

Modalità operative di cifratura

Paolo D'Arco
pdarco@unisa.it

Università di Salerno

Elementi di Crittografia

- 1 Modalità operative per stream cipher
- 2 Modalità operative per block cipher

Esistono diversi modi per cifrare in maniera sicura **messaggi lunghi** usando

- Stream cipher
- Cifrari a blocchi

Modalità operative per stream cipher

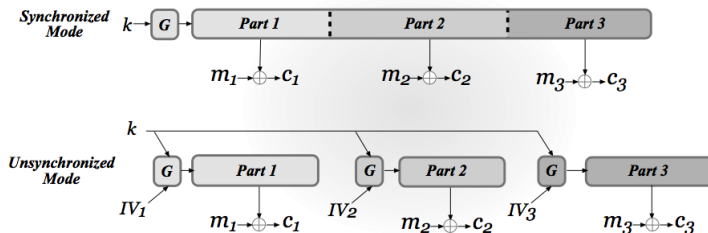
Abbiamo visto con la costruzione 3.17 (costruzione con PRG) come cifrare messaggi di lunghezza fissata in modo EAV-sicuro.

Gli stream cipher possono essere visti come PRG *flessibili*.

Possono essere utilizzati in

- modo sincrono (con stato)
- modo asincrono (senza stato)

Cifratura con stream cipher: sincrona e asincrona



Sincrono: viene generata una lunga stringa pseudocasuale e una parte diversa viene usata per ogni messaggio

Abbiamo visto come costruire G_ℓ , con ℓ fisso, da uno stream cipher

Possiamo estendere la costruzione per ottenere un generatore a lunghezza variabile

$$G_\infty(s, 1^\ell)$$

che invoca $Init(s)$ e poi $GetBits$ esattamente ℓ volte.

Cifratura e decifratura diventano

$$c := G_\infty(k, 1^{|m|}) \oplus m \qquad m := c \oplus G_\infty(k, 1^{|c|}).$$

Se le parti mantengono uno stato, i messaggi $m_1, m_2, \dots$, possono essere visti come un messaggio lungo.

$st_0 := \text{Init}(k)$

Per cifrare m_1 , il trasmittente

invoca *getBits* ℓ_1 volte e calcola $c_1 := \text{pad}_1 \oplus m_1$

Per decifrare c_1 , il ricevente

invoca *getBits* ℓ_1 volte e calcola $m_1 := \text{pad}_1 \oplus c_1$

Per cifrare m_2 , le parti riprendono le computazioni dallo stato st_{ℓ_1} , e procedono allo stesso modo.

Per i messaggi successivi si procede analogamente, partendo dallo stato raggiunto all'ultima cifratura.

Questa modalità é utile per una sessione tra le parti.

Non funziona bene per comunicazioni sporadiche o quando una parte può comunicare da diversi dispositivi.

Cifratura in modalità asincrona

Nella modalità asincrona la funzione $Init(\cdot)$ accetta un vettore di inizializzazione IV , cioè

$$st_0 = Init(s, IV).$$

Il generatore ha 3 input: seme, vettore di inizializzazione, lunghezza output desiderata

$$G_{\infty}(s, IV, 1^{\ell})$$

Per cifrare m di lunghezza ℓ usando la chiave k , il trasmittente

sceglie $IV \leftarrow \{0, 1\}^n$ e calcola $c = \langle c_1, c_2 \rangle := \langle IV, G_{\infty}(k, IV, 1^{\ell}) \oplus m \rangle$

Per decifrare c usando la chiave k , il ricevente

$$\text{calcola } m := G_{\infty}(k, c_1, 1^{\ell}) \oplus c_2$$

Se lo stream cipher ha la proprietà che, per ogni ℓ ,

$$F_k(IV) \stackrel{\text{def}}{=} G_\infty(k, IV, 1^\ell)$$

è una PRF, allora lo schema di cifratura è CPA sicuro.

Osservazione: i vettori di inizializzazione vengono scelti *uniformemente a caso*



F pertanto deve solo essere *debolmente* pseudocasuale, cioè pseudocasuale rispetto ad input scelti uniformemente a caso.

(Ricordo che nella definizione di funzione pseudocasuale gli input possono essere scelti in accordo a *qualsiasi* distribuzione)

Modalità operative per cifrari a blocchi

Abbiamo visto uno schema di cifratura CPA-sicuro basato su PRF (cifrario a blocchi)

- la lunghezza del cifrato é doppia rispetto alla lunghezza del messaggio

Le modalità operative per i cifrari a blocchi sono metodi per cifrare messaggi di lunghezza arbitraria con cifrati "corti".

Sia F un cifrario a blocchi con lunghezza di blocco n . Sia

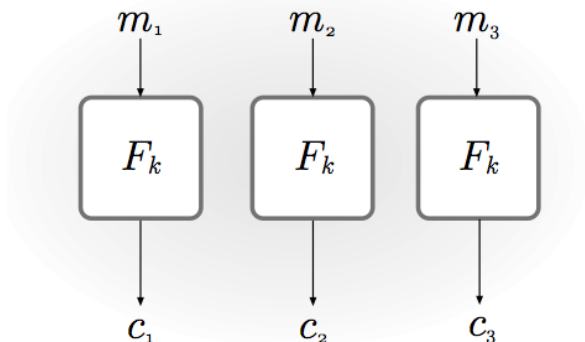
$$m = m_1 m_2 \dots m_\ell, \quad m_i \in \{0, 1\}^n \quad \text{per ogni } i = 1, \dots, \ell.$$

Se necessario l'ultimo blocco viene "completato" (padded) in modo tale che $|m_\ell| = n$

La modalità operativa più semplice é l'Electronic Code Book (ECB) mode

$$c := \langle F_k(m_1), F_k(m_2), \dots, F_k(m_\ell) \rangle$$

ECB mode



ECB é deterministico

- Non può dare sicurezza CPA
- Non ha neanche cifrature indistinguibili rispetto ad un eavesdropper
 - se un blocco si ripete nel messaggio in chiaro, allora il blocco si ripeterà nel cifrato

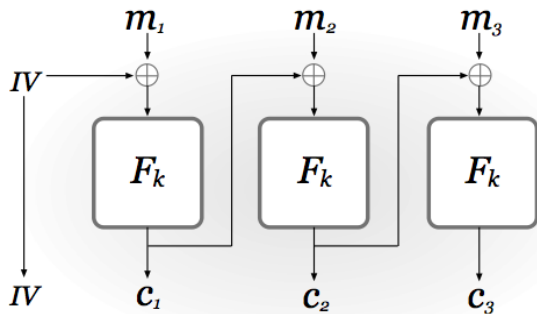
$$m_1 m^* m_3 m_4 m^* \dots \quad \Rightarrow \quad c_1 c^* c_3 c_4 c^* \dots$$

É facile allora distinguere tra due messaggi

- Adv sceglie m_0 con un blocco ripetuto ed m_1 con blocchi tutti distinti e analizza i blocchi del cifrato

In conclusione: ECB non dovrebbe mai essere usato

Modalità Cipher Block Chaining - CBC mode



Viene scelto uniformemente il vettore di inizializzazione $IV \in \{0,1\}^n$. Poi,

$$c_0 = IV, \quad c_i = F_k(c_{i-1} \oplus m_i), \quad \text{per } i = 1, \dots, \ell.$$

Modalità Cipher Block Chaining - CBC mode

La decifratura del cifrato $\langle c_0, c_1, \dots, c_\ell \rangle$ viene effettuata calcolando

$$m_i = F_k^{-1}(c_i) \oplus c_{i-1}, \quad \text{per } i = 1, \dots, \ell.$$

CBC é una modalità *probabilistica*

Se F é una permutazione pseudocasuale $\Rightarrow$ la modalità CBC é CPA-sicura.

Inconveniente: la cifratura deve essere effettuata sequenzialmente.

Nota: se il vettore di inizializzazione IV *non* viene scelto uniformemente in $\{0, 1\}^n$ ma semplicemente in modo che i valori siano distinti

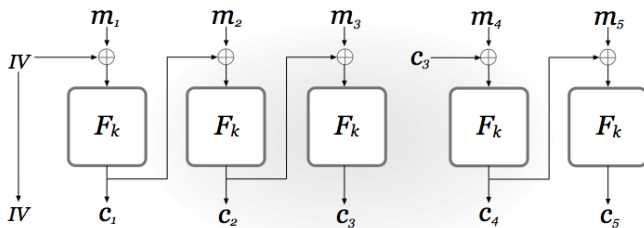


la variante *non* é piú sicura.

Chained CBC mode

Una variante di CBC utilizzata é la modalità con concatenazione.

Idea: invece di usare un nuovo IV per ciascuna cifratura, usa *l'ultimo blocco* c_ℓ del cifrato precedente come IV .



É una variante con stato.

Usata in SSL 3.0 e in TLS 1.0.

Sembra *tanto* sicura *quanto* CBC.

Purtroppo **non** é cosí!

É vulnerabile ad attacchi di tipo chosen-plaintext.

L'attacco si basa sul fatto che il vettore *IV* é noto *prima* che la seconda cifratura abbia luogo.

Adv sa che $m_1 \in \{m_1^0, m_1^1\}$

- Osserva il primo cifrato $c := \langle IV, c_1, c_2, c_3 \rangle$ di $m_1 m_2 m_3$
- Chiede una cifratura all'oracolo di $m_4 m_5$ dove $m_4 = IV \oplus m_1^0 \oplus c_3$
- Ottiene $\langle c_4, c_5 \rangle$
- Se $c_4 = c_1$ allora dá in output $b' = 0$ (i.e., $m_1 = m_1^0$).
Altrimenti, $b' = 1$.

È facile verificare che $c_4 = c_1$ se e solo se $m_1 = m_1^0$. Infatti:

$$c_4 = F_k(m_4 \oplus c_3) = F_k((IV \oplus m_1^0 \oplus c_3) \oplus c_3) = F_k(IV \oplus m_1^0)$$

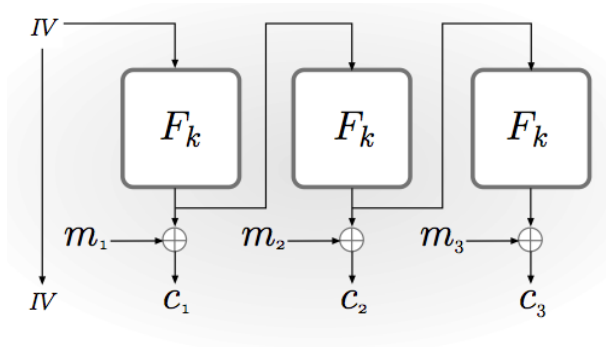
mentre

$$c_1 = F_k(IV \oplus m_1).$$

Lezione: ogni modifica ad uno schema crittografico può essere pericolosa, anche se questa modifica sembra ragionevole ed innocua.

Modalità Output Feedback Mode - OFB mode

Realizza uno stream cipher asincrono



Modalità Output Feedback Mode - OFB mode

Viene scelto uniformemente il vettore di inizializzazione $IV \in \{0,1\}^n$.

Si genera la stringa pseudocasuale

$$y_0 = IV, \quad y_i = F_k(y_{i-1}), \quad \text{per } i = 1, \dots, \ell.$$

Ciascun blocco m_i del messaggio in chiaro viene cifrato calcolando

$$c_0 = y_0, \quad c_i = m_i \oplus y_i, \quad \text{per } i = 1, \dots, \ell.$$

La decifratura del cifrato $\langle c_0, c_1, \dots, c_\ell \rangle$ viene effettuata calcolando

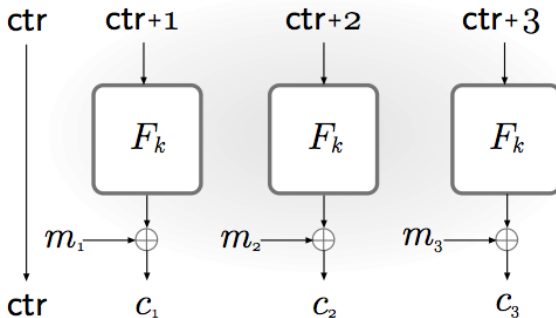
$$y_0 = c_0, \quad y_i = F_k(y_{i-1}), \quad \text{e} \quad m_i = c_i \oplus y_i, \quad \text{per } i = 1, \dots, \ell.$$

Osservazioni:

- Non é necessario che F sia invertibile (neanche che sia una permutazione).
- Non é necessario che il messaggio abbia una lunghezza multipla di n . La stringa pseudocasuale può essere troncata dove serve.
- La variante con stato (senza IV per cifrature successive) é sicura. Equivale ad uno stream cipher sincronizzato.
- La modalità OFB é CPA-sicura se F é una funzione pseudocasuale.
- Il "pad" pseudocasuale può essere pre-computato.

Modalità Counter Mode - CTR mode

Puó essere vista come uno stream cipher asincrono costruito a partire da un cifrario a blocchi.



Modalità Counter Mode - CTR mode

Viene scelto uniformemente $ctr \in \{0, 1\}^n$.

Si procede poi come segue

$$y_0 = ctr, \quad y_i = F_k((ctr + i) \bmod 2^n), \quad \text{per } i = 1, \dots, \ell.$$

Ciascun blocco m_i del messaggio in chiaro viene cifrato calcolando

$$c_0 = y_0, \quad c_i = m_i \oplus y_i, \quad \text{per } i = 1, \dots, \ell.$$

La decifratura del cifrato $\langle c_0, c_1, \dots, c_\ell \rangle$ viene effettuata calcolando

$$ctr = c_0, \quad y_i = F_k((ctr + i) \bmod 2^n), \quad \text{e} \quad m_i = c_i \oplus y_i, \quad \text{per } i = 1, \dots, \ell.$$

Osservazioni

- La decifratura non richiede che F sia invertibile (né tantomeno che sia una permutazione)
- Lo stream generato può essere troncato alla lunghezza del messaggio in chiaro e può essere generato in anticipo.
- La variante "con stato" della modalità CTR é sicura.
- CTR può essere *parallelizzata*.
- La decifratura può essere selettiva: decifrare direttamente l' i -esimo blocco del cifrato.

Teorema 3.2 Se F é una funzione pseudocasuale, allora la modalit  CTR   CPA-sicura.

Dim. Utilizziamo la stessa strategia dimostrativa usata in precedenza. Sia $\tilde{\Pi} = (\tilde{Gen}, \tilde{Enc}, \tilde{Dec})$ costruito a partire da $\Pi = (Gen, Enc, Dec)$, tale che

- $\tilde{\Pi}$ usa $f \in Func_n$ scelta uniformemente a caso
- Π usa F_k , dove k   scelta uniformemente a caso

Lo schema   naturalmente un'ipotesi di lavoro.

Per ogni Adv A PPT sia $q(n)$ un limite superiore al numero di query che $A(1^n)$ rivolge al suo oracolo per la cifratura.

È facile far vedere (come prima) che esiste una funzione trascurabile *negl* tale che

$$|Pr[PrivK_{A,\Pi}^{cpa}(n) = 1] - Pr[PrivK_{A,\tilde{\Pi}}^{cpa}(n) = 1]| \leq \text{negl}(n).$$

Mostriamo quindi che

$$Pr[PrivK_{A,\tilde{\Pi}}^{cpa}(n) = 1] \leq 1/2 + \frac{2 \cdot q(n)^2}{2^n},$$

da cui risulta che

$$Pr[PrivK_{A,\Pi}^{cpa}(n) = 1] \leq 1/2 + \frac{2 \cdot q(n)^2}{2^n} + \text{negl}(n).$$

Vedi il libro di testo per i dettagli.

- Le modalità operative di cifratura vengono spesso comparate anche rispetto alla capacità di protezione rispetto alla contraffazione dell'informazione.
 - Tuttavia la verifica dell'integrità richiede un tool differente
- La lunghezza del blocco nei cifrari a blocchi deve essere scelta con cura
 - Per esempio, nel CTR-mode, se il ctr è di ℓ bit, dopo $2^{\ell/2}$ ctr uniformi lo stesso valore ricompare. Il parametro ℓ non può essere troppo corto.
- Se il vettore IV non è scelto uniformemente la sicurezza non è garantita
 - Se IV si ripete, con OFB e CTR un Adv può recuperare informazioni importanti sul plaintext semplicemente calcolando l'xor dei due cifrati. Con CBC la situazione migliora.
- Le versioni "con stato" di OFB e CTR sono sicure.