The Clean Paper

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

Una prueba criptográfica puede ocultar su secreto haciendo difícil probar el simulador que falta

Las pruebas de conocimiento cero permiten demostrar una afirmación sin revelar el testigo. La teoría clásica dice que no se puede tener eso, en sentido pleno, con un solo mensaje, sin configuración confiable y con solidez perfecta. El artículo de Rahul Ilango no hace desaparecer esa imposibilidad. Cambia el objetivo: en lugar de exigir que un simulador exista realmente, pide que ningún sistema de prueba elegido pueda probar eficientemente que el simulador no existe. Bajo grandes hipótesis de complejidad de pruebas y criptografía, eso basta para recuperar consecuencias falsificables, basadas en juegos, del conocimiento cero propiedad por propiedad. El punto no es una primitiva de internet lista para enchufar. Es una forma, desde la teoría de la prueba, de convertir la indemostrabilidad matemática en cobertura criptográfica.

Written by Cinzia Vaglio · Cross-checked by Vera Rubin · 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

El truco no es demostrar que el secreto está escondido

Empecemos por la versión más simple del conocimiento cero.

Alice quiere convencer a Bob de que un sudoku tiene solución. Si le envía la solución, Bob queda convencido, pero el puzzle queda arruinado. Lo que ella quiere es más extraño: una prueba de que existe una solución, sin revelar la solución.

Esa es la promesa de una prueba de conocimiento cero. El probador (Alice) convence al verificador (Bob) de que una afirmación es verdadera sin revelar nada más allá de la verdad de la afirmación.

El problema es que esa promesa cuesta algo. Una prueba matemática ordinaria tiene dos rasgos cómodos. Es **un solo mensaje**: la escribes, la entregas y te vas. Y es **perfectamente sólida**: una afirmación falsa no tiene ninguna prueba válida. Los resultados clásicos de imposibilidad dicen que el conocimiento cero debe renunciar a ambos rasgos — y no solo a los dos juntos; cada uno está prohibido por sí solo.

Primero, una prueba de conocimiento cero necesita conversación. Si Alice envía un solo mensaje, sin una configuración confiable preparada de antemano, la garantía de conocimiento cero colapsa — y esto se mantiene por mucha solidez que se esté dispuesto a sacrificar a cambio.

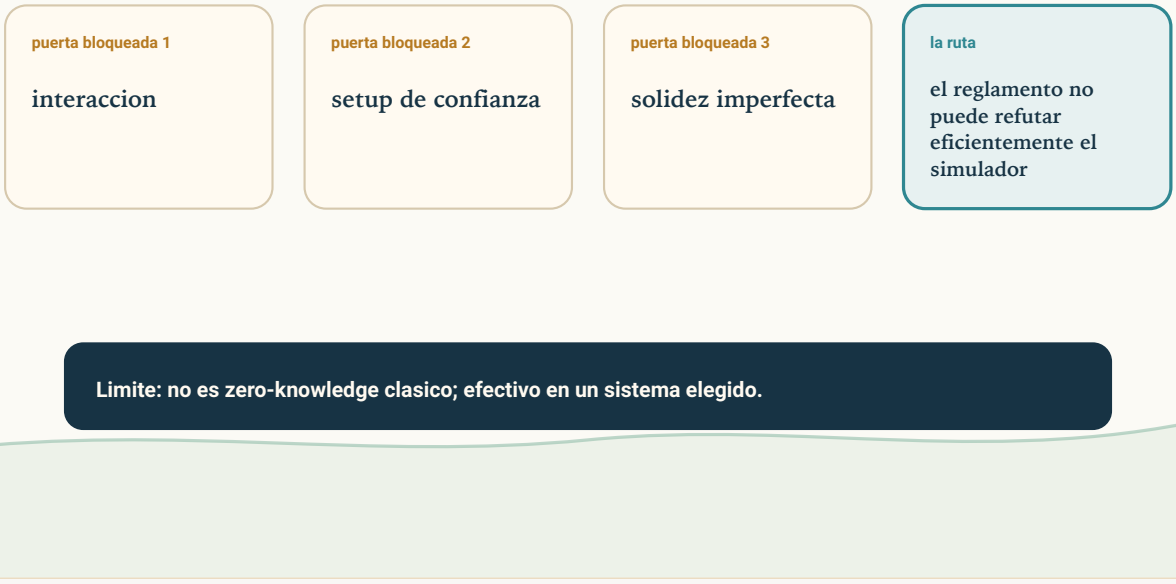
Segundo, una prueba de conocimiento cero necesita una pequeña tolerancia al error. Exigir solidez perfecta también destruye silenciosamente la interacción: un verificador que nunca puede ser engañado, sin importar qué elecciones aleatorias haga, podría fijar esas elecciones por adelantado — y una vez que el verificador es predecible, Alice puede responder a todo en un solo mensaje, que es exactamente el caso que ya se había roto.

El artículo de Rahul Ilango trata de una forma de rodear ese doble muro. No fingiendo que el muro no está ahí, ni produciendo conocimiento cero clásico en el escenario imposible. El movimiento es más sutil: debilitar lo que significa “no revela nada”, pero debilitarlo de una forma que preserve las propiedades de seguridad que los criptógrafos pueden realmente probar.

El resultado se llama **conocimiento cero efectivo**.

Una prueba sin revelar el testigo

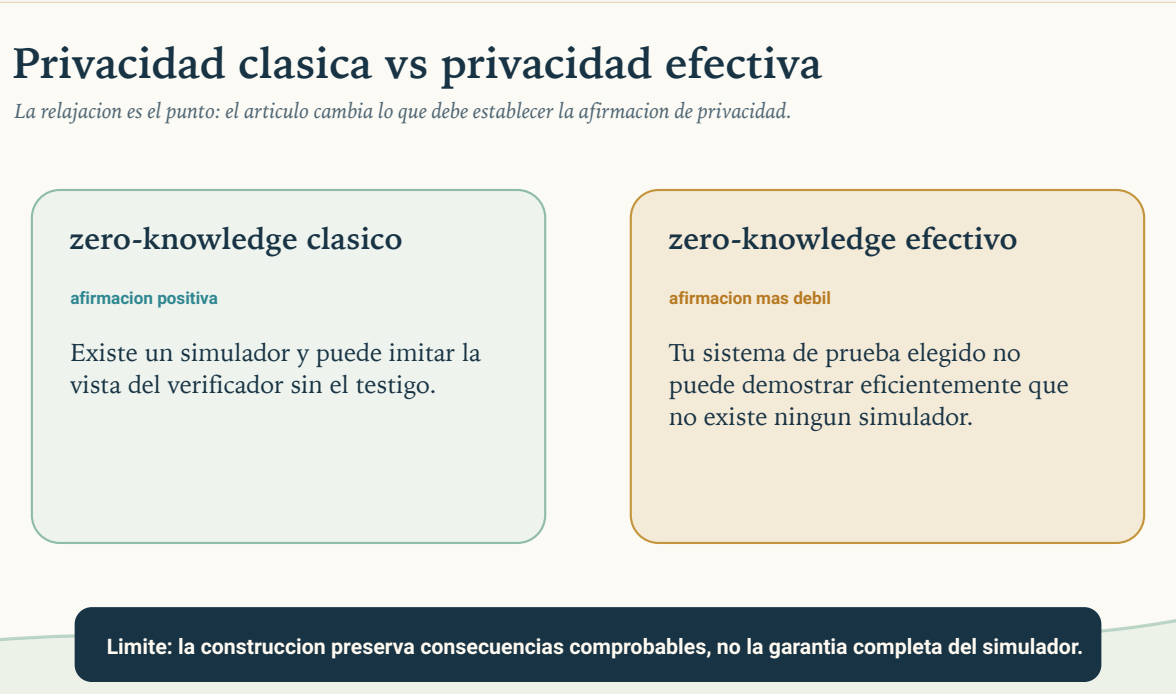
El articulo conserva la forma de una prueba ordinaria debilitando lo que la privacidad debe demostrar.



El conocimiento cero queda bloqueado en tres puertas: interacción, configuración confiable y solidez imperfecta. La construcción de llango pasa por otra: el reglamento no puede refutar eficientemente el simulador.

Privacidad clasica vs privacidad efectiva

La relajacion es el punto: el articulo cambia lo que debe establecer la afirmacion de privacidad.



El conocimiento cero clásico pregunta si existe un simulador; el «conocimiento cero efectivo» pregunta solo si

el reglamento elegido puede demostrar eficientemente que no puede existir. Esa pregunta más débil es lo que permite a la construcción conservar un solo mensaje, sin configuración y con solidez perfecta.

La prueba antigua: existe un simulador

La forma clásica de formalizar el conocimiento cero usa un ayudante ficticio llamado **simulador**.

La idea es esta: imagina a Jane, que **no** conoce el secreto de Alice. Si Jane puede generar, completamente por su cuenta, pruebas que se ven igual que las pruebas que Bob habría recibido de Alice, entonces las pruebas de Alice no le enseñaron nada nuevo a Bob. Jane ya podía fingir la experiencia sin el secreto de Alice.

Así que el conocimiento cero clásico pide un simulador real. Debe existir un algoritmo eficiente que pueda producir pruebas de aspecto falso-pero-convincente sin conocer el secreto — el *testigo*, en la jerga; para el sudoku, el testigo es simplemente la cuadrícula resuelta.

Esa definición es poderosa, pero también es exactamente donde muerde la vieja imposibilidad. Esta es la intuición. Una prueba verdaderamente no interactiva es solo una cadena. Una vez que Bob tiene esa cadena, puede mostrársela a otra persona: ha ganado la capacidad de probar la afirmación a otros, lo que ya suena como más que "nada". Los teoremas clásicos afilan esa intuición hasta las imposibilidades anteriores.

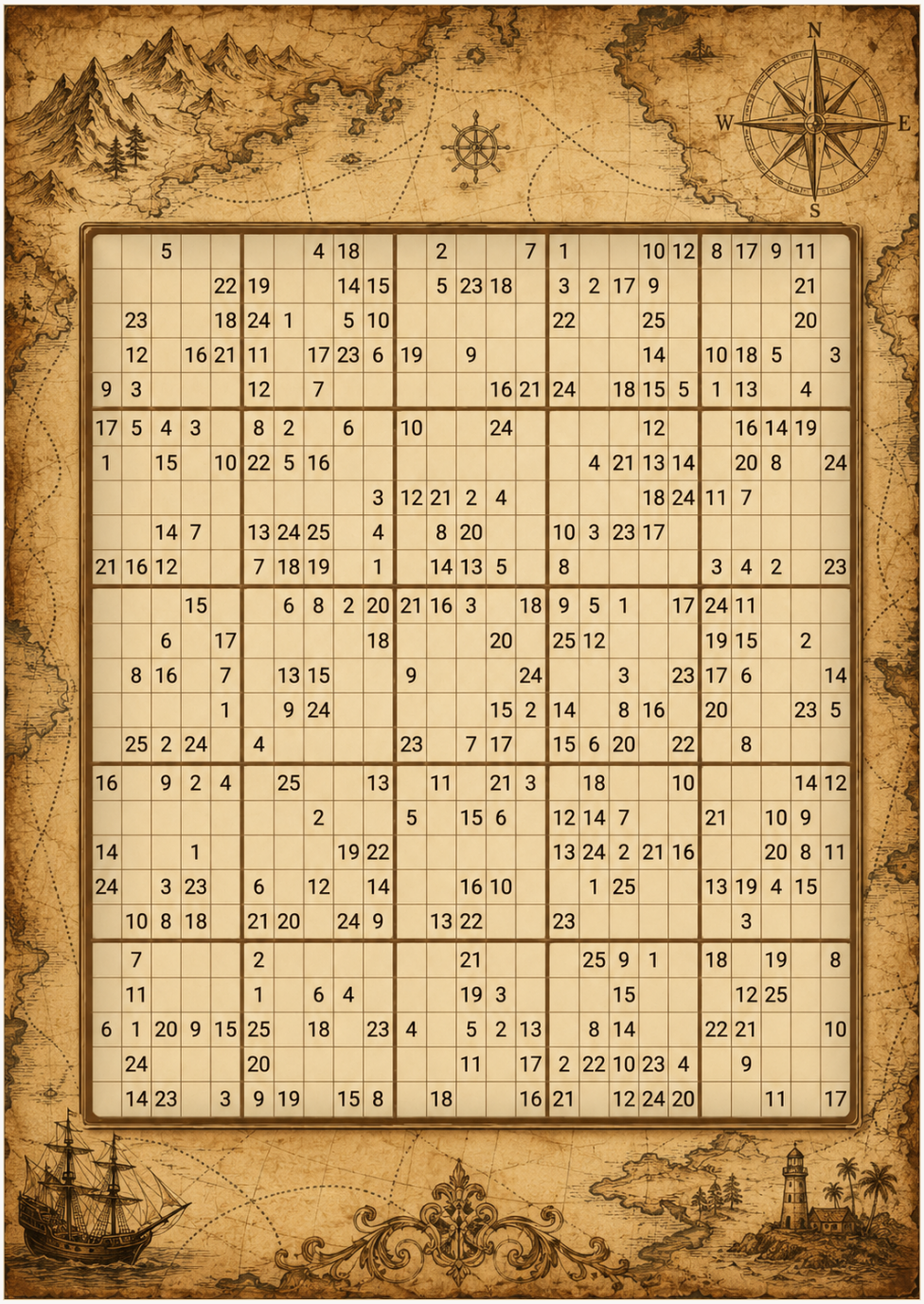
> **Las tres propiedades en las que insiste este artículo**
>
> El título del artículo nombra tres restricciones:
> **Sin interacción:** Alice envía una sola cadena de prueba. No hay protocolo de ida y vuelta.
> **Sin configuración:** Alice y Bob no dependen de una cadena común de referencia confiable ni de otra aleatoriedad pública preparada de antemano. Muchos sistemas llamados "conocimiento cero no interactivo" todavía dependen de una configuración; este artículo quiere decir configuración cero.
> **Solidez perfecta:** una afirmación falsa no tiene ninguna prueba válida. No "casi nunca aceptada"; no existe prueba válida.
> Esas tres propiedades son exactamente las que tiene la matemática escrita ordinaria — y, como se explicó arriba, el conocimiento cero clásico no puede conservarlas.

Una versión mega-Sudoku de la diferencia

Esta es una forma deliberadamente simplificada de sentir la diferencia.

No uses un sudoku ordinario de 9 por 9 para la parte seria de la analogía. Es demasiado pequeño y demasiado finito: una computadora puede simplemente resolverlo, o probar que no tiene solución.

Imagina en cambio una familia de puzzles **MegaSudoku(n)**. Escala la regla usual: elige un tamaño de bloque *n*, deja *N* = *n*^2, y construye una cuadrícula *N* por *N* dividida en bloques *n* por *n*, con *N* símbolos. El sudoku ordinario es solo el caso diminuto *n* = 3, *N* = 9: una cuadrícula 9 por 9, bloques 3 por 3 y nueve símbolos. La historia de complejidad de pruebas empieza solo cuando *n* puede crecer, y cuando la cuadrícula puede llevar gadgets adicionales que la hagan comportarse como una fórmula SAT disfrazada de sudoku. Una fórmula SAT es solo una lista de restricciones sí/no: ¿puedes asignar valores verdadero/falso a las variables de modo que cada restricción quede satisfecha?



Un sudoku de 25x25: sus reglas pueden verificarse sin revelar la cuadrícula terminada — una representación visual de una prueba que verifica una solución oculta, el testigo.

Sudoku y SAT: el mismo puzzle con dos disfraces

La afirmación de que un sudoku puede "comportarse como una fórmula SAT" no es una metáfora. La traducción va en ambas direcciones, y la dirección fácil puede escribirse completa.

De sudoku a SAT. SAT solo habla verdadero/falso, así que dale una variable booleana por cada triplete (fila, columna, valor): *x(r,c,v)* significa "la celda de la fila *r*, columna *c*, contiene el valor *v*". Un sudoku 4 por 4 (bloques 2 por 2, valores 1-4) necesita 4·4·4 = 64 variables; el clásico 9 por 9 necesita 729. Cada regla de sudoku se convierte entonces en un lote de cláusulas. (Una cláusula es un OR de variables o de sus negaciones; toda la fórmula es el AND de todas sus cláusulas.)

Cada celda contiene al menos un valor — una cláusula por celda:

x(1,1,1) ∨ x(1,1,2) ∨ x(1,1,3) ∨ x(1,1,4)

Cada celda contiene como máximo un valor — una cláusula "no ambos" para cada par de valores:

¬x(1,1,1) ∨ ¬x(1,1,2) ¬x(1,1,1) ∨ ¬x(1,1,3) … y así con los seis pares.

Cada fila contiene cada valor — para la fila 1 y el valor 3: al menos una vez,

x(1,1,3) ∨ x(1,2,3) ∨ x(1,3,3) ∨ x(1,4,3)

y como máximo una vez: ¬x(1,1,3) ∨ ¬x(1,2,3), y así para cada par de celdas de la fila.

Columnas y bloques — lotes idénticos; solo cambia el grupo de celdas. Para el bloque superior izquierdo y el valor 2:

x(1,1,2) ∨ x(1,2,2) ∨ x(2,1,2) ∨ x(2,2,2)

más las cláusulas por pares de "no ambos".

Las pistas impresas — la parte más simple: cada pista es una cláusula con una sola variable. Un 3 impreso en la esquina superior izquierda se convierte en la cláusula

x(1,1,3)

El AND de todo esto es satisfacible exactamente cuando el sudoku tiene solución — y una asignación satisfactoria *es* la solución: se leen los x(r,c,v) que son verdaderos y se rellena la cuadrícula. Para un 9 por 9 esto da 729 variables y unos pocos miles de cláusulas, que un solucionador SAT moderno despacha en milisegundos. Observa la cláusula de pista x(1,1,3): dice "esta celda es exactamente 3", no "estas celdas son todas distintas" — la misma asimetría que obligará al truco adicional para las celdas de pista en la nota de protocolo más abajo.

De SAT a sudoku. El artículo necesita la dirección opuesta, más difícil: dada una fórmula SAT *arbitraria*, construir un mega-Sudoku que tenga solución exactamente cuando la fórmula la tiene. Las reglas nativas del sudoku solo pueden decir "estas celdas son todas distintas", así que las restricciones lógicas arbitrarias deben *construirse* — y eso es precisamente lo que son los gadgets. Un gadget es un pequeño grupo prefabricado de celdas, uno por cláusula de la fórmula, en el que celdas designadas desempeñan el papel de variables (el símbolo que contienen codifica verdadero o falso) y las restricciones internas del grupo están

diseñadas para que sus únicos rellenados legales correspondan a asignaciones que satisfacen esa cláusula. Es la artesanía estándar de las pruebas de NP-completitud; para el sudoku generalizado, Yato y Seta la hicieron en 2003.
Juntas, las dos direcciones dicen que el sudoku N por N y SAT son el mismo problema con trajes distintos. Eso es lo que autoriza a este artículo — y al paper — a contar una historia sobre todo NP usando cuadrículas y símbolos.

El testigo sigue siendo fácil de imaginar. Alice conoce un rellenado completo y válido del mega-Sudoku. Bob quiere convencerse de que tal rellenado existe, pero Alice no quiere revelarlo. Si envía todo el rellenado, Bob queda convencido, pero el secreto desaparece.

En la versión clásica de conocimiento cero, Alice y Bob interactúan. Un viejo modelo mental usa fichas cubiertas. Alice oculta la cuadrícula resuelta, renombra en secreto los símbolos antes de cada ronda y deja que Bob inspeccione una restricción local elegida al azar: una fila, una columna, una caja o un gadget. Si las celdas abiertas muestran símbolos todos distintos, Bob gana confianza. Luego todo se cubre de nuevo y los símbolos se renombran otra vez. (Una complicación: las pistas dadas del puzzle necesitan un truco adicional, porque renombrar los símbolos también las oculta. La nota siguiente explica cómo lo resuelven los protocolos clásicos; la imagen-juguete basta para lo que sigue.)

Cómo tratan realmente los protocolos clásicos las celdas con pistas

El truco de renombrar tiene un punto ciego. Las reglas de fila, columna y caja dicen todas "estas celdas son todas distintas", y *todas distintas* sobrevive a cualquier renombrado de los símbolos. Pero una pista dice "esta celda contiene exactamente 5", y tras renombrar Bob solo ve $\sigma(5)$ — algún símbolo enmascarado — sin conocer el renombrado σ. No puede comprobar nada. Sin arreglarlo, Alice podría probar que existe *alguna* cuadrícula válida mientras ignora por completo las pistas impresas, lo que no prueba nada sobre *este* puzzle. La literatura clásica tiene dos reparaciones estándar.
La paleta. Añadir una fila extra de N celdas a la cuadrícula oculta — una paleta que Alice llena con los símbolos 1…N en un orden público fijo, y luego renombra junto con todo lo demás, de modo que contiene $\sigma(1)…\sigma(N)$. El desafío aleatorio de Bob tiene entonces una opción más. Además de elegir una fila, columna, caja o gadget para abrir, puede elegir **la paleta más una celda de pista**. Alice descubre ambas; la paleta revela el renombrado de esa ronda, y Bob comprueba que la celda de pista muestra exactamente la versión renombrada de la pista impresa. Esto sigue siendo conocimiento cero porque Bob solo aprende σ — que se elige de nuevo cada ronda y no vale nada por sí mismo — y el valor de una celda que ya conocía por el puzzle. No se filtra nada de las celdas secretas, y un simulador puede fingir la vista eligiendo un σ aleatorio. Es sólido porque una Alice tramposa es atrapada con probabilidad fija por ronda, y las rondas se repiten hasta que la duda es despreciable.
Compilar las pistas fuera. Una variante más estructural elimina el desafío especial en lugar de añadirlo. En vez de *verificar* el valor de la pista, lo fuerza con restricciones de diferencia: enlazar la celda de pista con cada celda de la paleta salvo la que porta su propio valor — "distinta

de $\sigma(1)$, distinta de $\sigma(2)$, …, distinta de todo salvo $\sigma(5)$". El único símbolo que la celda puede tener legalmente es el de la pista. Cada restricción vuelve a ser del tipo "estos dos difieren" — invariante bajo renombrado, comprobable exactamente como una fila. Es la misma maniobra usada para vértices precoloreados en el protocolo clásico de coloreo de grafos, y es el espíritu de la palabra *gadgets* arriba: en la imagen MegaSudoku-como-SAT, las pistas se compilan en gadgets de desigualdad como cualquier otra restricción.

El protocolo físico. El protocolo con cartas reales para sudoku (Gradwohl, Naor, Pinkas y Rothblum, 2007) no usa renombrado alguno y resuelve las pistas antes de que empiece el ocultamiento. Para cada celda, Alice coloca tres cartas idénticas con el valor de la celda — boca abajo para celdas secretas, pero **boca arriba para celdas de pista**, de modo que Bob ve con sus propios ojos que las pistas se respetan antes de que las cartas se volteen. Luego una carta de cada celda va al paquete de su fila, una al de su columna y una al de su caja; cada paquete se baraja y se revela, y Bob comprueba que contiene todos los N símbolos. El barajado destruye la información de posición (eso es el conocimiento cero), pero las pistas ya habían quedado fijadas al repartir.

En cualquier caso, la lección es la misma a la que este artículo vuelve una y otra vez: un protocolo de conocimiento cero es una contabilidad cuidadosa de *qué* hechos sobreviven al ocultamiento. Renombrar preserva "todos distintos" y borra "igual a 5" — así que "igual a 5" debe reintroducirse por otros medios.

Ese no es el protocolo del artículo. Es el modelo mental del conocimiento cero clásico:

- Alice y Bob van y vienen.
- Bob elige comprobaciones aleatorias.
- Alice revela solo consistencia local, no toda la solución.
- La prueba de privacidad funciona mostrando que la vista de Bob podría haberse generado sin la solución secreta de Alice.

Así que el conocimiento cero clásico se construye alrededor de un hecho positivo:

> Un simulador realmente existe.

Ahora quita las partes cómodas. Alice envía una sola cadena de prueba y se va. No hay configuración confiable, no hay cadena aleatoria compartida preparada de antemano, y Bob no debe aceptar nunca un puzzle falso. Ese es el escenario en el que el conocimiento cero clásico no puede sobrevivir.

Hace falta un personaje más antes del truco. Fija un **sistema formal**: un sistema de prueba formal, en el sentido lógico — un conjunto fijo de axiomas más reglas mecánicas para comprobar pruebas matemáticas escritas. ZFC, los axiomas estándar de las matemáticas, es el ejemplo canónico. Todo lo que sigue se formula relativo a un sistema elegido de antemano, y la elección es flexible: la construcción funciona para cualquier sistema que fijes, incluido ZFC.

(Una nota sobre palabras, tomada del propio artículo: aquí "sistema de prueba" siempre significa este sistema formal — el sistema que comprueba pruebas matemáticas — nunca los mensajes que Alice envía. La maquinaria de Alice y Bob se llama "el probador y el verificador".)

La versión estilo Gödel conserva la historia del mega-Sudoku pero cambia la prueba.

Elige un segundo sistema de restricciones del mismo tamaño mostrado, llámalo **D**. Para la historia, S y D son dos MegaSudoku(n) en el mismo formato. Entre bastidores, D puede haber empezado como una fórmula lógica difícil de otro tamaño; si hace falta, puede rellenarse con restricciones ficticias inocuas para que encaje en la misma cuadrícula. D se construye a partir de una fórmula lógica que es realmente **insatisfacible**: no hay asignación posible de valores que haga verdaderas todas sus restricciones, igual que un puzzle roto no tiene rellenado legal. Un ejemplo-juguete sería una fórmula que exige a la vez "X es verdadero" y "X es falso". Así que D no tiene rellenado válido.

Pero D no debe ser un puzzle roto *fácil de exponer*. El ejemplo-juguete anterior falla: cualquier sistema formal refuta "X y no-X" en una línea. D tiene que ser falso de una manera que el sistema elegido no pueda certificar con un argumento corto. Si el sistema pudiera refutar D con una prueba corta, la historia siguiente colapsaría: la ruta alternativa que podría haber producido pruebas sin el secreto de Alice podría descartarse formalmente, y con ella la garantía de privacidad. Así que D se elige de una familia que el sistema fijado no puede refutar eficientemente: no hay una prueba corta, dentro de ese sistema, de que D no tiene solución.

La prueba de un solo mensaje de Alice trata entonces de una afirmación de uno-u-otro:

> o bien el verdadero mega-Sudoku S tiene solución, o bien el señuelo D tiene solución.

Este es el vínculo lógico. D **no** se genera de una manera mágica que haga verdadero a S. La prueba no argumenta "D no tiene solución, por tanto S tiene solución". Prueba la disyunción **S o D**. La solidez perfecta dice que una disyunción falsa no puede tener una prueba válida. Como D es falso en realidad — no tiene solución — la única forma de que la disyunción sea verdadera es que S lo sea. Así que si se acepta la prueba, S debe tener solución. El señuelo no puede convertir en verdadero un S falso.

Pero para la parte estilo conocimiento cero, pregunta qué pasaría si D *sí* tuviera solución. Esa solución señuelo actuaría como testigo alternativo. Permitiría producir pruebas sin conocer la verdadera solución del mega-Sudoku de Alice — un simulador, en otras palabras. En realidad D no tiene solución, así que esta ruta de simulación está cerrada. El punto es que el sistema formal no puede probar eficientemente que está cerrada.

D tiene, por tanto, dos trabajos. Para la **solidez**, D es falso, así que una prueba válida de "S o D" fuerza S. Para el **conocimiento cero efectivo**, D es difícil de refutar, así que el sistema formal no puede descartar rápidamente la ruta señuelo que habría hecho posible la simulación.

Así que la prueba de seguridad ya no es:

> ¿Podemos probar que un simulador realmente existe?

Pasa a ser:

> ¿Puede tu sistema formal probar eficientemente que el simulador es imposible?

Si la respuesta es no, sigue algo sorprendentemente fuerte: toda garantía de seguridad que (a) pueda observarse ejecutando una prueba y (b) se siga demostrablemente — dentro de ese sistema formal — de la existencia de un simulador, se cumple de hecho. Un ataque exitoso contra cualquiera de ellas equivaldría a la refutación corta que falta, y esa refutación corta no existe. Esa es la parte "efectiva" del conocimiento cero efectivo.

Así que el contraste de aula es:

Conocimiento cero clásico: las pruebas son seguras porque existe un simulador.

Conocimiento cero efectivo estilo Gödel: las pruebas se tratan como seguras para pruebas de seguridad observables porque el sistema formal no puede probar eficientemente que el simulador es imposible.

La segunda afirmación es más débil. También es la razón por la que el artículo puede conservar los tres rasgos que rompieron la versión clásica: un mensaje, sin configuración y solidez perfecta.

La nueva prueba: no puedes demostrar que el simulador falta

La relajación de Ilango cambia la pregunta.

El conocimiento cero clásico pregunta:

> ¿Existe un simulador?

El conocimiento cero efectivo pregunta algo más débil:

> ¿Puede tu sistema formal elegido probar eficientemente que no existe ningún simulador?

Suena como una evasiva técnica, pero es la idea central. La construcción vive en un estado extraño: un simulador no existe realmente — el artículo es explícito sobre esto — pero el sistema formal que fijaste no puede probar eficientemente que no existe. Si toda mala consecuencia que te importa requeriría tal refutación, el sistema aún se comporta como conocimiento cero para esas consecuencias.

Aquí entra Gödel. No como decoración, y no como "Gödel hace segura la criptografía". La conexión pertenece a la teoría de la prueba. Un sistema formal se llama **óptimo** si es, en un sentido preciso, el mejor posible: cada vez que cualquier sistema formal puede refutar una fórmula del tipo relevante con una prueba corta, el sistema óptimo también puede hacerlo, con una prueba a lo sumo polinómicamente más larga. Krajíček y Pudlák conjeturaron en 1989 que **no existe ningún sistema de prueba óptimo**: sea cual sea el sistema que fijes, algún otro sistema prueba alguna familia de afirmaciones verdaderas de forma mucho más sucinta. Es una de las conjeturas centrales de la complejidad de pruebas, y es el primo finito y de complejidad del teorema de incompletitud de Gödel: algunas afirmaciones verdaderas no tienen prueba corta en el sistema que fijaste — no porque sean

impronunciables en principio, sino porque todo sistema fijo deja algunas verdades cortas sin pruebas cortas.

El artículo supone esta conjetura (en una forma "infinitamente a menudo" algo más fuerte, estándar cuando las conjeturas se usan criptográficamente). El beneficio, por un teorema de Krajíček y Pudlák, es concreto: para todo sistema formal hay una secuencia de fórmulas genuinamente insatisfacibles que el sistema no puede refutar con pruebas cortas — y, crucialmente, que un algoritmo eficiente puede *generar*. Esa última propiedad, la uniformidad, convierte toda la idea de una afirmación de existencia en un algoritmo real que Alice puede ejecutar: sus señuelos D salen de una línea de montaje, no de la nada.

El movimiento criptográfico es poner a trabajar esa falta de poder de prueba.

Qué hace la construcción

Esta es la construcción del artículo, reducida a su forma.

Fija un sistema formal — ZFC, digamos. Bajo la hipótesis de complejidad de pruebas, existe una secuencia eficientemente generable de fórmulas realmente insatisfacibles, pero el sistema formal no tiene una prueba corta de que sean insatisfacibles.

Ahora construye una prueba de un mensaje con esta forma:

> o bien la afirmación real es satisfacible, o bien esta fórmula especial difícil es satisfacible.

La fórmula especial difícil no es satisfacible. Así que si la maquinaria de prueba subyacente es perfectamente sólida, aceptar el mensaje sigue significando que la afirmación real es verdadera. Eso da solidez perfecta.

Pero para la seguridad tipo conocimiento cero, imagina que la fórmula especial difícil *fuera* satisfacible. Entonces su testigo podría usarse para simular pruebas sin conocer el testigo real. La fórmula no es satisfacible en realidad — pero el sistema formal no puede probarlo eficientemente. Así que no puede probar eficientemente que el simulador es imposible.

Esa es la bisagra. El sistema no esconde el secreto produciendo un simulador clásico. Esconde el secreto, para una gran clase de pruebas de seguridad observables, detrás de la incapacidad del sistema formal para certificar que el simulador está ausente.

Qué afirma el artículo

El teorema principal llega por capas. El resultado central es este:

Bajo una hipótesis criptográfica estándar — la existencia de **pruebas no interactivas de indistin-**

guibilidad de testigo, objetos bien estudiados que se siguen de varios paquetes de hipótesis establecidos — y bajo la conjetura de complejidad de pruebas de que **no existe ningún sistema de prueba óptimo (infinitamente a menudo)**, el artículo construye, para toda elección de sistema formal, un probador y un verificador de un mensaje para NP/SAT con **solidez perfecta** y sin configuración, que son efectivamente de conocimiento cero respecto de ese sistema. (NP/SAT es el "denominador común más difícil" estándar de los problemas tipo puzzle; el mega-Sudoku es uno de sus disfraces.)

Para la afirmación más amplia sobre preservar propiedades de seguridad falsificables, el artículo añade una hipótesis estándar más, la creencia de desaleatorización **P = BPP** (aproximadamente: la aleatoriedad no da a los algoritmos poder esencial adicional).

Traducido fuera del lenguaje de teoremas:

- La prueba es un mensaje.
- No hay configuración confiable.
- Las afirmaciones falsas no pueden probarse.
- El probador no es conocimiento cero clásico — no tiene simulador.
- Pero toda consecuencia falsificable, basada en juegos, del conocimiento cero clásico puede lograrse en este entorno.

"Falsificable" importa. Significa que una falla de seguridad puede ponerse a prueba ejecutando un adversario en un juego. Muchas definiciones criptográficas de seguridad tienen esta forma: ¿puede el adversario distinguir dos cifrados, invertir una función, recuperar un testigo o ganar algún experimento especificado? El teorema da un probador para cada propiedad falsificable, una a la vez. Un único probador que disfrute *todas* las propiedades falsificables a la vez probablemente es imposible — el viejo ataque de reutilización ("Bob puede mostrar la prueba a otros") es en sí mismo una propiedad falsificable, y aquí falla de verdad. La propuesta del artículo es que un único probador puede plausiblemente cubrir todas las propiedades falsificables *naturales* — las que aparecen realmente en la práctica criptográfica — pero esa parte es un teorema condicional que descansa en una noción informal de "natural", más una conjetura explícita. La garantía apunta a fallas observables, no a cada significado filosófico o basado en simulación del secreto.

Vale la pena nombrar un corolario concreto: la construcción produce las primeras pruebas no interactivas *witness hiding* con un probador uniforme — "una prueba de un puzzle no te ayuda a encontrar su solución", sin interacción y sin configuración — un objeto de sonido modesto que se había resistido durante décadas.

Qué no dice esto

No dice que los viejos teoremas de imposibilidad estuvieran equivocados. La construcción los evita cambiando la definición.

No da conocimiento cero ordinario, clásico, sin interacción, sin configuración y con solidez perfecta. El artículo dice explícitamente que el probador construido no tiene simulador.

No significa que la prueba no pueda reutilizarse. Una prueba de un mensaje todavía puede mostrarse a otra persona; el artículo no preserva propiedades de tipo denegabilidad. (El conocimiento cero no interactivo con configuración confiable tiene la misma limitación.)

No significa que esto sea un protocolo práctico listo para desplegar. Es teoría de la complejidad y fundamentos criptográficos. El resultado depende de grandes hipótesis de complejidad de pruebas y criptografía, y la construcción trata de lo que es posible en principio.

No convierte "Gödel" en una primitiva mágica de seguridad. La conexión con Gödel pasa por sistemas de prueba, sistemas de prueba óptimos y análogos finitos de la incompletitud. La intuición útil no es "la incompletitud protege tu contraseña". Es: si un sistema formal no puede probar eficientemente que un simulador es imposible, entonces los ataques que requerirían esa prueba pueden bloquearse al nivel de las definiciones de seguridad.

Por qué sigue siendo interesante

La criptografía convierte a menudo la dificultad en seguridad. Factorizar es difícil, así que las hipótesis tipo RSA se vuelven útiles. Los problemas de retículos son difíciles, así que la criptografía de retículos se vuelve útil. Aquí la dificultad es más extraña: no "difícil de calcular un secreto", sino "difícil de probar que cierto objeto de prueba no puede existir".

Por eso el artículo se siente inusual. Trata axiomas y sistemas formales casi como recursos criptográficos. La imposibilidad habitual dice que hay una tensión entre solidez y simulación. El movimiento de Ilango es poner esa tensión detrás de una cortina de teoría de la prueba: el simulador está ausente, pero el sistema formal no puede exponer eficientemente esa ausencia.

Para un lector, lo sorprendente no es que esto vaya a sustituir los sistemas de conocimiento cero actuales. Probablemente no lo hará, al menos no directamente. Lo sorprendente es que una limitación de la lógica matemática pueda usarse constructivamente: no solo como muro, sino como una especie de cobertura.

Qué tan fuerte es la evidencia

Este es un artículo de teoremas, así que "evidencia" significa algo distinto que en un artículo de biología o astronomía. La pregunta no es si un experimento se replicó. La pregunta es si las definiciones, hipótesis y cadena de prueba sostienen la afirmación.

La prueba es formal, y el artículo es explícito sobre sus hipótesis. Las hipótesis no son casuales. Las pruebas no interactivas de indistinguibilidad de testigo son objetos estándar en criptografía y se siguen de varios paquetes de hipótesis establecidos. La conjetura de no existencia de sistemas de prueba óptimos es una conjetura central en complejidad de pruebas. P = BPP es una creencia estándar de desaleatorización usada solo para el teorema más amplio de propiedades falsificables.

El artículo también argumenta que las hipótesis son el precio correcto, no un andamiaje arbitrario: prueba una recíproca que muestra que son esencialmente necesarias — si existen construcciones como esta, entonces deben existir pruebas no interactivas de indistinguibilidad de testigo y (con funciones unidireccionales estándar) no puede existir un sistema de prueba óptimo. Y las hipótesis son "ganar-ganar": refutar cualquiera de ellas sería por sí mismo un descubrimiento importante en complejidad de pruebas, criptografía o teoría de la complejidad.

Pero como el resultado es condicional, su confianza también lo es. Si esas hipótesis fallan, cambia la interpretación del teorema. E incluso si se sostienen, la garantía no es conocimiento cero clásico completo; es la versión relajada del artículo, formulada en teoría de la prueba.

Así que la confianza correcta es alta en que el artículo establece un resultado coherente de posibilidad condicional; moderada en que sus hipótesis describan el mundo criptográfico en el que vivimos; y baja para cualquier consecuencia práctica inmediata.

Por qué importa

El artículo abre una ruta que se suponía cerrada.

La teoría clásica dice: el conocimiento cero completo no puede ser un mensaje sin configuración, y no puede ser perfectamente sólido. El artículo de Ilango dice: si pedimos las consecuencias del conocimiento cero que pueden probarse en juegos de seguridad, y si permitimos que la definición de seguridad dependa de lo que un sistema formal puede o no puede refutar eficientemente, entonces gran parte del comportamiento útil puede recuperarse — con un mensaje, sin configuración y con solidez perfecta.

No es un pequeño ajuste definicional. Es una forma distinta de pensar las garantías criptográficas. En lugar de preguntar solo qué existe, preguntar qué puede descartar tu sistema formal. En lugar de tratar la indemostrabilidad como una molestia filosófica, usarla como estructura.

El mundo práctico quizá no cambie mañana. Pero el mapa conceptual sí. Ahora existe un sentido formal en el que "nadie puede probar eficientemente que el secreto se filtró" puede ser lo bastante fuerte para recuperar muchas de las protecciones basadas en juegos que queríamos de "el secreto no se filtró".

Por eso Gödel pertenece al título.

Resumen limpio

Las pruebas de conocimiento cero permiten a un probador convencer a un verificador de que una afirmación es verdadera sin revelar el testigo. Los resultados clásicos de imposibilidad dicen que el conocimiento cero no puede comprimirse en un mensaje sin configuración, y no puede tener solidez perfecta. El artículo de Rahul Ilango no refuta esas imposibilidades. Define una noción más

débil, conocimiento cero efectivo: en lugar de exigir que un simulador exista realmente, exige que un sistema de prueba elegido — un sistema formal como ZFC — no pueda probar eficientemente que no existe ningún simulador. Bajo grandes hipótesis de criptografía (pruebas no interactivas de indistinguibilidad de testigo) y complejidad de pruebas (no existe sistema de prueba óptimo), el artículo construye probadores de un mensaje para NP/SAT sin configuración y con solidez perfecta que logran las consecuencias falsificables y basadas en juegos del conocimiento cero, propiedad por propiedad. Un único probador que cubra todas esas propiedades "naturales" es una extensión adicional, en parte conjetural — y cubrir literalmente toda propiedad falsificable probablemente sea imposible, porque las pruebas siguen siendo reutilizables. El resultado es teórico y condicional, no una primitiva desplegada, pero muestra una nueva forma de usar la indemostrabilidad en teoría de la prueba como recurso criptográfico.

No-BS check

Lo que muestra el artículo: Bajo las hipótesis declaradas, se pueden construir probadores de un mensaje, sin configuración y perfectamente sólidos para NP/SAT, que son efectivamente de conocimiento cero relativos a cualquier sistema de prueba elegido, y que logran cada consecuencia falsificable basada en juegos del conocimiento cero clásico.

Lo que es plausible pero no está probado incondicionalmente: Que se sostengan las hipótesis necesarias de complejidad de pruebas y criptografía. Son hipótesis serias y muy estudiadas — y el artículo muestra que son esencialmente necesarias además de suficientes — pero siguen siendo hipótesis.

Lo que no muestra: Conocimiento cero clásico sin interacción, sin configuración y con solidez perfecta; un sistema práctico listo para despliegue; denegabilidad o no reutilización de pruebas; ni que el teorema de incompletitud de Gödel por sí solo asegure la criptografía.

Principales limitaciones: La garantía es una relajación del conocimiento cero; la versión más amplia depende de varias hipótesis; las afirmaciones de probador universal único siguen siendo en parte conjeturales; y el resultado es principalmente fundacional.

Cuánta confianza debería tener un lector general: Alta en que este es un resultado teórico condicional importante si se aceptan las definiciones. Moderada en que las hipótesis capturen la realidad. Baja para despliegue práctico inmediato. La conclusión segura es: el artículo no rompe las imposibilidades del conocimiento cero; encuentra una nueva forma desde la teoría de la prueba de rodear las partes de ellas que importan para muchos juegos de seguridad.

Sources

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