PhD-FSTM-2022-138
The Faculty of Science, Technology and Medicine

École normale supérieure of Paris, PSL

DISSERTATION

Defence held on 25/10/2022 in Esch-sur-Alzette
to obtain the degree of

DOCTEUR DE L'UNIVERSITÉ DU LUXEMBOURG

EN INFORMATIQUE

AND

DOCTEUR DE L'UNIVERSITE PARIS SCIENCES LETTRES PREPARE A L'ECOLE NORMALE SUPERIEURE

by

Fatima-Ezzahra EL ORCHE
Born on 18 August 1992 in Ben M'Sik Sidi Othmane, Casablanca (Morocco)

NEW SECURITY PRIMITIVES AND BETTER PERFORMANCE THROUGH PARAMETER TRADE-OFFS

Dissertation defence committee

Prof. Dr Peter Y.A. Ryan, dissertation supervisor
Professor, Université du Luxembourg

Prof. Dr David NACCACHE
Professor, Ecole Normale Supérieure of Paris

Prof. Dr Marcus VÖLP, Chairman
Professor, Université du Luxembourg

Prof. Dr Jacques PATARIN
Professor, Université de Versailles

Dr Peter Browne ROENNE, Vice Chairman
Université de Lorraine

Dr Rémy FEVRIER
Maître de Conférence HDR, CNAM
(Invited Researcher at ENS, Paris)

THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PSL

Préparée à l'Ecole Normale Supérieure de Paris
Dans le cadre d'une cotutelle avec l'Université de Luxembourg

Nouvelles Primitives de Sécurité et Meilleures Performances à travers des Compromis de Paramètres
New Security Primitives and Better Performance through Parameter Trade-Offs

Soutenue par
Fatima-Ezzahra EL ORCHE
Le 25 Octobre 2022

Ecole doctorale n° 386
Sciences Mathématiques de Paris Centre

Spécialité
Informatique

Composition du jury :

Marcus, VÖLP Professeur, Université de Luxembourg	*Président*
Jacques, PATARIN Professeur, Université de Versailles	*Rapporteur*
Peter Browne, ROENNE Dr, Université de Lorraine	*Examinateur*
Rémy, FEVRIER Maître de Conférence HDR, CNAM (Chercheur invité à l'ENS)	*Examinateur*
Peter Y.A., RYAN Professeur, Université de Luxembourg	Co-*Directeur de thèse*
David, NACCACHE Professeur, ENS Paris	*Directeur de thèse*

New Security Primitives and Better Performance through Parameter Trade-Offs

Fatima-Ezzahra El Orche

New Security Primitives and Better Performance through Parameter Trade-Offs

A thesis done in the framework of a co-tutelle program between the University of Luxembourg and École Normale Supérieure of Paris by:

Fatima-Ezzahra El Orche

and supervised by:

Prof. Peter Y.A. Ryan | Prof. David Naccache

Abstract

Security protocols are an important component in the field of information security. They ensure the information and communications exchange within the network in a secure way. Protecting communications between the users and the server against any type of attack requires having high security level protocols. However, increasing the level of security usually becomes very costly which implies that the productivity level decreases. The question that boils down here is therefore: what are the parameter trade-offs to take into consideration in order to improve the performance? We provide in this thesis an answer to this question based on two simple ideas of *duplicating* and *masking*. We develop several security protocols and analyze the effect of each parameter on the others.

The first contribution concerns the design of a new authenticated key exchange protocol. The particularity of this new protocol is to not rely on mathematical and number-theoretic cryptography and therefore is seen to be post-quantum secure. It entirely relies on symmetric primitives and approaches some of the functionalities of public-key cryptography. We show that we reach a good security level with our simple constructions by handing out more keys per user.

The second contribution consists of designing a new version of the protocol OV-Net, making it resistant to DoS attacks. We show that having multiple elections increases the computational complexity level but improves the protocol's robustness. We investigate the effect of parallel OV-Net on the different security parameters: time, privacy, robustness and accuracy. We stress that our protocol is well suited for decision-making applications.

The third contribution is the proposition of a new security primitive. Having information with a foreseeable lifespan is a new method for storing information which guarantees its destruction after a certain time. The storage is done on DNA-RNA molecules. We provide in this work a theoretical study of the method aging model. We consider that the information goes through three different periods of life: life, agony and death, and we define its security based on this consideration. We present three different algorithms on how to control the information lifetime.

The fourth contribution discusses the idea of masking voting ballots. We propose an optimization of the RLT [25] protocol. We investigate the implications of masking some parts of each ballot on privacy, coercion and vote selling, and how it can alleviate these issues.

Nouvelles Primitives de Sécurité et Meilleures Performances à travers des Compromis de Paramètres

Cette thèse de doctorat est réalisée dans le cadre d'un programme de co-tutelle entre l'université de Luxembourg et l'école normale supérieure de Paris par:

Fatima-Ezzahra El Orche

et encadrée par:

Prof. David Naccache | Prof. Peter Y.A. Ryan

Résumé

Les protocoles de sécurité constituent un élément très important dans le domaine de la sécurité de l'information. Grâce à eux, l'échange des informations et des communications se passe en toute sécurité au sein du réseau. De ce fait, la protection de ces informations qui sont communiquées entre les utilisateurs et le serveur contre tout type d'attaques nécessite un haut niveau de sécurité. Cependant, augmenter le niveau de sécurité devient très coûteux, ce qui impacte la productivité. La question qui se pose ici est donc la suivante: quels sont les compromis de paramètres à prendre en considération afin d'assurer des meilleurs performances? Nous répondons à cette question le long de cette thèse en se basant sur deux simples idées: *duplication* et *masquage*. Nous développons plusieurs protocoles de sécurité et analysons l'effet de chaque paramètre sur l'autre.

La première contribution concerne la conception d'un nouveau protocole d'échange de clés authentifié. La particularité de ce nouveau protocole est qu'il ne s'appuie pas sur la cryptographie mathématique, comme par exemple la théorie des nombres ou autres, ce qui le rend donc vu comme étant un protocole résistant aux attaques quantiques. Il s'appuie entièrement sur des primitives

symétriques et se rapproche de certaines des fonctionnalités de la cryptographie à clé publique. Nous montrons que nous atteignons un bon niveau de sécurité avec nos simples constructions en distribuant plusieurs clés par utilisateur.

La deuxième contribution consiste à développer une nouvelle version du protocole OV-Net, le rendant résistant aux attaques de déni de service. Nous montrons que le fait d'avoir plusieurs élections, certes augmente le niveau de complexité de calcul, mais améliore la robustesse du protocole. Nous étudions l'effet du parallèle OV-Net sur les différents paramètres de sécurité : temps, confidentialité, robustesse et précision du décompte des votes. Nous soulignons que notre protocole est bien adapté aux applications de prise de décisions.

La troisième contribution est la proposition d'une nouvelle primitive de sécurité. Disposer d'informations à durée de vie prévisible est une nouvelle méthode pour stocker l'information qui garantit sa destruction au bout d'un certain temps. Le stockage se fait sur des molécules d'ADN-ARN. Nous proposons dans ce travail une étude théorique du modèle de vieillissement de l'information. Nous considérons que l'information passe par trois périodes différentes au cours de sa vie: la vie, l'agonie et la mort, et nous définissons sa sécurité en fonction de cette considération. Nous présentons trois algorithmes différents sur la manière de contrôler la durée de vie de l'information.

La quatrième contribution discute l'idée de masquer un sous-ensemble des éléments des bulletins de vote. Nous proposons une optimisation du protocole RLT [25] et nous étudions les implications du masquage de certaines parties de chaque bulletin de vote sur la confidentialité, la coercition et la vente de votes, et comment cela peut atténuer ces problèmes.

Acknowledgement

First and foremost, I would like to thank my supervisors Prof. David Naccache and Prof. Peter Y.A. Ryan for their continuous support and help during all my PhD lifecycle. Especially, I would like to thank Prof. David Naccache for connecting me with Prof. Peter Y.A. Ryan and making this cotutelle between the Ecole Normale Superieure of Paris and the University of Luxembourg possible. I am as well grateful to both of them for the freedom they offered me to conduct research in topics that I am interested in.

I would like also to thank Dr. Peter Roenne for his daily support and for his insightful comments and suggestions during the years of my PhD. I am grateful also to Natacha Laniado for her assistance and coordination.

I would like to extend my sincere thanks to all APSIA members for being nice colleagues. My thanks go as as well to my collaborators, with whom I had deep and enriching discussions and conversations.

I would like to offer special thanks to all my friends for listening to me whenever I had difficult times during my PhD and for living with me all the remarkable moments I had during my PhD journey.

Finally, big thanks go to my family, especially to my mother Rabia, my father Mohammed and my sisters Hiba and Sara.

List of Publications

This thesis is based on scientific research articles that have been previously either published on conferences or put on ePrint. The third publication (Who was that masked voter? The tally won't tell!) is done in collaboration with an other PhD student (Najmeh Soroush) and is, therefore, is also part of her dissertation.

1. Fatima-Ezzahra El Orche, Marc Beunardeau, Diana Maimuţ, David Naccache, Peter B. Rønne, Peter Y.A. Ryan. **Authenticated Key Distribution: When the Coupon Collector is Your Enemy** *Published on SecITc 2019 in Bucharest, Romania.*

2. Fatima-Ezzahra El Orche, Gergei Bana, Remi Géraud-Stewart, David Naccache, Peter Rønne, Peter Y A Ryan, Marco Biroli, Megi Dervishi and Hugo Waltsburger **Time, Privacy, Robustness, Accuracy: Trade-Offs for the Open Vote Network Protocol** *Published on E-Vote-Id 2022, Bregenz, Austria.*

3. Peter Y. A. Ryan, Peter B. Rønne, Dimiter Ostrev, Fatima-Ezzahra El Orche, Najmeh Soroush, Philip B. Stark. **Who was that masked voter? The tally won't tell!**. *Published on E-Vote-Id 2021, Bregenz, Austria.*

4. Fatima-Ezzahra El Orche, Marcel Hollenstein, Sarah Houdaigoui, David Naccache, Daria Pchelina, Peter B. Rønne, Peter Y.A. Ryan, Julien Weibel, Robert Weil. **Taphonomical Security: (DNA) Information with a Foreseeable Lifespan**. *Accepted to FICC 2023, San Francisco, USA.*

Contents

List of Figures

List of Tables

Chapter 1

Résumé

1.1 Introduction

L'analyse de la sécurité des protocoles de sécurité est un sujet de préoccupation presque quotidien pour les chercheurs dans ce domaine. L'intérêt se porte beaucoup sur l'amélioration de leurs performances, qui demande une étude approfondie de l'impact de chaque propriété de sécurité sur l'autre. Par ailleurs, une meilleure performance nécessite d'avoir un bon compromis de paramètres.

Plusieurs types de protocoles de sécurité existent; par exemple, les protocoles d'authentification, les protocoles d'échange de clés secrètes, les protocoles de vote électronique, et plein d'autres, y compris les protocoles de stockage de l'information.

Selon le type de protocole à étudier, on distingue plusieurs types de propriétés de sécurité qui entrent en jeu. Il est important de garantir la confidentialité des messages, l'authentification et l'indiscernabilité des clés dans les protocoles d'échange de clés. Dans les protocoles de vote électronique, la confidentialité et la vérifiabilité du vote sont les principales propriétés examinées, en plus de l'exactitude du décompte, de la résistance à la coercition, et d'autres propriétés.

Le long de ce manuscrit de thèse, on étudie la possibilité d'avoir une meilleure sécurité de protocoles grâce à des ajustements et des compromis de paramètres.

1.1.1 Intuition Derrière la Thèse

Il y a deux principales idées que nous utilisons le long de cette thèse pour auguementer les performances de nos protocoles de sécurité: le *doublement* ou la *multiplication* et le *masquage*. La première idée est utilisée dans Chapter 3, Chapter 4 et Chapter 5, alors que la deuxième idée est utilisée dans Chapter 6.

Dans notre travail dans Chapter 3, on a conçu un système avec plusieurs grilles de clès (râteliers), qui peut être vue comme une matrice de clés, et l'utilisateur est invité à choisir un nombre fixe de clés de chaque colomne de la matrice de clés.

Dans notre deuxième travail dans Chapter 4, disposant de plusieurs élections et permettant au voteur de voter dans une sélection d'entre-elles contribue à l'augmentation la robustesse du système. Bien que cela a positivement impacté le niveau de robustesse du protocol, on montre que cela se fait au détriment de la confidentialité du vote. On présente une analyse bien détaillée de ça.

Dans notre travail Chapter 5, avoir plusieurs copies de la même molécule d'ADN-ARN contrôle efficacement la durée de vie des informations stockées dedans.

Finalement, dans notre travail Chapter 6 où on utilise l'idée de masquage, nous montrons que le masquage d'un sous-ensemble de composants de chaque bulletin de vote optimise les compromis entre différentes mesures de sécurité.

1.2 Un Nouveau Protocole d'Échange de Clés Authentifié

On présente dans cette partie un nouveau protocole d'échange de clés authentifié. Ce protocole, d'un côté, ne recoure pas aux configurations standard de clé publique avec des hypothèses correspondantes aux problèmes de calcul difficiles mais, d'autre part, il est plus efficaces que la distribution de clés symétriques entre les participants.

1.2.1 Le Protocole

Il y a trois entités qui participent au protocole. Deux utilisateurs et une autorité centrale (CA), qu'on suppose être honnête et qui intervient seulement dans la phase d'enregistrement. Le scénario est le suivant: il y a r râteliers chaqun contenant ℓ clés secrètes, dont l'autorité CA a générés au préalable. CA distribue à chaque utilisateur u clés distinctes choisies aléatoirement de chaque râtelier, soit au total $u \times r$ clés par utilisateur, avec de plus un matériel supplémentaire de clés.

La description du protocole est faite en deux étapes pour bien clarifier le scenario. On commence par décrire le cas basique et ensuite on le présente sous une forme générale. A l'issue de l'exécution du protocole, les deux utilisateurs arrivent à avoir une même clé.

1.2.2 Avantage Adversaire

L'objectif d'un attaquant est de retrouver les clès générées par CA. Pour cela, il essaye de corrompre un ensemble d'utilisateurs pour retrouver leurs clés secrètes. Cela ressemble au problème du collectionneur des vignettes. On calcule le nombre moyen d'utilisateurs qu'un attaquant peut corrompre pour retrouver toutes les clès envoyées par CA, ainsi que les différentes probabilités de succès selon si l'attaquant vise les clés d'un utilisateur spécifique ou d'un utilisateur arbitraire, et on discute la possibilité d'optimiser certains paramètres.

1.2.3 Sécurité du Protocole

La clé commune obtenue est en effet indiscernable: 1) d'une clé aléatoire pour un adversaire passif observant la communication entre les deux participants, 2) pour un adversaire qui modifie les messages envoyés, 3) ou encore pour un adversaire qui essaie de se faire passer pour quelqu'un d'autre, s'il ne possède pas toutes leurs clés ou s'il lui manque une des clès de chacun.

1.2.4 Analyse du paramètres

Etant donné que le protocole est cassé seulement si l'adversaire réussit à retrouver les clés secrètes envoyées par CA, on analyse dans ce chapitre la relation entre les différents paramètres et on donne un compromis de paramètres à prendre en considération pour avoir un système sécurisé avev une bonne probabilité.

1.2.5 Conclusion

Le long de cette partie, l'idée du doublement a été manifestée à travaers le fait d'avoir plusieurs râteliers de clés. En effet, cette idée s'est montrée efficace pour augmenter le niveau de sécurité du protocole étudié. Un bon choix du paramètres peut effectivement rendre la tâche difficile à l'adversaire pour récupérer toutes les clès et par conséquant casser le protocole.

1.3 Conception d'un Protocole de Vote Résistant aux Attaques de Déni de Service

OV-Net est un protocole de vote permettant à un ensemble de voteurs de calculer la somme de leurs votes sans révéler aucune information sur la valeur du vote de chacun. Il se déroule en deux tours et nécessite la participation de tous les voteurs pour pouvoir arriver à la somme finale des votes.

Malheureusement, si un ou plusieurs voteurs se désitent (malicieusement ou accidentellement) durant un des tours, on risque de devoir ajouter un tour supplementaire avec le reste des participants ou complètement commencer le protocole du début. Une telle situation peut côuter des retards sérieux. La solution qu'on propose pour résoudre ce problème est d'organiser plusieurs élections à la fois.

1.3.1 OV-Net Parallèl

Comme solution au problème décrit ci-dessus, on propose d'étendre le protocole OV-Net en plusieurs exécutions en organisant plusieurs élections en parallèl, disant M éléctions, et permettre à chaque participant de choisir aléatoirement de participer dans k d'entre-elles. Ce mécaniseme permet par conséquant de tolérer plusieurs participants qui ne répondent pas et de continuer le protocol jusqu'au bout. Cependant, le prix à payer est une perte en confidentialité, une augmentation des calculs et une perte statistique de la précision du décompte.

1.3.2 La Résistance du Protocole OV-Net Parallèl aux Attaques de Déni de Service

Rappelons que chaque voteur participe dans une sélection aléatoire et indépendante d'élections. Cette nouvelle manière d'utiliser le protocole OV-Net auguemente le niveau de robustesse du système de telle sorte que le désistement d'un voteur n'implique pas forcément un échec total du protocole. Cela implique seulement le rejet des élections dans lesquelles il a participées, mais pas toutes les élections du système.

Certes, un échec entier du protocole est bien prévu, et cela arrive quand les sélections du voteurs désistants couvrent toutes les élections. Cependant, la probabilité que cela se produise est faible car chaque sélection est aléatoire et indépendante, et de nombreux voteurs désistants auront des sessions en commun.

1.3.3 Algorithmes de Décompte du Vote

On propose trois algorithmes pour calculer la somme des votes en utilisant le nouveau protocole OV-Net en parallèl. La première approche est de simplement calculer la moyenne des votes de chaque éléction et de redimensionner en tenant compte des sessions perdues. Pour le deuxième et le troisième algorithmes, on propose deux méthodes pour calculer le décompte final: la méthode de la variance minimale et la méthode du zéro biais.

1.3.4 Analyse de la Confidentialité de l'OV-Net Parallèl

Il est clair que la participation d'un voteur dans plusieurs éléctions divulgue plus d'informations sur la valeur de son propre vote, et cela est le prix à payer pour gagner en robustesse. L'analyse se fait tout d'abord en étudiant le cas basique, à savoir pour le cas d'une seule élections, puis on donne la forme générale pour plusieurs élections. Le détail des calculs est donné plus loin dans ce manuscrit.

1.3.5 Conclusion

L'idée du doublement, exprimée par le fait d'avoir plusieurs éléctions à la fois, a effectivement aidé à auguementer la robustesse du stystème, par contre cela a impacté negativement le calcul de décompte et de la confidentialité du vote. Dans ce travail, nous avons vu que l'amélioration d'un paramètre se fait au détriment d'autres.

1.4 Sécurité Taphonomique : Information (ADN) à Durée de Vie Contrôlée

On présente ici le concept d'information à durée de vie prévisible via une nouvelle méthode d'encodage et de stockage de l'information dans des séquences ADN-

ARN. L'objectif est que, au lieu de résister aux effets inéluctables du temps, nous essayons de les exploiter : plutôt que de laisser l'information se détruire lentement et progressivement, nous visons à un effacement rapide et complet de l'information.

1.4.1 Un Support de Stockage Biochimique

Notre moyen de stockage de l'information est l'ADN et l'ARN. Il s'agit d'un support d'information approprié qui nous permet de contrôler la durée de vie du message intégré dans les molécules d'ADN/ARN synthétisées en manipuler le taux de dégradation des nucléotides d'ARN. Étant donné qu'il y a une grande difference entre la vitesse de dégradtion de l'ADN et de l'ARN (l'ARN se dégrade 10^5 fois plus rapidement que l'ADN), on considère dans notre analyse mathématique le long de cette thèse que l'ADN ne se dégrade pas.

1.4.2 La Méthode Proposée

Nous proposons de synthétiser des molécules d'ADN et d'ARN dont la durée de vie peut être approximativement ajustée. Un tel "fusible temporel" peut garantir, par exemple, qu'un secret cryptographique (généralement un texte en clair crypté sous un hachage des informations ADN) ne peut pas être utilisé ou récupéré au-delà d'une certaine date d'expiration.

Notre idée est d'incorporer des fragments d'ARN dans des oligonucléotides d'ADN en utilisant une synthèse standard en phase solide et de produire des séquences chimériques ADN-ARN pour former un nouvel oligonucléotide chimérique ADN/ARN. Cet oligonucléotide chimérique ADN/ARN contiendra k nucléotides d'ARN et $k+1$ fragments d'ADN. Nous synthétisons n copies de cette molécule et la gardons dans un fluide.

1.4.3 Contrôle de la Durée de Vie de l'Information

Afin de contrôler la durée de vie des informations intégrées dans les molécules d'ADN/ARN, nous introduisons un modèle probabiliste et déterminons mathématiquement les bornes sur la durée de vie de l'information. Nous considérons que l'information stockée dans la molécule d'ADN/ARN passe par trois périodes différentes que nous appelons : vie, agonie et mort. Nous décrivons chaque période séparément et donnons le modèle mathématique permettant de déterminer les bornes de chacune d'elles.

1.4.4 Conclusion

Ayant plusieurs copies de la même molécule permet de contrôler efficacement la durée de vie de l'information. Cependant, sur le plan pratique, cela semble difficile à implémenter étant donné que la fabriquation de très longs oligonucléotides est un défi synthétique. Notre analyse théorique fonctionne comme une preuve de principe et la réponse à ces questions de recherche est laissée pour un travail futur.

1.5 Bulletins de Vote Masqués

RLT [25] est un protocole de vote qui consiste à ne révèler qu'un échantillon aléatoire de bulletins de vote. Il a été précédemment proposées pour atténuer certaines menaces de coercition. Le masquage de certains bulletins de vote offre aux électeurs contraints un déni plausible, tandis que les techniques de limitation des risques garantissent que le niveau de confiance requis dans le résultat de l'élection est atteint.

Cependant, ce protocole est jugé non démoctratique parce que les bulletins de vote de certains voteurs ne contribuent pas au décompte final. Pour faire face à ce problème, nous proposons de masquer certaines composantes de chaque bulletin de vote et d'inclure tous les bulletins dans le décompte final, et par consquant, tous les voteurs auront des chances égales pour participer au décompte final et la démocratie sera réalisée.

1.6 Conclusion

Globalement, nous avons vu le long de cette thèse plusieurs protocoles de sécurité avec plusieurs paramètres de sécurité. En fonction de chaque paramètre, nous avons proposé de nouvelles façons d'ajuster et d'optimiser les paramètres afin d'obtenir de meilleures performances de sécurité. Les principales idées utilisées dans ce manuscrit de thèse sont l'adoption des mécanismes de dédoublement et de masquage: plusieurs râteliers pour distribuer les clés, plusieurs élections pour éxecuter le protocole OV-Net, multiples copies des molécules d'ADN-ARN pour encoder l'information, et le masquage d'un sous-ensemble des composants de chaque bulletin de vote.

Chapter 2

Introduction

In this chapter, we motivate the main ideas of our work, and we present some mathematical background as well as some security properties and definitions.

Contents

2.1 Motivation

Security protocols are an important element in the information security field. Analyzing their security is almost an everyday topic of concern to researchers in this field. Improving security protocol performances requires an in-depth study of the impact of each security property on the other and proposing a good balance. Several types of security protocols exist, for example, authentication protocols, secret key exchange protocols, electronic voting protocols, and many others, including storage protocols.

Depending on the type of protocol to be studied, several security property types can be distinguished, which come into play. It is important to ensure messages confidentiality, authentication, and key indistinguishability in key exchange protocols. In e-voting protocols, vote privacy and verifiability are the main examined properties besides tally accuracy, coercion resistance and other properties. This thesis manuscript investigates the possibility of having better protocol security through parameters fine-tunes and trade-offs.

There are two main ideas we are using in this thesis: *doubling* or *multiplying* and *masking*.

In our work in Chapter 3, we conceived a system with multiple key-racks, and the user is assigned a fixed number of keys from each rack. We showed that this technique increased the security level and made the protocol *semi-decentralized*: the central authority participates only in the registration phase but does not engage in the real key distribution. In the second work in Chapter 4, having multiple elections and allowing the voter to vote in a selection of them helped raise the system's robustness. In our work in Chapter 5, having several copies of the same DNA-RNA molecule effectively controls the lifespan information. Finally, the idea of *masking* is used in Chapter 6. We show that masking some components of each ballot optimises the trade-offs between various security measures.

The following sections introduce some mathematical background useful for the theoretical study throughout this thesis without digging into more details.

2.2 Useful Mathematical Background

Mathematics plays an important role in the field of information security and participates in the building blocks of several security primitives. For example, it is widely used in designing digital signatures, hash functions, symmetric and asymmetric protocols... etc. Note that number theory is one of the mathematics disciplines that studies the relations between integers, which helps providing genius solutions for secretly transmitting information. Probability theory is another important discipline of mathematics employed to predict the degree of information protection. It analyses the data distributions and tells us about our solution's degree of certainty and effectiveness.

This section is devoted to the basic mathematical definitions used in this thesis: notions from algebra and probability theory.

2.2.1 Group Theory:

Group theory is extensively used in cryptography (Section 2.3.1). We give the following brief description To understand the notion and some of its properties.

Definition 1 A *group* is a couple $(G, \star)$ such that G is a set of elements, $\star$ is an application from G to G and the following requirements, called *group axioms*, are satisfied:

- $\star$ is *associative*: $\forall (a, b, c) \in G : (a \star b) \star c = a \star (b \star c)$
- $\star$ has the *identity element* e which satisfies: $\forall a \in G : a \star e = e \star a = a$
- each element in G has an *inverse*, i.e. $\forall a \in G, \exists b : a \star b = b \star a = e$. We note $b = a^{-1}$

If moreover $\star$ is *commutative*, i.e. $\forall a, b \in G, a \star b = b \star a$, $(G, \star)$ is called *commutative* or *abelian* group.

Proposition 1 *If $(G, \star)$ is a group, then we have the following statements:*

- *the identity element e is unique*
- $\forall a \in G, a^{-1}$ *is unique*

Proposition 2 *Let $(G, \star)$ be a group. We have then:*

- $\forall a, b, c \in G : ab = ac \Rightarrow b = c$
- $\forall a, b, c \in G : ba = ca \Rightarrow b = c$

PROOF Demonstrations of the previous propositions are straight-forward.

2.2.2 Probability Theory:

Among different research fields, probability theory greatly impacts information security. We must always know our result's degree of certainty and how our data distribution would behave in some circumstances. The following section is dedicated to some notions from probability theory.

Definition 2 A *random experiment* is an experiment that is repeated several times and that gives an unpredicted outcome in each time. We call the set of all the possible elementary outcomes of a random experiment the *sample space*, and we note it by Ω.

We distinguish between *discrete* and *continuous* sample spaces, depending on the nature of the set Ω if it is discrete or continuous.

Definition 3 An *event* is a set of elementary outcomes, which is, in other words, a subset of the sample space Ω.
The *probability* of an event $A \subset \Omega$, denoted by $P(A)$, is a real number obtained by applying the function P, which verifies the following properties:

- $0 \leq P(A) \leq 1$
- $A = \Omega \Rightarrow P(A) = 1$

We distinguish between *dependent* and *independent* events. A and B are two *independent* events if $A \cap B = \emptyset$. In this case, we have:

- $P(A \cup B) = P(A) + P(B)$
- $P(A \cap B) = P(A).P(B)$

If A and B are dependent events, we have:

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

Definition 4 A *random variable* X is a measurable function $X : \Omega \to E$ from a set of possible outcomes Ω to a measurable space E. The probability that X takes on a value in a measurable set $A \subset E$ is written as:

$$P(X \in A) = P(\omega \in \Omega | X(\omega) \in A)$$

Definition 5 If X is a discrete random variable, the *mass function of X (PMF)* is the function that X is exactly equal to some value x: $P(X = x)$.

Definition 6 If X is a continuous random variable, the *density function (PDF)* of X is the function f_X such that:

$$P[a < X < b] = \int_b^a f_X(x).dx$$

Definition 7 The *cumulative distribution function* of a random variable X is the function F_X such that:

$$F_X(x) = \begin{cases} P(X \leq x) & \text{if X is a discrete random variable} \\ \int_{-\infty}^{x} f_X(t).dt & \text{if X is a continuous random variable} \end{cases}$$

Independence concerns random variables as well. For example, if X and Y are two random variables and the distribution of X is not influenced by the values taken by Y, and vice versa, the two random variables are said to be *independent*, and are dependent otherwise.
Formally, two discrete random variables are independent if:

$$\forall x, y \in \Omega : P(X = x, Y = y) = P(X = x).P(Y = y)$$

Some known probability distributions examples and their properties

Researchers in probability and statistics affirm the possibility of mathematically describing and modelling random phenomena through probability distributions. Among the very known probability distributions, we denote the following distributions that we used in our works for this thesis:

- **Binomial Distribution:** Consider n independent Bernoulli experiments, each of which has its own boolean outcome: success or failure. Suppose that p is the probability of success (and $1-p$ is the probability of failure). Then, the number of successes in these n experiments is a discrete random variable, called *binomial distribution* of parameters n and p.
  We note: $X \sim B(n,p)$ if the random variable X follows the binomial distribution with parameters n and p.

- **Multinomial Distribution:** The multinomial distribution is a generalization of the binomial distribution. Consider a k-sided die rolled n times and $X_i, 1 \leq i \leq k$, the random variable modelling the number of success of side i during these n experiments. Denote by p_i the probability of successes of side i. For n independent trials, the multinomial distribution gives the probability of any particular combination of numbers of successes for the different categories.

- **Exponential Distribution:** The *exponential distribution* of parameter λ is a continuous distribution that models a process in which events occur continuously and independently at a constant average rate.
  We write: $X \sim \text{Exp}(\lambda)$ if the random variable X follows the exponential distribution of parameter λ.

The reader is referred to Table 2.1 for more details about the above probability distributions.

Probability Distribution	**PMF / PDF**	**CDF**	**Expected Value**	**Variance**
$X \sim B(n,p)$	$P(X=k) = \binom{n}{k}.p^k.(1-p)^{n-k}$	$P(X \leq k) = \sum_{i=0}^{k-1} \binom{n}{k}.p^i.(1-p)^{n-i}$	$\mathbb{E}[X] = np$	$Var[X] = np(1-p)$
$X = (X_1, \cdots, X_k) \sim M(n, p_1, \cdots, p_k)$	$P(X_1 = x_1$ and $\cdots$ and $X_k = x_k) = \frac{n!}{x_1! \cdots x_k!} \cdot p_1^{x_1} \cdots p_k^{x_k}$ when $\sum_{i=1}^{k} x_1 = n$	$-$	$\mathbb{E}[X_i] = np_i$	$Var[X_i] = np_i(1-p_i)$
$X \sim \text{Exp}(\lambda)$	$f(x,\lambda) = \lambda e^{-\lambda \cdot x}$ if $x \geq 0$	$F(x,\lambda) = 1 - e^{-\lambda \cdot x}$ if $x \geq 0$	$\mathbb{E}[X] = \lambda$	$\text{Var}[X] = \frac{1}{\lambda^2}$

Table 2.1: Some known probability distributions examples and proprieties

2.3 Definitions and Security Properties

Protocols are designed to achieve a predetermined goal. In particular, security protocols are used to ensure data transmission or data storage or securely. Various examples of security protocols, such as key-exchange protocols, e-voting protocols, auction protocols etc, exist. We are interested in this thesis in studying key-exchange and e-voting protocols. Therefore, we give in this section an overview of the main security definitions and properties needed and used in this thesis.

2.3.1 Cryptography

Cryptography is the discipline that studies data protection by ensuring confidentiality, integrity and authentication. It is present constantly in our life to protect our daily transactions such as sending emails, performing online payments, having conversations on our cell phones etc. Using public channels to perform such sensitive transactions without security exposes internet-based activities to serious security problems. Therefore, the use of cryptography becomes inevitably in our daily life.

The main objective of cryptography is to hide the meaning of the stored, transmitted or processed messages. Therefore, we distinguish between two bijective operations: *encryption*, which is the process of rendering a message non-understandable for any opponent, and *decryption*, which is the process of returning the original meaning. The cryptographic protocol that accomplishes these two operations is called cryptosystem. It inputs two parameters: a plaintext and a key, and outputs a ciphertext (for encryption, and vice versa for decryption). Whether the same key is used for both encryption and decryption, we differentiate between symmetric (Figure 2.1, also called private key) and asymmetric (Figure 2.2, also called public key) cryptography.

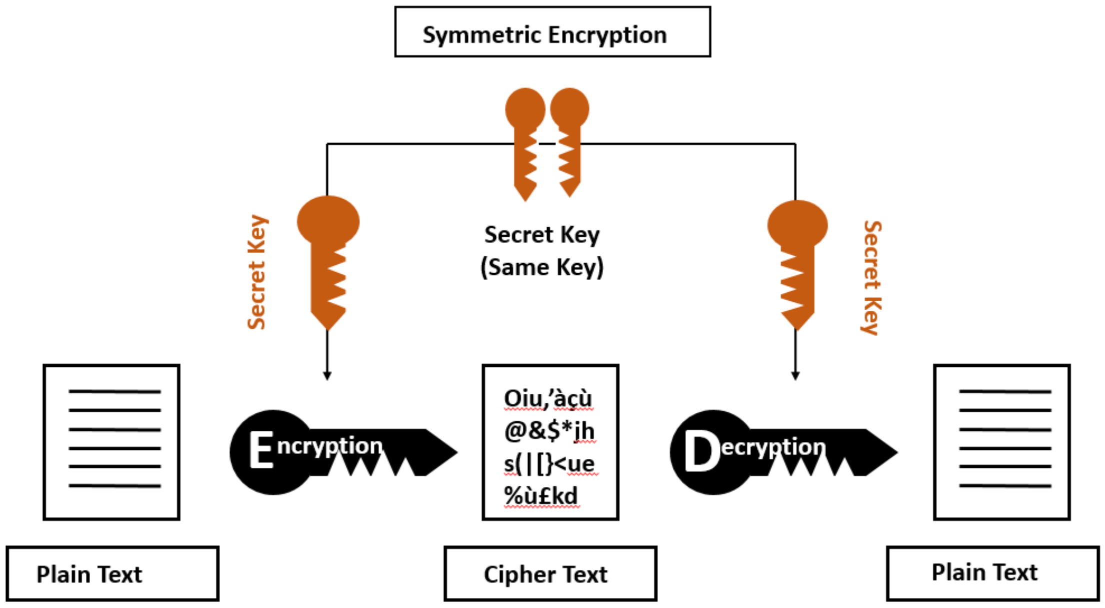


Figure 2.1: Symmetric Encryption Illustration

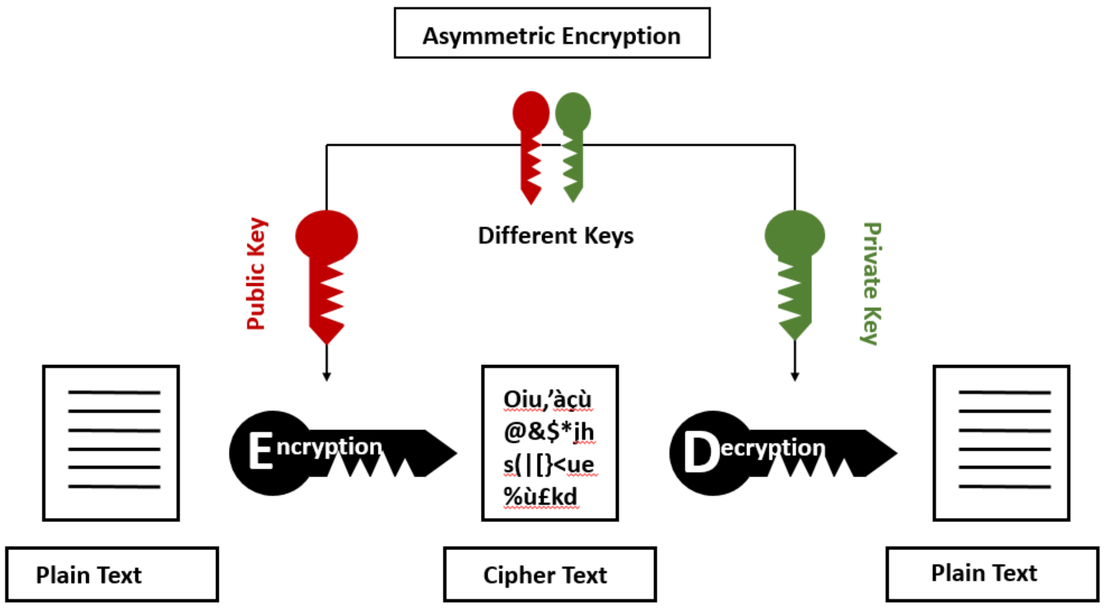


Figure 2.2: Asymmetric Encryption Illustration

The first users of this notion were the Egyptians, 2000 years before Jesus Christ. Then, many years later, Julius Caesar introduced the earliest cryptosystem for encrypting messages. It is a simple mechanism where the ciphertext is obtained by replacing each letter in the plaintext at an equal distance in alphabetical order. This cryptosystem revolutionized the history of cryptography and gave birth to many others. Although we will not go deeper into this point, the reader is referred to [42], for example, for more details. Subsequently, other means came along, such as Jefferson's disk and the German-made Enigma machine.

Cryptography is used to ensure some information security properties, such as confidentiality, integrity and authentication. In the next section, we define the notion of authentication in more detail, which we need in the rest of this thesis, especially for our work in Chapter 3.

Authentication

A service provider protects its services from any unauthorized use by applying some security measures. Among others, we find identification and authentication, steps that precede authorization and giving access. In this part, we are interested in defining what authentication is; then we describe its factors, types, and the methods used in some real case examples.

Authentication is the process that aims to determine whether a person or something is effectively the person or the thing that claims to be. Recall the example of the card with the ATM. In this scenario, a PIN code is requested from the ATM to the user in order to give him access or not. The user should be able to prove to the ATM that effectively he is the cardholder by entering the correct PIN code. The authentication succeeds when the PIN code given by the user is the same as the one stored in the ATM, and fails otherwise. This is a classic example to illustrate how the authentication procedure works.

There are other ways how a user or a computer can be authenticated. Besides password-based authentication, which is a common authentication method, we find multi-factor authentication, certificate-based authentication, biometric authentication, token-based authentication ... etc. These methods and technologies are adopted to prevent some security attacks such as identity falsification and impersonation.

Practically, several protocols to ensure authenticated communication exist. We note the *Secure Socket Layer (SSL)* protocol, *Transport Layer Security (TLS)* protocol, *Kerberos* protocol etc. In this thesis, we present a new authenticated key-exchange protocol, post-quantum secure, a notion we describe in the next section.

Post-Quantum Cryptography

The security of most security protocols used in our daily lives relies on some mathematical problems that the ordinary computer cannot solve. Even though the current high-performance computer cannot, for example, factorize a product of two big size integer numbers (e.g. 2048) nor determine the discrete logarithm of an integer in a cyclic group in polynomial time.

However, with the research progress on the possibility of having a quantum computer able to solve these problems in polynomial time, the security of such protocols becomes threatened. Therefore, other solutions have been proposed to deal with the possible security holes resulting from the use of quantum computers, such as adopting protocols based on lattices, codes, multivariate, hashes and isogenies, which is the core of post-quantum cryptography.

This seems very resistant to any kind of exploitation by the quantum computer and is supposed to ensure a high level of security. A protocol called *post-quantum* secure is a protocol such that even the quantum computer with its formidable computing power cannot break the security. This notion has become a hot topic nowadays and is of great interest to researchers in the information security field. We find in the literature several attempts to build post-quantum secure protocols, and the NIST agency encourages research in this field.

In our work in Chapter 3, we present a new key-exchange protocol based on simple construction, making it a post-quantum secure protocol.

2.3.2 Key Exchange Protocols

A key is a bit string of letters, numbers, or special characters. It purposely has no meaning and is generated using a key generator, which can be a random number generator (RNG) or a pseudorandom generator (PRNG). A key is characterized by its size and serves for encrypting and decrypting messages. There exists a difference between secret keys and passwords. Unlike a key, a password is often created to be memorized by the human and it is a part of his identity. Usually, secret key security is stronger than password security due to its low entropy, randomness, and human-readable properties. Generally, a secret key can help strengthen password security.

In what follows, we define a key exchange mechanism and its usefulness.

Definition and Example

A key exchange protocol is a mechanism for distributing secret keys between Alice and Bob to derive a common key based on their keys securely. This notion was firstly introduced by Diffie and Hellman [16] in 1976 and is commonly used to build a secure communication channel between Alice and Bob over a public channel in order to exchange secret messages. This channel allows Eve to observe the protocol's execution, but she cannot infer useful information. This mechanism is explained using colors as a typical example [1]. To come up with a secret common color, Alice and Bob do the following:

1. Agree on a random and common color to start with.
2. Decide their color without informing the other party about it.
3. Combine their secret color with the common color.
4. Send the resulted color to the other party.
5. Add their secret color with the received color.
6. As a result, they both come out with the same color (which is the color "Brown" in our example in Figure 2.3).

The example in Figure 2.3 shows that Alice and Bob both end up with the same color without even sending their secret keys in plaintext. Thus, if Eve listens to their exchanges, she cannot find the two colors used to get the mixed color (Brown).

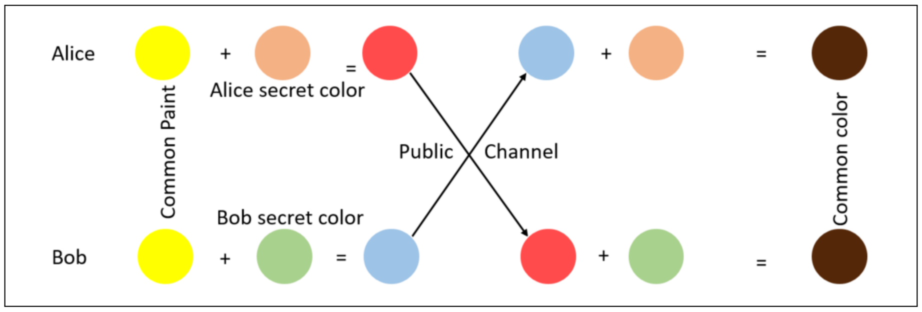


Figure 2.3: Typical example for key exchange using colors

Diffie-Hellman Key Exchange:

It is possible to mathematically express the idea of the example in Figure 2.3 using number theory. To this end, Diffie and Hellman came up with the following scenario. Let G be a cyclic group of order q and generated by g. Similarly to the example above, Alice and Bob do the following:

1. First, Alice chooses a random integer x from the set $[1, q-1]$ and computes $u = g^x$.
2. Then, Alice sends u to Bob.
3. Bob receives u from Alice, chooses a random integer y from the set $[1, q-1]$ and computes $v = g^y$ and $k_B = u^y$.
4. Bob sends k_B to Alice.
5. Alice receives k_B from Bob and computes k_A.
6. Alice and Bob both take $k_A = k_B$ as the protocol's output.

Alice		Bob
$x \in_R [1, q-1]$		
$u \leftarrow g^x$		
	$\xrightarrow{\text{Alice },u}$	$y \in_R [1, q-1]$
	$\xleftarrow{\text{Bob },v}$	$v \leftarrow g^y$
$k_A \leftarrow v^x$		$k_B \leftarrow u^y$
Outputs k_A		Outputs k_B
	$k_A = k_B$	

Table 2.2: Diffie-Hellman Key Exchange Illustration

The benefit of running a key exchange protocol is to develop a common secret key between the participants. For instance, this is very useful when we want to use symmetric cryptography since the key, in this case, is a sensitive element that requires a secure exchange.

2.3.3 E-Voting Protocols

Voting is the process of agreeing on a collective decision between a group of persons. It is used in many areas, such as politics, to ensure democracy through decision-making and electoral systems or referendums, in meetings and gatherings, or deliberate assemblies. There are several methods on how to vote. The conventional forms of vote include paper ballots and hand-counting starting from the beginning. These methods are considered trustworthy since they assure integrity and secrecy of the vote, the fundamental guarantees for a healthy democracy.

Then, with the advances in communication and information-sharing technologies, voting has become an electronic operation accomplished through the Internet. Even though this seems to bring huge benefits in terms of efficiency, unfortunately, it increases many security risks. The vote fundamental guarantees, secrecy and integrity, become too vulnerable to online attacks. Therefore, current research leans towards the design of electronic voting (e-voting in short) protocols satisfying some security properties such as privacy, verifiability, coercion resistance or receipt freeness.

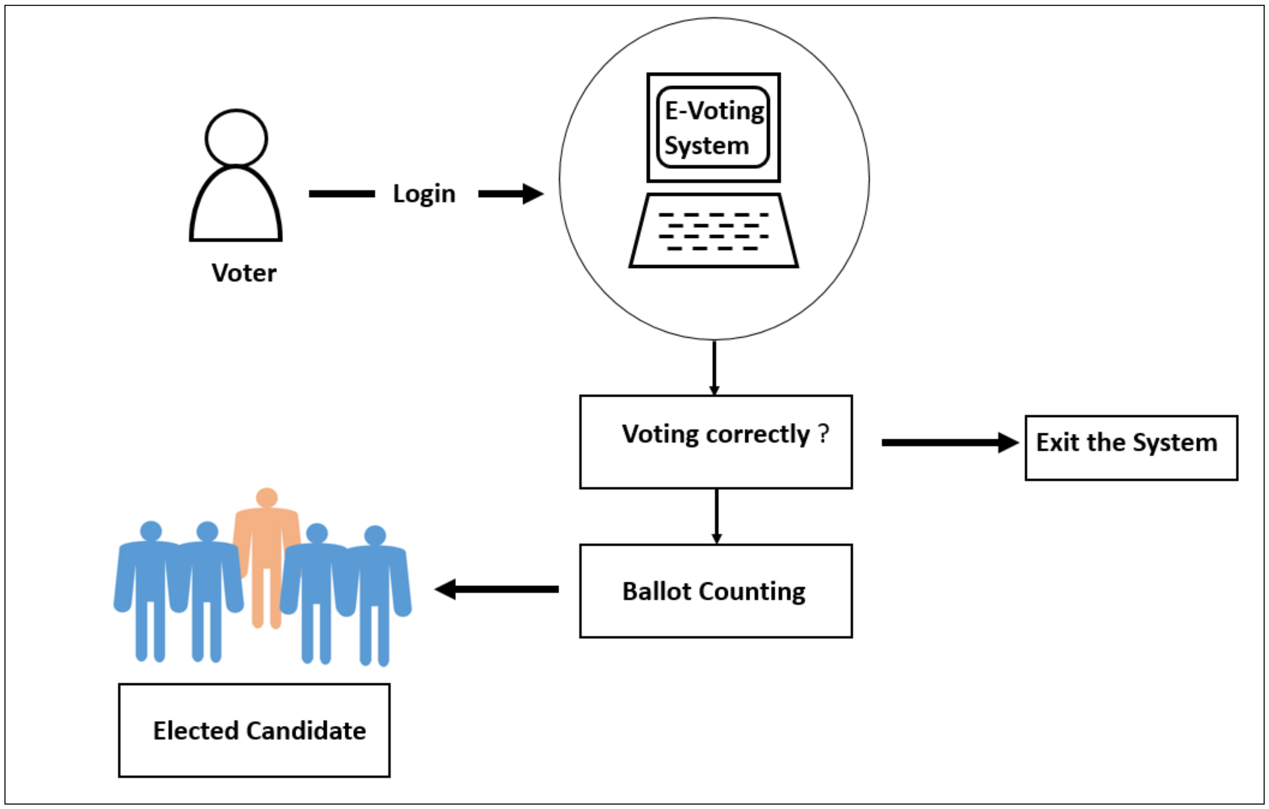


Figure 2.4: The E-Voting process

Vote Privacy

Vote privacy is one of the important security properties an e-voting protocol should ensure. It is considered a fundamental human right and allows the voter to keep the secrecy of her vote entirely; no one should know who this voter voted for.

In other words, a protocol ensuring the privacy of the vote is a protocol guaranteeing that a voter who voted in a specific way cannot be revealed to anyone. However, this is not true in all cases if we consider when all voters voted for the same candidate.

Taking the example of a scenario where the voter is asked to vote ”Yes” or ”No” for a candidate, Küsters in [36] presented a formal definition of vote privacy, denoted by δ. It says that the adversary should be unable to distinguish whether the voter voted ”Yes” or ”No” by any strategy adopted by the adversary. A good e-voting protocol for privacy-preserving should have δ close to the one of the ideal protocol.

Other derivations of the notion of vote privacy comprise:

- Ballot-secrecy: the system should not reveal what the voter voted for.
- Receipt-freeness: the voter should not prove how and what she voted to any third party. To this end, the system should not give any evidence to the voter in this regard.
- Coercion-Resistance: the voter votes the way she chooses without being obliged to obey the coercer's instruction even if it seems to cooperate with him.

The importance of the notion of vote privacy also depends on the application domain. For instance, e-voting protocols for presidential elections are much more important to preserve than decision-making applications. It is important to preserve the secrecy of a voter voting for the presidential elections to prevent vote-buying. In Chapter 4, the protocol we are designing is well-fitting for decision-making applications.

Vote Verifiability

This notion is studied more in our work in Chapter 6 and remains an essential notion to know when discussing e-voting protocol. Therefore, we give a brief definition of it, the reader is referred to [23] for more details.

Either way in the electronic or the paper ballot voting schemes, the voter should be able to verify her vote to trust the integrity of the election results issued by the voting machines and the polling staff. Some existing mechanisms in real-world elections include exit polls, random audits and opening the tallying process to the public. Contrarily, research in the literature focuses more on ensuring vote verifiability by providing the voter with solutions based on the cryptography trying to detach her from any personnel or mechanical dependability. For example, the work of Sako and Kilian in [50] indicates that we can distinguish between two forms of vote verifiability:

- **Individual verifiability:** ensures that the voter can verify that her vote belongs to the set of all votes.
- **Universal verifiability:** allows an observer to verify that all the votes were correctly included in the final tally.

Other Vote Properties

In addition to the vote properties mentioned above, there exist other important ones that we give in the following:

- Eligibility verifiability: an observer can verify the voting eligibility of a voter whose vote belongs to the set of all cast votes.
- Accountability: the voter should be responsible for proving the vote verification failure, which can happen due to error or fraud, to the authority without compromising her ballot secrecy.

- Robustness: the system should be resistant enough to malfunction and still deliver a correct result.
- Usability: the voter can easily and effectively use the voting system.
- Accessibility: the system should grant the same access and participation opportunities, especially for vote privacy and verifiability, to voters with disabilities.

2.4 Contributions

The first contribution concerns the design of a new authenticated key-exchange protocol. The particularity of this new protocol is to not rely on mathematical and number-theoretic cryptography and therefore is seen to be post-quantum secure. Instead, it entirely relies on symmetric primitive and approaches some of the functionalities of public-key cryptography. We show that we reach a good security level with our simple constructions by handing out more keys per user.

The second contribution consists of designing a new version of the protocol OV-Net, making it resistant to DoS attacks. We show that having multiple elections increases the computational complexity level but improves the protocol's robustness. We investigate the effect of parallel OV-Net on the different security parameters: time, privacy, robustness and accuracy. We stress that our protocol is well suited for decision-making applications.

The third contribution is the proposition of a new security primitive. Having information with a foreseeable lifespan is a new method for storing information and guaranteeing its destruction after a certain time. The storage is done on DNA-RNA molecules. We provide in this work a theoretical study of the method aging model. We consider that the information goes through three different periods of life: life, agony and death, and we define its security based on this consideration. We present three different algorithms on how to control the information lifetime.

The fourth contribution discusses the idea of masking voting ballots. We propose an optimization of the RLT [25] protocol. We investigate the implications of masking some parts of each ballot on privacy, coercion and vote-selling, and how it can alleviate these issues.

2.5 General Structure

The outline of this thesis is divided into 7 chapters:

Chapter 2: Introduction In this chapter, we introduce our work's ideas and overview of the different mathematical notions and security definitions and properties used in this thesis. The knowledge from this chapter is not new and serves as a theoretical background to understand the rest of the thesis.

Chapter 3: A New Authenticated Key Exchange Protocol In this chapter we present a new way of exchanging secret keys between participants in an authenticated way without relying on number-theoretic cryptography. This makes the protocol post-quantum secure.

Chapter 4: Design of a DoS-Resistant Voting Protocol This chapter presents a new version of the OV-Net protocol, which is DoS-resistant. Allowing the voter to vote in a random selection of elections reduces the risk of stopping the process (maliciously or accidentally) without reaching the final vote.

Chapter 5: Taphonomical Security: (DNA) Information with Foreseeable Lifespan In this chapter, we present a new method for storing information and guaranteeing its destruction after a certain time. We conjecture that information with a foreseeable lifespan is a new security primitive that can serve as a security building block. Our work works as proof of principle by being aware of the biological limitations behind the method construction.

Chapter 6: Masked Ballots In this chapter, we discuss the idea of masking voting ballots. We propose an optimization of the RLT [25] protocol. Then e investigate the implications of masking some parts of each ballot on privacy, coercion and vote-selling, and how it can alleviate these issues.

Chapter 7: Conclusion In this chapter, we summarize the different ideas we presented, mention the results we achieved in this thesis, and we propose some new directions for future work.

Chapter 3

A New Authenticated Key Exchange Protocol

This work was done in collaboration with: David Naccache, Peter B. Rønne, Diana Maimuţ and Peter YA Ryan. It was published at the SecITC 2019 in Bucharest, Romania. We reproduce in this chapter the paper [9].

Contents

Summary

In this work, we introduce new authenticated key exchange protocols which, on the one hand, do not resort to standard public key setups with corresponding assumptions of computationally hard problems but, on the other hand, are more efficient than distributing symmetric keys among the participants. To this end, we rely on a trusted central authority distributing key material whose size is independent of the total number of users, and which allows the users to obtain shared secret keys. Furthermore, we analyze the security of our construction, considering various attack models. Importantly, only symmetric primitives are needed in the protocol making it an alternative to quantum-safe key exchange protocols, which rely on hardness assumptions.

Keywords: Symmetric cryptography, key exchange protocol, authentication, provable security, post-quantum cryptography.

3.1 Introduction

Symmetric key primitives are the preferred choice for fast encryption applications. On the other hand, public-key cryptography is widely adopted for ensuring (authenticated) key exchange functionalities. As a result, many currently deployed applications take the best of both worlds and use key encapsulation mechanisms where keys are exchanged using public key protocols and are subsequently used as input to efficient symmetric primitives.

In this chapter, we propose an intermediate construction. We introduce a cryptographic protocol approaching some of the functionalities of public-key encryption while relying entirely on *symmetric* primitives. Before we proceed, we stress that our models are very different from those of classical public-key cryptography and their security and efficiency metrics. However, it appears that *in many practical settings*, the proposed constructions can successfully replace classical public-key encryption.

Given that our techniques do not resort to number-theoretic cryptography, the construction is naturally resistant to attacks from quantum computers.

3.2 Related Work

Key exchange protocols play an important role in protecting end-to-end communications. Initially introduced in [15], the previously mentioned notion revolutionized cryptology. These protocols allow two parties to generate securely a common secret key, which will be used later for different cryptographic purposes such as sending authenticated and encrypted messages. Another closely related flavour of such protocols may be defined as authenticated key exchange protocols. The first basic understandings of this category of schemes were presented in [5, 6]. Considering that such constructions could lead to practical and efficient protocols, the authors focused on formalizing the security notions related to entity authentication and key distribution.

Contrary to the Needham-Schroeder symmetric key protocol [44], the central authority is only active in the enrolment phase in our protocol, not during the actual key establishment.

ID-based secret key cryptography was first presented in [27]. While the paradigm similarity between this paper and [27] is obvious (*i.e.* mimicking public-key cryptography with symmetric primitives), the technical details are different and granular. We stress that even though [27] introduces applications like a challenge-response authentication protocol and an ID-based MAC algorithm, it does not provide an in-depth security analysis. Moreover, our key exchange protocol can use more than one key per user, which, as we will see, allows us to optimize security non-trivially.

3.3 The Protocol

Participants. Let n be the number of the users in the system (n can be very large, for instance, a billion), each with a unique identity ID_i, where $i \in [1, n]$. In the following, ID_i will designate both the (alphanumeric) name of user i and the user itself as a physical entity. The proposed protocol relies on a central authority (CA) which creates r key tables (called "racks"), each containing ℓ random κ-bit keys. CA distributes, to each user, u distinct keys chosen randomly from each rack, *i.e.* $u \times r$ keys per user. CA also provides each user with supplementary key material described later. Figure 3.1 illustrates the protocol flow.

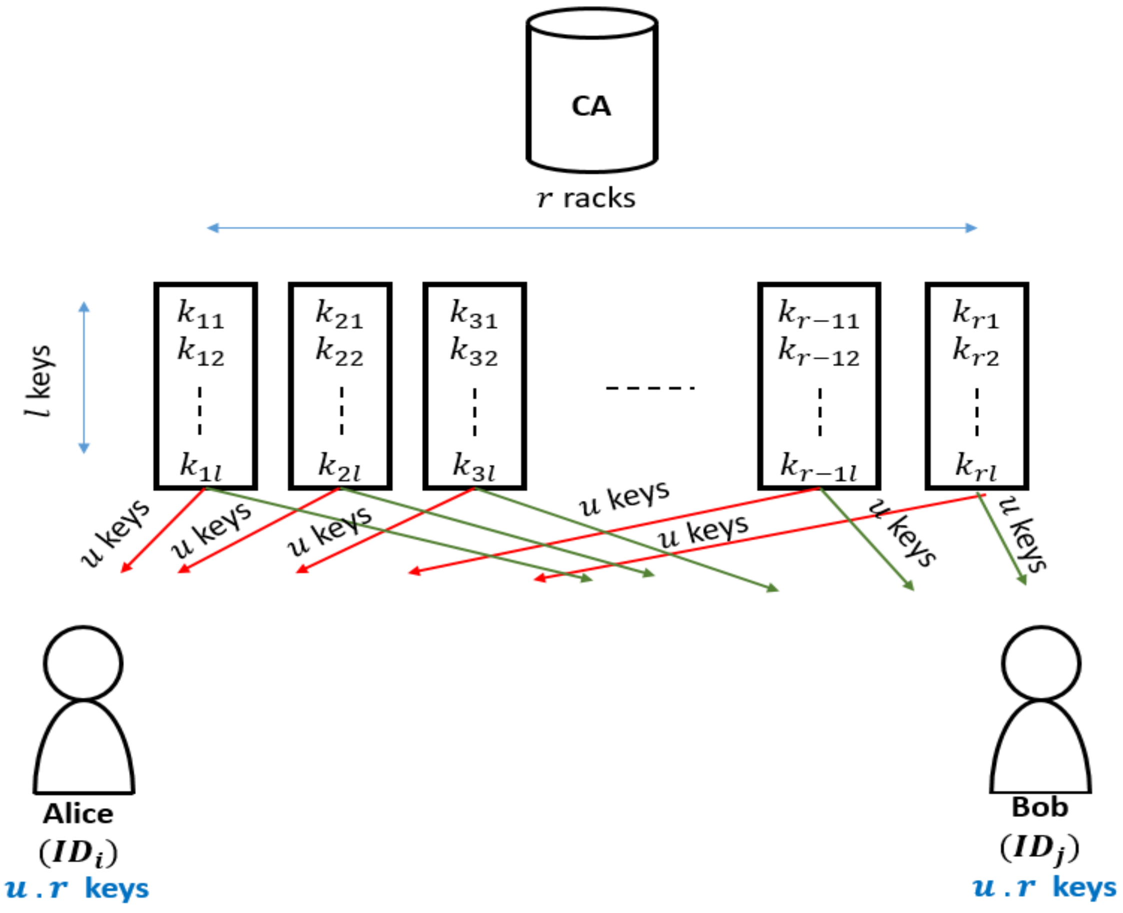

CA
r racks
l keys
k_{11} k_{12} k_{1l}
k_{21} k_{22} k_{2l}
k_{31} k_{32} k_{3l}
k_{r-11} k_{r-12} k_{r-1l}
k_{r1} k_{r2} k_{rl}
u keys
Alice
(ID_i)
$u.r$ keys
Bob
(ID_j)
$u.r$ keys

Figure 3.1: Protocol illustration

Building-Blocks. Let $f(k, m)$ be a MAC function, where k is the key and m is the message. The protocol also uses a hash function h.

For the sake of clarity, we describe the protocol in steps. We first consider and analyze a basic one-rack case ($r = 1$) and one key per user ($u = 1$).

3.3.1 Basic Scheme: ($r = 1$ and $u = 1$)

Key Generation.

CA generates one rack of ℓ secret keys: $\{k_1, \ldots, k_\ell\}$.[1].

User Enrolment.

CA then gives to ID_i:

- A secret key $k_{I(i)}$, where $I(i) \in_R [1, \ell]$;
- A table T_i containing the ℓ derived keys: $T_i = \{t_{i,1}, \ldots, t_{i,\ell}\}$ where $t_{i,j} = f(k_{I(j)}, \mathrm{ID}_i)$.

Figure 3.2 illustrates the process.

[1]In practice, ℓ is much smaller than n. For $n \simeq 10^9$, we may take $\ell \simeq 100$.

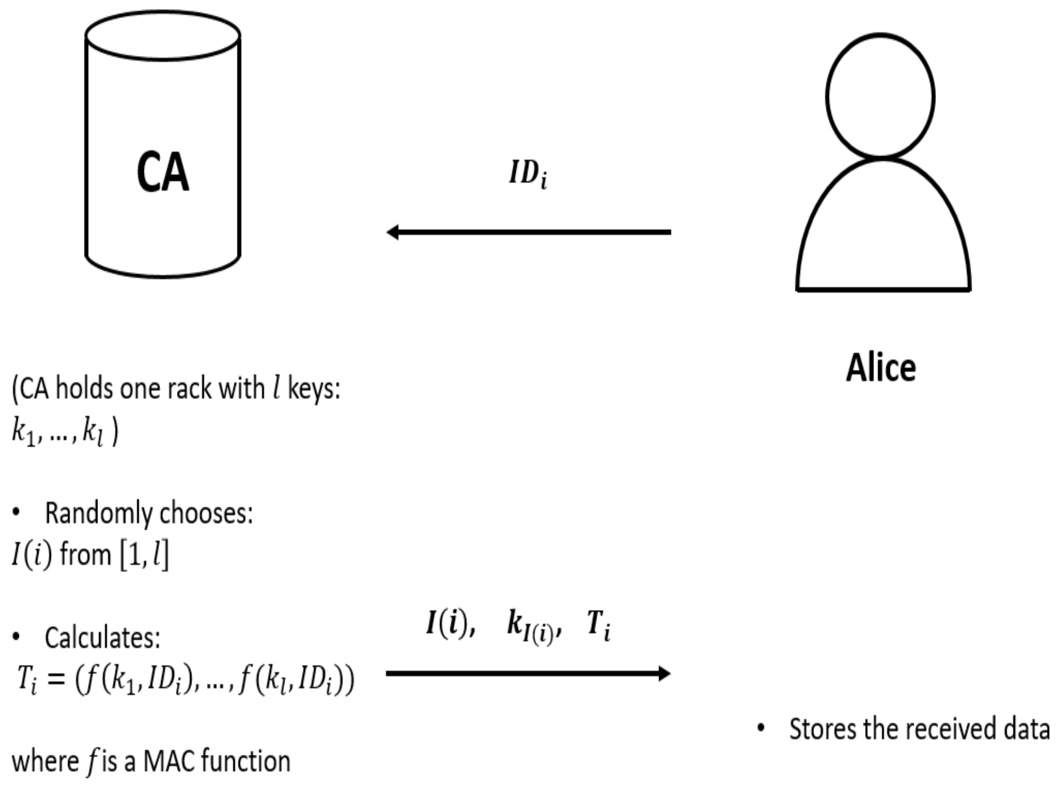

CA
ID_i
Alice
(CA holds one rack with l keys: $k_1, \dots, k_l$)
• Randomly chooses: $I(i)$ from $[1, l]$
• Calculates: $T_i = (f(k_1, ID_i), \dots, f(k_l, ID_i))$
where f is a MAC function
$I(i)$, $k_{I(i)}$, T_i
• Stores the received data

Figure 3.2: User Enrollment for the Basic Scheme

Remark 1 Two users, ID_i and ID_j, may get (and in reality are actually expected to get) from CA the same $k_{I(i)} = k_{I(j)}$. Note, however that $T_i \neq T_j$ as key tables are derived from identities.

Key Exchange.

Assume now that users i and j want to establish a secure communication channel Figure 3.3. They proceed as follows:

1. Exchange $I(i)$ and $I(j)$;
2. User j generates $t_{i,I(j)} = f(k_{I(j)}, \mathrm{ID}_i)$;
3. User i generates $t_{j,I(i)} = f(k_{I(i)}, \mathrm{ID}_j)$;
4. Both users generate the common key $\mathsf{sk} = h(t_{i,I(j)}, t_{j,I(i)})$ and use sk to protect their communications.

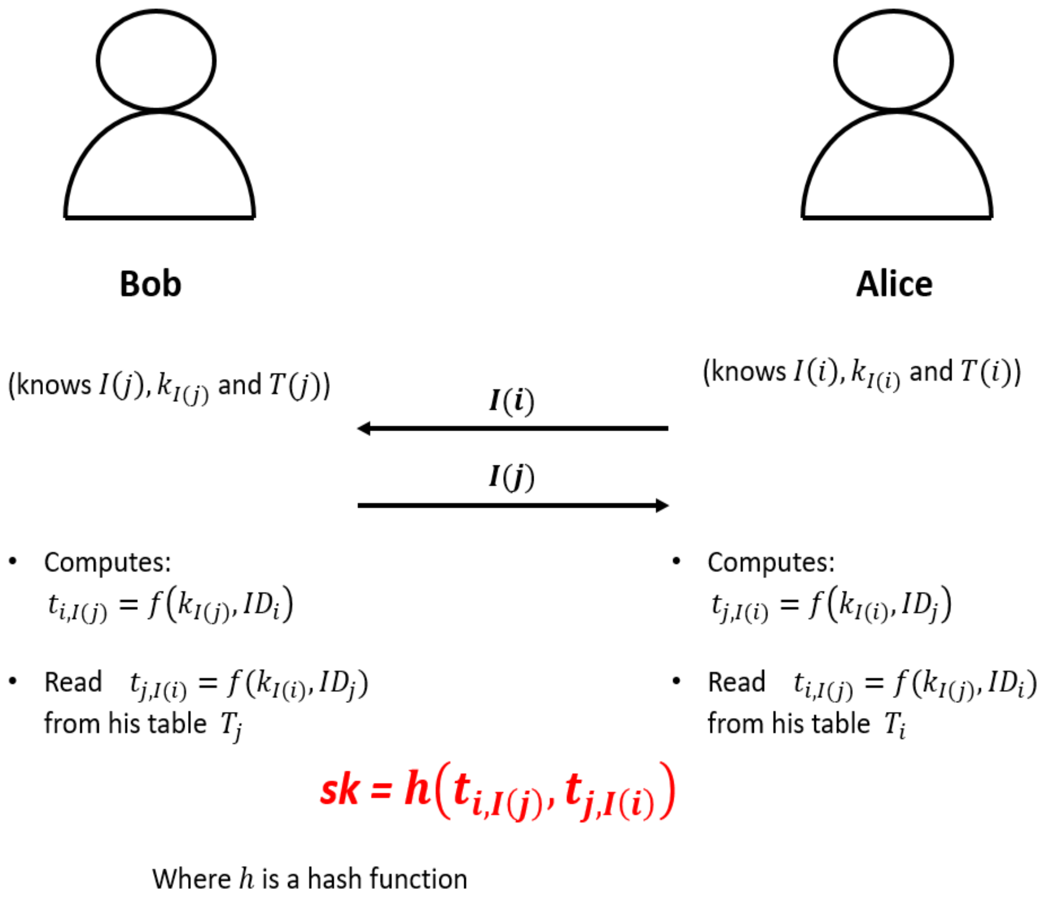

Bob
Alice
(knows $I(j), k_{I(j)}$ and $T(j)$)
(knows $I(i), k_{I(i)}$ and $T(i)$)
$I(i)$
$I(j)$
• Computes: $t_{i,I(j)} = f\left(k_{I(j)}, ID_i\right)$
• Read $t_{j,I(i)} = f(k_{I(i)}, ID_j)$ from his table T_j
• Computes: $t_{j,I(i)} = f\left(k_{I(i)}, ID_j\right)$
• Read $t_{i,I(j)} = f(k_{I(j)}, ID_i)$ from his table T_i
$sk = h\left(t_{i,I(j)}, t_{j,I(i)}\right)$
Where h is a hash function

Figure 3.3: Key Exchange for the Basic Scheme

Remark 2 To avoid ambiguities in the order of parameters of h, we assume that $\text{ID}_i > \text{ID}_j$.

Informally, here is the intuition behind this protocol: we first note that to gain the capacity to listen in to *all* communications; an opponent would need to set his hands on *all* the k_is; this assumes compromising at least ℓ *chosen* devices. Indeed, if at least $t_{i,I(j)}$ or $t_{j,I(i)}$ is unknown, sk is still safe. Evidently, this is not as satisfactory as classical public-key cryptography. Nonetheless, the achieved protection is still useful in many practical scenarios where choosing the target ID_i is impossible[2]. The number of compromised devices required for learning all the ℓ keys with a given probability p is known as the coupon collector's problem (cf. *infra*).

[2]For instance, if the ID_is are identity cards, the attacker needs to collect and compromise enough cards hoping to complete his collection of k_is.

The coupon collector's problem is a famous question introduced in graduate probability lectures. If each box of cookies contains a coupon, and there are ℓ different coupons, what is the probability that more than t boxes need to be bought to collect all ℓ coupons? An alternative statement is: Given ℓ coupons, how many coupons do you expect you need to draw with a replacement before having drawn each coupon at least once? The mathematical analysis of the problem reveals that the expected number of trials needed grows as

$$\ell \log(\ell) + \gamma\ell + \frac{1}{2} + O(\frac{1}{\ell}) \text{ where } \gamma = 0.57721\ldots$$

For example, when $\ell = 50$, it takes about 225 trials on average to collect all 50 coupons. We hence see that the defender enjoys *a little advantage* over the attacker. However, can this advantage be amplified by engaging in several draws? This is the goal of the next sections.

3.3.2 General Case: $r \geq 1$ and $u \geq 1$

In this scenario, each user gets u distinct keys per rack. The function I is therefore generalized by taking three indices:

- ① i denoting the concerned user
- ② ρ denoting the rack
- ③ μ an index running from 1 to u

In other words, $k^{\rho}_{I(i,\mu,\rho)}$ denotes that the μ-th key from rack ρ is given to user i. Note that $k_{I(i)}$ defined in the previous section just corresponds to $k^{1}_{I(i,1,1)}$.

Key Generation:

CA generates r racks of ℓ distinct keys: $R_\rho = \{k^\rho_1, \ldots, k^\rho_\ell\}$, where $\rho \in [1, r]$.

User Enrolment.

CA gives to user ID$_i$:

- $u \times r$ secret keys:

$$\begin{array}{cccc}
k^1_{I(i,1,1)} & k^2_{I(i,1,2)} & \cdots & k^r_{I(i,1,r)} \\
k^1_{I(i,2,1)} & k^2_{I(i,2,2)} & \cdots & k^r_{I(i,2,r)} \\
\vdots & \vdots & & \vdots \\
k^1_{I(i,u,1)} & k^2_{I(i,u,2)} & & k^r_{I(i,u,r)}
\end{array}$$

where $\forall \rho \in [1, r], \forall \mu \in [1, u], I(i, \mu, \rho) \in_R [1, \ell]$

- A table T_i of $\ell \times r$ derived keys:

$$T_i = \begin{bmatrix} t_{i,1}^1 & t_{i,1}^2 & \cdots & t_{i,1}^r \\ t_{i,2}^1 & t_{i,2}^2 & \cdots & t_{i,2}^r \\ \vdots & \vdots & & \vdots \\ t_{i,\ell}^1 & t_{i,\ell}^2 & \cdots & t_{i,\ell}^r \end{bmatrix}$$

  where $\forall \rho \in [1, r], \forall j \in [1, \ell], t_{i,j}^{\rho} = f(k_j^{\rho}, \mathrm{ID}_i)$

Figure 3.4 illustrates the process.

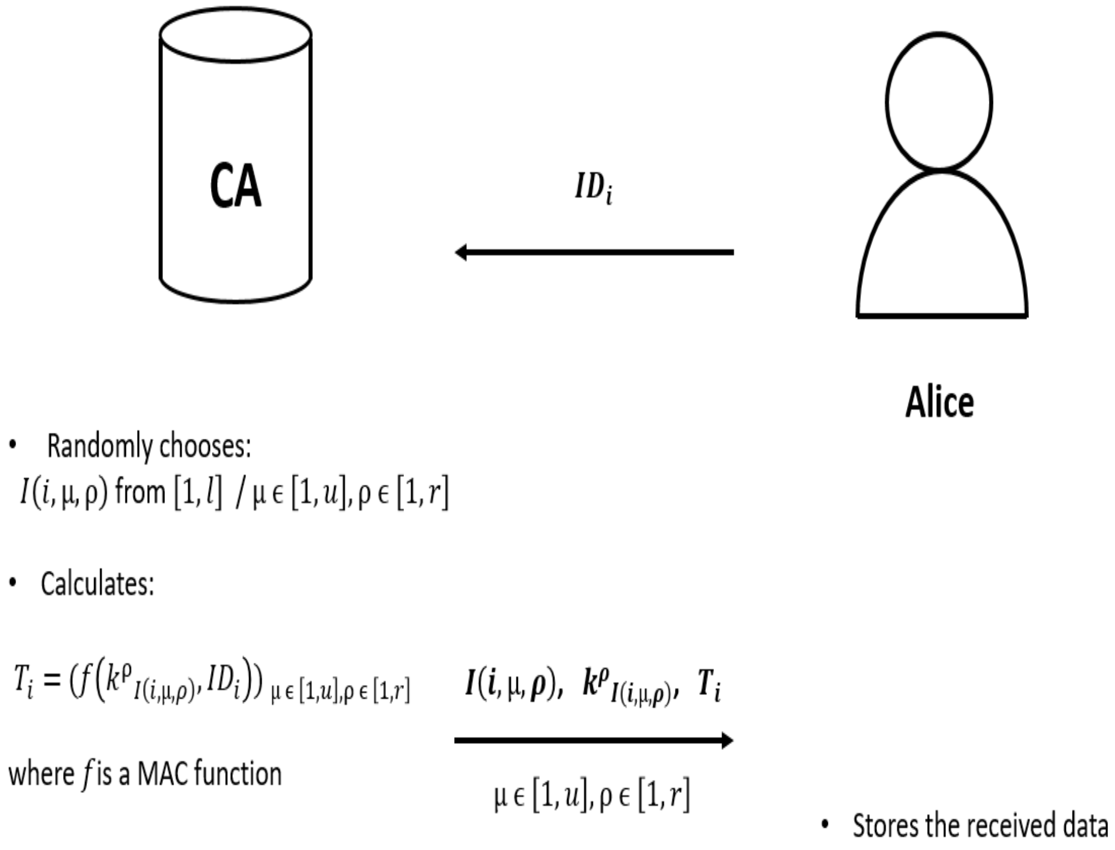

CA
ID_i
Alice
• Randomly chooses:
$I(i, \mu, \rho)$ from $[1, l]$ / $\mu \in [1, u], \rho \in [1, r]$
• Calculates:
$T_i = (f(k^{\rho}{}_{I(i,\mu,\rho)}, ID_i))_{\mu \in [1,u], \rho \in [1,r]}$
where f is a MAC function
$I(i, \mu, \rho)$, $k^{\rho}{}_{I(i,\mu,\rho)}$, T_i
$\mu \in [1, u], \rho \in [1, r]$
• Stores the received data

Figure 3.4: User Enrollment for the General Scheme

Remark 3 Note that the user can derive the table values for his own keys and, in principle, does not need to store these. This way, memory can be saved at the cost of computational efficiency during key derivation.

Key Exchange:

Let's assume now that users ID_i and ID_j want to establish a secure communication channel. To generate their common secret key, they do the following:

1. Exchange their indices $I(i,\mu,\rho)$ and $I(j,\mu,\rho)$ for $\mu \in [1,u], \rho \in [1,r]$;

2. User ID_i:

   - generates $u \times r$ derived keys:

   $$t^{\rho}_{j,I(i,\mu,\rho)} = f(k^{\rho}_{I(i,\mu,\rho)}, \mathrm{ID}_j), \quad \forall\mu \in [1,u], \quad \forall\rho \in [1,r]$$

   - reads $u \times r$ derived keys from his table T_i:

   $$t^{\rho}_{i,I(j,\mu,\rho)} = f(k^{\rho}_{I(j,\mu,\rho)}, \mathrm{ID}_i), \quad \forall\mu \in [1,u], \quad \forall\rho \in [1,r]$$

3. User ID_j:

   - generates $u \times r$ derived keys:

   $$t^{\rho}_{i,I(j,\mu,\rho)} = f(k^{\rho}_{I(j,\mu,\rho)}, \mathrm{ID}_i) \quad \forall\mu \in [1,u], \quad \forall\rho \in [1,r]$$

   - reads $u \times r$ derived keys from his table T_j:

   $$t^{\rho}_{j,I(i,\mu,\rho)} = f(k^{\rho}_{I(i,\mu,\rho)}, \mathrm{ID}_j) \quad \forall\mu \in [1,u], \quad \forall\rho \in [1,r]$$

4. Both users ID_i and ID_j generate a common session key by using h to combine the $2u \times r$ derived keys:

   $$\mathsf{sk} = h\big(t^{\rho}_{i,I(j,1,1)}, \ldots, t^{\rho}_{i,I(j,u,r)}, t^{\rho}_{j,I(i,1,1)}, \ldots, t^{\rho}_{j,I(i,u,r)}\big).$$

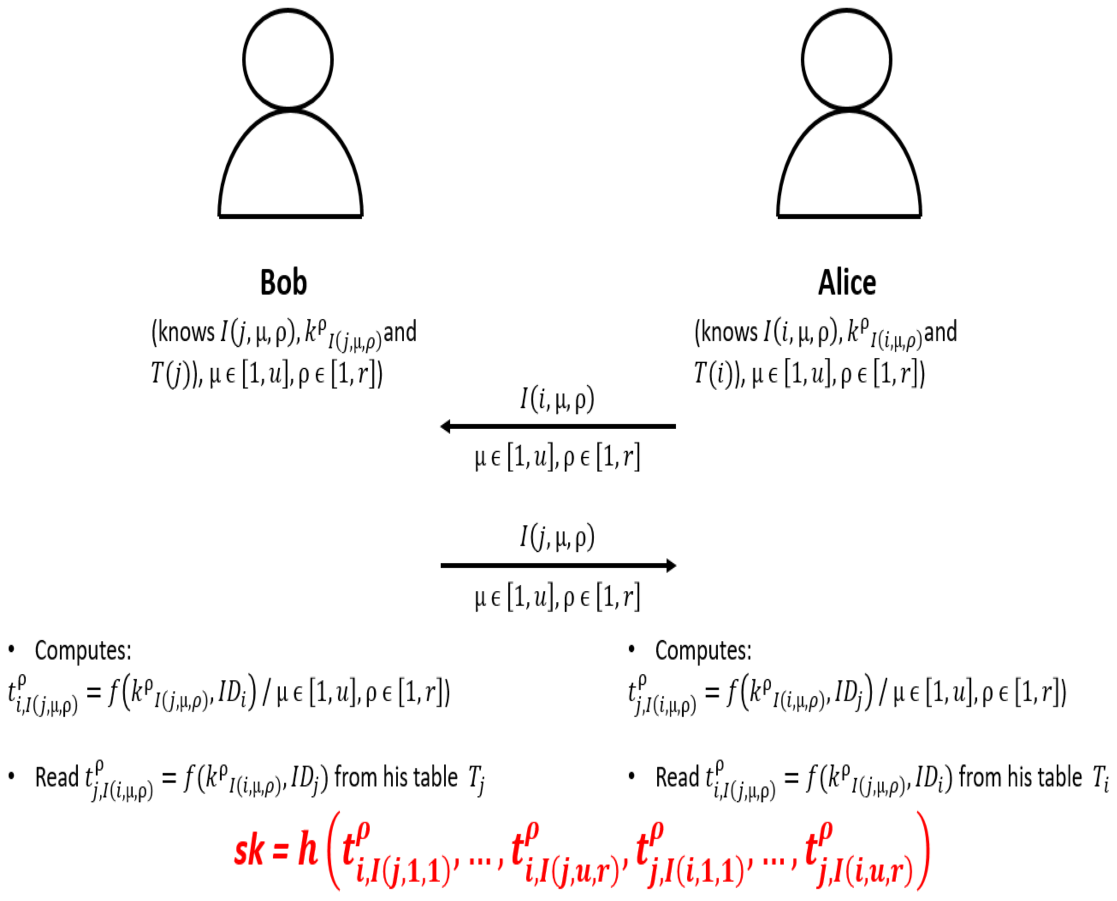

Bob
(knows $I(j,\mu,\rho)$, $k^{\rho}_{I(j,\mu,\rho)}$ and $T(j)$), $\mu \in [1,u], \rho \in [1,r]$)
Alice
(knows $I(i,\mu,\rho)$, $k^{\rho}_{I(i,\mu,\rho)}$ and $T(i)$), $\mu \in [1,u], \rho \in [1,r]$)
$I(i,\mu,\rho)$
$\mu \in [1,u], \rho \in [1,r]$
$I(j,\mu,\rho)$
$\mu \in [1,u], \rho \in [1,r]$
• Computes: $t^{\rho}_{i,I(j,\mu,\rho)} = f\left(k^{\rho}_{I(j,\mu,\rho)}, ID_i\right) / \mu \in [1,u], \rho \in [1,r]$)
• Read $t^{\rho}_{j,I(i,\mu,\rho)} = f(k^{\rho}_{I(i,\mu,\rho)}, ID_j)$ from his table T_j
• Computes: $t^{\rho}_{j,I(i,\mu,\rho)} = f\left(k^{\rho}_{I(i,\mu,\rho)}, ID_j\right) / \mu \in [1,u], \rho \in [1,r]$)
• Read $t^{\rho}_{i,I(j,\mu,\rho)} = f(k^{\rho}_{I(j,\mu,\rho)}, ID_i)$ from his table T_i
$sk = h\left(t^{\rho}_{i,I(j,1,1)}, \dots, t^{\rho}_{i,I(j,u,r)}, t^{\rho}_{j,I(i,1,1)}, \dots, t^{\rho}_{j,I(i,u,r)}\right)$

Figure 3.5: Key Exchange for the General Scheme

3.4 Adversarial Advantage

This section considers an adversary who has corrupted n_c out of the n users and obtained their key material, *e.g.* by physically attacking the IoT devices containing those keys. The corruption can happen before the user gets the key material or afterwards; however, the main assumption of this section is that the indices of the stolen keys are random. We will consider targeted attacks in Section 3.5.3.

We compute probabilities and expectation values for the adversarial advantage as well as the optimal selection of security parameters in Section 3.6 for fixed memory.

3.4.1 Expected Number of Collected Keys

Let N_{key} be the number of distinct keys that the adversary gets on average after corrupting n_c users. Since two users may share keys, we get less than $u \times n_c$ keys per rack. The precise calculation is given below:

Lemma 1 (The expected number of keys obtained by the adversary) *Assuming that the adversary corrupts n_c users, the expected total number of distinct keys that the adversary holds is*

$$N_{\text{key}} = \ell \times \left(1 - \left(1 - \frac{u}{\ell}\right)^{n_c}\right).$$

Proof From the first user, the adversary gets u keys. The second gives, on average $u \times \left(1 - \frac{u}{\ell}\right)$ new keys since u keys are already taken. In general, let N_i be the number of new keys gotten from the i^{th} user. We then have the average number of keys with n_c corrupted users:

$$N_{\text{key}}(n_c) = E(\sum_i N_i) = \sum_i E(N_i) \ .$$

We note that we have a recursion

$$E(N_i) = u \times \left(1 - \frac{\sum_{j=1}^{i-1} E(N_j)}{\ell}\right) \ .$$

To see this let $p(k)$ be the probability of having k different keys just before the i^{th} corrupted users, that is $\sum_{j=1}^{i-1} E(N_j) = \sum_k k \cdot p(k)$. Given k keys the probability of getting m new keys is $\binom{u}{m}\left(\frac{l-k}{l}\right)^m \left(\frac{k}{l}\right)^{u-m}$. Thus

$$E(N_i) = \sum_{m=0}^{u} \sum_k m \binom{u}{m} \left(\frac{l-k}{l}\right)^m \left(\frac{k}{l}\right)^{u-m} p(k) \ .$$

Using standard differentiation methods, rewriting and solving we find that $\sum_{m=0}^{u} m \binom{u}{m} \left(\frac{l-k}{l}\right)^m \left(\frac{k}{l}\right)^{u-m} = u(1 - \frac{k}{l})$, from which the relation follows.

We can rewrite the recursion as:

$$N_{\text{key}}(n_c) = N_{\text{key}}(n_c - 1) + u \times \left(1 - \frac{N_{\text{key}}(n_c - 1)}{\ell}\right) \ ,$$

with the solution

$$N_{\text{key}}(n_c) = \ell \times \left(1 - \left(1 - \frac{u}{\ell}\right)^{n_c}\right).$$

3.4.2 Probabilities

We now consider the probability for the adversary to get a non-corrupted user's keys, *i.e.* the targeted user's key indices, are all among the key indices obtained from the corrupted users.

In the following, we denote by K a random variable taking values from 1 to ℓ. We let $K_a^i, i \in [1,n]$ and $a \in [1,u]$ be the random variables defining the keys indices for user i (considering only one rack, *i.e.* $r = 1$). Note that these variables are not independent since we assume each user gets u distinct indices. Let $\mathcal{C}$ be the set of corrupted users and $\mathcal{H}$ the set of non-corrupted ones. We define $n_c := \mathrm{card}(\mathcal{C})$ and $n_h := \mathrm{card}(\mathcal{H})$, *i.e.* $n = n_c + n_h$.

Lemma 2 (The probability to get a targeted user's keys) *With the above notations, for some given $i_0 \in \mathcal{H}$, the attacker's probability of getting a specified user's keys is denoted by $P_1 = P(\forall \mu \in [1,u] : K_\mu^{i_0} \in \{K_b^j\}_{j\in\mathcal{C}, b\in[1,u]})$, and the value is:*

$$P_1 = 1 - \sum_{i=1}^{u} (-1)^{i+1} \binom{u}{i} \prod_{j=1}^{i} a_j^{n_c} \quad \textit{with} \quad a_j = \frac{\ell - j + 1 - u}{\ell - j + 1}.$$

For $r > 1$ we have:

$$P_1 = \left(1 - \sum_{i=1}^{u} (-1)^{i+1} \binom{u}{i} \prod_{j=1}^{i} a_j^{n_c}\right)^r .$$

Proof For a given $i_0 \in \mathcal{H}$, we have:

$$\begin{aligned} P_1 &= P(\forall \mu \in [1,u] : K_\mu^{i_0} \in \{K_\mu^j\}_{j\in\mathcal{C},\mu\in[1,u]}) \\ &= 1 - P(\exists \mu \in [1,u] : K_\mu^{i_0} \notin \{K_\mu^j\}_{j\in\mathcal{C},\mu\in[1,u]}) \\ &= 1 - P(K_1^{i_0} \notin \{K_\mu^j\}_{j\in\mathcal{C},\mu\in[1,u]} \text{ or } \ldots \text{ or } K_u^{i_0} \notin \{K_\mu^j\}_{j\in\mathcal{C},\mu\in[1,u]}) \\ &= 1 - P_1' \end{aligned}$$

Let $A_i = \{K_i^{i_0} \notin \{K_\mu^j\}_{j\in\mathcal{C},\mu\in[1,u]}\}$ for all $i \in [1,u]$.

Since $P_1' = P(A_1 \cup A_2 \cup \ldots \cup A_u) = \sum_{i=1}^{u} (-1)^{i+1}\binom{u}{i} P(A_1 \cap \ldots \cap A_i)$, it only remains to compute $P(A_1 \cap \ldots \cap A_i)$ for all $i \in [1,u]$ to complete the calculation of P_1:

$$\begin{aligned} P(A_1 \cap \ldots \cap A_i) &= P(K_1^{i_0} \notin \{K_\mu^j\}_{j\in\mathcal{C},\mu\in[1,u]} \text{ and } \ldots \text{ and } K_i^{i_0} \notin \{K_\mu^j\}_{j\in\mathcal{C},\mu\in[1,u]}) \\ &= \prod_{j=1}^{n_c} P(K_1^{i_0} \notin \{K_\mu^j\}_{\mu\in[1,u]} \text{ and } \ldots \text{ and } K_i^{i_0} \notin \{K_\mu^j\}_{\mu\in[1,u]}) \\ &= P(K_1^{i_0} \notin \{K_\mu^1\}_{\mu\in[1,u]} \text{ and } \ldots \text{ and } K_i^{i_0} \notin \{K_\mu^1\}_{\mu\in[1,u]})^{n_c} \\ &= \left(\frac{\ell-u}{\ell} \cdot \frac{\ell-u+1}{\ell-1} \cdots \frac{\ell-i+1-u}{\ell-i+1}\right)^{n_c} \\ &= \prod_{j=1}^{i} \left(\frac{\ell-j+1-u}{\ell-j+1}\right)^{n_c} \end{aligned}$$

The value for $r > 1$ follows from independence between the racks.

Based on the probability P_1, we can also find the probability of getting an arbitrary user's keys:

Lemma 3 (The probability to get an arbitrary user's key) *The probability of an attacker to get an arbitrary user's key, $P_2 = P(\exists i \in \mathcal{H} \quad \forall a = 1, \dots, u : K_a^i \in \{K_b^j\}_{j\in\mathcal{C}, b\in[1,u]})$, is given by (also valid for $r > 1$):*

$$P_2 = 1 - (1 - P_1)^{n-n_c}.$$

Proof We have:

$$\begin{aligned}
P_2 &= P(\exists i \in H, \forall \mu \in [1,u] : K_\mu^i \in \{K_b^j\}_{j\in\mathcal{C}, b\in[1,u]}) \\
&= 1 - P(\forall i \in H, \exists \mu \in [1,u] : K_\mu^i \notin \{K_b^j\}_{j\in\mathcal{C}, b\in[1,u]}) \\
&= 1 - P(\bigcap_{i\in H} \{\{K_1^i, \dots, K_\mu^i\} \notin \{K_b^j\}_{j\in\mathcal{C}, b\in[1,u]}\}) \\
&= 1 - P(\{K_1^i, \dots, K_\mu^i\} \notin \{K_b^j\}_{j\in\mathcal{C}, b\in[1,u]})^{n-n_c} \\
&= 1 - (1 - P_1)^{n-n_c}
\end{aligned}$$

Finally, we can consider the probability of getting the keys for two users, allowing the attacker to break a session's keys.

Lemma 4 (The probability of getting two targeted users' keys) *The probability of the attacker to get two targeted users' keys and hence to break a shared key between them is $P_3 = (P_1)^2$.*

Lemma 5 (The probability of breaking an arbitrary session key) *The probability of an attacker to get two arbitrary users' keys and thus to break a session key is $P_4 = (P_2)^2$.*

Both Lemmas 4 and 5 follow directly from the independence of the allocated keys between users.

3.4.3 Optimizing u

Given the probability P_1 defined in the last subsection, we can pose the question of whether there exists a non-trivial optimal value for u. Specifically, we fix a risk-level p and determine the maximal number of users that can be corrupted, n_c, while satisfying $P_1 \leq p$. The optimal value of u is the one allowing the largest number of corrupted users, n_c. The problem is non-trivial since increasing u makes it harder for the adversary to get all keys from the targeted user, but on the other hand, the attacker gets more keys per corrupted user. Figure 3.6 shows that we indeed have non-trivial optimal values.

To make a precise analysis, we consider a large ℓ limit. A power expansion of P_1 gives

$$P_1 = \left(1 - \left(1 - \frac{u}{\ell}\right)^{n_c}\right)^u + \mathcal{O}\left(\frac{1}{\ell^2}\right).$$

We then observe that $n_c \sim \log(1 - P_1^{\frac{1}{u}})/\log(1 - \frac{u}{\ell})$, *i.e.*

$$n_c/\ell \sim -u^{-1} \cdot \log(1 - P_1^{\frac{1}{u}}) \ ,$$

see Figure 3.6. We can use this expression to find the optimal value for u, forcing the adversary to corrupt as many as possible users. By differentiation, we find the optimal u-value as

$$u = -\frac{\log P_1}{\log 2}.$$

To be able to have a risk level $P_1 = 2^{-m}$, the optimal u-value is $u = m$ and the adversary needs to corrupt approximately

$$n_c \sim -\ell \frac{\log^2 2}{\log P_1} = \ell \frac{\log 2}{m}$$

users. If we naively used $u = 1$, the attacker needs to corrupt

$$n_c \sim -\ell \log(1 - P_1) \sim \ell P_1 = \ell 2^{-m}$$

users to breach the risk level, whereas in the last approximation, we assumed P_1 small. That is choosing the optimal u gives a significant advantage, actually logarithmic in the desired risk level. However, the flip side of increasing u is that the adversary has to corrupt fewer users to break the system entirely.

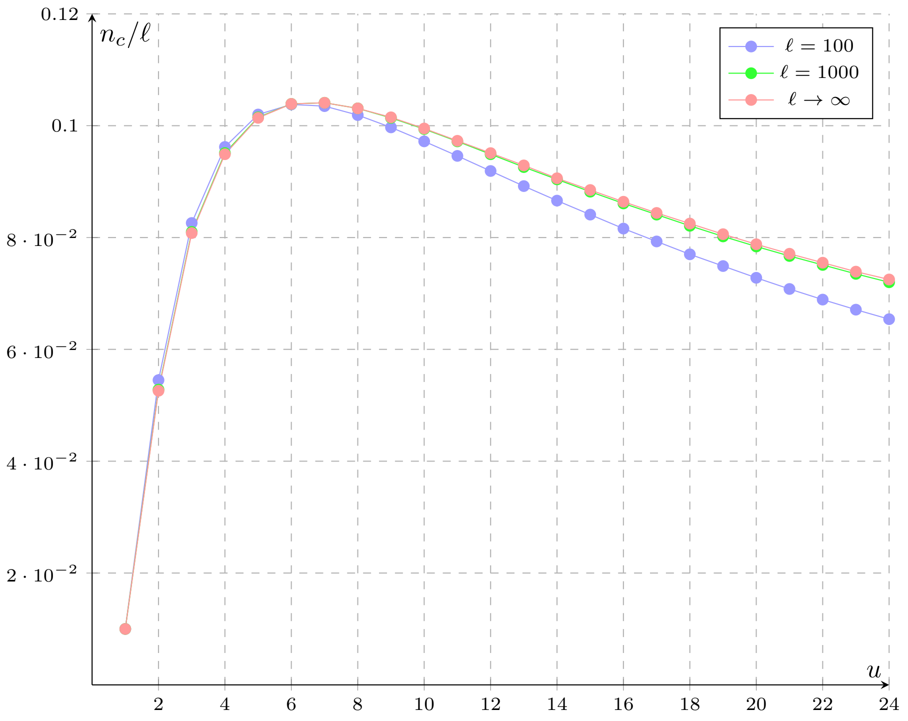


Figure 3.6: The relationship between u and n_c/ℓ for different values of ℓ ($\ell = 100, 1000, \infty$) and $P_1 = 1\%; \ell = 100, 1000$ have been found directly via the formula for P_1 in Lemma 2, whereas the curve for infinite ℓ plotted using the approximation above.

3.4.4 Expected Number of Corrupted Users to Full Breach – the Coupon Collector Problem

We now consider the expected number of users that the adversary needs to corrupt to reveal all keys, *i.e.* fully break the system. As discussed above, for $u = r = 1$, we have the classical coupon collector problem, where $n_c = \ell H(\ell)$ with H being the harmonic series.

For $u > 1, r = 1$, we clearly have $n_c \leq \ell H(\ell)/u$. The problem was analyzed in the context of data package scheduling in [53] and the solution is

$$n_c = \sum_{i=0}^{\ell-1} \left(1 - \binom{i}{u} \Big/ \binom{l}{u}\right)^{-1}.$$

For large ℓ the speed-up is actually close to u:

$$\lim_{\ell \to \infty} \frac{n_c}{\ell \log \ell} = \frac{1}{u}.$$

In the case $u = 1, r > 1$, we have only obtained an upper bound on n_c. Let n_i^ρ be the number of users needed to corrupt to get the i^{th} new key in rack ρ. Each n_i^r is geometrically distributed with probability parameter $p_i = (\ell - i + 1)/\ell$. Denoting the expectation value by E, for $r = 1$, we have that:

$$n_c = E(\sum_{i=1}^{\ell} n_i) = \sum_{i=1}^{\ell} E(n_i) = \sum_{i=1}^{\ell} 1/p_i = \ell H(\ell)$$

as mentioned above. For general r we have $n_c = E\left(\max_{\rho\in[1,r]}(\sum_{i=1}^{\ell} n_i^\rho)\right)$. However, even for the average of the maximum of geometric random variables, we do not have an explicit large r limit [17], only a closed sum formula. Nevertheless, using the bound for the maximum of geometric variables given in [17], we can get the following rough upper bound

$$\begin{aligned} n_c &\leq \sum_{i=1}^{\ell} E\Big(\max_{\rho\in[1,r]}(n_i^\rho)\Big) \\ &\leq \sum_{i=1}^{\ell} \Big(1 - \frac{H(r)}{\log(1-p_i)}\Big) \\ &\leq \ell + \sum_{i=1}^{\ell} \frac{H(r)}{p_i} \\ &\leq \ell + H(r)\ell H(\ell) \end{aligned}$$

Thus in limit $r \to \infty$ we see that $n_c/\ell \log \ell$ is bounded by $\log r$.

3.5 Security Analysis

The protocol can be seen as a special form of authenticated key exchange where the outcome is a fixed key. Moreover, the authentication is implicit, *i.e.* Alice and Bob will hold the same key at the end of an undisturbed protocol run. In contrast, an adversary who actively interrupts the communication and alters the transmitted indices, can make the keys might end up non-matching. However, the security guarantee we can give is an adversary who holds neither Alice's nor Bob's secret keys cannot distinguish the obtained secret key(s) from a random key.

Standard key-exchange protocols require complicated analyses of concurrent sessions. Nevertheless, in our case, we have a simple fixed protocol, and we can split our analysis into three cases:

- ① a passive attacker only monitoring the communication
- ② a man-in-the-middle attacker changing the information of the indices exchanged
- ③ an active attacker trying to impersonate Alice or Bob

Remark 4 (Explicit Authentication) An extra key-confirmation round can be added for a protocol with explicit authentication to be achieved. One way to do this is by Alice sending $h_2(\text{sid}, \mathsf{sk})$ and Bob sending $h_3(\text{sid}, \mathsf{sk})$, where h_2 and h_3 are independent hash functions and the session identity sid contains Alice and Bob's ID, and the indices exchanged. But, again, this is for one-time use only; otherwise, we need to include nonces in the protocol to ensure freshness.

General Assumptions.

Security, in general, relies on an honest setup ensured by a CA without information leakage (see, however, Section 3.5.4 on how to distribute the trust in CA). We also assume that the identities ID_i are publicly known and that they uniquely identify the users.

3.5.1 Passive Attacker

We start our analysis with the weakest attacker model, where the attacker can only observe the communication (*i.e.* see the indices $I(i)$ and $I(j)$ exchanged between participants ID_i and ID_j). Clearly, if the adversary holds the secret keys of both participants (*i.e.* $k^{\rho}_{I(i,\mu,\rho)}$ and $k^{\rho}_{I(j,\mu,\rho)}$ for all $\rho, \mu,$), then he will be able to reconstruct their secret key. We will now show that the obtained key is indeed indistinguishable from a random key for the adversary if he doesn't have all the keys.

We consider two cases. First, where the combiner function h is modelled in the ROM and f is EUF-CMA-secure.

Theorem 1 *Let the combining function h be modelled in the ROM and assume that f is an* EUF-CMA-*secure MAC. Then, a passive attacker cannot distinguish the secret key sk obtained by ID_i and ID_j from a random key with a non-negligible probability unless he has obtained all of their keys $k^{\rho}_{I(i,\mu,\rho)}, k^{\rho}_{I(j,\mu,\rho)}$ for all $\mu \in [1, u], \rho \in [1, r]$.*

PROOF The secret key is $\mathsf{sk} = h\big(t^{\rho}_{i,I(j,1,1)}, \ldots, t^{\rho}_{i,I(j,u,r)}, t^{\rho}_{j,I(i,1,1)}, \ldots, t^{\rho}_{j,I(i,u,r)}\big)$. In the ROM, this key can only be distinguished from random if the input value has been computed. This is only possible if all the MACs are either computed or already known by the adversary. Regarding the latter, the known tabulated MACs from the corrupted users are useless since they contain the wrong ID. Thus the adversary has to compute the MACs, which, by the EUF-CMA assumption, is only possible using the corresponding keys. If even a single key is unknown by the adversary, the probability of distinguishing sk from random is, thus, bounded by the advantage in the EUF-CMA game. Note that the adversary's known keys reduce the space of possible keys since the keys in each rack are distinct, but for ℓ maximally polynomial in the key size, this is a negligible advantage.

Remark 5 The probability of breaking some session key for a static passive adversary or an adversary corrupting random users is given by P_4. The probability of breaking an sk between two specific users is P_3.

Remark 6 Note that if the two users have the same index, the theorem still holds, but it is simply easier for the adversary to obtain all the keys.

Remark 7 The EUF-CMA assumption is too strong because we only need the adversary to be unable to compute the MAC of the identities. Even choosing f as a hash function of the ID and the key is safe in the ROM following the same proof structure.

Remark 8 It is also possible to relax the ROM and only consider h and f randomness extractors. This ensures that the adversary does not learn anything useful from the T_i tables of the corrupted users. Further, if just a single key is unknown, the obtained sk will still be indistinguishable from random.

3.5.2 Man-in-the-Middle and Authentication Attacks

We now consider an attacker who alters the sent messages or even tries to pose as someone else to break authenticity. Note that in this case, we do not have any sk-security in the Canetti-Krawczyk model since the attacked users will not end up with the same key, but a key confirmation would help.

We also note that if the adversary gets all of Alice's keys, he can pretend to be any ID_j to Alice. The adversary simply sends an index, $I(j')$, from one of the corrupted users. Note that Alice is not supposed to keep a record of indices, so Alice will probably not detect that the wrong index is being sent. The adversary can now calculate sk using that all keys are known; and hence the MACs can be constructed.

Nevertheless, if the adversary is missing one of Alice's keys, he cannot distinguish the key computed by Alice from random.

Theorem 1 *Let the combining function h be modelled in the ROM and assume that f is* EUF-CMA*-secure MAC. Consider a user ID_i wanting to establish a key with ID_j. Even if the adversary alters the sent indices, he cannot distinguish the secret key sk obtained by ID_i from random with a non-negligible probability unless he has obtained all of the keys $k^{\rho}_{I(i,\mu,\rho)}$ for all $\mu \in [1,u], \rho \in [1,r]$.*

The proof follows as before, and all remarks about relaxing the assumption given in Section 3.5.1 also hold here.

Remark 9 An active adversary can thus successfully attack a specified user with probability P_1 and some arbitrary user with probability P_2.

3.5.3 Adaptive Corruption

In the protocol, the key indices are sent in clear. However, this is problematic in the case of adaptive attackers. If the adversary wants to target a specific user, he can then observe any key establishment to learn the index of that particular user. The adversary can then look for other users with the same index who might be easier to corrupt.

One possible countermeasure would be to use hybrid security techniques to make the indices private. Nonetheless, a more interesting approach would be to use the fact that both users entering into a key establishment already know that the resulting key will be one of ℓ different possible keys (here, we take $r = u = 1$). As an example, ID_i wanting to talk to ID_j knows that the key is going to be $\mathsf{sk} = h(t_{i,I(j)}, t_{j,I(i)})$, and she can then simply compute all possibilities for $I(j) = 1, \ldots, \ell$. The two users could hash their corresponding possibilities – the correct key will yield the same hash on both sides. They could now exchange these hashes in random order and thus determine the shared key without revealing the indices. This could be done even with logarithmic efficiency.

3.5.4 The Central Authority

As our proposed protocol relies on a trusted third party (TTP), for analyzing security, we assume that the CA is not malicious. However, in real-life applications, this is not always the case. For example, due to the distributed nature of IoT devices, various dedicated authenticated key exchange protocols appeared in the literature. Therefore, we are particularly interested in the results of [2] in terms of cryptographic layer separation and, more precisely, role distribution. Building on the model proposed in [2, Section 2.1] involving different roles for achieving different goals, we believe that distributing the power that a single CA normally has in a classical architecture can be useful especially in the context of our coupon-collector security-based protocol. Furthermore, as we introduced the idea of having r racks of keys, we may naturally distribute a rack per CA to minimize the security impact of a malicious third party. Nonetheless, other more exotic secret sharing schemes may be used to distribute the power between several CAs.

On another note, the idea presented in [27] bases its security on a TTP which *"also serves as an arbitrator when disputes arise due to a user denying certain actions"*. Therefore, besides relying on various CAs as previously mentioned, we stress various methods of circumventing issues like trusting TTPs.

3.5.5 Post-Quantum Security

The primitives used in our proposed protocol (such as MACs and hash functions) seem to be good quantum-safe candidates. The main (optimal) quantum algorithm to break these is Grover's algorithm, which only gives a quadratic speed-up.

3.6 Parameter Choice and Efficiency Analysis

In this section, we analyze the efficiency of our protocols based on the consideration of two types of attacks: small-scale attacks and full breach attacks that we define in the next sections.

The user's global memory usage is determined by r and ℓ (as $r \times \ell$). Hence it is natural to fix $r \times \ell$ to some reasonable constant (*e.g.*, 1Mb) and assume that keys are 128 bits long (as in NIST's PQ-cryptography standardization), which implies that $r \times \ell = 2^{13}$. Thus the question boils down to finding the optimal u, r (and by implication the corresponding $\ell = 2^{13}/r$) maximizing n_c (the expected number of corrupted users) for a given n.

3.6.1 Low-Threat Scenario

Definition 8 A *low-threat scenario* happens when the adversary succeeds to break the $u \times r$ keys of a (targeted or random) user with probability greater than or equal to ϵ, which we call later *the risk-level.*

This section provides numerical values for lowering the adversary's success probability below ϵ. In the following, we consider the two different values of $\epsilon = 1‰,\ 0.01‰$ and evaluate the attack probabilities found in Section 3.4.

Attack 1: Breaking the Keys of a Targeted User.

This attack happens with probability P_1, which does not depend on n. Therefore, we are interested in finding the optimal u and r allowing maximizing n_c under the constraint:

$$P_1(\frac{2^{13}}{r}, u, n_c, r) \leq 1‰, 0.01‰$$

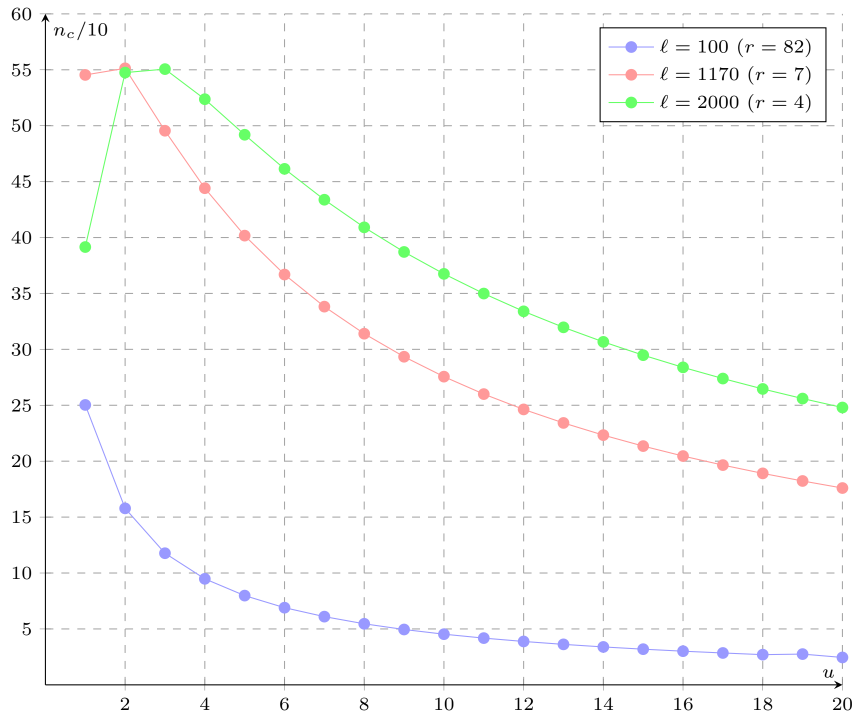


Figure 3.7: u and n_c for (ℓ, r) values s.t. $\ell r = 2^{13}$ and $P_1 = 1‰$.

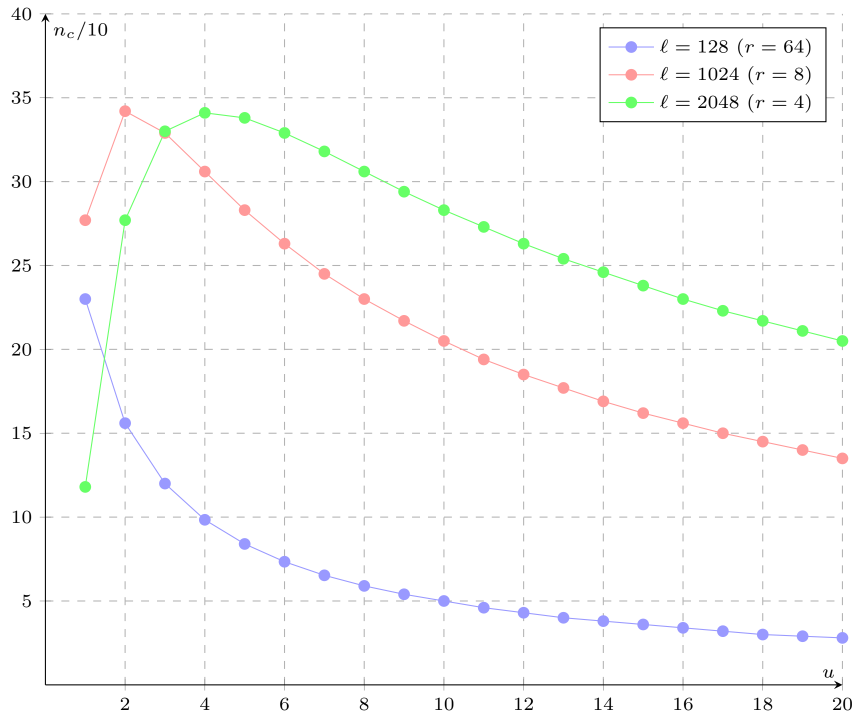


Figure 3.8: u and n_c for (ℓ, r) values s.t. $\ell r = 2^{13}$ and $P_1 = 0.01‰$.

Repeating the analysis from Section 3.4.3 for general u, r with a fixed memory size $\ell r = 2^{13}$, we find for large ℓ that the optimal parameter choice is reached for $ur = -\frac{\log P_1}{\log 2}$. We see clearly in Figure 3.7 and Figure 3.8 the presence of an optimum in some curves ($r = 4, 7$ in Figure 3.7 and $r = 4, 8$ in Figure 3.8). This optimum corresponds to a non-trivial optimal ur, and it increases when P_1 is decreasing ($ur \sim 10$ for $P_1 = 1‰$ and $ur \sim 16$ for $P_1 = 0.01‰$). No optimum is noticed when $r > -\frac{\log P_1}{\log 2}$. Moreover, n_c reaches the highest values when the probability risk-level is large ($n_c^{\max} \sim 569$ for $P_1 = 1‰$ and $n_c^{\max} \sim 341$ for $P_1 = 0.01‰$).

To see whether we can differentiate the parameters satisfying $ur = -\log P_1 / \log 2$, we further compute the expected number of corrupted users n_c needed for a full breach for the corresponding parameters (r, u, ℓ). We take $(r, u, \ell) = (r, -\frac{1}{r}\frac{\log P_1}{\log 2}, \frac{2^{13}}{r})$. Table 3.1 and Table 3.2 show the numerical values.

(r, u, ℓ)	$(1, 10, 2^{13})$	$(2, 5, 2^{12})$	$(5, 2, 1638)$	$(10, 1, 819)$
$n_c^{\max}$	7343	7885	7859	7853

Table 3.1: $n_c^{\max}$ to fully breach the system with $P_1 = 1‰$ and $\ell \times r = 2^{13}$

(r, u, ℓ)	$(1, 16, 2^{13})$	$(2, 8, 2^{12})$	$(4, 4, 2^{11})$	$(8, 2, 2^{10})$	$(16, 1, 2^{9})$
$n_c^{\max}$	4929	4919	5065	4732	4928

Table 3.2: $n_c^{\max}$ to fully breach the system with $P_1 = 0.01‰$ and $\ell \times r = 2^{13}$

We notice from Table 3.1 and Table 3.2 that for all the possible optimal (r, u, ℓ) combinations, $n_c^{\max}$ always takes approximately the same value (well within the standard deviation of the Monte Carlo simulations used to obtain the tables) and only depends on the chosen P_1 level. Hence, the optimal combination (r, u, ℓ) is not *unique*. We conjecture that there is a duality between u and r with constrained memory. Note that we could try to explain this, *e.g.* for $(r, u, \ell) = (1, 2, 2\ell) \mapsto (2, 1, \ell)$ by splitting a rack of size $2 \times \ell$ into two of size ℓ. However, two random keys from the original rack only have probability of around 1/2 of being split into separate racks. Thus, further analysis is needed, which we postpone for future research.

Attack 2: Breaking the Keys of a Random User.

This attack happens with probability P_2 which depends on n. Therefore, we are interested in finding the optimal u and r allowing maximizing n_c for a given n under the constraint:

$$P_2(\frac{2^{13}}{r}, u, n, n_c, r) \leq 1‰, 0.01‰$$

Tables 3.3, 3.4 investigate this for $n' = \log_{10}(n) = 1, \ldots, 6$. Values were obtained using a Python code.

	$u = 1$	$u = 2$	$u = 3$	$u = 4$	$u = 5$
$n' = 2$	$(9, 7)$	$(2, 57)$	$(1, 60)$	$(1, 67)$	$(1, 69)$
$n' = 3$	$(1, 1)$	$(3, 15)$	$(2, 46)$	$(2, 83)$	$(2, 114)$
$n' = 4$	$(1, 1)$	$(3, 5)$	$(2, 21)$	$(2, 45)$	$(2, 69)$
$n' = 5$	$(1, 1)$	$(2, 2)$	$(2, 10)$	$(2, 26)$	$(1, 43)$
$n' = 6$	$(1, 1)$	$(1, 1)$	$(2, 5)$	$(2, 15)$	$(1, 27)$

Table 3.3: $(r^{\text{opt}}, n_c^{\max})$ for $P_2 = 1‰$

	$u = 1$	$u = 2$	$u = 3$	$u = 4$	$u = 5$
$n' = 2$	$(9, 4)$	$(6, 35)$	$(3, 66)$	$(2, 65)$	$(1, 72)$
$n' = 3$	$(1, 1)$	$(4, 9)$	$(4, 28)$	$(3, 50)$	$(2, 70)$
$n' = 4$	$(1, 1)$	$(3, 3)$	$(3, 13)$	$(3, 28)$	$(2, 43)$
$n' = 5$	$(1, 1)$	$(4, 2)$	$(4, 7)$	$(3, 16)$	$(2, 27)$
$n' = 6$	$(1, 1)$	$(1, 1)$	$(2, 3)$	$(2, 9)$	$(2, 17)$

Table 3.4: $(r^{\text{opt}}, n_c^{\max})$ for $P_2 = 0.01‰$

From Table 3.3 and Table 3.4 we see that the highest value of n_c is reached when $(u, r, n) = (5, 2, 1000)$ ($n_c^{\max} = 114$) for $P_2 = 1‰$ and when $(u, r, n) = (5, 1, 100)$ ($n_c^{\max} = 72$) for $P_2 = 0.01‰$.

Remark 10 We notice that the value of $n_c^{\max}$ for Attack 1 is about 5 times bigger than the one for Attack 2 ($n_c^{\max}(P_1 = 1‰) = 569 > n_c^{\max}(P_2 = 1‰) = 114$ and $n_c^{\max}(P_1 = 0.01‰) = 341 > n_c^{\max}(P_2 = 0.01‰) = 72$).

3.6.2 Full Breach

Definition 9 A *full system breach* happens when the adversary succeeds to recover all the $\ell \times r$ secret keys given by the CA.

In the following, we are interested in finding the maximal value of n_c needed to breach the system fully and the corresponding r and u for a fixed memory size $M = \ell \times r = 2^{13}$. Table 3.5 shows the numerical values obtained after running the Monte Carlo simulation in Python and taking $N = 1000$.

u	1	2	3	4	5	6	7
$n_c^{\max}$	40159	39270	25430	18323	15544	13283	10606
u	8	9	10	11	12	13	14
$n_c^{\max}$	10219	8413	7361	6884	6645	6326	5636

Table 3.5: Values of $n_c^{\max}$ to fully breach the system ($\ell \times r = 2^{13}$ and $N = 1000$). In all cases $r^{\text{opt}} = 1$.

$n_c^{\max}$ is strictly decreasing when u increases and reaches the highest value when $u = r = 1$.

3.6.3 Expected Number of Corrupted Users to Fully Breach the System

This section gives numerical values of n_c to breach the system fully. For example, the following values are obtained using Monte Carlo simulation in Python and taking $N = 10000$.

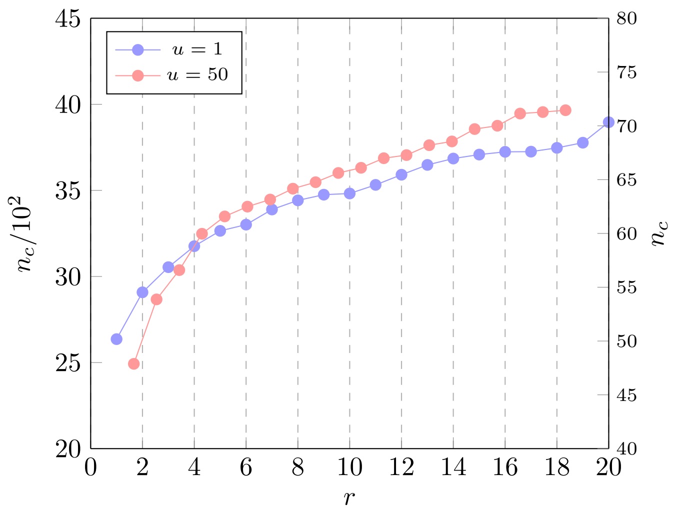


Figure 3.9: r and n_c for $\ell = 400$ and $u = 1, 50$ to recover all secret keys.

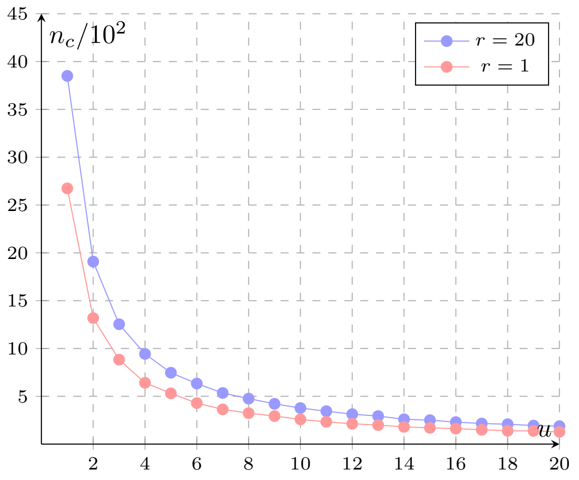


Figure 3.10: u and n_c for $\ell = 400$ and $r = 1, 20$ to recover all secret keys.

From Figure 3.9 and Figure 3.10, we see clearly that n_c is increasing when r is increasing and decreasing when u is increasing. The values obtained through this simulation are very close to the theoretical results of Section 3.4.4. We consider the main cases ($u = r = 1$, $u > 1, r = 1$ and $u = 1, r > 1$) and refer the reader to Table 3.6 for precise values.

Cases	$u = r = 1$	$u > 1, r = 1$	$u = 1, r > 1$
$(n_c^{\text{simu}}, n_c^{\text{theo}})$	$(2627, 2630)$	$(47, 47)$	$(3850 < 9854)$

Table 3.6: n_c values from Monte Carlo simulation and theoretical formulas

3.7 Conclusion and Discussion

This chapter presented a new authenticated key exchange protocol entirely based on symmetric primitives and analyzed its security. We also discussed parameter choices and efficiency; we found especially interesting ways of improving security by handing out more keys per user while keeping memory usage constant.

Future Work. A natural research direction would be to formally analyze both the similarities of our proposed construction with standard public-key cryptography schemes and the post-quantum nature of our key distribution protocol. Furthermore, for a more precise security assessment, it is important to achieve better bounds for the expected number of corrupted users required to get a full breach in the case of general r, u – a problem which is an interesting coupon collector problem in its own right. Finally, it would also be interesting to understand in detail the u and r duality phenomenon seen in Section 3.6 when dealing with constrained memory and a large ℓ.

Another possible avenue of future research is to consider hybrids of the current protocol, *e.g.* by achieving forward secrecy relying on a computational assumption.

Chapter 4

Design of a DoS-Resistant Voting Protocol

This work has been done in collaboration with: David Naccache, Gergei Bana, Rémi Géraud-Stewart, Peter B. Rønne, Marco Biroli, Megi Dervishi and Peter YA Ryan and was published on E-vote-Id 2022. We reproduce in this chapter the paper [3].

Contents

Summary

The open vote network (OV-Net, [22]) is a secure two-round multi-party protocol facilitating the computation of a sum of integer votes without revealing their individual values. This is done without a central authority trusted for privacy, and thus allows decentralised and anonymous decision-making efficiently. As such, it has also been implemented in other settings, such as financial applications, see e.g. [40, 52].

An inherent limitation of OV-Net is its lack of robustness against denial-of-service attacks, which occur when at least one of the voters participates in the first round of the protocol but (maliciously or accidentally) not in the second. Unfortunately, such a situation is likely to occur in any real-world implementation of the protocol with many participants. This could incur serious time delays from either waiting for the failing parties or perhaps having to perform extra protocol rounds with the remaining participants.

This chapter provides a solution to this problem by extending OV-Net with mechanisms tolerating several unresponsive participants, the basic idea being to run several sub-elections in parallel. The price to pay is a carefully controlled privacy loss, an increase in computation, and a statistical loss in accuracy, which we demonstrate how to measure precisely.

Keywords Multi-party computation, open vote network, denial of service, decentralised voting.

4.1 Introduction

Open vote network (OV-Net, [22]) is a secure multi-party protocol allowing to compute a sum of integer votes without revealing their values. As such, it has several applications in social choice and financial applications [40, 52].

An inherent limitation of OV-Net is its lack of robustness against denial-of-service attacks, which occur when at least one of the voters initiates the protocol but (maliciously or accidentally) does not complete it. Unfortunately, such a situation is very likely to occur in any real-world implementation of the protocol. This will cost serious time delays from either waiting for the failing parties or perhaps having to perform extra protocol rounds with the remaining participants.

This chapter provides a solution to this problem by extending OV-Net with mechanisms tolerating several unresponsive participants. However, the price to pay is a carefully controlled privacy loss, an increase in computation, and a statistical loss in the accuracy.

4.2 Related Work

Cryptographic voting protocols allow mutually-distrusting entities to reach a consensus on some value. This has applications in social choice (e.g., elections) and financial mechanisms [40, 52]. Most of these protocols involve a trusted authority responsible for running the election or tallying the results. However, there exist a number of so-called "boardroom" or "self-tallying" schemes that do away with the need for a central authority [31]. In such schemes, the election is an interactive protocol between the voters only. Whether a centralised or decentralised protocol is better suited to a given situation depends on practical and context-specific concerns, such as the protocol's scalability or whether the trusted authority assumption makes sense. For instance, the self-tallying protocol of Groth [20] has $n+1$ rounds for n voters, which makes it impractical to use for larger elections.

Open vote network (OV-Net) is a self-tallying voting scheme proposed by Hao, Ryan and Zieliński [22]. Improving upon Hao and Zielński's earlier AV-net [24, 21], it is a 2-round protocol which makes it an appealing candidate for larger-scale elections. However, one of OV-Net's limitations, according to Hao–Ryan–Zieliński, is that the protocol cannot handle denial-of-service (DoS) events:

> *" (...) For example, if some voters refuse to send data in round 2, the tallying process will fail. This kind of attack is overt; everyone will know who the attackers are. To rectify this, voters need to expel the disrupters and restart the protocol; their privacy remains intact. However the voting process would be delayed, which may prove costly for large scale (countrywide) elections (...)"* — [22, Sec 3.4]

While the protection of privacy and the identification of culprits are desirable properties, the need to restart the protocol every time a voter drops out is a very strong limitation. This weakness is what we set out to rectify in this paper by extending OV-Net to handle DoS events gracefully. Our modifications come at a cost, which we investigate.

Some earlier works have already tried to improve the security and efficiency of OV-Net. In [30], fairness (i.e. preventing that voters get partial results before casting their vote) was guaranteed by committing to the vote in the first round. Further, the robustness against denial of service attacks was improved by introducing a recovery round: if some voters did not participate in the second round, the remaining voters perform a third round to achieve the partial tally for their cast votes. However, this does not guarantee that there are no fall outs in the recovery round. In [19], it was shown that using a bilinear group setting and assuming a public key infrastructure; the voting protocol can be made non-interactive, i.e. 1-round. This increases the run time considerably but does not in itself remove the robustness problem. Finally, in [40], the OV-Net was implemented via a smart contract that financially punishes voters who drop out of the election. This gives an economic incentive to participate in the second round but does not prevent dedicated DoS attacks, or involuntary drop outs e.g. due to lack of network access, and it assumes that the participants are willing to risk the economic punishment in the first place.

4.3 The OV-Net Protocol

We recall here the OV-Net protocol in the case of a *referendum*: there are two choices $A = 1$ and $B = 0$ and n voters; each voter will cast a vote v_i for either A or B (i.e., $v_i \in \{0, 1\}$) and the final tally will reveal the sum of all votes. (Ultimately, we may set a threshold to adopt A or B as a final decision based on the tally, but this is beyond the scope of OV-Net.)

We assume that all participants have agreed ahead of time to use a given cyclic group $\mathbb{G}$ of generator g in which the decisional Diffie–Hellman problem is intractable. Let q be the order of $\mathbb{G}$. Each voter $i \in \{1, \ldots, n\}$ samples a random value $x_i \xleftarrow{\$} \mathbb{Z}_q$ as a secret.

1. **Round 1:** Each voter $i \in \{1, \ldots, n\}$ publishes g^{x_i} along with a zero-knowledge proof of knowledge (ZKP) of x_i. In practice, ZKP can be a Schnorr signature [51].
   When this round finishes, each voter $i \in \{1, \ldots, n\}$ does the following:
   - checks the validity of the ZKP for all x_j, $j \in \{1, \ldots, n\} \backslash \{i\}$,
   - computes:
   $$g^{y_i} = \prod_{j=1}^{i-1} g^{x_j} / \prod_{j=i+1}^{n} g^{x_j}$$
2. **Round 2:** Each participant $i \in \{1, \ldots, n\}$ publishes $g^{x_i y_i} g^{v_i}$ and a ZKP for v_i showing that $v_i \in \{0, 1\}$. In practice, this proof can be implemented using the technique of Cramer–Damgård–Schoenmakers [13].

Remark 11 The OV-Net protocol can be extended to more than two candidates by an appropriate encoding of v_i [14, 4], with the final tally requiring a (superincreasing) knapsack resolution after a discrete logarithm computation [22, Sec. 2.2]. Therefore, we focus on the simpler case of two candidates.

4.3.1 Denial of Service

In the description of OV-Net, we implicitly assume that all participants are honest, to the extent that the proofs of knowledge are valid and that they follow the protocol. However, suppose one or several voters publish an incorrect proof of knowledge or do not follow the protocol. In this case, reaching a conclusion for this particular vote event is impossible. This is called a denial of service (DoS) event.

When a DoS event occurs, the non-compliant voters can be identified and removed from a subsequent vote. However, the results for that particular vote must be discarded (or cannot be computed), and a fresh vote must take place. This is troublesome for several reasons. One reason is that as n becomes large, disconnection or time-out events become more common and, therefore, the protocol's failure probability increases. Another reason is that accounting for protocol errors and re-voting adds complexity to real-world OV-Net implementations.

4.4 The Parallel **OV-Net** Protocol

We consider a modification of OV-Net where users participate in several voting sessions in parallel. However, as we now explain, not all voters take part to all votes. Let n be the number of voters and M the number of parallel vote sessions. Each voter will participate in k pseudo-randomly chosen sessions amongst M.

More precisely, voter i picks k sessions before the protocol is run, which we call i's *selection*. We assume that this selection is pseudo-random, i.e. that any given selection happens with the same probability $1/\binom{M}{k}$. As a result, not all sessions have the same number of voters, a phenomenon that we will need to account for.

Remark 12 A natural question is whether we could impose a more intelligent rule, that would guarantee that there is always the same number of voting opportunities for each of them. Indeed, a solution is provided, in some cases, by Steiner systems [11]: a Steiner system with parameters t, k, n, written $S(t, k, n)$, is an n-element set S together with a set of k-element subsets of S (called *blocks*) with the property that each t-element subset of S is contained in exactly *one* block.

The existence of Steiner systems is deeply connected to number-theoretic properties. In particular, the existence of a $S(t, k, n+1)$ system precludes that of a $S(t, k, n)$. Thus, although we could initially form a balanced set of voters in some initial setting, it cannot be done if any of the voters bails out (or is disconnected).

However, it is not obvious how a decentralised pool of voters could agree on such a setting in a non-mutually-trusting way and without leaking private information. It also remains an interesting question whether approximately balanced block designs exist that are "stable" in the sense that they retain this property when elements are removed.

Should one voter drop out during one voting session, this particular session will be discarded, but all sessions in which this voter didn't participate will go through. Unfortunately, this also discards all the votes of honest voters in the dropped session. To overcome this exclusion, we allow each voter to vote k times: in other words, each voter will cast k votes into k independent ballots amongst the M.

Our claim is that, in this case, the final tally's result reflects the choice of honest voters even after discarding all the sessions that a dishonest voter blocked. Furthermore, when several voters are dishonest, their cumulative effect on the final tally is weighed down by the fact that they shared many vote sessions. Concretely, for $k = M/2$, the first dishonest voter makes about $M/2$ sessions invalid; but amongst the remaining sessions, only about $M/4$ can share a second dishonest voter, etc. Hence, this setting tolerates roughly $\log_2 M$ droppers at the price of running M sessions.

In summary, by running several sessions, several competing phenomena occur:

1. The overall protocol's resilience against DoS events is improved as we run more sessions — more sessions, however, bring an additional computational and communication cost;
2. Sessions have a varying number of voters in them, and not every voter partakes in every session, which introduces a bias — we can expect this bias to become small when many sessions are run;
3. The list of participants in each session is public; therefore, some information about individual voters' preferences is leaked — running more sessions results in an increased loss of privacy.

There is, therefore, a balance to be struck, and we must quantify these phenomena more precisely.

4.5 Parallel OV-Net DoS Resilience

Let ℓ be the number of voters causing a DoS event; they cause a (random) number X_ℓ of sessions to be discarded. The protocol fails when all sessions have been discarded, i.e., when $X_\ell \geq M$ — this cannot happen when $\ell < M/k$. If $\ell \geq M/k$ then it is possible to stop the protocol entirely when the selections of dropping voters cover all sessions. However, the likelihood of this happening is low when each selection is random and independent, as many of the dropping voters will have sessions in common.

This is a particular variant of the famous coupon collector's problem, which has been extensively studied.

Lemma 6 *The average number of DoS events necessary to cause an overall failure when we run M parallel sessions and each voter partakes in k of them is:*

$$\mathbb{E}[\ell \mid \textit{overall protocol failure}] = \binom{M}{k} \sum_{r=1}^{M} (-1)^{r-1} \frac{\binom{M}{r}}{\binom{M}{k} - \binom{M-r}{k}}$$

Proof Let ℓ be the number of DoS events causing a protocol failure. W.l.o.g. these events correspond to dropping voters $r = 1, \ldots, \ell$. Let $\mathrm{sel}(r)$ be the selection of r and let T_j be the index of the first dropping voter that has j in its selection. Thus $\ell = \max_j T_j$.

Let L be an integer, note that $\max_j T_j > L$ if and only if there is an index v such that $T_v > L$. Similarly, for all $1 \leq j_1 < j_2 < \cdots < j_v \leq M$ we have $\min\{T_{j_1}, \ldots, T_{j_v}\} > L$ if and only if T_{j_u} for all $u \in \{1, \ldots, v\}$. The inclusion-exclusion formula then gives:

$$\Pr[\ell > L] = \sum_{v=1}^{M} (-1)^{v-1} \sum_{1 \leq j_1 < \cdots < j_v \leq M} \Pr[\min\{T_{j_1}, \ldots, T_{j_v}\} > L]$$

The right-hand side happens only when none of the sessions $j_1, \ldots, j_v$ belongs to the first u selections; using independence, we have

$$\Pr[\min\{T_{j_1}, \ldots, T_{j_v}\} > L] = \left(\frac{\binom{M-v}{k}}{\binom{M}{k}} \right)^L$$

hence

$$\begin{aligned}\Pr[\ell > L] &= \sum_{v=1}^{n} (-1)^{v-1} \sum_{1 \leq j_1 < \cdots < j_v \leq M} \left(\frac{\binom{M-v}{k}}{\binom{M}{k}} \right)^L \\ &= \sum_{v=1}^{n} (-1)^{v-1} \binom{M}{v} \left(\frac{\binom{M-v}{k}}{\binom{M}{k}} \right)^L\end{aligned}$$

from which we obtain the claimed formula by computing the expected value

$$\mathbb{E}[\ell] = \sum_{L \geq 0} \Pr[\ell > L].$$

Figure 4.1 compares simulation results to the formula of Lemma 6, showing excellent agreement. The simulation is for $M = 50$ and k varying from 1 to 49, over 10^5 runs . Using this information, we can choose parameters M and k to accommodate a given number of potential drop-outs.

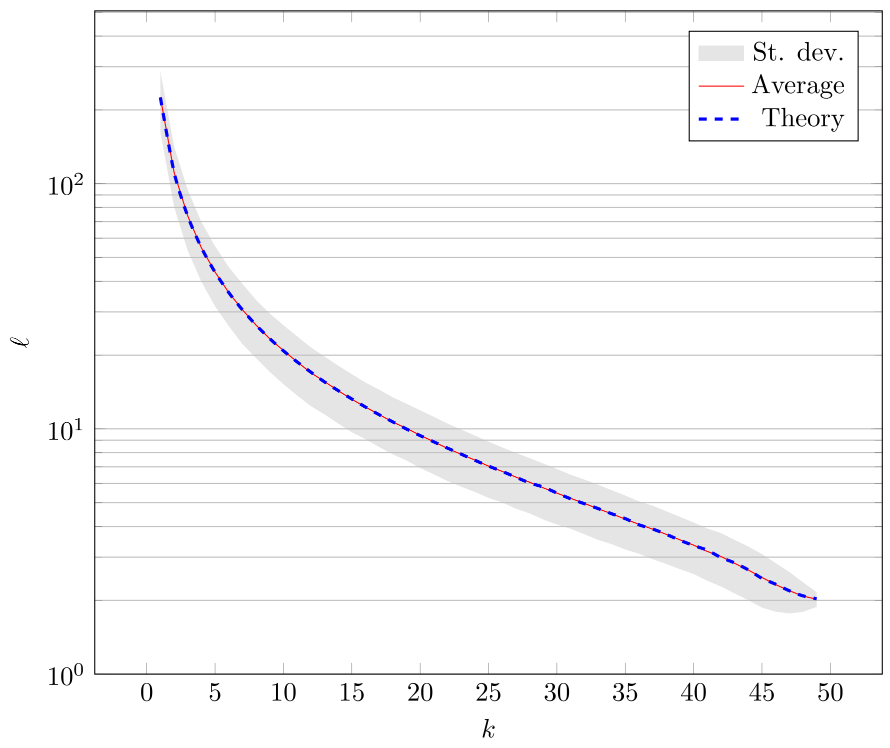


Figure 4.1: Simulated and predicted minimum number of DoS events necessary to cause an overall protocol failure, for $M = 50$ and $k = 1, 2, \ldots, 49$.

When we have fewer than the critical number of DoS events, the remaining sessions can be tallied. We can estimate the number of remaining valid sessions as $\mu = M - X_\ell$:

Lemma 7

$$\mathbb{E}(\mu) = (M-k)\left(1 - \frac{k}{M}\right)^{\ell-1}$$

Proof We compute the expected number of surviving sessions $\mathbb{E}(M - X_\ell)$ when ℓ voters have quit the voting process, and we deduce $\mathbb{E}(X_\ell)$ by induction:

- $\mathbb{E}(X_1) = k = M - (M-k)(M-k)^0$
- Assume that the formula is correct until ℓ. When a new dropout happens, an average of $\frac{k}{M}\mathbb{E}(M - X_\ell)$ sessions will be discarded, thus:

$$\mathbb{E}(M - X_{\ell+1}) - \mathbb{E}(M - X_\ell) = -\frac{k}{M}\mathbb{E}(M - X_\ell)$$
$$M - \mathbb{E}(X_{\ell+1}) - M + \mathbb{E}(X_\ell) = -\frac{k}{M}(M - \mathbb{E}(X_\ell))$$

Therefore

$$\begin{aligned}
\mathbb{E}(X_{\ell+1}) &= \mathbb{E}(X_\ell)\left(1-\frac{k}{M}\right)+k \\
&= \left(M-(M-k)\left(1-\frac{k}{M}\right)^{\ell-1}\right)\left(1-\frac{k}{M}\right)+k \\
&= M-(M-k)\left(1-\frac{k}{M}\right)^{\ell}
\end{aligned}$$

Finer results about the distribution X_ℓ are given in the following section.

Distribution of X_ℓ

We have the following recurrence relation:

Lemma 8

$$\Pr[X_{\ell+1}=x|X_\ell=r]=\frac{\binom{M-r}{x-r}\cdot\binom{r}{k-x+r}}{\binom{M}{k}}.$$

Let us explain this formula term by term: for $X_{i+1}=j$, knowing that $X_i=\ell$ we know that a not-yet-accounted-for quitter voted in $j-\ell$ sessions that did not overlap with the previous ones and in $k-j+\ell$ sessions that overlapped with the previous ones. There are $M-\ell$ *live* sessions and ℓ discarded ones, the choice of which sessions the quitter will vote in therefore gives us the first two terms at the numerator. The denominator counts all the possible votes.

In matrix form:

$$A_{i,j}=\Pr(X_i=j) \qquad \text{and} \qquad B_{i,j}=\frac{\binom{M-j}{i-j}\binom{j}{k-i+j}}{\binom{M}{k}}$$

This enables us to simulate the probability distribution functions for any set of parameters. Figure 4.2 shows numerical computations for the distribution of X_q, with $q\in[\![1,8]\!]$, $M=500$ and $k=25$, and Figure 4.3 shows the very good agreement between theory and experiment.

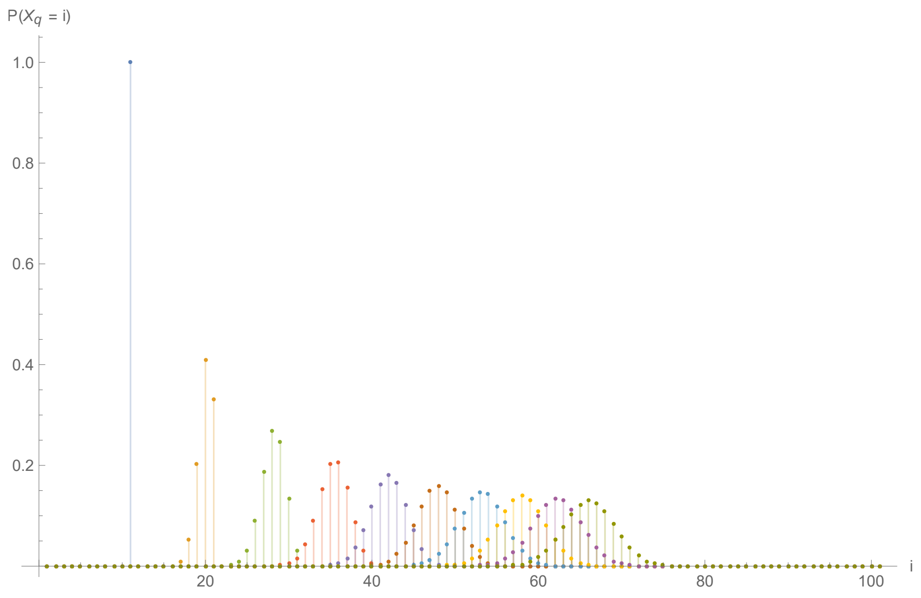


Figure 4.2: PDF of X_q for $q \in [\![1, 8]\!]$

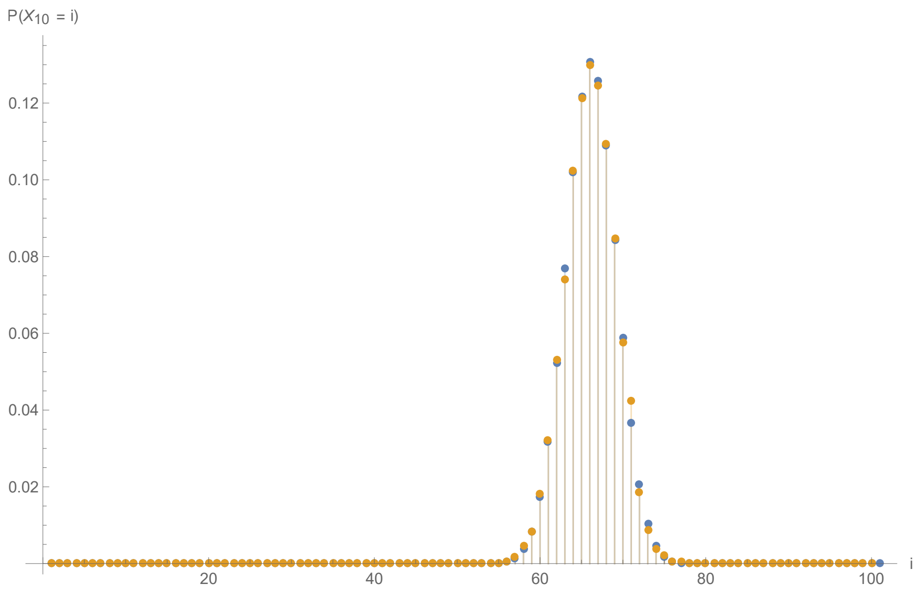


Figure 4.3: Theoretical results (blue) vs experimental values in Python (orange).

4.6 Tally-combining algorithms

In this section, we formalise how a final result can be obtained from the parallel OV-Net protocol. It is practical at this point to use vector notations.

We make the assumptions that voters are consistent, i.e., that they make the same choice across all the voting sessions in which they participate[1]. We denote v_i the choice of voter i, and collect this (unknown) information into a vector $\boldsymbol{v} = (v_1, \dots, v_n)$. If the vote went through with no incident, we would obtain the final tally

$$V = \sum_{i=1}^{n} v_i = \boldsymbol{v} \cdot \mathbf{1}.$$

When a voter drops out, all the sessions in which he participated are discarded. Let $0 < \mu \leq M$ be the number of remaining sessions and for each session $j \in \{1, \dots, \mu\}$ let $s_{j,i}$ be the number of times that voter i participated in session j; hence $s_{j,i}$ can take values in $\{0, 1\}$ with the minimum value meaning that voter i did not partake in session j, and the maximum value indicating that they voted during session j. The tally for session j is, therefore:

$$t_j := \sum_{i=1}^{n} s_{j,i} v_i = \boldsymbol{v} \cdot \boldsymbol{s}_j \quad \text{where } \boldsymbol{s}_j := (s_{j,1}, \dots, s_{j,n}).$$

By definition, $s_{j,i} = 0$ if voter i dropped out, and $\boldsymbol{s}_j$ is non-zero (otherwise $\mu = 0$). At the end of the procedure, the following information is public knowledge:

$$\boldsymbol{T} := (t_1, \dots, t_\mu) \qquad\qquad \boldsymbol{S} := (\boldsymbol{s}_1, \dots, \boldsymbol{s}_\mu)$$

The question is now: given $(\boldsymbol{S}, \boldsymbol{T})$, and the parameters $\mathsf{pp} = (n, k, M, \mu)$, how well can we approximate V? To answer this question, we need a precise definition of the error.

Definition 10 (Average- and worst-case error) Let $\mathcal{A}$ be an algorithm taking as input $\boldsymbol{S}$, $\boldsymbol{T}$ and (implicitly) pp, and returning a real number. We refer to $\mathcal{A}$ as a *tally-combining algorithm*, and we write $\delta(\boldsymbol{v}, \boldsymbol{S}) := V - \mathcal{A}(\boldsymbol{S}, \boldsymbol{T})$ the *tallying error*.

Since δ depends on a choice of $\boldsymbol{v}$, which is not public information, and since $\boldsymbol{S}$ is a collection of randomly chosen selections, it is more meaningful to consider the *average error*:

$$\pi^{\mathcal{A}}_{\text{avg}} := \mathbb{E}_{\boldsymbol{v},\boldsymbol{S}}[\delta(\boldsymbol{v}, \boldsymbol{S})],$$

where $\boldsymbol{v}$ and $\boldsymbol{S}$ span all their possible values.

While $\mathcal{A}$ may give results that are close to V on average, there may be corner cases in which the predicted value wanders substantially away from V; this phenomenon is controlled by the *worst-case error*:

$$\pi^{\mathcal{A}}_{\text{wc}} := \max_{\boldsymbol{v},\boldsymbol{S}} |\delta(\boldsymbol{v}, \boldsymbol{S})|,$$

[1]This makes our analysis simpler, but in practice a voter casting inconsistent votes simply weakens his own position.

where again $\boldsymbol{v}$ and $\boldsymbol{S}$ span all their possible values.

A simple tally-combining algorithm is given by averaging the tallies and rescaling to account for lost sessions, i.e.

$$\mathcal{A}_{\text{naïve}}(-, T) = \frac{M}{\mu k}(\mathbf{1} \cdot \boldsymbol{T})$$

(we must divide by k since each voter casts k votes).

Lemma 9 *The naïve tally-combining algorithm gives:*

$$\pi^{\textit{naïve}}_{\textit{avg}} = 0$$

Proof Let V' be the result of the naïve tally-combining algorithm, i.e.

$$\begin{aligned} V' &= \frac{M}{\mu k}(\mathbf{1} \cdot \boldsymbol{T}) = \frac{M}{\mu k}\sum_{j=1}^{\mu} t_j = \frac{M}{\mu k}\sum_{j=1}^{\mu}\sum_{i=1}^{n} s_{j,i} v_i \\ &= \frac{M}{\mu k}\sum_{i=1}^{n}\sum_{j=1}^{\mu} s_{j,i} v_i = \sum_{i=1}^{n} v_i \underbrace{\left(\frac{M}{\mu k}\sum_{j=1}^{\mu} s_{j,i}\right)}_{\beta_i} = \boldsymbol{v} \cdot \boldsymbol{\beta} \end{aligned}$$

where $\boldsymbol{\beta} = (\beta_1, \dots, \beta_n)$. We now make use of the fact that $\boldsymbol{v}$ and $\boldsymbol{s}$ are independent and of the linearity of $\mathbb{E}$ to compute the average error:

$$\begin{aligned} \pi^{\text{naïve}}_{\text{avg}} &= \mathbb{E}_{\boldsymbol{v},\boldsymbol{s}}[V' - V] = \mathbb{E}_{\boldsymbol{v},\boldsymbol{s}}[\boldsymbol{v} \cdot (\boldsymbol{\beta} - \mathbf{1})] = \frac{1}{2}\mathbf{1} \cdot \mathbb{E}_{\boldsymbol{s}}[\boldsymbol{\beta} - \mathbf{1}] \\ &= -\frac{n}{2} + \frac{1}{2}\sum_{i=1}^{n} \mathbb{E}_{\boldsymbol{s}}[\beta_i] = -\frac{n}{2} + \frac{1}{2}\frac{M}{\mu k}\sum_{i=1}^{n} \mathbb{E}_{\boldsymbol{s}}\left[\sum_{j=1}^{\mu} s_{j,i}\right] \\ &= -\frac{n}{2} + \frac{1}{2}\frac{M}{\mu k}\sum_{i=1}^{n}\sum_{j=1}^{\mu} \mathbb{E}_{\boldsymbol{s}}[s_{j,i}] \\ &= -\frac{n}{2} + \frac{1}{2}\frac{M}{\mu k}\sum_{j=1}^{\mu} \mathbb{E}_{\boldsymbol{s}}\left[\sum_{i=1}^{n} s_{j,i}\right] \\ &= -\frac{n}{2} + \frac{1}{2}\frac{M}{\mu k}\sum_{j=1}^{\mu} \mathbb{E}_{\boldsymbol{s}}\left[\mathbf{1} \cdot \boldsymbol{s}_j\right] \end{aligned}$$

Since every voter participates in k sessions, on average each session has nk/M participants. Thus $\pi^{\text{naïve}}_{\text{avg}} = -n/2 + n/2 = 0$.

More generally, let $\boldsymbol{x} = (x_1, \dots, x_\mu)$ be a vector of real coefficients, and define the *weighed tally-combining algorithm* $\mathcal{A}_{\boldsymbol{x}}(T) = \boldsymbol{x} \cdot \boldsymbol{T}$, which gives the result

$$V_{\boldsymbol{x}} = \boldsymbol{x} \cdot \boldsymbol{T} = \boldsymbol{v} \cdot \left(\sum_{j=1}^{\mu} x_j \boldsymbol{s}_j \right) = \boldsymbol{v} \cdot \boldsymbol{\beta}_{\boldsymbol{x}}.$$

How do we choose $\boldsymbol{x}$? The following result partially answers this question

Theorem 2 *A sufficient condition for the bias of $\mathcal{A}_{\boldsymbol{x}}$ to be zero in average is $\mathbf{1} \cdot (\mathbf{1} - \boldsymbol{w}) = 0$ where $\boldsymbol{w} = x_1 \boldsymbol{s}_1 + \cdots + x_\mu \boldsymbol{s}_\mu$. Furthermore, under these conditions, standard deviation is proportional to $\|\mathbf{1} - \boldsymbol{w}\|_2^2$.*

Proof Suppose $V = V_0$ for some fixed V_0. Because V does not change by permuting the votes of the voters, adding all the $\boldsymbol{v}$'s that give V_0, we get a multiple of $\mathbf{1}$. Therefore, $E[\boldsymbol{v}|V = V_0] = \frac{V_0}{n}\mathbf{1}$. Therefore

$$\begin{aligned} E[V - V_{\boldsymbol{x}}|V = V_0] &= E[\boldsymbol{v} \cdot (\mathbf{1} - \boldsymbol{w})|V = V_0] \\ &= E[\boldsymbol{v}|V = V_0] \cdot (\mathbf{1} - \boldsymbol{w}) \\ &= \frac{V_0}{n}\mathbf{1} \cdot (\mathbf{1} - \boldsymbol{w}). \end{aligned}$$

Thus, if $\mathbf{1} \cdot (\mathbf{1} - \boldsymbol{w}) = 0$ then the bias is zero.

We now look at the variance: for such $\boldsymbol{w}$, we have $E[(V - V_{\boldsymbol{x}})^2|V = V_0] = c_{V_0}|\mathbf{1} - \boldsymbol{w}|^2$ for some c_{V_0} integer.

- When $V_0 = 0$, the statement is trivial as there is only one such $\boldsymbol{v}$, so there is no variance.
- Suppose that $V = 1$, that is $\boldsymbol{v}$ has the form $\{0, ..., 0, 1, 0, ..., 0\}$. Hence,

$$E[(V - V_{\boldsymbol{x}})^2|V = 1] = \sum_{j=1}^{n}(1 - w_j)^2 = |1 - \boldsymbol{w}|^2.$$

- Now suppose that $V = 2$, that is $\boldsymbol{v}$ has the form $\{0, ..., 0, 1, 0, ...0, 1, 0, ..., 0\}$. In this case,

$$\begin{aligned} E[(V - V_{\boldsymbol{x}})^2|V = 2] &= \sum_{i,j \in \{1,\ldots,n\}, i \neq j} ((1 - w_i) + (1 - w_j))^2 \\ = \sum_{i,j \in \{1,\ldots,n\}, i \neq j} &((1 - w_i)^2 + 2(1 - w_i)(1 - w_j) + (1 - w_j)^2). \end{aligned}$$

  Note that in the total sum, the kind of terms $2(1 - w_i)(1 - w_j)$ for some fixed i differing and j appear only once. On the other hand, the kind of terms of the form $(1 - w_i)^2$ for some i also appear the same number of times, namely as many times as 2 elements, can be chosen from the n such that one of the 2 is i, which is $n-1$. This means that the terms of the kind

$(1-w_i)^2$ appear $n-2$ times more than those of the kind $2(1-w_i)(1-w_j)$. Let $c_2 := n-2$. Then

$$E[(V-V_{\boldsymbol{x}})^2|V=2] - (n-2)\sum_i^n (1-w_i)^2$$
$$= \sum_{i,j\in\{1,\dots,n\}} (1-w_i)(1-w_j)$$

This, however is 0, because

$$\sum_{i\in\{1,\dots,n\}} (1-w_i) = \mathbf{1}\cdot(\mathbf{1}-\boldsymbol{w}) = 0$$

Hence, we obtain

$$E[(V-V_{\boldsymbol{x}})^2|V=2] = (n-2)\sum_i^n (1-w_i)^2 = (n-2)|1-\boldsymbol{w}|^2$$

- The same idea works for any $V_0 = m$: In this case,

  $$E[(V-V_{\boldsymbol{x}})^2|V=m] =$$
  $$\sum_{i_1,\dots,i_m\in\{1,\dots,n\},i_j\neq i_l} \Big(\sum_{j=1}^m (1-w_{i_j})\Big)^2.$$

  Again, once we carry out the squaring, in the total sum, the kind of terms $2(1-w_i)(1-w_j)$ for any fixed i differing and j appear exactly the same times; namely the number of times m elements can be chosen from n such that i and j are included. Let c''_m denote this number. On the other hand, the kind of terms of the form $(1-w_i)^2$ for some i also appear the same number of times, namely as many times as m elements can be chosen from the n such that i is among them. Let c'_m denote this number. Let $c_m := c'_m - c''_m$. Then the same as before,

  $$E[(V-V_{\boldsymbol{x}})^2|V=m] - c_m\sum_i^n (1-w_i)^2 = 0$$

  because $\sum_{i\in\{1,\dots,n\}}(1-w_i) = \mathbf{1}\cdot(\mathbf{1}-\boldsymbol{w}) = 0$. So we obtain

  $$E[(V-V_{\boldsymbol{x}})^2|V=m] = c_m\sum_i^n (1-w_i)^2 = c_m|1-\boldsymbol{w}|^2.$$

If $\boldsymbol{S}$ spans $\mathbb{R}^n$, then by definition of a generating family, we can find $\{x_1,\dots,x_\mu\}$ such that $\boldsymbol{w} = \mathbf{1}$.[2] Concretely, we can construct an orthonormal basis of $\mathbb{R}^n$ from vectors of $\boldsymbol{S}$ and project $\mathbf{1}$ onto each coordinate. We dub

[2]The average value of μ such that $\boldsymbol{S}$ spans $\mathbb{R}^n$ is $\sum_{k=1}^n \frac{2^k}{2^k-1}$. See [12] for more precise results.

this method of computing $\boldsymbol{x}$ the *minimum variance tally-combining algorithm* (MV, Table 4.1). When $\boldsymbol{S}$ span $\mathbb{R}^n$, the MV algorithm gives an exact result (zero bias and variance).

```
Input: $\boldsymbol{S} = \{\boldsymbol{s}_j\}$, $\boldsymbol{T}$, $\mu$, $n$
Output: $V_{\boldsymbol{x}}$, $\boldsymbol{x}$, $\boldsymbol{w}$

    1. $Z \leftarrow \emptyset$
    2. For each $\boldsymbol{s}_j \in \boldsymbol{S}$, if $\boldsymbol{s}_j$ is linearly independent from $Z$, $Z \leftarrow Z \cup \boldsymbol{s}_j$
    3. $\widehat{Z} \leftarrow$ GramSchmidtOrthogonalisation($Z$)
    4. For each $\hat{\boldsymbol{z}}_j$, let $\hat{x}_j \leftarrow \mathbf{1} \cdot \hat{\boldsymbol{z}}_j$
    5. $\boldsymbol{w} \leftarrow \sum_j \hat{x}_j \cdot \hat{\boldsymbol{z}}_j$
    6. $M \leftarrow (\boldsymbol{z}_j \cdot \hat{\boldsymbol{z}}_\ell)_{j,\ell}$
    7. $\boldsymbol{x} \leftarrow (M^\top)^{-1} \cdot \boldsymbol{w}$
    8. $V_x \leftarrow \boldsymbol{x} \cdot \boldsymbol{T}$
    9. Return $V_x, \boldsymbol{x}, \boldsymbol{w}$
```

Table 4.1: Algorithm for minimum variance tally combining (MV).

However, when $\boldsymbol{S}$ does not span $\mathbb{R}^n$, the MV algorithm can only find a vector $\boldsymbol{w}$ close to $\mathbf{1}$, namely the closest such vector in terms of Euclidean distance that can be expressed in terms of vectors in $\boldsymbol{S}$. This is still the solution resulting in the smallest variance, but no longer the solution with the least bias!

This leads us to consider the following approach: we can construct tally-combining algorithms that guarantee zero bias and select amongst these an algorithm that minimizes variance. Indeed, the constraint $\mathbf{1} \cdot (\mathbf{1} - \boldsymbol{w}) = 0$ can be guaranteed by determining x_1 as a linear function of other variables[3]. It remains to minimize $\|\mathbf{1} - \boldsymbol{w}\|_2^2$, which is simply a quadratic form in $\mu - 1$ variables. Therefore its minimum is easy to find as it amounts to solving a linear system in $\mu - 1$ rational variables. We call the corresponding algorithm the *zero-bias minimum variance tally-combining algorithm* (ZBMV, Table 4.2). In table 4.2, "symbolic expression" refers to the notion that $x_1, \ldots, x_\mu$ are not evaluated but are symbols to be manipulated formally.

4.6.1 Comparing tally-combining algorithms

Let's consider a toy example to illustrate how the three discussed tally-combining algorithms compare. Throughout this section, we take $n = 4$,

[3]There is nothing special about $\boldsymbol{s}_1$, any other vector of $\boldsymbol{S}$ can be used. Note that $\mathbf{1} \cdot \boldsymbol{s}_1 \neq 0$.

```
Input: $\boldsymbol{S} = \{\boldsymbol{s}_j\}$, $\boldsymbol{T}$, $\mu$, $n$
Output: $V_{\boldsymbol{x}}$, $\boldsymbol{x}$

  1. Let $x_1$ be the symbolic expression $\frac{1}{\mathbf{1}\cdot\boldsymbol{s}_1}\left(n - \sum_{j=2}^{\mu} x_j(\mathbf{1}\cdot\boldsymbol{s}_j)\right)$
  2. Let $D$ be the symbolic expression $\|\mathbf{1} - \sum_{j=1}^{\mu} x_j\boldsymbol{s}_j\|_2^2$
  3. $(x_2^\star, \ldots, x_\mu^\star) \leftarrow$ solutions of the linear system $\nabla D = 0$
  4. $x_1^\star \leftarrow \frac{1}{\mathbf{1}\cdot\boldsymbol{s}_1}\left(n - \sum_{j=2}^{\mu} x_j^\star(\mathbf{1}\cdot\boldsymbol{s}_j)\right)$
  5. $\boldsymbol{x} \leftarrow (x_1^\star, \ldots, x_\mu^\star)$
  6. $V_x \leftarrow \boldsymbol{x}\cdot\boldsymbol{T}$
  7. Return $V_x, \boldsymbol{x}$
```

Table 4.2: Algorithm for zero-bias minimum variance tally combining.

$M = 6$, $\mu = 3$, $k = 3$ and $\boldsymbol{s}_1 = (1,1,1,0)$, $\boldsymbol{s}_2 = (1,1,0,0)$, $\boldsymbol{s}_3 = (0,1,0,1)$ and $\boldsymbol{T} = (1,0,0)$.[4] The results are summarized in Table 4.3.

Tally-combining algorithm	**Bias**	**Variance**	**Tally**
	$\mathbf{1}\cdot(\mathbf{1}-\boldsymbol{w})$	$\|\mathbf{1}-\boldsymbol{w}\|_2^2$	$\boldsymbol{x}\cdot\boldsymbol{T}$
Naïve algorithm	-2/3	4/3	2/3
ZBMV	0	5/7	6/7
MV	1/3	1/3	1

Table 4.3: Comparison between tally-combining algorithms on the toy example.

Algorithm 1 (Zero-bias minimum variance) *We can express x_1 in terms of x_2 and x_3 to ensure zero bias:*

$$x_1 = \frac{1}{\mathbf{1}\cdot\boldsymbol{s}_1}(n - x_2(\mathbf{1}\cdot s_2) - x_3(\mathbf{1}\cdot s_3)) = \frac{1}{3}(4 - 2x_2 - 2x_3)\,.$$

We are left to determine x_2 and x_3, which we choose to minimize the distance

[4]Note that in this example, knowing the tallies t_1 and t_2 reveals one participant's vote. This privacy issue is addressed later in the paper.

of $\boldsymbol{w} = x_1\boldsymbol{s}_1 + \cdots + x_3\boldsymbol{s}_3$ to $\mathbf{1}$, i.e. the quantity

$$\begin{aligned}\|\mathbf{1} - \boldsymbol{w}\|_2^2 &= \sum_{i=1}^{n} (1 - w_i)^2 \\ &= (1 - x_1 - x_2)^2 + (1 - x_1 - x_2 - x_3)^2 \\ &+ (1 - x_1)^2 + (1 - x_3)^2 \\ &= \frac{1}{3}(4 + 5x_2^2 + 2x_2(x_3 - 3) + 3x_3^2 - 2x_3)\end{aligned}$$

This achieves its global minimum value of $5/7$ at $x_2^\star = 4/7$ and $x_3^\star = 1/7$. Therefore, we have:

$$\boldsymbol{x} = \frac{1}{7}(6, 4, 1).$$

In particular, $\boldsymbol{w} = x_1^\star\boldsymbol{s}_1 + \cdots + x_3^\star\boldsymbol{s}_3 = \frac{1}{7}(10, 11, 6, 1)$ (note that computing this vector is not necessary for the algorithm).

Algorithm 2 (Minimum variance) *We begin by computing an orthonormal basis $\hat{Z}$ from $\boldsymbol{S}$:*

$$\hat{z}_1 = \frac{1}{\sqrt{3}}(1, 1, 0, 0) \qquad \hat{z}_2 = \left(\frac{1}{\sqrt{6}}, \frac{1}{\sqrt{6}}, -\sqrt{\frac{2}{3}}, 0\right)$$

$$\hat{z}_3 = \left(-\frac{1}{\sqrt{6}}, \frac{1}{\sqrt{6}}, 0, \sqrt{\frac{2}{3}}\right)$$

which gives $\hat{x}_1 = \sqrt{3}$, $\hat{x}_2 = 0$, $\hat{x}_3 = \sqrt{2/3}$, from which we get $\boldsymbol{w} = \frac{1}{3}(2, 4, 3, 2)$ and finally

$$\boldsymbol{x} = \left(1, -\frac{1}{3}, \frac{2}{3}\right).$$

As expected, this tally-combining algorithm has smaller variance (since $\|\mathbf{1} - \boldsymbol{w}\|_2^2 = 1/3$) compared with the ZBMV algorithm in of Algorithm 1, but its bias is not guaranteed to be zero (since $\mathbf{1} \cdot (\mathbf{1} - \boldsymbol{w}) = 1/3$).

Algorithm 3 (Naïve tally combining) *Let's use the naïve tally-combining algorithm, i.e.,*

$$\boldsymbol{x} = \frac{M}{\mu k}\mathbf{1}.$$

We assume here that $M = 6$, $\mu = 3$ and $k = 3$ so that $\boldsymbol{x} = \frac{2}{3}\mathbf{1}$, yielding $\boldsymbol{w} = (\frac{4}{3}, 2, \frac{2}{3}, \frac{2}{3})$. The bias for this algorithm is $-2/3$, however this algorithm has larger variance than the other two, since $\|1 - \boldsymbol{w}\|_2^2 = 4/3$.

4.7 Privacy of Parallel OV-Net

In this section, we investigate the decrease in privacy, which we can expect due to the multiple parallel elections which can be tallied individually, thus giving the adversary extra information. As an example, let us consider a simple yes/no referendum. If all voters happen to vote the same, we have, of course, lost privacy. However, the probability of this might be small. However, if we split the voters into two elections, the probability is roughly the square root of the old probability, i.e. much higher.

Recall that M is the number of the parallel and independent elections, n is the total number of voters, and k is the number of elections that each voter has randomly chosen to participate in. We denote by M_i the set of voters who participated in election i, and we consider that the elections are enumerated from 1 to M. Let $\mathrm{Res}(M_i)$ be the random variable that gives the number of 'Yes' votes in the set M_i. We also recall that Y_i is the random variable that gives the number of voters in the set M_i.

4.7.1 Definitions and Assumptions

To quantify privacy, we use the δ-privacy definition for voting from [36], which assumes that, besides the voting elements of a voting protocol, there exists an additional party called an observer O, who can observe publicly available information. Moreover, we assume that among the n honest voters, there exists a voter V_{obs} who is under observation. For the sake of clarity, V_{obs} will refer at the same time to the voter under observation and to its vote.

Definition 11 Let P be a voting protocol and V_{obs} be the voter under observation. We say that P achieves δ-privacy if the difference between the probabilities

$$\mathbb{P}[(\pi_O || \pi_{V_{\text{obs}}}(\text{Yes}) || \pi_v)^{(l)} \to 1]$$

and

$$\mathbb{P}[(\pi_O || \pi_{V_{\text{obs}}}(\text{No}) || \pi_v)^{(l)} \to 1]$$

is δ-bounded as a function of the security parameter ℓ, where π_O, $\pi_{V_{\text{obs}}}$ and π_v are respectively the programs run by the observer O, the voter under observation V_{obs} and all the honest voters v (clearly without V_{obs}).

To calculate the privacy we use the following result from [36]

$$\delta(n) = \sum_{r \in M^*_{\text{Yes,No}}} (A_r^{\text{No}} - A_r^{\text{Yes}}) \tag{4.1}$$

where $M^*_{\text{Yes,No}} = \{r \in \mathbb{R} : A_r^{\text{Yes}} \leq A_r^{\text{No}}\}$, $\mathbb{R}$ is the set of all possible election results and A_r^j denotes the probability that the choices of the honest voters yield the result r of the election given that V_{obs}'s choice is j.

We consider a referendum with n honest voters with a uniform distribution between yes and no votes. For simplicity, we will assume that nobody abstains. We also assume that no voters are corrupted. This is reasonable since instructing corrupted voters to vote in a special way does not give further advantage compared to simply knowing the corrupted voters' votes. Moreover, we assume that at least one of the elections in which V_{obs} participated is surviving.

4.7.2 Basic Cases: $M = k = 1$ and $M \geq 1, k = 1$

The δ for a single referendum is :

$$\begin{aligned}\delta(n) &= \left(\frac{1}{2}\right)^n \frac{1}{n} \sum_{a=0}^{n} \binom{n}{a} |2a - n| \\ &= \begin{cases} 2^{-n}\binom{n}{\frac{n}{2}} & \text{if } n \text{ is even} \\ \dfrac{2^{1-n}}{n}\binom{n}{1+[\frac{n}{2}]}\left(1 + \left[\dfrac{n}{2}\right]\right) & \text{Otherwise} \end{cases}\end{aligned}$$

where the first equality holds using the result from (4.1) and the second one using the binomial theorem.

The formula above refers to the case $M = k = 1$ where all voters had chosen to vote in the same and unique election 1. For the case $M > 1$ and $k = 1$, δ becomes a random variable and the expected value of δ of the election in which V_{obs} is participating can be defined as follows:

$$\delta_{\text{expected}}(n, M) = \sum_{n'=1}^{n} \mathbb{P}(Y_i' = n')\delta(n') \tag{4.2}$$

where Y_i' is the random variable that gives the number of voters who participated in the election i, including V_{obs}; and $Y_i' \sim 1 + \text{BD}(n-1, \frac{k}{M})$. Equation (4.2) for $k = 1$ and $M > 1$ becomes:

$$\delta_{\text{expected}}(n, M) = \sum_{n'=1}^{n} \binom{n-1}{n'-1} \left(\frac{1}{M}\right)^{n'-1} \left(1 - \frac{1}{M}\right)^{n-n'} \delta(n')$$

Figure 4.4 shows that privacy is almost lost when $M \gg n$.

4.7.3 General Case

In this part, we give a general formula of δ. To this end, we consider the following. Let $y = (y_1, \dots, y_M)$ be an assignment of voters such that $\text{Card}(M_i) = y_i$ for $i \in [1, M]$. We can obtain all the possible assignments of voters by respecting the condition $\sum_{i=1}^{M} y_i = nk$. Let $r = (r_1, \cdots, r_M)$ be a possible result corresponding to the assignment y with $r_i = \text{Res}(M_i)$ for $i \in [1, M]$. r verifies the conditions $(\sum_{i=1}^{M} r_i) \bmod k = 0$ and $r_i \leq y_i$ for $i \in [1, M]$. Remember that $\text{Res}(M_i)$ gives the number of "Yes" votes in M_i. We have $\text{Res}(M_i) \sim \text{BD}(y_i, \frac{1}{2})$ for $i \in [1, M]$. Intuitively, δ can be expressed as the following:

$$\begin{aligned}\delta(n, M, k) = &\sum_{y_1 + \cdots + y_M = nk} \mathbb{P}(Y_1 = y_1, \dots, Y_M = y_M) \\ &\cdot \sum_{r \in M^*_{\text{Yes,No}}} (A_r^{\text{No}} - A_r^{\text{Yes}})\end{aligned}$$

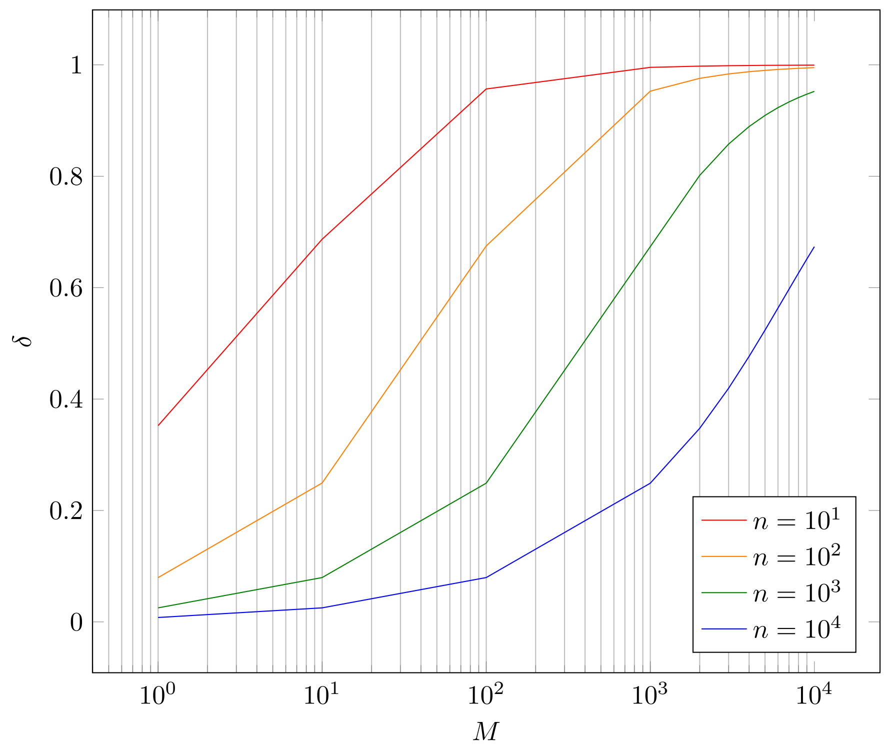


Figure 4.4: The relationship between M and δ_{expected} for different values of $n = 10, 10^2, 10^3, 10^4$.

By definition of A_r^j we have

$$A_r^j = \mathbb{P}(\mathrm{Res}(M_1) = r_1, \dots, \mathrm{Res}(M_M) = r_M / V_{\mathrm{obs}} = j)$$

with $j \in \{\text{Yes, No}\}$.

To proceed, we will introduce an additional notation. Remember that M_i denotes the voters in election i. Define Σ_k as the subsets of $\{1, \dots, M\}$ of cardinality k. For $\sigma \in \Sigma_k$, we define $M'_\sigma = \bigcap_{i \in \sigma} M_i$, i.e. the voters participating in the elections in the set σ. Note that the assignment of voters to elections is uniformly random, i.e. each voter is assigned uniformly and uniquely to an M'_σ. Also Z_σ is the random variable determining the number of voters in M'_σ.

There are $c = \binom{M}{k}$ possible M'_σs . Suppose that σs are enumerated from 1 to c. Let $z = (z_{\sigma_1}, \dots, z_{\sigma_c})$ be an assignment of voters such that $z_{\sigma_i} = \mathrm{Card}(M'_{\sigma_i})$, for $(\sigma_i, i) \in \Sigma_k \times [1, c]$. All the possible assignments of voters z are obtained by respecting the condition $\sum_{\sigma_i \in \Sigma_k} z_{\sigma_i} = n$.

The variables Z_σ, $\sigma \in \Sigma_k$ correspond to the problem of putting n indistinguishable balls into c distinguishable boxes, i.e. the vector $Z = (Z_{\sigma_1}, \dots, Z_{\sigma_c})$ follows a multinomial distribution with equal parameters $p_i = 1/c$, and $\sum_{\sigma \in \Sigma} z_\sigma = n$ including V_{obs}. We can now calculate the probability for the assignment of the voters, and rewrite our formula as:

$$\delta(n, M, k) = \sum_{z_1 + \cdots + z_c = n} \mathbb{P}(Z_{\sigma_1} = z_{\sigma_1}, \cdots, Z_{\sigma_c} = z_{\sigma_c}) \cdot \sum_{r \in M^*_{\mathrm{Yes,No}}} (A_r^{\mathrm{No}} - A_r^{\mathrm{Yes}})$$

Let $r' = (r'_{\sigma_1}, \dots, r'_{\sigma_c})$ such that $r'_{\sigma_i} = \mathrm{Res}(M'_{\sigma_i})$ for $(\sigma_i, i) \in \Sigma_k \times [1, c]$. The variables $\mathrm{Res}(M'_\sigma)$, $\sigma \in \Sigma_k$, are independent and follow the binomial distribution of parameters z_σ and $1/2$.
In the case $M = c$, which means $k = M - 1$ or $k = 1$, there is a one-to-one correspondence between the sets $(M_i)_{i \in [1,M]}$ and $(M'_\sigma)_{\sigma \in \Sigma_k}$. However, this is not true in general and we have a relation between r and r' defined by the function f as follows:

$$\begin{pmatrix} r_1 \\ \vdots \\ r_M \end{pmatrix} = B \cdot \begin{pmatrix} r'_{\sigma_1} \\ \vdots \\ r'_{\sigma_c} \end{pmatrix} = f(r'_{\sigma_1}, \cdots, r'_{\sigma_c})$$

where $B = (b_{i\sigma})_{\substack{1 \le i \le M \\ \sigma \in \Sigma_k}}$ and $b_{i\sigma} = \mathbf{1}_{i \in \sigma}$.

We can now calculate the probability A_r^{v} as:

$$A_r^{\mathrm{v}} = \sum_{r' | r = f(r')} A'^{\mathrm{v}}_{r'}$$

and we have

$$A'^{\mathrm{v}}_{r'} = \mathbb{P}(\mathrm{Res}(M'_{\sigma_1}) = r'_{\sigma_1}, \cdots, \mathrm{Res}(M'_{\sigma_c}) = r'_{\sigma_c} / V_{\mathrm{obs}} = \mathrm{v})$$

Suppose that V_{obs} is in the subset M'_{σ_1}. It is symmetric to choose any other subset. We have:

$$A'^{\text{v}}_{r'} = \left(\frac{1}{2}\right)^{z_{\sigma_1}-1} \cdot h(z_{\sigma_1}, r'_{\sigma_1}) \cdot \prod_{i=2}^{c} \left(\frac{1}{2}\right)^{z_{\sigma_i}} \cdot \binom{z_{\sigma_i}}{r'_{\sigma_i}}$$

where

$$h(x,y) = \begin{cases} \binom{x-1}{y-1} & \text{if v = "Yes"} \\ \binom{x-1}{y} & \text{if v = "No"} \end{cases}$$

Remember that

$$M^*_{\text{Yes,No}} = \{r' : A'^{\text{Yes}}_{r'} \leq A'^{\text{No}}_{r'}\},$$

and $A'^{\text{No}}_{r'} \geq A'^{\text{Yes}}_{r'}$ is true when $r'_{\sigma_1} \in [0, [\frac{z_{\sigma_1}}{2}]]$. We have

$$\sum_{r'_1=0}^{[\frac{z_{\sigma_1}}{2}]} (A'^{\text{No}}_{r'} - A'^{\text{Yes}}_{r'}) = \frac{1}{2} \sum_{r'_1=0}^{z_{\sigma_1}} |A'^{\text{No}}_{r'} - A'^{\text{Yes}}_{r'}|.$$

Since V_{obs} is in M'_{σ_1}, the vector to consider is

$$Z' = (Z_{\sigma_1} - 1, Z_{\sigma_2}, \cdots, Z_{\sigma_c}).$$

The formula of δ becomes

$$\begin{aligned} \delta(n, M, k) &= a_n \cdot \sum_{z_{\sigma_1}=1}^{n} \frac{E(z_{\sigma_1})}{z_{\sigma_1}!} \sum_{z_{\sigma_2}=0}^{n} \cdots \sum_{z_{\sigma_c}=0}^{n} \frac{\delta_{\sum_{\sigma \in \Sigma} z_\sigma, n}}{z_{\sigma_2}! \cdots z_{\sigma_c}!} \\ &= a_n \cdot \sum_{z=1}^{n} \frac{E(z)}{z!} \cdot \frac{(c-1)^{n-z}}{(n-z)!} \end{aligned}$$

with

$$a_n = \frac{(n-1)!}{c^{n-1}} \cdot \left(\frac{1}{2}\right)^n, \quad E(z) = 2^{n-z+1} \binom{z}{[\frac{z}{2}]} \cdot \left[\frac{z}{2}\right]$$

and $\delta_{i,j}$ is the Kroenecker delta function.

4.8 Conclusion and Future Work

Conclusion In this chapter, we presented a new version of the protocol OV-Net which runs multiple elections in parallel to achieve robustness against DoS failures without resorting to additional time-consuming rounds. We quantitatively computed the increase in robustness by having M parallel elections with each voter participating in k of them, and demonstrated how robustness can be significantly improved. The improvement in time and robustness, come at a cost in terms of accuracy and privacy. We presented three different algorithms on how to optimally compute the tally using this new OV-Net version and we

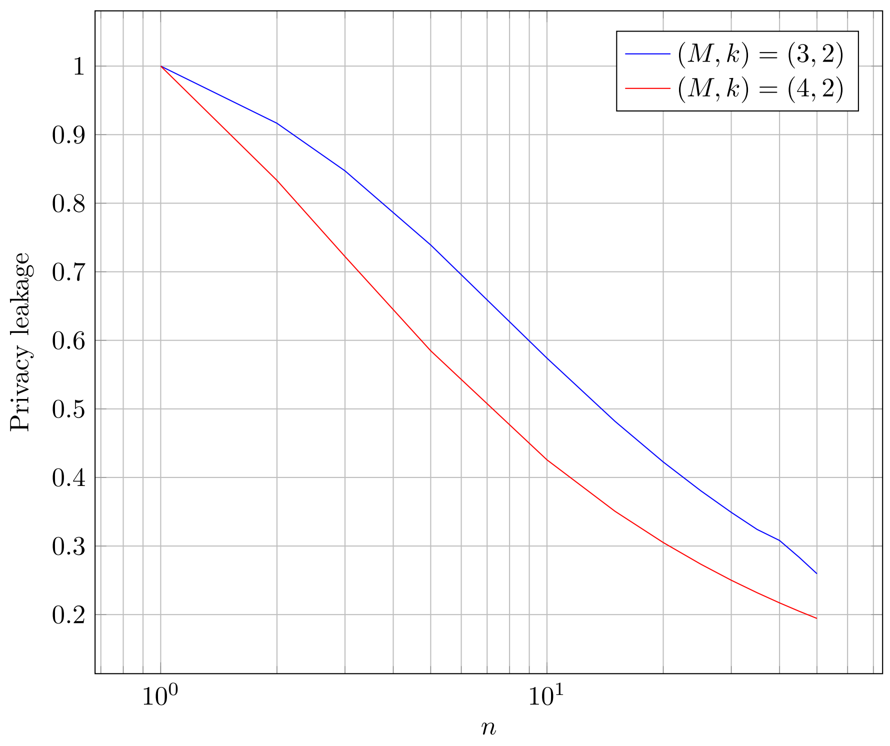


Figure 4.5: Privacy leakage as function of n for the cases $(M, k) = (3, 2), (4, 2)$.

quantitatively measured the privacy decrease that is expected due to the multiple partial election results. The results allow the protocol initiator to choose parameters to carefully balance the wanted robustness with a controlled privacy loss and statistical loss in accuracy, as well as increased computation.

Further work Redistribution is the process by which elections are conducted in several electoral districts. Unlike general elections, where the final result is known for the entire country only, in redistributed elections, each voter votes in their own district; results are consolidated per district and only then added up. Perhaps it is possible to confine problematic voters to a district of their own: partition the n voters into d districts of $n' = n/d$ voters, then run a vote in each of them. Then recompose the result by adding up the final tally.

This strategy confines the DoS problem to districts that do not influence each other. However, DoS tolerance is not exactly multiplied by d because each district is not allowed to exceed k unresponsive voters. In other words, tolerance is multiplied by d as long as the constraint that there are no more than k unresponsive voters per district is respected.

Chapter 5

Taphonomical Security: (DNA) Information with a Foreseeable Lifespan

This work has been done in collaboration with: David Naccache, Peter B. Rønne, Marcel Hollenstein, and Peter YA Ryan, and was accepted to FICC23 in San Francisco, USA.

Contents

Summary

This work introduces the concept of information with a foreseeable lifespan and explains how to achieve this primitive via a new method for encoding and storing information in DNA-RNA sequences.

The storage process can be divided into three time-frames. Within the first (life), we can easily read out the stored data with high probability. The second time-frame (agony) is a parameter-dependent state of uncertainty; the data is not easily accessible, but still cannot be guaranteed to be inaccessible. During the third (death), the data can with high probability not be recovered without a large computational effort which can be controlled via a security parameter. The quality of such a system, in terms of a foreseeable lifespan, depends on the brevity of the agony time-frame, and we show how to optimise this.

In the present paper, we analyse the use of synthetic DNA and RNA as a storage medium since it is a suitable information carrier, and we can manipulate the RNA nucleotide degradation rate to help control the lifespan of the message embedded in the synthesized DNA/RNA molecules. Other media such as Bisphenol A thermal fax paper or unstable nonvolatile memory technologies can be used to implement the same principle but the decay models of each of those phenomena should be re-analysed and the formulae given in this paper adapted correspondingly.

Keywords: Cryptography, Information with foreseeable lifespan, Data Storage, Information theory, DNA, RNA.

5.1 Introduction

Over time, the physical media on which we store information degrades. Traditionally, much effort has been put into *protecting* media against degradation to achieve more robust and durable storage mechanisms.

In this chapter, instead of resisting the time's unavoidable effects, we try to *exploit* them: rather than allowing information to slowly and progressively get destroyed, we aim at a swift and complete erasure. Just as a thermal fax machine paper that fades with time, we propose synthesising DNA and RNA molecules whose lifetime can be approximately tuned. Such a "time fuse" can guarantee, for instance, that a cryptographic secret (typically a plaintext encrypted under a hash of the DNA information) cannot be used or recovered beyond some expiry date.

Since DNA is a reasonably stable molecule, we assume in this paper that DNA does not degrade at all. By contrast, RNA nucleotides quickly decay over time. Hence, we propose to incorporate RNA nucleotides in DNA molecules synthetically. The DNA nucleotides will store the cryptographic secret whereas RNA will serve as a natural countdown mechanism. This technique guarantees that, with high probability, the whole secret will be recoverable before some target time $t_{\text{target } 1}$ but will not be reconstructible after $t_{\text{target } 2}$. A mathematical analysis allows tuning the t_i as a function of the molecules' molecular decay probability distribution and the storage environment parameters such as temperature and exposure to radiation.

5.2 Biochemical Preliminaries

For the article to be understandable, it is necessary to present a few biochemical notions about DNA and RNA [49]. The following sections will deal with DNA and RNA composition and degradation. The acronym "NA" (Nucleic Acid) will denote both DNA and/or RNA.

5.2.1 NA Composition

DNA and RNA belong to the category of NAs, which are bio-macro-molecules; both are chains of nucleotides. A nucleotide comprises a nucleobase, a pentose sugar and one phosphate group. In nature, there exist five different nucleotides: adenine (A), thymine (T), uracil (U), cytosine (C) and guanine (G). DNA contains A,T,C,G, whereas RNA contains A,U,C,G. Figure 5.1 shows the structures of NAs, while Figure 5.2 details the four DNA nucleotides.

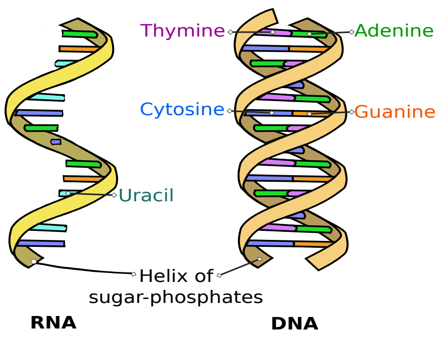


Figure 5.1: RNA and DNA structures

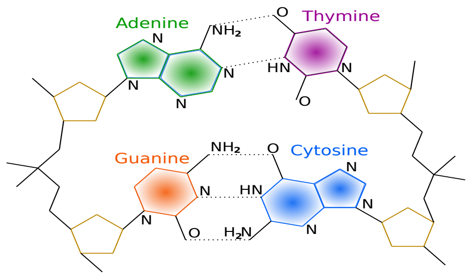


Figure 5.2: Closeup structure of DNA with the four nucleotides represented: Adenine, Thymine, Cytosine, and Guanine.

5.2.2 NA Bonds Degradation Over Time

NAs degrade over time. The main degradation reaction of RNA nucleotides is called *transesterification*, while the main degradation phenomena for DNA are *phosphodiester hydrolysis*, *oxidative cleavage*, and *cleavage* as a result of *depurination*. Whilst we will not dive deeper into the particularities of those biochemical processes, the reader may wonder why the same degradation reactions do not apply to both DNA and RNA. This results from the fact that the difference in the pentose considerably influences on the reactions leading to degradation. Briefly, in DNA, a hydroxide ion (OH -) will attack the phosphorous center, eventually leading to hydrolysis of the phosphodiester bond. Such a mechanism can also occur in RNA, but since RNA displays a 2'-OH moiety, this hydroxyl group can be activated (by deprotonation under basic conditions or by metal coordination) and attack the phosphorous center in an intramolecular rather than in an intermolecular manner [55, 43]. In particular, there is a big difference between the two degradation speeds: under representative physiological conditions, RNA hydrolysis is 10^5 times faster than DNA hydrolysis. DNA degradation speed is hence almost negligible compared to RNA degradation. In what follows, we are chiefly interested in considering RNA degradation in our mathematical analysis. Figure 5.3 illustrates how RNA degrades.

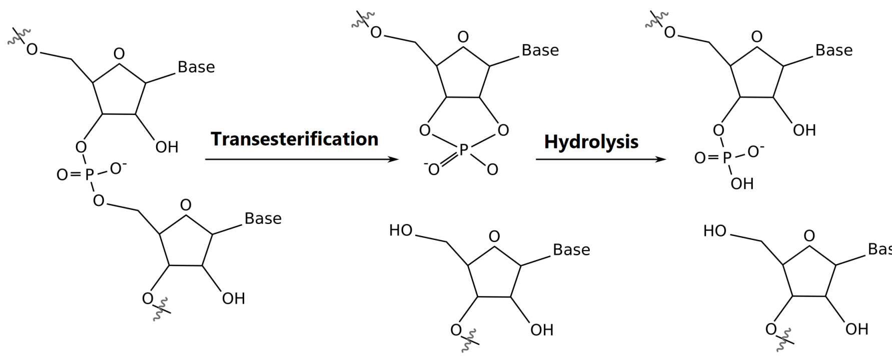


Figure 5.3: RNA degradation through transesterification

5.2.3 On the Synthesis of RNA-DNA Chimeric Oligonucleotides

Oligonucleotides can be generally synthesized by two main approaches: a chemical synthesis based on solid-phase methods and enzymatic synthesis. In this section, we will briefly describe both methods and then highlight how DNA-RNA chimeric oligonucleotides required for our purposes could be made.

Chemical synthesis

Automated DNA and RNA solid-phase synthesis grants the main synthetic access to RNA and DNA oligonucleotides. In this approach, activated nucleoside units called phosphoramidites are sequentially added to a first nucleoside bound to a solid support. Each cycle encompasses a coupling step (where the incoming phosphoramidite is reacted with a free 5'-OH unit of solid support bound nucleoside) followed by a capping step (to avoid the reaction of unreacted hydroxyl moieties in subsequent steps). The coupling and capping steps are followed by oxidation of the newly created linkages (P(III) to P(V)) and removal of the next protecting group to enable the continuation of the synthesis (the interested reader is directed to more comprehensive review articles dedicated to this topic [34, 41]). Such syntheses are usually carried out on synthesizers (Figure 5.4) on scales ranging from µmoles to moles. This method is routinely used to synthesize DNA and RNA oligonucleotides either based on standard chemistry or encompassing chemical modifications required for in vivo applications. However, this method is restricted to rather short (i.e. around 100-150 nucleotides for DNA and around 100 nucleotides for RNA [34, 41, 18] oligonucleotides due to low yields for fragments exceeding 100 nucleotides because of folding on solid support during synthesis and due to the inherent nature of the coupling yields (even with a 99% coupling efficiency, the maximum theoretical yield that can be obtained for 100 nucleotide long sequence would be 0.99100 37%). Hence, this approach is ideal for synthesizing short DNA-RNA chimeric oligonucleotides but is unlikely to apply to longer sequences.

Figure 5.4: DNA synthesizer used for solid-phase synthesis of DNA and RNA oligonucleotides.

Enzymatic Methods

The most popular enzymatic methods for the synthesis of oligonucleotides include polymerase-assisted synthesis using nucleoside triphosphates and ligation of shorter fragments into long oligonucleotides. In the first strategy, nucleoside triphosphates are recognized by enzymes called polymerases which add these nucleotides onto a growing chain of DNA or RNA. For DNA synthesis, the presence of a primer and a template are strictly required since the polymerase will add nucleotides at the 3'-end of the primer while the sequence composition of the template will dictate the polymerase which nucleotide needs to be incorporated. On the other hand, RNA polymerases are primer independent and only require the presence of a DNA template to mediate transcription of DNA into RNA. Such a method can be coupled with chemical modifications to generate mRNA vaccines [45, 28] and other functional nucleic acids [10, 33]. This method is not restricted to any size limitation and is compatible with numerous chemical modifications. On the other hand, the sequence specific incorporation of distinct RNA nucleotides in long DNA oligonucleotides will be difficult to achieve by this method.

In the second method, DNA or RNA ligases mediate the formation of phosphodiester linkages between the terminal 3'-OH residue of an oligonucleotide with the 5'-end (usually phosphorylated) of a second oligonucleotide [54] . Often a “splint” oligonucleotide is required as a template since this guide oligonucleotide is partially complementary to the termini of both oligonucleotides that need to be ligated together. This method is compatible with the synthesis of longer oligonucleotides [46] as well as with different chemistries in oligonu-

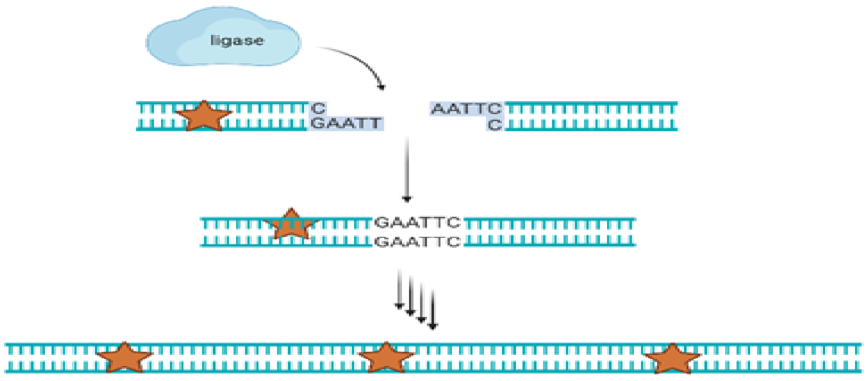


Figure 5.5: Schematic representation of the synthesis of long DNA oligonucleotides containing RNA nucleotides (star symbols) using a combination of solid-phase synthesis (short blue fragments) and DNA ligation reactions.

cleotides [29, 39], and hence is deemed as the method of choice for this project.

Synthesis of RNA-DNA Chimeric Oligonucleotides

To synthesize long DNA oligonucleotides containing RNA nucleotides at distinct and specific positions in the future, we propose synthesizing short DNA-RNA sequences using solid-phase synthesis and combining these fragments by (repeated) ligation reactions, as highlighted in Figure 5.5. This protocol will circumvent the drawbacks of all the different methods and should yield the desired oligonucleotides.

5.2.4 RNA Degradation in Further Detail

RNA nucleotide degradation probability follows an exponential distribution of parameter λ. λ depends on numerous factors such as temperature, pH and the concentration of some ions[1]. Degradation also depends on the sequence context and the 3D structure of the RNA oligonucleotide.

We will use Equation (5.1) of [37] to model λ:

$$\lambda = \lambda_0 \cdot 10^e \cdot c_K \cdot [K^+]^{d_K} \cdot c_{\mathrm{Mg}} \cdot [\ \mathrm{Mg}^{2+}]^{d_{\mathrm{Mg}}} \tag{5.1}$$

where

$$e = a_{\mathrm{pH}}(\mathrm{pH} - b_{\mathrm{pH}}) + a_K([K^+] - b_K) + a_T(T - b_T)$$

$\lambda_0 = 1.3 \cdot 10^{-9}\ \mathrm{min}^{-1}$ and the constants are indicated in Table 5.1 with their corresponding units.

1 $[K^+]$, $[\mathrm{Mg}^{2+}]$

constant	a_{pH}	b_{pH}	a_{K}	b_{K}	c_{K}
value	0.983	6	0.24	3.16	3.57
unit	none	none	$\mathrm{L.mol}^{-1}$	$\mathrm{mol.L}^{-1}$	$(\mathrm{mol.L}^{-1})^{-d_k}$

constant	d_K	c_{Mg}	d_{Mg}	a_{T}	b_{T}
value	-0.419	69.3	0.80	0.07	23
unit	none	$(\mathrm{mol.L}^{-1})^{-d_{\mathrm{Mg}}}$	none	$°\mathrm{C}^{-1}$	°C

Table 5.1: Values of the constants in Equation (5.1).

Note that Equation (5.1) is an approximation, and λ_0 needs to be updated depending on the considered range of physical parameters. See [37] for more details. We give in Table 5.2 several λ values under different conditions and the corresponding λ_0 values. The expected time for one RNA nucleotide to degrade (which is $1/\lambda$) is given in the table to get an idea of the order of magnitude of time. This will prove helpful in the rest of the paper to determine which λ to work with when tuning the information's lifetime. The chemical parameters mentioned in this table are taken from Table 1 and using Equation (e) from [37].

T(°C)	pH	$[K^+]$	$[\mathrm{Mg}^{2+}]$	λ_0(mins^{-1})	λ(mins^{-1})	$1/\lambda$
23	13	0.1	0	$1.3 \cdot 10^{-9}$	10^{-3}	1000 minutes
23	12.5	0.03	0	$1.3 \cdot 10^{-9}$	$4,5 \cdot 10^{-4}$	37 hours
37	7.4	0.25	0.005	$1.4 \cdot 10^{-7}$	$4.06 \cdot 10^{-5}$	17.1 months
4	10.7	0.25	0.005	$1.3 \cdot 10^{-9}$	$3.3 \cdot 10^{-6}$	210.43 days
23	7	0.25	0.005	10^{-8}	$1.22 \cdot 10^{-7}$	15.5 years

Table 5.2: Order of magnitude of the time for different values of λ

5.3 The Proposed Method

This section presents our new method for encoding and storing information using DNA and RNA nucleotide. We propose a method to synthesise a new DNA/RNA molecule and we show that we can achieve a good security level with this method.

5.3.1 Description of the Method

The idea is to incorporate RNA fragments into DNA oligonucleotides using standard solid-phase synthesis and produce DNA-RNA chimeric sequences to form a new DNA/RNA chimeric oligonucleotide. A DNA fragment can be composed of one nucleotide base (A, C, T, G) or a juxtaposition of several nucleotides bases linked together $(AA, CC, TT, GG, AC, AT, GT, ACT, \cdots$ etc). The chain's length depends on the size of the key that we want to encode and

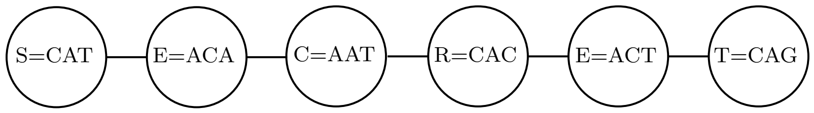


Figure 5.6: An explanatory illustration of one copy of the DNA/RNA oligonucleotide encoding the key `SECRET`. Beads represent DNA fragments and inter-bead links are RNA nucleotides.

store. This DNA/RNA chimeric oligonucleotide will contain k RNA nucleotides and $k+1$ DNA fragments. We synthesize n copies of this molecule and keep it in a fluid.

To understand the insertion/encoding mechanism, we refer the reader to Figure 5.6, which illustrates the insertion of the key `SECRET`. In this example, we have 5 RNA nucleotides and 6 DNA fragments and an alphabetic substitution for each letter in which we arbitrarily assigned different fragments to different letters of the English alphabet.

Note that below we will actually use distinct DNA molecule fragments and encode the stored information into the permutation of these.

5.3.2 Encrypting Information

Encrypt the information to be time-protected using some symmetric cipher (e.g. AES [26]) and encode the key as a DNA/RNA oligonucleotide using a permutation of pairwise distinct DNA fragments.

Erase the plaintext and the electronic version of the key and assume that the ciphertext is accessible by the opponent. Hence, the plaintext is recoverable as long as we can reconstruct the key from the DNA/RNA oligonucleotide.

We will now focus our attention on the recovery of the key from the DNA/RNA oligonucleotide, as long as this molecule is physically reconstructible.

5.3.3 Key Reconstruction

We assume that the DNA fragments are pairwise distinct by construction. Call the DNA/RNA oligonucleotide w and remember that it contains k RNA nucleotides. Remember as well that there are n copies of w floating in a liquid. Suppose that all the copies of w are cut randomly in pieces. We are given the set of these pieces, and we seek to restore the initial oligonucleotide w if such a reconstruction is still possible. Figure 5.7 shows the evolution of 3 copies of a key made of 9 DNA fragments.

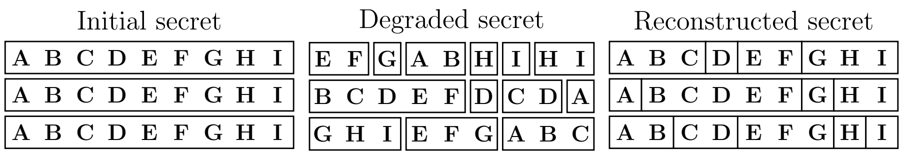


Figure 5.7: An evolution of a secret with 3 copies and 9 fragments in each copy

The following algorithm outlines how we can recover the information after it has begun degrading if such a reconstruction is possible at all.

- ① First, we can easily obtain all fragments of w by analysing the pieces we are given. For each fragment x, we will try to find the "next" fragment next[x] (the one which follows x in the molecule w). If for all fragments except one (which is w's last fragment), the next fragment is found, we can restore w.

- ② For any two fragments x and y, if there exists a third piece where y follows x, then in the initial molecule w, y also follows x, and thus next[x] $= y$. Therefore, we have just to treat each piece as follows: for every fragment x except the last, define next[x] as the fragment following x inside that piece. Since all fragments of w were distinct, there is at most one possible value of next[x] for all fragments x. Figure 5.8 represents this relation in a graph.

- ③ After this procedure, we have to find a fragment which follows nothing, and if it's unique, we set it as the first and then add the next fragments one by one until there is nothing to add. This allows us to reconstruct w. If there are several such fragments, w cannot be recovered without brute-force guessing.

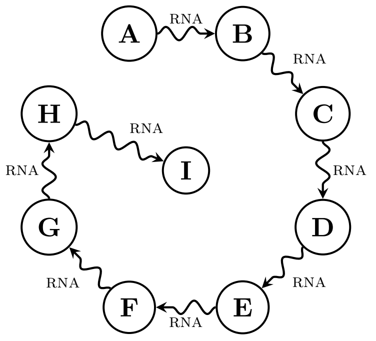


Figure 5.8: The graph representing the next[x] relation in the algorithm

If we have several fragments following nothing at the end of the algorithm, it is impossible to recover the initial molecule having no information except the input. We can only obtain separated pieces of w applying the last step of the algorithm to each of the "first" fragments. This situation happens if and only if at least one *cut* occurred during degradation, i.e. in all molecules, the RNA nucleotides broke at the same specific location in all the molecule. We will explore the likelihood of this happening in the next section.

5.3.4 Security

We now turn to measure our method's security,

but before doing so, let us define what the term *cut* means.

Definition 1 A *cut* happens at the i^{th} position if for all $1 \leq i \leq k$, the i^{th} bond of each of the n copies is broken.

Assume that the secret contains cuts. For each cut, the next fragment after this cut follows nothing according to the reconstruction algorithm. But, simultaneously, for all other fragments except the first one of the initial molecule, there is at least one piece where this fragment follows another one. This means that the only pieces which can be recovered are any piece delimited by two cuts or the pieces delimited by one cut and an endpoint of the initial molecule.

Since we cannot guess the link between two described pieces, the only strategy to recover the whole initial molecule is to test all possible permutations of the reconstructed pieces. If there are k cuts, then $(k+1)!$ possibilities need to be browsed. This fact defines security: a given number of cuts guarantees a security of ≈ 80, 100, 128, 256 bits. (A trivial $\log_2(k+1)!$ lookup table given in Table 5.3 for the ease of quick reference) provides the correspondence between the number of security bits versus the number of cuts and DNA fragments. The number of security bits, which is called the security parameter and that we denote a, is simply: $a = \log_2(n_{\text{D}}!)$, where n_{D} is the number of the DNA fragments.

Security bits $a = \log_2(k+1)!$	Number of cuts
$a = 84$	24
$a = 103$	28
$a = 133$	34
$a = 260$	57

Table 5.3: The number of security bits versus the number of cuts and DNA fragments. Note that the number of DNA fragments needed is always the number of cuts plus one.

Current biological limitations. It is currently technically feasible to have an NA chain of about 100 nucleobase pairs [34, 41, 18]. Each DNA chunk is linked by an RNA pair. In the security analysis in this paper, we assume that each DNA chunk is unique since copies reduce the security (and would makes the following analysis harder).

Since the security stems directly from the number of RNA fragments, we should construct chains containing the maximal number of RNA bonds while observing the distinctness of the DNA fragments.

Hence, we start by generating all 1-digit integers in base 4 (there are $u_1 = 4$ of them), then all two-digit integers in base 4 (there are $u_2 = 4^2 = 16$ of them) and finally, we will fill in with 3-digit integers in base 4.

Linking the $u_1 + u_2 = 20$ 1- and 2-digit pairs requires $u_1 + u_2 - 1 = 19$ RNA nucleobase pairs. Hence, all in all we are already at a molecule comprising:

$$u_1 + 2u_2 + u_1 + u_2 - 1 = 4 + 2 \times 16 + 4 + 16 - 1 = 55 \text{ pairs}$$

To proceed, the 45 remaining pairs, permitted by the current technological synthesis capacity, must be constructed using 3-digit integers linked with 1 RNA bond.

We can hence solve $45 \geq 3u_3 + u_3$ to get $u_3 = 11$ (we need one RNA bond for each 3-digit fragment and one RNA to bind to the rest of the string).

The security level of the resulting scheme is $\log_2((u_1+u_2+u_3)!) = \log_2 31! \simeq 113$ bits.

Note that it is possible to construct new types of DNA molecules artificially, see [32], where two new types have been constructed. Assuming that we have 6 different types of DNA molecules at hand, the analysis above can be repeated. First, we have 6 1-digit integers in base 6. We now have 6 1-digit integers in base 6 and 36 2-digit integers in base 6. We first choose $u'_1 = 6$ and solve $100 - (u'_1 + u'_1 - 1) = 89$ molecules left. We then solve $89 \geq 2u'_2 + u'_2$ to get $u'_2 = 29$. In this case, we don't need 3-integers. In total we now have 34 RNA bonds, $\log_2((u'_1 + u'_2)!) = \log_2 35! \simeq 133$ bits.

5.4 Controlling the Information Lifetime

In order to understand and control the lifetime of the information embedded in the DNA/RNA molecules, we introduce a probabilistic model and mathematically determine the bounds on the information lifetime.

5.4.1 Probabilistic Model

Recall that k denotes the number of RNA nucleotides in each of the n identical copies of the initial molecule w. Denote by $L_{i,j}$ the random variable giving the degradation time of the j^{th} RNA nucleotide of the i^{th} copy. Our main assumption is that the $L_{i,j}$, for all i, j, are independent and identically distributed random variables following the exponential distribution of parameter λ. Denote by T_j the random variable representing the time for the cut at the j^{th} position to appear, and by t_x the random variable giving the x^{th} cut time to appear. t_x is the x^{th} order statistic of $(T_j)_{1\leq j\leq k}$, i.e. the x^{th} smallest element of $\{T_1, \cdots T_k\}$. By definition, $T_j = \max\limits_{1\leq i\leq n} L_{i,j}$ and in compactified notation, $t_x = T_{(x)}$.

5.4.2 The Information Lifetime Bounds

We consider that the information stored in the NA molecule goes through three different periods that we call: *life*, *agony* and *death*. In this section, we describe each period separately and give the mathematical model allowing us to determine the bounds of each one of them. Figure 5.9 shows the different periods represented on a time axis:

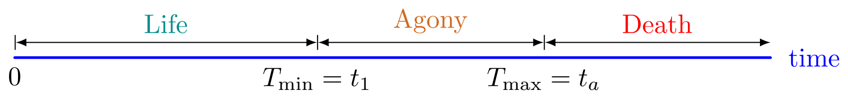


Figure 5.9: The information lifespan

Life:

The information embedded in the NA molecule is fully accessible during the first phase. This happens when no cut has occurred, i.e. for $t \in [0, t_1[$. Let $T_{\min} = \min\limits_{j \leq k} T_j$ be the random variable giving the time at which the first cut to occur. We have $t_1 = T_{min}$ and :

$$\mathbb{P}(T_{\min} > t) = (1 - (1 - \exp(-\lambda t))^n)^k \tag{5.2}$$

Agony:

Agony starts after the first cut has appeared. We can only recover the information by at least brute-force guessing. For each guess, the probability p that a guess gives the correct secret is equal to $\frac{1}{(x+1)!}$, where x is the number of cuts at the time t. Agony ends when the a^{th} cut appears, i.e. we have $t \in [t_1, t_a]$.

Death:

After the a^{th} cut, we consider the information to be *dead* – it is no longer feasible to brute-force a recovery of the information. We have $t_a = T_{\max}$ and:

$$\mathbb{P}(T_{\max} \leq t) = 1 - \sum_{i=0}^{a-1} \binom{k}{i} p(t)^i (1 - p(t))^{k-i} \tag{5.3}$$

with $p(t) = (1 - \exp(-\lambda t))^n$. In this case $t \in]t_a, \infty[$ and it is computationally infeasible within the chosen security parameter to recover the secret information.

Proof The derivations of Equation (5.2) and Equation (5.3) are given below:

For time t, the probability that the information is still accessible at this moment is given by $\mathbb{P}(T_{\min} > t)$ and we have:

$$\begin{aligned}\mathbb{P}(T_{\min} > t) &= \mathbb{P}(\forall j \leq k,\ T_j > t) \\ &= \prod_{j=1}^{k} \mathbb{P}(T_j > t) \\ &= \mathbb{P}(T_1 > t)^k \\ &= (1 - \mathbb{P}(T_1 \leq t))^k \\ &= (1 - \mathbb{P}(L_{1,1} \leq t)^n)^k \\ &= (1 - (1 - \exp(-\lambda t))^n)^k\end{aligned}$$

and this is true by definitions of T_i and $L_{i,j}$, and by independence and uniform distribution of $\{T_i\}_{i \leq k}$ and $\{L_{i,j}\}_{i \leq n, j \leq k}$.

For time t, the probability that the information is completely destroyed after this time is given by:

$$\mathbb{P}(T_{\max} < t) \qquad \text{where: } T_{\max} = T_{(a)}$$

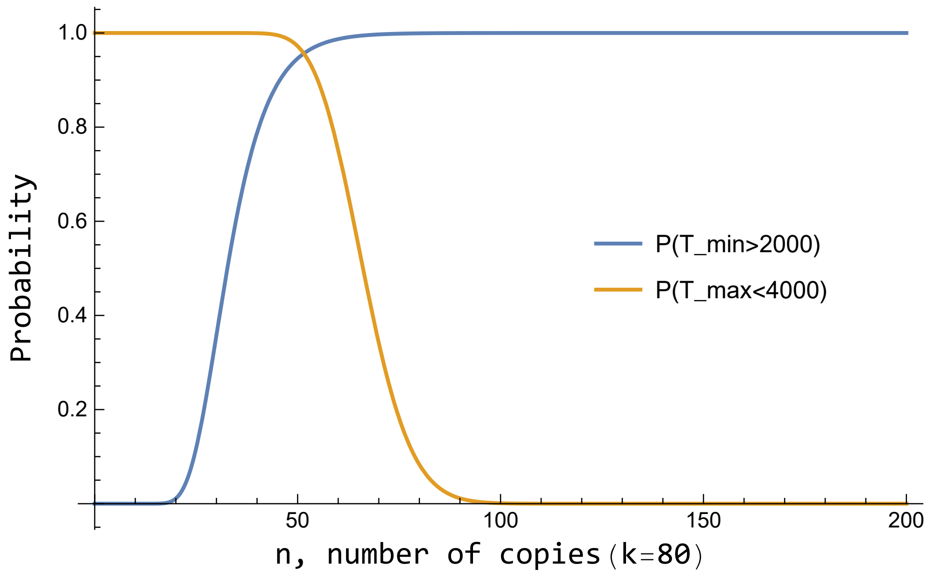
and a is the security parameter. We introduce the random variable

$$Z = \sum_{i=1}^{n} \mathbb{1}(T_i \leq t)$$

and we define $p(t) = \mathbb{P}(T_i \leq t) = (1 - \exp(-\lambda t))^n$. We have then:

$$\begin{aligned}\mathbb{P}(T_{\max} \leq t) &= \mathbb{P}(\#\{i \mid T_i \leq t\} \geq a) \\ &= \mathbb{P}(Z \geq a)\end{aligned}$$

and thus $1 - \mathbb{P}(T_{\max} \leq t) = \sum_{i=0}^{a-1} \binom{k}{i} p(t)^i (1 - p(t))^{k-i}$



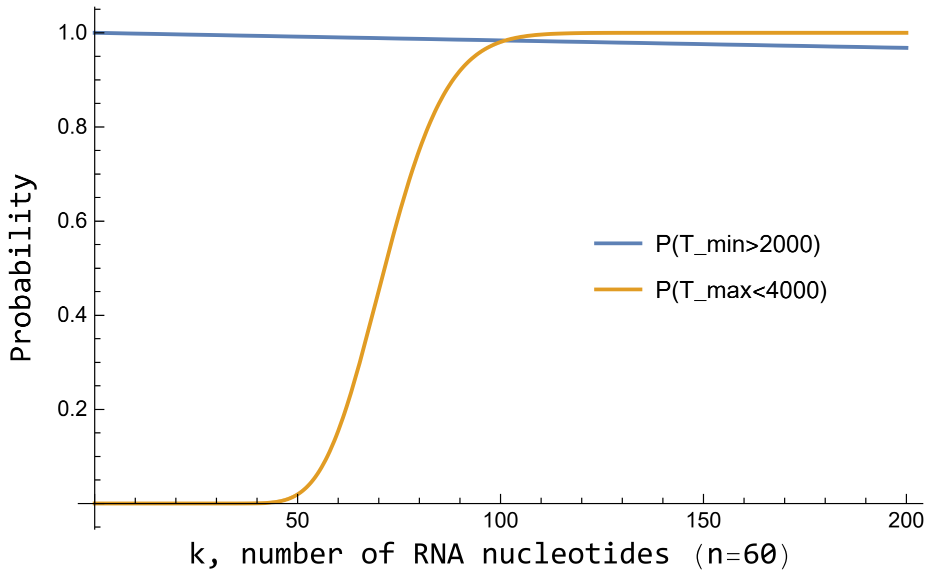


Figure 5.10: Probabilities as functions of the number of copies n, with a fixed number of RNA bonds $k = 80$ (top), and as functions of the number of RNA bonds k with a fixed number of copies $n = 60$ (bottom). Here $a = 24$ and $\lambda = 0.001$ min^{-1}.

We note that $\mathbb{P}(T_{\min} > t)$ is increasing in n and decreasing in k, t and λ, and $\mathbb{P}(T_{\max} \leq t)$ is increasing in k, in $p(t)$ and thus in t and λ, but decreasing in n.

Lemma 10 (Evolution of the number of cuts over time) *Let $C(t)$ denote the number of cuts at the time t. C is a random variable and we have:*

$$\mathbb{E}[C(t)] = k \times (1 - \exp(-\lambda t))^n$$

Proof The number of cuts as a function of the time is a random process, which we will denote by $C(t)$. We can get the expected number of cuts at time t as follows:

$$\begin{aligned}\mathbb{E}[C(t)] &= \sum_{i=1}^{k} \mathbb{E}[\mathbb{1}_{T_i \leq t}] \\ &= \sum_{i=1}^{k} \prod_{j=1}^{n} \mathbb{P}(L_{i,j} \leq t) \\ &= k \times \mathbb{P}(L_{1,1} \leq t)^n \\ &= k \times (1 - \exp(-\lambda t))^n\end{aligned}$$

which is true using independence and uniform distribution of random variables $\{L_{i,j}\}_{j \leq n, i \leq k}$.

The following Lemma allows us to calculate the expected life spans and their variance.

Lemma 11 (The average time and the variance for the x^{th} cut to appear) *The average time when the x^{th} cut appears, $\mathbb{E}(t_x)$, and the corresponding variance, $V(t_x)$, are given by the following formulas:*

$$\mathbb{E}(t_x) = \frac{1}{\lambda} \sum_{s=1}^{kn} C_x(k, n, s)$$

and

$$\mathbb{V}(t_x) = \frac{1}{\lambda^2} \left[2 \sum_{s=1}^{kn} \frac{C_x(k, n, s)}{s} - \left(\sum_{s=1}^{kn} C_x(k, n, s) \right)^2 \right]$$

where:

$$C_x(k, n, s) = \frac{(-1)^{s+1}}{s} \sum_{m=x}^{k} \sum_{p=0}^{s} \sum_{i=0}^{k-m} (-1)^i \binom{k}{m} \binom{mn}{p} \binom{k-m}{i} \binom{ni}{s-p}$$

Proof This proof is done by calculating three different elements: the cumulative distribution function, the density function and the expected value of t_x:

- Cumulative distribution function of t_x:

$$
\begin{aligned}
F_{t_x}(t) &= \mathbb{P}(\exists i_1, \dots, i_x \in [1,k] : T_{i_1}, \dots, T_{i_x} \leq t) \\
&= \sum_{m=x}^{k} \binom{k}{m} \mathbb{P}(T_1 \leq t, \dots, T_m \leq t, T_{m+1} > t, \dots T_k > t) \\
&= \sum_{m=x}^{k} \binom{k}{m} \mathbb{P}(T_1 \leq t)^m \cdot \mathbb{P}(T_1 > t)^{k-m} \\
&= \sum_{m=x}^{k} \binom{k}{m} \mathbb{P}(T_1 \leq t)^m \cdot (1 - \mathbb{P}(T_1 \leq t))^{k-m} \\
&= \sum_{m=x}^{k} \binom{k}{m} (1 - e^{-\lambda t})^{m \cdot n} \cdot (1 - (1 - e^{-\lambda t})^n)^{k-m} \\
&= \sum_{m=x}^{k} \binom{k}{m} \cdot \left(\sum_{p=0}^{m \cdot n} \binom{m \cdot n}{p} (-1)^p e^{-\lambda p t} \right) \cdot \\
&\left(\sum_{a=0}^{k-m} \binom{k-m}{a} (-1)^a \cdot \left(\sum_{b=0}^{n \cdot a} \binom{n \cdot a}{b} (-1)^b e^{-\lambda b t} \right) \right) \\
&= \sum_{m=x}^{k} \sum_{p=0}^{m \cdot n} \sum_{a=0}^{k-m} \sum_{b=0}^{n \cdot a} \binom{k}{m} \binom{m \cdot n}{p} \binom{k-m}{a} \binom{n \cdot a}{b} (-1)^{p+a+b} e^{-\lambda (p+b) t} \\
&= \sum_{m=x}^{k} \sum_{p=0}^{m \cdot n} \sum_{a=0}^{k-m} \sum_{s=p}^{n \cdot a + p} \binom{k}{m} \binom{m \cdot n}{p} \binom{k-m}{a} \binom{n \cdot a}{s-p} (-1)^{s+a} e^{-\lambda s t} \\
&= \sum_{s=0}^{kn} \tilde{C}_x(k,n,s) e^{-\lambda s t}
\end{aligned}
$$

This result follows from the independence of T_i, for $i \in [1,k]$, and using the Newton binomial formula three times. Here:

$$
\begin{aligned}
\tilde{C}_x(k,n,s) &= \sum_{m=x}^{k} \sum_{p=0}^{m \cdot n} \sum_{a=0}^{k-m} \sum_{p''=p}^{n \cdot a + p} \binom{k}{m} \binom{m \cdot n}{p} \binom{k-m}{a} \binom{n \cdot a}{b-p} (-1)^{b+a} \delta_{b,s} \\
&= \frac{(-1)^{s+1}}{s} \sum_{m=x}^{k} \sum_{p=0}^{s} \sum_{a=0}^{k-m} (-1)^a \binom{k}{m} \binom{mn}{p} \binom{k-m}{a} \binom{na}{s-p}
\end{aligned}
$$

where $\delta_{s,b}$ is the Kroenecker delta function.

- Density function of t_x:

$$
\begin{aligned}
f_{t_x}(t) = F'_{t_x}(t) &= \sum_{s=0}^{kn} -\lambda s \tilde{C}_x(k,n,s)e^{-\lambda st} \\
&= \sum_{s=1}^{kn} -\lambda s \tilde{C}_x(k,n,s)e^{-\lambda st}
\end{aligned}
$$

where we start from $s = 1$ since the constant term vanishes after differentiation.

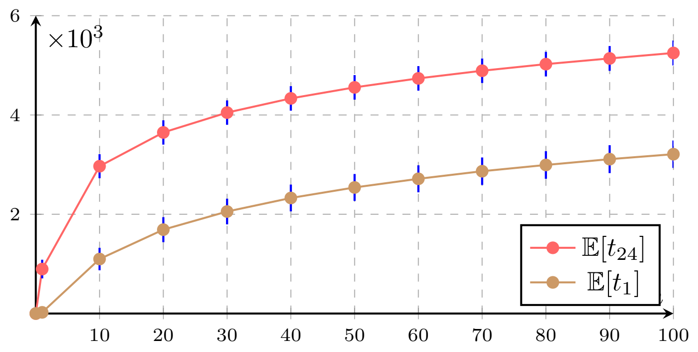
- Expected value of t_x:

$$
\begin{aligned}
E(t_x) &= \int_0^{+\infty} t f_{t_x}(t)dt \\
&= \sum_{s=1}^{kn} -\lambda s \tilde{C}_x(k,n,s) \int_0^{+\infty} te^{-\lambda st}dt \\
&= \frac{1}{\lambda}\sum_{s=1}^{kn} C_x(k,n,s)
\end{aligned}
$$

where: $\int_0^{+\infty} te^{-\lambda st}dt = \dfrac{1}{\lambda^2 s^2}$ and $C_x(k,n,s) = \dfrac{-\tilde{C}_x(k,n,s)}{s}$. This result follows from the fact that we have finite sums.

- Variance of t_x:

$$
\begin{aligned}
V(t_x) &= E(t_x^2) - E(t_x)^2 \\
&= \int_0^{+\infty} t^2 f(t_x)dt - \left(\int_0^{+\infty} tf(t_x)dt\right)^2 \\
&= \sum_{s=1}^{kn} \frac{-2}{\lambda^2 s^2}\tilde{C}_x(k,n,s) - \left(\frac{1}{\lambda}\sum_{s=1}^{nk} C_x(k,n,s)\right)^2 \\
&= \frac{1}{\lambda^2}\left[2\sum_{s=1}^{kn}\frac{C_x(k,n,s)}{s} - \left(\sum_{s=1}^{kn} C_x(k,n,s)\right)^2\right]
\end{aligned}
$$



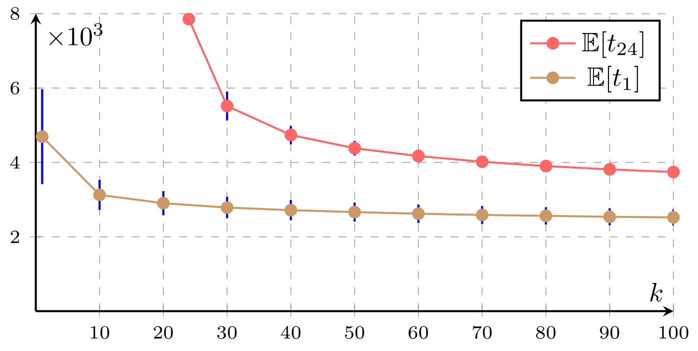


Figure 5.11: $\mathbb{E}[t_{24}]$ and $\mathbb{E}[t_1]$ as functions of n for $k = 40$ (left) and as functions of k for $n = 60$ (right).

$\mathbb{E}(t_x)$ is increasing in n under fixed k and it is decreasing in k under fixed n.

5.4.3 Finding (n, k) for Target Times t and t'

For specific t and t' values, we want our data to still be accessible up to t and completely destroyed after t'; what is the (n, k) pair to consider? To answer this question, n and k should satisfy the following criteria:

$$\mathbb{P}(T_{\min} > t) \simeq 1 \text{ and } \mathbb{P}(T_{\max} \leq t') \simeq 1$$

To this end, we fix a tolerance level, Δ, and require:

$$\mathbb{P}(T_{\min} > t) \geq 1 - \epsilon_\Delta \text{ and } \mathbb{P}(T_{\max} \leq t') \geq 1 - \epsilon_\Delta$$

with $\epsilon_\Delta = \Delta \cdot 10^{-2}$.

Figure 5.12 shows that a solution exists at the intersection point of the two curves. For the example in Figure 5.12, the solution for $t = 3000$ min and $t' = 5000$ min is: $(n, k) = (150, 86)$. This means that if we manufacture 150 NA copies containing 86 RNA nucleotides in each, there is a chance of 96% that

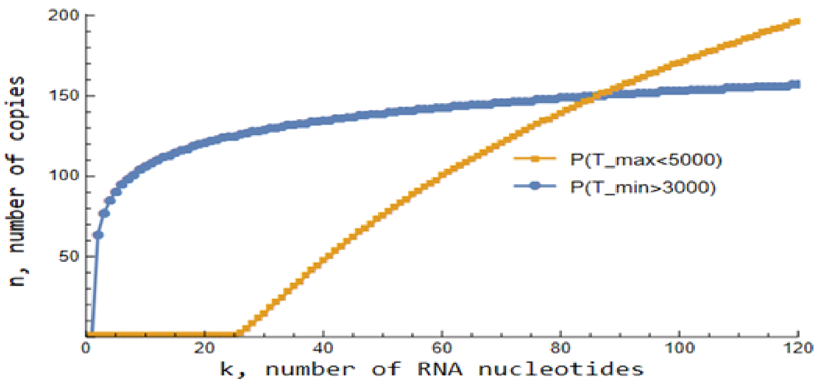


Figure 5.12: Domains bounds of existing solutions satisfying $\mathbb{P}(T_{\min} > 3000) \geq 96\%$ (blue indicates lower bound) and $\mathbb{P}(T_{\max} \leq 5000) \geq 96\%$ (orange indicates upper bound). Here $\Delta = 4$ and $\lambda = 0.001$ min^{-1}. The time is in min.

the secret is still accessible before 3000 minutes and completely destroyed after 5000 minutes. Other solutions for other target t and t' are given in Table 5.4.

$t \setminus t'$	2000	3000	4000	5000
1000	$(22, 819)$	$(17, 74)$	$(15, 38)$	$(15, 30)$
2000	-	$(71, 1243)$	$(53, 83)$	$(48, 40)$
3000	-	-	$(206, 1492)$	$(150, 86)$
4000	-	-	-	$(572, 1584)$

Table 5.4: (n, k) solutions for different values of target t and t' (in minutes). Here $\lambda = 0.001$ min^{-1} and $\Delta = 4$.

The results of Figure 5.12 and Table 5.4 were obtained after running a search code in Python available from the authors. Note that these (n, k) values represent the solutions having the lowest cost in terms of n and k.

5.4.4 Finding (n, k) with Lowest Agony Ratio

What if we want that the data stored in the NA molecule to be fully accessible before some time t and then gets quickly destroyed after some time t' ? Depending on a specified risk level, expressed through α, we want that the time from $\mathbb{E}(t_1) - \alpha\delta_{t_1}$, where we are confident to have the information fully available, until time $\mathbb{E}(t_a) + \alpha\delta_{t_a}$, where we are confident that it is destroyed, is as short as possible. Here δ is the standard deviation. We thus define the *agony ratio* as:

$$f(n,k) = \frac{\mathbb{E}(t_a) + \alpha\delta_{t_a}}{\mathbb{E}(t_1) - \alpha\delta_{t_1}} \ ,$$

and aim to find the (n, k) pair giving the smallest ratio, i.e. being as close to one as possible.

Note that the agony ratio f does not depend on λ. This is particularly useful if we can adjust the fluid's chemical properties to determine the actual life span, refer to (table in Table 5.2 in Section 5.2.4 to have an idea about the order of magnitude of the time for different values of λ.

We expect, at least for large k and n, that the probabilities $p_1 = \mathbb{P}(\mathbb{E}(t_1) - \alpha\delta_{t_1} < t_1)$ and $p_2 = \mathbb{P}(\mathbb{E}(t_a) + \alpha\delta_{t_a} > t_a)$ are close to the ones derived from a normal distribution. This is confirmed by Table 5.5, which gives numerical values of p_1 and p_2 for the cases $\alpha = 1$ and $\alpha = 2$.

$n \setminus k$	120	160	200	240	280
120	$(0.84, 0.84)$	$(0.84, 0.84)$	$(0.84, 0.82)$	$(0.83, 0.82)$	$(0.84, 0.84)$
160	$(0.83, 0.83)$	$(0.82, 0.83)$	$(0.82, 0.84)$	$(0.84, 0.84)$	$(0.85, 0.86)$
200	$(0.84, 0.84)$	$(0.85, 0.86)$	$(0.85, 0.84)$	$(0.82, 0.84)$	$(0.83, 0.82)$
240	$(0.85, 0.86)$	$(0.84, 0.83)$	$(0.85, 0.82)$	$(0.83, 0.84)$	$(0.84, 0.83)$
280	$(0.84, 0.83)$	$(0.83, 0.84)$	$(0.84, 0.84)$	$(0.84, 0.83)$	$(0.84, 0.84)$

$n \setminus k$	120	160	200	240	280
120	$(0.97, 0.97)$	$(0.97, 0.97)$	$(0.96, 0.97)$	$(0.96, 0.97)$	$(0.96, 0.97)$
160	$(0.96, 0.97)$	$(0.96, 0.97)$	$(0.96, 0.97)$	$(0.96, 0.97)$	$(0.95, 0.97)$
200	$(0.96, 0.97)$	$(0.97, 0.98)$	$(0.96, 0.97)$	$(0.95, 0.97)$	$(0.96, 0.97)$
240	$(0.97, 0.98)$	$(0.96, 0.97)$	$(0.97, 0.97)$	$(0.96, 0.97)$	$(0.96, 0.97)$
280	$(0.96, 0.97)$	$(0.96, 0.97)$	$(0.96, 0.97)$	$(0.96, 0.97)$	$(0.96, 0.97)$

Table 5.5: (p_1, p_2) probabilities. Top $\alpha = 1$ and bottom $\alpha = 2$. Here $\lambda = 0.001$.

Table 5.6 shows that we effectively have lower agony ratios when n and k are significant; the best (n, k) pair is then the one with the largest values of n and k. This suggests going further with more significant values of n and k, when the resources allow it, and when actual acceptable timings can be found depending on λ.

$n \setminus k$	120	160	200	240	280
120	1.50	1.46	1.44	1.42	1.41
160	1.46	1.42	1.40	1.38	1.37
200	1.43	1.40	1.37	1.36	1.34
240	1.41	1.38	1.35	1.34	1.33
280	1.40	1.37	1.34	1.33	1.31

$n \setminus k$	120	160	200	240	280
120	1.67	1.62	1.58	1.56	1.54
160	1.60	1.56	1.53	1.50	1.49
200	1.56	1.52	1.49	1.47	1.45
240	1.53	1.49	1.46	1.44	1.43
280	1.52	1.48	1.44	1.43	1.41

Table 5.6: $f(n, k)$ values for different values of n and k. Left $\alpha = 1$ and right $\alpha = 2$

As an example, we take $(n, k) = (280, 280)$ and we give, in Table 5.7, numerical values for $(t, t') = (\mathbb{E}(t_1) - 2\delta_{t_1}, \mathbb{E}(t_a) + 2\delta_{t_a})$ for different values of λ and $\alpha = 2$. In this case, $f(280, 280) \simeq 1.41$ and $(p_1, p_2) \simeq (0.96, 0.97)$. Remember that getting other values for (t, t') requires choosing other values for λ and hence adjusting the chemical properties of the fluid accordingly.

λ(in mins^{-1})	10^{-3}	$4.5 \cdot 10^{-4}$	$4.06 \cdot 10^{-5}$	$3.3 \cdot 10^{-6}$	$1.22 \cdot 10^{-7}$
(t, t')	$(57.5, 81.5)$ hours	$(5.3, 7.5)$ days	$(2, 2.8)$ months	$(2, 2.8)$ years	$(53.3, 75.7)$ years
(p_1, p_2)	$(0.96, 0.97)$	$(0.96, 0.97)$	$(0.96, 0.97)$	$(0.96, 0.97)$	$(0.96, 0.97)$
agony ratio	1.41	1.41	1.41	1.41	1.41

Table 5.7: The best (t, t') solutions in terms of the lowest agony ratio for different values of λ. Here $(n, k) = (280, 280)$, $\alpha = 2$ and $a = 24$.

5.4.5 Finding (n, k) for Target Time t_{target} with the Least Variance

For a target time t_{target} we want that the secret data is inaccessible after t_{target}, what is the best (n, k) to consider? To answer this question n and k should satisfy the following approximation:

$$\mathbb{E}(n, k, t_a) - t_{target} \simeq 0$$

We are, therefore, looking for (n, k) pairs minimizing the distance between $\mathbb{E}(t_a)$ and t_{target}. However, we also want to be as confident as possible that this is the time that the information is destroyed. Hence, we would prefer the (n, k) pair for which $\mathbb{E}(t_a)$ has the least variance. Figure 5.13 represents the optimal (n, k) solutions verifying $\mathbb{E}(n, k, t_a) \simeq 2000$ min. We see that we have the least variance when n and k are large. Table 5.8 gives the corresponding k for each n.

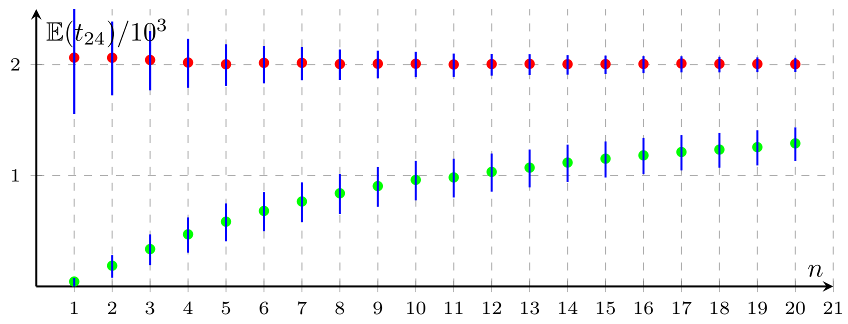


Figure 5.13: Optimal Solutions for $t_{\text{target}} = 2000$ *mins*. The blue lines represent the standard deviation, the red points represent the expected values $\mathbb{E}(t_{24})$, and the green ones represent $\mathbb{E}(t_1)$. Here $\lambda = 0.001$ min^{-1}.

As seen in the last section, when n and k are significant, we have a lower agony ratio as well.

Search Algorithm:

Our search for finding the optimal solutions consists of finding the optimal k for each n, which we call k_n. The pair (n, k_n) ensures: $t_{target} \simeq \mathbb{E}(n, k_n, t_a)$. We can use the monotonicity properties of $\mathbb{E}(n, k_n, t_a)$ to proceed efficiently as follows:

- Initialisation: Start with $n = 1$ and $k = a$ and compute $\mathbb{E}(n, k, t_a)$. For fixed n, $\mathbb{E}(t_a)$ is decreasing with k, and for fixed k, $\mathbb{E}(t_a)$ is increasing with n. Hence if $\mathbb{E}(t_a) < t_{target}$, increase n until $\mathbb{E}(t_a) > t_{target}$.
- Increasing k: We can now increase k until we find $\mathbb{E}(n, k, t_a) \simeq t_{target}$, and we can take $k_n = k$. Since the expectation value is monotonically decreasing with k, this is easily determined and can be accelerated via the bisection method. For an integer Δ, consider the interval $I_\Delta = [k, k + \Delta]$ and compute $c = g(n, k) \cdot g(n, k + \Delta)$, where $g(n, k) = \mathbb{E}(n, k, t_a) - t_{target}$. If $c < 0$, $k_{n+1} \in I_\Delta$. In this case, bisect I_Δ and repeat the same operation. If not, $k_n > k + \Delta$, and we can choose a new interval from $k + \Delta$.
- Increasing n: Bearing in mind that $t_{target} \simeq \mathbb{E}(n, k_n, t_a) \leq \mathbb{E}(n+1, k_n, t_a)$, k_{n+1} is either k_n or bigger. Hence, in the next iteration for n, we initialise k to k_n and proceed using the last step.
- Finally, the optimal n, k_n value is chosen to minimise the variance of $\mathbb{E}(n, k, t_a)$.

Remark 13 We can get a wide range of values of λ if we can adjust the chemical parameters such as temperature, PH and the concentration of particular ions. Moreover, since the $\mathbb{E}(n, k, t_a)$ and its standard deviation are inversely proportional to λ, this can help us to find even better solutions.

Note that we got all of our numerical values using a Python simulation since it is faster than working with the theoretical formula of $\mathbb{E}(t_a)$ directly. Table 5.8 illustrates our findings of optimal k of n for $t_{target} = 2000$ min.

n	1	2	3	4	5	6	7	8	9	10
Optimal k	27	31	36	42	49	57	65	76	87	101

n	11	12	13	14	15	16	17	18	19	20
Optimal k	118	136	156	182	210	243	280	324	374	435

Table 5.8: Optimal k for n verifying $E(n, k, t_a) - t_{target} \simeq 0$. $\lambda = 0.001$ and $a = 24$.

5.5 Conclusion and Discussion

In this chapter, we presented a new method for encoding and storing information using synthetic DNA and RNA. We showed that our method allows having information with a foreseeable lifespan. Moreover, we analyzed its security and discussed parameter choice and efficiency. Finally, we proposed three different algorithms on how to tune the information lifetime.

Other media supports such as bisphenol A thermal fax paper or unstable nonvolatile memory technologies can be used to implement the same principle. Still, the decay models of each of those phenomena should be re-computed, and the formulae given in this paper adapted. For instance, in the case of thermal paper, for instance, the number of copies can be replaced by pixel size.

Chapter 6

Masked Ballots

This work has been done in collaboration with Peter Y.A. Ryan, Peter Roenne, Najmeh Soroush and Philip B. Stark and was published in E-Vote-ID 2021; and it is part of the dissertation of Najmeh Soroush too. We reproduce in this chapter a part of the paper [48].

Contents

Summary

We consider elections that publish anonymised voted ballots or anonymised cast-vote records for transparency or verification purposes, investigating the implications for privacy, coercion, and vote-selling and exploring how partially masking the ballots can alleviate these issues.

Risk Limiting Tallies (RLT), which reveal only a random sample of ballots, were previously proposed to mitigate some coercion threats. Masking some ballots provides coerced voters with plausible deniability, while risk-limiting techniques ensure that the required confidence level in the election result is achieved.

Here we show how these ideas can be generalised and made more flexible and effective by masking at a finer level of granularity: at the level of ballots components. In particular, we consider elections involving complex ballots, where RLT may be vulnerable to pattern-based vote buying. We propose various measures of verifiability and coercion-resistance and investigate how several sampling/masking strategies perform against these measures.

We also define new quantitative measures for the level of coercion-resistance without plausible deniability and the level of vote-buying-resistance without "free lunch" vote sellers.

These results and the different strategies for masking ballots are of general interest for elections that publish ballots for auditing, verification, or transparency purposes.

6.1 Introduction

Some voting systems, including many end-to-end verifiable systems and some conventional elections, publish the (plaintext) ballots. This is typically quite safe if these ballots are suitably anonymised, by for example, verifiable mixes published on a bulletin board. But in some contexts, revealing such information may be problematic: certain corner cases, such as unanimous votes or absence of any votes for a candidate and coercion threats, such as signature attacks.

In [25], the idea of Risk-Limiting Tallies (RLT) and Risk-Limiting Verification (RLV) was proposed to mitigate such threats. The idea is to shroud a proportion of the (anonymised) votes so voters can plausibly claim to have complied with the coercer, even though no votes appear for the candidate demanded by the coercer or no ballot with the pattern demanded by the coercer shows up in the tally. The proportion left shrouded can be adjusted using risk-limiting techniques to ensure that the confidence in the announced outcome achieves the required threshold, e.g., 99%.

In this work, we note that despite the pleasing features of the constructions of [25], there are still some drawbacks, in particular if the ballots are rather complex. While RLT may disincentivize *coercion*, there may still be an incentive for *vote buying*: the voter might still cast the required pattern vote in the hope that it will be revealed. Further, it has been suggested that RLT is arguably undemocratic in that some voters' ballots do not contribute to the final tally.

The second objection can be countered by arguing that every vote has an equal probability of being included in the count and that the outcome will be, with whatever confidence level required, a correct reflection of all votes cast. Nonetheless, it is an aspect that some people find troubling.

A pleasing side effect of our construction is that all ballots are treated equally.

These observations suggest exploring different ways to apply RLT

when ballots are complex: rather than shrouding entire ballots at random, we shroud, at random, some preferences on each ballot. In effect, we are filtering the tally horizontally rather than vertically. This hits both of the issues above: the chance any given pattern remains identifiable after the filtering is reduced, and every ballot contributes to the outcome, albeit not necessarily to every contest. In the *full tally* construction below, every ballot contributes fully to the announced outcome, but we shroud the link between the tracker and some components of the ballots. For tracker-based schemes, the voters can verify some but not all of their selections. This paper seeks to quantify these effects and explore trade-offs among them.

This ballot-masking method and its privacy implications are interesting not only for RLT and RLV but for all schemes where all or some ballots are published for auditing, verification, or transparency. For example, Colorado is currently redacting cast-vote records (CVRs) by removing entire CVRs, e.g., for rare ballot styles; partial masking has been considered as an alternative. We note, however, that masking parts of the ballot might make it hard to detect ill-formed, e.g., over-votes etc.

We also note that this idea has similarities to the SOBA constructions for

Risk-Limiting-Audits (RLAs), [8], which also publishes each audited ballot "dis-assembled" into different contests, whereas the auditors will see the intact ballot. The VAULT approach [7] also uses homomorphic encryption of the cast-vote records to achieve the SOBA goals more easily. (VAULT was used for the first time in a risk-limiting audit in Inyo County, California, in 2020.)

The purpose and underlying cryptographic constructions are quite different, but our analysis applies to these cases as well.

For some tally algorithms, we can separate ballots into their atomic parts and reveal them independently after anonymising them, which effectively counters signature attacks. However, that reduces public transparency and may reduce public confidence in the election result.

For Selene, where voters verify their votes via trackers, this separation provides a method to verify without revealing individual ballots: we simply assign a distinct tracker to each ballot element. Voters can then verify some or all components of their ballot using those trackers. A coerced voter could use the Selene tracker-faking mechanism to assemble a ballot that matches the coercer's instructions. Technically this is straightforward but from a usability standpoint seems problematic. Moreover, even if the voter was prepared to go the effort of concocting such a fake ballot, the necessary ingredients might not be available, so coercion threats will remain, and the probability that one of atomic trackers is the same as the coercer's increases. Thus it makes sense to look for alternatives.

6.2 Masking Complex Ballots

Many elections use simple plurality voting: the voter selects at most one candidate from a set, in the simplest case, a referendum, a choice between "yes" and "no." The next level of complexity is single-winner plurality, aka "first past the post." More complex social choice functions and correspondingly more complex ballots are common. Perhaps the next level in complexity is *approval voting* in which the voter can cast votes for several candidates for a single office, and multi-winner plurality, in which a voter can vote for up to k candidates for k offices. In some cases, voters may have a quota of votes and is allowed to cast more than one vote for a given candidate, up to some limit. Finally, some methods allow voters to give a preference ranking to the candidates.

Common to all of these social choice functions, if the ballots are published, is that they are vulnerable to signature attacks (also known as "Italian" attacks), i.e. a coercer chooses a particular, unlikely pattern, instructs the victim to mark a ballot with that pattern and checks whether a ballot with that pattern appears in the tally.

Let us assume that the ballots are of the form $(v_1, v_2, \ldots, v_k)$ with k the number of candidates and v_i taking values from a specified set $\mathcal{V}$. $\mathcal{V}$ might for example just be $\{0, 1\}$ or a set of integers plus a blank: $\{1, ..., s\} \bigcup \{\text{blank}\}$ etc.

In many types of elections, these ballot-level selections, or subsets thereof, will reappear as part of the tally procedure (e.g. in electronic mixnet tallies), as part of an audit trail or for transparency (electronic scans of paper ballots), in Risk-Limiting Audits using samples of votes, or verification procedures (e.g. in tracker-based schemes such as Selene). In order to preserve privacy, the mapping between the published votes and the voter is normally anonymised.

As mentioned above, revealing these ballots may endanger the receipt-freeness of the election. With *Masked Tallies*, introduced here, only parts of each ballot are revealed:

$$(\,\mathsf{msk}_{i1}(v_1^{(i)}), \mathsf{msk}_{i2}(v_2^{(i)}),\,\dots\,, \mathsf{msk}_{ik}(v_k^{(i)})\,) \quad \text{for } i = 1, \dots, n.$$

The functions msk_{ij} are either the identity, displaying the component of the vote, or a constant, .e.g. $*$ ($\notin \mathcal{V}$), masking the component. n is the number of ballots cast.

Risk-Limiting Tallies [25], involved unmasking only as many randomly selected ballots as are needed to determine the election result with a chosen risk limit. The remaining ballots were kept completely masked. Here we suggest a generalization, allowing partial masking of the ballots, and we will discuss the impact on risk limits, privacy, coercion-resistance, and resistance to vote-buying.

6.3 Partially Masked RLTs and RLVs

We reprise risk-limiting tallies and verification, RLT and RLV [25], before extending these to general masks. First we recapitulate the idea of tracker-based verification in terms of Selene.

Outline of Selene

Selene [47] enables verification by posting the votes in the clear on the BB along with private tracking numbers. Voters are only notified of their tracker some time after the vote/tracker pairs have been publicly posted, giving a coerced voter the opportunity to choose an alternative tracker to placate the coercer. The voter is able to fake the tracker and related cryptographic data using a secret trapdoor key. The notification of the trackers is carefully designed to provide assurance to the voter that it is their correctly assigned tracker, i.e. unique to them, while being deniable to any third party.

Assuming that votes are encrypted component-wise, at the end of the mixing we will have encrypted votes and trackers on the bulletin board:

$$(\{tr_i\}_{PK}, (\{v_1^{(i)}\}_{PK}, \{v_2^{(i)}\}_{PK}, ...\{v_k^{(i)}\}_{PK}))$$

where $\{\cdot\}_{PK}$ denotes encryption under the public key PK. These ballots can now be verifiably decrypted to reveal the vote/tracker pairs that can be checked by the voters, and anyone can compute the tally directly on the plaintext votes.

Risk-Limiting Tallies and Verification with Partially Masked Ballots

In the original approach to RLT (where ballots are without trackers) and RLV (with trackers for individual verification), see [25], the idea was to only decrypt a random subset of the ballots. The number decrypted being controlled by a risk-limit that bounds the probability that the announced election result will be wrong.

In the new masked RLV and RLT approach, we instead reveal randomly selected components of the ballots (and the trackers for RLV). If there is more than one contest on the ballot, the contests can be treated independently. How much we reveal will again be governed by a specified risk limit, as in [25]. A natural choice is to first decrypt m of the k entries in each ballot at random, and to increase m if necessary to meet the risk limit. This is simplest and will be used in the analysis below. In practice, it may make sense to dynamically change the rate of openings per candidate, e.g. if a candidate is popular we might be able to decrease the rate of unmasking of votes for that candidate, maintaining the risk limit while improving coercion-resistance.

Using this masked approach for RLV with tracker verification, the masking means that only parts of the ballot can be verified, but unlike to the original RLV every voter can verify *something*. We will quantify how much.

6.4 Quantitative Privacy-Type Properties

We now want to measure and compare privacy-properties for different masked tally methods. When computing concrete values, we will consider approval voting with k candidates only 0 or 1 is allowed for each candidate, without any overall constraint, $(v_1, \ldots, v_k) \in \{0,1\}^k$. For the n honest voters, we assume for simplicity that the probability to vote $v_i = 1$ is p_i and these probabilities are independent. As a special concrete case we consider a student election with $n = 1001$ voters (one voter is under observation), $k = 5$ candidates with probabilities $(0.6, 0.4, 0.01, 0.01, 0.01)$, i.e. two popular candidates and three unpopular.

6.4.1 Privacy

In order to compare the different approaches, we first consider the quantitative δ-privacy definition from [36]. The parties are an observer O, who can use public data, n_h honest voters and an additional voter under observation V_{obs}, whose vote the observer tries to guess.

Definition 12 (δ-privacy) Let P be a voting protocol and V_{obs} be the voter under observation. We say that P achieves δ-privacy if

$$\Pr[(\pi_O || \pi_{V_{obs}}(v_0^O) || \pi_v)^{(l)} \to 1] - \Pr[(\pi_O || \pi_{V_{obs}}(v_1^O) || \pi_v)^{(l)} \to 1]$$

is δ-bounded as a function of the security parameter ℓ for all vote choices v_0^O and v_1^O of the observed voter. Here π_O, $\pi_{V_{obs}}$ and π_v are respectively the programs run by the observer O, the voter under observation V_{obs} and all the honest voters.

The value δ will depend on the chosen vote distribution, and we see that it is especially relevant to penalize signature attacks: if we assume that there is a vote choice $v^* = (v_1^*, \ldots, v_k^*)$ which rarely gets selected and has a probability close to zero, then an unmasked tally which reveals all cast plaintext ballots, even in anonymised form, will have $\delta = 1$ – the adversary simply checks if v^* appears.

Full ballot disclosure When we reveal all ballots, we can consider the case where the observer tries to distinguish a voter casting the most unpopular vote vs the most popular vote, as in a signature attack. That is, in the definition we let $v_0^O = (v_1, \ldots, v_k)$ with $v_i = 1$ if $p_i \leq 1/2$ and $v_i = 0$ if $p_i > 1/2$, and we have $v_1^O = (1 - v_1, \ldots, 1 - v_k)$. Denote the corresponding probability p_{min}. Now a good strategy is simply to check if at least one $(v_1, \ldots, v_k)$ appears in the disclosed ballots, and the algorithm then outputs "1". This means $\Pr[(\pi_O || \pi_{V_{obs}}(v_0^O) || \pi_v)^{(l)} \to 1] = 1$ but $(\pi_O || \pi_{V_{obs}}(v_1^O) || \pi_v)$ will also output "1" if another voter chooses v_0^O. This happens with probability $1 - (1 - p_{min})^{n_h}$. We conclude that $\delta \geq (1 - p_{min})^{n_h}$. For the case of the student election we have that $v_0^O = (0, 1, 1, 1, 1)$ with $p_{min} = 0.4^2 \cdot 0.01^3 = 1.6 \cdot 10^{-7}$. Thus for $n_h = 1000$ we have $\delta \geq (1 - p_{min})^{n_h} \approx 0.99984$, i.e. close to 1.

Result Only We now consider the case where we only reveal the overall result $r = (r_1, \ldots, r_k)$. In this case, we can follow an analysis close to [36, 38] for calculating δ. For every possible result r, we calculate the probability that the result happened if the observed voter cast v_0^O or v_1^O. The algorithm will then output one if the former probability is larger. We get:

$$\delta = \sum_{r \in M^*_{v_0^O, v_1^O}} \left(A_r^{v_0^O} - A_r^{v_1^O}\right)$$

where $M^*_{v_0^O, v_1^O} = \{r \in \mathbb{R} : A_r^{v_1^O} \leq A_r^{v_0^O}\}$, $\mathbb{R}$ is the set of all possible results of the election, and A_r^v denotes the probability that the choices of the honest voters yield the result r given that V_{obs}'s choice is v. These probabilities can explicitly be calculated since each candidate count from the honest voters, X_i, is binomially distributed, $X_i \sim BD(n_h, p_i)$. We thus have

$$\begin{aligned} A_r^v &= \mathbb{P}(X_1 = r_1 - v_1) \cdots \mathbb{P}(X_k = r_k - v_k) \\ &= \prod_{i=1}^{k} \binom{n-1}{r_i - v_i} p_i^{r_i - v_i} (1 - p_i)^{n - r_i + v_i - 1} \end{aligned}$$

RLT In the original RLT method, we keep a certain fraction, f_{blind}, of the ballots hidden, that is $(1 - f_{blind})n$ ballots are published. If we consider the optimal algorithm from the full ballot disclosure and the corresponding δ_{full} we see that $\delta = (1 - f_{blind})\delta_{full}$ since the probability that the observed voter's ballot is hidden is $(1 - f_{blind})$.

Masked RLT We now consider the case of masked RLTs where we release all ballots but with only m out of k components unmasked. A good strategy to lower bound δ is to count the number N_b of colliding ballots v, which satisfy $\mathsf{msk}_v v = \mathsf{msk}_v v_b^O$ for $b = 0, 1$. We choose v_0^O as the most unlikely ballot, as above and take v_1^O as the opposite ballot to discriminate optimally between the two counts. The main distinguishing power comes from N_0, and we let the distinguishing algorithm output "1" if the probability of the honest voters casting $N_0 - 1$ colliding votes is higher than getting N_0 collisions. The probability for each honest voter to have a collision is:

$$p_{col} = 1/\binom{k}{m} \cdot \sum_{1 \le i_1 < i_2 < \ldots < i_m \le k} p_{i_1} \ldots p_{i_m}$$

and $N_0 \sim BD(n_h, p)$, where p_i is the probability of a match in the ith candidate.

In Figure 6.1, we have displayed the probabilities for the student election example. The algorithm above will then simply give the probability at the mode of the binomial distribution with p_{col}. For $m = 3$, we find $\delta \ge 0.6$ for the student election.

Probabilities for p_0

$m \setminus p$	p_{col}	$(1 - p_{col})^n$
1	0.1660	$1.4657E - 79$
2	0.0184	$8.3421E - 9$
3	0.0005	0.6039
4	$9.76E - 6$	0.9902
5	$1.6E - 7$	0.9998

Probabilities for p_1

$m \setminus p$	p_{col}	$1 - (1 - p)^n$	$(1 - p)^n$
1	0.8340	~ 1	0
2	0.6864	~ 1	0
3	0.5567	~ 1	0
4	0.4445	~ 1	$4.2335E - 256$
5	0.3493	~ 1	$2.3752E - 187$

Figure 6.1: LHS shows the probability, p, that a single honest voter cast a vote, which after masking equals the mask of $v_0^O = (0, 1, 1, 1, 1)$. RHS: the same for $v_1^O = (1, 0, 0, 0, 0)$.

In [35], the authors present a definition of quantitative coercion-resistance following similar ideas as in Definition 12. We will here use their strategy version and not go into all details. We let S denote the election system with a specified number candidates, honest (n_h) and dishonest voters (mostly neglected here), a ballot distribution, an attacker, C_S, and voter, V_S, interactive Turing machine models. We let γ denote a property defining the goal of the coerced voter, e.g. to vote for a specified candidate.

Definition 13 S achieves δ^{cr}-coercion-resistance if for all dictated coerced strategies $\pi_{V_{co}} \in V_S$ there exists a counter-strategy $\tilde{\pi}_{V_{co}} \in V_S$ s.t. for all coercer programs $\pi_c \in C_S$:

- $\Pr[(\pi_c || \tilde{\pi}_{V_{co}} || \pi_v)^{(l)} \mapsto \gamma]$ is overwhelming.
- $\Pr[(\pi_c || \pi_{V_{co}} || \pi_v)^{(l)} \mapsto 1] - \Pr[(\pi_c || \tilde{\pi}_{V_{co}} || \pi_v)^{(l)} \mapsto 1]$ is δ^{cr}-bounded.

With bounded and overwhelming defined in the security parameter. The first part says that the voter is able to achieve her goal (e.g. vote for a specific candidate) and the second part says that the coercer's distinguishing power is bounded by δ^{cr}. This level of coercion-resistance depends on several parameters, especially the probability distribution on the candidates.

Whereas this definition gives a level of coercion-resistance, it does not tell the full story. To see this, let us consider two different election systems. System A outputs voter names and corresponding votes with probability 1/2, completely breaking privacy; otherwise, it only outputs the election result. Neglecting the information from the election result, we get $\delta^A = 1/2$. In system B, the voter secretly gets a signed receipt of her vote with probability 1/2; otherwise the protocol works ideally. In this case, a coerced voter can always cast her own choice and claim that no receipt was received. A voter following the coercer's instruction will, with probability 1/2, give the corresponding receipt, i.e. we again have $\delta^B = 1/2$. However, the two systems are very different from the point of view of the voter: in system A, the coerced voter gets caught cheating with probability 1/2, whereas in system B, the voter always has plausible deniability.

Since plausible deniability is an essential factor for the usability of coercion-resistance mechanisms, we need a new definition to be able to measure this aspect.

6.4.2 No Deniability

The level of plausibility of a voter claiming to have followed the coercer, while actually following the counter strategy, relates to the probability of false positives when the coercer tries to determine if the voter disregarded the instructions. In the following, we assume, without loss of generality, that the coercer outputs 1 when blaming the voter. We now want to define the maximal probability of getting caught without any deniability, i.e. we consider the case where $\Pr[(\pi_c||\pi_{V_{co}}||\pi_v)^{(l)} \mapsto 1] = 0$ or negligible, i.e. the coercer only uses strategies where he never blames an honest voter.

Definition 14 *S achieves $\delta^{cr,no-d}$-coercion-resistance if for all dictated coerced strategies $\pi_{V_{co}} \in V_S$ there exists a counter-strategy $\tilde{\pi}_{V_{co}} \in V_S$ s.t. for all coercer programs $\pi_c \in C_S$:*

- $\Pr[(\pi_c||\tilde{\pi}_{V_{co}}||\pi_v)^{(l)} \mapsto \gamma]$ *is overwhelming.*
- $\Pr[(\pi_c||\tilde{\pi}_{V_{co}}||\pi_v)^{(l)} \mapsto 1]$ *is $\delta^{cr,no-d}$-bounded and* $\Pr[(\pi_c||\pi_{V_{co}}||\pi_v)^{(l)} \mapsto 1]$ *is negligible.*

Note that the coercer's optimal strategy to obtain this $\delta^{cr,no-d}$ and the voter's strategy might be different from the ones in Definition 13 but $\delta^{cr,no-d} \leq \delta^{cr}$.

The no deniability probability clearly separates the RLT approaches. The original RLT always has plausible deniability if we choose to keep some ratio of ballots shrouded and the voter can claim her ballot was not revealed. This is e.g. important for RLV giving deniability against an attack where the coercer provides a ciphertext to cast and ask for its decrypted vote.

In the case of masked ballots, there can be a chance of getting caught undeniably. This will depend strongly on the number of revealed ballot components m, the vote distribution and the voter's goal. For the student election analysed above, the worst case is when the goal of the voter is to cast $(1,0,0,0,0)$. The coercer's optimal strategy is then to demand a vote for $(0,1,1,1,1)$. The coercer will blame the voter if there is no matching masked ballot, i.e. if no honest voters produce a collision, which happens with probability $(1-p_{col})^{n_h+1}$ computed in Fig. 6.1. The probability of no deniability is then $p = 8 \cdot 10^{-9}$ for $m = 2$ but jumps abruptly to $p = 0.6$ for $m = 3$.

An interesting case is when the voter has a relaxed goal allowing to cast a signature part or not, and when the vote distribution has some ballots strictly zero probability. For example, let us consider a three candidate 0/1 election with 1-vote probabilities $(1/2, 1/2, 0)$. The voter's goal is to cast a 1 for the first candidate. The coercer's optimal strategy is to demand a signature ballot $(0,0,1)$. The voter has two counter-strategies:

1. cast a vote $(1,0,0)$ without the 0 probability signature part
2. casting a vote $(1,0,1)$ with the signature part

For 1) there is no deniability if no other voter casts a matching ballot and the coerced voter's ballot does not match either. For $m = 1$ this happens with $p = (2/3)^{n_h+1}$ and for $m = 2$ with $p = (11/12)^{n_h}$, both are small if we have many voters. For 2) there will always be a matching vote if the first part of the coerced voter's ballot is masked. However, if the last part is revealed, the coercer can deduce this ballot comes from the coerced voter since this candidate had probability 0. If the 1 vote in the first part is revealed, the voter is caught with no deniability.

Thus is no deniability with a probability of $1/3(2/3)^{n_h}$ for $m = 1$ and $1/3 + 1/3(11/12)^{n_h}$ for $m = 2$. Thus for $m = 1$, strategy 2) is always better, but for $m = 2$, strategy 1) is better when we have more than 13 voters. In some cases the voter strategy thus depends on m, which might not be known beforehand.

Finally, it is also natural to define the level of plausability we can provide. The average plausability that a voter has, e.g. in Definition 13, is a useful quantity for the voter. Still, it would be more useful to guarantee that the voter always has a certain level for coercion-resistance. We leave a precise definition for future work.

6.4.3 Receipt-Freeness

Following [35], definition 13 also covers receipt-freeness. However, we again argue that modelling some variants is useful. The following definition is based on a swap of $\pi_{V_{co}}$ and $\pi_{\tilde{V}_{co}}$ in Definition 14, and models vote buyers who do not want to pay a "free lunch" to vote sellers who follow their own goal. The voter goal γ can here be to cast a specified vote or set of votes.

Definition 15 (Weak Vote Buying Resistance) For a given small p_{fl}, S achieves δ^{wvb}-coercion-resistance if for all dictated coerced strategies $\pi_{V_{co}} \in V_S$ there exists a counter-strategy $\tilde{\pi}_{V_{co}} \in V_S$ s.t. for all coercer programs $\pi_c \in C_S$:

- $\Pr[(\pi_c || \tilde{\pi}_{V_{co}} || \pi_v)^{(l)} \mapsto \gamma]$ is overwhelming.
- $\Pr[(\pi_c || \pi_{V_{co}} || \pi_v)^{(l)} \mapsto 1] - \Pr[(\pi_c || \tilde{\pi}_{V_{co}} || \pi_v)^{(l)} \mapsto 1]$ is δ^{wvb}-bounded and $\Pr[(\pi_c || \tilde{\pi}_{V_{co}} || \pi_v)^{(l)} \mapsto 1]$ is p_{fl}-bounded.

Here, we interpret outputting "1" as paying the vote seller. This definition bounds how often an instruction-following vote seller gets paid by a vote-buyer (by $\delta^{wvb} + p_{\text{fl}}$), but under the condition that a voter who casts another vote is only paid with a (very) small probability p_{fl}. This is a weakened vote-buyer model but interesting since a vote buyer should avoid vote sellers going for a "free lunch". If the probability of an honest vote seller getting paid is low, it would help curb vote-selling (even though the vote buyer could increase the price and create a "vote selling lottery"). In this definition, it also makes sense to drop the quantification over the coercer's strategies to see the resistance to vote-buying for different vote choices.

RLT In the original RLT, a signature ballot will get revealed with probability $1 - f_{blind}$. If the vote buyer sees this, he can pay the vote seller. However, he will only pay the voter seller wrongly with a small probability p_{fl} equal to the probability that one of the honest voters cast the signature ballot, i.e. $\delta^{vb} \simeq 1 - f_{blind}$ which can be rather high and protects badly against vote buying.

Masked RLT For the masked ballots, we can, however, choose m such that several ballots will have the same masking as the signature ballot, making it hard for the vote buyer to assess if the signature ballot was cast. For the student election, we see from Fig. 6.1 that the number of matches with the optimal signature ballot $(0, 1, 1, 1, 1)$ is binomially distributed with an expectation value of 18.4 colliding ballots and a standard deviation of around 4.

For a more precise example, we can consider three candidates election with probabilities $(1/2, 1/2, 0)$ as above and assume that the goal of the voter is to cast 0 for candidate 1 and $p_{\text{fl}} = 0$. For $m = 1$, we will have $\delta^{vb} = 0$, but for $m = 2$, the vote-buyer can demand a vote for candidate 1 and 3 and payout if he sees $(1, *, 1)$. Any counter-strategy with 0 for candidate 1 gives $\delta^{vb} = 1/3$.

We note that the new quantitative definitions for no deniability coercion-resistance (Def. 14), the weak vote-buying resistance (Def. 15) and the original δ^{cr}-coercion-resistance (Def. 13) are considering different aspects of coercion-resistance and stating the three different δ-values gives a more nuanced description of the security of a given voting protocol. Also, note that the δ values are calculated using potentially different strategies for the coercer and voter. Finding unified strategies optimising the parameters is an interesting line of future work. Finally, there are natural, more fine-grained definitions extending these, which should be also considered in the future.

6.5 Conclusion

We have shown that the idea of risk-limiting tallies and risk-limiting verification can be applied effectively to complex ballots. By partially masking each ballot rather than simply masking a subset of the ballots as in the original RLT and RLV we gain far greater flexibility in terms of masking strategies. This will be explored further in order to optimise the trade-offs between the various measures defined here in future work.

The approach is more robust against any claims of being undemocratic: all ballots are counted .The only compromise then is some reduction in the level of verifiability, but this can be adjusted and is probably acceptable. If we compare this with ThreeBallot, there the chance of detecting a manipulated ballot is $1/3$, assuming that the attacker does not learn which ballot was retained by the voter. In our case, we can achieve a good level of coercion mitigation with, say, a shrouding of $\pm 1/2$ of each ballot.

Finally, we did a preliminary analysis of the quantitative privacy for the different tally methods and the coercion-resistance, in particular, the probability a coerced voter gets undeniably caught. The new masked tallies, however, are more appropriate for receipt-freeness, whereas the old RLT provides good plausible deniability to coerced voters. This suggests combining both methods when possible, but future work is needed to define the precise level of vote-buying resistance.

Chapter 7

Conclusion

In this chapter, we give an overall summary of the different projects carried during this PhD journey and we give insights for future work.

Contents

7.1 Summary

In this thesis manuscript, we designed, studied and analyzed the security of four security protocols.

Our first work involved designing a new authenticated key exchange protocol approaching some functionalities of public key cryptography. We presented a simple way of establishing a common secret key between participants without the central authority being active in the real distribution keys phase. Our design is a coupon-collector security-based protocol and is strong against three adversary models, which was easy to prove thanks to the simple constructions. We showed that it is secure against a passive attacker trying only to monitor the communications, against a man-in-the-middle attack where the adversary is trying to alter the sent messages and, finally, secure against an active attacker who is trying to impersonate one or more of the participants.

An attacker is able to break the system with a non-negligible probability if and only if he succeeds to break all the participant(s) keys, which happens with a certain probability that we calculated. Otherwise, the adversary is unable to distinguish the common shared secret key from a random one. Furthermore, we discussed parameters choice and we provided the optimal parameters to take into consideration for a better security.

In the second project, we focused on improving the robustness of the OV-Net protocol [22]. Remember that this protocol is conducted in two rounds and requires all participants to participate in obtaining the sum of all votes collaboratively. However, this protocol fails in achieving its goal if accidentally or maliciously one or more participants refuse to pursue the protocol. To solve this issue, we proposed to run the OV-Net in several parallel elections and to give each participant the possibility to vote in a selection of elections randomly chosen. This has improved a lot the protocol and made it more DoS-resistant; however, this comes at the cost of privacy, accuracy and computation. Nevertheless, we believe that our protocol is well suggested for decision-making applications.

Moreover, we presented three different algorithms on how to tally-combine the votes in this new setting depending on whether we want a naive computation of the votes, meaning an average of all the votes, a tally with minimum variance, or a tally with zero bias.

The third project deals with the concept of information with a foreseeable lifespan. We proposed a new method for storing information on DNA-RNA oligonucleotides and guaranteeing its destruction after a certain time. We proposed to synthetically manufacture a new molecule by linking DNA and RNA molecule in order to form a new DNA-RNA molecule ready to hold our secret information. We mention that the coding mechanism is a simple correspondence between bits and DNA nucleobases, which we keep as a private information.

We have modeled the information lifetime considering three different periods: life, agony and death, and we described each one in detail. We have showed that the information could be fully recovered during the first phase, partially recovered within the second with a non-negligible probability and completely destroyed with a high probability when it is dead, and this is according to the number of cuts, a notion that we defined based on the RNA degradation, that we have in each phase.

The last project is about the masked ballots. We presented an improvement of the previous work [25]. Recall that although the technique used in that paper is considered to counter the vote privacy, the coercion-resistance and the vote-buying issues with a good probability, it, unfortunately, does not allow all the ballots to be counted, which makes it an undemocratic approach. Its principle consists of shrouding a portion of the ballots and counting the tally based on the unshrouded ballots only. The shrouded ballot selection is randomly chosen in a way guaranteeing a certain confidence level of the output.

Our improvement is to mask each ballot rather than simply masking a subset of the ballots as in the original RLT. Instead of a full ballot disclosure, only a subset of the ballot components is known to the public. This approach is more robust against any claims of being undemocratic because all ballots are counted. We conducted a preliminary analysis of the different security measures: privacy, coercion-resistance, receipt-freeness, no deniability.

Overall, we have seen several security protocols with several security parameters in this thesis. According to each setting, we have proposed new ways on how to tune and optimize the parameters in order to get better security performances. The dominant ideas of this thesis manuscript are to adopt the mechanisms of doubling and masking: considering several racks for distributing keys, running the OV-Net in parallel elections, manufacturing multiple copies of the DNA-RNA molecules and masking a subset of the components of each ballot.

7.2 Future Work

We have shown during this thesis that the simple doubling/multiplying or masking ideas we adopted in designing all of our protocols demonstrated indeed a good effect on improving their security; however, it also generated some negatives issues. Therefore, we propose as a future work the following points:

- A natural research direction for our first project would be to formally analyze both the similarities of our proposed construction with standard public-key cryptography schemes and the post-quantum nature of our key distribution protocol.
- Another possible venue of future research is to consider hybrids of the current protocol, *e.g.* by achieving forward secrecy relying on a computational assumption.
- We have seen that the improvement in robustness that we achieved by running the OV-Net protocol in multiple elections came at the cost of privacy and accuracy. This is indeed tolerable, for instance, for decision making or financial applications; however, it is problematic for presidential elections for example. A possible future direction would be to explore other mechanisms to improve privacy and accuracy.
- As we have seen in our theoretic analysis, having multiple and very long oligonucleotides enhances a lot the security of our system. However, it is problematic in the biological viewpoint, and it can yield different hydrolysis rates compared to small-length oligonucleotides. Solving this biochemical challenge can be the subject of future work.
- Another possible future work for our last project is to invest in protecting information from copying mechanisms within the accessed period of time. Indeed, this perpendicular issue is not addressed in our work but it is a real security issue that should be well investigated.
- Propose an optimization of the trade-offs between the various measures defined in Chapter 6.
- When possible, combine both the original RLT method with our masked ballots to define the precise level of vote-buying resistance.

Bibliography

[1] URL: https://www.encryptionconsulting.com/.

[2] Gildas Avoine, Sébastien Canard, and Loïc Ferreira. "IoT-Friendly AKE: Forward Secrecy and Session Resumption Meet Symmetric-Key Cryptography". In: *ESORICS'19*. Vol. 11736. Lecture Notes in Computer Science. Springer, 2019, pp. 463–483.

[3] Gergei Bana et al. "Time, Privacy, Robustness, Accuracy: Trade Offs for the Open Vote Network Protocol". In: *Cryptology ePrint Archive* (2021).

[4] Olivier Baudron et al. "Practical multi-candidate election system". In: *Proceedings of the Twentieth Annual ACM Symposium on Principles of Distributed Computing, PODC 2001, Newport, Rhode Island, USA, August 26-29, 2001*. Ed. by Ajay D. Kshemkalyani and Nir Shavit. ACM, 2001, pp. 274–283. DOI: 10.1145/383962.384044. URL: https://doi.org/10.1145/383962.384044.

[5] Mihir Bellare and Phillip Rogaway. *Entity Authentication and Key Distribution*. http://cseweb.ucsd.edu/~mihir/papers/eakd.pdf. full version. 1993.

[6] Mihir Bellare and Phillip Rogaway. "Entity Authentication and Key Distribution". In: *Advances in Cryptology - CRYPTO'93*. Vol. 773. Lecture Notes in Computer Science. Springer, 1994, pp. 232–249.

[7] Josh Benaloh, Philip B Stark, and Vanessa Teague. "VAULT: Verifiable Audits Using Limited Transparency". In: *E-Vote-ID 2019* (2019), p. 69.

[8] Josh Benaloh et al. "SOBA: Secrecy-preserving Observable Ballot-level Audit." In: *EVT/WOTE* 11 (2011).

[9] Marc Beunardeau et al. "Authenticated key distribution: when the coupon collector is your enemy". In: *International Conference on Information Technology and Communications Security*. Springer. 2019, pp. 1–20.

[10] Yee-Wai Cheung et al. "Evolution of abiotic cubane chemistries in a nucleic acid aptamer allows selective recognition of a malaria biomarker". In: *Proceedings of the National Academy of Sciences* 117.29 (2020), pp. 16790–16798.

[11] Charles J. Colbourn and Jeffrey H. Dinitz. *Handbook of combinatorial designs*. CRC press, 2006.

[12] Colin Cooper, Alan M. Frieze, and Wesley Pegden. “On the rank of a random binary matrix”. In: *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*. Ed. by Timothy M. Chan. SIAM, 2019, pp. 946–955. DOI: `10.1137/1.9781611975482.58`. URL: `https://doi.org/10.1137/1.9781611975482.58`.

[13] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. “Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols”. In: *Advances in Cryptology - CRYPTO ’94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994, Proceedings*. Ed. by Yvo Desmedt. Vol. 839. Lecture Notes in Computer Science. Springer, 1994, pp. 174–187. DOI: `10.1007/3-540-48658-5\_19`. URL: `https://doi.org/10.1007/3-540-48658-5%5C_19`.

[14] Ronald Cramer et al. “Multi-Autority Secret-Ballot Elections with Linear Work”. In: *Advances in Cryptology - EUROCRYPT ’96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*. Ed. by Ueli M. Maurer. Vol. 1070. Lecture Notes in Computer Science. Springer, 1996, pp. 72–83. DOI: `10.1007/3-540-68339-9\_7`. URL: `https://doi.org/10.1007/3-540-68339-9%5C_7`.

[15] Whitfield Diffie and Martin Hellman. “New Directions in Cryptography”. In: *IEEE Transactions on Information Theory* 22.6 (Sept. 1976), pp. 644–654.

[16] Whitfield Diffie and Martin Hellman. “New directions in cryptography”. In: *IEEE transactions on Information Theory* 22.6 (1976), pp. 644–654.

[17] Bennett Eisenberg. “On the Expectation of the Maximum of IID Geometric Random Variables”. In: *Statistics & Probability Letters* 78.2 (2008), pp. 135–143.

[18] Marie Flamme et al. “Chemical methods for the modification of RNA”. In: *Methods* 161 (2019), pp. 64–82.

[19] Rosario Giustolisi, Vincenzo Iovino, and Peter B Rønne. “On the possibility of non-interactive e-voting in the public-key setting”. In: *International Conference on Financial Cryptography and Data Security*. Springer. 2016, pp. 193–208.

[20] Jens Groth. “Efficient Maximal Privacy in Boardroom Voting and Anonymous Broadcast”. In: *Financial Cryptography, 8th International Conference, FC 2004, Key West, FL, USA, February 9-12, 2004. Revised Papers*. Ed. by Ari Juels. Vol. 3110. Lecture Notes in Computer Science. Springer, 2004, pp. 90–104. DOI: `10.1007/978-3-540-27809-2\_10`. URL: `https://doi.org/10.1007/978-3-540-27809-2%5C_10`.

[21] Feng Hao. “A 2-Round Anonymous Veto Protocol”. In: *Security Protocols, 14th International Workshop, Cambridge, UK, March 27-29, 2006, Revised Selected Papers*. Ed. by Bruce Christianson et al. Vol. 5087. Lecture Notes in Computer Science. Springer, 2006, pp. 212–214. DOI: `10.1007/978-3-642-04904-0\_29`. URL: `https://doi.org/10.1007/978-3-642-04904-0%5C_29`.

[22] Feng Hao, Peter Y. A. Ryan, and Piotr Zielinski. “Anonymous voting by two-round public discussion”. In: *IET Information Security* 4.2 (2010), pp. 62–67. DOI: `10.1049/iet-ifs.2008.0127`. URL: `https://doi.org/10.1049/iet-ifs.2008.0127`.

[23] Feng Hao and Peter YA Ryan. *Real-world electronic voting: Design, analysis and deployment*. CRC Press, 2016.

[24] Feng Hao and Piotr Zielinski. “A 2-Round Anonymous Veto Protocol”. In: *Security Protocols, 14th International Workshop, Cambridge, UK, March 27-29, 2006, Revised Selected Papers*. Ed. by Bruce Christianson et al. Vol. 5087. Lecture Notes in Computer Science. Springer, 2006, pp. 202–211. DOI: `10.1007/978-3-642-04904-0\_28`. URL: `https://doi.org/10.1007/978-3-642-04904-0%5C_28`.

[25] Wojciech Jamroga et al. “Risk-Limiting Tallies”. In: *International Joint Conference on Electronic Voting*. Springer. 2019, pp. 183–199.

[26] Vincent Rijmen Joan Daemen. *The Design of Rijndael: AES — The Advanced Encryption Standard (Information Security and Cryptography)*. Springer, 2002.

[27] Marc Joye and Sung-Ming Yen. “ID-based Secret-key Cryptography”. In: *SIGOPS Oper. Syst. Rev.* 32.4 (Oct. 1998), pp. 33–39.

[28] Katalin Karikó et al. “Incorporation of pseudouridine into mRNA yields superior nonimmunogenic vector with increased translational capacity and biological stability”. In: *Molecular therapy* 16.11 (2008), pp. 1833–1840.

[29] Donaat Kestemont et al. “XNA ligation using T4 DNA ligase in crowding conditions”. In: *Chemical Communications* 54.49 (2018), pp. 6408–6411.

[30] Dalia Khader et al. “A fair and robust voting system by broadcast”. In: *Lecture Notes in Informatics (LNI), Proceedings-Series of the Gesellschaft fur Informatik (GI)* (2012), pp. 285–299.

[31] Aggelos Kiayias and Moti Yung. “Self-tallying Elections and Perfect Ballot Secrecy”. In: *Public Key Cryptography, 5th International Workshop on Practice and Theory in Public Key Cryptosystems, PKC 2002, Paris, France, February 12-14, 2002, Proceedings*. Ed. by David Naccache and Pascal Paillier. Vol. 2274. Lecture Notes in Computer Science. Springer, 2002, pp. 141–158. DOI: `10.1007/3-540-45664-3\_10`. URL: `https://doi.org/10.1007/3-540-45664-3%5C_10`.

[32] Michiko Kimoto and Ichiro Hirao. “Genetic alphabet expansion technology by creating unnatural base pairs”. In: *Chemical Society Reviews* 49.21 (2020), pp. 7602–7626.

[33] David Kodr et al. "Carborane-or Metallacarborane-Linked Nucleotides for Redox Labeling. Orthogonal Multipotential Coding of all Four DNA Bases for Electrochemical Analysis and Sequencing". In: *Journal of the American Chemical Society* (2021).

[34] Pawan Kumar and Marvin H Caruthers. "DNA Analogues Modified at the Nonlinking Positions of Phosphorus". In: *Accounts of Chemical Research* 53.10 (2020), pp. 2152–2166.

[35] Ralf Küsters, Tomasz Truderung, and Andreas Vogt. "A game-based definition of coercion resistance and its applications". In: *J. Comput. Secur.* 20.6 (2012), pp. 709–764. DOI: 10.3233/JCS-2012-0444.

[36] Ralf Küsters, Tomasz Truderung, and Andreas Vogt. "Verifiability, privacy, and coercion-resistance: New insights from a case study". In: *2011 IEEE Symposium on Security and Privacy*. IEEE. 2011, pp. 538–553.

[37] Yingfu Li and Ronald R. Breaker. *Kinetics of RNA Degradation by Specific Base Catalysis of Transesterification Involving the 2¢-Hydroxyl Group.* 1999.

[38] J Liedtke et al. "Ordinos: A Verifiable Tally-Hiding Electronic Voting Protocol". In: *IEEE 5th European Symposium on Security and Privacy (EuroS&P 2020).* 2020.

[39] Cailen M McCloskey et al. "Ligase-mediated threose nucleic acid synthesis on DNA templates". In: *ACS synthetic biology* 8.2 (2019), pp. 282–286.

[40] Patrick McCorry, Siamak F. Shahandashti, and Feng Hao. "A Smart Contract for Boardroom Voting with Maximum Voter Privacy". In: *Financial Cryptography and Data Security - 21st International Conference, FC 2017, Sliema, Malta, April 3-7, 2017, Revised Selected Papers*. Ed. by Aggelos Kiayias. Vol. 10322. Lecture Notes in Computer Science. Springer, 2017, pp. 357–375. DOI: 10.1007/978-3-319-70972-7_20. URL: https://doi.org/10.1007/978-3-319-70972-7%5C_20.

[41] Luke K McKenzie et al. "Recent progress in non-native nucleic acid modifications". In: *Chemical Society Reviews* (2021).

[42] Alfred J Menezes, Paul C Van Oorschot, and Scott A Vanstone. *Handbook of applied cryptography.* CRC press, 2018.

[43] Satu Mikkola, Tuomas Lönnberg, and Harri Lönnberg. "Phosphodiester models for cleavage of nucleic acids". In: *Beilstein journal of organic chemistry* 14.1 (2018), pp. 803–837.

[44] Roger M Needham and Michael D Schroeder. "Using Encryption for Authentication in Large Networks of Computers". In: *Communications of the ACM* 21.12 (1978), pp. 993–999.

[45] Fernando P Polack et al. "Safety and efficacy of the BNT162b2 mRNA Covid-19 vaccine". In: *New England Journal of Medicine* 383.27 (2020), pp. 2603–2615.

[46] Marleen Renders et al. “A method for selecting modified DNAzymes without the use of modified DNA as a template in PCR”. In: *Chemical Communications* 51.7 (2015), pp. 1360–1362.

[47] Peter Y A Ryan, Peter B Rønne, and Vincenzo Iovino. “Selene: Voting with transparent verifiability and coercion-mitigation”. In: *International Conference on Financial Cryptography and Data Security*. Springer. 2016, pp. 176–192.

[48] Peter YA Ryan et al. “Who was that masked voter? The tally won’t tell!” In: *International Joint Conference on Electronic Voting*. Springer. 2021, pp. 106–123.

[49] Heller Sadava Hillis and Berenbaum. *Life, The Science Of Biology*. 2009.

[50] Kazue Sako and Joe Kilian. “Receipt-free mix-type voting scheme”. In: *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 1995, pp. 393–403.

[51] Claus-Peter Schnorr. “Efficient Signature Generation by Smart Cards”. In: *J. Cryptol.* 4.3 (1991), pp. 161–174. DOI: `10.1007/BF00196725`. URL: `https://doi.org/10.1007/BF00196725`.

[52] Mohamed Seifelnasr, Hisham S. Galal, and Amr M. Youssef. “Scalable Open-Vote Network on Ethereum”. In: *Financial Cryptography and Data Security - FC 2020 International Workshops, AsiaUSEC, CoDeFi, VOTING, and WTSC, Kota Kinabalu, Malaysia, February 14, 2020, Revised Selected Papers*. Ed. by Matthew Bernhard et al. Vol. 12063. Lecture Notes in Computer Science. Springer, 2020, pp. 436–450. DOI: `10.1007/978-3-030-54455-3\_31`. URL: `https://doi.org/10.1007/978-3-030-54455-3%5C_31`.

[53] Masoud Sharif and Babak Hassibi. “Delay Considerations for Opportunistic Scheduling in Broadcast Fading Channels”. In: *IEEE Transactions on Wireless Communications* 6.9 (2007), pp. 3353–3363.

[54] Sandeep Verma et al. “Modified Oligonucleotides: Synthesis and Strategy for Users. Biochem. 1998. 67: 99-134”. In: *1998 by Annual Reviews* ().

[55] De-Min Zhou and Kazunari Taira. “The hydrolysis of RNA: from theoretical calculations to the hammerhead ribozyme-mediated cleavage of RNA”. In: *Chemical reviews* 98.3 (1998), pp. 991–1026.

RÉSUMÉ

Les protocoles de sécurité constituent un élément très important dans le domaine de la sécurité de l'information. Grâce à eux, l'échange des informations et des communications se passe en toute sécurité au sein du réseau. De ce fait, la protection de ces informations qui sont communiquées entre les utilisateurs et le serveur contre tout type d'attaques nécessite un haut niveau de sécurité. Cependant, augmenter le niveau de sécurité devient très coûteux, ce qui impacte la productivité. La question qui se pose ici est donc la suivante: quels sont les compromis de paramètres à prendre en considération afin d'assurer des meilleurs performances? Nous répondons à cette question le long de cette thèse en se basant sur deux simples idées : *duplication* et *masquage*. Nous développons plusieurs protocoles de sécurité et analysons l'effet de chaque paramètre sur l'autre.

MOTS CLÉS

Sécurité, Echange de clés, Secrets, Vote Electronique, Stockage d'information

ABSTRACT

Security protocols are an important component in the field of information security. They ensure the information and communications exchange within the network in a secure way. Protecting communications between the users and the server against any type of attack requires having high security level protocols. However, increasing the level of security usually becomes very costly which implies that the productivity level decreases. The question that boils down here is therefore: what are the parameter trade-offs to take into consideration in order to improve the performance? We provide in this thesis an answer to this question based on two simple ideas of *duplicating* and *masking*. We develop several security protocols and analyze the effect of each parameter on the others.

KEYWORDS

Security, Key Exchange, Secrets, Electronic Voting, Data Storage