $SE = (KeyGen, Enc, Dec)$ se esiste un avversario PR-CPA questo può essere "trasformato" in un avversario IND-CPA che utilizza più o meno le stesse risorse. Per semplicità supponiamo che SE sia senza stati. Per far questo dobbiamo prima definire formalmente un avversario PR-CPA.

Sia SE un cifrario simmetrico senza stati, definiamo un avversario B come segue:



L'avversario cerca di indovinare il messaggio di un testo cifrato casuale. In questo caso il vantaggio di B è pari a:

$$Adv(B) = Pr[Esp_{SE}^{pr-cpa}(B) = 1]$$

ovvero la probabilità che l'avversario sia in grado di ricavare il testo del messaggio partendo da un testo cifrato.

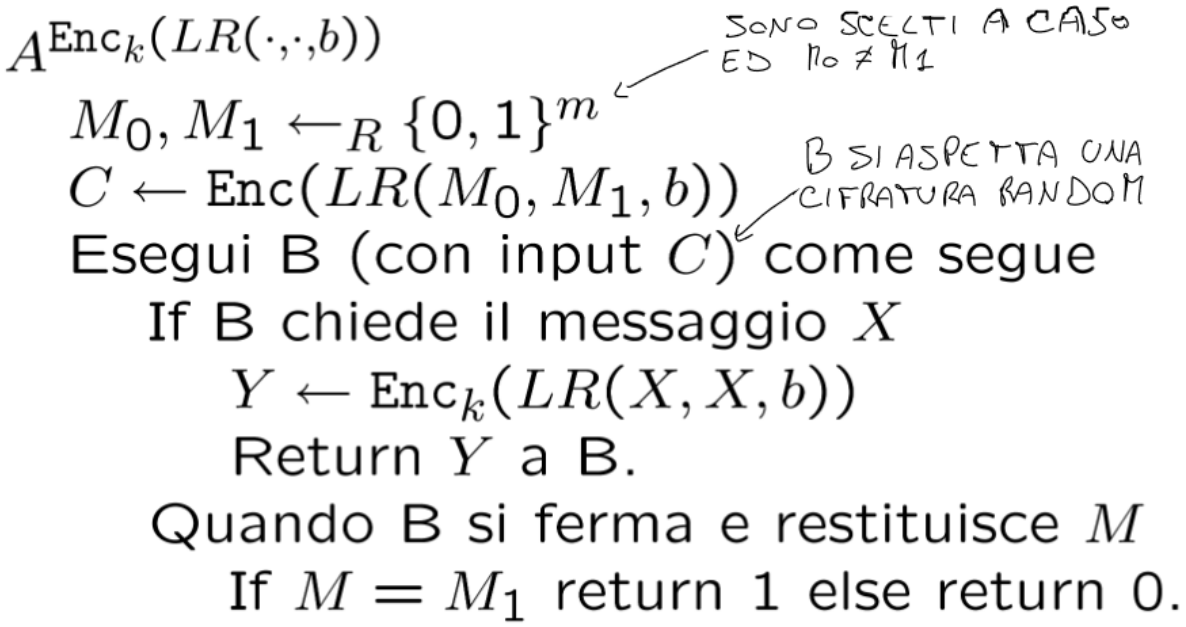
Teorema

Sia $SE = (KeyGen, Enc, Dec)$ un cifrario simmetrico senza stati. Sia $\{0,1\}^m$ lo spazio dei messaggi e B l'avversario pr-cpa che fa q domande. Allora esiste un avversario ind-cpa A, che fa $q+1$ domande tale che:

$$Adv(B) \leq Adv(A) + \frac{1}{2^m}$$

A ha a disposizione un oracolo $LR(\cdot,\cdot,b)$ e deve stabilire il valore di b. Per far questo A può sfruttare B. A dunque deve essere in grado di:

1. Rispondere alle domande di B;
2. Utilizzare il messaggio restituito da B per determinare b.



Quindi:

$$Pr[Esp^{ind-cpa-1}(A) = 1] = Adv(B)$$

$$Pr[Esp^{ind-cpa-0}(A) = 1] = \frac{1}{2^m}$$

Allora:

$$Adv(A) = Adv(B) - \frac{1}{2^m}$$

Abbiamo dimostrato che una proprietà necessaria è implicata dall'indistinguibilità.

Un approccio più generale

Quanto detto suggerisce un metodo ancora più generale per provare che la definizione è quella giusta.
Consideriamo una qualunque proprietà di sicurezza e mostriamo che uno schema sicuro in senso IND-CPA la possiede.

Si consideri un cifrario SE e scegliamo un predicato:

$$P : M \to \{0,1\}$$

Sia $\{0,1\}^m$ lo spazio dei messaggi e sia B un avversario che fa q domande e dato un critto-testo $C = Enc_k(M)$ determina $P(M)$. Allora esiste un avversario IND-CPA A, che fa $q+1$ domande tale che:

$$Adv(B) = Adv(A)$$

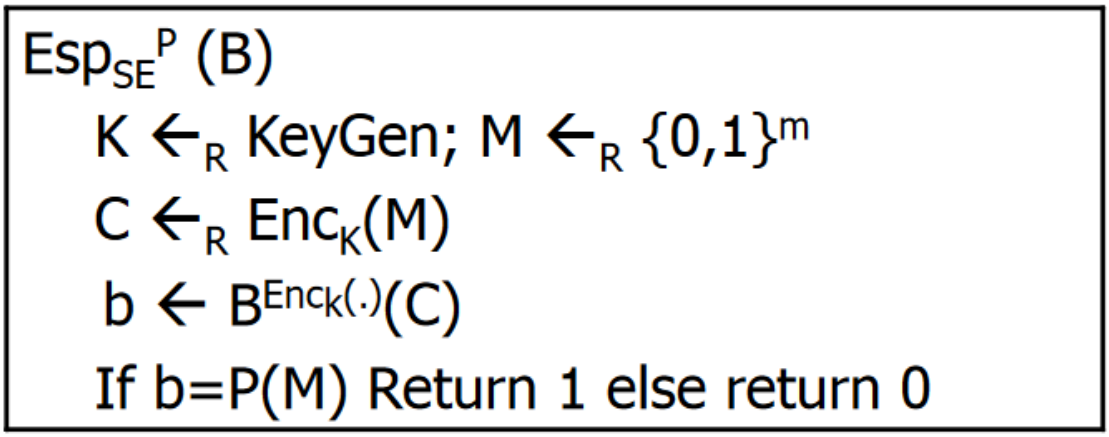
Se esistesse questo avversario allora non è possibile che il cifrario garantisca affidabilità.

L'avversario A ha a disposizione un oracolo $LR(\cdot,\cdot,b)$ e deve stabilire il valore di b. Per far questo A può sfruttare B. A dunque deve essere in grado di:

- Rispondere alle domande di B;
- Utilizzare il bit restituito da B per determinare b.

Definiamo l'esperimento di B come segue:

- Sia SE un cifrario simmetrico senza stati, allora:



Il vantaggio di B è definito come segue:

$$Adv(B) = 2 \cdot Pr[Esp_{SE}^{P}(B) = 1] - 1$$

Supponiamo che il predicato sia il **least significant bit** (lsb), ovvero l'avversario B è sempre in grado di estrarre il bit meno significativo e quello che vogliamo dimostrare è che se l'avversario B conosce tale informazione, allora esiste un avversario A per cui il cifrario non è sicuro in senso IND-CPA.

Possiamo osservare che l'avversario B prende in input un critto-testo casuale, utilizzando una distribuzione di probabilità R, quindi se si esegue l'algoritmo più volte, B avrà sempre un critto-testo diverso. Successivamente B usa un oracolo al quale farà delle domande e darà in output un valore b attraverso il quale proverà ad indovinare il predicato.

Bisogna costruire un avversario che sfruttando B sia in grado di rompere il cifrario. Quindi costruiamo l'avversario A come segue:

$A^{\mathrm{Enc}_k(LR(\cdot,\cdot,b))}$

$M_0, M_1 \leftarrow_R \{0,1\}^m \ (lsb(M_d = d))$
$C \leftarrow \mathrm{Enc}(LR(M_0, M_1, b))$
Esegui B (con input C) come segue
If B chiede il messaggio X
$Y \leftarrow \mathrm{Enc}_k(LR(X, X, b))$
Return Y a B.
Quando B si ferma e restituisce b'
If $b' = 1$ return 1 else return 0.

L'avversario A sceglierà due messaggi M_0, M_1 tali che $lsb(M_0) = 0$ e $lsb(M_1) = 1$. L'avversario A userà il proprio oracolo per cifrare uno dei due messaggi e il critto-testo sarà passato in input a B il quale userà il proprio oracolo per fare delle domande, alle quali A è in grado di rispondere. Alla fine B restituirà un bit che è il risultato del predicato, che viene usato da A per determinare se si trova nel Mondo 0 o nel Mondo 1.

Dimostrazione

$$Pr[Esp^{ind-cpa-1}(A) = 1] = Pr[B \to 1] = Pr[Esp^P(B) = 1] \ \ (1)$$

$$Pr[Esp^{ind-cap-0}(A) = 1] = 1 - Pr[Esp^P(B) = 1] \ \ (2)$$

Sottraendo membro a membro (1) e (2) si ottiene:

$$Pr[Esp^{ind-cpa-1}(A) = 1] - Pr[Esp^{ind-cap-0}(A) = 1] = Pr[Esp^P(B) = 1] - 1 + Pr[Esp^P(B) = 1]$$
$$Adv^{ind-cpa}(A) = 2 \cdot Pr[Esp^P(B) = 1] - 1$$

Quello che abbiamo dimostrato ci permette di affermare che se un cifrario garantisce indistinguibilità non può rilevare nemmeno un bit di informazione. Possiamo, inoltre, affermare che questo che è stato appena dimostrato è la definizione equivalente del teorema di Shannon per le PRF.

Sicurezza dei modi CTR

Ricordiamo che la modalità di funzionamento CTR (contatore) di una famiglia di funzioni è disponibile in due varianti: la versione randomizzata (senza stato) CTR$ e il meccanismo basato su contatore (stateful) CTRC. Entrambe le modalità raggiungono l'indistinguibilità sotto CPA, ma, curiosamente, la sicurezza quantitativa è leggermente diversa. La differenza nasce dal fatto che CTRC raggiunge la perfetta indistinguibilità se si usa una famiglia di funzioni casuali $Func(n)$, ma CTR$ non otterrebbe nemmeno indistinguibilità perfetta, a causa della possibilità di collisioni prodotte dalle "sovrapposizioni" nello pseudo one-time pad.

Si seguito sono riportati i due schemi:

$\mathbf{Enc}_k(M)$ Modo CTR$

$m \leftarrow \lceil |M|/L \rceil$; $R \leftarrow_R \{0,1\}^\ell$
$Pad \leftarrow F_k(R+1)||F_k(R+2)||\ldots||F_k(R+m)$
$Pad \leftarrow$ Primi $|M|$ bits di Pad
$C' \leftarrow M \oplus Pad$
$C \leftarrow R||C'$
return C

$\mathbf{Enc}_k(M)$ Modo CTRC

$m \leftarrow \lceil |M|/L \rceil$;
If $ctr+m \geq 2^\ell$ return $\perp$
$Pad \leftarrow F_k(ctr+1)||F_k(ctr+2)||\ldots||F_k(ctr+m)$
$Pad \leftarrow$ Primi $|M|$ bits di Pad
$C' \leftarrow M \oplus Pad$
$ctr \leftarrow ctr+m$
return $\langle ctr-m, C \rangle$

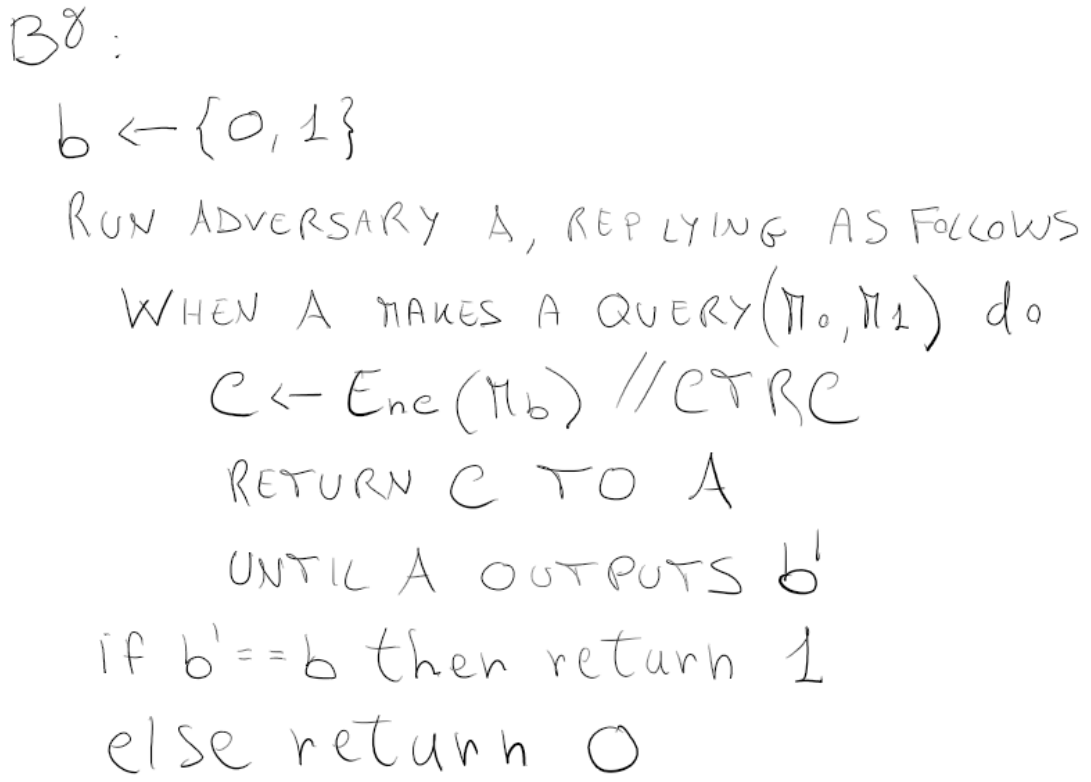
Teorema - sicurezza del modo CTRC

Sia $F : K \times \{0,1\}^l \rightarrow \{0,1\}^L$ una famiglia di funzioni e sia $SE = (KeyGen, Enc, Dec)$ il corrispondente schema di cifratura simmetrica CTRC. Sia A un avversario (per attaccare la sicurezza IND-CPA di SE) che viene eseguito in tempo al massimo t e richiede al massimo q query, che totalizzano al massimo σ blocchi di L bit. Allora esiste un avversario B (che attacca la sicurezza PRF di F) tale che:

$$Adv^{ind-cpa}_{SE}(A) \leq 2 \cdot Adv^{prf}_{F}(B)$$

Inoltre B richiede al massimo un tempo $t' = t + O(q + L\sigma)$ e richiede al massimo $q' = \sigma$ query al proprio oracolo.

Ricorda che B prende un oracolo $g : \{0,1\}^l \rightarrow \{0,1\}^L$. Questo oracolo viene estratto a caso da F o da $Func(l, L)$ e B non sa quale. Per scoprirlo, B userà A, eseguendolo come una subroutine. Ma ricorda che anche A ottiene un oracolo, vale a dire un oracolo di cifratura lr. Dal punto di vista di A, questo oracolo è semplicemente una subroutine: A può scrivere, in qualche posizione, una coppia di messaggi, e riceve una risposta da qualche entità che chiama il suo oracolo. Quando B esegue A come subroutine, è B che "simulerà" l'oracolo di cifratura lr per A, il che significa che B fornirà le risposte a qualsiasi query che A effettua. Ecco la descrizione di B:



L'avversario B stesso sceglie il bit di sfida b che rappresenta la scelta dei mondi per A, e quindi vede se A riesce o meno a indovinare il valore di questo bit. In caso affermativo, scommette che g sia un'istanza di F , altrimenti scommette che g sia un'istanza di $Func(l, L)$. Per l'analisi, affermiamo che:

$$Pr[Esp_F^{prf-1}(B) = 1] = Pr[Esp^{ind-cpa-cg}(A) = 1] = Pr[Esp_F^{prf-0}(B) = 1]$$

Quindi si ha che:

$$Pr[Esp_F^{prf-1}(B) = 1] = \frac{1}{2} + \frac{1}{2} \cdot Adv^{ind-cpa}(A)$$
$$Pr[Esp_F^{prf-0}(B) = 1] = \frac{1}{2} + \frac{1}{2} \cdot Adv^{ind-cpa}(A)$$

Allora:

$$Adv_F^{prf}(B) = Pr[Esp_F^{prf-1}(B) = 1] - Pr[Esp_F^{prf-0}(B) = 1] = \frac{1}{2} + \frac{1}{2} \cdot Adv^{ind-cpa}(A) - \frac{1}{2} - \frac{1}{2} \cdot Adv^{ind-cpa}(A)$$

Prima di concludere con il risultato introduciamo il seguente lemma:

Sia A un qualsiasi avversario IND-CPA che attacca $SE[Func(l, L)]$, allora:

$$Adv^{ind-cpa}(A) = 0$$

Questo lemma è dimostrato dal fatto che l'avversario utilizza delle funzioni casuali ed è come se usasse il one time-pad, quindi il suo vantaggio è pari a zero.

Sfruttando il lemma possiamo quindi affermare che:

$$Pr[Esp_F^{prf-0}(B) = 1] = \frac{1}{2} + \frac{1}{2} \cdot Adv^{ind-cpa}(A) = \frac{1}{2}$$

Quindi:

$$Adv_F^{prf}(B) = Pr[Esp_F^{prf-1}(B) = 1] - Pr[Esp_F^{prf-0}(B) = 1] =$$
$$\frac{1}{2} + \frac{1}{2} \cdot Adv^{ind-cpa}(A) - \frac{1}{2} =$$
$$= \frac{1}{2} \cdot Adv^{ind-cpa}(A)$$

Dalla quale segue la tesi:

$$Adv^{ind-cpa}(A) = 2 \cdot Adv_F^{prf}(B)$$

Teorema - sicurezza del modo CTR$

Sia $F : K \times \{0,1\}^l \to \{0,1\}^L$ una famiglia di funzioni e sia $SE = (KeyGen, Enc, Dec)$ un cifrario $CTR\$$. Sia A un avversario IND-CPA contro SE, che fa al più q domande.
Allora esiste B, avversario contro la sicurezza PRF di F, tale che:

$$Adv_{SE}^{ind-cpa}(A) \leq Adv_F^{prf}(B) + \frac{0.5\sigma^2}{2^l}$$

Supponiamo di prendere due messaggi M_1 e M_2 con la stessa distribuzione di probabilità R. Allora $C_1 = M_1 \oplus PAD$ e $C_2 = M_2 \oplus PAD$. Se conosco uno dei due messaggi in chiaro, riesco a trovare PAD e le probabilità che capiti lo stesso R ce le dà il paradosso del compleanno. Quindi conoscendo PAD potrei essere in grado di decifrare messaggi futuri.

Conclusione

I difetti non sono evidenti nel CTR a prima vista. Ma forse esistono. È molto difficile vedere come ci si può convincere che non esistono, quando non si può esaurire lo spazio di tutti i possibili attacchi che si potrebbero tentare. Eppure è proprio questa la difficoltà che i teoremi di cui sopra aggirano.
Dicono che la modalità CTR non ha difetti di progettazione. Dicono che finché si usi un buon cifrario a blocchi, si ha la certezza che nessuno infrangerà lo schema di cifratura. Non si può chiedere di più, poiché se non si utilizza un buon cifrario a blocchi, non c'è motivo di aspettarsi comunque la sicurezza del proprio schema di cifratura. Ci stiamo quindi convincendo che tutti gli attacchi falliscono anche se non sappiamo nemmeno esattamente come potrebbero funzionare. Questo è il potere dell'approccio. Può essere utile fare un esempio concreto.

Esempio

Supponiamo che F sia il cifrario a blocchi AES, in modo che $n = 128$. Supponiamo di voler cifrare $q = 2^{30}$ messaggi, ciascuno lungo un kilobyte (2^{13} bit). Si sta quindi cifrando un totale di 2^{43} bit, ovvero $\sigma = 2^{36}$ blocchi. (Questo è circa un terabyte). Posso farlo in modo sicuro utilizzando CTR$? Lasci che A sia un avversario che attacca la privacy della mia cifratura. Allora esiste un B che sfrutta A. Quanto può essere grande $Adv_{AES}^{prf}(B)$? Egli fa $\sigma = 2^{36}$ query, ed è coerente con il nostro stato di conoscenza della sicurezza di AES, supponi che un tale avversario non possa fare di meglio che montare un attacco del compleanno, il che significa che il suo vantaggio è non più di $\frac{\sigma^2}{2^{128}}$. Sotto tale assunzione, il teorema ci dice che $Adv_{SE}^{ind-cpa}(A)$ è al più:

$$\frac{\sigma^2}{2^{128}} + \frac{0.5\sigma^2}{2^{128}} = \frac{1.5 \cdot 2^{72}}{2^{128}} \leq \frac{1}{2^{55}}$$

Questo è davvero un numero molto piccolo, che dice che la nostra cifratura è sicura, almeno partendo dal presupposto che il miglior attacco alla sicurezza PRF di AES sia un attacco di compleanno.

Cosa ci dice questo esempio?

Se cifriamo più di $\sigma = 2^{l/2}$ blocchi con CTR$ non possiamo più dimostrare alcuna sicurezza, indipendentemente da quanto è buono il cifrario a blocchi che utilizziamo. D'altro canto CTRC rimane sicuro fino a σ blocchi. Generalmente tale distinzione nella pratica non ha grande importanza.

Sicurezza del modo CBC$

Sia $E : K \times \{0,1\}^n \to \{0,1\}^n$ un cifrario a blocchi e $SE = (KeyGen, Enc, Dec)$ il cifrario $CBC\$$. Sia A un avversario IND-CPA contro SE, che fa al più q domande, per un totale di σ blocchi di n-bit.
Allora esiste B, un avversario contro la sicurezza PRF di E, tale che:

$$Adv_{SE}^{ind-cpa}(A) \leq Adv_{E}^{prf}(B) + \frac{\sigma^2}{2^{n+1}}$$

Se in due cifrature si utilizza la stessa randomness che dà luogo allo stesso IV, allora si hanno problemi precedenti a quelli visti sopra.

Attacchi a crittotesto scelto (CCA)

Finora abbiamo considerato la privacy sotto attacco a testo in chiaro scelto. A volte vogliamo considerare la privacy quando l'avversario è in grado di sferrare un tipo di attacco più forte, vale a dire un attacco con testo cifrato scelto. In questo tipo di attacco, un avversario ha accesso a un **oracolo di decrittazione**. Può alimentare questo oracolo con un testo cifrato e recuperare il corrispondente testo in chiaro.
Come potrebbe verificarsi una situazione del genere? Una situazione che si potrebbe immaginare è che un avversario a un certo punto ottenga l'accesso temporaneo all'apparecchiatura che esegue la decrittazione. Può alimentare i testi cifrati dell'apparecchiatura e vedere quali testi in chiaro emergono. (Supponiamo che non possa estrarre direttamente la chiave dall'apparecchiatura, tuttavia.)

Se un avversario ha accesso a un oracolo di decrittazione, la sicurezza all'inizio sembra discutibile, poiché dopotutto può decifrare tutto ciò che vuole. Per creare una nozione significativa di sicurezza, abbiamo posto una restrizione all'uso dell'oracolo di decrittazione. Per vedere di cosa si tratta, esaminiamo più da vicino la formalizzazione. Come nel caso degli attacchi con testo in chiaro scelto, consideriamo due mondi:

- **Mondo 0**: all'avversario viene fornito $Enc_k(LR(\cdot,\cdot,0))$ e l'oracolo $Dec_k(\cdot)$.
- **Mondo 1**: all'avversario viene fornito $Enc_k(LR(\cdot,\cdot,1))$ e l'oracolo $Dec_k(\cdot)$.

L'obiettivo dell'avversario è lo stesso degli attacchi con testo in chiaro scelto: vuole capire in quale mondo si trova. C'è un modo semplice per farlo. Vale a dire, interrogare l'oracolo di cifratura lr su due distinti messaggi di uguale lunghezza M_0, M_1 per ottenere un testo cifrato C, e ora chiamare l'oracolo di decrittazione su C. Se il messaggio restituito dall'oracolo di decrittazione è

M_0, allora l'avversario è nel mondo 0, e se il messaggio restituito dall'oracolo di decrittazione è M_1, allora l'avversario è nel mondo 1. La restrizione che imponiamo è semplicemente che questa chiamata, appena descritta sopra, all'oracolo di decrittazione non è consentita. Più in generale, chiamare una query C all'oracolo di decrittazione è illegittima se C è stato precedentemente restituito dall'oracolo di cifratura lr; altrimenti una query è legittima. Sono consentite solo query legittime. Nella formalizzazione di seguito, l'esperimento restituisce semplicemente 0 se l'avversario effettua una query illegittima.

Questa restrizione lascia ancora all'avversario molto potere. In genere, un attacco con testo cifrato scelto con successo procede prendendo un testo cifrato C restituito dall'oracolo di cifratura lr, modificandolo in un testo cifrato correlato C' e interrogando l'oracolo di decrittazione con C'. L'attaccante cerca di creare C' in modo tale che la sua decrittazione dica all'attaccante quale fosse il messaggio sottostante M.

Definizione

Sia $SE = (KeyGen, Enc, Dec)$ uno schema di cifratura simmetrica, sia A un algoritmo che ha accesso a due oracoli, e sia b un bit. Consideriamo il seguente esperimento:

$Esp_{SE}^{ind\text{-}cca\text{-}1}(A)$	$Esp_{SE}^{ind\text{-}cca\text{-}0}(A)$
$K \leftarrow_R KeyGen$	$K \leftarrow_R KeyGen$
$b \leftarrow A^{Enc_k(LR(.,.,1)),Dec_k(.)}$	$b \leftarrow A^{Enc_k(LR(.,.,0)),Dec_k(.)}$
If A imbroglia Return 0	If A imbroglia Return 0
else return b	else return b

Il vantaggio IND-CCA di A è definito come segue:

$$Adv^{ind-cca}(A) = |Pr[Esp_{SE}^{ind-cca-1}(A) = 1] - Pr[Esp_{SE}^{ind-cca-0}(A) = 1]|$$

Le convenzioni relative alle misure delle risorse sono le stesse utilizzate nel caso degli attacchi con testo in chiaro scelto. In particolare, la lunghezza di una query M_0, M_1 all'oracolo di cifratura lr è definita come la lunghezza di M_0.
Consideriamo uno schema di cifratura "sicuro contro l'attacco del testo cifrato scelto" se un avversario polinomialmente limitato non può ottenere un vantaggio "significativo" nel distinguere i casi $b = 0$ e $b = 1$ dato l'accesso agli oracoli. La nozione tecnica è chiamata indistinguibilità sotto l'attacco del testo cifrato scelto, indicato con IND-CCA.

Diciamo che A imbroglia se interroga $Dec_k(\cdot)$ su un crittotesto già restituito da $Enc_k(LR(\cdot, \cdot, 1))$.

Attacco CCA ai modi CTR

Gli attacchi con testo cifrato scelto sono abbastanza potenti da violare tutte le modalità operative standard, anche quelle come CTR e CBC che sono sicure contro attacchi con testo in chiaro scelto. Lo schema One-time pad è anche vulnerabile a un attacco di testo cifrato scelto. Analizziamo il caso CTR$.

Attacco allo schema CTR$

Sia $F : K \times \{0,1\}^n \to \{0,1\}^n$ una famiglia di funzioni e sia $SE = (KeyGen, Enc, Dec)$ lo schema di cifratura simmetrica CTR$ associato. La debolezza dello schema che lo rende suscettibile ad un attacco a testo cifrato scelto è la seguente. Diciamo che $\langle r, C \rangle$ è un testo cifrato di un messaggio M di n bit , e capovolgiamo il bit i di C, creando un nuovo testo cifrato $\langle r, C' \rangle$. Sia M' il messaggio ottenuto decifrando il nuovo testo cifrato. Allora M è uguale a M' con l'i-esimo bit capovolto. Per convincerci di quanto detto basta analizzare il modo in cui vengono cifrati e decifrati i messaggi nello schema CTR$.
Quindi, effettuando una query all'oracolo di decrittazione di $\langle r, C' \rangle$ si può apprendere M' e quindi M. Di seguito, mostriamo come questa idea può essere applicata per rompere lo schema nel nostro modello, scoprendo in quale mondo è stato collocato un avversario.

Preposizione

Sia $F : K \times \{0,1\}^n \to \{0,1\}^n$ una famiglia di funzioni e sia $SE = (KeyGen, Enc, Dec)$ il corrispondente schema di cifratura simmetrica CTR$. Quindi:

$$Adv_{SE}^{ind-cca}(t, 1, n, 1, 2n) = 1$$

per $t = O(n)$ più il tempo per un'applicazione di F.

Il vantaggio di questo avversario è 1 anche se utilizza pochissime risorse: solo una query per ogni oracolo. Questa è chiaramente un'indicazione che lo schema è insicuro.

Dimostrazione

Presenteremo un algoritmo avversario A, con complessità temporale t, che esegue 1 query al suo oracolo di cifratura lr, questa query è di lunghezza n, effettua 1 query al suo oracolo di decrittazione, questa query è di lunghezza $2n$ e ha:

$$Adv_{SE}^{ind-cca}(A) = 1$$

Ricorda che l'oracolo di cifratura lr $E_k(LR(\cdot, \cdot, b))$ accetta in input una coppia di messaggi e restituisce una cifratura del messaggio sinistro o destro della coppia, a seconda del valore di b. L'obiettivo di A è determinare il valore di b. Il nostro avversario funziona così:

$$\begin{array}{l} \text{Adversary } A^{\mathcal{E}_K(\mathrm{LR}(\cdot,\cdot,b))\,,\,\mathcal{D}_K(\cdot)} \\ \quad M_0 \leftarrow 0^n \;;\; M_1 \leftarrow 1^n \\ \quad \langle r, C \rangle \leftarrow \mathcal{E}_K(\mathrm{LR}(M_0, M_1, b)) \\ \quad C' \leftarrow C \oplus 1^n \\ \quad M \leftarrow \mathcal{D}_K(\langle r, C' \rangle) \\ \quad \text{If } M = M_0 \textbf{ then } \textbf{ return } 1 \text{ else return } 0 \end{array}$$

La singola query all'oracolo di cifratura lr dell'avversario è la coppia di messaggi distinti M_0, M_1, ***ciascuno lungo un blocco***. Viene restituito un testo cifrato $\langle r, C \rangle$. Si effettua il complementare di C per ottenere C' e quindi invia il testo cifrato $\langle r, C' \rangle$ all'oracolo di decrittazione. Scommette sul mondo 1 se ottiene M_0, altrimenti sul mondo 0. Nota che $\langle r, C' \rangle \neq \langle r, C \rangle$, quindi la query di decrittazione è legittima. Ora, noi affermiamo che:

$$Pr[Esp_{SE}^{ind-cca-1}(A) = 1] = 1$$
$$Pr[Esp_{SE}^{ind-cca-0}(A) = 1] = 0$$

Quindi $Adv_{SE}^{ind-cpa}(A) = 1 - 0 = 1$. A ha ottenuto questo vantaggio effettuando solo una query all'oracolo di cifratura lr, la cui lunghezza, che secondo le nostre convenzioni è solo la lunghezza di M_0, ovvero n bit, e solo una query all'oracolo di decrittazione, la cui lunghezza è di $2n$ bit (assumendo una codifica di $\langle r, X \rangle$ come $n + |X|$ bit). Quindi:

$$Adv_{SE}^{ind-cca}(t, 1, n, 1, 2n) = 1$$

Perché le due equazioni sopra riportate sono vere?

Bisogna tornare alle definizioni delle due quantità in questione, così come alla descrizione dello schema stesso, e percorrerlo.

Nel mondo 1, che significa $b = 1$, sia $\langle r, C \rangle$ il testo cifrato restituito dall'oracolo di cifratura lr. Allora:

$$C = F_k(r + 1) \oplus M_1 = F_k(r + 1) \oplus 1^n$$

Si noti che:

$$\begin{gathered} M = D_k(\langle r, C' \rangle) = \\ = F_k(r + 1) \oplus C' = \\ = F_k(r + 1) \oplus C \oplus 1^n = \\ = F_k(r + 1) \oplus (F_k(r + 1) \oplus 1^n) \oplus 1^n = \\ = 0^n = \\ = M_0 \end{gathered}$$

Pertanto, l'oracolo di decrittazione restituirà M_0 e A ritornerà 1.

Nel mondo 0, che significa $b = 0$, sia $\langle r, C \rangle$ il testo cifrato restituito dall'oracolo di cifratura lr. Allora:

$$C = F_k(r + 1) \oplus M_1 = F_k(r + 1) \oplus 0^n$$

Si noti che:

$$\begin{gathered} M = D_k(\langle r, C' \rangle) = \\ = F_k(r + 1) \oplus C' = \\ = F_k(r + 1) \oplus C \oplus 1^n = \\ = F_k(r + 1) \oplus (F_k(r + 1) \oplus 0^n) \oplus 1^n = \\ = 1^n = \\ = M_1 \end{gathered}$$

Pertanto, l'oracolo di decrittazione restituirà M_1 e A ritornerà 0, con probabilità zero.

Considerazione

L'attacco è indipendente dalla qualità della funzione F. Se anche fosse una funzione casuale, l'avversario avrebbe lo stesso vantaggio. Questo dimostra che gli attacchi CCA sono potenzialmente molto più devastanti degli attacchi CPA.