The Clean Paper

WHAT THE PAPER SAYS · WHAT IT DOESN'T · WHY IT MATTERS

Una prova crittografica può nascondere il suo segreto rendendo difficile provare l'assenza del simulatore

Le prove a conoscenza zero permettono di provare un enunciato senza rivelare il testimone. La teoria classica dice che non puoi avere questo, in senso pieno, con un solo messaggio, senza setup fidato e con solidità perfetta. Il paper di Rahul Ilango non fa sparire quell'impossibilità. Cambia bersaglio: invece di richiedere che un simulatore esista davvero, chiede che nessun sistema di prova scelto possa provare efficientemente che il simulatore non esiste. Sotto grandi assunzioni di complessità delle prove e crittografia, questo basta a recuperare le conseguenze falsificabili, basate su giochi, della proprietà zero-knowledge, una proprietà alla volta. Il punto non è una primitiva internet pronta all'uso. È un modo proof-theoretic di trasformare l'improvabilità matematica in copertura crittografica.

Written by Cinzia Vaglio · Cross-checked by Vera Rubin · Edited by Michele Renda · Figures by Nova Vela · AI-assisted, human-reviewed

Paper: *Gödel in Cryptography: Effectively Zero-Knowledge Proofs for NP with No Interaction, No Setup, and Perfect Soundness*, Rahul Ilango, FOCS 2025 / IACR ePrint 2025/1296 · DOI: 10.1109/FOCS63196.2025.00058

Il trucco non è provare che il segreto è nascosto

Partiamo dalla versione più semplice della conoscenza zero.

Alice vuole convincere Bob che un Sudoku ha una soluzione. Se gli manda la soluzione, Bob è convinto, ma il puzzle è rovinato. Quello che vuole è più strano: una prova che una soluzione esiste, senza rivelare la soluzione.

Questa è la promessa di una prova zero-knowledge. Il **dimostratore** (*prover*), Alice, convince il **verificatore** (*verifier*), Bob, che un enunciato è vero senza rivelare altro che la sua verità.

Il problema è che questa promessa costa qualcosa. Una prova matematica ordinaria ha due caratteristiche comode. È **un messaggio**: la scrivi, la consegni e te ne vai. Ed è **perfettamente solida**: un enunciato falso non ha alcuna prova valida. I risultati classici di impossibilità dicono che lo zero-knowledge deve rinunciare a entrambe le cose, e non solo alla coppia: ciascuna è vietata anche da sola.

Primo, una prova zero-knowledge ha bisogno di conversazione. Se Alice manda un solo messaggio, senza una configurazione fidata preparata in anticipo, la garanzia zero-knowledge collassa, qualunque solidità tu sia disposto a sacrificare in cambio.

Secondo, una prova zero-knowledge ha bisogno di una piccola tolleranza all'errore. Chiedere solidità perfetta finisce per distruggere in silenzio anche l'interazione: un verificatore che non può mai essere ingannato, qualunque scelta casuale faccia, potrebbe fissare quelle scelte in anticipo; e quando il verificatore è prevedibile, Alice può rispondere a tutto con un solo messaggio, cioè il caso che si era già rotto.

Il lavoro di Rahul Ilango riguarda un modo per aggirare quel doppio muro. Non fingendo che il muro non esista, e non producendo zero-knowledge classico nel contesto che i risultati classici rendono impossibile. La mossa è più sottile: indebolire ciò che significa "non rivela nulla", ma indebolirlo in un modo che preserva le proprietà di sicurezza che i crittografi possono davvero testare.

Il risultato si chiama **zero-knowledge effettivo** (*effectively zero-knowledge*).

Una prova senza rivelare il testimone

Il paper conserva la forma della prova ordinaria indebolendo che cosa deve provare la privacy.

La conoscenza zero è bloccata a tre porte: interazione, configurazione fidata e solidità imperfetta. La costruzione di Ilango passa da una porta diversa: il sistema formale scelto non può dimostrare efficientemente che un simulatore sia impossibile.

Privacy classica vs privacy effettiva

La rilassazione e il punto: il paper cambia cosa deve stabilire il claim di privacy.

Lo zero-knowledge classico chiede se esista un simulatore; lo **zero-knowledge effettivo** chiede soltanto se il

sistema formale scelto possa provare efficientemente che un simulatore non esiste. È questa domanda più debole a permettere alla costruzione di mantenere un solo messaggio, nessuna configurazione iniziale e solidità perfetta.

Il vecchio test: esiste un simulatore

Il modo classico di formalizzare lo zero-knowledge usa un aiutante fittizio chiamato **simulatore**.

L'idea è questa: immagina Jane, che **non** conosce il segreto di Alice. Se Jane può generare, da sola, prove che sembrano proprio le prove che Bob avrebbe ricevuto da Alice, allora le prove di Alice non hanno insegnato a Bob niente di nuovo. Jane poteva già fingere quell'esperienza senza il segreto di Alice.

Lo zero-knowledge classico chiede quindi un simulatore reale. Deve esistere un algoritmo efficiente che produca prove dall'aspetto autentico senza conoscere il segreto, il **testimone** (*witness*) nel gergo della teoria della complessità; per il Sudoku, il testimone è semplicemente la griglia risolta.

Questa definizione è potente, ma è anche esattamente dove mordono le vecchie impossibilità. L'intuizione è questa. Una prova davvero non interattiva è solo una stringa. Una volta che Bob ha quella stringa, può mostrarla ad altri: ha acquisito la capacità di provare l'enunciato ad altri, cosa che suona già come più di "nulla". I teoremi classici rendono quell'intuizione precisa nelle impossibilità sopra.

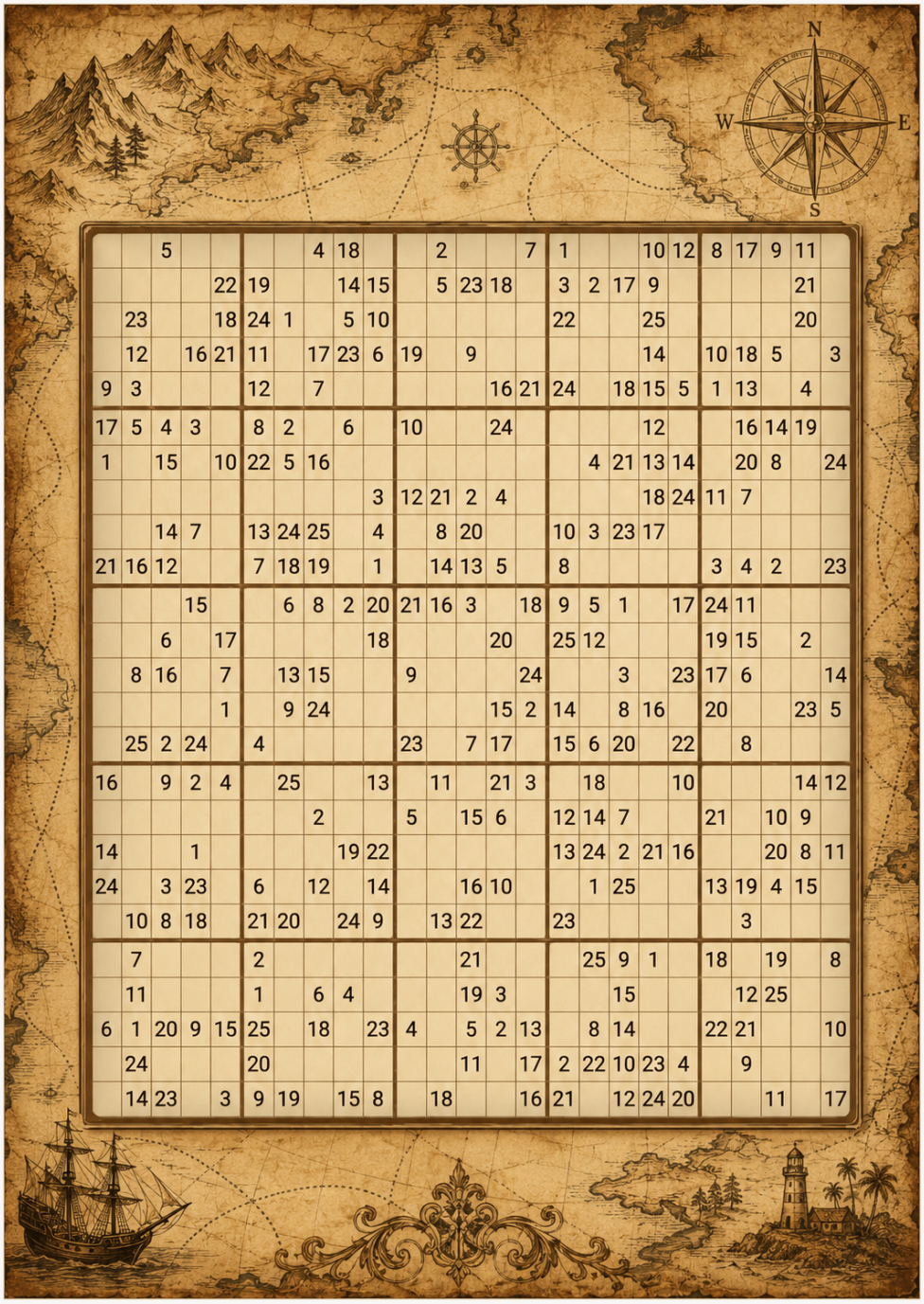
> **Le tre proprietà su cui questo lavoro insiste**
>
> Il titolo del lavoro nomina tre vincoli:
> **Nessuna interazione:** Alice manda una stringa di prova. Non c'è protocollo avanti e indietro.
> **Nessuna configurazione iniziale fidata:** Alice e Bob non si affidano a una stringa comune di riferimento o ad altra casualità pubblica predisposta in anticipo. Molti sistemi chiamati "zero-knowledge non interattivo" richiedono comunque una fase di configurazione; qui, invece, non ce n'è alcuna.
> **Solidità perfetta:** un enunciato falso non ha prova valida. Non "quasi mai accettato"; non esiste alcuna prova valida.
> Queste tre proprietà sono esattamente quelle della matematica scritta ordinaria e, come spiegato sopra, lo zero-knowledge classico non può mantenerle.

Una versione mega-Sudoku della differenza

Ecco un modo volutamente semplificato per sentire la differenza.

Non usare un Sudoku ordinario 9 per 9 per la parte seria dell'analogia. È troppo piccolo e troppo finito:

un computer può semplicemente risolverlo o provare che non ha soluzione. Immagina invece una famiglia di puzzle **MegaSudoku(n)**. Scala la regola solita: scegli una dimensione di blocco *n*, lascia *N* = *n^2*, e costruisci una griglia *N* per *N* divisa in blocchi *n* per *n*, con *N* simboli. Il Sudoku ordinario è solo il caso minuscolo *n* = *3*, *N* = *9*: una griglia 9 per 9, blocchi 3 per 3 e nove simboli. La storia di complessità delle prove inizia solo quando *n* può crescere, e quando la griglia può portare gadget extra che la fanno comportare come una formula SAT vestita da Sudoku. Una formula SAT è solo una lista di vincoli sì/no: puoi assegnare valori vero/falso alle variabili in modo che ogni vincolo sia soddisfatto?



Un Sudoku 25x25: le sue regole possono essere controllate senza rivelare la griglia finita, un sostituto visivo per una prova che verifica l'esistenza di una soluzione nascosta, cioè il testimone.

Sudoku e SAT: lo stesso puzzle in due costumi

L'affermazione che un Sudoku possa "comportarsi come una formula SAT" non è una metafora. La traduzione funziona in entrambe le direzioni, e la direzione facile può essere scritta per intero.

Da Sudoku a SAT. SAT parla solo vero/falso, quindi dagli una variabile booleana per ogni tripla (riga, colonna, valore): *x(r,c,v)* significa "la cella in riga *r*, colonna *c* contiene il valore *v*." Un Sudoku 4 per 4 (blocchi 2 per 2, valori 1-4) richiede 4·4·4 = 64 variabili; il classico 9 per 9 ne richiede 729. Ogni regola del Sudoku diventa poi un gruppo di clausole. Una clausola è un OR di variabili o delle loro negazioni; l'intera formula è l'AND di tutte le clausole.

Ogni cella tiene almeno un valore, ogni cella tiene al massimo un valore, ogni riga contiene ogni valore, e lo stesso vale per colonne e blocchi. Gli indizi stampati sono la parte più semplice: un 3 stampato nell'angolo in alto a sinistra diventa la clausola singola *x(1,1,3)*. L'AND di tutto questo è soddisfacibile esattamente quando il Sudoku ha una soluzione, e un'assegnazione soddisfacente *è* la soluzione: leggi quali *x(r,c,v)* sono vere e riempi la griglia. Per un 9 per 9 si arriva a 729 variabili e qualche migliaio di clausole, che un SAT solver moderno risolve in millisecondi.

Da SAT a Sudoku. Il lavoro ha bisogno della direzione opposta, più difficile: data una formula SAT *arbitraria*, costruire un mega-Sudoku che ha una soluzione esattamente quando la formula la ha. Le regole native del Sudoku possono dire solo "queste celle sono tutte diverse", quindi i vincoli logici arbitrari devono essere *costruiti*, e questo è precisamente il ruolo dei gadget. Un gadget è un piccolo gruppo prefabbricato di celle, uno per clausola della formula, in cui celle designate giocano il ruolo di variabili e i vincoli interni sono ingegnerizzati così che i soli riempimenti legali corrispondano ad assegnazioni che soddisfano quella clausola. È artigianato standard delle prove di NP-completezza; per il Sudoku generalizzato è stato realizzato da Yato e Seta nel 2003.

Insieme, le due direzioni dicono che Sudoku N per N e SAT sono lo stesso problema con due costumi diversi. È questa equivalenza che permette all'articolo di raccontare una storia su tutta NP usando griglie e simboli.

Il testimone resta facile da immaginare. Alice conosce un riempimento completo valido del mega-Sudoku. Bob vuole essere convinto che un tale riempimento esista, ma Alice non vuole rivelarlo. Se manda l'intero riempimento, Bob è convinto, ma il segreto è perso.

Nella versione classica zero-knowledge, Alice e Bob interagiscono. Un vecchio modello mentale usa tessere coperte. Alice nasconde la griglia risolta, rinomina segretamente i simboli prima di ogni turno e lascia Bob ispezionare un vincolo locale scelto a caso: una riga, una colonna, un blocco o un gadget. Se le celle aperte mostrano simboli tutti diversi, Bob guadagna fiducia. Poi tutto viene ricoperto e i simboli sono rinominati da capo.

Come i protocolli classici gestiscono davvero le celle-indizio

Il trucco della rinomina ha un punto cieco. Le regole di riga, colonna e blocco dicono tutte "queste celle sono tutte diverse", e *tutte diverse* sopravvive a qualunque rinomina dei simboli. Ma un indizio dice "questa cella contiene esattamente 5"; dopo la rinomina Bob vede solo $\sigma(5)$, un simbolo mascherato, senza conoscere σ. Così non può controllare nulla. Se lasciato così, Alice potrebbe provare che esiste *qualche* griglia valida ignorando gli indizi stampati, che non prova nulla su *questo* puzzle. La letteratura classica ha due riparazioni standard.

La palette. Aggiungi una riga extra di N celle alla griglia nascosta: una palette che Alice riempie con i simboli 1…N in un ordine pubblico fisso, e poi rinomina insieme a tutto il resto, così contiene $\sigma(1)$…σ(N). Il controllo casuale di Bob ha ora un'opzione extra. Oltre a scegliere una riga, colonna, blocco o gadget da aprire, può scegliere **la palette più una cella-indizio**. Alice scopre entrambe; la palette rivela la rinomina di quel round, e Bob controlla che la cella-indizio mostri esattamente la versione rinominata dell'indizio stampato. Questo resta zero-knowledge perché Bob impara solo σ, pescata fresca a ogni turno e inutile da sola, e il valore di una cella che conosceva già dal puzzle.

Compilare via gli indizi. Una variante più strutturale elimina il controllo speciale invece di aggiungerla. Invece di *verificare* il valore dell'indizio, lo forza con vincoli di differenza: collega la cella-indizio a ogni cella della palette tranne quella che porta il suo valore – "diversa da $\sigma(1)$, diversa da $\sigma(2)$, …, diversa da tutto tranne $\sigma(5)$." L'unico simbolo che la cella può legalmente contenere è quello dell'indizio. Ogni vincolo è di nuovo del tipo "queste due differiscono", invariante sotto rinomina e controllabile come una riga.

Il protocollo fisico. Il protocollo reale con carte per Sudoku (Gradwohl, Naor, Pinkas e Rothblum, 2007) non usa rinomina e sistema gli indizi prima ancora che inizi il nascondimento. Per ogni cella, Alice posa tre carte identiche con il valore della cella: coperte per le celle segrete, ma **scoperte per le celle-indizio**, così Bob vede con i propri occhi che gli indizi sono rispettati prima che le carte vengano voltate. Poi una carta da ogni cella va nel pacchetto della sua riga, una in quello della colonna, una in quello del blocco; ogni pacchetto viene mescolato e rivelato, e Bob controlla che contenga tutti gli N simboli. Il mescolamento distrugge l'informazione di posizione, cioè lo zero-knowledge, ma gli indizi erano già fissati al momento della distribuzione.

Questo non è il protocollo del lavoro. È il modello mentale dello zero-knowledge classico:

- Alice e Bob vanno avanti e indietro.
- Bob sceglie controlli casuali.
- Alice rivela solo consistenza locale, non l'intera soluzione.
- La prova di privacy funziona mostrando che la vista di Bob avrebbe potuto essere generata senza la soluzione segreta di Alice.

Lo zero-knowledge classico è quindi costruito attorno a un fatto positivo:

> Un simulatore esiste davvero.

Ora rimuovi le parti comode. Alice manda una stringa di prova e se ne va. Non c'è una configurazione

fidata, nessuna stringa casuale condivisa preparata in anticipo, e Bob non deve mai accettare un puzzle falso. È proprio il contesto in cui lo zero-knowledge classico non sopravvive.

Serve ancora un elemento. Fissiamo un **sistema formale di prova**: nel senso logico, un insieme di assiomi accompagnato da regole meccaniche per verificare dimostrazioni scritte. ZFC, il sistema assiomatico standard usato in gran parte della matematica, è l'esempio canonico. Da qui in poi tutto è relativo a un sistema formale scelto in anticipo; la costruzione funziona per qualunque sistema fissato, ZFC compreso.

La versione in stile Gödel mantiene la storia del mega-Sudoku ma cambia la prova.

Scegli un secondo sistema di vincoli della stessa dimensione mostrata, chiamalo **D**. Per la storia, S e D sono due puzzle MegaSudoku(n) nello stesso formato. Dietro le quinte, D può essere nato come una formula logica difficile di dimensione diversa; se serve, può essere riempito con vincoli fittizi innocui per adattarsi alla stessa griglia. D è costruito da una formula logica che è in realtà **insoddisfacibile**: non esiste alcuna assegnazione di valori che renda veri tutti i suoi vincoli, proprio come un puzzle rotto non ha griglia completata valida.

Ma D non deve essere un puzzle rotto facile da smascherare. Deve essere falso in un modo che il sistema formale scelto non possa certificare con una dimostrazione breve. Se quel sistema potesse confutare D rapidamente, la costruzione perderebbe il suo punto: la via alternativa che avrebbe potuto produrre prove senza il segreto di Alice verrebbe esclusa formalmente, e con essa la garanzia di privacy. D viene quindi scelto da una famiglia di formule che il sistema fissato non può confutare in modo efficiente.

La prova a un messaggio di Alice riguarda allora un enunciato o/o:

> o il vero mega-Sudoku S ha una soluzione, oppure l'esca D ha una soluzione.

Questo è il legame logico. D **non** è generato in qualche modo magico che renda vero S. La prova non dice "D non ha soluzione, quindi S ha soluzione." Prova la disgiunzione **S o D**. La solidità perfetta dice che una disgiunzione falsa non può avere una prova valida. Poiché D è falso nella realtà, l'unico modo in cui la disgiunzione può essere vera è che S sia vero. Quindi, se la prova è accettata, S deve avere una soluzione.

Ma per la parte in stile zero-knowledge, chiedi che cosa succederebbe se D *avesse* una soluzione. Quella soluzione-esca funzionerebbe da testimone alternativo. Permetterebbe di produrre prove senza conoscere la vera soluzione mega-Sudoku di Alice: un simulatore. In realtà D non ha soluzione, quindi questa via è chiusa. Il punto è che il sistema formale non può dimostrare efficientemente che quella via è chiusa.

D ha quindi due lavori. Per la **solidità**, D è falso, quindi una prova valida di "S o D" forza S. Per lo **zero-knowledge effettivo**, D è difficile da confutare, quindi il sistema formale non può escludere rapidamente la via alternativa che avrebbe reso possibile la simulazione.

Il test di sicurezza non è più:

> Possiamo provare che un simulatore esiste davvero?

Diventa:

> Il sistema formale scelto può dimostrare efficientemente che un simulatore è impossibile?

Se la risposta è no, segue qualcosa di sorprendentemente forte: ogni garanzia di sicurezza che (a) può essere osservata eseguendo un test e (b) segue dimostrabilmente, all'interno di quel sistema formale, dall'esistenza di un simulatore, vale davvero. Un attacco riuscito a una di esse ammonterebbe alla confutazione breve mancante, e quella confutazione breve non esiste. È questo il senso di "effettivo" in *zero-knowledge effettivo*.

Il contrasto da aula è:

Zero-knowledge classico: le prove sono sicure perché esiste un simulatore.

Zero-knowledge effettivo in stile Gödel: le prove sono trattate come sicure per test di sicurezza osservabili perché il sistema formale non può dimostrare efficientemente che un simulatore è impossibile.

La seconda affermazione è più debole. Ed è anche il motivo per cui il lavoro può mantenere le tre caratteristiche che rompevano la versione classica: un solo messaggio, nessuna configurazione iniziale e solidità perfetta.

Il nuovo test: non puoi provare che il simulatore è assente

Il rilassamento di Ilango cambia la domanda.

Lo zero-knowledge classico chiede:

> Esiste un simulatore?

Lo **zero-knowledge effettivo** chiede qualcosa di più debole:

> Il sistema formale scelto può dimostrare efficientemente che non esiste alcun simulatore?

Sembra un trucco tecnico, ma è il cuore dell'idea. La costruzione vive in uno stato strano: un simulatore in realtà non esiste, e il lavoro lo dice esplicitamente, ma il sistema formale fissato non può dimostrare efficientemente che non esiste. Se ogni conseguenza cattiva che ti interessa richiederebbe una tale confutazione, il sistema si comporta comunque come zero-knowledge per quelle conseguenze.

Qui entra Gödel. Non come decorazione, e non come "Gödel rende sicura la crittografia". Il collegamento passa per la **teoria della dimostrazione**. Un sistema di prova è detto **ottimale** se è, in un senso preciso, il migliore possibile: quando qualunque altro sistema può confutare una formula

del tipo rilevante con una prova breve, anche quello ottimale può farlo con una prova al massimo polinomialmente più lunga. Krajíček e Pudlák congetturarono nel 1989 che **non esiste alcun sistema di prova ottimale**. Qualunque sistema tu fissi, ne esiste un altro capace di dimostrare alcune famiglie di enunciati veri in modo molto più conciso.

Il lavoro assume questa congettura in una forma leggermente più forte, valida **infinitamente spesso** (*infinitely often*), standard quando le congetture sono usate crittograficamente. La conseguenza, tramite un teorema di Krajíček e Pudlák, è concreta: per ogni sistema formale esiste una sequenza di formule realmente insoddisfacibili che quel sistema non può confutare con prove brevi e che, tuttavia, un algoritmo efficiente può generare. Quest'ultima proprietà, l'uniformità, trasforma l'idea da esistenza astratta ad algoritmo reale che Alice può eseguire: le formule-esca D escono da una catena di montaggio.

La mossa crittografica è mettere al lavoro quella scarsità di potenza dimostrativa.

Che cosa fa la costruzione

Ecco la costruzione del lavoro, spogliata della forma.

Fissa un sistema formale, per esempio ZFC. Sotto l'assunzione di complessità delle prove, esiste una sequenza generabile efficientemente di formule che sono davvero insoddisfacibili, ma per cui quel sistema non dispone di alcuna prova breve di insoddisfacibilità.

Ora costruisci una prova a un messaggio di questa forma:

> o il vero enunciato è soddisfacibile, oppure questa speciale formula difficile è soddisfacibile.

La formula speciale non è soddisfacibile. Quindi, se la macchina di prova sottostante è perfettamente solida, accettare il messaggio significa comunque che il vero enunciato è vero. Questo dà solidità perfetta.

Ma per la sicurezza in stile zero-knowledge, immagina che la formula speciale *fosse* soddisfacibile. Allora il suo testimone potrebbe essere usato per simulare prove senza conoscere il testimone reale. La formula non è soddisfacibile nella realtà, ma il sistema formale non può dimostrarlo efficientemente. Quindi non può provare efficientemente che il simulatore è impossibile.

Questo è il cardine. Il sistema non nasconde il segreto producendo un simulatore classico. Nasconde il segreto, per una grande classe di test di sicurezza osservabili, dietro l'incapacità del sistema formale di certificare l'assenza del simulatore.

Che cosa afferma il lavoro

Il teorema principale arriva a strati. Il risultato centrale è questo:

Sotto un'assunzione crittografica standard, l'esistenza di **prove non interattive con indistinguibilità del testimone** (*non-interactive testimone-indistinguishable proofs*), oggetti ben studiati che seguono da diversi insiemi di assunzioni consolidate, e sotto la congettura di complessità delle prove che **non esiste alcun sistema di prova ottimale, infinitamente spesso**, il lavoro costruisce, per ogni sistema formale scelto, un dimostratore e un verificatore a un solo messaggio per NP/SAT, con **solidità perfetta** e senza configurazione iniziale, che è zero-knowledge effettivo rispetto a quel sistema. NP/SAT è il denominatore comune "più difficile" dei problemi tipo puzzle; il mega-Sudoku è un costume che indossa.

Per l'affermazione più ampia sul preservare proprietà di sicurezza falsificabili, il lavoro aggiunge un'altra assunzione standard, la credenza di derandomizzazione **P = BPP**: in breve, la casualità non dà agli algoritmi potere essenziale in più.

Tradotto fuori dal linguaggio dei teoremi:

- La prova è un messaggio.
- Non serve una configurazione fidata.
- Gli enunciati falsi non possono essere provati.
- Il dimostratore non è zero-knowledge classico: non esiste un simulatore.
- Ma ogni conseguenza di sicurezza falsificabile, basata su giochi, dello zero-knowledge classico può essere ottenuta in questo contesto.

"Falsificabile" conta. Significa che un fallimento di sicurezza può essere testato eseguendo un avversario in un gioco. Molte definizioni crittografiche hanno questa forma: l'avversario può distinguere due cifrati, invertire una funzione, recuperare un testimone o vincere un esperimento specificato? Il teorema fornisce un dimostratore per ciascuna proprietà falsificabile, una alla volta. Un singolo dimostratore che soddisfi *ogni* proprietà falsificabile insieme è probabilmente impossibile: il vecchio attacco di riutilizzabilità ("Bob può mostrare la prova ad altri") è esso stesso una proprietà falsificabile, e qui fallisce davvero. La proposta del lavoro è che un singolo dimostratore possa plausibilmente coprire tutte le proprietà falsificabili *naturali*, quelle che compaiono davvero nella pratica crittografica, ma quella parte è un teorema condizionale appoggiato a una nozione informale di "naturale" più una congettura esplicita. La garanzia punta ai fallimenti osservabili, non a ogni significato filosofico o basato sulla simulazione di segretezza.

Vale la pena nominare un corollario concreto: la costruzione dà le prime prove non interattive che **nascondono il testimone** (*witness hiding*) con dimostratore uniforme, "una prova di un puzzle non ti aiuta a trovarne la soluzione", senza interazione e senza configurazione iniziale; un oggetto dal suono modesto che aveva resistito per decenni.

Che cosa non dice

Questa è la sezione che tiene onesto il pezzo.

Non dice che i vecchi teoremi di impossibilità fossero sbagliati. La costruzione li evita cambiando la definizione.

Non dà zero-knowledge ordinario, classico, senza interazione, senza configurazione iniziale e con solidità perfetta. Il lavoro dice esplicitamente che il dimostratore costruito non ha un simulatore.

Non significa che la prova non possa essere riutilizzata. Una prova a un messaggio può ancora essere mostrata a qualcun altro; il lavoro non preserva proprietà di **negabilità** (*deniability*). Lo zero-knowledge non interattivo con configurazione fidata ha la stessa limitazione.

Non significa che questo sia un protocollo pratico pronto per l'impiego pratico. È teoria della complessità e fondamenti della crittografia. Il risultato dipende da grandi assunzioni di complessità delle prove e crittografia, e la costruzione riguarda ciò che è possibile in linea di principio.

Non rende "Gödel" una primitiva magica di sicurezza. Il collegamento con Gödel passa per sistemi di prova, sistemi di prova ottimali e analoghi finiti dell'incompletezza. L'intuizione utilizzabile non è "l'incompletezza protegge la tua password". È: se un sistema formale non può dimostrare efficientemente che un simulatore è impossibile, allora attacchi che richiederebbero quella prova possono essere bloccati a livello delle definizioni di sicurezza.

Perché è comunque interessante

La crittografia trasforma spesso la durezza in sicurezza. Fattorizzare è difficile, quindi assunzioni in stile RSA diventano utili. Problemi reticolari sono difficili, quindi la crittografia reticolare diventa utile. Qui la durezza è più strana: non "difficile calcolare un segreto", ma "difficile provare che un certo oggetto di prova non può esistere".

Per questo il lavoro sembra insolito. Tratta assiomi e sistemi formali quasi come risorse crittografiche. L'impossibilità usuale dice che c'è una tensione tra solidità e simulazione. La mossa di Ilango mette quella tensione dietro uno schermo della teoria della dimostrazione: il simulatore è assente, ma il sistema formale non può esporre efficientemente quell'assenza.

Per un lettore, la parte sorprendente non è che questo sostituirà i sistemi zero-knowledge di oggi. Probabilmente non lo farà, almeno non direttamente. La parte sorprendente è che un limite della logica matematica possa essere usato in modo costruttivo: non solo come muro, ma come una forma di copertura.

Quanto sono solide le prove?

È un lavoro teorico, quindi qui la solidità si valuta diversamente che in biologia o astronomia. La domanda non è se un esperimento sia stato replicato, ma se definizioni, assunzioni e passaggi della dimostrazione sostengano davvero la conclusione.

La prova è formale, e il lavoro è esplicito sulle assunzioni. Le assunzioni non sono casuali. Le prove non interattive con indistinguibilità del testimone sono oggetti standard in crittografia e seguono da diversi pacchetti di assunzioni stabiliti. La congettura di non esistenza di sistemi di prova ottimali è una congettura centrale in complessità delle prove. P = BPP è una credenza standard di derandomizzazione usata solo per il teorema più ampio sulle proprietà falsificabili.

Il lavoro sostiene anche che le assunzioni sono il prezzo giusto, non un'impalcatura arbitraria: prova un converso secondo cui sono essenzialmente necessarie. Se costruzioni come questa esistono, allora devono esistere prove non interattive con indistinguibilità del testimone e, concedendo funzioni unidirezionali standard, non può esistere alcun sistema di prova ottimale. E le assunzioni sono "win-win": confutarne una sarebbe già una scoperta importante in complessità delle prove, crittografia o teoria della complessità.

Ma poiché il risultato è condizionale, anche la fiducia è condizionale. Se quelle assunzioni falliscono, l'interpretazione del teorema cambia. E anche se tengono, la garanzia non è zero-knowledge classico pieno; è la versione rilassata, formulata in termini di teoria della dimostrazione, proposta dal lavoro.

La fiducia giusta è quindi alta che il lavoro stabilisca un risultato condizionale coerente di possibilità; moderata che le sue assunzioni descrivano il mondo crittografico in cui viviamo; bassa per qualunque conseguenza pratica immediata.

Perché conta

Il lavoro apre una strada che doveva essere chiusa.

La teoria classica dice: lo zero-knowledge pieno non può essere un messaggio senza configurazione iniziale e non può essere perfettamente solido. Il lavoro di Ilango dice: se chiediamo le conseguenze dello zero-knowledge che possono essere testate in giochi di sicurezza, e se permettiamo alla definizione di sicurezza di dipendere da ciò che un sistema formale può o non può confutare efficientemente, allora molto del comportamento utile può essere recuperato, con un messaggio, senza configurazione iniziale e con solidità perfetta.

Non è un piccolo ritocco definitorio. È un modo diverso di pensare le garanzie crittografiche. Invece di chiedere solo che cosa esiste, chiedi che cosa il sistema formale scelto può escludere. Invece di trattare l'improvabilità come un fastidio filosofico, usala come struttura.

Il mondo pratico forse non cambierà domani. Ma la mappa concettuale sì. Ora esiste un senso formale in cui "nessuno può provare efficientemente che il segreto sia trapelato" può essere abbastanza forte da recuperare molte delle protezioni basate su giochi di sicurezza che volevamo da "il segreto

non è trapelato".

È per questo che Gödel sta nel titolo.

In breve

Le prove zero-knowledge permettono a un dimostratore di convincere un verificatore che un enunciato è vero senza rivelare il testimone. I risultati classici di impossibilità dicono che lo zero-knowledge non può essere compresso in un messaggio senza configurazione iniziale e non può avere solidità perfetta. Il lavoro di Rahul Ilango non confuta quelle impossibilità. Definisce una nozione più debole, lo **zero-knowledge effettivo** (*effectively zero-knowledge*): invece di richiedere che un simulatore esista davvero, richiede che un sistema di prova scelto, un sistema formale come ZFC, non possa provare efficientemente che nessun simulatore esiste. Sotto grandi assunzioni di crittografia (prove non interattive con indistinguibilità del testimone) e complessità delle prove (non esiste alcun sistema di prova ottimale), il lavoro costruisce dimostratori a un solo messaggio per NP/SAT, senza configurazione iniziale e con solidità perfetta, che ottengono le conseguenze falsificabili basate su giochi di sicurezza dello zero-knowledge proprietà per proprietà. Un singolo dimostratore che copra tutte queste proprietà "naturali" è un'estensione ulteriore, in parte congetturale; coprire letteralmente ogni proprietà falsificabile è probabilmente impossibile, perché le prove restano riutilizzabili. Il risultato è teorico e condizionale, non una primitiva già impiegata in sistemi reali, ma mostra un modo nuovo di usare l'improvabilità nella teoria della dimostrazione come risorsa crittografica.

Valutazione critica

Cosa mostra il lavoro: sotto assunzioni dichiarate, si possono costruire dimostratori a un solo messaggio, senza configurazione iniziale e perfettamente solidi per NP/SAT che sono zero-knowledge effettivo rispetto a qualunque sistema di prova scelto, e che ottengono ciascuna conseguenza falsificabile basata su giochi di sicurezza dello zero-knowledge classico.

Cosa è plausibile ma non provato incondizionatamente: che le assunzioni necessarie di complessità delle prove e crittografia tengano. Sono assunzioni serie e studiate, e il lavoro mostra che sono essenzialmente necessarie oltre che sufficienti, ma restano assunzioni.

Cosa non mostra: zero-knowledge classico senza interazione, senza configurazione iniziale e con solidità perfetta; un sistema pratico pronto all'uso; negabilità (*deniability*) o non-riutilizzabilità delle prove; o che il teorema di incompletezza di Gödel da solo renda sicura la crittografia.

Limiti principali: la garanzia è un rilassamento dello zero-knowledge; la versione più ampia dipende da più assunzioni; le affermazioni su un singolo dimostratore universale restano in parte congetturali; e il risultato è soprattutto fondazionale.

Quanta fiducia dovrebbe avere un lettore generale? Alta che questo sia un importante risultato teorico condizionale se si accettano le definizioni. Moderata che le assunzioni catturino la realtà.

Bassa per un impiego pratico immediato. La conclusione prudente è: il lavoro non rompe le impossibilità dello zero-knowledge; trova un nuovo modo, fondato sulla teoria della dimostrazione, per aggirare le parti che contano per molti giochi di sicurezza.

Fonti

Ilango, IACR ePrint 2025/1296
https://eprint.iacr.org/2025/1296

FOCS 2025, DOI 10.1109/FOCS63196.2025.00058
https://doi.org/10.1109/FOCS63196.2025.00058