The Clean Paper

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

Uma prova criptográfica pode esconder seu segredo tornando difícil provar a ausência do simulador

Provas de conhecimento zero deixam alguém provar uma afirmação sem revelar a testemunha. A teoria clássica diz que você não pode ter isso, no sentido completo, com uma mensagem, nenhuma configuração confiável e solidez perfeita. O artigo de Rahul Ilango não faz essa impossibilidade desaparecer. Ele muda o alvo: em vez de exigir que um simulador realmente exista, pede que nenhum sistema de provas escolhido consiga provar eficientemente que o simulador não existe. Sob grandes suposições de complexidade de provas e criptografia, isso basta para recuperar consequências falsificáveis, baseadas em jogos, do conhecimento zero propriedade por propriedade. O ponto não é uma primitiva de internet plug-and-play. É uma forma teórico-probatória de transformar indemonstrabilidade matemática em cobertura criptográfica.

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

O truque não é provar que o segredo está escondido

Comece pela versão mais simples de conhecimento zero.

Alice quer convencer Bob de que um quebra-cabeça Sudoku tem solução. Se ela envia a solução, Bob fica convencido, mas o quebra-cabeça fica arruinado. O que ela quer é mais estranho: uma prova de que existe uma solução, sem revelar a solução.

Essa é a promessa de uma prova de conhecimento zero. O provador (Alice) convence o verificador (Bob) de que uma afirmação é verdadeira sem revelar nada além da verdade da afirmação.

O problema é que essa promessa custa alguma coisa. Uma prova matemática comum tem duas características confortáveis. Ela é **uma mensagem**: você a escreve, entrega e vai embora. E ela é **perfeitamente sólida**: uma afirmação falsa não tem prova válida alguma. Resultados clássicos de impossibilidade dizem que conhecimento zero precisa abrir mão das duas características — e não apenas das duas juntas; cada uma, sozinha, já fica fora de alcance.

Primeiro, uma prova de conhecimento zero precisa de conversa. Se Alice envia uma única mensagem, sem nenhuma configuração confiável combinada de antemão, a garantia de conhecimento zero colapsa — e isso vale por mais solidez que você esteja disposto a trocar em compensação.

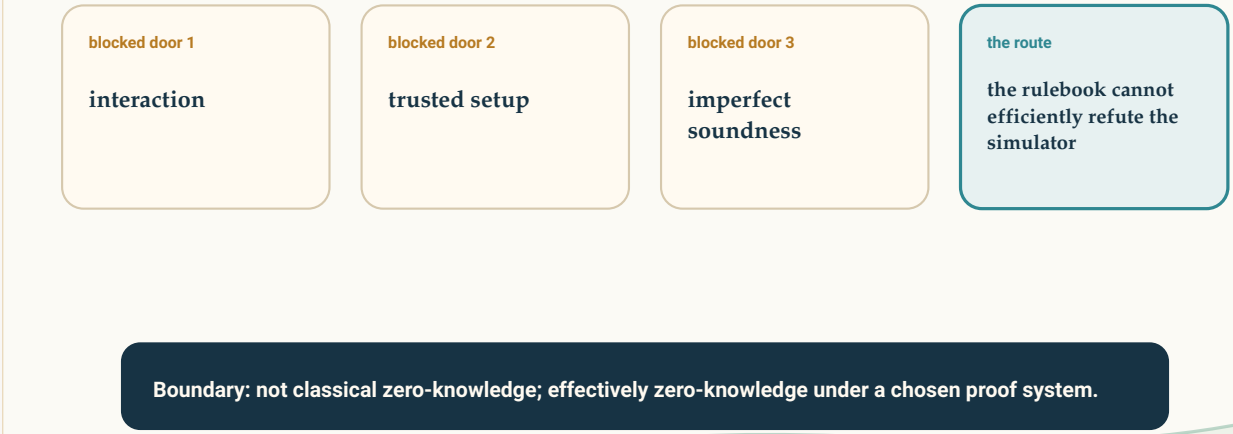
Segundo, uma prova de conhecimento zero precisa de uma pequena tolerância a erro. Exigir solidez perfeita acaba destruindo silenciosamente também a interação: um verificador que nunca pode ser enganado, não importa quais escolhas aleatórias faça, poderia muito bem fixar essas escolhas de antemão — e, uma vez que o verificador é previsível, Alice consegue responder a tudo em uma única mensagem, que é exatamente o caso que já quebrou.

O artigo de Rahul Ilango trata de uma forma de contornar esse muro duplo. Não fingindo que o muro não existe, e não produzindo conhecimento zero clássico no cenário impossível. O movimento é mais sutil: enfraquecer o que “não revela nada” significa, mas enfraquecê-lo de uma forma que preserve as propriedades de segurança que criptógrafos conseguem de fato testar.

O resultado se chama **conhecimento zero efetivo**.

A proof without leaking the witness

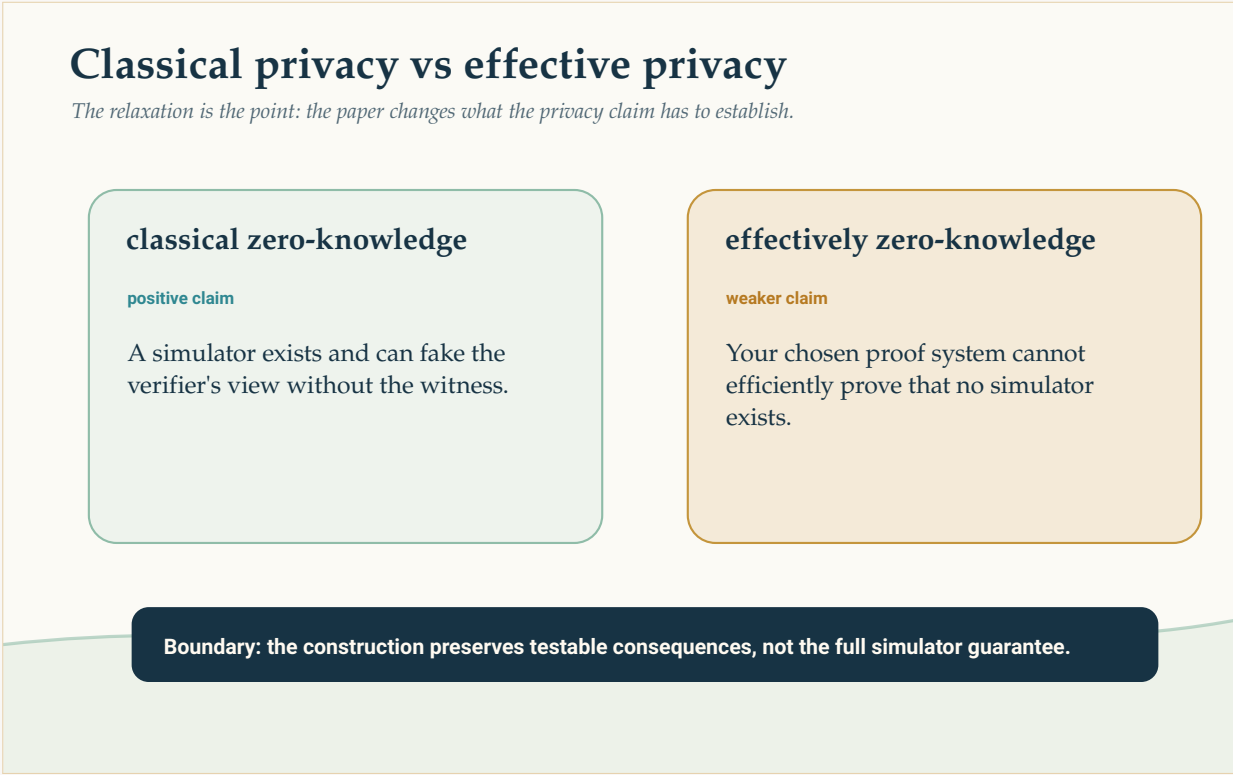
The paper keeps the ordinary-proof shape by weakening what privacy must prove.



Conhecimento zero é bloqueado em três portas — interação, configuração confiável e solidez imperfeita. A construção de Ilango passa por uma porta diferente: o livro de regras não consegue refutar eficientemente o simulador.

Classical privacy vs effective privacy

The relaxation is the point: the paper changes what the privacy claim has to establish.



Conhecimento zero clássico pergunta se existe um simulador; "conhecimento zero efetivo" pergunta apenas se

o seu livro de regras escolhido consegue provar eficientemente que não existe um. A pergunta mais fraca é o que permite à construção manter uma mensagem, nenhuma configuração e solidez perfeita.

O teste antigo: existe um simulador

A forma clássica de formalizar conhecimento zero usa um ajudante fictício chamado **simulador**.

A ideia é esta: imagine Jane, que **não** conhece o segredo de Alice. Se Jane consegue gerar, inteiramente sozinha, provas que se parecem exatamente com as provas que Bob teria recebido de Alice, então as provas de Alice não ensinaram nada novo a Bob. Jane já poderia falsificar a experiência sem o segredo de Alice.

Então o conhecimento zero clássico pede um simulador real. Precisa haver um algoritmo eficiente capaz de produzir provas com aparência falsa sem conhecer o segredo — a *testemunha*, no jargão; para Sudoku, a testemunha é simplesmente a grade resolvida.

Essa definição é poderosa, mas é também exatamente onde a velha impossibilidade morde. Eis a intuição. Uma prova verdadeiramente não interativa é apenas uma string. Quando Bob tem essa string, pode mostrá-la a outra pessoa: ele ganhou a capacidade de provar a afirmação a outros, o que já soa como mais do que "nada". Os teoremas clássicos afiam essa intuição nas impossibilidades acima.

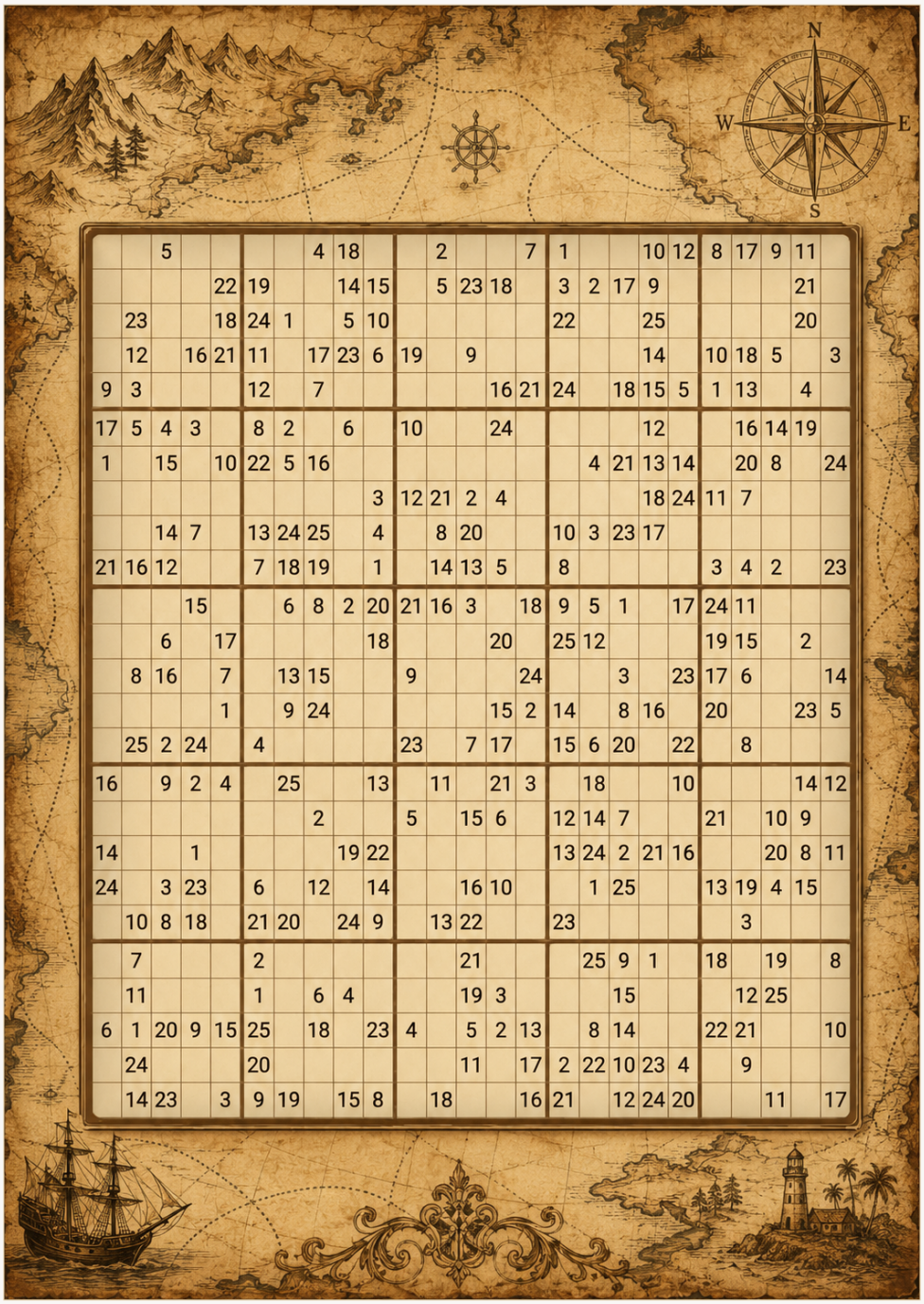
> **As três propriedades em que este artigo insiste**
>
> O título do artigo nomeia três restrições:
> **Sem interação:** Alice envia uma string de prova. Não há protocolo de ida e volta.
> **Sem configuração:** Alice e Bob não dependem de uma string comum de referência confiável nem de outra aleatoriedade pública previamente combinada. Muitos sistemas chamados de "conhecimento zero não interativo" ainda dependem de configuração; este artigo quer configuração zero.
> **Solidez perfeita:** uma afirmação falsa não tem prova válida. Não "quase nunca aceita"; nenhuma prova válida existe.
> Essas três propriedades são exatamente o que a matemática escrita comum tem — e, como explicado acima, o conhecimento zero clássico não consegue mantê-las.

Uma versão MegaSudoku da diferença

Eis uma forma deliberadamente simplificada de sentir a diferença.

Não use um Sudoku comum 9 por 9 para a parte séria da analogia. Ele é pequeno demais e finito demais: um computador pode simplesmente resolvê-lo ou provar que ele não tem solução. Em vez

disso, imagine uma família de quebra-cabeças **MegaSudoku(n)**. Escale a regra usual: escolha um tamanho de bloco *n*, deixe *N* = *n*^2 e construa uma grade *N* por *N* dividida em blocos *n* por *n*, com *N* símbolos. O Sudoku comum é apenas o minúsculo caso *n* = 3, *N* = 9: uma grade 9 por 9, blocos 3 por 3 e nove símbolos. A história de complexidade de provas só começa quando *n* pode crescer, e quando a grade pode carregar gadgets extras que a fazem se comportar como uma fórmula SAT fantasiada de Sudoku. Uma fórmula SAT é apenas uma lista de restrições sim/não: você consegue atribuir valores verdadeiro/falso às variáveis de modo que toda restrição seja satisfeita?



Um Sudoku 25x25: suas regras podem ser checadas sem revelar a grade finalizada — um substituto visual para uma prova que verifica uma solução escondida, a testemunha.

AI-generated editorial thumbnail — The Clean Paper CC BY 4.0

Sudoku e SAT: o mesmo quebra-cabeça em dois trajes

A alegação de que um Sudoku pode "se comportar como uma fórmula SAT" não é uma metáfora. A tradução funciona nas duas direções, e a direção fácil pode ser escrita por completo.

De Sudoku para SAT. SAT só fala verdadeiro/falso, então dê a ele uma variável booleana para cada tripla (linha, coluna, valor): *x(r,c,v)* significa "a célula na linha *r*, coluna *c* contém o valor *v*." Um Sudoku 4 por 4 (blocos 2 por 2, valores 1-4) precisa de 4·4·4 = 64 variáveis; o 9 por 9 clássico precisa de 729. Cada regra do Sudoku vira então um lote de cláusulas. (Uma cláusula é um OR de variáveis ou de suas negações; a fórmula inteira é o AND de todas as cláusulas.)

Cada célula contém pelo menos um valor — uma cláusula por célula:

$$x(1,1,1) \vee x(1,1,2) \vee x(1,1,3) \vee x(1,1,4)$$

Cada célula contém no máximo um valor — uma cláusula "não ambos" para cada par de valores:

$$\neg x(1,1,1) \vee \neg x(1,1,2) \quad \neg x(1,1,1) \vee \neg x(1,1,3)$$

... e assim por diante para todos os seis pares.

Cada linha contém cada valor — para a linha 1 e o valor 3: pelo menos uma vez,

$$x(1,1,3) \vee x(1,2,3) \vee x(1,3,3) \vee x(1,4,3)$$

e no máximo uma vez: $\neg x(1,1,3) \vee \neg x(1,2,3)$, e assim por diante para cada par de células na linha.

Colunas e blocos — lotes idênticos; só muda o grupo de células. Para o bloco superior esquerdo e o valor 2:

$$x(1,1,2) \vee x(1,2,2) \vee x(2,1,2) \vee x(2,2,2)$$

mais as cláusulas par a par de "não ambos".

As pistas impressas — a parte mais simples: cada pista é uma cláusula com uma única variável. Um 3 impresso no canto superior esquerdo vira a cláusula

$$x(1,1,3)$$

O AND de tudo isso é satisfatível exatamente quando o Sudoku tem uma solução — e uma atribuição satisfatória *é* a solução: leia quais x(r,c,v) são verdadeiros e preencha a grade. Para um 9 por 9, isso dá 729 variáveis e alguns milhares de cláusulas, que um resolvedor SAT moderno despacha em milissegundos. Note a cláusula da pista x(1,1,3): ela diz "esta célula é exatamente 3", não "estas células são todas diferentes" — a mesma assimetria que vai forçar o truque extra para células de pista na nota de protocolo mais abaixo.

De SAT para Sudoku. O artigo precisa da direção oposta, mais difícil: dada uma fórmula SAT *arbitrária*, construir um mega-Sudoku que tenha solução exatamente quando a fórmula tem. As regras nativas do Sudoku só conseguem dizer "estas células são todas diferentes", então restrições lógicas arbitrárias precisam ser *construídas* — e isso é exatamente o que são os gadgets. Um gadget é um pequeno aglomerado pré-fabricado de células, um por cláusula

da fórmula, em que células designadas desempenham o papel de variáveis (o símbolo que contêm codifica verdadeiro ou falso) e as restrições internas do aglomerado são projetadas para que seus únicos preenchimentos legais correspondam a atribuições que satisfazem aquela cláusula. Essa é uma construção padrão em provas de NP-completude; para Sudoku generalizado, foi feito por Yato e Seta em 2003.
Juntas, as duas direções dizem que Sudoku N por N e SAT são o mesmo problema usando trajes diferentes. É isso que permite ao artigo contar uma história sobre todo NP usando grades e símbolos.

A testemunha ainda é fácil de visualizar. Alice conhece um preenchimento completo e válido do mega-Sudoku. Bob quer ser convencido de que esse preenchimento existe, mas Alice não quer revelá-lo. Se ela envia o preenchimento inteiro, Bob fica convencido, mas o segredo acaba.

Na versão clássica de conhecimento zero, Alice e Bob interagem. Um velho modelo mental usa peças cobertas. Alice esconde a grade resolvida, renomeia secretamente os símbolos antes de cada rodada e deixa Bob inspecionar uma restrição local escolhida ao acaso: uma linha, uma coluna, uma caixa ou um gadget. Se as células abertas mostram símbolos todos diferentes, Bob ganha confiança. Depois tudo é coberto novamente e os símbolos são renomeados de novo. (Uma complicação: as pistas dadas do quebra-cabeça precisam de um truque extra, porque renomear os símbolos também as esconde. A nota abaixo explica como os protocolos clássicos resolvem isso; a imagem de brinquedo basta para o que vem.)

Como os protocolos clássicos realmente lidam com as células de pista

O truque de renomeação tem um ponto cego. As regras de linha, coluna e caixa dizem todas "estas células são todas diferentes", e *todas diferentes* sobrevive a qualquer renomeação dos símbolos. Mas uma pista diz "esta célula contém exatamente 5", e depois da renomeação Bob só vê $\sigma(5)$ — algum símbolo mascarado — sem conhecer a renomeação σ. Ele não consegue checar nada. Se isso ficar sem conserto, Alice poderia provar que *alguma* grade válida existe ignorando completamente as pistas impressas, o que não prova nada sobre *este* quebra-cabeça. A literatura clássica tem dois reparos padrão.
A paleta. Acrescente uma linha extra de N células à grade escondida — uma paleta que Alice preenche com os símbolos 1…N em uma ordem pública fixa, e então renomeia junto com todo o resto, de modo que contenha $\sigma(1)$…$\sigma(N)$. O desafio aleatório de Bob agora tem uma opção extra. Além de escolher uma linha, coluna, caixa ou gadget para abrir, ele pode escolher **a paleta mais uma célula de pista**. Alice descobre ambas; a paleta revela a renomeação daquela rodada, e Bob checa que a célula de pista mostra exatamente a versão renomeada da pista impressa. Isso continua sendo conhecimento zero porque Bob aprende apenas σ — que é sorteado de novo a cada rodada e não vale nada sozinho — e o valor de uma célula que ele já conhecia pelo quebra-cabeça. Nada sobre as células secretas vaza, e um simulador pode falsificar a visão sorteando um σ aleatório. É sólido porque uma Alice trapaceira é pega com probabilidade fixa por rodada, e as rodadas são repetidas até a dúvida se tornar desprezível.
Compilar as pistas para fora. Uma variante mais estrutural remove o desafio especial em vez de acrescentá-lo. Em vez de *verificar* o valor da pista, force-o com restrições de diferença:

ligue a célula da pista a cada célula da paleta exceto aquela que carrega seu próprio valor — "diferente de $\sigma(1)$, diferente de $\sigma(2)$, …, diferente de tudo exceto $\sigma(5)$". O único símbolo que a célula pode legalmente conter é o da pista. Toda restrição agora volta a ser do tipo "estes dois diferem" — invariante sob renomeação, checável exatamente como uma linha. É a mesma manobra usada para vértices pré-coloridos no protocolo clássico de coloração de grafos, e é o espírito da palavra *gadgets* acima: no quadro MegaSudoku-como-SAT, as pistas são compiladas em gadgets de desigualdade como qualquer outra restrição.

O protocolo físico. O protocolo de cartas do mundo real para Sudoku (Gradwohl, Naor, Pinkas e Rothblum, 2007) não usa renomeação alguma e resolve as pistas antes mesmo de começar a ocultação. Para cada célula, Alice coloca três cartas idênticas com o valor da célula — viradas para baixo nas células secretas, mas **viradas para cima nas células de pista**, de modo que Bob veja com seus próprios olhos que as pistas são respeitadas antes de as cartas serem viradas. Depois uma carta de cada célula vai para o pacote de sua linha, uma para o de sua coluna e uma para o de sua caixa; cada pacote é embaralhado e revelado, e Bob checa que ele contém todos os N símbolos. O embaralhamento destrói a informação de posição (isso é o conhecimento zero), mas as pistas já tinham sido cravadas no momento da distribuição.

De qualquer forma, a lição é a mesma a que este artigo sempre retorna: um protocolo de conhecimento zero exige controle rigoroso sobre *quais* fatos continuam verificáveis depois que o restante é ocultado. Renomear preserva "todos diferentes" e apaga "igual a 5" — então "igual a 5" precisa ser contrabandeado de volta por outros meios.

Esse não é o protocolo do artigo. É o modelo mental para conhecimento zero clássico:

- Alice e Bob vão e voltam.
- Bob escolhe checagens aleatórias.
- Alice revela apenas consistência local, não a solução inteira.
- A prova de privacidade funciona mostrando que a visão de Bob poderia ter sido gerada sem a solução secreta de Alice.

Então o conhecimento zero clássico é construído em torno de um fato positivo:

> Um simulador realmente existe.

Agora remova as partes confortáveis. Alice envia uma string de prova e vai embora. Não há configuração confiável, nenhuma string aleatória compartilhada preparada de antemão, e Bob nunca deve aceitar um quebra-cabeça falso. Esse é o cenário em que o conhecimento zero clássico não consegue sobreviver.

Mais um personagem é necessário antes do truque. Fixe um **livro de regras**: um sistema formal de provas, no sentido dos lógicos — um conjunto fixo de axiomas mais regras mecânicas para checar provas matemáticas escritas. ZFC, os axiomas padrão da matemática, é o exemplo canônico. Tudo daqui em diante é formulado em relação a um livro de regras escolhido de antemão, e a escolha é flexível: a construção funciona para qualquer livro de regras que você fixe, incluindo ZFC.

(Uma nota sobre palavras, emprestada do próprio artigo: “sistema de provas” aqui sempre significa esse livro de regras — o sistema formal que checa provas matemáticas —, nunca as mensagens que Alice envia. A maquinaria de Alice e Bob é chamada de “o provador e o verificador”.)

A versão ao estilo Gödel mantém a história do mega-Sudoku, mas muda a prova.

Escolha um segundo sistema de restrições do mesmo tamanho exibido, chame-o de **D**. Para a história, S e D são dois quebra-cabeças MegaSudoku(n) no mesmo formato. Nos bastidores, D pode ter começado como uma fórmula lógica difícil de outro tamanho; se necessário, pode ser preenchido com restrições fictícias inofensivas para caber na mesma grade. D é construído a partir de uma fórmula lógica que é de fato **insatisfatível**: não há atribuição possível de valores que torne todas as suas restrições verdadeiras, assim como um quebra-cabeça quebrado não tem grade completa legal. Um exemplo de brinquedo seria uma fórmula que exige ao mesmo tempo “X é verdadeiro” e “X é falso”. Então D não tem preenchimento válido.

Mas D não pode ser um quebra-cabeça quebrado *fácil de expor*. O exemplo de brinquedo acima falha nisso: qualquer livro de regras refuta “X e não-X” em uma linha. D precisa ser falso de uma forma que o livro de regras escolhido não consiga certificar com um argumento curto. Se o livro de regras pudesse refutar D com uma prova curta, a história abaixo colapsaria: a rota alternativa que poderia ter produzido provas sem o segredo de Alice seria formalmente descartada, e junto com ela a garantia de privacidade. Então D é escolhido de uma família que o livro de regras fixado não consegue refutar eficientemente: não há prova curta, dentro desse livro de regras, de que D não tem solução.

A prova de uma mensagem de Alice é então sobre uma afirmação ou/ou:

> ou o mega-Sudoku real S tem uma solução, ou o chamariz D tem uma solução.

Esse é o elo lógico. D **não** é gerado de alguma forma mágica que torna S verdadeiro. A prova não está argumentando “D não tem solução, portanto S tem solução”. Ela prova a disjunção **S ou D**. A solidez perfeita diz que uma disjunção falsa não pode ter uma prova válida. Como D é falso na realidade — ele não tem solução —, a única forma de a disjunção ser verdadeira é S ser verdadeiro. Então, se a prova é aceita, S deve ter uma solução. O chamariz não pode fazer um S falso virar verdadeiro.

Mas, para a parte ao estilo conhecimento zero, pergunte o que aconteceria se D *tivesse* uma solução. Essa solução chamariz agiria como uma testemunha alternativa. Ela deixaria alguém produzir provas sem conhecer a solução real do mega-Sudoku de Alice — um simulador, em outras palavras. Na realidade D não tem solução, então essa rota de simulação está fechada. O ponto é que o livro de regras não consegue provar eficientemente que ela está fechada.

Então D cumpre duas funções. Para a **solidez**, D é falso, então uma prova válida de “S ou D” força S. Para o **conhecimento zero efetivo**, D é difícil de refutar, então o livro de regras não consegue descartar rapidamente a rota chamariz que teria tornado a simulação possível.

Então o teste de segurança deixa de ser:

> Conseguimos provar que um simulador realmente existe?

Ele se torna:

> O seu livro de regras consegue provar eficientemente que o simulador é impossível?

Se a resposta é não, algo surpreendentemente forte se segue: toda garantia de segurança que (a) pode ser observada rodando um teste e (b) segue demonstravelmente — dentro desse livro de regras — da existência de um simulador, de fato vale. Um ataque bem-sucedido a qualquer uma delas equivaleria ele próprio à refutação curta ausente, e a refutação curta ausente não existe. Essa é a parte "efetiva" do conhecimento zero efetivo.

Então o contraste em sala de aula é:

Conhecimento zero clássico: as provas são seguras porque existe um simulador.

Conhecimento zero efetivo ao estilo Gödel: as provas são tratadas como seguras para testes de segurança observáveis porque o livro de regras não consegue provar eficientemente que o simulador é impossível.

A segunda alegação é mais fraca. É também por isso que o artigo consegue manter as três características que quebraram a versão clássica: uma mensagem, nenhuma configuração e solidez perfeita.

O novo teste: você não consegue provar que o simulador está ausente

A relaxação de Ilango muda a pergunta.

Conhecimento zero clássico pergunta:

> Existe um simulador?

Conhecimento zero efetivo pergunta algo mais fraco:

> O seu livro de regras escolhido consegue provar eficientemente que não existe simulador?

Isso soa como uma esquiva técnica, mas é a ideia central. A construção vive em uma situação peculiar: um simulador não existe de fato — o artigo é explícito sobre isso —, mas o livro de regras que você fixou não consegue provar eficientemente que ele não existe. Se toda consequência ruim com que você se importa exigiria tal refutação, o sistema ainda se comporta como conhecimento zero para essas consequências.

É aqui que Gödel entra. Não como decoração, e não como "Gödel torna a cripto segura". A conexão é teórico-probatória. Um livro de regras é chamado de **ótimo** se é, em um sentido preciso, o melhor possível: sempre que qualquer livro de regras consegue refutar uma fórmula do tipo relevante com uma prova curta, o livro de regras ótimo também consegue, com uma prova no máximo polinomialmente mais longa. Krajíček e Pudlák conjecturaram em 1989 que **não existe sistema de provas ótimo**:

qualquer que seja o livro de regras que você fixe, algum outro livro de regras prova alguma família de afirmações verdadeiras de forma muito mais sucinta. Essa é uma das conjecturas centrais da complexidade de provas, e funciona como um análogo finito, formulado em termos de complexidade, do teorema da incompletude de Gödel: algumas afirmações verdadeiras não têm prova curta no livro de regras que você fixou — não porque sejam em princípio indemonstráveis, mas porque todo livro de regras fixo deixa algumas verdades curtas sem provas curtas.

O artigo assume essa conjectura (em uma forma "infinitamente muitas vezes" um pouco mais forte, padrão quando conjecturas são usadas criptograficamente). O ganho, por um teorema de Krajíček e Pudlák, é concreto: para todo livro de regras há uma sequência de fórmulas que são genuinamente insatisfatíveis, que o livro de regras não consegue refutar com provas curtas — e, crucialmente, que um algoritmo eficiente consegue *gerar*. Essa última propriedade, uniformidade, é o que transforma toda a ideia de uma afirmação de existência em um algoritmo real que Alice pode rodar: seus chamarizes D saem de uma linha de montagem, não do nada.

O movimento criptográfico é pôr essa escassez de poder de prova para trabalhar.

O que a construção está fazendo

Eis a construção do artigo, reduzida à sua forma.

Fixe um livro de regras — ZFC, digamos. Sob a suposição de complexidade de provas, há uma sequência gerável eficientemente de fórmulas que são de fato insatisfatíveis, mas para as quais o livro de regras não tem prova curta de que são insatisfatíveis.

Agora construa uma prova de uma mensagem desta forma:

> ou a afirmação real é satisfatível, ou esta fórmula especial difícil é satisfatível.

A fórmula especial difícil não é satisfatível. Portanto, se a maquinaria de prova subjacente é perfeitamente sólida, aceitar a mensagem ainda significa que a afirmação real é verdadeira. Isso dá solidez perfeita.

Mas, para a segurança semelhante a conhecimento zero, imagine que a fórmula especial difícil *fosse* satisfatível. Então sua testemunha poderia ser usada para simular provas sem conhecer a testemunha real. A fórmula não é satisfatível na realidade — mas o livro de regras não consegue provar isso eficientemente. Então ele não consegue provar eficientemente que o simulador é impossível.

Essa é a dobradiça. O sistema não esconde o segredo produzindo um simulador clássico. Ele esconde o segredo, para uma grande classe de testes de segurança observáveis, por trás da incapacidade do livro de regras de certificar que o simulador está ausente.

O que o artigo afirma

O teorema principal vem em camadas. O resultado central é este:

Sob uma suposição criptográfica padrão — a existência de **provas não interativas com indistinguibilidade de testemunha**, objetos bem estudados que seguem de vários pacotes estabelecidos de suposições — e sob a conjectura de complexidade de provas de que **não existe sistema de provas ótimo (infinitamente muitas vezes)**, o artigo constrói, para toda escolha de livro de regras, um provador e um verificador de uma mensagem para NP/SAT com **solidez perfeita** e sem configuração, que é conhecimento zero efetivo relativo a esse livro de regras. (NP/SAT é o "denominador comum mais difícil" padrão dos problemas parecidos com quebra-cabeças; mega-Sudoku é uma fantasia que ele veste.)

Para a alegação mais ampla sobre preservar propriedades de segurança falsificáveis, o artigo acrescenta mais uma suposição padrão, a crença de derandomização **P = BPP** (aproximadamente: aleatoriedade não dá aos algoritmos nenhum poder essencial extra).

Traduzido para fora da linguagem de teoremas:

- A prova é uma mensagem.
- Não há configuração confiável.
- Afirmações falsas não podem ser provadas.
- O provador não é conhecimento zero clássico — ele não tem simulador.
- Mas toda consequência de segurança falsificável, baseada em jogos, do conhecimento zero clássico pode ser alcançada nesse cenário.

"Falsificável" importa. Significa que uma falha de segurança pode ser testada rodando um adversário em um jogo. Muitas definições criptográficas de segurança têm essa forma: o adversário consegue distinguir duas cifragens, inverter uma função, recuperar uma testemunha ou vencer algum experimento especificado? O teorema dá um provador para cada propriedade falsificável, uma por vez. Um único provador que tenha *todas* as propriedades falsificáveis de uma vez provavelmente é impossível — o velho ataque de reutilização ("Bob pode mostrar a prova a outros") é ele próprio uma propriedade falsificável, e aqui ela de fato falha. A proposta do artigo é que um único provador pode plausivelmente cobrir todas as propriedades falsificáveis *naturais* — aquelas que de fato ocorrem na prática criptográfica —, mas essa parte é um teorema condicional apoiado em uma noção informal de "natural", mais uma conjectura explícita. A garantia mira falhas observáveis, não todo significado filosófico ou baseado em simulação de segredo.

Vale nomear um corolário concreto: a construção produz as primeiras provas não interativas de *ocultação de testemunha* com provador uniforme — "uma prova de um quebra-cabeça não ajuda você a encontrar sua solução", sem interação e sem configuração —, um objeto de aparência modesta que resistia à construção havia décadas.

O que isto não diz

Esta é a seção que mantém a peça honesta.

Ela **não** diz que os velhos teoremas de impossibilidade estavam errados. A construção os evita mudando a definição.

Ela **não** dá conhecimento zero clássico, comum, sem interação, sem configuração e com solidez perfeita. O artigo diz explicitamente que o provador construído não tem simulador.

Ela **não** significa que a prova não possa ser reutilizada. Uma prova de uma mensagem ainda pode ser mostrada a outra pessoa; o artigo não preserva propriedades do estilo deniability. (Conhecimento zero não interativo com configuração confiável tem a mesma limitação.)

Ela **não** significa que isto seja um protocolo prático pronto para implantação. Isto é teoria da complexidade e fundamentos criptográficos. O resultado depende de grandes suposições de complexidade de provas e criptografia, e a construção trata do que é possível em princípio.

Ela **não** transforma "Gödel" em uma primitiva mágica de segurança. A conexão com Gödel passa por sistemas de provas, sistemas de provas ótimos e análogos finitos da incompletude. A intuição utilizável não é "a incompletude protege sua senha". É: se um livro de regras não consegue provar eficientemente que um simulador é impossível, então ataques que exigiriam essa prova podem ser bloqueados no nível das definições de segurança.

Por que é interessante mesmo assim

Criptografia muitas vezes transforma dificuldade em segurança. Fatorar é difícil, então suposições no estilo RSA se tornam úteis. Problemas de reticulados são difíceis, então criptografia baseada em reticulados se torna útil. Aqui a dificuldade é mais estranha: não "difícil calcular um segredo", mas "difícil provar que certo objeto de prova não pode existir".

É por isso que o artigo parece incomum. Ele trata axiomas e livros de regras quase como recursos criptográficos. A impossibilidade usual diz que há uma tensão entre solidez e simulação. O movimento de Ilango é colocar essa tensão atrás de uma cortina teórico-probatória: o simulador está ausente, mas o sistema formal não consegue expor eficientemente essa ausência.

Para um leitor, a parte surpreendente não é que isso vá substituir os sistemas de conhecimento zero de hoje. Provavelmente não vai, pelo menos não diretamente. A parte surpreendente é que uma limitação da lógica matemática pode ser usada construtivamente: não apenas como um muro, mas como uma espécie de cobertura.

Quão sólidas são as evidências?

Este é um artigo de teorema, então "evidência" significa algo diferente de um artigo de biologia ou astronomia. A pergunta não é se um experimento replicou. A pergunta é se as definições, suposições e a cadeia de prova sustentam a alegação.

A prova é formal, e o artigo é explícito sobre suas suposições. As suposições não são casuais. Provas não interativas com indistinguibilidade de testemunha são objetos padrão em criptografia e seguem de vários pacotes estabelecidos de suposições. A conjectura de inexistência de sistema de provas ótimo é uma conjectura central em complexidade de provas. P = BPP é uma crença padrão de derandomização usada apenas para o teorema mais amplo sobre propriedades falsificáveis.

O artigo também argumenta que as suposições são o preço certo, não um andaime arbitrário: prova uma recíproca mostrando que elas são essencialmente necessárias — se construções desse tipo existem, então provas não interativas com indistinguibilidade de testemunha precisam existir, e (assumindo funções unidirecionais padrão) nenhum sistema de provas ótimo pode existir. E as suposições são "ganha-ganha": refutar qualquer uma delas seria, por si só, uma descoberta marcante em complexidade de provas, criptografia ou teoria da complexidade.

Mas, como o resultado é condicional, sua confiança também é condicional. Se essas suposições falham, a interpretação do teorema muda. E mesmo que as suposições valham, a garantia não é conhecimento zero clássico completo; é a versão relaxada, teórico-probatória, do artigo.

Então a confiança certa é alta de que o artigo estabelece um resultado de possibilidade condicional coerente; moderada de que suas suposições descrevem o mundo criptográfico em que de fato vivemos; e baixa para qualquer consequência prática imediata.

Por que isso importa

O artigo abre uma rota que deveria estar fechada.

A teoria clássica diz: conhecimento zero pleno não pode ser uma mensagem sem configuração, e não pode ser perfeitamente sólido. O artigo de Ilango diz: se pedirmos as consequências de conhecimento zero que podem ser testadas em jogos de segurança, e se permitirmos que a definição de segurança dependa do que um livro de regras consegue ou não consegue refutar eficientemente, então muito do comportamento útil pode ser recuperado — com uma mensagem, sem configuração e com solidez perfeita.

Isso não é um pequeno ajuste de definição. É uma forma diferente de pensar sobre garantias criptográficas. Em vez de perguntar apenas o que existe, pergunte o que o seu livro de regras consegue descartar. Em vez de tratar a indemonstrabilidade como um incômodo filosófico, use-a como estrutura.

O mundo prático talvez não mude amanhã. Mas o mapa conceitual muda. Há agora um sentido formal em que "ninguém consegue provar eficientemente que o segredo vazou" pode ser forte o bastante

para recuperar muitas das proteções baseadas em jogos que queríamos de “o segredo não vazou”.

É por isso que Gödel pertence ao título.

Em resumo

Provas de conhecimento zero deixam um provador convencer um verificador de que uma afirmação é verdadeira sem revelar a testemunha. Resultados clássicos de impossibilidade dizem que conhecimento zero não pode ser comprimido em uma mensagem sem configuração, e não pode ter solidez perfeita. O artigo de Rahul Ilango não refuta essas impossibilidades. Ele define uma noção mais fraca, conhecimento zero efetivo: em vez de exigir que um simulador realmente exista, exige que um sistema de provas escolhido — um livro formal de regras como ZFC — não consiga provar eficientemente que nenhum simulador existe. Sob grandes suposições da criptografia (provas não interativas com indistinguibilidade de testemunha) e da complexidade de provas (não existe sistema de provas ótimo), o artigo constrói provadores de uma mensagem para NP/SAT, sem configuração e com solidez perfeita, que alcançam as consequências falsificáveis, baseadas em jogos, de conhecimento zero propriedade por propriedade. Um único provador que cubra todas essas propriedades “naturais” é uma extensão adicional, em parte conjectural — e cobrir literalmente toda propriedade falsificável provavelmente é impossível, porque as provas continuam reutilizáveis. O resultado é teórico e condicional, não uma primitiva implantada, mas mostra uma nova forma de usar indemonstrabilidade teórico-probatória como recurso criptográfico.

Leitura crítica

O que o artigo mostra: Sob suposições declaradas, pode-se construir provadores de uma mensagem, sem configuração e perfeitamente sólidos para NP/SAT que são conhecimento zero efetivo relativo a qualquer sistema de provas escolhido, e que alcançam cada consequência falsificável, baseada em jogos, do conhecimento zero clássico.

O que é plausível, mas não provado incondicionalmente: Que as suposições necessárias de complexidade de provas e criptografia valham. Elas são suposições sérias e bem estudadas — e o artigo mostra que são essencialmente necessárias, além de suficientes —, mas continuam sendo suposições.

O que ele não mostra: Conhecimento zero clássico sem interação, sem configuração e com solidez perfeita; um sistema prático pronto para implantação; deniability ou não reutilização de provas; ou que o teorema da incompletude de Gödel, por si só, torne a criptografia segura.

Principais limitações: A garantia é uma relaxação de conhecimento zero; a versão mais ampla depende de múltiplas suposições; as alegações sobre um único provador universal permanecem em parte conjecturais; e o resultado é principalmente fundacional.

Quanta confiança um leitor geral deve ter? Alta de que este é um resultado teórico condicional importante se as definições forem aceitas. Moderada de que as suposições capturem a realidade. Baixa

para implantação prática imediata. A conclusão segura é: o artigo não quebra as impossibilidades de conhecimento zero; encontra uma nova forma teórico-probatória de contornar as partes delas que importam para muitos jogos de segurança.

Fontes

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