Random Oracle Model

Paolo D'Arco
pdarco@unisa.it

Universitá di Salerno

Elementi di Crittografia

Contenuti

Difficoltá di esibire prove

Per diverse costruzioni che utilizzano funzioni hash **non** si conoscono riduzioni di sicurezza basate su assunzioni tipo

- la funzione hash é collision-resistant
- la funzione hash é second-preimage resistant
- la funzione hash é pre-image resistant

Come comportarsi? Che fare?

Individuare e scrutinare nuove assunzioni ragionevoli sulle funzioni hash o sulla specifica funzione hash usata nella costruzioni. Puó richiedere tempo.

Schemi per cui é possibile esibire una riduzione sono spesso molto *meno efficienti* di schemi che usano funzioni hash per cui una riduzione non c'é.

Random Oracle Model

Un approccio che ha avuto un notevole successo consiste nell'usare un **modello idealizzato** per derivare una misura di confidenza nella robustezza della costruzione progettata.

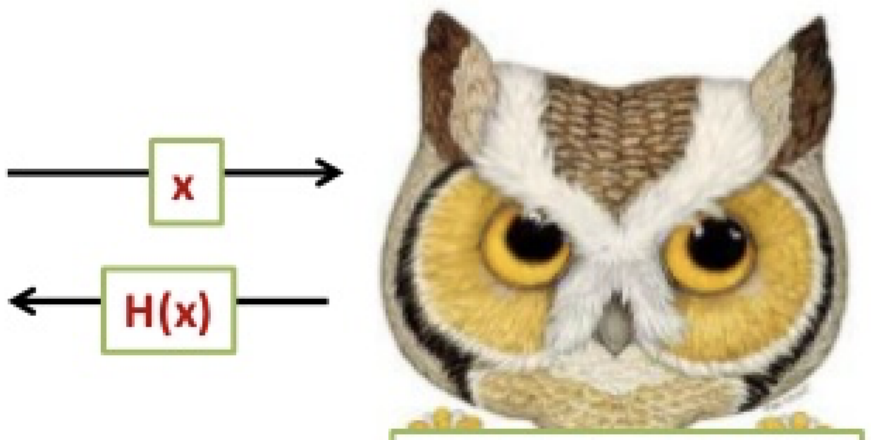
L'esempio piú significativo é il **Random Oracle Model**.

Nel modello si assume la presenza di

$$H : \{0,1\}^* \rightarrow \{0,1\}^{\ell} \qquad \text{funzione } \textbf{totalmente casuale}$$

pubblica, che puó essere valutata soltanto attraverso query ad un oracolo.

Random Oracle Model



Tutte le parti in uno schema crittografico, partecipanti onesti **e** avversario, possono interagire con il random oracle

Nota: un random oracle **non** esiste nella realtá. É una astrazione. Memorizzare una funzione casuale richiede spazio esponenziale.

Random Oracle Model

Passi della metodologia

1. Uno schema viene progettato e provato sicuro nel modello in cui é presente il random oracle
2. Per implementare lo schema nel mondo reale, **instanziamo** il random oracle con una funzione hash crittografica $\hat{H}$

Quando lo schema impone ad una delle parti di **inviare** una query all'oracolo, nell'implementazione la parte **calcola** l'output della funzione hash $\hat{H}$

Se $\hat{H}$ é **sufficientemente buona** nell'emulare il random oracle, la prova di sicurezza data nel modello **dovrebbe** estendersi allo schema reale.

Random Oracle Model

Sfortunatamente:

- non c'é nessuna giustificazione teorica per la precedente affermazione
- ci sono esempi (artificiosi...) di schemi sicuri nel random oracle model, che risultano insicuri per **qualsiasi** instanziazione concreta con $\hat{H}$
- non é chiaro cosa significhi "sufficientemente buona" nell'emulare il random oracle

$\hat{H}$ é deterministica e fissata

$\Downarrow$

non puó comportarsi come una funzione totalmente casuale

Random Oracle Model

Analizziamo la metodologia nel dettaglio

- le parti oneste e Adv possono inviare query x all'oracolo ricevendo $H(x)$
- i meccanismi interni dell'oracolo non sono noti (scatola nera)
- le query sono **private**
  - nessun altro conosce la query x fatta da una parte
  - non sa neanche che ha inviato una query
- le query vengono risposte **consistentemente**, i.e., stessa query - stessa risposta

Random Oracle Model

Ricordo che la scelta di una funzione casuale puó essere effettuata in due modi

1. *one-shot*: scegliere in un solo colpo nell'insieme di tutte le funzioni su dominio e codominio specificati
2. *on-the-fly*: scegliere uniformemente al volo le risposte alle query e memorizzarle in tabella

La seconda opzione é piú facile da gestire ed anche piú appropriata quando H é definita su un dominio infinito

Random Oracle Model

Osservazione: abbiamo giá usato funzioni casuali in precedenza ma **in modo diverso**.

Per definire funzioni pseudocasuali: in quel caso sono usate peró come **termine di paragone**

$$\Downarrow$$

cioé come un modo per definire **cosa significa** essere pseudocasuale

Nel random oracle model, la funzione random H é **parte della costruzione stessa**

$$\Downarrow$$

in una implementazione deve essere instanziata con una $\hat{H}$

Random Oracle Model

Le definizioni che abbiamo considerato fino ad ora nel cosiddetto *modello standard* (per distinguerlo dal *random oracle model*) hanno la forma:

Π é sicura se, $\forall$ A ppt, la probabilitá di un certo evento é sotto una determinata soglia, dove la probabilitá é calcolata sulle scelte casuali

- delle parti oneste
- dell'avversario A

Nel random oracle model (ROM, per brevitá), Π poggia su un oracolo:

Π é sicura se, $\forall$ A ppt, la probabilitá di un certo evento é sotto una determinata soglia, dove la probabilitá é calcolata sulle scelte casuali

- delle parti oneste
- dell'avversario A
- **dell'oracolo H**

Prove nel ROM

Le dimostrazioni nel modello ROM possono sfruttare il fatto che H viene scelta uniformemente a caso e puó essere valutata soltanto tramite query.

Tre proprietá risultano utili:

- *uniformitá:* se x non é stato ancora usato per una query ad H, allora il valore $H(x)$ é uniformemente distribuito

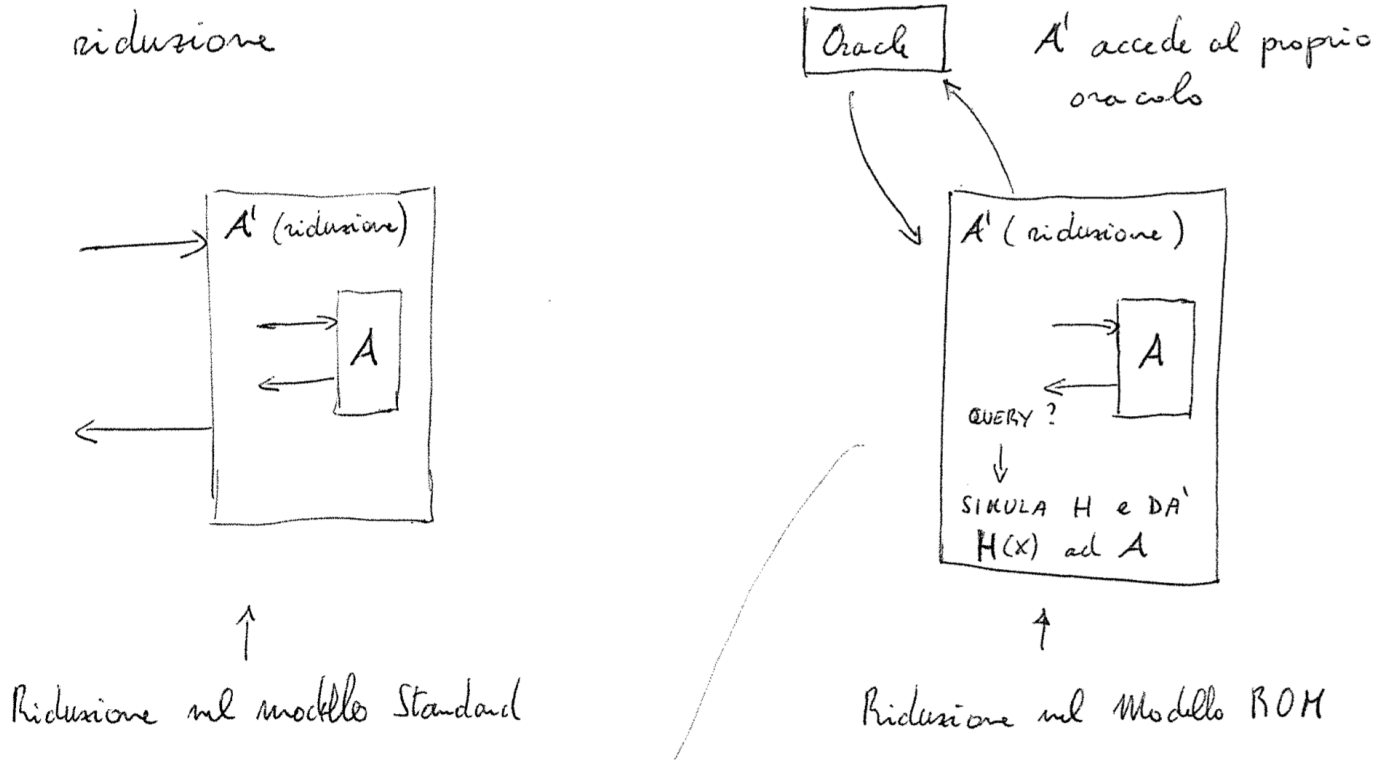
Si noti la differenza tra H ed un PRG

$G(x)$ é pseudocasuale per un osservatore ppt, assumendo che x sia scelto uniformemente e sia totalmente sconosciuto all'osservatore.

$H(x)$ é *totalmente casuale* fino a quando un osservatore non ha inviato la query x all'oracolo. **Anche** se x é noto o se x non é scelto uniformemente.

Prove nel ROM

Le altre due proprietá sono relative alle prove di sicurezza:



Prove nel ROM

La riduzione A' nel ROM deve simulare l'oracolo H con cui interagisce A, rispondendo opportunamente alle sue query.

- *estraibilitá:* se A invia la query x ad H, la riduzione A' *vede* questa query ed *acquisisce* il valore di x
- *programmabilitá:* la riduzione A' puó fissare il valore $H(x)$ ad un valore di propria scelta, posto che il valore sia uniformemente distribuito

Osservazione: quando H, in una implementazione, viene instanziata con una funzione concreta $\hat{H}$, **non** c'é nessuna controparte per le proprietá dell'estraibilitá e della programmabilitá

$$\Downarrow$$

Lo schema "concreto" diverge dallo schema nel modello e la prova di sicurezza usa possibilitá che **non** esistono nell'analisi dello schema concreto.

Uso del ROM: qualche esempio

Dispondendo di un oracolo casuale molte primitive crittografiche possono essere realizzate facilmente.

Supponiamo di disporre di

$$H : \{0,1\}^{\ell_{in}(n)} \rightarrow \{0,1\}^{\ell_{out}(n)}$$

con $\ell_{in}(n), \ell_{out}(n) > n$ (parametro di sicurezza)

Un oracolo casuale dove $\ell_{out}(n) > \ell_{in}(n)$ puó essere usato come un generatore pseudocasuale.

Infatti, $\forall$ A ppt, $\exists$ una funzione trascurabile negl tale che

$$|Pr[A^{H(\cdot)}(H(x)) = 1] - Pr[A^{H(\cdot)}(y) = 1]| \leq negl(n)$$

Uso del ROM: casi semplici

prima prob. calcolata su

- scelta uniforme di $H(\cdot)$
- scelta uniforme di $x \in \{0,1\}^{\ell_{lin}(n)}$
- scelte casuali di A

seconda prob. calcolata su

- scelta uniforme di $H(\cdot)$
- scelta uniforme di $y \in \{0,1\}^{\ell_{out}(n)}$
- scelte casuali di A

Sia $S = \{\text{punti usati da } A \text{ come query per } H\}$

$|S|$ é poly e la probabilitá che $x \leftarrow \{0,1\}^{\ell_{lin}(n)}$ appartenga ad S é $poly(n)/2^{\ell_{lin}(n)}$

Inoltre, condizionato all'evento $x \notin S$, l'input di A é in entrambi i casi uniforme ed indipendente dalle risposte alle query di A.

Uso del ROM: casi semplici

Pertanto $|Pr[A^{H(\cdot)}(H(x)) = 1] - Pr[A^{H(\cdot)}(y) = 1]|$ risulta

$$
\begin{aligned}
&= |(Pr[A^{H(\cdot)}(H(x)) = 1 \wedge x \in S] + \\
&\quad + Pr[A^{H(\cdot)}(H(x)) = 1 \wedge x \notin S]) - Pr[A^{H(\cdot)}(y) = 1]| \\
&\quad \text{(essendo la seconda e la terza probabilitá uguali)} \\
&\leq P[x \in S] \\
&= poly(n)/2^{\ell_{lin}(n)} \quad \text{(che é } negl(n)\text{)}.
\end{aligned}
$$

Uso del ROM: casi semplici

Altro esempio. Se $\ell_{in}(n) > \ell_{out}(n)$ un oracolo casuale puó essere usato come una funzione resistente a collisioni.

Per veder ció, si consideri l'esperimento che segue:

Rom-Coll

1. Viene scelta una funzione uniformemente a caso H
2. A dá in output x ed x' distinti ed ha successo se $H(x) = H(x')$

La probabilitá di successo di *qualsiasi* A é trascurabile. Infatti, assumiamo che:

- A dá in output soltanto valori x ed x' precedentemente inviati come query ad H
- A non ripete mai la stessa query

Uso del ROM: casi semplici

Siano $x_1, \ldots, x_q$ le query (numero polinomiale)

$$Pr[A \text{ ha successo }] \leq Pr[H(x_i) = H(x_j) \text{ per qualche } i \neq j]$$

$$\Downarrow$$

Prob. di prendere q stringhe y_i in $\{0,1\}^{\ell_{out}(n)}$

indip. ed uniformemente a caso tali che $y_i = y_j, \quad i \neq j$

$\Rightarrow$ A ha successo con probabilitá $O(q^2/2^{\ell_{out}(n)})$ (prob. del compleanno).

Uso del ROM: casi semplici

Ultimo esempio. Un oracolo puó essere usato per costruire una funzione pseudocasuale.

Sia $\ell_{in}(n) = 2n$ ed $\ell_{out}(n) = n$, e sia $F_k(x) \stackrel{def}{=} H(k||x)$, dove $|k| = |x| = n$.

Si consideri il seguente esperimento:

Rom-PRF

1. Vengono scelti uniformemente a caso una funzione H, un $k \in \{0,1\}^n$ ed un $b \in \{0,1\}$
2. Se $b = 0$, A riceve accesso ad un oracolo per $F_k(\cdot) = H(k||\cdot)$; altrimenti, A riceve accesso ad una funzione casuale $\overline{H}$ (indipendente da H)
3. A dá in output un bit b' ed ha successo se $b' = b$

Uso del ROM: casi semplici

Al passo 2. l'avversario A puó anche accedere all'oracolo casuale H (oltre all'oracolo $F_k(\cdot)$ o $\overline{H}(\cdot)$, a seconda se $b = 0$ o $b = 1$).

É possibile dimostrare che:

$\forall$ A ppt, $\exists$ una funzione trascurabile negl tale che la probabilitá di successo di A nell'esperimento *Rom-PRF* é al piú $1/2 + negl(n)$

Osservazione: si noti che tutti i claim precedenti continuano a valere **anche se** gli avversari sono computazionalmente **illimitati**, fino a quando possono effettuare soltanto un numero polinomiale di query.

ROM: Metodologia robusta?

Cosa garantisce una riduzione nel modello ROM?

Il dibattito é acceso nella comunitá.

Obiezioni al ROM:

- Non c'é prova o giustificazione rigorosa che $\hat{H}$ invece dell'oracolo H mantenga lo schema sicuro nel mondo reale come fa H nel ROM
- Nessun $\hat{H}$ puó comportarsi come un oracolo casuale. Un Adv acquisisce al **descrizione** di $\hat{H}$. Gli output sono determinati univocamente.
- Non sappiamo cosa significa che $\hat{H}$ é sufficientemente buona nell'emulare H
- Funzioni hash concrete, e.g., SHA2, possono andar bene in alcune applicazioni non in tutte.

ROM: Metodologia robusta?

Si noti che esiste una differenza qualitativa nelle assunzioni.

1. SHA-2 si comporta come un oracolo casuale
2. SHA-2 é collision resistant
3. AES é una pseudorandom function

Per la prima non c'é una definizione soddisfacente di cosa significhi. Per la seconda e la terza esistono delle buone definizioni di queste proprietá.

$$\Downarrow$$

l'assunzione *ROM* é qualitativamente differente da assunzioni "ben definite".

ROM: Metodologia robusta?

Motivazioni a supporto del ROM (che spiegano perché usato ampiamente)

1. Rende possibile la progettazione di protocolli spesso piú **efficienti** di quelli realizzati nel modello standard
2. Le prove nel ROM sono quasi universalmente accettate/riconosciute come elementi che accrescono la **confidenza** nella sicurezza di uno schema
3. Una prova offre garanzie sulla **soliditá** del progetto. Soltanto attacchi nel mondo reale che sfruttano le debolezze di $\hat{H}$ possono avere successo
4. Ad oggi **non** ci sono stati attacchi che hanno avuto successo nel mondo reale su schemi che sono stati provati sicuri nel ROM, quando H é stato instanziato propriamente