Azzolini Riccardo 2021-03-29

Modalità di funzionamento per la cifratura a blocchi

1 Modalità di funzionamento

Gli algoritmi di cifratura a blocchi visti finora prendono in input blocchi di dimensione fissa, che in genere sono molto più piccoli dei tipici messaggi che si desidera cifrare. Allora, è necessario adottare degli approcci, chiamati **modalità di funzionamento**, che definiscano come utilizzare la cifratura a blocchi su messaggi di lunghezza arbitraria.

Siccome chi cifra e chi decifra un messaggio devono usare la stessa modalità di funzionamento, queste modalità sono standardizzate:

- Nel 1981 fu pubblicato un primo standard (NIST FIPS 81 / ANSI X3.106-1983) che definiva 4 modalità di funzionamento per il DES. Queste stesse modalità sono comunque utilizzabili anche per altri algoritmi di cifratura, come ad esempio AES.
- Nel 2001 (NIST Special Publication 800-38A) fu aggiunta una quinta modalità.
- Nel 2010 (NIST Special Publication 800-38E) venne standardizzata una modalità specifica per la cifratura con AES dei dispositivi di memorizzazione dei dati (hard disk, ecc.).

Tralasciando quella specifica per i dispositivi di memorizzazione, che non verrà presentata, le 5 principali modalità di funzionamento standard sono:

- **ECB** — **E**lectronic **C**ode**b**ook;
- **CBC** — **C**ipher **B**lock **C**haining;
- **CFB** — **C**ipher **F**eed**b**ack;
- **OFB** — **O**utput **F**eed**b**ack;
- **CTR** — **C**oun**t**e**r**.

Di solito, per indicare la modalità di funzionamento usata si aggiunge la sigla di tale modalità al nome dell'algoritmo di cifratura: ad esempio, per indicare l'uso di AES con chiave a 128 bit (AES-128) in modalità Cipher Block Chaining si può scrivere AES-128-CBC.

2 Electronic Codebook

La più semplice modalità di funzionamento è **ECB**, **Electronic Codebook**: il messaggio in input viene suddiviso in blocchi della dimensione richiesta dall'algoritmo di cifratura (64 bit per DES e 128 bit per AES), e ogni blocco viene cifrato indipendentemente con la stessa chiave (perché mittente e destinatario condividono una sola chiave, altrimenti si avrebbe lo stesso problema dello one-time pad: la necessità di condividere una quantità di informazioni segrete proporzionale alla dimensione del messaggio). Il destinatario esegue la decifratura allo stesso modo, cioè indipendentemente per ciascun blocco del messaggio, e poi riassembla i blocchi decifrati per ottenere il messaggio completo.

Questa modalità ha un grosso svantaggio: una volta fissata la chiave, uno stesso blocco di testo in chiaro viene sempre trasformato nello stesso blocco di testo cifrato, quindi eventuali blocchi uguali nel plaintext saranno uguali anche nel ciphertext. Così, studiando i blocchi ripetuti, un attaccante otterrebbe informazioni sul messaggio in chiaro, che potrebbe sfruttare in vari modi. Per questo motivo, la modalità ECB risulta in pratica adeguata solo per la cifratura di messaggi che hanno dimensioni non superiori a pochi blocchi di input dell'algoritmo di cifratura e una probabilità sufficientemente bassa di contenere ripetizioni. Un esempio tipico di messaggio per cui ECB va bene sono le chiavi, che di solito sono piuttosto piccole e generate in modo sostanzialmente casuale, quindi è altamente improbabile che contengano blocchi ripetuti.

Il termine "codebook" che dà il nome a questa modalità si riferisce proprio al fatto che ogni blocco in chiaro è associato sempre allo stesso blocco cifrato, così come un tradizionale codebook associa a ogni parola che può comparire nel testo in chiaro una determinata parola in codice.

Un altro svantaggio di ECB è la necessità di inserire dei bit di **padding** (riempimento) alla fine del messaggio quando questo ha una lunghezza che non è un multiplo della dimensione del blocco, perché la cifratura può essere effettuata solo su blocchi interi. Al fine di interpretare correttamente il messaggio decifrato, il destinatario deve sapere quale parte dell'ultimo blocco decifrato sia da considerare padding, in modo da poterla scartare: questa è un'informazione in più che le due entità comunicanti devono condividere, ed è dunque un ulteriore aspetto che è stato necessario standardizzare.

Un vantaggio di ECB è invece la possibilità di eseguire in parallelo la cifratura o la decifratura di più blocchi, ma in pratica ciò non è utile per i messaggi piccoli per cui ECB è adeguata.

3 Cipher Block Chaining

Una possibile soluzione al problema dei blocchi uguali di ECB è modificare, "sporcare" il plaintext, aggiungendo del *rumore*, prima che questo vada in input all'algoritmo di cifratura, in modo da garantire che blocchi di testo in chiaro uguali diventino diversi,

così che la cifratura, fatta sempre con la stessa chiave, produca blocchi cifrati diversi. L'aggiunta del rumore deve essere invertibile: quando il destinatario esegue l'algoritmo di decifratura, ottiene in output il plaintext modificato, e per recuperare quello originale deve rimuovere il rumore.

Un'operazione che permette di aggiungere e rimuovere facilmente il rumore è lo XOR: se il rumore è una sequenza di bit lunga quanto il blocco, allora può essere aggiunto a un blocco in chiaro facendo lo XOR bit a bit del blocco con il rumore, e può essere rimosso facendo di nuovo lo XOR del blocco "sporco" con la stessa sequenza di bit del rumore. Se poi il rumore è diverso per ogni blocco da cifrare, lo XOR assicura appunto che eventuali blocchi uguali diventino diversi.

Lo schema appena descritto risolve il problema dei blocchi uguali, ma ne introduce un altro: la sequenza di valori di rumore usati per i vari blocchi di un messaggio dovrebbe idealmente essere casuale, e deve essere condivisa tra mittente e destinatario per consentire la cifratura e la decifratura, dunque ci si ritroverebbe a dover condividere un'informazione lunga quanto il messaggio (come nel caso dello one-time pad), il che non è fattibile in pratica.

Quest'ultimo problema viene risolto *usando come rumore per ciascun blocco in chiaro il ciphertext del blocco precedente*: in simboli, se P_i è l'i-esimo blocco in chiaro e C_{i-1} è il blocco cifrato precedente, allora l'i-esimo blocco in chiaro modificato è $P'_i = P_i \oplus C_{i-1}$. Il destinatario riceve (come sempre) tutti i blocchi cifrati, perciò conosce il rumore da mettere in XOR con i blocchi in chiaro modificati ottenuti dalla decifratura per recuperare i blocchi in chiaro originali: $P_i = P'_i \oplus C_{i-1}$. La modalità di funzionamento così definita prende il nome di **CBC**, **Cipher Block Chaining**, perché realizza una "catena" di operazioni di cifratura, nella quale *ciascun blocco cifrato dipende da tutti i blocchi in chiaro precedenti.*

Per completare la definizione della modalità CBC rimane ancora un ultimo aspetto da gestire: quando si cifra il primo blocco in chiaro, P_1, non si ha ancora a disposizione un ciphertext di un blocco precedente da usare come rumore, quindi il valore iniziale del rumore deve essere fornito in un qualche altro modo. A tale scopo, si usa un numero casuale lungo quanto il blocco, che prende il nome di **vettore di inizializzazione** (**initialization vector**, **IV**): il primo blocco del plaintext modificato viene calcolato come $P'_1 = P_1 \oplus IV$. Allora, affinché possa risalire al testo in chiaro originale, anche il destinatario deve conoscere il vettore di inizializzazione: questa è un'informazione in più da condividere, ma non è richiesto che sia segreta, dunque può facilmente essere spedita in chiaro insieme al messaggio cifrato.

Una definizione più formale della cifratura con la modalità CBC è

$$\boxed{\begin{aligned} &C_0 = IV \\ &C_i = E_K(P_i \oplus C_{i-1}) \quad \text{per } i \geq 1 \end{aligned}}$$

dove E_K rappresenta la cifratura con una chiave K. Da qui si può ricavare la definizione della decifratura,

$$\begin{aligned} C_i &= E_K(P_i \oplus C_{i-1}) \\ D_K(C_i) &= D_K(E_K(P_i \oplus C_{i-1})) = P_i \oplus C_{i-1} \\ D_K(C_i) \oplus C_{i-1} &= P_i \oplus C_{i-1} \oplus C_{i-1} = P_i \end{aligned}$$

cioè

$$\boxed{P_i = D_K(C_i) \oplus C_{i-1} \quad \text{per } i \geq 1}$$

dove D_K rappresenta la decifratura con una chiave K (la stessa usata per la cifratura), e si pone ancora $C_0 = IV$.

3.1 Caratteristiche

I pregi e i difetti delle diverse modalità di funzionamento possono essere confrontati valutando alcune caratteristiche importanti: la necessità di padding, la possibilità di eseguire la cifratura o decifratura in parallelo e la propagazione dell'errore.

- La modalità CBC, così come ECB, ha lo svantaggio di poter operare solo su blocchi interi, dunque necessita di padding quando la lunghezza del messaggio non è un multiplo della lunghezza del blocco.
- Un altro svantaggio della modalità CBC è che la cifratura *non può* essere eseguita in parallelo, bensì solo in modo sequenziale, perché l'operazione di XOR effettuata su ciascun blocco in chiaro prima di cifrarlo richiede l'output della cifratura del blocco precedente. Invece, la decifratura *può* essere eseguita in parallelo: come si può vedere dalla formula $P_i = D_K(C_i) \oplus C_{i-1}$, per decifrare l'$i$-esimo blocco è sufficiente conoscere i due blocchi cifrati C_i e C_{i-1}, che il destinatario ha immediatamente a disposizione (se si assume che il riceva insieme tutti i blocchi del ciphertext).
- La **propagazione dell'errore** conta il numero di blocchi in chiaro che vengono corrotti se si verifica un errore di trasmissione in un singolo blocco del ciphertext. Inevitabilmente, a prescindere dalla modalità di funzionamento, un errore di trasmissione nel blocco cifrato C_i comporta la decifratura errata di C_i stesso: l'output della decifratura non può essere il blocco in chiaro P_i corretto. Tuttavia, nel caso di CBC, C_i partecipa anche allo XOR per la rimozione del rumore nella decifratura del blocco C_{i+1}, dunque non si ottiene correttamente neanche P_{i+1}. Si ha allora una propagazione dell'errore di un passo, cioè un errore di trasmissione in un blocco cifrato viene propagato su due blocchi decifrati.