Sicurezza e Crittografia

Anno Accademico 2018-2019

Homework 3

Matteo Berti

27 settembre 2019

Esercizio 1.
Per ogni predicato hc esiste sempre una funzione one-way f per la quale hc **non** è hardcore. Sia hc il predicato hardcore per la funzione one-way g allora è sempre possibile costruire un'altra funzione one-way f tale che:

$$f(x) = (g(x), hc(x))$$

Di fatto f mantiene tutte le caratteristiche di una funzione one-way, in quanto se si riuscisse ad invertirla si invertirebbe anche g, tuttavia hc viene rivelato nell'output e questo lo rende non idoneo ad essere un predicato hardcore per f.

Si prenda come caso specifico il predicato hc che restituisce l'ultimo bit della stringa passata come input:

$$hc(x_0, ..., x_n) = x_n$$

Considerando la funzione one-way $f(a, b)$ composta come segue:

$$f(a, b) = (g(a), b)$$

In cui $g(a)$ è una fuzione one-way che ha come predicato hardcore hc e b una porzione di bit > 1, è facile notare come hc non sia predicato hardcore per la funzione one-way $f(a, b)$.

Esercizio 2.
Perchè un gruppo $(\mathbb{G}, \circ)$ sia considerato abeliano deve soddisfare cinque assiomi: **chiusura**, **associatività**, presenza dell'**elemento identità** e e per ogni elemento $g \in \mathbb{G}$ deve esistere un **inverso** g^{-1}.

Si vuole dimostrare che l'elemento **identità** è **unico**. Per far ciò si suppone per contrapposizione che l'identità non sia unica, ma vi siano due elementi e ed e' distinti che siano entrambi identità. In questo caso possiamo applicare ad e l'operazione su e' ottenendo nuovamente e in quanto essendo e' l'identità, e non viene alterato:

$$e \circ e' = e$$

La stessa operazione può essere eseguita invertendo gli elementi, essendo anche e l'identità il risultato sarà e':

$$e' \circ e = e'$$

I gruppi abeliani godono della proprietà commutativa quindi:

$$e \circ e' = e' \circ e$$

E di conseguenza:

$$e = e'$$

Questo prova che non è possibile siano presenti due identità distinte all'interno di un gruppo abeliano, quindi l'*identità è unica.*

Si vuole dimostrare che l'elemento **inverso** è **unico**. Per far ciò si suppone per contrapposizione che l'elemento inverso di un generico g non sia unico, ma vi siano due inversi g^{-1} e g'^{-1}. In questo caso possiamo vedere g^{-1} come se stesso applicato all'identità:

$$g^{-1} = g^{-1} \circ e$$

Sapendo che l'identità è data da $e = g \circ g'^{-1}$ andando a sostituire abbiamo:

$$g^{-1} = g^{-1} \circ (g \circ g'^{-1})$$

Per la proprietà associativa ciò equivale a:

$$g^{-1} = (g^{-1} \circ g) \circ g'^{-1}$$

$$g^{-1} = e \circ g'^{-1}$$

$$g^{-1} = g'^{-1}$$

Quindi g^{-1} e g'^{-1} sono del tutto equivalenti, e ciò prova che all'interno di un gruppo abeliano non vi sono due elementi inversi distinti per un unico elemento, ma l'*inverso è unico.*

Esercizio 3.
È sempre possibile ottenere uno schema a chiave pubblica da un protocollo a due fasi. Intuitivamente vi è una prima fase in cui A genera una chiave privata e la utilizza per creare una seconda chiave pubblica che verrà poi inviata a B. Questa fase avviene già ad esempio nel protocollo per lo scambio di chiavi Diffie-Hellman.
A questo punto, dato che il fine è lo scambio di messaggi, B genera una chiave privata che utilizzerà sia per creare una nuova chiave s a partire dalla chiave pubblica di A che per creare la relativa chiave pubblica. Successivamente fonde il messaggio in chiaro m con la chiave "mista" s, e invia ad A quest'ultimo valore insieme alla propria chiave pubblica.
Ora A avrà tutti gli elementi necessari per risalire ad s e ad m, concludendo lo scambio di chiave e di messaggio.

$$\begin{aligned}
& (\mathbb{G}, q, g) \leftarrow GenCG(1^n) && (1)\\
A \quad & x \leftarrow \mathbb{Z}_q && (2)\\
& h \leftarrow g^x && (3)\\
& \textbf{Send } (\mathbb{G}, q, g, h) \textbf{ to } B && (4)\\
& && (5)\\
& y \leftarrow \mathbb{Z}_q && (6)\\
& s \leftarrow h^y && (7)\\
B \quad & i \leftarrow g^y && (8)\\
& c \leftarrow m \circ s && (9)\\
& \textbf{Send } (i, c) \textbf{ to } B && (10)\\
& && (11)\\
& s \leftarrow i^x && (12)\\
A \quad & m \leftarrow c \circ s^{-1} && (13)\\
& && (14)
\end{aligned}$$

Sappiamo essere *corretto* in quanto $i \leftarrow g^y$ quindi $s \leftarrow i^x = g^{yx}$ e viceversa se $h \leftarrow g^x$ allora $s \leftarrow h^y = g^{xy}$.
Per dimostrare che questo protocollo è *sicuro contro attacchi passivi* è necessario dimostrare che:

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

Prima di tutto sappiamo che un avversario A non può risalire ad x e y in quanto richiederebbe la risoluzione del problema del logaritmo discreto, ritenuto essere difficile. Inoltre sappiamo grazie all'assunzione computazionale di Diffie-Hellman che risalire a g^{xy} senza conoscere x o y è difficile. Non solo, ma grazie all'assunzione decisionale di Diffie-Hellman sappiamo che la probabilità per un avversario di riuscire a distinguere g^{xy} da un arbitrario elemento del gruppo $\mathbb{G}$ è trascurabile:

$$|Pr(A(\mathbb{G}, q, g, g^x, g^y, g^z)) - Pr(A(\mathbb{G}, q, g, g^x, g^y, g^{xy}))| \leq \epsilon(n)$$

Dove x, y e $z \in \mathbb{Z}_q$ sono *casuali*. Per passare da questo scenario in cui l'elemento random è preso all'interno del gruppo $\mathbb{G}$ allo scenario di $KE^{eav}_{A,\Pi}(n)$ in cui l'elemento random è una stringa presa con distribuzione uniforme è possibile convertire l'elemento del gruppo in una stringa tramite una funzione chiamata "estrattore di casualità". Quindi per un avversario riuscire a distinguere tra un valore random e la chiave usata all'interno del protocollo a due fasi descritto sopra $[Pr(KE^{eav}_{A,\Pi}(n) = 1)]$ equivale a "tirare a caso" più una quantità trascurabile data dalla presenza di g^{xy} non "puramente" random, questo conferma la **sicurezza** del protocollo.

Il protocollo descritto precedentemente è di fatto uno schema di codifica a chiave pubblica Π tale che:

- *Gen*: idealmente prende in input un parametro di sicurezza e fornisce in output due chiavi, in cui dalla prima sia possibile costruire la seconda ma non viceversa. Nello specifico si può pensare prenda in input un valore nella forma 1^n e restituisca una coppia $(x,\ g^x)$ tale per cui $x \in \mathbb{Z}_q$ dati $(\mathbb{G}, q, g) \leftarrow GenCG(1^n)$.

- Enc_h: idealmente prende in input la chiave dalla quale non è possibile risalire all'altra ed un messaggio da cifrare, in output restituisce il testo cifrato e una chiave dalla quale è possibile risalire a quella usata per cifrare il messaggio solo se si è in possesso della chiave correlata a quella passata in input. Nello specifico si può pensare prenda in input (oltre ad una chiave pubblica h) un messaggio da cifrare m, conoscendo $(\mathbb{G}, q, g)$ viene creato un elemento $y \leftarrow \mathbb{Z}_q$ da usare nella creazione di $s \leftarrow h^y$ elemento con il quale si cifra effettivamente il messaggio in chiaro $c \leftarrow m \circ s$. In output viene restituito il testo cifrato insieme ad $i \leftarrow g^y$ elemento con il quale sarà possibile ricostruire s.

- Dec_i: idealmente prende in input un testo cifrato ed una chiave e restituisce in output il messaggio in chiaro dopo aver ricostruito la chiave di cifratura usando una chiave segreta e quella ricevuta in input. Nello specifico si può pensare prenda in input (oltre ad un elemento i) un testo cifrato $c = m \circ s$ per risalire ad s e ricostruire m è sufficiente calcolare i^x, infatti $i^x = g^{yx} = g^{xy} = h^y = s$ da qui calcolare l'inversa di s conoscendo $(\mathbb{G}, q, g)$ è triviale.

Per dimostrare che questo schema di codifica a chiave pubblica è **sicuro contro attacchi passivi** è necessario dimostrare che:

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

Per far ciò è necessario pensare ad un possibile avversario A che in $KE^{eav}_{A,\Pi}$, ovvero nell'esperimento che dimostra che il *protocollo* è sicuro, riesce con vantaggio non trascurabile a vincere, questo grazie all'utilizzo di un avversario A' che supponiamo vincere in $PubK^{eav}_{A',\Pi}$. Si sta ipotizzando quindi si riesca a rompere lo schema di codifica pur sapendo il protocollo sia sicuro. A questo punto se A' fosse in grado di vincere con successo nell'esperimento $PubK^{eav}_{A',\Pi}$, A sarebbe in grado di distinguere se k^* è o meno la chiave prodotta dallo scambio con vantaggio non trascurabile. Questo avviene passando ad A' la chiave pubblica prodotta dal primo agente nello scambio, dopo la produzione di due messaggi e la cifratura di uno da parte del secondo agente nello scambio, A' sarebbe ancora in grado di indovinare con probabilità non trascurabile quale messaggio è stato scelto, restituendo ad A il risultato corretto il quale a sua volta sarebbe in grado di dire se k^* è o meno la chiave relativa al *transcript*. Ma questo porterebbe ad un assurdo in quanto abbiamo dimostrato in precedenza non sia possibile grazie al DDH distinguere tra una chiave prodotta dallo scambio e un valore k^* casuale. Di conseguenza anche l'avversario A' per $PubK^{eav}_{A',\Pi}$ non è in grado distinguere quale messaggio sia stato cifrato confermando la sicurezza contro attacchi passivi dello schema prodotto.

Chiaramente se uno schema a chiave pubblica è sicuro contro attacchi passivi come visto sopra allora **lo schema è CPA-sicuro**. Questo in quanto nell'esperimento $PubK^{eav}_{A,\Pi}$ la chiave pubblica è già nota all'avversario, che è in grado di crittare qualunque messaggio, anche senza l'accesso ad oracoli.