The Clean Paper

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

Une preuve cryptographique peut cacher son secret en rendant le simulateur manquant difficile à prouver

Les preuves à divulgation nulle de connaissance permettent de prouver un énoncé sans révéler le témoin. La théorie classique dit qu'on ne peut pas obtenir cela, au sens plein, avec un seul message, sans setup de confiance et avec une solidité parfaite. L'article de Rahul Ilango ne fait pas disparaître cette impossibilité. Il change la cible : au lieu d'exiger qu'un simulateur existe vraiment, il demande qu'aucun système de preuve choisi ne puisse prouver efficacement que le simulateur n'existe pas. Sous de grandes hypothèses de complexité des preuves et de cryptographie, cela suffit à récupérer des conséquences falsifiables, fondées sur des jeux, du zéro connaissance, propriété par propriété. Le point n'est pas une primitive internet prête à brancher. C'est une façon, issue de la théorie des preuves, de transformer l'improuvabilité mathématique en couverture cryptographique.

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

Le truc n'est pas de prouver que le secret est caché

Commençons par la version la plus simple du zéro connaissance.

Alice veut convaincre Bob qu'une grille de Sudoku a une solution. Si elle envoie la solution, Bob est convaincu, mais le puzzle est gâché. Ce qu'elle veut est plus étrange : une preuve qu'une solution existe, sans révéler la solution.

C'est la promesse d'une preuve à divulgation nulle de connaissance. Le prouveur (Alice) convainc le vérificateur (Bob) qu'un énoncé est vrai tout en ne révélant rien au-delà de la vérité de l'énoncé.

Le problème est que cette promesse coûte quelque chose. Une preuve mathématique ordinaire a deux traits confortables. Elle est **un seul message** : on l'écrit, on la remet, et l'on s'en va. Et elle est **parfaitement saine** : un énoncé faux n'a aucune preuve valide. Les résultats classiques d'impossibilité disent que le zéro connaissance doit abandonner ces deux traits — et pas seulement les deux ensemble ; chacun est interdit à lui seul.

D'abord, une preuve à divulgation nulle de connaissance a besoin d'une conversation. Si Alice envoie un seul message, sans dispositif de confiance arrangé à l'avance, la garantie de zéro connaissance s'effondre — et cela reste vrai quelle que soit la quantité de solidité que l'on accepte d'échanger.

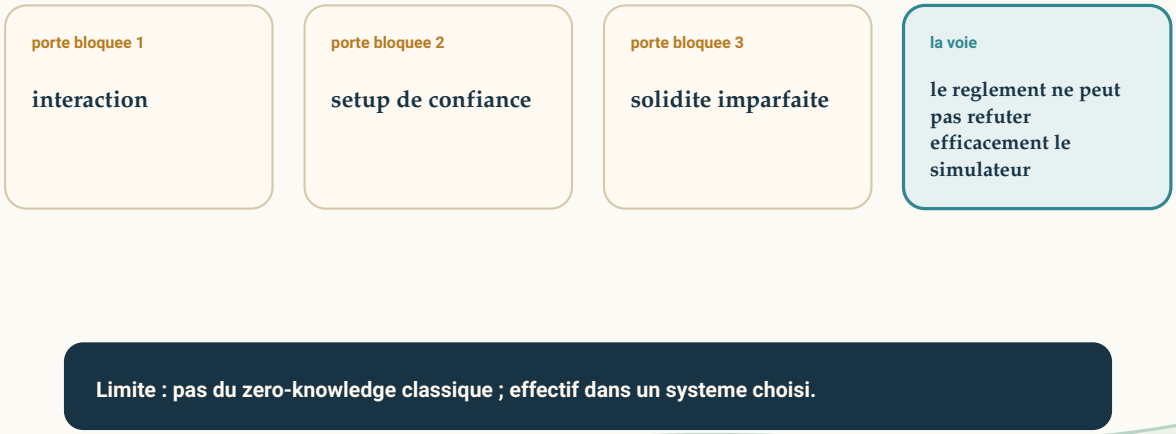
Ensuite, une preuve à divulgation nulle de connaissance a besoin d'une petite tolérance à l'erreur. Exiger une solidité parfaite détruit discrètement l'interaction aussi : un vérificateur qui ne peut jamais être trompé, quels que soient ses choix aléatoires, pourrait tout aussi bien fixer ces choix à l'avance — et dès que le vérificateur devient prévisible, Alice peut répondre à tout dans un seul message, ce qui est exactement le cas qui avait déjà cassé la propriété.

L'article de Rahul Ilango porte sur une façon de contourner ce double mur. Pas en prétendant que le mur n'existe pas, et pas en produisant du zéro connaissance classique dans le cadre impossible. Le mouvement est plus subtil : affaiblir ce que « ne révèle rien » signifie, mais l'affaiblir d'une manière qui préserve les propriétés de sécurité que les cryptographes peuvent réellement tester.

Le résultat s'appelle **zéro connaissance effectif**.

Une preuve sans reveler le temoin

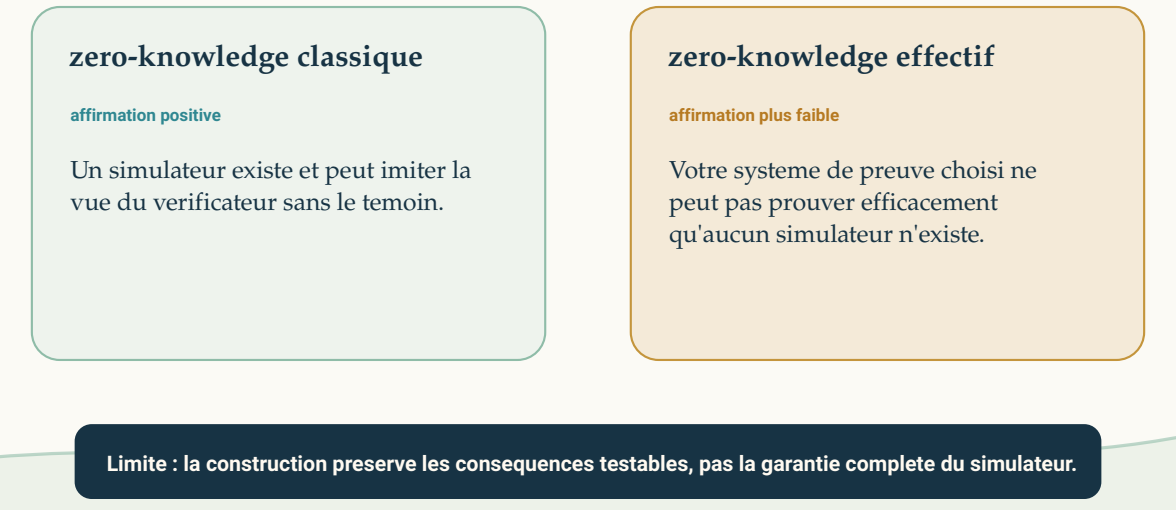
L'article garde la forme d'une preuve ordinaire en affaiblissant ce que la confidentialite doit prouver.



Le zéro connaissance est bloqué par trois portes — interaction, dispositif de confiance et solidité imparfaite. La construction d'Ilango passe par une autre : le règlement ne peut pas réfuter efficacement le simulateur.

Confidentialite classique vs confidentialite effective

La relaxation est le point central : l'article change ce que la revendication de confidentialite doit etablir.



Le zéro connaissance classique demande si un simulateur existe ; le « zéro connaissance effectif » demande seulement si votre règlement choisi peut prouver efficacement qu'il n'en existe pas. Cette question plus faible

est ce qui permet à la construction de garder un seul message, sans dispositif de confiance et avec une solidité parfaite.

L'ancien test : un simulateur existe

La manière classique de formaliser le zéro connaissance utilise un auxiliaire fictif appelé **simulateur**.

L'idée est la suivante : imaginez Jane, qui ne connaît **pas** le secret d'Alice. Si Jane peut générer, entièrement seule, des preuves qui ressemblent exactement aux preuves que Bob aurait reçues d'Alice, alors les preuves d'Alice n'ont rien appris de nouveau à Bob. Jane pouvait déjà imiter l'expérience sans le secret d'Alice.

Le zéro connaissance classique demande donc un vrai simulateur. Il doit exister un algorithme efficace capable de produire des preuves d'apparence authentique sans connaître le secret — le *témoin*, dans le jargon ; pour un Sudoku, le témoin est simplement la grille résolue.

Cette définition est puissante, mais c'est aussi précisément là que mord l'ancienne impossibilité. Voici l'intuition. Une preuve vraiment non interactive n'est qu'une chaîne de caractères. Une fois que Bob a cette chaîne, il peut la montrer à quelqu'un d'autre : il a gagné la capacité de prouver l'énoncé à d'autres, ce qui ressemble déjà à plus que « rien ». Les théorèmes classiques affûtent cette intuition pour donner les impossibilités ci-dessus.

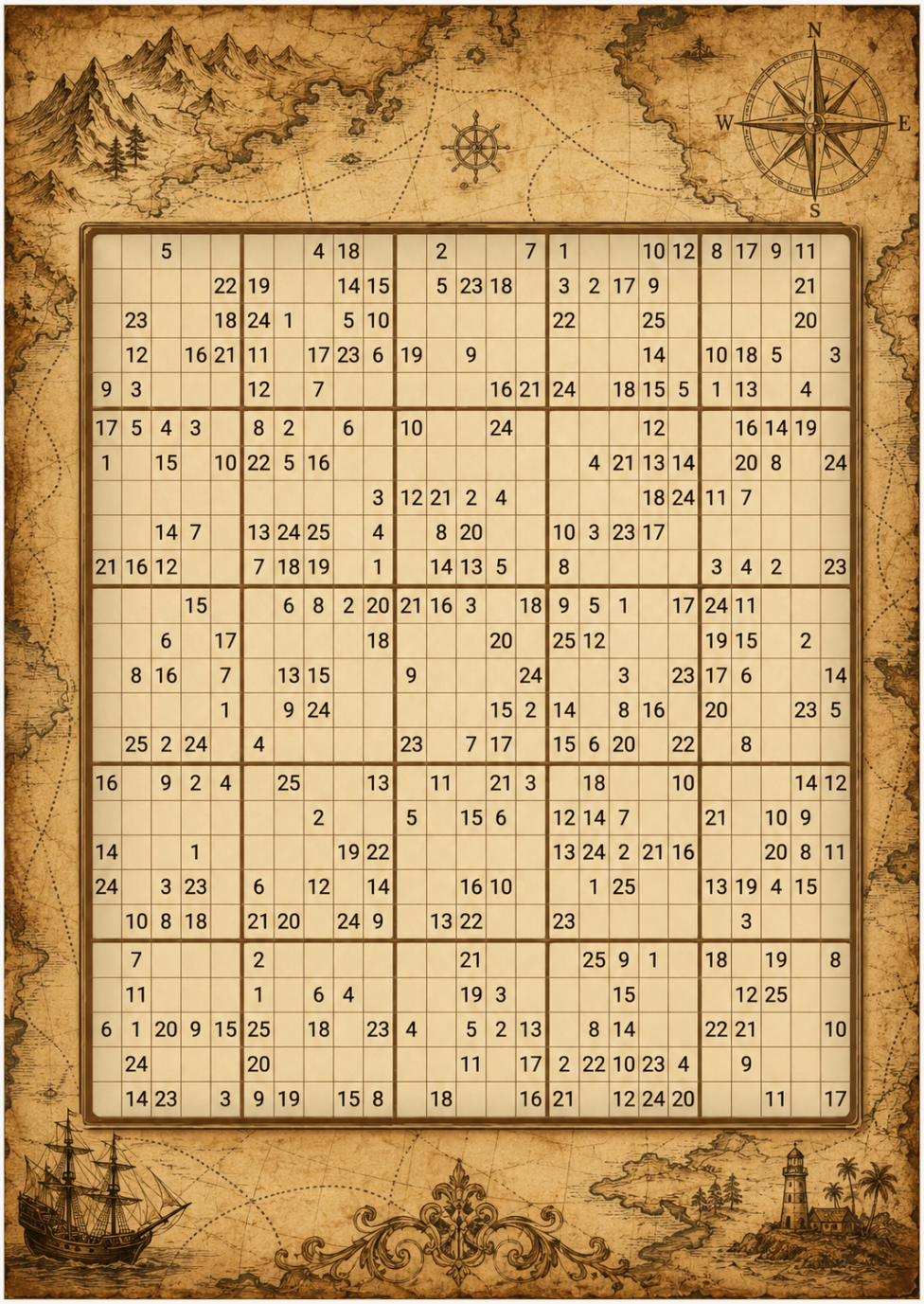
> **Les trois propriétés sur lesquelles insiste cet article**
>
> Le titre de l'article nomme trois contraintes :
> **Aucune interaction :** Alice envoie une seule chaîne de preuve. Il n'y a pas de protocole aller-retour.
> **Aucun setup :** Alice et Bob ne s'appuient pas sur une chaîne de référence commune de confiance ni sur un autre aléa public préparé à l'avance. Beaucoup de systèmes appelés « zéro connaissance non interactif » reposent encore sur un setup ; cet article veut dire zéro setup.
> **Solidité parfaite :** un énoncé faux n'a aucune preuve valide. Pas « presque jamais accepté » ; aucune preuve valide n'existe.
> Ces trois propriétés sont exactement celles des mathématiques écrites ordinaires — et, comme expliqué ci-dessus, le zéro connaissance classique ne peut pas les garder.

Une version méga-Sudoku de la différence

Voici une façon volontairement simplifiée de sentir la différence.

N'utilisez pas un Sudoku ordinaire 9 par 9 pour la partie sérieuse de l'analogie. Il est trop petit et trop fini : un ordinateur peut simplement le résoudre, ou prouver qu'il n'a pas de solution. Imaginez plutôt

une famille de puzzles **MegaSudoku(n)**. On met la règle habituelle à l'échelle : choisir une taille de bloc *n*, poser *N* = *n*^2, et construire une grille *N* par *N* divisée en blocs *n* par *n*, avec *N* symboles. Le Sudoku ordinaire n'est que le minuscule cas *n* = 3, *N* = 9 : une grille 9 par 9, des blocs 3 par 3 et neuf symboles. L'histoire de complexité de preuve ne commence que lorsque *n* peut grandir, et lorsque la grille peut porter des gadgets supplémentaires qui la font se comporter comme une formule SAT déguisée en Sudoku. Une formule SAT n'est qu'une liste de contraintes oui/non : peut-on affecter des valeurs vrai/faux aux variables de sorte que chaque contrainte soit satisfaite ?



Un Sudoku 25x25 : ses règles peuvent être vérifiées sans révéler la grille terminée — un substitut visuel pour une preuve qui vérifie une solution cachée, le témoin.

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

Sudoku et SAT : le même puzzle en deux costumes

L'affirmation qu'un Sudoku peut « se comporter comme une formule SAT » n'est pas une métaphore. La traduction fonctionne dans les deux sens, et le sens facile peut être écrit complètement.

Du Sudoku vers SAT. SAT ne parle que vrai/faux, donc donnez-lui une variable booléenne pour chaque triplet (ligne, colonne, valeur) : *x(r,c,v)* signifie « la case à la ligne *r*, colonne *c*, contient la valeur *v* ». Un Sudoku 4 par 4 (blocs 2 par 2, valeurs 1-4) a besoin de 4·4·4 = 64 variables ; le classique 9 par 9 en a besoin de 729. Chaque règle de Sudoku devient ensuite un paquet de clauses. (Une clause est un OU de variables ou de leurs négations ; toute la formule est le ET de toutes ses clauses.)

Chaque case contient au moins une valeur — une clause par case :

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

Chaque case contient au plus une valeur — une clause « pas les deux » pour chaque paire de valeurs :

$$\neg x(1,1,1) \vee \neg x(1,1,2) \quad \neg x(1,1,1) \vee \neg x(1,1,3)$$ … et ainsi de suite pour les six paires.

Chaque ligne contient chaque valeur — pour la ligne 1 et la valeur 3 : au moins une fois,

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

et au plus une fois : $\neg x(1,1,3) \vee \neg x(1,2,3)$, et ainsi de suite pour chaque paire de cases de la ligne.

Colonnes et blocs — mêmes paquets ; seul le groupe de cases change. Pour le bloc en haut à gauche et la valeur 2 :

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

plus les clauses « pas les deux » par paires.

Les indices imprimés — la partie la plus simple : chaque indice est une clause avec une seule variable. Un 3 imprimé dans le coin supérieur gauche devient la clause

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

Le ET de tout cela est satisfiable exactement quand le Sudoku a une solution — et une affectation satisfaisante *est* la solution : on lit les x(r,c,v) vrais et on remplit la grille. Pour un 9 par 9, cela donne 729 variables et quelques milliers de clauses, qu'un solveur SAT moderne traite en quelques millisecondes. Notez la clause d'indice x(1,1,3) : elle dit « cette case vaut exactement 3 », pas « ces cases sont toutes différentes » — la même asymétrie qui imposera l'astuce supplémentaire pour les cases d'indice dans la note de protocole plus bas.

De SAT vers Sudoku. L'article a besoin du sens inverse, plus difficile : à partir d'une formule SAT *arbitraire*, construire un méga-Sudoku qui a une solution exactement quand la formule en a une. Les règles natives du Sudoku ne savent dire que « ces cases sont toutes différentes », donc des contraintes logiques arbitraires doivent être *construites* — et c'est précisément ce que sont les gadgets. Un gadget est un petit groupe préfabriqué de cases, un par clause de la formule,

dans lequel certaines cases désignées jouent le rôle de variables (le symbole qu'elles portent encode vrai ou faux) et les contraintes internes du groupe sont conçues pour que ses seuls remplissages légaux correspondent à des affectations satisfaisant cette clause. C'est l'artisanat standard des preuves de NP-complétude ; pour le Sudoku généralisé, Yato et Seta l'ont réalisé en 2003.
Ensemble, les deux directions disent que le Sudoku N par N et SAT sont le même problème dans des costumes différents. C'est ce qui autorise cet article — et l'article scientifique — à raconter une histoire sur tout NP avec des grilles et des symboles.

Le témoin reste facile à imaginer. Alice connaît un remplissage complet et valide du méga-Sudoku. Bob veut être convaincu qu'un tel remplissage existe, mais Alice ne veut pas le révéler. Si elle envoie tout le remplissage, Bob est convaincu, mais le secret a disparu.

Dans la version classique du zéro connaissance, Alice et Bob interagissent. Un ancien modèle mental utilise des tuiles couvertes. Alice cache la grille résolue, renomme secrètement les symboles avant chaque tour, et laisse Bob inspecter une contrainte locale choisie au hasard : une ligne, une colonne, une boîte ou un gadget. Si les cases ouvertes montrent des symboles tous différents, Bob gagne en confiance. Puis tout est recouvert et les symboles sont fraîchement renommés. (Une difficulté : les indices donnés du puzzle ont besoin d'une astuce supplémentaire, parce que renommer les symboles les cache aussi. La note ci-dessous explique comment les protocoles classiques résolvent cela ; l'image-jouet suffit pour la suite.)

Comment les protocoles classiques traitent vraiment les cases d'indice

L'astuce de renommage a un angle mort. Les règles de ligne, colonne et boîte disent toutes « ces cases sont toutes différentes », et *toutes différentes* survit à n'importe quel renommage des symboles. Mais un indice dit « cette case contient exactement 5 », et après renommage Bob ne voit que $\sigma(5)$ — un symbole masqué — sans connaître le renommage σ. Il ne peut rien vérifier. Sans correction, Alice pourrait prouver qu'une grille valide *quelconque* existe tout en ignorant complètement les indices imprimés, ce qui ne prouve rien sur *ce* puzzle. La littérature classique a deux réparations standard.
La palette. Ajouter une ligne supplémentaire de N cases à la grille cachée — une palette qu'Alice remplit avec les symboles 1…N dans un ordre public fixé, puis renomme avec tout le reste, de sorte qu'elle contient $\sigma(1)…\sigma(N)$. Le défi aléatoire de Bob a alors une option de plus. En plus de choisir une ligne, une colonne, une boîte ou un gadget à ouvrir, il peut choisir **la palette plus une case d'indice**. Alice découvre les deux ; la palette révèle le renommage de ce tour, et Bob vérifie que la case d'indice montre exactement la version renommée de l'indice imprimé. Cela reste zéro connaissance parce que Bob n'apprend que σ — fraîchement tiré à chaque tour et inutile seul — et la valeur d'une case qu'il connaissait déjà dans le puzzle. Rien sur les cases secrètes ne fuit, et un simulateur peut imiter la vue en tirant un σ aléatoire. C'est solide parce qu'une Alice tricheuse est prise avec une probabilité fixe par tour, et les tours sont répétés jusqu'à rendre le doute négligeable.
Compiler les indices ailleurs. Une variante plus structurelle retire le défi spécial au lieu

de l'ajouter. Plutôt que de *vérifier* la valeur de l'indice, on la force avec des contraintes de différence : relier la case d'indice à chaque case de palette sauf celle qui porte sa propre valeur — « différente de $\sigma(1)$, différente de $\sigma(2)$, …, différente de tout sauf $\sigma(5)$ ». Le seul symbole que la case peut légalement porter est celui de l'indice. Chaque contrainte est de nouveau du type « ces deux choses diffèrent » — invariante par renommage, vérifiable exactement comme une ligne. C'est la même manoeuvre que pour les sommets précolorés dans le protocole classique de coloriage de graphe, et c'est l'esprit du mot *gadgets* plus haut : dans l'image MegaSudoku-comme-SAT, les indices sont compilés en gadgets d'inégalité comme toutes les autres contraintes.

Le protocole physique. Le protocole avec des cartes réelles pour Sudoku (Gradwohl, Naor, Pinkas et Rothblum, 2007) n'utilise aucun renommage et règle les indices avant même que le masquage commence. Pour chaque case, Alice pose trois cartes identiques avec la valeur de la case — face cachée pour les cases secrètes, mais **face visible pour les cases d'indice**, de sorte que Bob voie de ses propres yeux que les indices sont respectés avant que les cartes soient retournées. Puis une carte de chaque case va dans le paquet de sa ligne, une dans celui de sa colonne, une dans celui de sa boîte ; chaque paquet est mélangé et révélé, et Bob vérifie qu'il contient tous les N symboles. Le mélange détruit l'information de position (c'est le zéro connaissance), mais les indices étaient déjà fixés au moment de la donne.

Dans tous les cas, la leçon est celle à laquelle cet article revient sans cesse : un protocole zéro connaissance est une comptabilité soigneuse des faits qui survivent au masquage. Le renommage préserve « tous différents » et efface « égal à 5 » — donc « égal à 5 » doit être réintroduit par d'autres moyens.

Ce n'est pas le protocole de l'article. C'est le modèle mental du zéro connaissance classique :

- Alice et Bob échangent.
- Bob choisit des vérifications aléatoires.
- Alice ne révèle qu'une cohérence locale, pas toute la solution.
- La preuve de confidentialité fonctionne en montrant que la vue de Bob aurait pu être générée sans la solution secrète d'Alice.

Le zéro connaissance classique repose donc sur un fait positif :

> Un simulateur existe vraiment.

Retirez maintenant les parties confortables. Alice envoie une seule chaîne de preuve et s'en va. Il n'y a pas de setup de confiance, pas de chaîne aléatoire partagée préparée à l'avance, et Bob ne doit jamais accepter un puzzle faux. C'est le cadre dans lequel le zéro connaissance classique ne peut pas survivre.

Un personnage de plus est nécessaire avant l'astuce. Fixez un **système formel** : un système de preuve formel, au sens des logiciens — un ensemble fixé d'axiomes plus des règles mécaniques pour vérifier des preuves mathématiques écrites. ZFC, les axiomes standard des mathématiques, en est l'exemple canonique. Tout ce qui suit est formulé relativement à un système choisi à l'avance, et le choix est

flexible : la construction fonctionne pour tout système que vous fixez, ZFC inclus.

(Une note de vocabulaire, empruntée à l'article lui-même : ici, « système de preuve » signifie toujours ce système formel — le système qui vérifie les preuves mathématiques — jamais les messages qu'Alice envoie. La mécanique d'Alice et de Bob s'appelle « le prouveur et le vérificateur ».)

La version à la Gödel garde l'histoire du méga-Sudoku mais change la preuve.

Choisissez un second système de contraintes de même taille affichée, appelons-le **D**. Pour l'histoire, S et D sont deux MegaSudoku(n) dans le même format. En coulisse, D peut avoir commencé comme une formule logique difficile d'une autre taille ; si nécessaire, on peut le compléter avec des contraintes factices inoffensives pour qu'il tienne dans la même grille. D est construit à partir d'une formule logique qui est réellement **insatisfiable** : aucune affectation de valeurs ne peut rendre toutes ses contraintes vraies, comme un puzzle cassé n'a pas de grille complète légale. Un exemple-jouet serait une formule qui exige à la fois « X est vrai » et « X est faux ». D n'a donc pas de remplissage valide.

Mais D ne doit pas être un puzzle cassé *facile à exposer*. L'exemple-jouet ci-dessus échoue : n'importe quel système formel réfute « X et non-X » en une ligne. D doit être faux d'une manière que le système choisi ne peut pas certifier par un argument court. Si le système pouvait réfuter D par une courte preuve, l'histoire ci-dessous s'effondrerait : la route alternative qui aurait pu produire des preuves sans le secret d'Alice pourrait être formellement exclue, et avec elle la garantie de confidentialité. D est donc choisi dans une famille que le système fixé ne peut pas réfuter efficacement : il n'existe pas de courte preuve, dans ce système, que D n'a pas de solution.

La preuve en un seul message d'Alice porte alors sur un énoncé soit/ou :

> soit le vrai méga-Sudoku S a une solution, soit le leurre D a une solution.

C'est le lien logique. D n'est **pas** généré par un procédé magique qui rend S vrai. La preuve ne dit pas « D n'a pas de solution, donc S a une solution ». Elle prouve la disjonction **S ou D**. La solidité parfaite dit qu'une disjonction fausse ne peut pas avoir de preuve valide. Puisque D est faux en réalité — il n'a pas de solution — la seule façon que la disjonction soit vraie est que S soit vrai. Donc si la preuve est acceptée, S doit avoir une solution. Le leurre ne peut pas rendre vrai un S faux.

Mais pour la partie de type zéro connaissance, demandez ce qui arriverait si D *avait* une solution. Cette solution-leurre servirait de témoin alternatif. Elle permettrait à quelqu'un de produire des preuves sans connaître la vraie solution du méga-Sudoku d'Alice — un simulateur, autrement dit. En réalité D n'a pas de solution, donc cette route de simulation est fermée. Le point est que le système formel ne peut pas prouver efficacement qu'elle est fermée.

D a donc deux tâches. Pour la **solidité**, D est faux, donc une preuve valide de « S ou D » force S. Pour le **zéro connaissance effectif**, D est difficile à réfuter, donc le système formel ne peut pas exclure rapidement la route-leurre qui aurait rendu la simulation possible.

Le test de sécurité n'est donc plus :

> Pouvons-nous prouver qu'un simulateur existe vraiment ?

Il devient :

> Votre système formel peut-il prouver efficacement que le simulateur est impossible ?

Si la réponse est non, quelque chose d'étonnamment fort suit : toute garantie de sécurité qui (a) peut être observée en exécutant un test, et (b) découle prouvablement — dans ce système formel — de l'existence d'un simulateur, tient effectivement. Une attaque réussie contre l'une d'elles constituerait elle-même la courte réfutation manquante, et cette courte réfutation n'existe pas. C'est la partie « effective » du zéro connaissance effectif.

Le contraste de salle de classe est donc :

Zéro connaissance classique : les preuves sont sûres parce qu'un simulateur existe.

Zéro connaissance effectif à la Gödel : les preuves sont traitées comme sûres pour les tests de sécurité observables parce que le système formel ne peut pas prouver efficacement que le simulateur est impossible.

La seconde affirmation est plus faible. C'est aussi pourquoi l'article peut garder les trois traits qui avaient brisé la version classique : un message, aucun setup et une solidité parfaite.

Le nouveau test : vous ne pouvez pas prouver que le simulateur est absent

La relaxation d'Ilango change la question.

Le zéro connaissance classique demande :

> Un simulateur existe-t-il ?

Le zéro connaissance effectif demande quelque chose de plus faible :

> Votre système formel choisi peut-il prouver efficacement qu'aucun simulateur n'existe ?

Cela ressemble à une esquive technique, mais c'est l'idée centrale. La construction vit dans un état étrange : aucun simulateur n'existe réellement — l'article est explicite là-dessus — mais le système formel que vous avez fixé ne peut pas prouver efficacement qu'il n'existe pas. Si chaque mauvaise conséquence qui vous importe exigerait une telle réfutation, le système se comporte encore comme du zéro connaissance pour ces conséquences.

C'est ici que Gödel entre en scène. Pas comme décoration, et pas comme « Gödel rend la crypto sûre ». Le lien relève de la théorie des preuves. Un système formel est dit **optimal** s'il est, en un sens précis, le meilleur possible : chaque fois qu'un système formel quelconque peut réfuter une formule du type

pertinent avec une courte preuve, le système optimal le peut aussi, avec une preuve au plus polynomialement plus longue. Krajíček et Pudlák ont conjecturé en 1989 qu'**aucun système de preuve optimal n'existe** : quel que soit le système fixé, un autre système prouve certaines familles d'énoncés vrais bien plus succinctement. C'est l'une des conjectures centrales de la complexité des preuves, et c'est le cousin fini, en complexité, du théorème d'incomplétude de Gödel : certains énoncés vrais n'ont pas de courte preuve dans le système que vous avez fixé — non parce qu'ils seraient improuvables en principe, mais parce que tout système fixé laisse certaines vérités courtes sans courte preuve.

L'article suppose cette conjecture (dans une forme « infiniment souvent » un peu plus forte, standard quand des conjectures sont utilisées cryptographiquement). Le gain, par un théorème de Krajíček et Pudlák, est concret : pour tout système formel, il existe une suite de formules réellement insatisfiables que ce système ne peut pas réfuter par de courtes preuves — et, point crucial, qu'un algorithme efficace peut *générer*. Cette dernière propriété, l'uniformité, transforme toute l'idée d'une affirmation d'existence en un véritable algorithme qu'Alice peut exécuter : ses leurres D sortent d'une chaîne de production, pas du néant.

Le mouvement cryptographique consiste à mettre ce manque de puissance de preuve au travail.

Ce que fait la construction

Voici la construction de l'article, ramenée à sa forme.

Fixez un système formel — ZFC, par exemple. Sous l'hypothèse de complexité des preuves, il existe une suite efficacement générable de formules réellement insatisfiables, mais pour lesquelles le système formel n'a pas de courte preuve d'insatisfiabilité.

Construisez maintenant une preuve en un message de cette forme :

> soit le vrai énoncé est satisfiable, soit cette formule spéciale difficile est satisfiable.

La formule spéciale difficile n'est pas satisfiable. Donc, si la machinerie de preuve sous-jacente est parfaitement saine, accepter le message signifie encore que le vrai énoncé est vrai. C'est ce qui donne la solidité parfaite.

Mais pour la sécurité de type zéro connaissance, imaginez que la formule spéciale difficile *soit* satisfiable. Son témoin pourrait alors servir à simuler des preuves sans connaître le vrai témoin. La formule n'est pas satisfiable en réalité — mais le système formel ne peut pas le prouver efficacement. Il ne peut donc pas prouver efficacement que le simulateur est impossible.

C'est la charnière. Le système ne cache pas le secret en produisant un simulateur classique. Il cache le secret, pour une grande classe de tests de sécurité observables, derrière l'incapacité du système formel à certifier que le simulateur est absent.

Ce que l'article affirme

Le théorème principal arrive par couches. Le résultat central est celui-ci :

Sous une hypothèse cryptographique standard — l'existence de **preuves non interactives à indistinguabilité de témoin**, objets bien étudiés qui découlent de plusieurs paquets d'hypothèses établis — et sous la conjecture de complexité des preuves selon laquelle **aucun système de preuve optimal (infiniment souvent) n'existe**, l'article construit, pour tout choix de système formel, un prouveur et un vérificateur en un message pour NP/SAT avec **solidité parfaite** et sans setup, qui sont effectivement zéro connaissance relativement à ce système. (NP/SAT est le « plus grand commun dénominateur » standard des problèmes de type puzzle ; le méga-Sudoku est l'un de ses costumes.)

Pour l'affirmation plus large sur la préservation des propriétés de sécurité falsifiables, l'article ajoute une autre hypothèse standard, la croyance de dérandomisation **P = BPP** (en gros : le hasard ne donne pas de pouvoir essentiel supplémentaire aux algorithmes).

Traduit hors du langage des théorèmes :

- La preuve est un seul message.
- Il n'y a pas de setup de confiance.
- Les énoncés faux ne peuvent pas être prouvés.
- Le prouveur n'est pas zéro connaissance classique — il n'a pas de simulateur.
- Mais toute conséquence falsifiable, basée sur des jeux, du zéro connaissance classique peut être obtenue dans ce cadre.

« Falsifiable » compte. Cela signifie qu'un échec de sécurité peut être testé en faisant tourner un adversaire dans un jeu. Beaucoup de définitions de sécurité cryptographique ont cette forme : l'adversaire peut-il distinguer deux chiffrés, inverser une fonction, récupérer un témoin ou gagner une expérience spécifiée ? Le théorème donne un prouveur pour chaque propriété falsifiable, une à la fois. Un seul prouveur jouissant de *toutes* les propriétés falsifiables à la fois est probablement impossible — l'ancienne attaque de réutilisabilité (« Bob peut montrer la preuve à d'autres ») est elle-même une propriété falsifiable, et elle échoue réellement ici. La proposition de l'article est qu'un seul prouveur peut plausiblement couvrir toutes les propriétés falsifiables *naturelles* — celles qui apparaissent réellement dans la pratique cryptographique — mais cette partie est un théorème conditionnel reposant sur une notion informelle de « naturel », plus une conjecture explicite. La garantie vise les échecs observables, pas toute signification philosophique ou simulationnelle du secret.

Un corollaire concret mérite d'être nommé : la construction donne les premières preuves non interactives *witness hiding* avec un prouveur uniforme — « une preuve d'un puzzle ne vous aide pas à trouver sa solution », sans interaction et sans setup — un objet au son modeste qui résistait à la construction depuis des décennies.

Ce que cela ne dit pas

Cela ne dit **pas** que les anciens théorèmes d'impossibilité étaient faux. La construction les évite en changeant la définition.

Cela ne donne **pas** du zéro connaissance ordinaire, classique, sans interaction, sans setup et avec solidité parfaite. L'article dit explicitement que le prouveur construit n'a pas de simulateur.

Cela ne signifie **pas** que la preuve ne peut pas être réutilisée. Une preuve en un message peut toujours être montrée à quelqu'un d'autre ; l'article ne préserve pas les propriétés de type déniabilité. (Le zéro connaissance non interactif avec setup de confiance a la même limite.)

Cela ne signifie **pas** que c'est un protocole pratique prêt à être déployé. C'est de la théorie de la complexité et des fondements cryptographiques. Le résultat dépend d'hypothèses majeures de complexité des preuves et de cryptographie, et la construction porte sur ce qui est possible en principe.

Cela ne fait **pas** de « Gödel » une primitive de sécurité magique. Le lien avec Gödel passe par les systèmes de preuve, les systèmes de preuve optimaux et des analogues finis de l'incomplétude. L'intuition utilisable n'est pas « l'incomplétude protège votre mot de passe ». Elle est : si un système formel ne peut pas prouver efficacement qu'un simulateur est impossible, alors les attaques qui exigeraient cette preuve peuvent être bloquées au niveau des définitions de sécurité.

Pourquoi c'est intéressant quand même

La cryptographie transforme souvent la difficulté en sécurité. La factorisation est difficile, donc les hypothèses de type RSA deviennent utiles. Les problèmes de réseaux sont difficiles, donc la cryptographie sur réseaux devient utile. Ici, la difficulté est plus étrange : non pas « difficile de calculer un secret », mais « difficile de prouver qu'un certain objet de preuve ne peut pas exister ».

C'est pourquoi l'article paraît inhabituel. Il traite les axiomes et les systèmes formels presque comme des ressources cryptographiques. L'impossibilité habituelle dit qu'il existe une tension entre solidité et simulation. Le mouvement d'Ilango consiste à placer cette tension derrière un rideau de théorie des preuves : le simulateur est absent, mais le système formel ne peut pas exposer efficacement cette absence.

Pour un lecteur, la partie surprenante n'est pas que cela remplacera les systèmes zéro connaissance d'aujourd'hui. Ce ne sera probablement pas le cas, du moins pas directement. La partie surprenante est qu'une limitation de la logique mathématique peut être utilisée constructivement : pas seulement comme un mur, mais comme une forme de couverture.

Quelle est la solidité des preuves ?

C'est un article de théorème, donc « preuve » signifie autre chose que dans un article de biologie ou d'astronomie. La question n'est pas de savoir si une expérience s'est répliquée. La question est de savoir si les définitions, les hypothèses et la chaîne de preuve soutiennent l'affirmation.

La preuve est formelle, et l'article est explicite sur ses hypothèses. Ces hypothèses ne sont pas légères. Les preuves non interactives à indistinguabilité de témoin sont des objets standard en cryptographie et découlent de plusieurs paquets d'hypothèses établis. La conjecture de non-existence de système de preuve optimal est une conjecture centrale de la complexité des preuves. P = BPP est une croyance de dérandomisation standard utilisée seulement pour le théorème plus large sur les propriétés falsifiables.

L'article soutient aussi que les hypothèses sont le bon prix, pas un échafaudage arbitraire : il prouve une réciproque montrant qu'elles sont essentiellement nécessaires — si de telles constructions existent, alors les preuves non interactives à indistinguabilité de témoin doivent exister, et (en accordant les fonctions à sens unique standard) aucun système de preuve optimal ne peut exister. Et les hypothèses sont « gagnant-gagnant » : réfuter l'une d'elles serait déjà une découverte majeure en complexité des preuves, cryptographie ou théorie de la complexité.

Mais comme le résultat est conditionnel, la confiance l'est aussi. Si ces hypothèses échouent, l'interprétation du théorème change. Et même si elles tiennent, la garantie n'est pas le zéro connaissance classique complet ; c'est la version relaxée, formulée en théorie des preuves, de l'article.

La bonne confiance est donc élevée sur le fait que l'article établit un résultat de possibilité conditionnel cohérent ; modérée sur le fait que ses hypothèses décrivent le monde cryptographique où nous vivons ; et faible pour toute conséquence pratique immédiate.

Pourquoi c'est important

L'article ouvre une route que l'on pensait fermée.

La théorie classique dit : le zéro connaissance complet ne peut pas être un seul message sans setup, et ne peut pas être parfaitement solide. L'article d'Ilango dit : si l'on demande les conséquences du zéro connaissance qui peuvent être testées dans des jeux de sécurité, et si l'on permet à la définition de sécurité de dépendre de ce qu'un système formel peut ou ne peut pas réfuter efficacement, alors une grande partie du comportement utile peut être récupérée — avec un message, sans setup et avec solidité parfaite.

Ce n'est pas un petit ajustement définitionnel. C'est une autre façon de penser les garanties cryptographiques. Au lieu de demander seulement ce qui existe, demander ce que votre système formel peut exclure. Au lieu de traiter l'improuvabilité comme une nuisance philosophique, l'utiliser comme structure.

Le monde pratique ne changera peut-être pas demain. Mais la carte conceptuelle, elle, change. Il existe maintenant un sens formel dans lequel « personne ne peut prouver efficacement que le secret a fui » peut être assez fort pour récupérer beaucoup des protections fondées sur les jeux que l'on voulait obtenir de « le secret n'a pas fui ».

Voilà pourquoi Gödel appartient au titre.

En bref

Les preuves à divulgation nulle de connaissance permettent à un prouveur de convaincre un vérificateur qu'un énoncé est vrai sans révéler le témoin. Les résultats classiques d'impossibilité disent que le zéro connaissance ne peut pas être comprimé dans un seul message sans setup, et ne peut pas avoir de solidité parfaite. L'article de Rahul Ilango ne réfute pas ces impossibilités. Il définit une notion plus faible, le zéro connaissance effectif : au lieu d'exiger qu'un simulateur existe vraiment, il exige qu'un système de preuve choisi — un système formel comme ZFC — ne puisse pas prouver efficacement qu'aucun simulateur n'existe. Sous de grandes hypothèses de cryptographie (preuves non interactives à indistinguabilité de témoin) et de complexité des preuves (aucun système de preuve optimal n'existe), l'article construit des prouveurs en un message pour NP/SAT, sans setup et avec solidité parfaite, qui obtiennent les conséquences falsifiables et basées sur des jeux du zéro connaissance propriété par propriété. Un prouveur unique couvrant toutes les propriétés « naturelles » de ce type est une extension supplémentaire, en partie conjecturale — et couvrir littéralement toutes les propriétés falsifiables est probablement impossible, parce que les preuves restent réutilisables. Le résultat est théorique et conditionnel, pas une primitive déployée, mais il montre une nouvelle façon d'utiliser l'improuvabilité en théorie des preuves comme ressource cryptographique.

Mise au point

Ce que montre l'article : Sous les hypothèses énoncées, on peut construire des prouveurs en un message, sans setup, parfaitement solides pour NP/SAT, qui sont effectivement zéro connaissance relativement à tout système de preuve choisi, et qui atteignent chaque conséquence falsifiable, fondée sur des jeux, du zéro connaissance classique.

Ce qui est plausible mais pas prouvé inconditionnellement : Que les hypothèses nécessaires de complexité des preuves et de cryptographie tiennent. Ce sont des hypothèses sérieuses et bien étudiées — et l'article montre qu'elles sont essentiellement nécessaires autant que suffisantes — mais elles restent des hypothèses.

Ce que cela ne montre pas : Du zéro connaissance classique avec absence d'interaction, absence de setup et solidité parfaite ; un système pratique prêt au déploiement ; la déniabilité ou la non-réutilisabilité des preuves ; ni que le théorème d'incomplétude de Gödel sécurise à lui seul la cryptographie.

Principales limites : La garantie est une relaxation du zéro connaissance ; la version la plus large dépend de plusieurs hypothèses ; les affirmations de prouveur universel unique restent en partie conjecturales ; et le résultat est surtout fondationnel.

Quel niveau de confiance un lecteur général devrait-il avoir ? Élevé sur le fait que c'est un résultat théorique conditionnel important si l'on accepte les définitions. Modéré sur le fait que les hypothèses capturent la réalité. Faible pour un déploiement pratique immédiat. Le bon résumé prudent est : l'article ne casse pas les impossibilités du zéro connaissance ; il trouve une nouvelle voie par la théorie des preuves autour des parties qui comptent pour beaucoup de jeux de sécurité.

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`