Sicurezza computazionale

Paolo D'Arco
pdarco@unisa.it

Università di Salerno

Elementi di Crittografia

Contenuti

Cifratura simmetrica

La segretezza perfetta ha limitazioni intrinseche

La crittografia moderna nasce dall'esigenza di superarle

Concetti chiave:

- Segretezza **computazionale**: una chiave può essere usata per cifrare molti messaggi lunghi, e.g., chiave di 128 bit per diversi GigaByte di dati
  - Più debole di quella perfetta ma ... sufficiente
- **Pseudocasualità** (pseudorandomness): cattura l'idea che un oggetto può sembrare completamente casuale anche se non lo è
  - Centrale per lo sviluppo di tutta la crittografia moderna, con implicazioni in altri campi

Cifratura simmetrica

Più in dettaglio, la segretezza perfetta garantisce

- assolutamente nessuna informazione per l'Adv che ascolta
- Adv ha potere illimitato

Da un punto di vista pratico uno schema che

- rivela informazioni con probabilità 2^{-60}
- ad Adv che possono investire 200 anni di sforzo computazionale

è ancora molto buono.

Cifratura simmetrica

Le definizioni di sicurezza che tengono di conto

- i limiti computazionali dell'Adv
- ammettono una piccola probabilità di errore

vengono dette **computazionali** (per distinguerle da quelle basate sulla Teoria dell'Informazione)

In questo caso

- la sicurezza è soltanto garantita rispetto ad Adv efficienti
- Adv possono potenzialmente avere successo ma soltanto con probabilità molto piccola.

Cifratura simmetrica

Discende che

- Le risorse richieste per "rompere" uno schema devono essere maggiori di quelle di cui Adv può disporre
- Le probabilità devono essere realmente piccole

Ci sono due approcci per sviluppare una teoria significativa

- Concreto
- Asintotico

Approccio concreto

Limita esplicitamente la probabilità di successo massima di qualsiasi Adv probabilistico che esegue per una specificata quantità di tempo

> *Uno schema è (t, ϵ)-sicuro se qualsiasi Adv che può eseguire per tempo al più t ha successo nella rottura dello schema con probabilità al più ϵ.*

Parametri reali potrebbero essere oggi: $t = 2^{80}$ cicli di CPU, $\epsilon = 2^{-60}$.

Esempio 3.1. I moderni schemi di cifratura simmetrici sono considerati quasi ottimali se, quando la chiave è lunga n bit (spazio delle chiavi 2^n elementi) qualsiasi Adv che esegue per tempo t ha successo nella rottura dello schema con probabilità al più $\frac{ct}{2^n}$, per qualche costante c fissata.

Approccio concreto

La condizione precedente corrisponde ad un attacco di forza bruta

Essenzialmente dice che non ci sono attacchi migliori della ricerca esaustiva

Assumiamo $c = 1$, lunghezza chiave 60 bit ed un PC che esegue 4×10^9 cicli per secondo. La ricerca esaustiva richiede:

$$\frac{2^{60}}{4 \times 10^9} \text{ secondi, che corrispondono a circa 9 anni}$$

Rispetto ad un computer molto più potente che può effettuare 2×10^{16} cicli per sec

$$\frac{2^{60}}{2 \times 10^{16}} \text{ secondi, che corrispondono a circa 1 minuto.}$$

Con lo stesso computer, per una chiave lunga 80 bit

$$\frac{2^{80}}{2 \times 10^{16}} \text{ secondi, che corrispondono a circa 2 anni.}$$

La lunghezza raccomandata oggi è $n = 128$ bit.

Approccio concreto

Per avere un'idea di quanto siano grandi questi numeri ...

- 2^{128} è la cardinalità dello spazio delle chiavi
- $2^{58} \approx$ il numero di secondi di vita dell'Universo dal Big Bang ad oggi

E per avere un'idea di quanto siano piccole le probabilità ...

Se un Adv ha successo in un anno di tempo con prob. 2^{-60}, è molto più probabile che Alice e Bob siano colpiti entrambi da un fulmine nello stesso periodo di tempo

Svantaggi dell'approccio concreto

Difficoltà di interpretazione di un'affermazione.

Nessun Adv che esegue per 5 *anni ha successo con prob. maggiore di* ϵ

- con quanto potere computazionale?
- tiene conto di sviluppi futuri (legge di Moore)?
- considera implementazioni generiche o ad hoc degli attacchi?
- cosa possiamo dire di un Adv che esegue per 2 anni?
- cosa di un Adv che esegue per 10 anni?

Approccio asintotico

Introduce un *parametro di sicurezza* n intero

- Parametrizza sia lo schema che le parti coinvolte: partecipanti, Adv
- Può essere pensato al momento come la lunghezza della chiave
- È noto ad Adv

I tempi di esecuzione di Adv, dei partecipanti e la prob. di successo sono funzioni di n.

- Avversari efficienti: algoritmi probabilistici di tempo polinomiale in n
- Partecipanti: algoritmi probabilistici di tempo polinomiale in n
- Probabilità di successo piccola: probabilità più piccola dell'inverso di *ogni* polinomio in n, i.e., $1/p(n)$, dette probabilità trascurabili (negligible)

Approccio asintotico

Usando l'acronimo PPT per *probabilistico di tempo polinomiale* diremo che

Uno schema è sicuro se qualsiasi Adv PPT ha successo nella rottura dello schema con al più probabilità trascurabile

Questa nozione è *asintotica* perchè la sicurezza dipende dal comportamento dello schema per valori del parametro n sufficientemente grandi.

Esempio 3.2 Disponiamo di uno schema asintoticamente sicuro. Un Adv che

- esegue per n^3 minuti (tempo polinomiale)
- ha successo con prob. $2^{40} \cdot 2^{-n}$ (prob. trascurabile. i.e., $< \frac{1}{p(n)}$)

riesce

Approccio asintotico

- per $n \leq 40$, esegue per 40^3 minuti (6 settimane), ed ha successo con prob. 1.
- per $n = 50$, esegue per 50^3 minuti (3 mesi), ed ha successo con prob. $\approx \frac{1}{1000}$.
- per $n = 500$, esegue per 500^3 minuti (200 anni), ed ha successo con prob. $\frac{1}{2^{460}}$.

Il parametro di sicurezza n è un meccanismo che permette di *calibrare* la sicurezza dello schema al livello desiderato.

Relativamente agli schemi di cifratura

- guardare al parametro di sicurezza n come alla lunghezza della chiave corrisponde essenzialmente al fatto che il tempo per una ricerca esaustiva cresce esponenzialmente nella lunghezza della chiave

Approccio asintotico

Nota: incrementando n, incrementa anche il tempo di esecuzione per le parti oneste.

Occorre trovare un giusto equilibrio tra

- tempo richiesto alle parti oneste (più piccolo possibile)
- tempo richiesto all'avversario (più grande possibile)
- probabilità di successo (più piccole possibili)

Approccio asintotico

Esempio 3.3 Computer *più veloci* giocano a favore delle parti oneste.

Consideriamo un protocollo crittografico in cui:

- le parti oneste eseguono per $10^6 \cdot n^2$ cicli di CPU
- Adv esegue per $10^8 \cdot n^4$ cicli di CPU, ed ha successo con prob. $2^{-\frac{n}{2}}$.

Per computer a 2GHz, ed $n = 80$

- le parti oneste eseguono per $10^6 \cdot 80^2$ cicli di CPU, $3,2$ secondi
- Adv esegue per $10^8 \cdot 80^4$ cicli di CPU, circa 3 settimane, ed ha successo con prob. 2^{-40}.

Per computer a 8GHz, ed $n = 160$ (incrementando la lunghezza della chiave)

- le parti oneste eseguono per $10^6 \cdot 160^2$ cicli di CPU, ancora $3,2$ secondi
- Adv esegue per $10^8 \cdot 160^4$ cicli di CPU, *più di* 13 *settimane*, ed ha successo *con prob.* 2^{-80}.

Approccio asintotico

Pertanto, computer più veloci $\Rightarrow$ lavoro di Adv più difficile.

Nota: anche quando si usa l'approccio asintotico, garanzie di sicurezza concrete sono necessarie in pratica. Come gli esempi mostrano, solitamente una valutazione asintotica può essere convertita in limitazioni di sicurezza concrete, per ogni valore di n.

Approccio asintotico in dettaglio

Algoritmi efficienti.

A esegue in tempo polinomiale se esiste un polinomio $p(\cdot)$ tale che, per ogni $x \in \{0,1\}^*$, $A(x)$ termina in al più $p(|x|)$ passi, dove $|x|$ indica la lunghezza dell'input.

Come anticipato, progetteremo schemi in funzione del parametro di sicurezza n

- forniremo, pertanto, il parametro n scritto in unario, i.e., 1^n (una stringa di n valori 1) come input agli algoritmi esplicitamente alcune volte

Le parti possono avere altri input ovviamente

- il tempo di esecuzione degli algoritmi corrispondenti deve essere polinomiale *nella lunghezza totale* dei loro input

Assumiamo per default tutti gli algoritmi come probabilistici: hanno accesso ad un nastro di random bit di lunghezza sufficiente. Dà più potere agli avversari.

Approccio asintotico in dettaglio

Probabilità di successo trascurabili.

Definizione 3.4. Una funzione $f : N \to R^+$ è trascurabile (negligible) se, per ogni polinomio positivo p, esiste un n_0 tale che, per tutti gli $n > n_0$, risulta $f(n) < \frac{1}{p(n)}$.

$$\approx$$

Per ogni costante c, esiste un n_0 tale che, per tutti gli $n > n_0$, risulta $f(n) < \frac{1}{n^c}$.
Denoteremo una generica funzione trascurabile con *negl.*

Esempio 3.5. Le funzioni

$$2^{-n} \qquad 2^{-\sqrt{n}} \qquad \text{ed} \qquad n^{-\log n}$$

sono tutte funzioni trascurabili.

Approccio asintotico in dettaglio

Tuttavia, esse tendono a zero in modi diversi.

Consideriamo, per esempio, il minimo valore di n per cui ogni funzione è $< 1/n^5$

- $2^{-n} < n^{-5}$ se e solo se $n > 5 \log n$, che vale da $n = 23$
- $2^{-\sqrt{n}} < n^{-5}$ se e solo se $n > 25 \log^2 n$, che vale da $n \approx 3500$
- $n^{-\log n} < n^{-5}$ se e solo se $\log n > 5$, che vale da $n = 33$

Attenzione: sembrerebbe che $2^{-\sqrt{n}}$ tende a zero più lentamente di $n^{-\log n}$, ma non è così. Per tutti gli $n > 65536$ risulta $2^{-\sqrt{n}} < n^{-\log n}$. Tuttavia, per valori di n più piccoli, una prob. di successo per Adv pari a $n^{-\log n}$ è preferibile ad una pari a $2^{-\sqrt{n}}$.

Lavorare con funzioni trascurabili è tecnicamente vantaggioso perchè soddisfano utili proprietà di chiusura.

Approccio asintotico in dettaglio

Proposizione 3.6. Siano $negl_1(n)$ e $negl_2(n)$ funzioni trascurabili

1. la funzione $negl_3(n) = negl_1(n) + negl_2(n)$ è trascurabile
2. per qualsiasi polinomio p, la funzione $negl_4(n) = p(n) \cdot negl_1(n)$

Osservazioni:

- un evento a prob. trascurabile *rimane* trascurabile anche se l'esperimento che potrebbe generarlo viene ripetuto $p(n)$ volte
- d'altra parte, se una funzione g *non* è trascurabile, allora la funzione $f(n) \stackrel{def}{=} \frac{g(n)}{p(n)}$ *non* è trascurabile, per ogni polinomio positivo.

Sicurezza asintotica in sintesi

Ogni definizione di sicurezza consiste di due parti:

- specifica di cosa è una rottura dello schema
- quale potere ha Adv

Il framework generale per le definizioni è il seguente

> *Uno schema è sicuro se, per ogni Adv PPT A che sferra un attacco di qualche tipo formalmente specificato, e per ogni polinomio positivo p, esiste un intero n_0 tale che, quando $n > n_0$, la probabilità che A abbia successo è minore di* $1/p(n)$.

Nota: nulla è garantito per valori di $n \leq n_0$.

Ragioni alla base della scelta asintotica

Arbitrarie ma conformi alle scelte in teoria della complessità

Algoritmi PPT:

- usuale per rappresentare computazioni efficienti
- indipendenza dal modello computazionale prescelto
  - Tesi di Church-Turing estesa: tutti i modelli *ragionevoli* sono polinomialmente equivalenti
- proprietà di composizione dei polinomi
  - A invoca subroutines. Se A è polinomiale e le subroutine sono polinomiali $\Rightarrow$ l'intero algoritmo è polinomiale

Probabilità trascurabili:

- proprietà di chiusura evidenziate in precedenza

Necessità del rilassamento

Segretezza computazionale vs segretezza perfetta

- rispetto ad Adv efficienti, con piccole probabilità di successo

Entrambe sono essenziali. Per capirlo, supponiamo di avere uno schema di cifratura in cui $|K| < |M|$

- Dato un cifrato c, Adv può decifrare c usando tutte le chiavi $k \in K$, ottenendo una lista L di messaggi. $L \not\supseteq M$. Pertanto, rivela informazioni sul messaggio m a cui c corrisponde
- Similmente, se Adv possedesse coppie

$$(m_1, c_1), (m_2, c_2), \ldots, (m_\ell, c_\ell)$$

potrebbe tentare di decifrare le coppie fino a trovare k ed usarla successivamente per decifrare un nuovo c.

Necessità del rilassamento

Ricerche esaustive come le precedenti permettono ad Adv di aver successo con probabilità 1 in tempo lineare in $|K|$

$$\Downarrow$$

Dobbiamo escludere tali ricerche, limitando il tempo di esecuzione di Adv

D'altro canto

- se Adv possedesse ancora le coppie

$$(m_1, c_1), (m_2, c_2), \ldots, (m_\ell, c_\ell)$$

potrebbe *scegliere a caso* una chiave e usare le coppie per verificare la correttezza della scelta. Adv esegue in tempo costante e ha successo con probabilità $1/|K|$

$$\Downarrow$$

Dobbiamo ammettere piccole probabilità di successo per escludere questi attacchi