# Enseigner les preuves à divulgation nulle sur des chiffrements sans prérequis d'arithmétique

Xavier Bultel

# Enseigner les preuves à divulgation nulle sur des chiffrements sans prérequis d'arithmétique

Xavier Bultel
LIFO, INSA Centre Val de Loire, Bourges, France

***Résumé*—Les preuves à divulgation nulle sont des outils cryptographiques théoriques puissants ayant de nombreuses applications concrètes. Elles sont notamment utilisées dans les protocoles de vote électronique et certaines crypto-monnaies. Cependant, leur utilisation en cryptographie nécessite un bagage théorique conséquent, ce qui peut rebuter les étudiants issus de filières techniques. Dans cet article, nous présentons des preuves à divulgation nulle sur des chiffrements ne nécessitant aucune connaissance particulière en cryptographie, à l'exception du concept de chiffrement à clé publique. Il n'est donc pas nécessaire de connaître le fonctionnement interne du chiffrement utilisé pour les comprendre. Nous présentons aussi une approche pédagogique basée sur l'utilisation de boîtes cadenassées pour expliquer de façon intuitive le fonctionnement de nos preuves. Cette approche permet d'aborder immédiatement et sans prérequis des applications très concrètes, comme par exemple le vote électronique.**

## I. Introduction

Les preuves à divulgation nulle sont un concept fascinant proposé par Shafi Goldwasser, Silvio Micali, et Charles Rackoff [2]. Elles permettent de démontrer une propriété qui n'est en principe vérifiable que par l'utilisation d'une valeur secrète, sans pour autant révéler la moindre information sur cette valeur. Ces preuves ont naturellement trouvé de nombreuses applications en cryptographie, puisqu'elles permettent de prouver certaines propriétés sur des données chiffrées sans dévoiler d'information supplémentaire sur les clés de déchiffrement ou sur les données. Elles sont notamment utilisées par des protocoles préservant l'anonymat, le vote électronique, et certaines crypto-monnaies. Par exemple, dans le vote électronique, les bulletins sont remplacés par des chiffrés contenants l'identifiant du candidat choisi. Le dépouillement consiste donc à déchiffrer les bulletins. Lorsque le résultat du vote est dévoilé, il est nécessaire de démontrer aux votants qu'il a été calculé correctement, cependant, il n'est pas possible de dévoiler la clé de déchiffrement, puisque cela désanonymiserait les votes. Il est donc nécessaire de *prouver* l'exactitude du résultat des votes *sans divulguer* le déchiffrement de chacun des bulletins.

L'apprentissage des preuves à divulgation nulle commence souvent par la présentation des preuves basées sur des problèmes de graphes, et particulièrement par la preuve d'existence d'un isomorphisme entre deux graphes, et la preuve de la 3-coloration d'un graphe. Ces constructions ne nécessitent pas de connaissances théoriques avancées pour être comprises, et permettent d'introduire les différents concepts efficacement. Cependant, ces constructions n'ont pas d'application concrète en cryptographie, et leur intérêt est donc limité à leur fort potentiel pédagogique.

Afin de présenter quelques exemples concrets de preuves appliquées à la cryptographie, il est courant de compléter l'apprentissage par l'étude des preuves de connaissance d'un logarithme discret dans un groupe d'ordre premier et ses variantes. Ces dernières permettent de construire des preuves de connaissance d'un message chiffré, ou de l'égalité entre deux messages chiffrés par un chiffrement de type ElGamal. Si cette fois les applications sont palpables, ces preuves nécessitent en revanche des connaissances approfondies en cryptographie et en arithmétique.

Dans cet article, nous présentons une preuve d'inégalité de messages chiffrés ne nécessitant presque aucun prérequis théorique. Cette preuve est applicable sur n'importe quel chiffrement à clé publique, du moment que celui-ci est *randomisable*, c'est à dire qu'on peut rafraîchir l'aléa utilisé pour chiffrer le message sans utiliser la clé de déchiffrement. Il n'est pas nécessaire de comprendre *pourquoi* le chiffrement est randomisable pour comprendre le fonctionnement de la preuve.

Cette preuve est conceptuellement très simple, et il est possible de l'introduire par une version non numérique, en utilisant des objets basiques de la vie de tous les jours. Ce protocole non numérique permet de montrer comment, étant données deux boîtes identiques fermées par des cadenas dont on possède la clé, il est possible de démontrer que les deux boîtes contiennent deux cartes différentes, sans pour autant révéler quelles sont ces cartes.

Notre preuve est sécurisée dans un modèle où le vérificateur est honnête mais curieux (le vérificateur ne s'écarte pas du protocole, mais exploite tout ce dont il dispose dans l'espoir d'obtenir de l'information sur les données secrètes). Nous montrons qu'il est possible, en utilisant un engagement cryptographique, d'adapter notre preuve d'inégalité à un modèle ou le vérificateur est malveillant. L'intérêt pédagogique de ne pas présenter la solution la plus sécurisée dès le début est de sensibiliser aux subtilités des modèles de sécurité en générant la surprise : il n'est pas évident de remarquer de prime abord que le vérificateur peut tricher.

Nous présentons une adaptation de notre preuve pour démontrer l'égalité entre deux chiffrés. Cela nécessite une hypothèse supplémentaire sur le chiffrement : il doit être possible de modifier le message chiffré sans connaître la clé secrète. Cette deuxième construction reste néanmoins très accessible, et conceptuellement aussi simple que la première.

Enfin, afin de donner un aspect plus concret à nos outils pédagogiques, nous montrons comment nos preuves peuvent être utilisées dans un protocole de vote électronique simple,

où chaque votant chiffre son bulletin lors d'un référendum.

Ces protocoles ont été (en partie) présentés dans l'article [1]. L'article contient aussi d'autres protocoles, légèrement plus techniques mais avec une structure plus classique (appelée structure sigma), et nécessitant la possibilité de randomiser la clé de chiffrement. L'adaptation de ces protocoles pour le vote est une contribution originale.

## II. Définitions

Dans cette section, nous définissons quelques concepts cryptographiques.

Un **chiffrement à clé publique** est défini par trois algorithmes : un algorithme générant une clé publique $\mathsf{pk}$ et une clé secrète $\mathsf{sk}$, un algorithme $\mathsf{Enc}_{\mathsf{pk}}(m;r)$ produisant un chiffré $c$ à partir d'une clé publique $\mathsf{pk}$, d'un message $m$, et une valeur aléatoire $r$, et un algorithme $\mathsf{Dec}_{\mathsf{sk}}(c)$ permettant de déchiffrer un chiffré $c$ à l'aide de la clé secrète $\mathsf{sk}$.

Pour nos preuves, nous aurons besoin de la propriété suivante : un chiffrement est dit **randomisable** si il existe un algorithme $\mathsf{Rand}_{\mathsf{pk}}(c,r)$ permettant de rafraîchir la valeur aléatoire utilisée dans un chiffré avec un nouvel aléa $r$, sans rien connaître à part le chiffré $c$ et la clé publique $\mathsf{pk}$ utilisée pour le produire. Concrètement, cela veut dire qu'en considérant un chiffré $c = \mathsf{Enc}_{\mathsf{pk}}(m;r)$, rechiffrer $m$ avec un nouvel aléa $r_1$ (c'est à dire calculer $\mathsf{Enc}_{\mathsf{pk}}(m;r_1)$) est équivalent à randomiser $c$ avec un aléa $r_2$ (c'est à dire calculer $\mathsf{Rand}_{\mathsf{pk}}(c,r_2)$).

Bien qu'il ne soit pas nécessaire de comprendre comment construire un chiffrement randomisable pour utiliser cette propriété dans nos preuves à divulgation nulle, le lecteur érudit remarquera que le chiffrement d'ElGamal est randomisable. Un chiffré ElGamal de $m$ avec un aléa $r$ pour une clé publique $\mathsf{pk}$ dans un groupe $\mathbb{G}$ généré par $g$ est un couple $(c_1, c_2) = (g^r, \mathsf{pk}^r \cdot m)$. Pour randomiser ce couple avec un aléa $s$, il suffit de calculer $(c_1 \cdot g^s, c_2 \cdot \mathsf{pk}^s) = (g^r \cdot g^s, \mathsf{pk}^r \cdot m \cdot \mathsf{pk}^s) = (g^{r+s}, \mathsf{pk}^{r+s} \cdot m)$.

Un **engagement cryptographique** est défini par deux algorithmes : un algorithme $\mathsf{Commit}(m, \mathsf{tr})$ produisant un engagement $\mathsf{cm}$ de $m$ avec une trappe $\mathsf{tr}$, et un algorithme $\mathsf{Open}(\mathsf{cm}, m, \mathsf{tr})$ qui valide l'engagement $\mathsf{cm}$ pour $m$ avec la trappe $\mathsf{tr}$.

Les engagement cryptographiques vérifient deux propriétés : la **contrainte**, qui garantit qu'il n'est pas possible de produire une nouvelle trappe $\mathsf{tr}'$ permettant de valider un autre message $m'$ que celui engagé, et le **secret**, qui garantit que sans la trappe, il est impossible de deviner la valeur engagée.

On peut construire un engagement cryptographique à partir d'une fonction de hachage. Dans ce cas, l'engagement est le haché de la trappe concaténée au message : $\mathsf{cm} = H(\mathsf{tr} \| m)$. La validation consiste à recalculer le haché $H(\mathsf{tr} \| m)$ et à le comparer avec $\mathsf{cm}$. Si $H$ est à sens unique, alors il n'est pas possible de retrouver $m$ sans connaître $\mathsf{tr}$. Si $H$ résiste aux collisions, alors il n'est pas possible de produire $\mathsf{tr}' \| m'$ tel que $H(\mathsf{tr} \| m) = H(\mathsf{tr}' \| m')$. Ce simple schéma d'engagement assure donc la contrainte et le secret.

Enfin, une **preuve à divulgation nulle** est un protocole entre un vérifieur et un prouveur. Ce dernier connaît, contrairement au vérifieur, une certaine valeur secrète lui permettant de vérifier une certaine propriété. Le vérifieur souhaite vérifier la propriété sans apprendre la valeur secrète. Dans cet article, on s'intéressera au cas où le vérifieur souhaite savoir si deux chiffrés chiffrent la même valeur ou non, et ou le prouveur connaît la clé secrète permettant de déchiffrer les messages. Une preuve à divulgation nulle vérifie les propriétés suivantes :

- La **robustesse**, qui garantit que la probabilité de prouver quelque chose de faux peut être aussi petite qu'on le souhaite. Concrètement, cela veut dire que cette probabilité est l'inverse d'une exponentielle en un paramètre de sécurité. La complexité du protocole doit, par contre, rester polynomiale en ce paramètre.
- La **non divulgation**, qui garantit que le vérifieur n'apprend rien de plus que la propriété qu'il souhaite vérifier. Concrètement, cela veut dire qu'il peut simuler son interaction avec le prouveur sans connaître le secret. La simulation doit produire des valeurs sur la même distribution probabiliste que le vrai protocole.

## III. Concept et preuve physique d'inégalité

Dans cette section, nous présentons une preuve d'inégalité entre deux messages chiffrés avec une même clé publique. Le concept de cette preuve est très simple et peut être expliqué à n'importe quel public, sans aucun prérequis scientifique. Pour cela, remarquons qu'un chiffrement peut être représenté par une boîte cadenassée. La clé publique est un cadenas ouvert, que n'importe qui peut refermer, et la clé secrète est la clé qui ouvre ce cadenas. Pour "chiffrer" un message en utilisant une boîte et un cadenas, on met le message dans la boîte, et on la ferme avec le cadenas. Seul le détenteur de la clé pourra ouvrir la boîte et lire le message.

Imaginons que deux cartes différentes, par exemple une dame de pique et une dame de cœur, soient chacune placée dans deux boîtes opaques, identiques en tout point. Ces boîtes sont ensuite fermées à l'aide de deux cadenas identiques. Il est donc impossible, sans posséder la clé permettant d'ouvrir les cadenas, de distinguer la boîte contenant la dame de pique de celle contenant la dame de cœur. Bob, qui ne détient pas la clé, aimerait savoir si les deux boîtes contiennent bien des cartes différentes. Dans ce qui suit, on va montrer comment Alice, qui possède la clé, peut démontrer cela à Bob, sans lui révéler ni sa clé, ni les cartes qui sont effectivement dans les boîtes.

Le protocole est une répétition des étapes suivantes. Ces étapes sont répétées $k$ fois, où $k$ est un paramètre de sécurité :

- Bob met les deux boîtes derrière son dos, et choisit une des deux boîtes au hasard (Alice ne sait pas quelle boîte Bob a choisie). Bob présente la boîte à Alice.
- Alice ouvre la boîte avec sa clé sans montrer son contenu à Bob. Elle retient mentalement la carte qui est dans la boîte sans la révéler, et referme la boîte avec le cadenas.
- Ensuite, Bob remet les boîtes derrière son dos, et choisit de nouveau une des deux boîtes au hasard, en prenant soin de noter mentalement si il a choisi la même boîte que la première fois ou non.
- Alice ouvre la boîte en cachette avec sa clé, la referme, et annonce à Bob si la boîte choisie est la même que

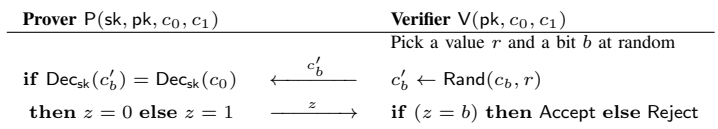


FIGURE 1. Une étape du protocole de preuve d'inégalité de messages chiffrés (à répéter $k$ fois).

la fois précédente ou non (pour cela, elle vérifie si la boîte contient la même carte ou non). Si Alice s'est trompée, Bob arrête le protocole. Sinon, Alice et Bob recommencent ces étapes.

Si Bob n'a pas interrompu le protocole, la preuve est réussie.

Puisque les deux boîtes sont identiques en tout point, si elles contenaient des cartes identiques, alors Alice ne pourrait pas savoir laquelle des boîtes elle reçoit la deuxième fois. Ainsi, elle aurait une chance sur deux de donner une mauvaise réponse à chaque répétition du protocole. Comme le protocole est répété $k$ fois, la probabilité de réussir à démontrer que les boîtes contiennent des cartes différentes alors que ce n'est pas le cas est de $1/2^k$, ce qui démontre que la preuve est **robuste**. En répétant le protocole quelques dizaines de fois, cette probabilité descend à une valeur négligeable en pratique.

La preuve est **à divulgation nulle** car Bob n'apprend rien pendant le protocole. Il sait déjà si les deux boîtes ont été interverties ou non, la réponse d'Alice ne lui donne donc aucune information.

## IV. Preuve d'inégalité sur des chiffrements

Nous pouvons construire une preuve d'inégalité de deux messages chiffrés avec un mécanisme similaire à celui de la section précédente. Supposons que deux messages différents $m_0$ et $m_1$ sont chiffrés avec un chiffrement randomisable, et qu'Alice, qui possède la clé de déchiffrement, veut convaincre Bob que les messages sont différents sans révéler ni sa clé ni les messages. Pour cela, Alice et Bob répètent $k$ fois le protocole suivant : Bob randomise un des deux chiffrés et envoie le résultat à Alice, puis Alice déchiffre ce résultat et détermine lequel des deux chiffrés a été randomisé. Pour cela elle compare le message $m$ qu'elle a déchiffré avec $m_0$ et $m_1$. Si Alice se trompe, alors Bob interrompt le protocole, et rejette la preuve d'Alice. Le schéma en Figure 1 résume ce protocole.

Si les deux messages sont égaux, alors on aura toujours $m = m_0 = m_1$, et Alice ne pourra pas déterminer quel chiffré a été randomisé. Autrement dit, si les messages ne sont pas différents, alors la probabilité de réussir cette preuve est de $1/2^k$, ce qui démontre que la preuve est **robuste**. D'autre part, cette fois encore, Bob n'apprend rien qu'il ne savait déjà, puisqu'il sait quel message il a randomisé, la preuve est donc **à divulgation nulle**.

On distingue deux modèles d'adversaires pour les preuves sans divulgation. Le vérifieur *honnête mais curieux*, qui respecte le protocole tout en essayant d'exploiter les informations dont il dispose, et le vérifieur *malhonnête*, qui s'autorise à ne pas respecter le protocole. Jusqu'à maintenant, nous nous sommes placés implicitement dans le cas d'un vérifieur honnête mais curieux, mais nous allons voir que cela peut s'avérer insuffisant. Imaginons par exemple que le vérifieur ait de forts soupçons sur le fait que l'un des deux chiffrés contienne le message $m$, et qu'il voudrait s'en convaincre. Pour cela, il entame un protocole de preuve d'inégalité de messages, mais au lieu de randomiser l'un des deux messages, il chiffre $m$ et envoie le résultat au prouveur. Le prouveur déchiffre alors $m$, et indique donc au vérifieur si l'un des deux messages chiffrés est égal à $m$. Dans ce cas, il est évident que le vérifieur apprend une information qu'il n'aurait pas pu déterminer tout seul. Cette attaque peut être expliquée avec le protocole utilisant des boîtes cadenassées : dans ce cas le vérifieur prépare une boîte identique aux deux autres contenant une dame de pique, et remplace la seconde boîte par la nouvelle lorsqu'il met les boîtes derrière son dos. Ainsi, si la première boîte contient aussi la dame de pique, le prouveur ne parviendra pas à produire une preuve valide.

Il est possible d'éviter ce genre d'attaques en utilisant un engagement cryptographique. Dans ce cas, le prouveur commence par engager sa réponse. Le vérifieur envoie alors le chiffré qu'il a randomisé et la valeur de l'aléa utilisé. Le prouveur vérifie que ces informations sont correctes en reproduisant la randomisation effectuée par le vérifieur, puis envoie la trappe permettant d'ouvrir son engagement. Le vérifieur peut finalement vérifier la réponse du prouveur. Le schéma en Figure 2 résume ce protocole.

Cette nouvelle preuve est toujours **robuste** puisque le vérifieur ne peut pas modifier la réponse qu'il a engagée après avoir pris connaissance de l'aléa utilisé par le vérifieur. Donc tant que l'engagement vérifie la contrainte, la probabilité de falsifier la preuve reste proche de $1/2^k$. La preuve est **à divulgation nulle** même si le vérifieur tente de tricher, puisque dans ce cas, le prouveur ne révèle ni sa réponse ni la trappe. Donc tant que l'engagement vérifie le secret, l'engagement ne dévoile aucune information sur la réponse du prouveur au vérifieur.

## V. Application au vote

Afin de compléter notre approche pédagogique, nous montrons une application concrète de cette preuve d'inégalité : un protocole de vote électronique simplifié. Notre système de vote fonctionne de la manière suivante. Une autorité de dépouillement publie une clé publique de chiffrement, et chaque votant chiffre son vote avec cette clé. On se mettra dans le cas d'un référendum, où chaque votant chiffre soit "oui" soit "non". L'autorité de dépouillement déchiffre ensuite les votes et donne le résultat de l'élection, sans révéler qui a voté pour quoi. Cependant, l'autorité pourrait donner un résultat erroné pour falsifier l'élection. Pour empêcher cela, nous complétons le protocole par une preuve à divulgation nulle, permettant de démontrer que le résultat du vote est correct, sans dévoiler le vote de chaque utilisateur.

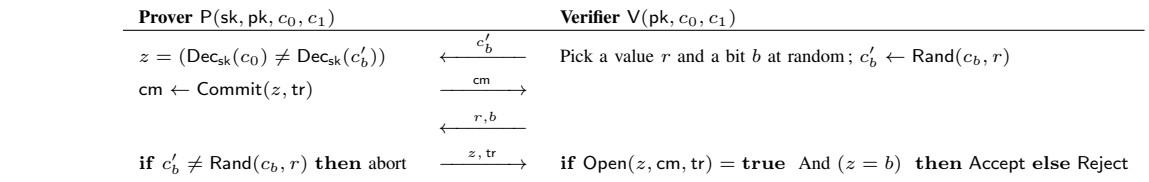


FIGURE 2. Une étape du protocole de preuve d'inégalité de messages chiffrés où le vérifieur est malhonnête (à répéter $k$ fois).

Remarquons d'abord que l'ensemble des votes peut être vu comme un seul *grand* chiffré du résultat de l'élection. Dans ce cas, en prenant par exemple le résultat *"3 oui contre 2 non"*, chiffrer le résultat revient à chiffrer 3 fois "oui" et 2 fois "non". Ce *grand* chiffré est randomisable si chacun des *petits* chiffrés qui le composent l'est aussi, à ceci près qu'il est nécessaire de réordonner aléatoirement les *petits* chiffrés en plus de les randomiser. Nous pouvons donc appliquer notre preuve d'inégalité sur l'ensemble des votes.

Il ne reste plus qu'a démontrer que le résultat du vote est bien celui annoncé. Pour cela, le vérifieur (dans notre cas les votants) va créer un ensemble de votes fictifs pour chaque autre résultat possible à l'élection, et le prouveur (dans notre cas l'autorité de dépouillement) va prouver que le résultat de la vraie élection, obtenu depuis l'ensemble des votes chiffrés par les votants, est différent du résultat chiffré dans l'ensemble des votes de chaque élection fictive. Par exemple, si le résultat annoncé est *"3 oui contre 2 non"*, le vérifieur va simuler des votes dont les résultats sont *"0 oui contre 5 non"*, *"1 oui contre 4 non"*, *"2 oui contre 3 non"*, *"4 oui contre 1 non"* et *"5 oui contre 0 non"*, et le prouveur va montrer que le résultat de la vraie élection est différent du résultat de chacune de ces élections fictives.

## VI. PREUVE D'ÉGALITÉ

En se basant sur la méthode que nous venons de présenter, il est aussi possible de construire une preuve d'égalité entre deux messages chiffrés. Pour cela, nous devons introduire une nouvelle propriété sur les chiffrements : on dira qu'un chiffrement est **malléable** si il existe un algorithme $\mathsf{Mall}_{\mathsf{pk}}(c, r)$ permettant de rafraîchir la valeur du message chiffré dans $c$ avec un nouvel aléa $r$, sans rien connaître à part le chiffré $c$ et la clé publique $\mathsf{pk}$ utilisée pour le produire. Concrètement, cela veut dire qu'en considérant un chiffré $c = \mathsf{Enc}_{\mathsf{pk}}(m; r)$, chiffrer un message aléatoire $m'$ avec $r$ (c'est à dire calculer $\mathsf{Enc}_{\mathsf{pk}}(m'; r)$) est équivalent à modifier $c$ avec un aléa $r'$ (c'est à dire calculer $\mathsf{Mall}_{\mathsf{pk}}(c, r')$). On dira que le chiffrement est **fortement malléable** lorsqu'il est possible, à partir du message original $m$ et du message randomisé $m'$, de retrouver l'aléa utilisé par $\mathsf{Mall}$ via un algorithme $\mathsf{MallExt}(m', m)$.

Par exemple, ElGamal est fortement malléable : en considérant un chiffré $(c_1, c_2) = (g^r, \mathsf{pk}^r \cdot m)$, on peut modifier le message $m$ avec un aléa $n$ en calculant : $(c_1, c_2 \cdot n) = (g^r, \mathsf{pk}^r \cdot (m \cdot n))$. On retrouve aisément l'aléa à partir de $m$ et $m \cdot n$ en calculant $m \cdot n/m$.

Supposons qu'Alice possède la clé de déchiffrement d'un chiffrement randomisable et fortement malléable et veut convaincre Bob que deux chiffrés chiffrent le même message. Pour cela, Alice et Bob répètent $k$ fois le protocole suivant : Bob randomise un des deux chiffrés, modifie le message qu'il contient, et envoie le résultat à Alice. Alice déchiffre ce résultat et détermine l'aléa utilisé pour modifier le message (c'est possible puisque le chiffrement est fortement malléable). Alice envoie cet aléa à Bob. Si Alice s'est trompée, Bob interrompt le protocole et considère que la preuve n'est pas valide.

Au cours du protocole, Alice reçoit le chiffré d'un message aléatoire avec un nouvel aléa, et n'a donc aucun moyen de discerner quel chiffré a été utilisé par Bob. Si les deux messages sont différents, alors Alice doit deviner quel chiffré Bob a utilisé pour déterminer correctement l'aléa et réussir la preuve. Alice a donc une chance sur deux de ne pas se tromper à chaque répétition de la preuve. Finalement, si les messages sont différents, la probabilité de réussir cette preuve est de $1/2^k$, ce qui démontre que la preuve est **robuste**. D'autre part, Bob n'apprend rien qu'il ne savait déjà puisqu'il sait quel message il a randomisé. La preuve est donc **à divulgation nulle**. Il est possible d'adapter ce protocole pour résister à un vérifieur malhonnête de la même façon que pour la preuve d'inégalité : Alice engage sa réponse, Bob révèle ses aléas, et si ceux-ci sont corrects, Alice révèle sa réponse.

Remarquons, sans détailler, qu'en introduisant le concept de chiffrement homomorphe pour l'addition (dont le chiffrement de Pailler est une instance), il est possible d'utiliser cette preuve pour améliorer notre protocole de vote : chaque votant chiffre 1 pour "oui" et 0 pour "non", ainsi la composition des votes est un chiffrement du nombre $n$, où $n$ est le nombre de "oui". Il suffit alors de rechiffrer $n$ et de prouver que ce chiffré et celui produit par la composition des votes chiffrent le même message.

## VII. CONCLUSION

Dans cet article, nous avons présenté des preuves sur chiffrements conceptuellement très simples pouvant être expliquées à un public d'étudiants n'ayant quasiment aucune connaissance théorique. Le concept de ces preuves peut être introduit de manière ludique avec un protocole utilisant des objets de la vie de tous les jours. Cette approche permet d'aborder rapidement (mais avec rigueur) des exemples d'applications très concrètes, comme l'utilisation de preuves à divulgation nulle pour le vote électronique.

## RÉFÉRENCES

[1] O. Blazy, X. Bultel, P. Lafourcade, and O. P. Kempner. Generic plaintext equality and inequality proofs. In *Financial Cryptography and Data Security*. Springer Berlin Heidelberg, 2021.

[2] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1) :186–208, 1989.