Sicurezza e Crittografia

Anno Accademico 2018-2019

Homework 1

Matteo Berti

27 settembre 2019

Esercizio 1.

Il cifrario a sostituzione polialfabetica può essere definito formalmente specificando lo schema di codifica $\Pi = (Gen, Enc, Dec)$ come segue:

- *Gen*: algoritmo PPT che prende in input una stringa nella forma 1^n e produce in output una chiave composta da n caratteri scelti in modo casuale e indipendente tra loro. Ogni carattere in output rappresenta una particolare traslazione dell'alfabeto su cui si basa il messaggio in chiaro chiamato Σ_C, generando così n alfabeti $\Sigma_1, ..., \Sigma_n$, in cui l'alfabeto Σ_i è un alfabeto traslato in modo tale che abbia come prima lettera l'$i-esimo$ carattere della chiave.
- *Enc*: algoritmo PPT che prende in input una coppia (k, m) di stringhe, precisamente la chiave e il messaggio in chiaro. L'operazione di cifratura, per ogni singolo carattere j del messaggio in chiaro:
  1. calcola la sua posizione all'interno dell'alfabeto chiaro Σ_C ottendendo p
  2. calcola il carattere u in posizione p all'interno dell'alfabeto Σ_i dove $i = j \ mod \ |k|$
  3. accoda il carattere u alla stringa c del messaggio cifrato.

  Infine restituisce in output la stringa c del messaggio cifrato.
- *Dec*: algoritmo PPT che prende in input una coppia (k, c) di stringhe, precisamente la chiave e il messaggio cifrato. L'operazione di decifratura, per ogni singolo carattere j del messaggio cifrato:
  1. calcola la sua posizione all'interno dell'alfabeto Σ_i dove $i = j \ mod \ |k|$ ottendendo p
  2. calcola il carattere v in posizione p all'interno dell'alfabeto in chiaro Σ_C
  3. accoda il carattere v alla stringa m del messaggio in chiaro.

  Infine restituisce in output la stringa m del messaggio in chiaro.

Per dimostrare che il cifrario non è sicuro contro attacchi passivi è necessario dimostrare che per almeno un A **non** vale:

$$Pr(PrivK^{eav}_{\Pi,A} = 1) = \frac{1}{2} + \epsilon(n)$$

Questo è fattibile dimostrando che tramite il test $PrivK^{eav}_{\Pi,A}$ si può sempre riuscire a distinguere tra due messaggi, se opportunamente creati, quale testo cifrato appartiene a quale messaggio in chiaro. Prima di tutto è necessario assumere che la lunghezza del periodo sia molto inferiore alla lunghezza dei messaggi, altrimenti se fossero simili basterebbe allungare la chiave alla dimensione di m per avere uno scenario simile a One Time Pad. Dopodichè è sufficiente prendere un m_0

come una serie di caratteri tutti uguali, mentre m_1 come una serie di caratteri tutti uguali ad m_0 tranne l'ultimo carattere che sarà differente. In questo modo distinguere la stringa di caratteri tutti uguali sarà banale, infatti una volta notato il pattern di ripetizione si controlla che questo pattern venga ripetuto fino alla fine, in questo caso il messaggio in chiaro è m_0. Se non viene ripetuto fino alla fine, ma l'ultimo carattere interrompe il pattern allora si è in presenza di m_1. In questo modo la probabilità di distinguere i messaggi da parte dell'avversario è 1 con conseguente rottura della definzione di sicurezza contro attacchi passivi data.

Esercizio 2.

Per definizione perchè G^ξ sia un generatore pseudocasuale (GP) è necessario che valgano le seguenti condizioni:

- Per ogni $n \in \mathbb{N}$ vale che $l^\xi(n) > n$, dove $l^\xi(n)$ è il fattore di espansione di G^ξ, definito come $l^\xi(n) = l(n) - \xi(n)$
- G^ξ è polytime
- Per ogni algoritmo PPT D esiste $\epsilon \in NGL$ tale che:

  $$|Pr(D(s) = 1) - Pr(D(G^\xi(r)) = 1)| \leq \epsilon(n)$$

  Con s, r casuali di lunghezza rispettivamente $l^\xi(n)$ e n.

Queste condizioni sappiamo valgono per G. È necessario quindi dimostrare che queste tre condizioni valgono anche per G^ξ.

1. Per il primo punto è necessario dimostrare che il fattore di espansione $l^\xi(n)$ sia maggiore di n. Per definzione in G^ξ si espandono stringhe fino ad $l(n) \geq n + 1 + \xi(n)$, sapendo che $l^\xi(n) = l(n) - \xi(n)$ sostituendo $l(n)$ si ottiene $l^\xi(n) \geq n + 1 + \xi(n) - \xi(n)$ che equivale a $l^\xi(n) \geq n + 1$ che è effettivamente sempre maggiore di n.

2. G^ξ è per forza anch'esso polytime in quanto il processo di produzione della stringa pseudocasuale rimane il medesimo, lasciando inaltrerata la complessità, semplicemente vengono sottratti elementi alla fine.

3. Si vuole dimostrare ora che esiste un fattore $\mu(n)$ anch'esso trascurabile, che identifica la possibilità di distinguere tra una stringa generata pseudocasualmente da G^ξ ed una stringa realmente casuale. Questo si può dimostrare per assurdo, supponiamo che esista un qualche distinguitore D^ξ per cui vale:

   $$|Pr(D^\xi(s) = 1) - Pr(D^\xi(G^\xi(r)) = 1)| = \mu(n)$$

   e $\mu(n)$ sia per assurdo non trascurabile. Allora esiste un distinguitore D che distingue le stringhe pseudocasuali di G da quelle realmente casuali con un successo non trascurabile. Prima di tutto se la stringa da distinguere è una stringa pseudocasuale generata da G allora "troncandola" si otterrebbe la stringa generata da G^ξ.

   Poichè D^ξ è un distinguitore per G^ξ, ovvero di stringhe di lunghezza $l^\xi(n) = l(n) - \xi(n)$ con vantaggio non trascurabile, allora la probabilità che D distingua una stringa pseudocasuale da una casuale **usando** D^ξ è esattamente la probabilità che D^ξ distingua la stringa generata da G^ξ da una casuale "troncata".

   Avendo supposto ciò allora D sarebbe in grado di distinguere stringhe pseudocasuali da stringhe casuali con probabilità non trascurabile $\mu(n)$, tuttavia ciò è assurdo in quanto si è assunto che G fosse un generatore pseudocasuale, e quindi non esistano distinguitori. L'assurdo sta nel fatto che si è dimostrato che esiste un distinguitore D se esiste un distinguitore per il generatore pseudocasuale G^ξ, quindi le premesse risultano sbagliate e le stringhe generate da G^ξ non hanno distinguitori. E di fatto:

   $$|Pr(D^\xi(s) = 1) - Pr(D^\xi(G^\xi(r)) = 1)| = |Pr(D(s) = 1) - Pr(D(G(r)) = 1)| = \mu(n)$$

Di fatto ulteriori operazioni di postprocessing non sono fondamentali per garantire maggiore sicurezza in quanto già dimostrato essere ugualmente sicuro.

Esercizio 3.

Per dimostrare che ogni schema sicuro rispetto ad attacchi passivi garantisce la segretezza degli ultimi 16 bit, sapendo che per ogni avversario PPT A vale che

$$Pr(PrivK^{eav}_{\Pi,A}(n) = 1) = \frac{1}{2} + \epsilon(n)$$

è necessario dimostrare che la stessa probabilità vale anche per l'esperimento $PrivK^{eav\_b}_{\Pi,A}$.

In cui l'esperimento $PrivK^{eav\_b}_{\Pi,A}$ non fa altro che, dato un messaggio m in input, chiede ad A di indovinare un bit (0 o 1) in una qualche posizione i (con $0 \leq i \leq 16$) della stringa in chiaro m conoscendo solo il testo cifrato c, l'esito dell'esperimento è 1 se A indovina il bit e 0 se non lo indovina. Dato lo schema $\Pi = (Gen, Enc, Dec)$:

Algorithm 1 $PrivK^{eav\_b}_{\Pi,A}$

$PrivK^{eav\_b}_{\Pi,A}(n, m)$:
$k \leftarrow Gen(1^n)$;
$c \leftarrow Enc(k, m)$;
$(i, b^*) \leftarrow A(c)$;
return $m_i = b^*$

Nota: si da per scontato che l'attaccante non scelga bit oltre i 16 anche se la dimostrazione sarebbe la stessa. Supponiamo ora per assurdo che esista una funzione $\mu(n)$ non trascurabile che dia un buon vantaggio ad un ipotetico avversario PPT A:

$$Pr(PrivK^{eav\_b}_{\Pi,A}(n) = 1) = \frac{1}{2} + \mu(n)$$

Allora A può definire un avversario PPT B che vinca con un vantaggio non trascurabile nell'esperimento base $PrivK^{eav}_{\Pi,B}$.

Quello che fa B altro non è che usare l'esperimento $PrivK^{eav}_{\Pi,B}$ per scegliere tra due messaggi m_0 ed m_1 (supponiamo uno composto da tutti 0 e l'altro da tutti 1), tuttavia per farsi aiutare chiede ad A con l'esperimento $PrivK^{eav_b}_{\Pi,A}$ di indovinare un bit dato il testo cifrato c di uno dei due messaggi. A questo punto A sarà in grado di restituire il valore di un bit e permettere a B di indovinare di quale messaggio si trattasse.

Dato che A è PPT, lo è anche B e per ipotesi A riesce nel suo intento con vantaggio non trascurabile $\mu(n)$ quindi si avrebbe:

$$Pr(PrivK^{eav\_b}_{\Pi,A}(n, m) = 1) = Pr(PrivK^{eav}_{\Pi,B}(n) = 1) = \frac{1}{2} + \mu(n)$$

Essendo $\mu(n)$ non trascurabile si avrebbe un assurdo e questo dimostra la trascurabilità della funzione in aggiunta a $\frac{1}{2}$.