Zero-knowledge 101

Gabriele Vanoni

Sommario

Le dimostrazioni interattive permettono la costruzione di protocolli per convincere efficientemente un altro agente della validità di un certo enunciato. Se la dimostrazione non comunica altro che la validità dell'asserzione allora diciamo che è a conoscenza zero. Questo strumento risulta applicabile a una vasta classe di problemi e di grande utilità in crittografia. Infatti nei protocolli di sicurezza spesso le parti coinvolte non possono fidarsi l'una dell'altra e non vogliono quindi rivelare i segreti di cui sono in possesso, ma solo convincere la controparte di esserne a conoscenza.

Indice

1. Introduzione **2**

2. Dalle dimostrazioni interattive alle dimostrazioni a conoscenza zero **2**

3. L'uso nelle applicazioni **4**

Appendice A. Schemi di commitment **7**

1. Introduzione

In questo breve report tracceremo un quadro sulle dimostrazioni a conoscenza zero. È un argomento ormai classico nell'ambito della crittografia sebbene sia considerato avanzato. Proveremo a trattare gli argomenti in maniera il più possibile chiara ed autocontenuta, rimandando a [Gol02] per una trattazione più formale e completa. Gli unici prerequisiti richiesti sono la conoscenza base della teoria della complessità computazionale e della probabilità.

Nella Sezione 2 caratterizzeremo le dimostrazioni a conoscenza zero da un punto di vista computazionale, mentre nella Sezione 3 ci si concentrerà su alcune semplici applicazioni. Infine è presente un'appendice contenente le principali definizioni riguardanti gli schemi di commitment. [Sma16] è stata una fonte di ispirazione per la stesura dell'intero lavoro.

2. Dalle dimostrazioni interattive alle dimostrazioni a conoscenza zero

Per introdurre la nozione di *dimostrazione a conoscenza zero*, Goldwasser, Micali e Rackoff in [GMR85] hanno prima dovuto introdurre quella di *dimostrazione interattiva.* Presentiamo il concetto prima con un esempio (mutuato da [Sip12]), passando successivamente alle definizioni formali.

Esempio 1. Supponiamo che un agente **P** con capacità computazionali illimitate chiamato *prover* voglia convincere un altro agente **V** con capacità computazionali limitate chiamato *verifier* che due grafi G_1 e G_2 non sono isomorfi. **V** non può fidarsi della parola di **P**, per cui chiede a **P** di giocare al seguente gioco.

1. **V** sceglie con probabilità uniforme un grafo tra G_1 e G_2 senza comunicarlo a **P**.
2. **V** operando una permutazione casuale dei nomi dei vertici genera un grafo H isomorfo a quello selezionato.
3. **V** spedisce H a **P**.
4. **P** comunica a **V** se H è stato generato da G_1 o G_2.

Risulta chiaro che se G_1 e G_2 sono isomorfi allora H sarà isomorfo ad entrambi, indipendentemente dalla scelta casuale operata da **V**. In questo caso **P** sta mentendo e sarà costretto a comunicare a **V** un grafo tra G_1 e G_2 estratto con probabilità uniforme. La probabilità di azzeccare è dunque $\frac{1}{2}$ e se il gioco viene ripetuto n volte **P** avrà probabilità $\left(\frac{1}{2}\right)^n$ di azzeccare tutte le risposte. Se invece G_1 e G_2 non sono isomorfi, **P** ha modo di trovare la soluzione corretta con probabilità 1. In questo modo **V** ha la certezza, a meno di una probabilità che tende a zero per n che tende all'infinito, che **P** non stia mentendo.

È interessante notare che il problema NonIso per cui abbiamo presentato una dimostrazione interattiva appartiene alla classe di complessità co-NP, e che in particolare non è noto se appartenga anche a P o a NP.

Definizione 2 ([Wig17])**.** Un sistema di prove interattivo per un insieme S è un gioco con due giocatori, **V** che esegue una strategia polinomiale probabilistica e **P** che esegue una strategia computazionalmente illimitata tale che le seguenti proprietà siano verificate:

- **Completezza**: dopo aver interagito con **P** sull'input comune x, **V** accetta sempre se $x \in S$.
- **Correttezza**: dopo aver interagito con **P** sull'input comune x, **V** rifiuta con probabilità almeno $\frac{1}{2}$ se $x \notin S$.

Chiamiamo IP (*interactive polynomial time*) la classe contenente tutti gli insiemi S per cui è possibile costruire un sistema di dimostrazioni interattivo.

Chiaramente $\mathsf{NP} \subseteq \mathsf{IP}$, dal momento che per ogni $S \in \mathsf{NP}$, **P** può fornire a **V** il certificato che prova l'appartenenza di x a S, che deve esistere per definizione di NP. È possibile però dare una caratterizzazione più precisa della classe IP attraverso il seguente

Teorema 3 ([Sha92])**.** IP=PSPACE.

Intuitivamente questo significa che l'interazione permette di risolvere efficientemente più problemi di quanti è possibile risolvere in assenza di essa. Si pensi a un docente che deve spiegare una dimostrazione. Certamente è più efficiente se lo fa in classe quando gli studenti possono intervenire per fare domande rispetto a lasciarla da fare a casa su delle dispense. In questo caso infatti è come se dovesse anticipare tutte le possibili domande, mentre in classe risponde solo a quelle che gli vengono poste.

Tutti i problemi rilevanti in crittografia possono dunque essere formulati attraverso dimostrazioni interattive. Per applicazioni crittografiche tuttavia vorremmo che la dimostrazione interattiva non fornisse altro che la validità dell'enunciato stesso. Infatti **P** vorrebbe poter convincere **V** di essere in possesso di un segreto ma senza rivelarglielo, non potendo in generale fidarsi. Se consideriamo ad esempio un'istanza x di Sat vorremmo che **P** potesse convincere **V** che x è soddisfacibile senza far conoscere l'assegnamento che rende x vera. Più formalmente,

Definizione 4 (Zero-knowledge)**.** Una dimostrazione interattiva è (computazionalmente) a conoscenza zero se l'insieme $\mathcal{V}$ delle trascrizioni delle esecuzioni del protocollo è (computazionalmente) indistinguibile dall'insieme $\mathcal{S}$ delle possibili simulazioni di tale protocollo.

Chiamiamo CZK la classe contenente tutti gli insiemi S per cui è possibile costruire un sistema di dimostrazioni interattivo computazionalmente a conoscenza zero.

In questo modo cioè viene assicurato che tutta l'informazione che **V** potrebbe estrarre interagendo con **P** tramite il protocollo, potrebbe essere estratta anche in assenza di interazione.

Ci chiediamo quale sia l'insieme dei problemi per cui è possibile fornire dimostrazioni a conoscenza zero. Attraverso un esempio mostriamo che nell'ipotesi di esistenza di funzioni *one-way* è possibile costruire dimostrazioni a conoscenza zero per ogni problema in NP.

Esempio 5. Consideriamo il problema 3-COLORING, ovvero se sia possibile assegnare a un grafo $G = (V, E)$ una colorazione dei vertici $\psi : V \to \{1, 2, 3\}$ tale che se $\{v_i, v_j\} \in E$ allora $\psi(v_i) \neq \psi(v_j)$.

Vogliamo che **P**, in possesso di ψ, dimostri a **V** che un grafo G è colorabile con tre colori senza rivelare ψ, dal momento che la dimostrazione deve essere a conoscenza zero. Consideriamo allora il seguente protocollo:

1. **P** seleziona uno schema di commitment $C(x, r)$ e una permutazione casuale π di $\{1, 2, 3\}$.
2. **P** manda a **V** i commitment $c_i = C(\pi(\psi(v_i)), r_i)$ per ogni $v_i \in V$.
3. **V** seleziona un lato $\{v_i, v_j\} \in E$ e lo manda a **P**.
4. **P** rivela a **V** $\pi(\psi(v_i))$ e $\pi(\psi(v_j))$.
5. **V** verifica che $\pi(\psi(v_i)) \neq \pi(\psi(v_j))$.

Perché il protocollo sia valido occorre mostrare che sia completo, corretto e a conoscenza zero.

- **Completezza.** Risulta chiaro che se **P** fornisce una colorazione valida ed esegue correttamente il protocollo di commitment allora **V** accetta con probabilità uno.
- **Correttezza.** Se **P** sta mentendo (cioè se G non è colorabile con tre colori), allora c'è almeno un lato $\{v_i, v_j\}$ di G tale che $\psi(v_i) = \psi(v_j)$. **V** allora ha probabilità almeno $\frac{1}{|E|}$ di selezionare un lato colorato in maniera scorretta e quindi probabilità al più $1 - \frac{1}{|E|}$ di accettare. Se il protocollo viene ripetuto n volte questa probabilità può essere fatta diventare piccola a piacere, per cui la probabilità di rifiuto diventa $1 - \left(1 - \frac{1}{|E|}\right)^n$ che tende a uno per $n \to \infty$.
- **Conoscenza zero.** Se lo schema di commitment è computazionalmente occultante allora l'ovvia simulazione e il protocollo effettivo saranno computazionalmente indistinguibili.

Considerando che 3-COLORING è NP-completo è immediato affermare il seguente

Teorema 6 ([WMG86])**.** *Sotto l'ipotesi di esistenza di funzioni one-way* $\mathsf{NP} \subseteq \mathsf{CZK}$.

La caratterizzazione completa di CZK è invece fornita dal seguente

Teorema 7 ([BOGG$^+$88])**.** *Sotto l'ipotesi di esistenza di funzioni one-way* $\mathsf{CZK} = \mathsf{IP}$.

È possibile dunque assumendo ipotesi standard in crittografia fornire dimostrazioni a conoscenza zero per ogni problema appartenente alla classe PSPACE.

3. L'uso nelle applicazioni

Presentiamo ora alcune applicazioni dei concetti che abbiamo esposto nella sezione precedente. In particolare introdurremo, all'inizio tramite un esempio, i *protocolli sigma*. Questi permettono di generare semplicemente schemi di autenticazione e di firma al costo

di un'ipotesi in più, l'onestà del verificatore (assumiamo cioè che **V** esegua correttamente il protocollo). I protocolli sigma sono composti da quattro fasi: *commitment*, *challenge*, *response*, *verification*.

Esempio 8 (Protocollo di identificazione di Schnorr)**.** L'idea è che per identificarsi un utente debba dimostrare di conoscere un segreto. Tuttavia non potendoci fidare del verificatore, non vogliamo che il segreto venga svelato all'atto dell'autenticazione. In questo caso il segreto posseduto da **P** è il logaritmo discreto x di y in base g in $\mathbb{Z}_q$, dove q è un primo grande. Il protocollo è il seguente.

1. commitment: **P** manda a **V** $r = g^k$, con k estratto uniformemente da $\mathbb{Z}_q$.
2. challenge: **V** manda a **P** e estratto uniformemente da $\mathbb{Z}_q$.
3. response: **P** manda a **V** $s = k + x \cdot e \pmod q$.
4. verification: **V** verifica che $r = g^s \cdot y^{-e}$.

Perché il protocollo sia valido occorre mostrare che sia completo, corretto e a conoscenza zero.

- **Completezza.** Se effettivamente $x = \log_g y$ allora $g^s \cdot y^{-e} = g^k \cdot (g^x)^e \cdot y^{-e} = r \cdot y^e \cdot y^{-e} = r$. **V** dunque accetta certamente se **P** esegue correttamente il protocollo.
- **Correttezza.** Se **P** non conosce x ha probabilità $\frac{1}{q}$ di riuscire a convincere **V**, cioè la probabilità di estrarre uniformemente x da $[1, q-1]$.
- **Conoscenza zero.** V può simulare esecuzioni corrette del protocollo estrendo uniformemente e ed s, e calcolando $r = g^s \cdot y^{-e}$.

Generalizzando possiamo dire che assumendo di avere un'istanza x di un linguaggio $\mathcal{L}$ in NP vogliamo che **P** dimostri a **V** di conoscere il certificato c che provi che $x \in \mathcal{L}$. Si chiede quindi qualcosa in più rispetto alle tradizionali dimostrazioni di appartenenza/esistenza. **V** non vuole solo essere convinto del fatto che x appartenga ad $\mathcal{L}$, vuole essere sicuro che **P** conosca il testimone dell'appartenenza. Chiamiamo questo fatto *dimostrazione di conoscenza*. Se la dimostrazione è a conoscenza zero, allora abbiamo una *dimostrazione di conoscenza a conoscenza zero*. Formalmente,

Definizione 9 (Zero-knowledge proof of knowledge)**.** Un protocollo è detto *zero-knowledge proof of knowledge* se è a conoscenza zero ed esite un estrattore E, cioè un algoritmo che può usare **P** per ottenere il certificato di appartenenza c.

Possiamo mostrare che il protocollo di identificazione di Schnorr (Esempio 8) è una dimostrazione di conoscenza a conoscenza zero dal momento che verifica una proprietà chiamata *correttezza speciale*, ovvero

Proposizione 10. *Date due trascrizioni del protocollo* (r, e, s) *e* (r, e', s') *possiamo estrarre il testimone* x*.*

Dimostrazione. $r = g^s \cdot y^{-e} = g^{s'} \cdot y^{-e'} = r = g^s \cdot g^{x \cdot (-e)} = g^{s'} \cdot g^{x \cdot (-e')}$. Da questa $s + x \cdot (-e) = s' + x \cdot (-e') \pmod q$. Dunque $x = \frac{s'-s}{e'-e}$. □

La dimostrazione fornisce l'estrattore E.

Riferimenti bibliografici

[BOGG+88] Michael Ben-Or, Oded Goldreich, Shafi Goldwasser, Johan Håstad, Joe Kilian, Silvio Micali, and Phillip Rogaway. Everything Provable is Provable in Zero-Knowledge. In *Advances in Cryptology — CRYPTO' 88*, Lecture Notes in Computer Science, pages 37–56. Springer, New York, NY, August 1988.

[GMR85] S Goldwasser, S Micali, and C Rackoff. The Knowledge Complexity of Interactive Proof-systems. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, STOC '85, pages 291–304, New York, NY, USA, 1985. ACM.

[Gol02] Oded Goldreich. Zero-Knowledge twenty years after its invention. Technical report, Electronic Colloquium on Computational Complexity (http://www.eccc.uni-trier.de/eccc/), Report No, 2002.

[Sha92] Adi Shamir. IP = PSPACE. *J. ACM*, 39(4):869–877, October 1992.

[Sip12] Michael Sipser. *Introduction to the Theory of Computation*. Cengage Learning, Boston, MA, third edition, June 2012.

[Sma16] Nigel P. Smart. *Cryptography Made Simple*. Springer, 2016.

[Wig17] Avi Wigderson. *Mathematics and Computation*. Draft, 2017.

[WMG86] Avi Wigderson, Silvio Micali, and Oded Goldreich. Proofs that yield nothing but their validity and a methodology of cryptographic protocol design. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 174–187, Los Alamitos, CA, USA, 1986. IEEE Computer Society.

Appendice A Schemi di commitment

Intuitivamente gli *schemi di commitment* vengono introdotti per risolvere il seguente problema. Vogliamo secretare un messaggio, affidarlo a qualcuno che lo conservi senza poterlo leggere e successivamente svelarlo assicurando che questo non sia stato modificato. Possiamo pensare ad un agente **A** che chiude un messaggio in una cassaforte che poi affida a **B**, che però non possiede la chiave. Passato del tempo **A** fornisce la chiave a **B** che così potrà leggere il messaggio essendo certo che il contenuto sia quello scritto in primo luogo da **A**, dal momento che nel mentre ha mantenuto la cassaforte al sicuro. Questo tipo di protocollo è adatto per una comunicazione tra due agenti che non si fidano l'uno dell'altro. **A** infatti teme che **B** possa accedere all'informazione prima del dovuto, mentre **B** teme che **A** possa modificarla dopo essersi già impegnato.

Esempio 11. Usando questa idea è possibile giocare a `carta-sasso-forbice` per posta. Il protocollo è il seguente:

1. **A** chiude la propria mossa nella cassaforte e la spedisce a **B**.
2. **B** spedisce la propria mossa ad **A**.
3. **A** spedisce la chiave della cassaforte a **B**.
4. Entrambi conoscono le mosse degli avversari e possono stabilire il vincitore.

Notiamo che **B** non conosce la mossa di **A** fino a che non gli viene spedita la chiave e che **A** non può cambiare la propria mossa dopo la scelta di **B**, perché la sua decisione è già nelle mani di **B**, anche se inizialmente egli non può accedervi.

Formalizziamo ora la nozione di schema di commitment e le due proprietà che ne garantiscono la sicurezza, ovvero l'occultamento e il fatto che sia vincolante.

Definizione 12. Chiamiano schema di commitment un algoritmo pubblico $C(x, r)$ dove x è il messaggio e r un valore random. Se **M** è il mittente e **R** il ricevente il commitment consiste in **M** che calcola e spedisce $c = C(x, r)$ a **R** mentre il decommitment in **M** che svela i valori di x' ed r' e **R** che controlla che $C(x', r') = c$.

Definizione 13 (Binding)**.** Uno schema di commitment $C(x, r)$ è detto (computazionalmente) vincolante se nessun avversario (computazionalmente limitato) noti x ed r può generare $x' \neq x$ ed r' tali che $C(x', r') = C(x, r)$.

Definizione 14 (Concealing o Hiding)**.** Uno schema di commitment $C(x, r)$ è detto (computazionalmente) occultante se nessun avversario (computazionalmente limitato) può indovinare b dato $c = C(x_b, r)$ con b estratto uniformemente da $\{0, 1\}$ ed essendo a lui noti x_0 e x_1.

Esempio 15. Un modo semplice per costruire uno schema di commitment computazionalmente sicuro è ricorrere a una funzione di hash crittografico H, ovvero considerando $C(x, r) = H(x||r)$. In particolare lo schema è vincolante in virtù della resistenza alla seconda preimmagine e occultante grazie alla resistenza alla preimmagine.