THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :

Institut Supérieur de l'Aéronautique et de l'Espace (ISAE)

Présentée et soutenue par :
Guillaume SMITH

le jeudi 4 décembre 2014

Titre :

Concevoir des applications temps-réel respectant la vie privée en exploitant les liens entre codes à effacements et les mécanismes de partages de secrets

Enabling private real-time applications by exploiting the links between erasure coding and secret sharing mechanisms

---

École doctorale et discipline ou spécialité :
ED MITT : Réseaux, télécom, système et architecture

Unité de recherche :
Équipe d'accueil ISAE-ONERA MOIS

Directeur(s) de Thèse :
M. Jérôme LACAN (directeur de thèse)
Mme Roksana BORELI (directrice de thèse)
M. Emmanuel LOCHIN (co-directeur de thèse)

Jury :
M. Aruna SENEVIRATNE - co-directeur de thèse
Mme Roksana BORELI, NICTA-UNSW - directrice de thèse
M. Mathieu CUNCHE, INSA Lyon
M. Jérôme LACAN, ISAE - directeur de thèse
M. Emmanuel LOCHIN, ISAE - co-directeur de thèse
M. Nicolas NORMAND, IRCCyN - Rapporteur
M. Vincent ROCA, INRIA - Rapporteur

Enabling Private Real-Time Applications by Exploiting the Links Between Erasure Coding and Secret Sharing Mechanisms

Guillaume Smith

A thesis submitted in fulfillment
of the requirements for the degree of
Doctor of Philosophy

The University of New South Wales
School of Electrical Engineering and Telecommunications
Sydney, Australia

in a cotutelle agreement with

Département Mathématiques, Informatique et Automatique - Équipe MARS
Institut Supérieur de l'Aéronautique et de l'Espace (ISAE) - Toulouse, France

September 2014

Abstract

A huge amount of personal data is shared in real time by online users, increasingly using mobile devices and (unreliable) wireless channels. There is a large industry effort in aggregation and analysis of this data to provide personalised services, and a corresponding research effort to enable processing of such data in a secure and privacy preserving way. Secret sharing is a mechanism that allows private data sharing, revealing the information only to a select group. A parallel research effort has been invested in addressing the performance of real time mobile communication on lossy wireless channel, commonly improved by using erasure codes. In this thesis, we bring together the theoretically related fields of secret sharing and erasure coding, to provide a rich source of solutions to the two problem areas. Our aim is to enable solutions that deliver the required performance level while being efficient and implementable. The thesis has the following contributions.

We evaluate the applicability of a new class of Maximum Distance Separable (MDS) erasure codes to transmission of real time content to mobile devices and demonstrate that the systematic code outperforms the non-systematic variant in regards to computation complexity and buffer size requirements, making it practical for mobile devices.

We propose a new Layered secret sharing scheme for real time data sharing in Online Social Networks (OSNs). The proposed scheme enables automated profile sharing in OSN groups with fine-grained privacy control, via a multi-secret sharing scheme comprising of layered shares. The scheme does not require reliance on a trusted third party. Compared to independent sharing of specific profile attributes (e.g. text, images

or video), the scheme does not leak any information about what is shared, including the number of attributes and it introduces a relatively small computation and communications overhead.

Finally, we investigate the links between MDS codes and secret sharing schemes, motivated by the inefficiency of the commonly used Shamir scheme. We derive the theoretical links between MDS codes and secret sharing schemes and propose a novel MDS code based construction method for strong ramp schemes. This allows the use of existing efficient implementations of MDS codes for secret sharing and secure computing applications. We demonstrate that strong ramp schemes deliver a significant reduction of processing time and communication overhead, compared to Shamir scheme.

Extrait

Une large quantité de données personnelles sont partagées en temps réel par des utilisateurs en ligne, utilisant de plus en plus des terminaux mobiles avec connexion sans-fil. L'industrie s'efforce d'accumuler et d'analyser ces données pour fournir de nouveaux services ou des améliorations. La recherche fournit un effort équivalent pour permettre de traiter ces données de façon sécurisée et protectrice de la vie privée. Les problèmes de performance des communcations temps réels sur terminaux mobiles sur un canal sans-fil sont aussi étudiés. Les codes à effacement sont un moyen courant d'améliorer ces performances. Le secret sharing est un mecanisme permettant de partagé des données privées, ne les révélant qu'à un groupe d'utilisateur choisi. Dans cette thèse, nous lions théoriquement les secret sharing schemes et les codes à effacement, pour fournir une source plus riche de solutions aux deux problèmes. Notre objectif est de fournir des solutions ayant le niveau de sécurité souhaité, tout en restant efficace et implémentable. Les contributions de cette thèse sont les suivantes.

Nous évaluons l'applicabilié d'une nouvelle classe de codes effacements à Maximum Distance Séparable (MDS) pour transférer du contenu temps réel à des terminaux mobiles, et nous démontrons que le code systématique réduit grandement la compléxité d'éxecution et la taille nécessaire des tampons en comparaison du code non systématique, faisant de lui un bon candidat pour une application mobile.

Nous proposons un nouveau Layered secret sharing scheme pour le partage en temps réel de données sur des réseaux sociaux (OSNs pour Online Social Network). Le procédé permet de partager automatiquement un profile dans un groupe défini dans

un OSN, en utilisant un multi-secret sharing scheme formé de multiples couches. Le procédé ne dépend nullement d'un tiers de confiance. Comparé à un partage simple de chaque attributs (pouvant être un texte, une image ou une vidéo), le procédé ne divulgue aucune information à propos de ce qui est partagé, pas même le nombre de ceux-ci, et il induit une augmentation relativement faible du temps de calcul et des données à envoyer.

Finalement, nous étudions les liens entre les codes MDS et les secret sharing schemes, ayant pour motivation l'inefficacité du trs populaire Shamir secret sharing scheme. Nous établissons les liens théoriques entre les deux domaines et nous proposons une nouvelle construction de strong ramp schemes á partir de codes MDS. Ceci permet d'utiliser les codes MDS existants et efficaces pour des applications de partage de secret et de calculs distribués et sécurisés. Nous évaluons et montrons une réduction significative de temps de calcul et du coût de communcation en utilisant un strong ramp scheme, en comparaison avec le procédé de Shamir.

Acknowledgements

I would like first to thank my supervisors, Roksana Boreli, Jérôme Lacan, Emmanuel Lochin and Aruna Seneviratne for their support throughout my work on this thesis. Roksana has been helping me so much from refining and expressing ideas to writing precisely these ideas on paper. Manu and Jérôme have always be supporting me even when my work was not as related to their working area. Aruna was a strong support at NICTA, without whom securing and extending my scholarship would have been much harder.

I'm thankful to NICTA, ISAE and TéSA to have funded my doctoral studies, and provides me a perfect environment during my PhD. I'm also thankful to Annie and Martine in Toulouse, who have born my slowness in regards to administrative hurdles, Prashanti, Rema, Victoria Cleo, Alman and Denise in Sydney always there if I had any questions concerning administrative issues.

This thesis and I have benefited so much from all my interactions with all the people I have met during my studies. I'm extremely grateful to Olivier and Guillaume who have always been supportive with their ideas, advices and their delicious own made beers. It was also very insightful to work on research papers with many co-authors (in no particular order, Pierre-Ugo, Mentari, Dali, Arik and Babil). I extend my gratitude to all the other people I have the chance to meet, both in ENSICA and NICTA: Patrick for his support and advices, Hugo and Rémi who have been able to handle my bad jokes and without whom I wouldn't have started running half-marathons, Thivya and Terence supporting each other during the writing period, the pizza-talk group with whom we had wonderful discussion (Yan, Benoy, Salim, Niko, Amit and all the others), the number of French interns with whom I improved so much my skills in table-football, and naturally so many other people who have made these labs exciting places to work, learn, develop new ideas and enjoy life.

Finally, I would like to thank my parents, who have always been so supportive, even with my choice of leaving France for so long, with only few occasions to see them. I also thank Dani for her understanding and support.

Table of Contents

List of Figures

List of Tables

List of Acronyms

ARQ	Automated Repeat reQuest
CPU	Central Processing Unit
FEC	Forward Error/Erasure Correction
IDA	Information Dispersal Algorithm
IPTV	Internet Protocol television
ISAE	Institut Supérieur de l'Aéronautique et de l'Éspace
LDPC	Low Density Parity Check
MDC	Multiple Description Coding
MDS	Maximum Distance Separable
MPC	Secure Multi-party Computation
NICTA	National ICT Australia
OSN	Online Social Network
P2P	Peer to Peer
PER	Packet Error/Erasure Rate
PSNR	Peak Signal-to-Noise Ratio
RTT	Round Trip Time
SEPIA	Security through Private Information Aggregation
SVC	Scalable Video Coding
TTP	Trusted Third Party

UNSW University of New South Wales

VoIP Voice over Internet Protocol

Résumé de la Thèse en Français

1 Introduction

1.1 Problèmes Liés à la Protection de la Vie Privée lors du Partage de Données Personnelles en Temps Réel: Contexte et Motivation

Une incroyable quantité de données privées sont partagées à l'aide des sites de réseaux sociaux (OSN pour *Online Social Network*) tels que Facebook, YouTube et les nombreux services gravitant autour de ceux-ci. Ces services sont devenus extrêmement populaire avec environ 1.3 milliard d'utilisateurs pour Facebook.[1] De même, plus d'un milliard d'utilisateurs visitent YouTube chaque mois.[2]

Les utilisateurs de ces réseaux partagent de plus en plus d'informations privées telles que leur nom, leur adresse, lieu d'étude et de travail, mais aussi de nombreuses photos et vidéos. Avec cette notion de partage, il faut comprendre qu'un utilisateur peut à la fois être créateur de données, mais aussi consommateurs.

L'accès à ces réseaux sociaux se fait de plus en plus souvent par terminal mobile (tel que tablette et intelliphone): 52% des utilisateurs de Facebook accèdent à ce service par ces plate-formes.

Cependant le partage de ces données personelles lève de nombreux problèmes si celles-ci sont accessibles par d'autres utilisateurs ou entités que ceux ciblés. Dans la majorité des cas, le fournisseur du service a aussi accès aux données personnelles de ses

[1] Facebook statistics as of January 2014, from http://www.statisticbrain.com/facebook-statistics/

[2] YouTube statistics as of March 2014, from http://www.youtube.com/yt/press/statistics.html

utilisateurs, les utilisant pour rentabiliser ce service en leur fournissant des publicités ciblées fonction des informations qu'ils ont renseigné dans leur profile. Les utilisateurs doivent souvent l'accepter car quitter un réseau social pour un autre est complexe sachant que chaque utilisateur a en moyenne une centaine d'amis, il doit les motiver pour changer à leur tour de site. Ensuite, les règles de confidentialité régulant l'accès aux profiles aux autres utilisateurs du même système (ou vu publiquement en dehors du service) sont souvent complexes et demandent une importante intervention manuelle [20]. Ceci nécessite le développement de mécanismes supplémentaires pour protéger le cararctère privé des informations des utilisateurs.

A propos de la transmission de données, l'utilisation des terminaux mobiles pour mettre en ligne des contenus personnels sur les serveurs de réseaux sociaux et pour télécharger ces données intéressant qu'un certain nombre d'utilisateurs, crée un ensemble de défis traditionnellement liés à la communication sans-fil sur des canaux bruités. Ce qui est particulièrement pertinent lorsque l'accès l'information doit être en temps réel ou en quasi temps réel, comme par exemple des contenus vidéos. Nous notons que dans cette thèse, nous utilisons les termes temps réel (ou quasi temps réel) dans un sens plus large que leur définition courante venant des domaines de la vidéo en flux continu (streaming) et de la voix sur IP défini par 3GPP [35]: nous nous référons à tout types de trafic ou services qui utilisent des données ayant une date limite d'arrivée.

Nous basons notre recherche sur des solutions en rapport avec les différent problèmes de la protection de la vie privée et de l'intégrité de la transmission de données, se focalisant sur les codes à effacement à Maximum Distance Séparabale (MDS). Ces mécanismes sont traditionnellement utilisés pour parer les erreurs ou pertes produites par les canaux sans-fil [65]. Le partage d'information dans un cadres assurant une protection de la vie privée est généralement obtenu en utilisant des mécanismes de partage de secrets (plus couramment nommés secret sharing scheme) [97]. Nous considérons l'utilisation de codes MDS pour faire du secret sharing afin d'améliorer la faible efficacité de ces derniers. Utiliser un unique mécanisme pour une

double effet et dans deux différents domaines permettrait d'enrichir et d'apporter de nouvelles solutions aux deux domaines et motiverait de plus amples développelements de tels mécanismes.

1.2 Questions de Recherche

Dans cette thèse, nous abordons les questions concernant le partage de données privées en temps réels à l'aide de terminaux mobiles.

Tout d'abord nous portons notre attention sur la transmission de ces données temps réel sur un canal sans fil. Nous nous interrogeons sur l'applicabilité de codes à effacements avancés (basé sur des codes MDS) qui ont récemment été proposés pour ce type de canal. Notre objectif est de déterminer si de tels codes sont utilisables par des terminaux mobiles (intelliphones, tablettes, etc.) qui ont des capacités de calculs réduites et des ressources limitées, telles que leur mémoire.

La seconde question porte sur la protection des données personnelles partagées sur des réseaux sociaux. Nous nous focalisons sur le cas commun où un utilisateur partage de multiples informations privées, l'ensemble formant son profile. Nous examinons comment partager plusieurs informations piveées, tout en assurant qu'aucun indice ou détail les concernant ne s'ébruite, comme leur nombre et leur taille - qui pourraient être utilisées par des attaquants pour en connatre plus sur les utilisateurs.

Finalement nous étudions l'applicabilité du partage sécurisé de données personnelles dans un contexte de calcul sécurisé sur celles-ci, lorsqu'elles peuvent être constituées de larges fichiers tels que des images ou des vidéos. Ce sont des scénario communs dans les réseaux sociaux lorsque certains attributs deviennent accessibles à de nouveaux participants. Le plus courant et le plus utilisé, le partage de secrets de Shamir, est n'est pas réaliste au vu de son coût de communication. Ce mécanisme génère des parts faisant la taille du secret à partager à parti duquel les parts sont construites. Nous nous intéressons à des solutions pratiques pour permettre de partager et faire des calculs sécurisés efficacement sur de telles données.

1.3 Plan de la Thèse

Cette thèse est organisée de la façon suivante. Le Chapitre 2 fournit le contexte et un aperçu des recherches effectuées dans les domaines des codes MDS, du secret sharing et du calcul sécurisé et distribué entre plusieurs parties (MPC). Le Chapitre 3 présente une analyse de l'applicabilité des schémas de codage à la volée pour des applications temps réel sur des terminaux mobiles. Le Chapitre 4 inclus une présentation d'un nouveau mécanisme de partage de multiples secrets (basé sur de l'encodage en couches) et une évaluation de ses performances pour une utilisation dans un réseau social. Le Chapitre 5 rapproche les deux domaines des codes à effacements et du partage de secret (ainsi que le MPC) et présente une nouvelle méthode de construction de mécanisme de partage de secret efficace construit à partir de code MDS. Nous concluons et exposons les grandes lignes de possibles futures recherches. En appendice, un travail fait durant la thèse est présenté. Bien que dans le domaine des applications temps réel et des codes à effacement, il ne fait pas parti du cœur de cette thèse car il ne repose pas sur l'utilisation de corps finis comme les autres travaux.

1.4 Contributions

Nos principales contributions présentées dans cette thèse sont les suivantes:

- Une nouvelle classe de code à effacements pour les applications ayant des contraintes de délai, les schémas de codage à la volée, ont récemment été présentés. De précédentes recherches ont établi que ces codes offraient des améliorations par rapport aux délais et aux capacités accessibles. Malgré leurs caractéristiques prometteuses, peu est connu à propos des complexités des variantes systématique ou non-systématique de ces codes, notamment pour une transmission en temps réel de contenu multimédia. Nous cherchons à compléter ce manque en se focalisant spécifiquement sur les métriques appropriées à des terminaux mobiles: la taille nécessaire des tampons et la complexité de calculs des récepteurs. Nos contributions sont les suivantes: nous évaluons les deux variantes sur un canal

à effacements uniformément répartis ou en rafales. Les résultats obtenus sont sans équivoque et prouvent que le code systématique est plus performant que sa contre-partie non-systématique, que ce soit la taille nécessaire des tampons tout aussi bien que la complexité du décodage. Ces résultats sont présentés dans le Chapitre 3.

- Nous proposons un nouveau code de partage de donneées en couches et son application à un réseau social. Dans les réseaux sociaux actuels et commercialement offerts, l'accès aux informations contenues dans le profil d'un utilisateur est controlé par le fournisseur du réseau social (e.g. Facebook et Google+), utilisant les préférences choisies par l'utilisateur. Un nombre limité de règles, telles que celles permettant de définir des groupes d'amis, permettent de préciser des niveaux de protection, cependant elles sont complexes et nécessitent la présence d'un parti de confiance (le fournisseur du service) pour assurer leurs applications. Le mécanisme que nous proposons permet de partager automatiquement son profil dans des groupes de réseaux sociaux avec un contrôle fin des accès, via un mécanisme de partage de multiples secrets formés de couches, créées à partir des attributs (multiples secrets) du profile d'un utilisateur, les parts étant directement distribuées aux membres des groupes, sans nécessiter la présence d'un tiers de confiance. Le mécanisme peut prendre la forme par exemple d'une extension d'un navigateur internet, permettant l'automatisation de toutes ces opérations. Nous étudions la sécurité du mécanisme contre des attaques visant à obtenir plus d'informations à propos du profile d'un utilisateur. Nous fournissons aussi une analyse théorique du niveau de sécurité pour chaque attributs du profile. La Chapitre 4 présente cette contribution.

- Le partage de secret est une primitive importante dans beaucoup de protocoles pour le MPC, et le mécanisme de Shamir est un des plus couramment utilisé dans ce contexte. Cependant, le procédé de Shamir introduit un surcoût de

communication, qui pourrait limiter son applicabilitité lorsque les données à évaluer sont volumineuses. Les Strong ramp schemes, qui ont été proposés pour minimiser le probème de surcoût, présentent un compromis entre l'efficacité et le niveau de sécurité. Cependant, il n'y a que peu de constructions de tels mécanismes, et à notre connaissance, aucune évaluation empirique de leurs performances dans un contexte MPC n'a été réalisée. Nous proposons une nouvelle méthode de construction des strong ramp schemes, en démontrant que les parts d'un strong ramp schemes peuvent être directement extraites d'un code à effacement systématique et MDS. Cette construction permet de tirer profit d'un large nombre d'implémentations performantes et existantes de codes MDS pour le partage de secret et son application au MPC. Nous proposons aussi une construction supplémentaire dérivée du mécanisme de Shamir. Finalement nous évaluons les bénéfices en terme de performance des strong ramp schemes dans un scénario de calcul distribué en implémentant deux de ces procédés dans le système SEPIA MPC, et en les comparant au procédé de Shamir. Nous montrons que dans un scénario de surveillance de pannes de réseaux avec 20 participants fournissant les données en entrées et 20 participants faisant les calculs, le temps de calcul est réduit d'un facteur d'environ 44, et le coût de communication est réduit 20 fois, comparé au MPC utilisant le procédé de Shamir. Ceci est présenté dans le Chapitre 5.

2 Contexte et Travaux Connexes

Dans le Chapitre 2 de cette thèse, nous décrivons le contexte de cette thèse. Nous commençons par présenter dans la Section 2.1 la définition des corps de Galois, des propriétés de probabilité et de théorie de l'information qui sont la base des travaux présentés par la suite. Ensuite nous présentons les codes à effacements dans la section 2.2 et nous introduisons deux sous-catégories de ces codes: les codes MDS et les schémas de codage à la volée. Nous présentons ensuite le second sujet majeur de cette

thèse, le partage de secret dans la section 2.3, incluant les définitions de mécanismes faits pour partager un ou de multiples secrets, et d'autres travaux connexes. Finalement nous présentons une vue d'ensemble sur les travaux de recherches liant les codes MDS et les ramp schemes, sous-catégorie des mécanismes de partage de secret.

3 Procédé d'Encodage à la Volée pour une Communication Multicast de Données Multimédia

Il existe deux classes de mécanismes de fiabilité utilisant respectivement des retransmissions et de la redondance. Les mécanismes ARQ récupèrent les paquets manquants et les retransmettent. Par conséquent, la récupération d'un paquet manquant induit un délai d'au moins un Round Trip Time (RTT). Cependant, cela peut ne pas être acceptable pour des applications ayant des contraintes temporelles, c'est à dire ayant une limite à partir de laquelle un paquet est considéré comme périmé et inutile pour l'application le recevant. Comme présenté dans la Section 2.2, une solution communément utilisée pour ne pas avoir ce délai additionnel est d'ajouter des paquets de redondance au flux de données. Ce qui peut être fait en utilisant des codes à effacements.

Comme précisé dans la Section 2.2.2.1, pour passer outre la limitation du nombre de pertes permis dans un bloc de paquets encodé à l'aide d'un code à effacement par bloc, d'autres approches utilisent des schémas de codage à la volée [73][101][104], qui appartiennent à la classe des codes convolutionnels. Dans [73], les auteurs utilisent des codes convolutionels non binaire et montrent que le délai dû au décodage peut être réduit par l'utilisation d'une fenêtre glissante d'encodage des paquets de redondance, plutôt que par bloc. Plus récemment dans [101] et [104], les auteurs proposent d'utiliser des schémas de codage à la volée qui utilisent une fenêtre d'encodage élastique et utilisent un canal de retour non fiable (quand accessible) pour réduire la complexité d'encodage pour l'émetteur, sans impacter la quantité de données transférée. Comparé

à [73], les deux propositions permettent d'avoir une communication fiable sous certaines conditions. Cependant, la principale différence entre [101] et [104] est que le premier propose un code non-systématique tandis que le second utilise sa variante systématique.

A notre connaissance, il n'existe pas d'étude quantifiant la compléxité des codes convolutionels ayant une fenêtre d'encodage infinie, et analysant la taille nécesaire pour leur mémoire tampon assurant un bon fonctionnement de ces codes. Dans un scénario avec une unique source et un unique receveur, l'analyse théorique est triviale, le code systématique proposant logiquement une amélioration en terme de délai. Cependant, dans un contexte multicast, il est beaucoup plus compliqué d'estimer l'effet du nombre de receveurs sur chacun des receveurs.

Dans le Chapitre 3, notre objectif est de mesurer les bénéfices et les besoins de fonctionnement des deux variantes de schéma de codage à la volée, d'un point de vue de la taille de la mémoire tampon nécessaire pour les receveurs et de la compléxité, tout ceci afin d'évaluer l'applicabilité de tels procédés à un scénario d'envoi de flux multimédia sur un canal multicast. En particulier, cette analyse nous permettrait de déterminer si ces codes sont utilisables dans un environnement multicast dans lequel les receveurs seraient des mobiles (par exemple des intelliphones), qui ont de plus faibles capacité de calculs et des ressources limitées, recevant par exemple des vidéos.

La Section 2.2 présente les caractéristiques des schémas de codage à la volée comparées aux codes en blocs. Ensuite nous analysons la taille des mémoires tampon nécessaire à de tels codes sur un canal à effacements répartis uniformément dans la Section 3.3 tandis que dans la Section 3.4 nous analysons leur complexité. Nous présentons aussi dans la Section 3.5 une étude de la taille des mémoires tampon et de la compléxité de ces codes sur un canal à pertes en rafales. Finalement, nous concluons le chapitre dans la Section 3.6.

4 Procédé de Partage de Secret en Couches pour le Partage Automatique d'un Profile dans un Réseau Social

Les procédés de partage de secret ont été largement utilisés dans de nombreuses applications liées à la cryptographie et aux calculs distribués [6], pour offrir des stockages de données sécurisés, pour le MPC, les transferts inconscients généraux (generalised oblivious transfer), etc. Comme présenté dans la Section 2.3, pour avoir un procédé de partage de secret parfait (Section 2.3.1.1), la taille des parts (shares) doit être supérieure ou égale à la taille du secret partagé. Les strong ramp schemes comme présenté dans la Section 2.3.1.3 propose un compromis entre sécurité et coût de communication pour partager de lourds fichiers.

Dans le Chapitre 4, nous étudions l'utilisation du partage de secret pour offrir à un utilisateur d'un OSN la possibilité de partager son profil avec ses amis appartenant à différents groupes, tout en lui laissant contrôler certains paramètres liés à la protection de sa vie privée. Un utilisateur sauvegarde en toute confiance ses données privées sur les serveurs du fournisseur des principaux OSN facilement accessibles au grand public. L'accès au profil est ensuite géré par des politiques d'accès définissant quels attributs (localisation, statut marital, date de naissance, intérêts, etc.) peuvent être vus et par qui (amis proches, lointains, ou public). De nombreux challenges liés à la protection de la vie privée dans les OSN ont été identifiés par des chercheurs [66], incluant la dépendance à un tiers parti de confiance (TTP pour Trusted Third Party) pour gérer les paramètres de confidentialités. Notons par ailleurs la compléxité discutable de ces paramètrages dans les OSN actuels et leurs flexibilités limitées (les cercles de Google+ et les listes de Facebook sont un premier pas dans la direction d'améliorer cette flexibilité). Ceci motive notre intérêt à utiliser des techniques de partage de secret pour offrir à l'utilisateur un contrôle direct et précis des paramètres de confidentialité concernant ses données personnelles.

Plus spécifiquement, nous considérons le besoin de l'utilisateur d'un OSN de part-

ager de multiples attributs (secrets) constituant son profil avec différent niveaux de sécurité (i.e. seuils d'accès), dans des groupes d'amis, sans dépendre d'un TTP. De plus, un utilisateur d'OSN pourrait préférer ne pas dévoiler son niveau de protection désiré, et/ou le nombre d'attributs qu'il partage avec différents amis (ou groupes d'amis). Il peut être aussi préférable de ne pas partager ses informations concernant la garantie de sécurité de vie privée, car comme toutes informations, elles peuvent être utilisées pour désidentifier des données anonymes [80]. Bien que ces données n'aient pas encore été prouvées comme utiles pour désidentifier un utilisateur, éviter la divulgation de celles-ci est une propriété requise lors du développement d'un mécanisme de protection de vie privée. Comme mentionné dans la Section 2.3.2, les limitations des procédés de partage de secret actuellement proposés se rapportent à les flexibilité de leurs seuils de sécurité et à la divulgation de leurs paramètres définissant le même seuil pour chaques secrets et/ou utilisant des parts publiques donnant explicitement le nombre de secrets partagés.

Pour passer outre ces limitations, nous proposons un nouveau procédé de partage de multiples secrets en couches, qui intègre récursivement les parts construites à partir de secrets singuliers, en couches ayant une protection croissante, et qui encrypte chaque couche avec une clé générée à partir d'une part additionnelle. Chaque secret est ainsi protégé par un procédé de partage de secret fait pour partager un unique secret avec son propre seuil, et le nombre de secrets partagés est caché dans les couches des parts, ce qui assure que les seuils restes inconnus. Les principales contributions du Chapitre 4 sont les suivantes.

Nous proposons un nouveau procédé de partage de secret en couches qui offre une flexibilité dans les paramètres de niveaux de confidentialité. Nous introduisons un partage de profile automatisé et garantissant la protection de la vie privée des utilisateurs d'un OSN comme une utilisation possible de notre procédé. En générant des parts en couches (comprenant un ensemble d'attributs choisis pour être partagés auprès d'un groupe voulu) et en distribuant une part à chaque membre du groupe, l'utilisateur de

l'OSN impose automatiquement le déploiement de sa politique de confidentialité au sein des membres de ce groupe, sans dépendre d'un TTP.

Nous analysons la sécurité de notre procédé contre des attaques visant à illégitimement obtenir des informations à propos des profiles des utilisateurs. Nous montrons donc qu'aucune part en couches ne peut être obtenue par des attaques ayant comme prémices des connaissances antérieures concernant les informations incluses dans le profile. Nous fournissons aussi une analyse du nombre de parts en couches qu'un attaquant peut obtenir, étant à un nombre arbitraire de sauts d'un utilisateur cible dans son graphe social. Cette analyse peut être utilisée pour fournir aux utilisateurs de l'OSN un seuil minimal de protection de leurs données. Nous démontrons, en utilisant un exemple de graphe issu de Facebook, qu'un utilisateur ayant un nombre d'amis variables (allant jusqu'à 100) peut garantir la sécurité de son profile même lorsque la probabilité de fuites de ses informations est de 10% (probabilité que soit ses amis soit d'autres attaquants transmettent leurs parts à d'autres adversaires), sans compromettre la capacité de ses amis à accéder aux informations du profile. Tout ceci se reposant sur le nombre minimum de parts reçues.

Nous évaluons la complexité et le surcoût de communication induit par notre procédé, en utilisant notre implémentation de génération et de décodage des parts en couches. Elles utilisent soit le procédé de Shamir, soit un strong ramp scheme pour chacune des couches. Notre procédé à un coût de communication similaire au mécanisme naïf, pour lequel chaque secret est partagé indépendamment avec son propre procédé de partage de secret. Le procédé en couches accroît la complexité, allant jusqu'à l'accroître de 160% pour le décodage. Cependant ce délai reste relativement faible, en étant de l'ordre de la seconde (en utilisant un ramp scheme), même lorsque le profile contient de large secrets (par exemple des images), ce qui peut être considéré comme acceptable pour une utilisation en temps-réel.

Le Chapitre 4 est organisé de la façon suivante: dans la Section 4.2, nous présentons les travaux connexes concernant la protection de la vie privée dans les OSN. Nous

présentons notre procédé de partage de multiple secrets en couches et son utilisation pour le partage de profile dans un OSN dans la Section 4.3. Les détails concernant l'analyse de la sécurité de ce procédé sont fournis dans la Section 4.4, et dans la Section 4.5, nous évaluons les performances de notre procédé en couches en termes de complexité et coût de commnunication. Nous discutons des compromis/limitations de notre procédé et des utilisations possibles d'autres schémas de partage de secret pour générer les couches. Puis nous concluons ce chapitre dans la Section 4.7.

5 Codes Systématiques MDS et Strong Ramp Scheme

De plus en plus de données personnelles et professionnelles sont collectées, agglomérées et analysées pour fournir de meilleurs services.[3] La recherche fournit un effort équivalent pour permettre de stocker et utiliser ces données de façon sécurisée en garantissant le respect de leurs clauses de confidentialités, en réponse aux préoccupations du public et aux stricts régulations protégeant ces données.[4] MPC [24] est un mécanisme par lequel plusieurs entités peuvent collaborer pour calculer une fonction choisie de leurs entrées, assurant la confidentialité des entrées et l'intégrité du résultat. Le calcul sécurisé d'entrées distribuées est applicable à de nombreux scénarios, permettant à de multiples organisations d'utiliser conjointement leurs données privées ou confidentielles pour offrir un meilleur service (par exemple, les fournisseurs d'accès à Internet peuvent détecter et/ou diagnostiquer des pannes de réseaux [30]), et permettant par exemple d'analyser des données personnelles stocker sur des terminaux mobiles de particuliers [49].

MPC utilise le plus généralement les propriétés des procédés de partage de secret ou des circuits brouillés (grabled circuits). Dans le Chapitre 5,nous portons notre attention sur le MPC se reposant sur le partage de secret, qui est le plus apte à gérer

[3]McKinsey report on big data, http://www.mckinsey.com/insights/business_technology/big_data_the_next_frontier_for_innovation

[4]EU data protection rules reform, http://ec.europa.eu/justice/newsroom/data-protection/news/120125_en.htm

un grand nombre de parties (les circuits brouillés sont principalement conçus pour du calcul à deux parties [81]). La confidentialité du calcul est garantie par la sécurité du procédé de partage de secret sous-jacent.

Le choix entre un procédé de partage de secret parfait tel que celui de Shamir et un strong ramp scheme est un compromis entre une sécurité parfaite offerte par un procédé de partage de secret parfait, en opposition à une compléxité réduite et un coût de communication moindre offerts par un strong ramp scheme. Cependant un strong ramp scheme garantie une meilleure sécurité qu'un ramp scheme générique, comme décrit dans la Section 2.3.1. La sécurité est ici définie par des propriétés venant de la théorie de l'information, et plus particulièrement de propriétés liées à l'entropie.

Comme présenté dans la Section 2.3.1.3, bien que les strong ramp scheme soient prometteurs, peu de travaux concernant leurs constructions existent [4]. De précédents travaux ont aussi étudié les liens entre les ramp schemes génériques et les codes MDS [25] [22]. Par exemple, les auteurs de [85] proposent une méthode pour construire de codes à seuil, utilisant des codes MDS. Mais ces méthodes ne sont pas applicables aux strong ramp schemes. Tout compte fait, peu de constructions de strong ramp schemes ont été proposées et, à notre connaissance, aucune évaluation de la performance de ces porcédés n'a été présentée. Dans le Chapitre 5, nous avons donc pour but de répondre à ces manques. Nos contributions peuvent se résumer de cette façon.

Tout d'abord, nous présentons une nouvelle méthode pour construire des strong ramp schemes en utilisant n'importe quel type de codes MDS (qu'ils soient systématiques ou non). A cette fin, nous démontrons que les paquets encodées (les éléments redondants) produits par n'importe quel code MDS systématique [65] vérifient les conditions définissant un strong ramp scheme. En utilisant les résultats de [59], nous proposons une méthode pour construire un strong ramp scheme à partir de n'importe que code MDS. Afin d'être exhaustif, nous montrons de plus comment construire des codes MDS à partir de n'importe quel ramp scheme.

Ensuite nous présentons une seconde construction d'un strong ramp scheme, dérivée

du procédé de Shamir. Cette méthode utilise une construction alternative (mais équivalente à l'originale [97]) des parts à partir de points d'un polynôme, au lieu d'utiliser les coefficients de celui-ci.

Finalement, nous évaluons les bénéfices des strong ramp schemes dans une application MPC pour contrôler les pannes de réseau. Notre évaluation compare les implémentations de (a) un strong ramp scheme basé sur un code MDS de Reed-Solomon [90] et (b) un strong ramp scheme dérivé du procédé de Shamir, tout deux intégrés dans la structure MPC de SEPIA [17]. Nous considérons un scénario réaliste avec 20 pairs fournissant leurs entrées et 20 pairs de calculs, utilisant les données fournies par le contrôle des pannes de réseau venant de fournisseurs d'accès Internet. Nous montrons par cet exemple que les strong ramp schemes peuvent réduire la compléxité (mesurée en temps de calcul CPU) d'environ 44 fois, et le coût de communication pour un pair offrant ses données de 20 fois, en comparaison des performances du procédé de référence qu'est celui de Shamir.

6 Conclusion

Cette thèse aborde le challenge de partager des informations privées en temps réel, à l'aide de terminaux mobiles ayant des ressources limitées (mémoires, capacité de calculs et débits). Plus particulièrement, nous abordons trois questions de recherches et proposons des contributions liés à ces problématiques. La première question est liée à la complexité et les besoins en mémoire de deux variantes de schémas de codage à la volée ayant une fenêtre d'encodage élastique conçu pour des applications temps réel. La seconde question abordée est le problème du partage du profile privé d'un utilisateur d'un OSN sans fuite d'information concernant ce profile et sans avoir à dépendre d'un tiers parti de confiance. Finalement, nous abordons le problème de l'applicabilité du partage sécurisé de large volume de données privées afin d'effectuer des calculs sécurisés sur ces données. Nous détaillons ces trois contributions par la suite.

Applicabilité des Schémas de Codage à la Volée sur des Terminaux Mobiles

Les procédés de codage à la volée avec une fenêtre d'encodage élastique, proposés et évalués dans de précédentes recherches, améliorent les performances des applications temps réel en diminuant les délais de décodage [103]. Dans le Chapitre 3, nous comparons les différences entre les variantes systématiques et non-systématiques de ces codes, d'un point de vue des tailles mémoires nécessaires et de la complexité de codage et de décodage, dans le cadre d'un scénario de multicast sans-fil dans lequel les receveurs sont des terminaux mobiles.

A cette fin nous avons simulé et considéré deux types de canaux à effacements: avec des pertes uniformément distribuées et le modèle de Gilbert-Elliott. Nos résultats montrent que l'approche systématique est plus performante que celle non-systématique pour les deux métriques considérées.

Procédé de Partage de Secret en Couches pour le Partage de Profiles sur des Services OSN

Le Chapitre 4 étudie l'applicabilité des procédés de partage de secret pour une application de partage de profiles dans un OSN, dans lequel les utilisateurs veulent se protéger de fuites concernant les informations de leur profile, que ce soit auprès de partis tiers (incluant les fournisseurs de l'OSN) ou d'autres utilisateurs malicieux. Les procédés de partage de secret existants permettant de partager de multiples secrets ne satisfont pas les contraintes fixées pour cette application, car par exemple, ils ne protègent pas le nombre d'attributs inclus dans le profile, *i.e.* le nombre de secrets partagés par un utilisateur.

Nous proposons un nouveau procédé de partage de secret en couches qui a les propriétés voulus. Nous analysons ensuite la sécurité de ce procédé contre des attaquants curieux mais honnêtes qui, après avoir reçus un ensemble de parts en couches, essayent de reconstruire un plus grand nombre de secrets que ceux auxquels ils ont légitimement accès. Ensuite, nous analysons le niveau de protection fourni par le procédé de partage

de secret en couches dans un OSN; en utilisant les résultats de cette analyse, basés sur l'étude d'un graphe social réel, nous apportons des lignes directrices sur le paramétrage du niveau de sécurité de notre procédé. Finalement, nous avons implémenté notre procédé et l'avons comparé avec une approche naïve consistant à utiliser un simple procédé de partage de secret pour chaque secret. La comparaison se porte sur les temps de calculs et le coût de communication. Nous avons montré qu'une meilleure sécurité a un coût additionnel en terme de compléxité. Cependant nous avons aussi montré qu'en sélectionnant avec soin les paramètres du procédé en couches (*ie.* en utilisant un ramp scheme comme primitive lorsque de lourds secrets sont partagés), noter procédé est utilisable par une application de partage de profiles, car les temps de génération et de décodage des parts en couches sont de l'ordre de la seconde.

Strong Ramp Schemes et Codes MDS Systématiques

Dans le Chapitre 5, nous avons étudié les liens entre les ramp schemes, les strong ramps schemes, les codes à effacements MDS et les codes à effacements MDS systématiques. Le principal aboutissement théorique de ce travail était de prouver que les redondances générées par un code MDS systématique vérifient les propriétés d'un strong ramp scheme. Nous avons implémenté et intégré deux strong ramp schemes dans la structure MPC de SEPIA, le premier procédé étant un dérivé du procédé de Shamir, et le second utilisant le codes à effacements Reed-Solomon. Nous avons évalué expérimentalement les différentes implémentations des strong ramps schemes en les comparant avec le procédé de Shamir (originellement implémenté dans SEPIA) dans un scénario de détection de pannes de réseau. Les résultats montrent que, premièrement, les strong ramp schemes fournissent un compromis réaliste entre la parfaite sécurité offerte par le procédé de Shamir et le temps de calcul nécessaire pour effectuer l'opération distribué voulue (étant ici une union de multiple ensembles). Nous avons aussi montré que le coût de communication du protocole MPC est réduit en utilisant des strong ramp schemes.

Futurs Travaux

Les recherches présentées dans cette thèse peuvent être étendues de nombreuses façons:

- Nous avons montré dans le Chapitre 5 qu'un strong ramp scheme peut être construit à partir de n'importe quel code MDS systématique. Nous avons utilisé le code Reed-Solomon basé sur les matrices de Vandermonde sur un corps de Galois binaire dans nos expérimentations. Le code que nous avons utilisé assume un scénario dans lequel les pertes ne sont pas prévisibles. Cependant, dans un contexte MPC comme celui mentionné dans le Chapitre 5, le nombre de participants et les connections entre ceux-ci sont statiques. Cela pourrait permettre aux participants de pré-calculer certains éléments nécessaires pour créer et/ou décoder les parts. Par conséquent, optimiser l'implémentation des codes MDS pour un tel scénario pourrait encore améliorer les performances de ces codes.

- Nous avons pris pour exemples d'applications de nos procédés de partage de secret des réseaux sociaux et des scénarios de réseaux. Un autre domaine prometteur pour de futures recherches serait celui du stockage distribué de données. Dans ce domaine, un certains nombre de travaux combinent cryptographie et codes à effacements, par exemple [88], [91] [63]. Dans le Chapitre 5, nous avons montré que les redondances d'un code MDS systématique peuvent être utilisées telles quelles comme un strong ramp scheme. Par conséquent la suite logique serait d'explorer la possibilité de supprimer la composante cryptographique utilisée dans les travaux cités précédemment, et n'utiliser que le strong ramp scheme pour stocker de façon sécurisée et distribuée des données. De futurs travaux pourraient examiner l'utilisation de procédés de partage de secret dans ce domaine et les possibles gains que cela apporterait.

- Dans le Chapitre 3, nous abordons le sujet de l'efficacité de nouveaux codes convolutionnels. En considérant les liens existants entre les codes en blocs et les procédés de partage de secret vu dans le Chapitre 5, une possibilité pour de

futures recherches seraient d'étendre les procédés de partage de secret dans la même direction que les codes en blocs ont évolué vers les codes convolutionels, puis vers les codes à fenêtre d'encodage élastique comme dans les schémas de codage à la volée.

Chapter 1

Introduction

1.1 Privacy Issues in Real-Time Sharing of Personal Data: Background and Motivation

There has been an explosion in the growth of shared personal data, with use of services like Online Social Networks (OSNs), as represented by Facebook, YouTube and a large number of related services. Such services have become hugely popular, with an estimated 1.3 Billion users for the largest OSN, Facebook[1]. Similarly, more than 1 Billion unique users visit YouTube each month[2].

OSN users are sharing both their personal information and content that relates to them as individuals. General information is commonly included in OSN user profiles, e.g., their name, contact details like place of residence, school, where they work, etc. Other information includes updates of their activities, with both photos and videos featuring prominently as preferred content. E.g., the average number of photos uploaded per user in Facebook is 205[1]. Around 100 hours of video are uploaded to YouTube every minute[3]. In line with data sharing, system users are both looking for and receiv-

[1]Facebook statistics as of January 2014, from http://www.statisticbrain.com/facebook-statistics/

[2]YouTube statistics as of March 2014, from http://www.youtube.com/yt/press/statistics.html

[3]Although YouTube official statistics does not differentiate between user generated and other content, earlier reports refer to such content making the majority of YouTube videos

ing content and information updates that were uploaded by people of interest (e.g., their friends).

OSN services are increasingly accessed via mobile devices, with a reported 52 % of all Facebook users (680) Million accessing the service from these platforms. Similar trend can be noted for video sharing, with mobile viewing making up almost 40% of total YouTube's global watch time.

Figure 1.1 shows the OSN system and various users (that may belong to a specific group, e.g., a group of friends). It also shows parties external to the system, that may attempt to access the information shared by a selected user.

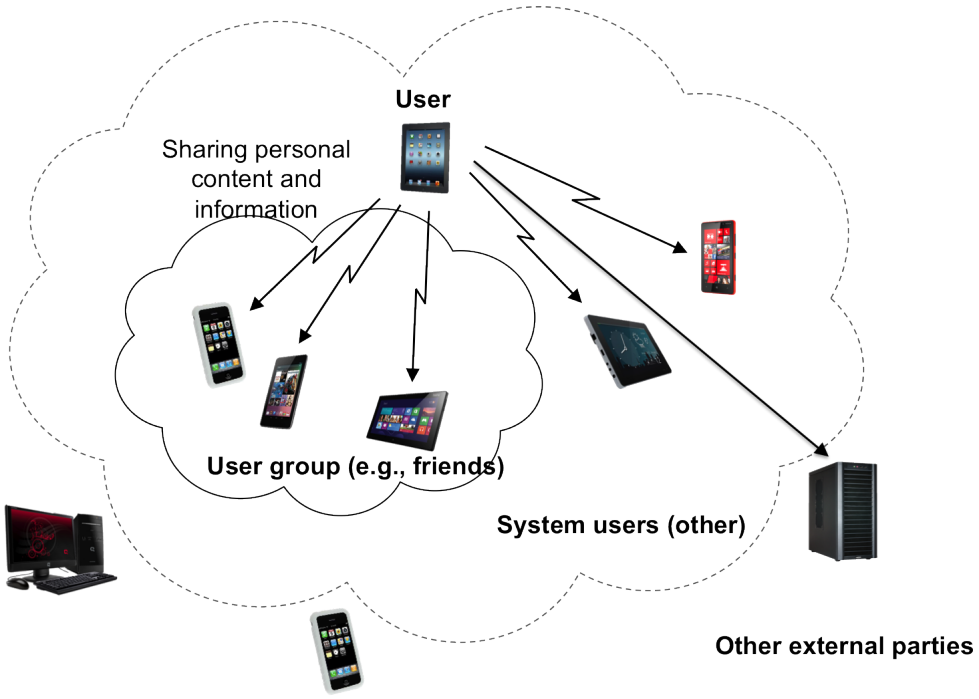


Figure 1.1: System for sharing personal information and content with user groups. Other system users and/or external parties may also have access, directly or indirectly, to this personal information.

Privacy issues arise when user's personal information and shared content (e.g. social or family photos and videos) is accessed either by other users of the system, that were not the intended recipients, or by external parties. In most cases, the system

itself accesses and uses this information to monetize their service, by analysing user characteristics and providing suitable targeted advertisements (either directly or via third parties). There are a number of reasons why this has become prevalent. The service itself is free for the users and has to look for ways to monetize to support the growth and provide support. Second, it is difficult for users to control their data use, as the only action available to them may be to move to an alternative service provider. However, moving is difficult as not only individual users but also their network of friends would need to change the service. Considering the high level of interconnections between users, this becomes impractical. Then, privacy settings regulating access to information by other users of the same system (or external visibility of information) are often complex and require a high level of manual intervention [20]. This necessitates the development of additional mechanisms to protect the privacy of user's data.

Considering data transmission, the use of mobile communications for both upload of personal content to OSN servers and download of such content of interest to specific users, brings a set of challenges traditionally related to wireless transmission over a lossy channel. This is particularly relevant for information accessed in real or near real-time, which is the case for e.g., video content. We note that in this thesis we use the term real time (and near-real time) in a broader sense than what is commonly referred to by the streaming or conversational traffic classes defined by 3GPP [35]. With real time, we refer to any type of traffic or service that has a deadline for the arrival of associated data.

We base our research work on the related solutions to different problems of privacy and data transmission integrity, focusing on Maximum Distance Separable (MDS) erasure codes. These codes have traditionally been used to combat impairments introduced by wireless links [65]. Sharing of information in a privacy preserving way is commonly achieved using secret sharing [97]. We consider the use of MDS codes for secret sharing to improve the low efficiency of such mechanisms. Using the same mechanism for a dual purpose and in two different domains will both enable richer and

more diverse solutions in each individual domain and motivate further development of such mechanisms.

1.2 Research Questions

In this thesis, we address the following research questions that relate to sharing of private and real-time content and information using mobile devices.

First, we focus specifically on transmission of real time content shared by users over wireless links, to mobile receivers. We question the practicality of advanced erasure coding mechanisms (based on MDS codes) that have been recently proposed for such links. Our aim is to determine whether such coding schemes are practical in an environment consisting of mobile devices (smartphones, tablets, etc.) that have lower processing capabilities than fixed devices and limited resources e.g. memory.

The second question relates to privacy of shared personal data in OSN services. We specifically focus on the common case where OSN users are sharing multiple instances of personal information. We consider how it may be possible to share a number of personal files or attributes, without revealing anything about the shared data, including the number of instances or their size - both of which may be used in privacy attacks.

Finally, we question the practicality of enabling both privacy preserving sharing of personal data and secure computations on such data, when the data includes large files, e.g. images or when there is a large volume of shared data. The latter is a common scenario in OSNs, when a new set of connections is given access to specific user's data. The most popular and commonly used secret sharing mechanism, Shamir secret sharing, is highly inefficient in regards to communications overhead. This mechanism generates shares (based on the secrets, i.e., specific shared data) that are of the same size as the original secret. Our interest is in practical solutions that would enable efficient sharing and private computations of such data.

1.3 Thesis Outline

The remainder of the thesis is organised as follows. Chapter 2 provides a background and an overview of prior research work in the relevant areas of MDS codes, secret sharing and secure Multi-Party Computation (MPC). Chapter 3 presents our analysis of the practicality of on-the-fly codes for real time applications on mobile devices. Chapter 4 includes a proposal for a novel multi-secret sharing scheme (based on layering of shares) and evaluation of it's performance in a OSN scenario. Chapter 5 brings together the fields of erasure coding and secret sharing (and MPC) and presents a novel construction method for efficient secret sharing schemes based on MDS codes. We conclude and outline areas for future work in Chapter 6. A related work done during the course of study, but not directly part of the main contributions, is presented in the Appendix A.

1.4 Contributions

The major contributions of this thesis are as follows:

- A new class of erasure codes for delay-constraint applications, on-the-fly codes, was recently introduced. Previous research has established that these codes offer improvements in terms of recovery delay and achievable capacity. Despite their promising characteristics, little is known about the complexity of the systematic and non-systematic variants of this code, notably for live transmission of multimedia content. Our paper aims to fill this gap and targets specifically the metrics relevant to mobile receivers with limited resources: buffer size requirements and computation complexity of the receiver. As our contribution, we evaluate both code variants on uniform and bursty erasure channels. Results obtained are unequivocal and demonstrate that the systematic codes outperform the non-systematic ones, in terms of both the buffer occupancy and computation overhead. This is presented in Chapter 3.

- We propose a novel Layered secret sharing scheme and its application to OSNs. In current, commercially offered OSNs, access to users profile information is managed by the service provider e.g. Facebook or Google+, based on the user defined privacy settings. A limited set of rules such as those governing the creation of groups of friends as defined by the user (e.g. circles, friend groups or lists) allow the users to define different levels of privacy, however they are arguably complex and rely on a trusted third party (the service provider) to ensure compliance. The proposed scheme enables automated profile sharing in OSN groups with fine grained privacy control, via a multi-secret sharing scheme comprising layered shares, created from users profile attributes (multiple secrets), that are distributed to group members; with no reliance on a trusted third party. The scheme can be implemented via e.g. a browser plugin, enabling automation of all operations for OSN users. We study the security of the scheme against attacks aiming to acquire knowledge about users profile. We also provide a theoretical analysis of the resulting level of protection for specific (privacy sensitive) attributes of the profile. Chapter 4 presents this contribution.

- Secret sharing is an important primitive in many protocols for MPC, and Shamir secret sharing scheme is one of the most commonly used schemes in MPC. However, Shamir scheme introduces a significant communication overhead, which might limit its applicability to MPC involving large volumes of data. Strong ramp schemes, which have been proposed to alleviate the overhead issue, present a compromise between efficiency and the level of security. However, there are only a few known constructions for strong ramp schemes, and, to the best of our knowledge, no empirical evaluation of their performance in MPC has been conducted so far. In this work we propose a novel construction method for strong ramp schemes, by demonstrating that the shares of a strong ramp scheme can be directly extracted from the encoded packets of a systematic MDS code. This construction allows to leverage a large number of existing efficient implementations

of MDS codes towards secret sharing and MPC applications. We also propose an additional construction method based on Shamir secret sharing. Finally, we evaluate the performance benefits of strong ramp schemes in MPC by implementing two of these schemes in the SEPIA MPC framework[4], and comparing them with Shamir secret sharing scheme. We show that in a network outage monitoring scenario with 20 input peers and 20 privacy peers, the processing time is reduced by around 44 times, and the communication overhead is reduced by 20 times, compared to MPC using Shamir scheme. This is presented in Chapter 5.

1.4.1 List of Publications

- G. Smith, J. Lacan, E. Lochin, R. Boreli, "Memory and Complexity Analysis of On-the-Fly Coding Schemes for Multimedia Multicast Communications", *IEEE ICC 2012*, 2012
- G. Smith, P-U. Tournoux, R. Boreli, J. Lacan, E. Lochin, "On the Limit of Fountain MDC Codes for Video Peer-To-Peer Networks", *IEEE WoWMoM Workshop on Video Everywhere*, 2012
- G. Smith, R. Boreli, M-A Kaafar, "A Layered Secret Sharing Scheme for Automated Profile Sharing in OSN Groups", *10th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (MOBIQUITOUS)*, December 2013
- G. Smith, R. Boreli, M-A Kaafar, "A Layered Secret Sharing Scheme and its Application to Online Social Networks", *Technical Report*, August 2013
- G. Sarwar, R. Boreli, E. Lochin, A. Mifdaoui, G. Smith, "Mitigating Receivers Buffer Blocking by Delay Aware Packet Scheduling in Multipath Data Transfer", *The 3rd International Workshop on Protocols and Applications with Multi-Homing Support (PAMS 2013)*, 2013

[4] http://sepia.ee.ethz.ch/

Chapter 2

Background and Related Work

In this chapter we describe the background and the context of this thesis. We begin by introducing Galois fields, some probability and information theory results, which are the mathematical background that most of the following work relies on. We then present the erasure codes and introduce two sub-categories of such codes: MDS and on-the-fly erasure codes. We then present the second major topic of this thesis, secret sharing, including the definitions of schemes designed to share either single or multiple secrets and related work in this area. Finally, we present an overview of research work that links MDS and a class of secret sharing schemes, ramp schemes.

2.1 Mathematical Background

2.1.1 Galois Fields

Galois fields are widely used in many areas of computer science, as they define arithmetic operations over a finite number of elements [45].

Definition 1. *In abstract algebra, a field is a non-empty set of elements* $\mathbb{F}$ *that allows addition* $+_{\mathbb{F}}$ *and multiplication* $\times_{\mathbb{F}}$ *operations, satisfying the following properties:*

Closure of $\mathbb{F}$ **under addition and multiplication** *for all* a *and* b *in* $\mathbb{F}$, $a +_{\mathbb{F}} b$ *and* $a \times_{\mathbb{F}} b$ *are in* $\mathbb{F}$.

Associativity of the addition and the multiplication *for all* a, b *and* c *in* $\mathbb{F}$, $(a +_{\mathbb{F}} b) +_{\mathbb{F}} c = a +_{\mathbb{F}} (b +_{\mathbb{F}} c)$ *and* $(a \times_{\mathbb{F}} b) \times_{\mathbb{F}} c = a \times_{\mathbb{F}} (b \times_{\mathbb{F}} c)$.

Commutativity of the addition and the multiplication *for all* a *and* b *in* $\mathbb{F}$, $a +_{\mathbb{F}} b = b +_{\mathbb{F}} a$ *and* $a \times_{\mathbb{F}} b = b \times_{\mathbb{F}} a$.

Existence of additive and multiplicative identity elements *there exists an element* $0_{\mathbb{F}} \in \mathbb{F}$ *such that for all* $a \in \mathbb{F}$, $0_{\mathbb{F}} +_{\mathbb{F}} a = a$ *and there exists* $1_{\mathbb{F}} \in \mathbb{F}/\{0_{\mathbb{F}}\}$ *such that for all* $a \in \mathbb{F}$, $a \times_{\mathbb{F}} 1_{\mathbb{F}} = a$.

Existence of additive and multiplicative inverse *for all* $a \in \mathbb{F}$, *there exists an element* $-_{\mathbb{F}}a \in \mathbb{F}$ *such that* $a +_{\mathbb{F}} (-_{\mathbb{F}}a) = 0_{\mathbb{F}}$, *and for all* $a \in \mathbb{F}/\{0_{\mathbb{F}}\}$, *there exists* $a^{-1} \in \mathbb{F}$ *such that* $a \times_{\mathbb{F}} a^{-1} = 1_{\mathbb{F}}$ *(which define the subtraction and the division).*

Distributive property of the multiplication over the addition *for all* a, b *and* c *in* $\mathbb{F}$, $a \times_{\mathbb{F}} (b +_{\mathbb{F}} c) = a \times_{\mathbb{F}} b +_{\mathbb{F}} a \times_{\mathbb{F}} c$.

While $(\mathbb{Q}, +, \times)$, $(\mathbb{R}, +, \times)$ and $(\mathbb{C}, +, \times)$ are well know examples of fields, they are infinite. Yet infinity is only theoretical in any real systems which are obviously limited in memory. Thus, we will focus on finite field, also referred as *Galois field* named in honour of Évariste Galois. We will denote it $\mathbb{GF}(q)$ where q is the order of the Galois field, *i.e.* the number of elements in the field.

The order of a Galois field is equal to p^m where p is prime and m an integer greater or equal than 1, and all Galois field of the same order are isomorphic. The proofs of these properties are not included in this thesis, as they are unrelated to our work. In the following, we focus on the Galois fields that are commonly used in coding theory and secret sharing.

2.1.1.1 Prime Galois Fields

A Galois field $\mathbb{GF}(q = p^m)$ is referred as a *prime field* if $m = 1$, *i.e.* $q = p$ is a prime number. The field $(\frac{\mathbb{Z}}{p\mathbb{Z}}, +_{\frac{\mathbb{Z}}{p\mathbb{Z}}}, \times_{\frac{\mathbb{Z}}{p\mathbb{Z}}})$ is a prime Galois field of order p, where $\frac{\mathbb{Z}}{p\mathbb{Z}}$ represents

the integers $0, \cdots, p-1$, $+_{\frac{\mathbb{Z}}{p\mathbb{Z}}}$ is the addition modulo p and $\times_{\frac{\mathbb{Z}}{p\mathbb{Z}}}$ is the multiplication modulo p. The additive and multiplicative neutral elements are, respectively, 0 and 1.

2.1.1.2 Binary Field

A Galois field is referred to as a binary field if it's order is equal to 2^m, thus all the elements in the field can be represented by precisely m bits. One way to construct such a field is to use a *polynomial basis representation.*

Let $\mathbb{F}$ be the set of polynomials having a degree lower or equal than $m-1$ with binary coefficients, *i.e.* in the field $\mathbb{GF}(2)$. If $a \in \mathbb{F}$, $a = \sum_{i=0}^{m-1} a_i X^i$, where $a_0, \cdots a_{m-1}$ are binary values. Let $+_{\mathbb{F}}$ be the polynomial addition: if $a = \sum_{i=0}^{m-1} a_i X^i$ and $b = \sum_{i=0}^{m-1} b_i X^i$, $a +_{\mathbb{F}} b = \sum_{i=0}^{m-1} (a_i + b_i) X^i$, where $+$ is the addition modulo 2, which is equivalent to a binary XOR. Thus $+_{\mathbb{F}}$ is the bitwise XOR of two strings of m bits. To define multiplication, first an *irreducible* polynomial P of degree m with binary coefficient is chosen. We note that a polynomial is irreducible if it cannot be factored into a product of two or more non-trivial polynomials. The multiplication on $\mathbb{F}$ is then define as the polynomial multiplication modulo P. Such a definition of $\mathbb{GF}(2^m) = (\mathbb{F}, +_{\mathbb{F}}, \times_{\mathbb{F}})$ is a Galois field of order 2^m.

For example, using the same notation for the Galois field and the set of elements, $(\mathbb{GF}(2^4), +_{\mathbb{GF}(2^4)}, \times_{\mathbb{GF}(2^4)})$ defines with $P = x^4 + x + 1$ is a Galois field of order 16. Let $a = x^3 + x^2 + 1$ and $b = x^2 + x + 1$ be two elements of $\mathbb{GF}(2^4)$, then:

$$\begin{aligned} a +_{\mathbb{GF}(2^4)} b &= (x^3 + x^2 + 1) +_{\mathbb{GF}(2^4)} (x^2 + x + 1) \\ &= x^3 + x \end{aligned}$$

$$\begin{aligned} a \times_{\mathbb{GF}(2^4)} b &= (x^3 + x^2 + 1) \times_{\mathbb{GF}(2^4)} (x^2 + x + 1) \\ &= (x^3 + x^2 + 1) \times (x^2 + x + 1) \; mod \; x^4 + x + 1 \\ &= x^5 + x^2 + 1 \; mod \; x^4 + x + 1 \\ &= x^2 + 1 \end{aligned}$$

These operations can be implemented in hardware using AND, XOR circuits and shift registers. This has resulted in their popularity for use in error correction code mechanisms, originally used in the physical layer [90].

2.1.1.3 Extended Galois Field

The polynomial representation used in the binary fields can be generalized for any fields $\mathbb{GF}(p^m)$ where p is a prime and $m \geq 1$. Thus, a field of order p^m can be constructed using the set of polynomials of degree at most equal to $m-1$, with coefficients in $\frac{\mathbb{Z}}{p\mathbb{Z}}$.

As previously noted, the operation of addition is realised via the polynomial addition, where the coefficients are summed over $\frac{\mathbb{Z}}{p\mathbb{Z}}$. Similarly, the multiplication operation is realised using the polynomial multiplication modulo an irreducible polynomial of degree m, with coefficients in $\frac{\mathbb{Z}}{p\mathbb{Z}}$.

2.1.1.4 Summary and Discussion

Galois fields, and more precisely prime fields and binary fields are commonly used due to the supported arithmetic operations. The choice of the type of field is commonly driven by the type of data that the codes will be applied to. For example, binary and prime fields are, respectively, better suited for binary and for integer input data types. We note that while the definitions of the operations are related to the type of field, all fields define an addition, a subtraction, a multiplication and a division, thus providing all the necessary tools to perform linear operations. Therefore in the remainder of this thesis we will not specifically refer to the type of field that is used for our proposals. The only exception to this rule is Chapter 5, where, due to the nature of the topic (MPC) we will differentiate between the binary fields and prime fields, as the different field types enable a different set of arithmetic operations in MPC.

2.1.2 Probability and Information Theory

In this thesis, a number of results are related to probability and entropy. The purpose of this section is to remind the reader of well-known definitions and properties that we use in the remainder of this thesis.

2.1.2.1 Probability and Conditional Probability

Let A and B be two random events, $Pr(A)$ and $Pr(B)$ their corresponding probability.

Definition 2 (Independent events)**.** *Two events A and B are independent if and only if*

$$Pr(A, B) = Pr(A)Pr(B) \ ,$$

where $Pr(A, B)$ is the probability of the events A and B happening simultaneously.

Definition 3 (Conditional probability)**.** *The probability of A given the event B is referred as $Pr(A|B)$ and is defined as*

$$Pr(A|B) = \frac{Pr(A, B)}{Pr(B)} \ ,$$

when $Pr(B) > 0$. If $Pr(B) = 0$, the probability of A given B if not formally defined, however it is usually set to 0.

It follows that if two events A and B are independent, $Pr(A|B) = P(A)$. For simplification but without loss of generality, we will consider in the following that all the probabilities of events are greater than 0.

Lemma 1. *Let A, B, C be three events, then*

$$Pr(A, B|C) = Pr(A|B, C)Pr(B|C) \ ,$$

where $Pr(A|B, C)$ is the probability of event A given events B and C.

Proof.

$$\begin{aligned} Pr(A, B|C) &\triangleq \frac{Pr(A, B, C)}{Pr(C)} \\ &= \frac{Pr(A, B, C)}{Pr(B, C)} \times \frac{Pr(B, C)}{Pr(C)} \\ &\triangleq Pr(A|B, C) \times Pr(B|C) \end{aligned} \tag{2.1}$$

□

Corollary 1. *Let $A_1, \cdots, A_n$ and B be $n+1$ events, then by recurrence of Lemma 1,*

$$Pr(A_1, \cdots, A_n|B) = \prod_{i=1}^{n} Pr(A_i|A_{i+1}, \cdots, A_n, B) \ .$$

2.1.2.2 Information Theory

Information theory, founded by Shannon in 1948 [98], focuses on quantification of the amount of information contained in a message. The most relevant aspect of Shannon's work in regards to this thesis is his definition of *entropy*, which quantifies the minimum number of bits needed to transfer a message, which can also be interpreted as its randomness.

Definition 4 (Entropy)**.** *The Shannon entropy function [98] for a random variable A with values from a finite non empty set $\mathbb{F}$, is defined as*

$$H(A) = -\sum_{a \in \mathbb{F}} Pr(A = a) \cdot \log_2(Pr(A = a)) \ .$$

$Pr(A = a)$ is the probability of A having a specific value a from the set $\mathbb{F}$.

Definition 5 (Conditional Entropy)**.** *The entropy of A given B, that is the amount*

of randomness in A knowing the value of B, is defined as

$$\begin{aligned} H(A|B) &= \sum_{b\in\mathbb{F}_{\mathbb{B}}} Pr(B=b)H(A|B=b) \\ &= \sum_{b\in\mathbb{F}_{\mathbb{B}}} Pr(B=b)(-\sum_{a\in\mathbb{F}_{\mathbb{A}}} Pr(A=a|B=b)\cdot\log_2(Pr(A=a|B=b))) \\ &= -\sum_{b\in\mathbb{F}_{\mathbb{B}}}\sum_{a\in\mathbb{F}_{\mathbb{A}}} Pr(A=a,B=b)\cdot\log_2(Pr(A=a|B=b)) \end{aligned} \tag{2.2}$$

Lemma 2. *Let A and B be two random variables, then*

$$H(A,B) = H(B) + H(A|B)$$

Proof.

$$\begin{aligned} H(A,B) &= -\sum_{a\in\mathbb{F}_{\mathbb{A}}}\sum_{b\in\mathbb{F}_{\mathbb{B}}} Pr(A=a,B=b)\cdot\log_2(Pr(A=a,B=b)) \\ &= -\sum_{a\in\mathbb{F}_{\mathbb{A}}}\sum_{b\in\mathbb{F}_{\mathbb{B}}} Pr(A=a,B=b)\cdot(\log_2(Pr(B=b))+\log_2(Pr(A=a|B=b))) \\ &= -\sum_{a\in\mathbb{F}_{\mathbb{A}}}\sum_{b\in\mathbb{F}_{\mathbb{B}}} Pr(A=a,B=b)\cdot\log_2(Pr(A=a|B=b)) \\ &\quad -\sum_{b\in\mathbb{F}_{\mathbb{B}}}\log_2(Pr(B=b))\left(\sum_{a\in\mathbb{F}_{\mathbb{A}}} Pr(A=a,B=b)\right) \\ &= H(A|B) - \sum_{b\in\mathbb{F}_{\mathbb{B}}}\log_2(Pr(B=b))\cdot Pr(B=b) \\ &= H(A|B) + H(B) \end{aligned} \tag{2.3}$$

□

Lemma 3. *If A and B are independent variables,*

$$H(A|B) = H(A)\ ,\ \textit{and}$$

$$H(A,B) = H(A) + H(B)\ .$$

Proof. Let A and B be independent variables, then using Definition 2,

$$\begin{aligned} H(A|B) &= -\sum_{b\in\mathbb{F}_{\mathbb{B}}}\sum_{a\in\mathbb{F}_{\mathbb{A}}} Pr(A=a,B=b)\cdot\log_2(Pr(A=a|B=b)) \\ &= -\sum_{b\in\mathbb{F}_{\mathbb{B}}}\sum_{a\in\mathbb{F}_{\mathbb{A}}} Pr(A=a)Pr(B=b)\log_2(Pr(A=a)) \\ &= -\sum_{a\in\mathbb{F}_{\mathbb{A}}} Pr(A=a)\log_2(Pr(A=a))\left(\sum_{b\in\mathbb{F}_{\mathbb{B}}} Pr(B=b)\right) \\ &= -\sum_{a\in\mathbb{F}_{\mathbb{A}}} Pr(A=a)\log_2(Pr(A=a))\times 1 \\ &= H(A)\ . \end{aligned} \tag{2.4}$$

Having $H(A,B) = H(B) + H(A|B)$, we can thus conclude that $H(A,B) = H(A) + H(B)$. □

Lemma 4. *Let A, B and C be three variables, then:*

$$H(A,B|C) = H(A|B,C) + H(B|C)\ .$$

Proof.

$$\begin{aligned} H(A,B|C) &= H(A,B,C) - H(C) \\ &= H(A|B,C) + H(B,C) - H(C) \\ &= H(A|B,C) + H(B|C) + H(C) - H(C) \\ &= H(A|B,C) + H(B|C) \end{aligned} \tag{2.5}$$

□

2.1.3 Markov Chain

2.1.3.1 Definitions

A Markov chain is defined as a succession of random variables $X_1, \cdots, X_t, \cdots$ with the *Markov property*: given the current known state, the future and the past states are independent [60]. More formally, it means that for any $t > 0$,

$$Pr(X_{t+1} = x | X_1 = x_1, \cdots, X_t = x_t) = Pr(X_{t+1} = x | X_t = x_t) \ .$$

Such process, referred as Markov chain can be represented by a directed graph where each node corresponds to a state of the process, i.e. and element of the set of the possible values of the X_i, and the links between the nodes as the probability to transition from the state x_t to x_{t+1} (the value is equal to $Pr(X_{t+1} = x | X_t = x_t)$).

2.1.3.2 Two State Markov Chains

In the remainder of this thesis, we only use two state Markov chains. Let x_1 and x_2 be the two possible states and X_i be the random process defining the Markov chain. Let p_1 be the probability to go from the state x_1 to x_2 and p_2 from the state x_2 to x_1. The probability to stay in x_1 is thus equal to $1 - p_1$, and to stay in x_2 is equal to $1 - p_2$. More formally: $Pr(X_{i+1} = x_2 | X_i = x_1) = p_1$ and $Pr(X_{i+1} = x_1 | X_i = x_2) = p_2$.

We will derive two results from the above definitions, relevant to the work in the remainder of this thesis: the first one being the average probability to be in the state x_2 when the system is in a steady state (assuming a considerable time period in which the system was functional); the second one is the average time the system was in the state x_2 knowing that the previous state was x_1.

Lemma 5. *The probability that the current state is x_2 when the system is in a steady state (regardless of the starting state) is equal to*

$$\lim_{i \to +\infty} Pr(X_i = x_2) = \frac{p_1}{p_1 + p_2} \ .$$

Proof. For all $i > 1$,

$$\begin{aligned}
Pr(X_{i+1} = x_2) &= Pr(X_{i+1} = x_2 \textit{ and } (X_i = x_1 \textit{ or } X_i = x_2)) \\
&= Pr((X_{i+1} = x_2 \textit{ and } X_i = x_1) \textit{ or } (X_{i+1} = x_2 \textit{ and } X_i = x_2)) \\
&= Pr(X_{i+1} = x_2 \textit{ and } X_i = x_1) + Pr(X_{i+1} = x_2 \textit{ and } X_i = x_2) \\
&= Pr(X_{i+1} = x_2 | X_i = x_1) \times Pr(X_i = x_1) \\
&\quad + Pr(X_{i+1} = x_2 | X_i = x_2) \times Pr(X_i = x_2) \\
&= p_1 \times Pr(X_i = x_1) + (1 - p_2) \times Pr(X_i = x_2) \ .
\end{aligned} \tag{2.6}$$

From we which we can write:

$$\begin{aligned}
\lim_{i \to +\infty} Pr(X_{i+1} = x_2) &= p_1 \times \lim_{i \to +\infty} Pr(X_i = x_1) + (1 - p_2) \times \lim_{i \to +\infty} Pr(X_i = x_2) \\
&= p_1 \times \lim_{i \to +\infty} Pr(X_{i+1} = x_1) + (1 - p_2) \times \lim_{i \to +\infty} Pr(X_{i+1} = x_2) \\
&= p_1 \times (1 - \lim_{i \to +\infty} Pr(X_{i+1} = x_2)) \\
&\quad + (1 - p_2) \times \lim_{i \to +\infty} Pr(X_{i+1} = x_2) \\
&= p_1 + (1 - p_1 - p_2) \times \lim_{i \to +\infty} Pr(X_{i+1} = x_2) \ .
\end{aligned} \tag{2.7}$$

Which proves that

$$\lim_{i \to +\infty} Pr(X_{i+1} = x_2) = \frac{p_1}{p_1 + p_2} \ .$$

□

Lemma 6. *The average length L of a continuous stay in the state x_2 is equal to $\frac{1}{p_2}$.*

Proof. The length of a stay in x_2 is equal to i if knowing that the original state is $X_0 = x_1$ and the following is $X_1 = x_2$, then for all $j = 1, \cdots, i$ $X_j = x_2$ and $X_{i+1} = x_1$.

This event has the probability

$$\begin{aligned} Pr_i &= Pr(X_{i+1} = x_1, X_i = x_2, \cdots X_2 = x_2 | X_1 = x_2, X_0 = x_1) \\ &= Pr(X_{i+1} = x_1 | X_i = x_2) \times Pr(X_i = x_2 | X_{i-1} = x_2) \times \cdots \times Pr(X_2 = x_2 | X_1 = x_2) \\ &= p_2 \times (1 - p_2)^{i-1} \ . \end{aligned} \tag{2.8}$$

Thus on the average,

$$\begin{aligned} L &= \sum_{i=1}^{+\infty} i \times Pr_i \\ &= \sum_{i=1}^{+\infty} i(p_2 \times (1 - p_2)^{i-1}) \\ &= p_2 \times \frac{1}{p_2^2} \\ &= \frac{1}{p_2} \ . \end{aligned} \tag{2.9}$$

□

2.2 Erasure Codes

A communication channel is commonly modelled either as an error channel, in which information can be erroneous, or an erasure channel, where the information received is error free, however with some portion missing [33]. In Figure 2.1, we represent both the binary error channel (left side of the figure) and the binary erasure channel (right side of the figure). We denote by p the probability that an input information unit (e.g. a bit) is incorrectly received; this means that the bit is modified in the error channel and erased in the erasure channel, with a probability p, *i.e.,* the Bit Error or Erasure Rate. The same bit is received correctly with a probability $1 - p$. This representation can be generalized to a packet error/erasure channel, where the input is an information

packet, and p denotes the Packet Error/Erasure Rate (PER). To enable recovery of the originally transmitted information in the presence of such errors or erasures, a common approach is to add redundancy to the information prior to transmission. This is done by applying Forward Error/Erasure Correction (FEC) codes [64].

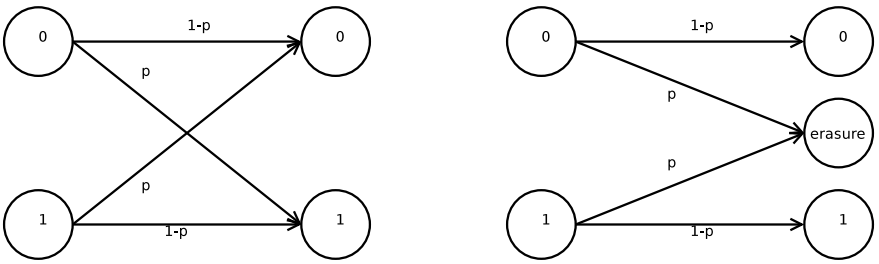


Figure 2.1: Representation of a binary error channel (left hand side) and a binary erasure channel (right hand side) with a probability of error/erasure p

In the remainder of the document, we will primarily focus on the erasure channel, although we note that the two areas are very closely related. Erasures can be encountered for a number of reasons, as for example a buffer overflow in a router (that results in packet losses), a physical connection impairment *e.g.,* cable damage or wireless transmission loss, where the errors have been detected (using a checksum for example [15]) in a known location, *e.g.,* a sequence number. We will also focus on linear codes, *i.e.* codes that rely on linear operations of the input data to construct the encoded data packets. These are commonly defined by their generator matrix, which is multiplied by the vector input to generate the vector output containing the redundancies. These schemes can be classified into two major classes: block and convolutional codes.

2.2.1 Block Codes

The main principle of a block code is to use k source packets to send n encoded packets (with $n \geq k$), that are built from the k packets using the encoding mechanism. In Figure 2.2, we illustrate the construction of a $(4, 6)$ block code: from 4 input packets, 6 packets are generated; this is repeated for each block of 4 input packets. The addition

of $n-k$ repair packets to a block of k source packets at the encoding side allows the decoder, on the receiving side, to rebuild all k source packets, if a maximum of $n-k$ packets are lost from the n packets sent. If more than $(n-k)$ losses occur within any block, decoding becomes impossible, as the coding mechanism is tightly coupled to a specific block size n.

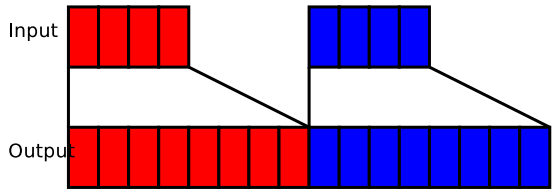


Figure 2.2: Generation of the encoded packets for a $(4, 6)$ block code.

Block codes can be categorised by their characteristics such as their correction capability and based on whether the original input data is directly included (sent) amongst the encoded packets. We define two of these characteristics which are relevant for the remainder of this thesis: the MDS and systematic properties of codes.

Definition 6 (MDS Code)**.** *A block code (k, n, d) is MDS if the minimum Hamming distance d between two codewords amongst the set of possible encoded codewords is equal to $n-k+1$, which is the Singleton bound [99]. MDS code can correct up to $\frac{n-k}{2}$ errors and $n-k$ erasures.*

For a packet based erasure block code, this simply means that from any k encoded packets (amongst the n generated), the k original information packets can be recovered. In this thesis, we will focus on MDS codes. Therefore block codes will be defined by the two parameters (k, n). We note that the size of the corresponding generator matrix for MDS codes is $n \times k$.

Definition 7 (Systematic code)**.** *A (k, n) code is systematic if the k input packets are directly included amongst the n encoded packets.*

The difference between various block codes (*e.g.* Low Density Parity Check (LDPC) [38] or Reed Solomon codes [93]) is related to the specific linear combination method

used to create the redundancy packets. The difference is reflected in both the encoding/decoding complexity (number of additions and multiplications defined in a chosen field, as explained in 2.1.1) and in the correction capability. *E.g.*, the LDPC code commonly uses XOR operations, resulting in a linear complexity [28], however this code does not belong to the MDS family. On the other hand, Reed-Solomon codes [90] are MDS codes, however they have a less efficient implementation than the LDPC block codes: [93] presents implementation details for a Reed-Solomon code, reporting a complexity of $O(nlog^2n)$ on a binary field.

Two examples of generator matrices defining a Reed-Solomon code are the Vandermonde matrix [90] and the Cauchy matrix [10]. Equation 2.10 shows how the k inputs P_i are encoded into the n outputs E_i using a Vandermonde matrix, where the coefficients α_i are all distinct. We note that the Cauchy variant of the Reed-Solomon code encodes the output packets in a similar way, however using the Cauchy matrix in place of the Vandermonde matrix.

$$\begin{pmatrix} 1 & \alpha_1 & \cdots & \alpha_1^{k-1} \\ 1 & \alpha_2 & \cdots & \alpha_2^{k-1} \\ \vdots & & \ddots & \vdots \\ 1 & \alpha_n & \cdots & \alpha_n^{k-1} \end{pmatrix} \cdot \begin{pmatrix} P_1 \\ P_2 \\ \vdots \\ P_k \end{pmatrix} = \begin{pmatrix} E_1 \\ E_2 \\ \vdots \\ E_n \end{pmatrix} \tag{2.10}$$

Block codes were originally used to correct errors or erasures in communication links. Their use has been extended to a number of other application areas, to name but a few: protection of information stored on a Compact Disc [115], distributed storage [63] and secret sharing schemes [29]. In Chapter 5, we propose a construction method for strong ramp scheme (defined later in this chapter, with Definition 10) based on the systematic MDS codes.

2.2.2 Convolutional Codes

Block codes can correct up to $n - k$ erasures in a single block. In a bursty erasure channel [2], erasures are not uniformly distributed over the code blocks (*i.e.* some blocks will incur a high number of losses while others will have no losses). Convolutional coding schemes have been proposed in order to better handle this problem. These codes follow the same concepts as the block codes, but additionally include *memory*. Therefore, convolutional coding schemes are defined by three parameters (k, n, m) as they are usually based on a sliding encoding window of size $k \times m$. We note that parameters k and n are the same as in the block code definition. To encode n packets which will be sent on the network, $k \times m$ previous information packets are used [73]. In Figure 2.3, we represent the generation of the encoded packets of a $(4, 6, 2)$ convolutional code. Each block of 4 input packets is used to generate 12 encoded packets (2 blocks of 6). The encoding window containing the packets used to generate the encoded packets is referred to as the *sliding* window, of a fixed size; an input packet stays in this window for m encoding operations.

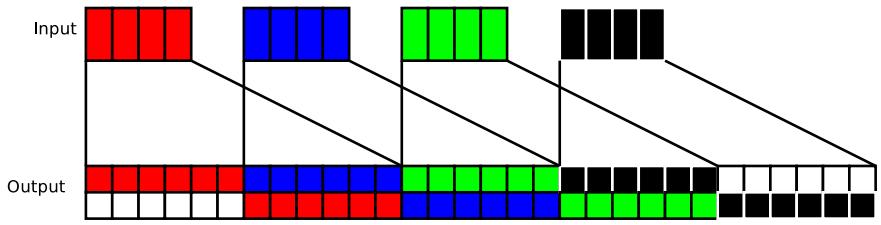


Figure 2.3: Generation of the encoded packets for a $(4, 6, 2)$ convolutional code.

2.2.2.1 On-the-fly Codes

Neither block nor convolutional codes proposed by [73] use acknowledgements, and as a result, they cannot enable full reliability. One possible solution would be to combine Automated Repeat reQuest (ARQ) with such coding mechanisms. ARQ mechanisms allow the receivers to send acknowledgements to the sender to ask for the missing packets, which can be retransmitted. Combining ARQ and erasure codes is known as

Hybrid ARQ [87], however this mechanism may not be practical for time constrained applications on links that have a long delay, as the missing packets may be retransmitted too late for the applications to make use of them. To handle this problem, authors in [101] and [104] propose an on-the-fly code with an infinite encoding memory (referred to as an *elastic encoding window*) and an acknowledgement path, used to decrease the number of packets in the encoding window. We note that the acknowledgement packets are only used, when possible, to decrease the encoding complexity.

In network coding, this approach enables the creation of "infinite" linear combinations of packets [102]. In this context, the purpose of having an infinite window is not to protect the data, but to fully use the network capacity, by sending only useful packets to every receiver. A packet is called useful when it is utilised at the receiver side to retrieve missing packets. In this case, only linear combinations of source data packets are sent i.e. the code is non-systematic. In [102], the authors use the concept of a "seen" packet, which enables the receiver to acknowledge a source data packet P_i when a repair packet, that contains a linear combination including P_i, is received. More precisely, P_i is acknowledged by a repair packet when P_i is the first not yet seen or obtained packet contained in this repair packet. This allows the receivers to acknowledge packets (even) before decoding them, thus enabling the source to reduce the size of the encoding window [101].

In [104], the authors propose to use an on-the-fly code with an infinite encoding window to protect the data. However, the main objective is to enable a fully reliable coding scheme for real-time applications such as voice over IP (VoIP) or streaming video [103] for which they prove the efficiency of such a scheme compared to commonly used block coding schemes. This code is systematic, i.e. the source data is included directly in the encoded data.

While bringing a potential for performance improvement of real-time applications compared to the case when block codes are used [103] and having beneficial properties in bursty erasure channels [73], an infinite window size is not practical in real systems

due to resource limitations such as the memory of the system. To the best of our knowledge, prior to our work on this topic, there has been no existing study that quantified the complexity and analysed the buffer size requirements of convolutional codes with an infinite encoding window. In a point to point scenario, the analysis is trivial as the systematic codes would logically produce an improvement in terms of delay. However in a multicast context, it is much more complex to estimate the impact of the multicast group size on each receiver within the group. We investigate this problem in Chapter 3.

2.3 Secret Sharing Schemes

Secret sharing schemes were introduced by both Shamir in [97] and Blakley [8] in 1979. The original motivating problem was to calculate the required number of locks and keys distributed amongst n participants to protect documents, that could be retrieved if t out of the n participants collected their keys. They solve this problem, in computer science, via secret sharing schemes. In a secret sharing scheme, a dealer securely shares a secret (any data) with a group of participants, by first generating n secret-based shares using a cryptographic function. The dealer subsequently distributes these shares to n participants, as shown in Figure 2.4. By aggregating shares, participants can gain access to the secret when the number of combined shares reaches the threshold t. Their methods are using respectively polynomial interpolation [97] and hyperplane projection [8].

Since these early works, this research area has been extensively studied. For example, new constructions of secret sharing schemes were proposed in [78] and [57] using, respectively, the Chinese remainder theorem and XOR operations to achieve the sharing of secrets in a secure way. Theoretical bounds for efficiency of these schemes were also investigated in *e.g.,* [11]. Additional-features were also introduced *e.g.,* the *robustness* of the scheme in [56]. This scheme can correctly recover the secret in the presence of a bounded number of corrupt shares. *Verifiability* of a scheme was first addressed

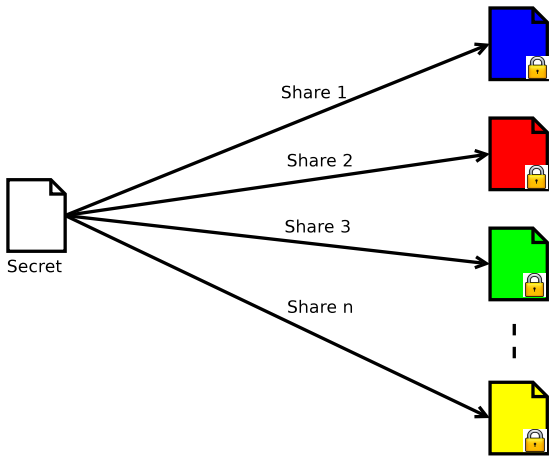


Figure 2.4: Secret sharing mechanism.

in [110], to detect a dishonest dealer. Finally, these schemes have been applied in new applications with specific constraints, such as in [114] for private localisation, and in [108] for key recovery in a distributed OSN.

In this thesis, we primarily focus on the most commonly used *threshold secret sharing schemes* that are based on linear operations on a Galois field. In the following sections, we introduce schemes that are designed for sharing, first, a single secret and then multi-secret sharing schemes. We consider the security offered by selected schemes and the communications requirements for the distribution of shares, by comparing the share size to that of the secret.

2.3.1 Single Secret Sharing

2.3.1.1 Threshold Secret Sharing Scheme

In a threshold secret sharing scheme, a dealer securely shares a secret S with a group of n participants, by generating n shares $E_1, \cdots, E_n$ from the secret using a specific cryptographic function. By aggregating a subset of t shares (threshold), with $t \leq n$, the original secret can be recovered. If strictly less than t shares are collected by any participant, the secret should remain protected. Formally:

Definition 8 (Threshold Secret Sharing Scheme)**.** *Let S be a secret and $E_1, \cdots, E_n$*

be shares. A (t, n) *threshold secret sharing scheme satisfies the following property, for any set of indices* $i_1, ..., i_x$*, where* x *is the number of shares available to any participant:*

$$H(S|E_{i_1}, \cdots, E_{i_x}) = \begin{cases} H(S), & \text{if } x < t \\ 0, & \text{if } t \leq x \leq n \end{cases} . \tag{2.11}$$

This definition of a threshold secret sharing scheme ensures the correctness of the scheme (the secret is known with more than t shares) and the information-theoretical security (strictly less than t shares do not provide any information about the secret). Threshold scheme with such a security are referred as $perfect$ secret sharing scheme. However it was proven in [12], that to achieve such a security, the size of each share needs to be, at a minimum, equal to the size of the secret.

We note that our focus is on linear schemes [6] (constructed from a linear combination of finite field elements, that can be represented as matrix and vector operations), as linear properties are necessary to enable share-based secure computations.

Shamir scheme is a popular linear threshold scheme in which the secret S and shares $E_1, \cdots, E_n$ are elements of a selected Galois field $\mathbb{GF}$. For a chosen threshold t, the dealer constructs a polynomial $P(X)$ of degree $t-1$:

$$P(X) = S + \sum_{i=1}^{t-1} r_i X^i \ ,$$

where the coefficients $r_1, ...r_{t-1}$ are chosen randomly from $\mathbb{GF}$ and $S = P(0)$. Shares $E_1, \cdots, E_n$ are then constructed as values of $P(x)$, for n chosen values, $x_1, ..., x_n$. The secret can be reconstructed by any participant who is in possession of t or more shares. This is accomplished by using the Lagrange interpolation polynomials, with

$$S = P(0) = \sum_{j=1}^{t} P(x_{i_j}) \prod_{l \neq j} \frac{-x_{i_l}}{x_{i_j} - x_{i_l}} \quad .$$

We note that *computationaly* secure secret sharing schemes have been introduced

to enable the reduction of share sizes, however also resulting in a lower security. Such schemes are based on the assumption that the discrete logarithm problem on a prime Galois field $\mathbb{GF}(p)$ with a large p has no efficient solution [32]. An example of such a scheme is the Information Dispersal Algorithm (IDA) proposed by Rabin [88] in which a large secret is first encrypted with a key. The cipher is then shared amongst n participants using an erasure code while the key is distributed using Shamir's scheme. MDS erasure codes being optimal, the overhead introduced by the secret sharing scheme is negligible if the file is large compared to the size of the key. However computational security is not easily quantifiable. *Ramp scheme* were thus introduced to reduce the size of the shares while maintaining an entropy definition of it's security.

2.3.1.2 Ramp schemes

Blakley introduced the concept of *ramp schemes* [9], that enable more efficient secret sharing in regards to the size of the shares. A (t, L, n) ramp scheme includes an additional parameter L (with $L \leq t$), that provides a differentiation between the security guarantee and the number of shares required to reconstruct the secret, t. In ramp schemes, having more than $t - L$ (but less than t) available shares will leak some information about the secret in a controlled way [84]. It was shown in [52] that the size of a share is lower bounded by the size of the secret divided by L. When the share size is equal to the lower bound, it the ramp scheme is *optimal*. Researcher also show, in [52], that to have an optimal ramp scheme and to maximise the security of the scheme, the ramp scheme needs to be *linear*. Thus in the remainder of this thesis, we will focus our work on linear ramp schemes that are also proven to be optimal. Formally:

Definition 9 (Linear Ramp Secret Sharing Scheme)**.** *Let S be a secret and $E_1, \cdots, E_n$ shares. A (t, L, n) linear ramp secret sharing scheme satisfies the following properties, for any set of indices $i_1, ..., i_x$, where x is the number of shares available to any parti-*

cipant:

$$H(S|E_{i_1},...,E_{i_x}) = \begin{cases} H(S), & \text{if } x < t-L \\ \frac{t-x}{L}H(S), & \text{if } t-L \leq x < t \\ 0, & \text{if } t \leq x \leq n \end{cases}$$

First, we note the linearity defined in [52] is not related to the operations used to generate the shares but to the linearity of the entropy when x shares are received with $t - L \leq x < t$. However as outlined in Section 2.3.1.1, we are focusing on schemes that are based on linear operations. Then, we note that a ramp scheme with $L = 1$ becomes a threshold (t, n) secret sharing scheme.

To implement such schemes using linear operations, the secret is represented as a vector and divided into L elements $\vec{S} = (S_1, ..., S_L)$. We assume that the elements S_i are (mutually) linearly independent and belong to a common Galois field $\mathbb{GF}$.

We note that the definition of ramp schemes provides a bound on the level of information leakage for the secret as a whole, rather than for each element of the secret vector $\vec{S}$.

Shamir scheme can be utilised in two different ways to construct a ramp scheme. These construction methods are related to the means of defining the polynomial P.

Polynomial Defined by it's Coefficients [29]: Let $r_{L+1}, \cdots, r_t$ be $t-L$ random values in $\mathbb{GF}$. $P(X)$ is defined as follows:

$$P(X) = \sum_{i=0}^{L-1} S_{i+1}X^i + \sum_{i=L}^{t-1} r_{i+1}X^i \ ,$$

where the L first coefficients being the element of the secret and the remaining ones are chosen randomly.

For this construction, the shares are defined similarly to the Shamir scheme (*i.e.* by points of the polynomial) and the secrets can be reconstructed from any t shares, as the polynomial P is uniquely defined by t of its points. While the reconstruction process is usually not described, we can represent the problem as a linear system with

t equations (the t shares) with t unknowns (the polynomial's coefficients). The system is non-singular as it can be represented as the product of a Vandermonde matrix with the unknown vector:

$$\begin{pmatrix} 1 & x_{i_1} & \cdots & x_{i_1}^{t-1} \\ 1 & x_{i_2} & \cdots & x_{i_2}^{t-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{i_{t-1}} & \cdots & x_{i_{t-1}}^{t-1} \end{pmatrix} \times \begin{pmatrix} S_1 \\ \vdots \\ S_L \\ r_{L+1} \\ \vdots \\ r_t \end{pmatrix} = \begin{pmatrix} P(x_{i_1}) \\ P(x_{i_2}) \\ \vdots \\ P(x_{i_{t-1}}) \end{pmatrix}.$$

A Vandermonde matrix is non-singular in any Galois field as long as all the x_i are distinct, thus:

$$\begin{pmatrix} S_1 \\ \vdots \\ S_L \\ r_{L+1} \\ \vdots \\ r_t \end{pmatrix} = \begin{pmatrix} 1 & x_{i_1} & \cdots & x_{i_1}^{t-1} \\ 1 & x_{i_2} & \cdots & x_{i_2}^{t-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{i_{t-1}} & \cdots & x_{i_{t-1}}^{t-1} \end{pmatrix}^{-1} \times \begin{pmatrix} P(x_{i_1}) \\ P(x_{i_2}) \\ \vdots \\ P(x_{i_{t-1}}) \end{pmatrix}$$

This proves the correctness of the scheme. The two first conditions of a ramp scheme results from consideration of the rank of a linear system, when strictly less than t shares are received. This is equivalent to solving a system of linear equations that has the number of equations lower than the number of unknown variables.

Polynomial Defined by t Points [9]: Let $x_1, \cdots, x_{t+n}$ be $t+n$ values in $\mathbb{GF}$ distinct from each other. Let $r_{L+1}, \cdots, r_t$ be $t-L$ random values in the same Galois field. We can define P of degree t as follows: for all i in $1, \cdots, L$, $P(x_i) = S_i$ and for all i in $L+1, \cdots, t$, $P(x_i) = r_i$. The shares can the be generated from these points

using Lagrange polynomial interpolations: for all j in $t+1, \cdots, t+n$,

$$P(x_j) = \sum_{i=1}^{t} P(x_i) \prod_{l \neq i} \frac{x_j - x_l}{x_i - x_l} \ .$$

As described previously, to decode the secret, the polynomial P is interpolated at the points $x_1, \cdots, x_L$ after the t shares are available. Franklin in [37] proposes a different construction of the polynomial, by defining a random polynomial $Q(X)$ of degree $t - L$ and building P such as

$$P(X) = Q(X) \prod_{i=1}^{L} (X - x_i) + \sum_{i=1}^{L} S_i \prod_{l \neq i} \frac{X - x_j}{x_i - x_j} \ .$$

This ensures that $P(x_i) = S_i$ for $i = 1, \cdots, L$ and that P is of degree $t-1$. While this construction of the polynomial is slightly different, we note that both constructions are equivalent.

2.3.1.3 Strong Ramp Schemes

Although the definition of a ramp scheme also theoretically defines the level of protection for the secret, provided by the scheme, this secret is only protected as a whole. We note that a part of the secret can be leaked by a scheme, while still satisfying the ramp scheme definition. As a simple example, if we consider a $(t, L = t, n)$ ramp scheme, the dealer could share (directly) any single element of $\vec{S}$ and still satisfy the condition qualifying this as a ramp scheme in which $L = t$. E.g. if S_1 was shared, and assuming equal entropy for all elements in $\vec{S}$, $H(\vec{S}|S_1) = \frac{t-1}{t} H(\vec{S})$.

To address this deficiency, Yamamoto [116] introduced *strong* ramp schemes, that provide a stronger security guarantee:

Definition 10 (Strong Ramp Scheme)**.** *Let $\vec{S} = (S_1, \cdots, S_L)$ be a vector secret and $E_1, \cdots, E_n$ shares. A (t, L, n) linear strong ramp scheme satisfies the following properties, for any set of indices $i_1, ..., i_x$, where x is the number of shares available to any*

participant:

$$H(\vec{S}|E_{i_1},...,E_{i_x}) = \begin{cases} H(\vec{S}), \textit{if } x < t-L \\ H(S_{j_1},\cdots,S_{j_{t-x}}|E_{i_1},...,E_{i_x}) = \frac{t-x}{L}H(\vec{S}), \\ \qquad \textit{if } t-L \leq x < t \textit{ for any } S_{j_1},\cdots,S_{j_{t-x}} \\ 0, \textit{if } t \leq x \leq n \end{cases}$$

The definition of a strong ramp scheme ensures the entropy of any subset of secret (vector) elements, rather than just the secret as a whole, is bounded to a specific value of $\frac{t-x}{L}H(\vec{S})$, which maximizes the ambiguity provided by such a scheme.

The Lack of Practical Solutions for Strong Ramp Schemes The work on strong ramp schemes has been predominantly of theoretical nature. In [116], Yamamoto presents a theoretical construction and outlines the conditions that a matrix needs to satisfy, to generate shares for a strong ramp scheme. [52] provides theoretical bounds on the size of the shares generated for a strong ramp scheme based on an entropy study. In [51], a theoretical construction of strong ramp scheme for a general access structure from a subcategory of ramp scheme (referred as partially decryptable) is explained. The first and only work with a practical construction of a strong ramp scheme that we are aware of is [4]. Such a strong ramp scheme is built from matrix projections operations but in this article, the proof of such a scheme being a strong ramp scheme has been omitted, and the remaining security proofs are light.

With the exception of these limited number of works, strong ramp scheme has not attracted the attention of the research community. In contrast to this, the use of the ramp schemes is becoming more popular, mainly due to the potential benefits of performing, in a private way, aggregated operations in cloud systems based on MPC ([23], [25]). The security weaknesses of ramp schemes could map to being the weak points of future systems. This, however, could be solved by using strong ramp schemes that have stronger security properties.

2.3.1.4 On the Links Between Secret Sharing Schemes and MDS Codes

MDS erasure block codes and secret sharing schemes are based on the same theoretical foundations and use the same Galois field based computations, to achieve their goals. More specifically, an MDS code and a secret sharing scheme have a common objective, which is to be able to decode the message (secret) when exactly k (or t, in the common secret sharing notation) amongst n packets (shares) are received. However, secret sharing adds a security constraint when a lower number of shares are received.

A number of prior research works have, in some way, addressed or used the similarities between these schemes and more generally between erasure codes and secret sharing mechanisms. In [76], the author noted the similarity between Shamir scheme and a Reed-Solomon code based on the Vandermonde matrix. Massey in [74] and [75] reduces the problem of constructing a perfect secret sharing scheme with a specific access structure to the problem of constructing an MDS block code, with the codewords defined by the access structure of the desired secret sharing scheme (we note that both problems are difficult to solve). Further reserch results extending these works have been reported in [18], [120] and [62], providing for example selected secret sharing schemes with a specific access structures, based on MDS codes. Further to this, the authors in [85] also construct a threshold secret sharing scheme from an MDS code, but incorporate the detection of incorrect shares and the identification of the cheating participants. However, all these works explore the links between MDS codes and perfect secret sharing schemes, that are not ramp schemes. As noted in Section 2.3.1.2, such schemes do not reduce the size of the shares when compared to Shamir scheme.

Focusing on ramp schemes, these schemes have been linked to erasure codes in a smaller number of research works, for example, [29] and [84]. However in [29], the authors limit their their study to a $(t, L = t, n)$ ramp scheme, and in [84], non-MDS codes are linked to ramp schemes. Finally, in [22] and [25], erasure codes and ramp schemes are considered for MPC applications.

To the best of our knowledge, there has been no prior published work on links

between strong ramp schemes and MDS codes. Furthermore, we are not aware of any practical study implementing ramp schemes (and a fortiori strong ramp schemes), and comparing their efficiencies. In Chapter 5, we will address the links between the strong ramp schemes and the systematic MDS codes. We first prove that a ramp scheme, derived from Shamir scheme where the polynomial is defined by it's points, is a strong ramp scheme. We note that researchers in [51] have previously shown, using an example, that the Shamir-based ramp scheme based on a polynomial defined by it's coefficients is not a strong ramp scheme. We then implement two variants of the strong ramp scheme and present an experimental evaluation of the computation overhead and communication cost of strong ramp schemes, when used in an MPC application.

2.3.2 Sharing Multiple Secrets

Using single secret sharing schemes may be restrictive for application designed to deal with multiple secrets. The most obvious approach of using, multiple times, a single secret sharing scheme may prove onerous both in terms of communications and computational overhead as each participant needs, for each shared secret, to receive and store a corresponding share [12].

The main goal of multi-secret sharing schemes is to reduce the number of private shares participants need to store, while protecting the secrets. A second approach is to use ramp schemes. These schemes can be applied either to a large secret (that is sub-divided into smaller components), or to a number of smaller secrets (*i.e.* if such secrets have values from $\mathbb{GF}$). Using a ramp scheme enables sharing of L secrets simultaneously, with a single share per user. The secrets are then considered as a block: they can either all be decoded, or alternatively the decoding will fail for all secrets in the block.

Another approach is to provide a way for the participants to *construct* the shares relating to the new secrets from a share they keep as private. This class of schemes is commonly referred to as dynamic schemes, as a single private share can be used for

a multitude of secrets, which are not necessarily shared at the same time. This and similar classes of schemes, however, rely on public shares. The first two such proposals were reported in [47] and [46]. The proposals include a public shift technique, to obtain the real shares from public shares and the successive use of a one-way hash function on private shares, to compute the specific shift values, for every secret and every participant. A drawback of this scheme is the limited number of secrets that could be shared (which is the number of successive uses of the hash function). To mitigate this issue, researchers in [48] proposed to use a two-variable one-way function in palce of the hash function, using the private share as the first variable and a public value identifying the secret to be decoded as the second variable. These works have then been extended in [110], [113], [21] where different shift techniques have been used and extended, so that the threshold per secret can be different. Their objective is to reduce the number of necessary public shares and the encoding/decoding computation time.

However all these schemes, while primarily relying on Shamir's scheme to generate shares, have two weak points: the presence of public shares and the use of hash-functions for security (they are used to protect the private shares of the participants).

Further extensions of these schemes have been proposed in for example [117] [83], in which they combine the ramp schemes and the use of two-variable hash functions to enable the sharing of multiple blocks of secrets with a unique private share. However, it must be noted that, in these schemes, the threshold value t is the same for all the shared secrets.

Due to the necessity to use public shares, applications using such schemes could leak some information about the secrets, *e.g.,* their numbers, their sizes and the threshold used for each of them. In Chapter 4, we address with these potential privacy leaks in a OSN profile sharing application, by developing a Layered secret sharing scheme designed for sharing multiple secrets with multiple thresholds, without using public shares.

2.4 Secure Multi-Party Computation

MPC is a mechanism that provides distributed privacy preserving computations in a collaborative environment. Based on input data from multiple parties, and using a set of cryptographic protocols, MPC enables computing of mathematical functions, while providing formal guarantees of the data confidentiality and the correctness of the computation result [24]. MPC were first introduced by Yao [118] to solve the *millionaires problem* in which two millionaires want to know who is the richer without revealing the size of their fortunes.

MPC can be achieved by a number of approaches, e.g. by using garbled circuits [5] or secret sharing [41]. Garbled circuit are mainly for two parties computation, i.e. when only two parties are present. However the most popular MPC method uses secret sharing such as the Shamir scheme [7]. To perform a secure computation, the input peers generate shares of their data, and distribute them to the privacy peers (typically one share per peer). The privacy peers compute the required operation and collaboratively reconstruct the final computation result, which is finally returned to input peers, as described in Figure 2.5.

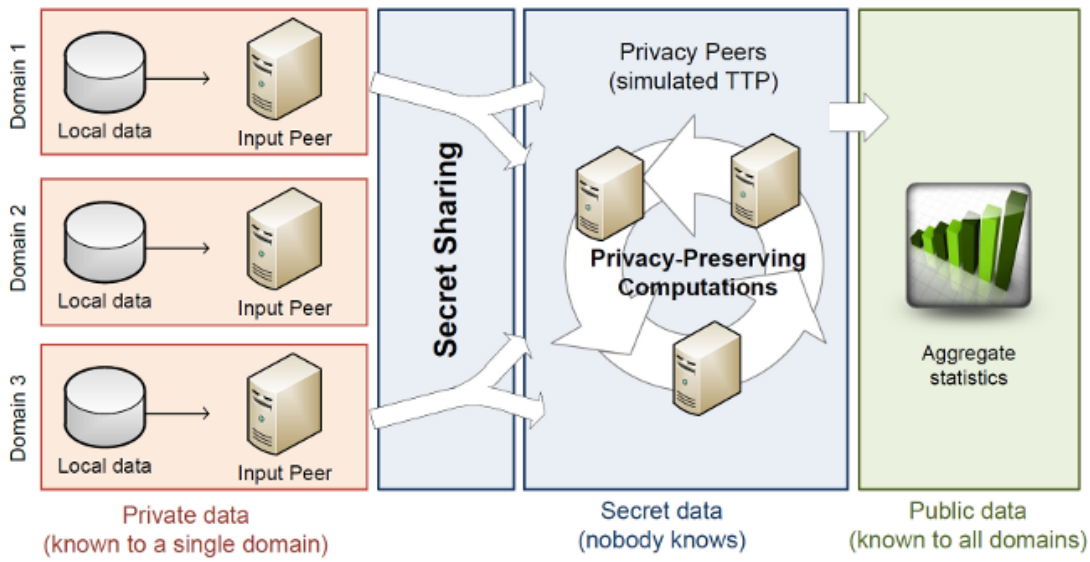


Figure 2.5: Secure multi-party computation (taken from SEPIA website http://sepia.ee.ethz.ch/)

The two most important operations required are the secure addition and multiplication. From both, a large number of additional operations can be computed such as comparisons (*e.g.* less than, equal), set operations (*e.g.* unions, intersections) [72]. [23] proposes to use ramp scheme to compute more efficiently multiple operations in parallel when each user has multiple secrets. However as far as we know, in all the MPC protocols relying on ramp scheme, the ramp scheme are never defined as strong. In [23] for example, Cramer proposes two different types of multiplication on multiple entries: the first being the atomic multiplication of two secret vectors, *i.e.* the element-wise multiplication, the second being the multiplication of two large secrets which need to be divided into smaller elements. We note that each operation is relying on its specific secret sharing scheme. We have implemented and tested a strong ramp scheme in an MPC use-case, to compare its practical efficiency compared to Shamir's scheme. We present these results in Chapter 5.

2.4.1 Security

There are two types of adversaries in MPC and the corresponding two types of attacks [24]. Honest-but-curious MPC parties follow the MPC protocol and will only collude to derive as much information as possible from the shares and to determine the computation result. Corrupt i.e. malicious parties may not follow the protocol and can also alter the shares or the local result of computation, thereby compromising the integrity of the computation result.

The authors in [7] prove the data confidentiality and result integrity guarantees of Shamir secret sharing based MPC protocol under the two attack models. MPC is secure against $t < n/2$ honest-but-curious participants who, while colluding, would not be able to obtain any information about either the data or the intermediate computations results. [7] also shows that MPC is secure against $t < n/3$ malicious participants: they can neither disrupt the computation nor get additional information about the secrets.

2.5 Real-Time Applications

One motivation of this thesis is to apply new schemes and new constructions to real-time systems. However as a real-time system or application is a wide notion, here we provide a definition of what we consider as real time in the context of this thesis.

Definition 11 (Real-time system [77])**.** *Computer systems in which the computer is required to perform its tasks within the time restraints of some process or simultaneously with the system it is assisting.*

Real-time systems or applications are defined by a time constraint, *i.e.* a threshold time under which the process required to be finished. In this thesis, the online applications we are focusing on are related to web access and multimedia such as VoIP, video streaming and accessing an online service *i.e.,* a user profile in an OSN. These applications could be referred to as *soft real-time* for which the usefulness of the result degrades after it's deadline without breaking the system if a deadline is not met.

In a VoIP application, the time constraint is defined by the time needed for a receiver to get the information from the sender who generates real-time content. The Mean-Opinion-Score (MOS) has been modelled to represent the quality of a VoIP application from network measurement [50], in which the maximum acceptable delay is set to 200 ms. For more general use cases, some rules of thumbs about the links between a user experience and time needed to provide the service work across power of 10[1]: under 0.1 s, users consider the system as instantaneous, under 1 s, users are still interacting with the system. Up to 10 s, the users may loose their focus. In [55] (pp. 403), an upper limit of 4 seconds has been considered as an upper limit for having a satisfactory interactive web browsing user experience over 3G.

In this thesis, our interest is to ensure the response time required to have a usable service, from a user's point of view. We will thus consider that delays of the order of magnitude of several seconds (up to 4 seconds, in line with 3G limits for interactive

[1] http://www.nngroup.com/articles/powers-of-10-time-scales-in-ux/

web browsing) as acceptable for OSN profile sharing application described in Chapter 4 and to perform specific MPC operations in Chapter 5.

Chapter 3

On-the-Fly Coding Schemes for Multimedia Multicast Communications

3.1 Introduction

There exist two classes of reliability mechanisms based, respectively, on retransmission and redundancy schemes. ARQ schemes recover all lost packets by utilizing retransmissions. As a consequence, the recovery of a lost packet incurs a delay of at least one additional Round Trip Time (RTT). However, this might not be suitable for time constrained applications, which define a threshold above which they consider a packet outdated and no longer useful to the receiving application. As discussed in Section 2.2, a well-known solution to prevent this additional delay is to add redundancy packets to the data flow. This can be done by using erasure coding schemes.

As mentioned in Section 2.2.2.1, to overcome the limitation of block code in regards to the limitation of losses occurring in a single block, other recent approaches have proposed on-the-fly coding schemes [73][101][104], which belongs to a class of convolutional codes. In [73], the authors use non-binary convolutional codes and show

Table 3.1: Representation of the source encoding window and the sending pattern in both scenario for a code $(3, 4)$. Coefficients in the linear operations are not represented, please note they are chosen to have a Maximum Distance Separable code.

Packet number	Encoding window	Non systematic sending	Systematic sending
P_1	P_1	P_1	P_1
P_2	P_1, P_2	$\sum_1^2 P_i$	P_2
P_3	P_1, P_2, P_3	$2 \times \sum_1^3 P_i$	P_3 and $\sum_1^3 P_i$
P_4	P_1 to P_4	$\sum_1^4 P_i$	P_4
P_5	P_1 to P_5	$\sum_1^5 P_i$	P_5
P_6	P_1 to P_6	$2 \times \sum_1^6 P_i$	P_6 and $\sum_1^6 P_i$

that the decoding delay can be reduced with the use of a sliding window, rather than a block, to generate the repair packets. More recently in [101] and [104], the authors propose an on-the-fly coding scheme that implements an elastic encoding window and uses an unreliable reverse feedback path (when available), to decrease the encoding complexity at the sender side, without impacting the communication data transfer. Compared to [73], both proposals enable a fully reliable service under certain conditions. However, the main difference between [101] and [104] is that the former proposes a non-systematic scheme while the later uses a systematic variant (the difference being illustrated in Table 3.1).

To the best of our knowledge, there is no existing study that quantifies the complexity and analyses the buffer size requirements of convolutional codes with an infinite encoding window. In a point to point scenario, the analysis is trivial as the systematic codes would logically produce an improvement in terms of delay. However in a multicast context, it is much more complex to estimate the impact of the multicast group size on each receiver within the group.

Thus in this chapter, our aim is to assess the benefit and implementation requirements of both variants of on-the-fly coding schemes, in terms of receiver buffer occupancy and computation overhead. The objective is to evaluate the applicability of such coding schemes in the context of multimedia communications over multicast services.

In particular, the resulting analysis would enable us to determine whether such coding schemes are practical in a multicast environment where the multicast group, comprising of mobile devices (e.g. smartphones) which have lower processing capabilities and limited resources, is receiving, e.g. video or any other multimedia content.

We refer to Section 2.2 for the characteristics of on-the-fly coding schemes compared to block codes. Then, we analyse the buffer size requirements of such codes over a uniform erasure channel in Section 3.3 while Section 3.4 addresses their computation complexity. We also present a study of buffer sizes and computation complexity over a bursty erasure channel in Section 3.5. Finally we conclude this chapter in Section 3.6.

3.2 Simulation Scenario and Parameters

We have implemented both version (systematic [101] and non-systematic [104]) elastic window codes in Matlab. We use a satellite-like multicast scenario where the source transmits to a number of independent receivers. We vary the number of receivers between 2 to 30. Although the number of receivers in a multicast group consisting of mobile devices may be significantly larger, we will show that the number of receivers used is sufficient to illustrate the differences between the two codes in terms of memory and complexity, as related to the group size.

Matlab is not a real-time simulator, so it was necessary to define a time scale i.e. a unit of time. During this period, the source may receive an ACK (if any), reduce its encoding window, or send a packet; the receivers may receive a packet, decode the repair packets, reduce their buffer sizes, or send an ACK (if needed). The RTT and the time elapsed between two acknowledgements, s, will consequently be expressed as a multiple of this unit of time.

To simplify the simulation, the matrices used in the encoding and decoding process are not created, therefore avoiding the need for complex operations like matrix inversion. We also assume that the codes used are MDS. As the encoding is based on MDS properties, we only consider that if a matrix is square, it can be inverted and the

decoding is therefore possible.

For our simulations, we always use a code $(3, 4)$ (which can correct up to 25% of erased packets), a packet error rate $PER = 20\%$ and an $RTT = 2$. For simplicity, we consider an identical delay on the uplink and downlink between the source and the receiver, equal to one unit of time. We vary the number of receivers and s. The choice of such code rate and PER is motivated by [103] in which they compare the systematic on-the-fly code with block codes having a similar rate, more precisely, the codes $(3, 4)$, $(6, 8)$, $(9, 12)$ and $(12, 16)$ which are reasonable block lengths for real-time video applications not to have long decoding delays.

We evaluate two cases: a uniform erasure channel and a bursty erasure channel. We consider that the links to the different receivers are independent, i.e. the losses (either bursty or uniform, as appropriate to the channel) are independent on both the uplink and the downlink. We note that for all the figures in this paper, each point in any of the graphs represents the average value obtained by 10 simulations, with each simulation consisting of the encoding and decoding process for 10.000 data packets.

3.3 Analysis of Buffer Size Requirements for a Uniform Erasure Channel

In this section, we evaluate the buffer size requirements of both systematic and non-systematic codes as a function of s and the number of multicast receivers. Of interest is the required buffer size in the sender and, most importantly (due to resource limitations) the receivers.

The simulation results obtained for both codes over a uniform erasure channel are shown in Fig. 3.1. We show: the average and the maximum number of packets in the source's buffer; the average number of packets in the receivers' buffer and the average of the maximum number of packets in the worst receivers' buffer.

It can be observed that the number of packets in the buffers increases with the

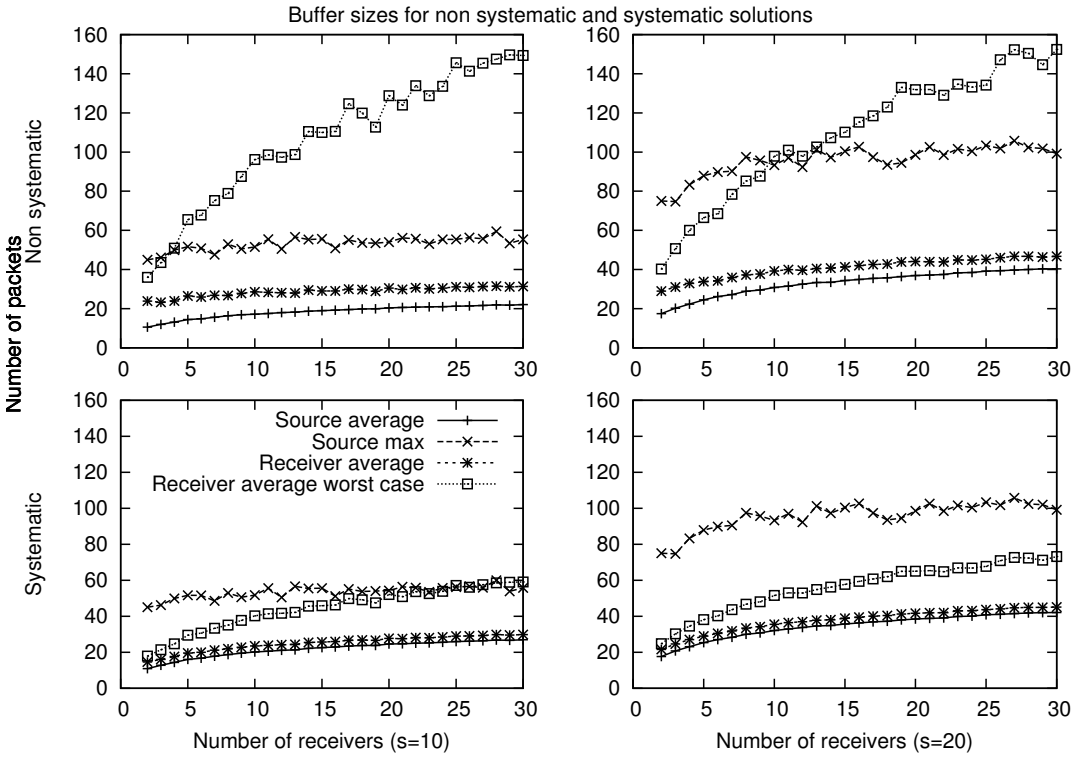


Figure 3.1: Number of packets in the nodes' buffer with with $s = 10$ (on the left) or $s = 20$ (on the right) for the non-systematic and systematic approach.

number of receivers. This is an expected result, as since all the receivers are independent, they may not receive and acknowledge the same packets and there is also a probability that an ACK packet will be lost (note that PER applies to both forward and return channels). In this case, even if the source receives an ACK from other receivers, the corresponding packet cannot be suppressed from the source's buffer and the receivers also cannot flush this packet from their buffers. Therefore, the loss of a single ACK packet impacts all nodes. Furthermore, when we increase s from 10 to 20, we can observe that the results are homothetic in regards to the sender buffer size. This result seems logical, as the source needs to store more packets between two ACKs if the receiver ACKs are less frequent. For both s values of 10 and 20, we can observe that there is a very limited difference between the two codes for the source side. This can be explained by the fact that in 10 or 20 units of time, both codes have

a high probability to obtain/decode every packet in the window, therefore they will likely acknowledge the same packets.

Considering the receivers, when changing s from 10 to 20, the growth of the curves representing the average and the worst number of packets remains constant for both codes. Actually, the packets present in the receiver buffers include both the encoded packets and the packets not yet acknowledged. These encoded packets remain in the buffer as long as the receivers cannot decode, therefore this number does not depend on s. The small increases with increased s are due to the packets which are decoded or received and need to be acknowledged. This process requires more time when s is larger.

The most significant result is observable when comparing both codes for the average worst case criteria: for $s = 20$ and 30 receivers, the value observed for the non-systematic code is close to 150, while for the systematic solution it is close to 75. I.e. for the average of the worst case receiver buffer occupancy, the non-systematic solution requires a receiver buffer two times larger than what is needed for the systematic code. We note the considerable buffer size is also required in absolute terms for the non-systematic code.

3.4 Evaluation of Code Complexity

In this section, we evaluate the computation complexity for the systematic and non-systematic code receivers and present results for a uniform erasure channel. We will use the same methodology for the bursty erasure channel, in Section 3.5. As previously noted, we consider multicast receivers to be mobile devices with limited resources.

As the simulation does not include a full encoder and decoder implementation we need to define a theoretical complexity. For this, we propose to estimate the following parameters which are directly related to complexity: the average size of the matrices which are inverted in the decoding process; the average number of non-null elements in the matrices when inverted (denoted sparsity of the matrices in the resulting figures)

and the average number of operations done per unit of time. To compute the latter value, we count the number of times a received packet is subtracted from an encoded packet and the number of operations needed to invert the matrices. The method used is similar to [100]. Please note that one operation represents a linear combination of two vectors, as this is the most complex component of an operation; we neglect the multiplication of a vector by a scalar and the size of the vector as these are simple operations. Fig. 3.2 shows the calculated complexity parameter values.

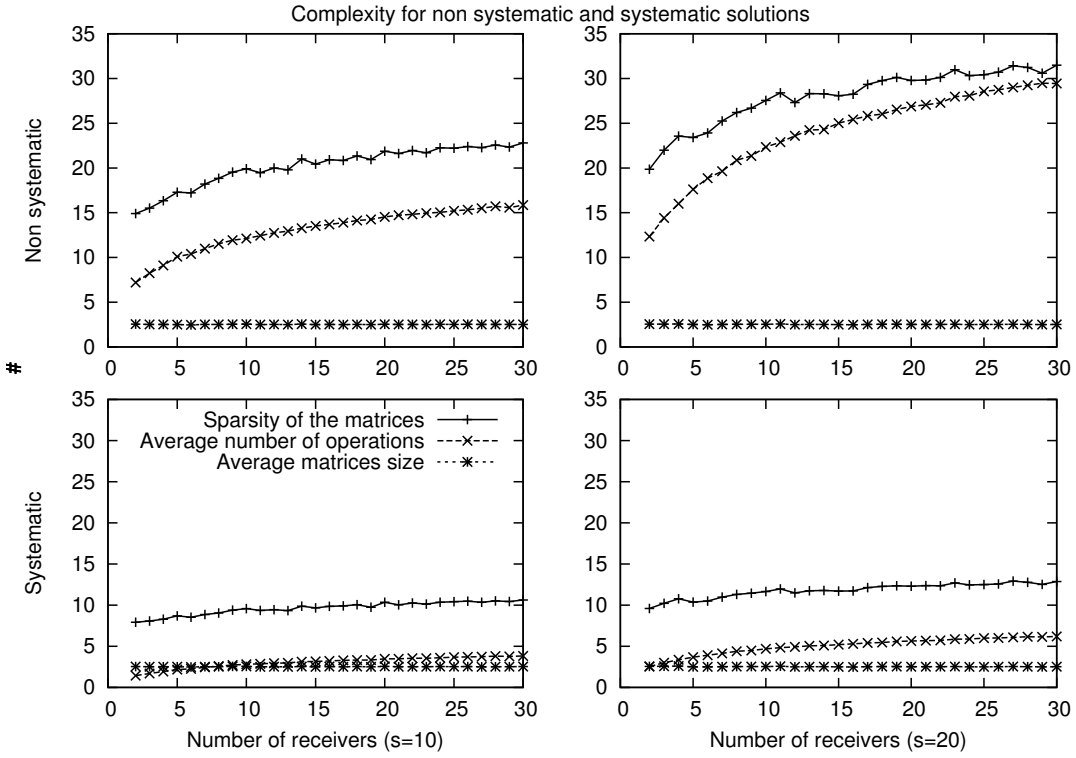


Figure 3.2: For $s = 10$ (on the left) or $s = 20$ (on the right), and $PER = 20\%$, all figures show **the complexity for the receivers** as the number of non-null elements in the matrix to invert, its size and the number of vector operations to invert it.

3.4.1 Average Matrix Size

We can observe from Fig. 3.2 that the average matrix size does not depend on s or the number of receivers. Indeed, the matrix's size only depends on the number of packets

which are lost, rather than the acknowledgements or the number of nodes involved. We can also observe that the size of the inverted matrices are very similar for both codes (for 30 receivers, the average size is 2.519 for the non-systematic and 2.522 for the systematic codes). We note that the non-systematic variant decodes more matrices, however a number of those have a size of one. This means that only one packet can be decoded after all the received packets have been subtracted from an encoded packet. However when a packet is lost, the systematic code needs to keep less packets in its buffer compared to the non-systematic code. For a $(3, 4)$ code rate, this corresponds to one in every four packets for the systematic code, while the non systematic code stores all encoded (all received) packets in the matrix.

3.4.2 Sparsity of the Matrices

The sparsity of the matrix represents the average number of non-null elements in the matrix when inverted. This parameter provides an insight into how the matrix may be populated (is it empty or full). It is used to estimate the number of operations required to invert the matrix (in Section 3.4.3) and it can also be used to better understand the size of the matrix.

First, it seems logical that when the number of receivers or s increases, the number of non-null elements in the matrix also increases. This is due to the increase of the encoding window size in the source (see Fig. 3.1), as each redundancy packet received is created from all the packets in the source buffer.

We can observe a lower bound of the variance of the size of the matrix by making the difference between the average number of non-null elements and the matrix' average size squared. In fact, the variance v verifies as:

$$v = [E(n^2) - E(n)^2] \geq [E(n_{non\_null}) - E(n)^2]$$

where n is the matrix' size when inverted and n_{non_null}, the number of non-null elements

in the matrix. Thus (note the values from Fig. 3.2) we can see that the lower bound of the variance is greater for the non-systematic codes than for the systematic ones. Furthermore, the variance is lower for the systematic case. Indeed, when any packet is lost, the non-systematic code inverts matrices of size one after the different subtractions when the systematic case does not have to decode the redundancy packets. Finally, when a packet is lost, the non-systematic code stores more encoded packets in its matrix than the systematic code, as they have to store every encoded packets after the lost one. This means one packet on four when the code is systematic but all of them in the other case.

3.4.3 The Average Number of Operations Per Unit of Time

As a criterion for the complexity, we choose the number of operations done per unit of time, rather than per matrix inversion. As the non-systematic code has to invert more matrices than the systematic code (e.g. it inverts matrices of size one even when all packets are received), the number of operations done per unit of time provides a better base for comparison of the two codes.

For both codes, it is logical that the average number of operations per unit of time increases when s and the number of receivers increase. The reason is twofold: as seen in Section 3.4.2, the matrix's sparsity value increases which implies that it is harder to invert the matrix. Furthermore, the source encoding window also increases, so when a receiver obtains a new encoded packet, it has to subtract more already received packets from it.

The main result of interest is a comparison of codes. We can observe that for all values of s and any receiver number, the systematic code outperforms the non-systematic one. As seen previously, two factors define the number of operations which need to be performed by the codes: the matrix inversion and the number of subtractions needed when an encoded packet is received. The systematic code has superior results for both factors. We already noted that on average the matrix size for decoding packets

is smaller for the systematic case, thus easier to invert, and the second point is that for the non-systematic code, all packets are encoded. Thus, every time a packet is received, the receiver has to perform a subtraction, as opposed to the systematic case, where this operation has to be done only when a repair packet is received. To illustrate the resulting impact, we can see that in the worst case (30 receivers and $s = 20$), the average number of operations needed for the non-systematic code is five times higher than for the systematic case.

3.5 Buffer Size and Complexity Analysis over a Bursty Erasure Channel

We now investigate the impact of bursty losses. We use a Gilbert-Elliot loss model [40, 34], defined by a two state Markov chain (consisting of a good and a bad state) as illustrated in Fig. 3.3. The average PER corresponds to the probability of being in the good state, while the erasure burst length correspond to the average time spent in the bad state. We choose an erasure burst length of 3. The parameters of this Markov chain are then calculated from the average PER chosen for the scenario. Knowing the average erasure burst length L and the formulas proven in Section 2.1.3.2: $PER = p_1/(1 + p_1 - p_2)$ and $L = 1/(1 - p_2)$, thus $p_2 = 1 - 1/L$ and $p_1 = PER/[L(1 - PER)]$.

Fig. 3.4 shows the buffer sizes and the complexity for $s = 20$ and $PER = 20\%$ for both codes.

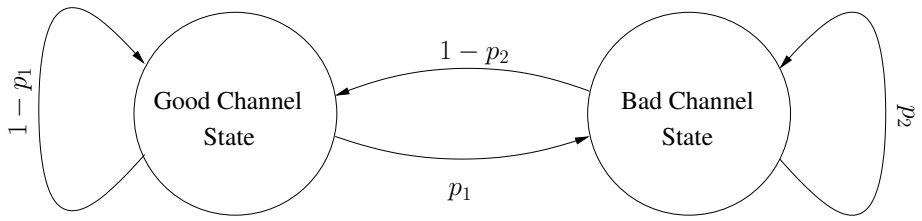


Figure 3.3: The first-order two-state Markov chain representing the Gilbert-Elliott channel model

3.5.1 Buffer Sizes

As shown in Fig. 3.4, the bursty erasure channel results, not unexpectedly, in an increased buffer size requirement for both codes. However we can observe that this channel has a significant impact only on the receivers' buffers. For the source, having a bursty channel results in a similar buffer sizes as previously observed for the non-systematic code, and the buffer slightly increases for the systematic case (the average number of packets is multiplied by 2 for the systematic code, but the worst case does not grow higher than 150 for both codes).

Concerning the receivers' buffers, we can see that all the results are multiplied by at least a factor of two. However having a bursty channel has more impact on the non-systematic code than on the systematic one. We note that for 30 receivers, the average worst case has increased by a factor of 4 for the non-systematic code when using the Gilbert-Elliot model, which shows that on the average, there is always a receiver which has 620 packets in its buffer. For the systematic code, this value is equal to 210 packets, which is still three times higher than for the uniform erasure channel. Thus we can note that the Gilbert-Elliot model further highlights the differences between the codes already observed with the uniform loss model.

3.5.2 Complexity

As the erasures occur in bursts, on the average, more packets are lost before the decoding process, so the average matrix size and the number of non-null elements in the receiver matrices are higher than in the uniform erasure channel. The most relevant result is that, compared to the resulting values on the uniform erasure channel, the average number of operations per unit of time slightly increases for the non-systematic code, while it increases by a factor of two to three for the systematic variant. Therefore, although the Gilbert-Elliot model increases the complexity of the two codes, on the average, the systematic code will again require two to three times less operations than the non-systematic (e.g. it can be observed that for 30 receivers, the systematic code

needs 15 operations per unit of time, while the non systematic needs 35).

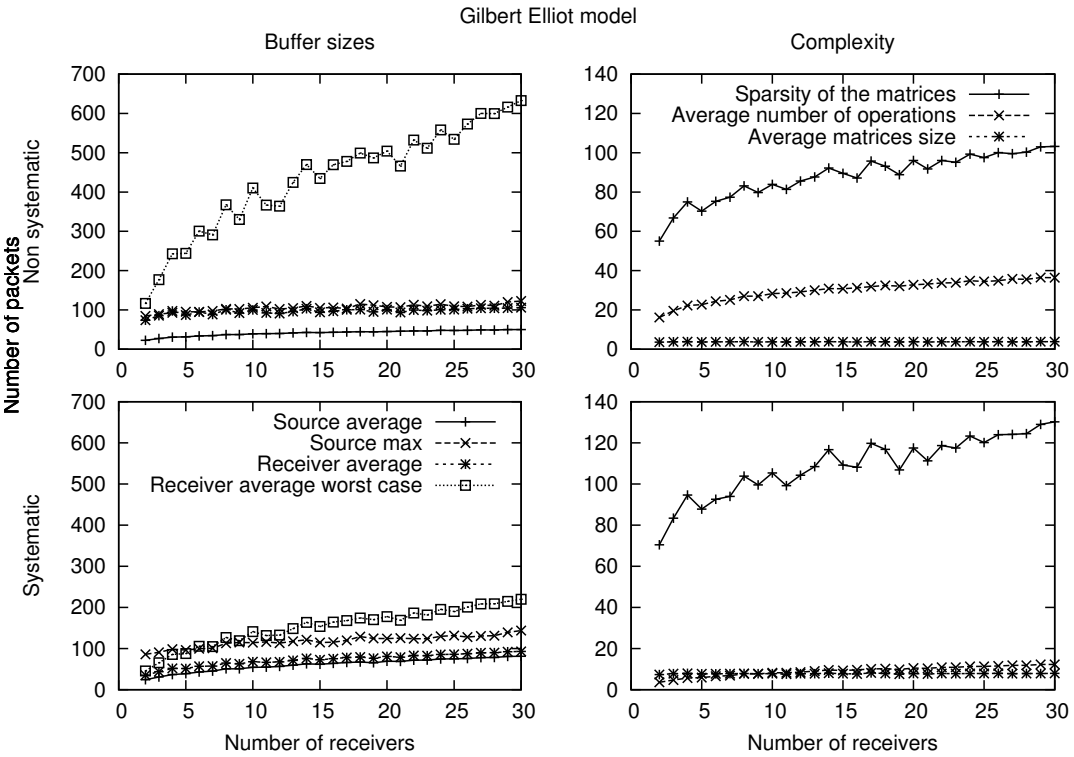


Figure 3.4: For $s = 20$ and $PER = 20\%$ using a Gilbert-Elliot losses model with an erasure burst length of 3, for both codes, on the left are the curves for the different buffers' sizes, on the right the complexities (number of non-null elements in the matrix to invert, its size and the number of vector operations to invert it).

3.6 Conclusion

We have presented an analysis of the implementation aspects of two classes of on-the-fly coding schemes for multimedia multicast communications. We have shown that the systematic approach has lower requirements in regards to the memory footprint and computation complexity of the receivers, thereby making it better suited for mobile devices. These points are crucial in the context of deployment of such schemes for Internet Protocol television (IPTV) or multimedia communications in mobile environments.

Chapter 4

Layered Secret Sharing Scheme for Automated Profile Sharing

4.1 Introduction

Secret-sharing schemes have been extensively used in a number of cryptographic and distributed computing applications [6], to enable secure storage, for MPC, generalized oblivious transfer, etc. As discussed in Section 2.3, to have a perfect secret sharing scheme (Section 2.3.1.1), the size of the shares needs to be at least the size of the secret. Strong ramp scheme as discussed in Section 2.3.1.3 provide a trade-off between security and communication cost in order to share large files.

In this chapter, we consider the use of secret-sharing to enable user-controlled and privacy preserving sharing of an OSN user's profile amongst groups of their direct connections (friends). In most commercially available OSNs, an OSN user currently stores their private data using the trusted OSN server. Access to their profile is managed by a policy, regulating which attributes (their location, marital status, date of birth, interests, etc.) may be obtained by their friends or other people. A number of challenges related to privacy in OSNs have been identified by researchers [66], including the reliance on the trusted third party (TTP) to manage the privacy control, the

arguable complexity of how those controls are implemented in current OSNs and the limited flexibility of access controls (we note Google+ circles and Facebook lists are a step in the direction of improving the flexibility). This motivates our interest in using secret-sharing to enable direct and fine grained user control of the accessibility of their private data.

We specifically consider the OSN user's requirements to share multiple profile attributes (secrets) with differing levels of security and privacy (*i.e.* thresholds), within groups of friends, without relying on a TTP. Additionally, an OSN user may not wish to disclose his level of desired privacy protection, and/or the difference in the number of attributes he is sharing with different friends (or groups of friends). It may also be preferable not to share this information from a privacy protection point of view, as any distinguishing information may be used to re-identify anonymous data [80]. Therefore, although linkage attack using such information has not been demonstrated up to now, avoiding the disclosure of this data is a desirable property when considering mechanisms for private sharing of information. As discussed in Section 2.3.2, the limitations of the currently proposed secret-sharing schemes relate to both threshold flexibility and disclosure of scheme's parametersby either fixing the same threshold for all the secrets and/or using public shares leaking information about the number of attributes a user is sharing.

To overcome such limitations, we propose a novel Layered multi-secret-sharing scheme, that recursively embeds shares related to individual secrets in layers with increased protection, and encrypts each layer with a key based on an additionally generated share. Each secret is then protected by a single secret-sharing scheme with its own threshold, and the number of secrets is hidden within the Layered scheme, which ensures that the thresholds remain unknown. The main contributions of this chapter are as follows.

We propose a new Layered secret sharing scheme that enables flexible levels of security and privacy. We introduce privacy-preserving automated profile sharing in OSN

groups as a possible use of our scheme. By generating Layered shares (comprising a selected set of attributes, corresponding to each group's privacy policy) and distributing a share to each member of the group, an OSN user automatically enforces the deployment of group's privacy policy and enables fine grained policy control within group's members, without relying on a TTP.

We analyse the security of the scheme for attacks that have a goal of illegitimately acquiring knowledge about users OSN profiles. Consequently, we show that no new Layered shares can be attained by any of the attacks that include a varying level of background knowledge. We also provide an analysis of the number of Layered shares that an attacker, who is at an arbitrary number of hops in the social graph from the node sharing his profile, may acquire. This analysis can be used to enable the profile owner to set the required level of protection offered to specific (privacy sensitive) attributes of the profile. We demonstrate, using a Facebook example, that a user with a varying number of friends (up to 100) can ensure their profile protection even when there is a high probability of leakage *i.e.* 10% (that either friends or adversaries will forward shares to other adversaries), without compromising the ability of their friends to access the profile, based on acquiring the minimum number of required shares.

We evaluate the computational and communication overhead of the proposed scheme, based on our implementation of the share-generation and decoder operations for Layered shares. These use both Shamir and strong ramp schemes as building blocks. Our scheme has a similar communications cost as a naive mechanism, where each of the secrets is shared independently using a selected scheme. The Layered scheme introduces a higher computational overhead, up to 160% for the decoding, however it still results in a relatively small computation delay of around 1 second (using ramp schemes) even when sharing profiles that include large secrets (*e.g.*, images), that we consider acceptable for real-time use.

The remainder of this Chapter is organized as follows: in Section 4.2 we present related work in privacy in OSNs. We present our Layered multi-secret-sharing scheme

and it's application to profile sharing in Section 4.3. Details of the security analysis are presented in Section 4.4 and in Section 4.5, we evaluate the performance of our Layered scheme, including the computation and communication overhead. In Section 4.6, we discuss the trade-offs and possible use of other single secret sharing schemes and we conclude in Section 4.7.

4.2 Background and Related Work on OSN Profile Sharing

A possible approach to selective sharing of different information on popular OSNs, e.g. Facebook or Google+ is to define groups of friends (friends lists or circles) and set up different privacy policies for each group. Access control is granted by the OSN operator, that owns all user data and applies the different policies. Alternatively, in order to prevent a central entity from accessing user data, [1] proposes an architecture where user data can be encrypted and stored on an untrusted server. Other proposals such as *Safebook* [26] aim at distributing the online social network itself. The data storage and selective access control (*i.e.*, attribute sharing) is provided via the "Matryoshkas" composed of rings of trust around the user. Likewise, in *PeerSon* [16] asymmetric encryption is used to provide decentralized access control to user data. In [44], authors consider two different approaches to encrypt and manage private information of user profiles. Specifically [44] provides a comparative analysis of profile management schemes, when relying either on a combination of symmetric encryption with shared keys, or involving broadcast encryption techniques.

Although decentralized privacy-preserving services offer many desirable properties including flexibility and security, in practice users may not be easily convinced to change from the well-established centralized OSNs only due to privacy concerns. [43], [69], [3] and [19] specifically address user's privacy in the context of existing OSNs, by providing browser-based plugins in order to enable user's control over

their personal data. In *NYOB* [43] and *FaceCloak* [69] for instance, fake but plausible attributes are stored on a user profile (to be compliant with Facebook's usage terms) while the way to access the actual attributes is shared privately with users' friends via an out of band channel (*e.g.*, emails).

With a similar objective of mitigating the privacy risks of current OSNs, in [68] authors propose *flyByNight*, a prototype Facebook application utilizing encryption and decryption of user personal data through proxy cryptography. *flyByNight* maintains servers that do not have access to data in the clear, and that enable proxy re-encryption to share user profiles between friends.

[3] and [19] provide users with finer access control policies than those offered by Facebook, which allows creating of groups of friends with whom to share attributes, based on relationships with other users and an established trust level.

Although there is user support for manually configured group based access control in OSNs, researchers have shown [20] that the majority of OSN users do not utilize this feature. Consequently automated algorithms for grouping friends (*e.g.*, based on common interests, family relationship, etc.) have been studied e.g. in [119]. Once the groups have been established, mechanisms like Attribute-Based Encryption (ABE) can be used to enforce access control rules [3]. However, although the access control can be defined for groups (circles) of friends, all users in the same group share a common access policy. Our proposed Layered secret-sharing scheme, as we will show, enables a finer grained access control policy that is based on the number of common friends, and automates the enforcement of this policy in each circle. This is implemented without having to rely on a trusted third party (*i.e.*, the OSN operator).

4.3 Layered Secret Sharing Scheme

In this section we describe a new Layered scheme for sharing multiple secrets. We consider the proposed scheme in an OSN application context, therefore we concentrate on the properties that can benefit this application. These include:

1. An OSN user should share his profile, comprising multiple attributes, with a specific group of his friends (contacts) via a single transaction.

2. Each attribute should have an individually selectable level of privacy per group.

3. Only the OSN friends of the profile owner (within the group) that can conform to the appropriate security level should be able to access a specific attribute of the profile.

4. The number of attributes shared by the profile owner, or their security levels should not be known to his OSN friends, other groups, or any other OSN user.

The following properties of our Layered secret sharing scheme ensure conformance to the above requirements:

- Each share created by the dealer contains information about all the secrets and can be re-used to access any of the secrets (as per requirement 1).

- Each secret is protected by a selectable threshold (as per requirement 2).

- A user can access a secret if and only if he has acquired the number of shares corresponding to the threshold of this secret (satisfying OSN requirement 3).

- The number of secrets and their thresholds are hidden from participants (as per requirement 4).

A dealer in our scheme generates Layered shares, which consist of a number of encrypted layers, each comprising (standard single secret sharing) shares related to a specific secret. Figure 4.1 shows the components of a Layered share.

The notations used in this section are listed in Table 4.1. A dealer shares k secrets with n participants P_j $j = 1, ..., n$. The secrets S_i, $i = 1, ..., k$, have t_i corresponding thresholds. We assume these are ordered in terms of increasing magnitudes, with $(t_1 \leq t_2 \leq ... \leq t_k)$ and that the secret S_1 is the system secret, chosen as any meaningful non-private text string. For the specific use case of OSN profile sharing, S_1 is a random

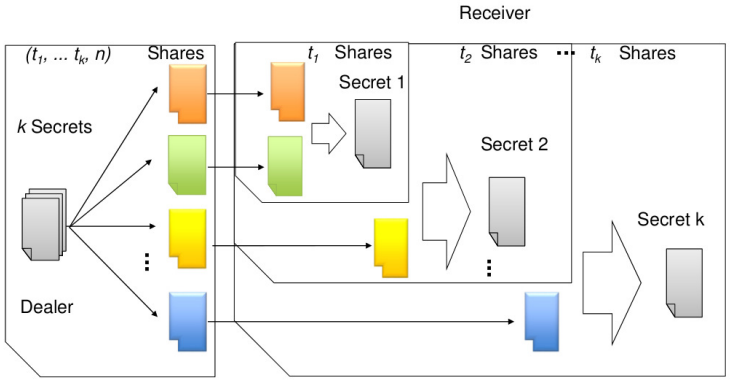


Figure 4.1: Components in the Layered secret sharing scheme

variable used for verification of Layered shares and S_2 is an OSN group's (circle) identifier.

Layered shares $\mathscr{L}_j = L_j^1$ with $j = 1, ..., n$ are generated according to steps outlined in Function *EncodeLayered*. The secrets are included recursively within the Layered shares, starting from the secret with the highest threshold (S_k). At the $(k-i)$th step, the secret S_i is to be included in L_j^{i+1} for $j = 1, ..., n$. To accomplish this, the dealer first computes $n+b_i$ standard (single secret sharing as described in Section 2.3.1) shares $(s_1^i, ..., s_{n+b_i}^i)$ with a threshold t_i using the Function *EncodeSingle*. *EncodeSingle* utilizes either Shamir's scheme [97] or the ramp scheme from [9], with the choice depending on the size of S_i and the required security level. Examples of such functions are described in Section 2.3.1.

b_i additional shares $s_{n+1}^i, \cdots, s_{n+b_i}^i$ are generated, then concatenated and hashed, using e.g. *SHA-2* to obtain the key K_i (ensuring the size of concatenation is at least the size of the encryption key, e.g. *AES* needs 128 or 256 bits keys).

For $j = 1, ..., n$, each share s_j^i related to secret S_i is concatenated with the corresponding encrypted layer $[L_j^{i+1}]_{K_i}$: $L_j^i = [t_i || s_j^i || [L_j^{i+1}]_{K_i}]$. These steps are repeated until all secrets are included in the layers.

The n Layered shares are distributed by the dealer, with a single unique share and the share's index $(j, \mathscr{L}_j)$ sent to each participant over a secure channel. As in other

Table 4.1: Notations used to describe the Layered secret sharing scheme

Notation	Description
n	Number of Layered shares and participants
P_j	Participant j for $j = 1, ..., n$
k	Number of secrets shared (including the system secret)
S_i	Secret i, for $i = 1, ..., k$
$H(.)$	One-way hash function, publicly known
$[.]_K$	Symmetric encryption using the key K
$[.]^K$	Symmetric decryption using the key K
$[A\|\|B]$	Concatenation of content A and B
s_j^i	Share j related to the secret S_i
L_j^i	Layer i of the Layered share, containing the shares related to the secrets S_i to S_k
$\mathscr{L}_j$	Layered share send to the participant P_j, $j = 1, ..., n$ (equivalent to L_j^1)
K_i	Key derived from the secret S_i
t_i	Threshold for the secret S_i
b_i	Number of additional shares needed to compute the key K_i

Function EncodeLayered($S_1, ..., S_k, t_1, ..., t_k$)

Initialization;
for $j \leftarrow 1$ **to** n **do**
  $L_j^{k+1} \leftarrow \varnothing$;
end
Genarate n Layered shares by constructing k individual layers;
for $i \leftarrow k$ **to** 1 **do**
  EncodeSingle produces $n + b_i$ (single secret sharing) shares according to the selected secret sharing mechanism
  $(s_1^i, ..., s_{n+b_i}^i) \leftarrow EncodeSingle(S_i, t_i, scheme)$;
  $K_i \leftarrow H([s_{n+1}^i || \cdots || s_{n+b_i}^i])$;
  for $j \leftarrow 1$ **to** n **do**
    $L_j^i \leftarrow [t_i || s_j^i || [L_j^{i+1}]_{K_i}]$;
  end
end

secret sharing mechanisms, participants aggregate their shares to gain access to the secret(s) S_i, as per the threshold t_i.

To describe the decoding process by which a participant recovers one or more secrets, we assume that he has acquired x Layered shares, L_z^1 where $z = 1, ..., x$ (to simplify the notation, without loss of generality we assume that Layered shares are received in the order of their indexes). We also assume that the participant has previously decoded i secrets $S_1, ..., S_i$ with thresholds $t_1, ..., t_i$ and their corresponding keys $K_1, ..., K_i$, where i is defined by $t_i \leq x - 1 < t_{i+1}$, using the previously received $x - 1$ Layered shares ($i = 0$ if no secrets have been decoded). Thus he has $x - 1$ layers L_z^{i+1}, where $z = 1, ..., x - 1$ and a Layered share L_x^1 that is yet to be processed. Function *DecodeLayered* describes the decoding process. In the first step, the "old" layers related to the previously decoded secrets need to, recursively, be removed and decrypted (using $K_1, ..., K_i$) from L_x^1. If $x == t_{i+1}$, function *DecodeSingle* can then be used to decode the secret S_{i+1} and also to compute as previously the key K_{i+1}. Finally the layers related to the secret S_{i+1} are removed from the x Layered shares and the following parts $[L_z^{i+2}]_{K_{i+1}}$ are decrypted, using the key K_{i+1}.

4.3.1 Using Layered Shares for Profile Sharing in OSN Groups

To initialize the scheme, the dealer must first define circles of friends, C_i, and the attributes (secrets) he intends to share amongst the friends in each circle. He should also allocate friends to circles, either manually or using any of the proposed mechanisms, e.g. [119]. Then, for each circle and each shared attribute, the dealer should choose a privacy level for accessing the attribute (*i.e.* the threshold for each secret), that will, in our scheme, be equivalent to the number of common friends that a user in this circle needs to have with the profile owner. We acknowledge that this criteria may not be universally applicable to all OSN groups. However, we believe, as per [119], that the number of common friends represents a relevant and easy to understand metric for the OSN users.

Function DecodeLayered($\mathscr{L}_x = L_x^1$)

Remove the layers related to the previously decoded secrets;
for $y \leftarrow 1$ **to** i **do**
    $[s_x^y || [L_x^{y+1}]_{K_y}] \leftarrow L_x^y$;
    $L_x^{y+1} \leftarrow [[L_x^{y+1}]_{K_y}]^{K_y}$;
end
Decode the secret S_{i+1} if there is a sufficient number of shares;
if $x == t_{i+1}$ **then**
    for $j \leftarrow 1$ **to** x **do**
        $[s_j^{i+1} || [L_j^{i+2}]_{K_{i+1}}] \leftarrow L_j^{i+1}$;
    end
    DecodeSingle decodes the standard shares, according to the selected single secret sharing scheme; outputs the secret S_{i+1} and the standard shares $s_{n+1}^{i+1}, \cdots, s_{n+b_i}^{i+1}$;
    $(S_{i+1}, s_{n+1}^{i+1}, \cdots, s_{n+b_i}^{i+1}) \leftarrow$`DecodeSingle(`$scheme, t_{i+1}, s_1^{i+1}, ..., s_x^{i+1}$`)`;
    Generate the encryption key from the share s_{n+1}^{i+1};
    $K_{i+1} \leftarrow H([s_{n+1}^{i+1} || \cdots || s_{n+b_i}^{i+1}])$;
    Remove the layer related to S_{i+1} for all (available) Layered shares;
    for $j \leftarrow 1$ **to** x **do**
        $L_j^{i+2} \leftarrow [[L_j^{i+2}]_{K_{i+1}}]^{K_{i+1}}$;
    end
end

The dealer then generates Layered shares (using *EncodeLayered*) and distributes them to friends in a specific circle over a secure channel. Subsequently, each friend contacts their friends and collects the available Layered shares from the friends in common with the dealer (that also belong to the same circle). Any friend can then decode the accessible attributes according to the number of collected Layered shares, using *DecodeLayered*.

We envisage the profile sharing application to be implemented as a browser plugin, where all the operations related to the shares are automated and secure. We note that for the scheme to be operational, all participating OSN users need to have the plugin installed.

For new friend connection requests, the dealer first needs to select the appropriate circle for this new friend. Then, he needs to generate a new Layered share and index and send it over a secure channel to the new friend.

Finally, if a user wishes to remove a friend from a circle or change his circle, he has to renew all Layered shares for this circle, to ensure that his Layered share will not be usable in future exchanges. An alternative would be to use the proxy re-encryption approach from [53], that does not require share renewal.

4.4 Security Analysis for the OSN Application

This section presents an analysis of the security of the proposed scheme and its application to the automated OSN profile sharing. For the sake of simplicity, we only consider the case where all user's friends belong to a single circle.

4.4.1 Adversary Model

The threat scenario consists of adversaries that will attempt to access more information than what they may obtain legitimately i.e. from the Layered shares acquired either directly from the dealer or from the dealer's friend connections. This includes an honest but curious adversary who is an insider (a user) within the system and has

access to a number of Layered shares, or a malicious user who obtains all information by monitoring communications of other nodes, by compromising nodes, *etc.* We note that the information any attacker may gain includes Layered shares and background information, comprising one or more decoded secrets (attributes of a user's profile), their locations (in layers) and their level of protection (thresholds).

4.4.2 Properties of Shamir's and Ramp Schemes

To assist with our analysis, we first remind relevant properties of the two (single) secret sharing schemes used in constructing layers in our proposal:

Property S.1 *Shamir's scheme is information theoretically secure, that is, for a* (t, n) *scheme, knowledge of less than* t *shares does not provide any information about the remaining shares or the secret (corresponding to the first property of the Equation 2.11 in the Section 2.3.1.1).*

Property S.2 *In Shamir's scheme, the secret is equivalent to a share, as it is a specific point on the curve defined by the polynomial used to both generate and decode shares (as mentioned in the Section 2.3.1.1, the secret is the point* $P(0)$ *of the polynomial, the shares other points of the same polynomial).*

Property R.1 *The ramp scheme is not perfect, however for a* $(t, L = t, n)$ *scheme, possession of* $x < t$ *(less than* t *shares) does not provide any information about any remaining (unknown) shares and any part of the secret, but the size of the entropy of the solution set is reduced from* $H(S)$ *to* $\frac{t-x}{t}H(S)$ *(Definition 10 in Section 2.3.1.3).*

Property R.2 *Knowledge of the secret in the ramp scheme with* $L = t$ *enables reconstruction of all the related shares (as the coordinates used for the shares computation are known).*

4.4.3 Privacy of the Scheme

We assume the attacker has x Layered shares, $x \in [0, n]$. This will enable him access to corresponding secrets S_1 to S_i, decoded from those shares, with i defined by the

threshold $t_i \leq x < t_{i+1}$. The attacker also may have background knowledge of n_S, $n_S \in [0, k]$ secrets, their position in the layers and related thresholds. In the following, we consider the different scenarios related to the position of known secrets and the resulting gain by the attacker.

We consider the use of both Shamir's and the ramp secret sharing scheme any layer and utilize the properties of these schemes, defined in Section 4.4.2, in the security analysis.

4.4.3.1 The Attacker has Access only to Layered Shares

The privacy protection our scheme provides is equivalent to that of the underlying scheme, as defined by properties S.1 and R.1, respectively for Shamir's and the ramp scheme.

4.4.3.2 Attacker has Access to both Layered Shares and Background Knowledge

We first consider the cases when one of the following secrets, S_{i+1} or S_{i+2} is known, then generalise the analysis to the attacker having access to any number of secrets (not acquired from the available Layered shares).

The adversary knows secret S_{i+1} in addition to Layered shares:

For Shamir's scheme, knowledge of the secret is equivalent to having access to an additional share related to S_{i+1} (see property S.2). If the new number of shares is below t_{i+1}, as per the property S.1, the attacker cannot progress further. If the threshold was reached, the attacker can recompute all related shares; consequently (from the additional shares), he can acquire the key K_{i+1} required to decrypt the next layer of shares within the available Layered shares.

For the ramp scheme, knowledge of the secret and the threshold t_{i+1}, as per property R.2, enables the attacker only to reconstruct the key K_{i+1} and access the next level of shares in the Layered scheme. Similarly to the case when Shamir's scheme is

used, the attacker can, as the maximum gain, succeed in having access to x shares in the next layer of the Layered shares.

The adversary knows secret S_{i+2} together with Layered shares: Here, the adversary aims to access the unknown secret S_{i+1} and potentially further information about the remaining secrets.

For Shamir's scheme (re. S.2), there is no additional gain.

For the ramp scheme, (as per R.2) the attacker can reconstruct all the shares related to S_{i+2}. This would enable him access to K_{i+2} for decrypting the next level of shares in the Layered scheme, related to the secret S_{i+3}. But first the attacker needs to reconstruct the key K_{i+1} to be able to decode the Layered shares L_j^{i+1} and then the Layered shares L_j^{i+2} using the key K_{i+2}.

A plaintext attack, in which the adversary partially knows the decrypted text string, i.e. shares related to S_{i+2}, could be used to gain K_{i+1} (we note that, to the best of our knowledge, there has been as yet no successful plaintext attack on *AES*). Assuming this is successful, the attacker would obtain K_{i+1}, i.e. a cryptographic hash of b_{i+1} shares related to S_{i+1}. Assuming the attacker has sufficient computational resources to derive these shares from the one-way hash function, the resulting gain would be b_{i+1} additional shares related to S_{i+1}; if the threshold t_{i+1} is reached, the attacker will gain the additional secret S_{i+1}.

The adversary knows any number of subsequent secrets: For any case where the next layer of shares (in the Layered scheme) is protected by Shamir's scheme, and assuming the secret is known to the attacker, the maximum possible gain can be the availability of the key to decrypt the next layer. When the next layer is protected by the ramp scheme, the certain gain is the same key. When the second-next layer is based on Shamir's secret sharing, assuming a known secret, there is no possible gain; when it is based on the ramp scheme, again with a known secret, both the previous layer and the following layer may be decrypted. We note that no new Layered shares can be gained by the attacker with an arbitrary level of background information.

Table 4.2: Variables used in our analysis

Variable	Description
$\mathscr{F}_u^j$	Set of j-degree friends for any user u (denoted as o for the profile owner and a for the attacker)
$\mathscr{A}_j$	Intersection of $\mathscr{F}_u^j$ and $\mathscr{F}_a$ where $a \in \mathscr{F}_u^{j+1}$
A_j	Random variable describing the cardinality of $\mathscr{A}_j$, having the value a_j
N	Random variable describing the cardinality of $\mathscr{F}_u$, having the value n
X	Random variable describing the number of Layered shares the attacker may obtain, having the value $x \in [0, n]$
p_j	Probability that a node in $\mathscr{F}_u^j$ sends his Layered shares to one of his direct friends in $\mathscr{F}_u^{j+1}$

4.4.4 Analysis of Attacker's Access to Layered Shares

We now provide an analysis of the number of Layered shares that an adversary potentially may have access to, in the OSN scenario where a user is sharing his profile with direct friends.

We assume that an adversary is a node belonging to the social graph of the user sharing his profile and that he obtains information (shares) from his direct connections. Second, we assume that the probability that an adversary will obtain shares decreases with the distance from the node sharing his profile (increasing degree) and we only consider the path with the shortest distance as relevant in obtaining information.

We first define the variables used in our analysis, summarized in Table 4.2. Let $\mathscr{F}_u^j$ be the set of j-degree friends of any user u (friends at a distance of j hops from the user u) in the OSN for $j \geq 1$, defined as: $\mathscr{F}_u^1$ being the set of direct friends of u for $j = 1$ and $\mathscr{F}_u^j = \{w \mid w \in \mathscr{F}_v, v \in \mathscr{F}_u^{j-1}, w \notin \cup_{i=1}^{j-1} \mathscr{F}_u^i\}$ for $j > 1$. This follows the nomenclature from [108], however in this analysis we assume that a node of the social graph does not have multiple degrees[1]. Let $\mathscr{F}_o^j$ and $\mathscr{F}_a^j$, for $j \geq 1$, be the sets of j-degree friends of, respectively, the profile owner and the attacker.

Assume the attacker is at a distance $j + 1$, for $j \geq 1$, from the owner in his social graph; thus the attacker belongs to $\mathscr{F}_o^{j+1}$. We define $\mathscr{A}_j = \mathscr{F}_o^j \cap \mathscr{F}_a^1$ as the set

[1]This follows from our assumption that the probability for an attacker to obtain information on any path is negligible compared to the probability of obtaining information on the shortest path.

consisting of direct friends of the attacker, who are also j-degree friends of the owner. Let the random variable $A_j = |\mathscr{A}_j|$ be the size of $\mathscr{A}_j$, having the values a_j. Let the random variable $N = |\mathscr{F}_o^1|$ be the number of friends that the profile owner has, having the value n. Let the attacker have X Layered shares, where X is a random variable having a specific value $x \in [0, n]$. Finally, let p_j be the probability that a node in $\mathscr{F}_o^j$ will forward his Layered shares to one of his direct friends in $\mathscr{F}_o^{j+1}$, for all $j \geq 1$.

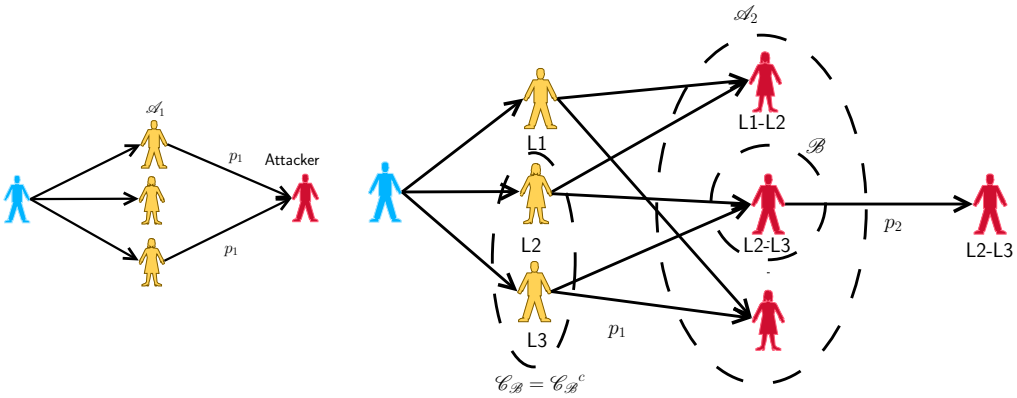
We assume that the direct friends of the owner follow the protocol and, although they may be tricked by the attacker masquerading as another friend, they would release to him only their own (single) share. Thus an attacker in $\mathscr{F}_o^2$ will receive one Layered share from an element in $\mathscr{F}_o^1$ with a probability p_1. All other nodes (on the social graph of the profile owner) who have shares are adversaries and they would forward all their available shares to the attacker. Therefore an attacker in $\mathscr{F}_o^{j+1}$ will receive all available Layered shares from the j-degree friends of the owner ($j > 1$) with a probability p_j.

In the following analysis, we derive the probability that an attacker in $\mathscr{F}_o^{j+1}$ will obtain x Layered shares, assuming the owner has n direct friends: $P_j(X = x \mid N = n)$. To this purpose, we consider three cases: we provide a detailed analysis for a close attacker, with $j = 1, 2$ and we generalize the analysis for $j \geq 3$. For the latter case ($j \geq 3$), we assume that all the elements in $\mathscr{F}_o^j$ receive Layered shares independently.

4.4.4.1 The Adversary is Close to the Profile Owner

First, we consider an *attacker who is at a distance two from the profile owner.* The attacker will successfully acquire x Layered shares from the friends he has in common with the profile owner and he will receive one Layered share from each node in $\mathscr{A}_1$ with a probability p_1. It follows that the attacker will not receive shares from the $a_1 - x$ elements of $\mathscr{A}_1$, therefore:

$$P_1(X = x \mid N = n) = \sum_{a_1=x}^{n} P(A_1 = a_1 \mid N = n) \binom{a_1}{x} p_1^x (1 - p_1)^{a_1 - x} \qquad (4.1)$$



(a) Attacker is in $\mathscr{F}_o^2$ $(j = 1)$ (b) Attacker is in $\mathscr{F}_o^2$ $(j = 1)$

Figure 4.2: Attacker is in $\mathscr{F}_o^2$ $(j = 1)$ or in $\mathscr{F}_o^3$ $(j = 2)$.

The distribution $P(A_1 = a_1 \mid N = n)$ can be computed for $a_1 \in [0, n]$ from the social graph of the selected OSN.

4.4.4.2 The Adversary is a 3-Degree Friend of the Profile Owner

The probability that an *attacker at a three hops distance* receives x Layered shares from nodes in $\mathscr{A}_2$, assuming the owner has n direct friends and the size of the set $\mathscr{A}_2$ is a_2, can be computed as the probability that the attacker successfully receives these x Layered shares from a subset of nodes $\mathscr{B} \subset \mathscr{A}_2$.

$$\begin{aligned} P_2(X = x \mid N = n) &= \sum_{a_2=0}^{\infty} P(X = x \wedge A_2 = a_2 \mid N = n) \\ &= \sum_{a_2=0}^{\infty} P(X = x \mid N = n \wedge A_2 = a_2) P(A_2 = a_2 \mid N = n) \end{aligned} \tag{4.2}$$

where $P(A_2 = a_2 \mid N = n)$ can be computed from the OSN's social graph. We note that the size $\mathscr{A}_2$ can vary from 0 to ∞ however in practice it is limited by the number of nodes in the OSN. Let the size of $\mathscr{B}$ be represented by the random variable $B = |\mathscr{B}|$,

with the values $b \in [0, a_2]$. We can then calculate:

$$P(B = b \mid A_2 = a_2 \wedge N = n) = \binom{a_2}{b} p_2^b (1 - p_2)^{a_2 - b} \tag{4.3}$$

The subset $\mathscr{B}$ provides the x Layered shares if and only if the nodes in this subset obtain this data from exactly x direct friends of the profile owner (note each direct friend only forwards a single Layered share). We define $\mathscr{C}_{\mathscr{B}}$ of size $C_{\mathscr{B}}$ as the set of common direct friends between the owner of the profile and the elements in $\mathscr{B}$: $\mathscr{C}_{\mathscr{B}} = \{w \in \mathscr{F}_o \cap \mathscr{F}_v \mid v \in \mathscr{B}\}$. We also define ${\mathscr{C}_{\mathscr{B}}}^c$ as the subset of $\mathscr{C}_{\mathscr{B}}$ from which the elements in $\mathscr{B}$ are actually receiving the information. Thus, ${C_{\mathscr{B}}}^c = |{\mathscr{C}_{\mathscr{B}}}^c|$ needs to be equal to x. Consequently we can calculate:

$$\begin{aligned} P(X = x \mid N = n \wedge A_2 = a_2) = \sum_{b=0}^{a_2} & P(B = b \mid A_2 = a_2 \wedge N = n) \\ & \times P({C_{\mathscr{B}}}^c = x \mid B = b \wedge A_2 = a_2 \wedge N = n) \end{aligned} \tag{4.4}$$

Then

$$\begin{aligned} P({C_{\mathscr{B}}}^c = x | B = b \wedge A_2 = a_2 \wedge N = n) = \sum_{l=x}^{n} & P(C_{\mathscr{B}} = l | B = b \wedge A_2 = a_2 \wedge N = n) \\ & \times P({C_{\mathscr{B}}}^c = x \wedge C_{\mathscr{B}} = l | B = b \wedge N = n) \end{aligned} \tag{4.5}$$

Let us denote the number of edges between a node in $\mathscr{C}_{\mathscr{B}}$ and the nodes in $\mathscr{B}$ by the random variable φ, with a value $z \in [1, b]$. Note that $P(\varphi = z \mid B = b \wedge N = n)$ can

be computed from the OSN's social graph. Finally:

$$P(C_{\mathscr{B}}{}^{c}{=}x{\wedge}C_{\mathscr{B}}{=}l|B{=}b{\wedge}N{=}\,n) = \binom{l}{x}\left(\sum_{z=1}^{b} P(\varphi{=}z|B=b{\wedge}N{=}n)(1-p_1)^z\right)^{l-x} \left(\sum_{z=1}^{b} P(\varphi=z|B=b{\wedge}N=n)(1-(1-p_1)^z)\right)^{x} \tag{4.6}$$

as each element in $\mathscr{C}_{\mathscr{B}}{}^{c}$ needs to be connected (with a probability p_1) to at least one element of $\mathscr{B}$, while each element in $\mathscr{C}_{\mathscr{B}} \backslash \mathscr{C}_{\mathscr{B}}{}^{c}$ does not have any connection to elements of $\mathscr{B}$. Finally, $P_2(X = x \mid N = n)$ can be computed by combining equations (4.2)-(4.6).

4.4.4.3 The Adversary is Distant from the Profile Owner

For the generic case of a *distant adversary*, we assume that the nodes at a distance $j+1$ for $j \geq 2$ receive independent sets of Layered shares (although any share can be in multiple sets), in line with the assumption that the further two nodes are from the profile's owner, the less chance they have to have common friends in $\mathscr{F}_o^j$. We can then calculate:

$$\begin{aligned} P_j(X = x \mid N = n) &= \sum_{a_j=0}^{\infty} P(X = x \wedge A_j = a_j \mid N = n) \\ &= \sum_{a_j=0}^{\infty} P(X = x \mid N = n \wedge A_j = a_j) P(A_j = a_j \mid N = n) \end{aligned} \tag{4.7}$$

where $P(A_j = a_j \mid N = n)$ can be computed from the OSN's social graph.

The probability that the attacker receives the Layered shares from k nodes in A_j is equal to $\binom{a_j}{k} p_j^k (1-p_j)^{a_j-k}$. Then the probability that one of the k nodes has x_y Layered shares for $y \in [1,k]$ is equal to $P_{j-1}(X = x_y \mid N = n)$. Finally we need to evaluate the probability that the size of the union of any k sets of size x_y amongst n of Layered shares is equal to x, denoted as $P(|U_1^k| = x \mid x_y)$. The derivation of this probability is included in Appendix B.

We finally derive the probability that the attacker receives x Layered shares:

$$P(X{=}x \mid N{=}n \wedge A_j{=}a_j) = \sum_{k=1}^{a_j} \binom{a_j}{k} p_j^k (1-p_j)^{a_j-k} \times \sum_{y=1}^{k} \sum_{x_y=0}^{n} P(|U_1^k|{=}x \mid x_y) \prod_{z=1}^{k} P_{j-1}(X{=}x_z \mid N{=}n) \tag{4.8}$$

4.4.5 Calibrating the Thresholds

Using the analysis from the previous sub-section, an OSN user could compute, for known properties of the OSN's social graph and for each distance $j+1$, the cumulative distribution $P_j(X \geq x \mid N = n)$. This is the probability that an attacker in $\mathscr{F}_o^{j+1}$ can access at least x Layered shares. This probability is equivalent to the proportion of attackers in $\mathscr{F}_o^{j+1}$ that may acquire this number of shares. The profile owner can therefore determine the minimum threshold t_{min}^j for generating the Layered shares, to ensure that less than a specific proportion c of attackers at distance $j+1$ can access the secrets,

$$t_{min}^j = \min\{x / P_j(X \geq x \mid N = n) \leq c\}. \tag{4.9}$$

The required (overall) minimum threshold can be calculated for adversaries at any distance as

$$t_{min} = \max_{j \geqslant 1}(t_{min}^j). \tag{4.10}$$

To evaluate the practicality of our proposal, we need to ensure that the number of Layered shares that friends of an OSN user can obtain, is always greater than the minimum threshold required to protect against attacks. To obtain representative OSN parameters, we use the Facebook New Orleans dataset [106], comprising around 60,000 nodes, to derive the distributions required for calculating $P_j(X \geq x \mid N = n)$, as per

Section 4.4.4. We choose three levels of resilience: such that less than, respectively, 1%, 0.1%, and 0.01% of the adversaries at selected distances from the dealer are able to access data if corresponding thresholds t^j_{min} are chosen. We also choose two values for the probabilities p_j (5% and 10%) that a friend of the dealer will leak information to an adversary and for an adversary to forward all Layered shares to another adversary.

Figure 4.3 shows the calculated minimum thresholds for selected parameters. This figure also shows the average number of Layered shares that friends of a dealer will be able to acquire (the number of friends is varied between one and 100, the average number of Facebook friends [105]). We can observe that even with a high (10%) leakage from dealer's friends and other adversaries, the number of available shares that the friends can acquire is greater than the minimum threshold required for the highest level of protection (99.99%) against adversaries. We can also confirm the potential for having flexible privacy protection for shared attributes: an OSN user with 40 friends can have 5 different levels of protection, while the average Facebook user who has 100 friends may to have around 10 different levels of shared data protection.

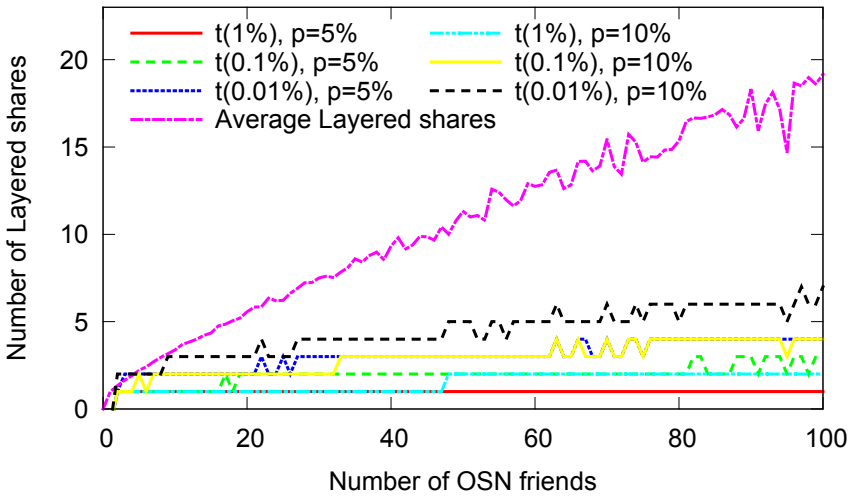


Figure 4.3: Thresholds required to ensure a chosen level of protection against attackers at distance $j = 2$ from the dealer.

4.5 Performance Evaluation

In this section, we evaluate the performance the of Layered scheme. We consider the use of both Shamir and ramp schemes for constructing layered shares. As a baseline, we also evaluate the naive approach, that shares each secret independently using the corresponding secret sharing scheme.

We measure the computational overhead, represented by the duration of the share generating and decoding operations and the communication overhead, as the volume of traffic required to perform the operations for each scheme when using selected parameters. Other than the performance comparison between the schemes, we are also interested in the absolute values of the time taken to perform the share generating and decoding operations, determining the *usability* of the scheme.

The OSN application assumes real time use, i.e. (although e.g., a browser plugin could be used as a means of integrating the implementation into a practical system) a user needs to generate shares and distribute them in real or near-real time. Similarly, user's OSN connections need to, once they have aggregated a sufficient number of shares through exchange with friends from the same OSN group, decode the shares and access the profile attributes (secrets) in real time. In line with the 3G QoS requirements for the interactive traffic class designed for web browsing, we consider a delay of the order of magnitude of several seconds as acceptable (note [55], pp. 403, lists an upper limit of 4 seconds).

We first describe our implementation and the experimental setup.

4.5.1 Implementation Details

We have developed both the Layered secret-sharing and the naive approach packages in Java. The packages comprise two main classes: first, the share generator, that constructs shares (Layered or single) from an input array of profile attributes and their corresponding thresholds t (and L if using a ramp scheme). Second, the decoder that inputs a series of (Layered or single) shares and outputs the decoded secrets i.e.

the profile attributes.

Our implementation of Shamir secret sharing share generator/decoder uses a class from the SEPIA [17] library. As SEPIA is originally designed for Secure Multi-Party Computation, a number of modifications were needed for our profile sharing application. These include the capability to generate and to decode an additional share for creating the key required for Layered shares. Our implementation of the share generator/decoder for the strong ramp scheme is based on Blakeley's generalisation of Shamir scheme [9]. It is more loosely based on the SEPIA code as more extensive modifications were required. We note that our proof of the strong ramp properties for this scheme and the description of the implementation have been reported in Chapter 5.

As both schemes are operating over the Galois field $GF(1,401,085,391)$ that is based on a 31-bit prime number, input and output data handling needs to be adjusted in line with the size of this number. In the following sub-section, we will detail the scenarios we have used for evaluation, here we note that we consider profile attributes to be either 140-character text strings or images. For both, we convert the input secrets into a number of 4-Byte (4 readable ASCII character) arrays that are subsequently individually shared as processed secrets, either using a Layered or a single sharing scheme. This includes some overhead, as, depending on the specific ASCII character values, the 4 Bytes may not be able to be accommodated within the 31-bit prime number defining the Galois field (for these cases, we only use 3 Bytes and pad the remaining values with 0s).

There is an additional step for handling images - these need first to be encoded into a string of readable ASCII characters. We use a BASE64 encoding as implemented in the standard Java library *DatatypeConverter* class. This increases the image size, as we will further discuss in the following sub-section.

Finally, we have used the AES encryption and decryption and the SHA-256 hash function implementations from the *Cipher* and *MessageDigest* classes of the Java lib-

Table 4.3: Attribute types and thresholds for each of the example profiles (T for Text).

Attribute	1	2	3	4	5	6	7	8	9	10
Profile 1	T	T	T	T	T	T	T	T	T	T
Profile 2	T	Profile picture	T	T	T	T	T	T	T	T
Profile 3	T	Profile picture	T	T	T	T	T	T	T	Picture
Threshold	5	10	15	20	25	30	35	40	45	50

rary.

4.5.2 Experimental Scenario and Setup

For the evaluation scenario, we consider the Facebook user profile comprising 40 attributes [105]. We assume the user is sharing only $k = 10$ attributes within a group of $n = 100$ friends [105], with secret-sharing thresholds varying between 5 and 50 (in increments of 5). We consider three profile types (Table 4.3).

Profile 1 All 10 profile attributes are text strings of 140 readable ASCII characters. We note that Facebook does not have an explicit limit on the length of the profile attributes. We therefore adopt the Twitter message size as a baseline for defining the size of text-based profile attributes[2].

Profile 2 This profile comprises nine text string attributes and a *profile photo*, with a size of 160×160 pixels[3]. We select the profile image as the second least protected attribute in the Layered scheme. We note that the choose the user's name as the lest protected (text) attribute.

Profile 3 Similarly to Profile 2, this profile includes the profile photo in the same position. The most protected attribute (last in the order of attributes when using the Layered scheme) is a second image, with a size of 520×520 pixels. All remaining attributes are texts strings

[2]Tweets can be up to 140 characters in length, https://support.twitter.com/articles/15367-posting-a-tweet.

[3]Facebook profile photo, http://havecamerawilltravel.com/photographer/images-photos-facebook-sizes-dimensions-types

We use a common reference image[4] for both the profile photo and the additional image. The respective file sizes for the images used in our experiments are 22.68 kB and 473.831 kB. We note that the encoding step required to convert the images to an array of strings increases the file sizes by about 33 %. We note, however, that it is only a by-product of our current implementation, and not a requirement of the underlying secret sharing schemes. Improvements to the efficiency of our implementation are planed for future work.

We run the share generating and decoding experiments on a cluster built from Dell PowerEdge 1950 servers equipped with 2×2.00 GHz Intel Quad Core Xeon E5405 processors, each with a 6 MB of cache, and with 16 GB of RAM. We note that each process runs on a single core and shares the accessible RAM with 7 other processes. Therefore, in our the experiments, the share generating and decoding for each of the example profiles was run on a single 2.00 GHz core CPU with at most 2 GB of RAM. We note that the computing capabilities of this platform are similar to what is available for the more advanced current mobile devices, such as the LG Nexus 5[5]. We envisage that the current trend of accessing OSN services using mobile devices, as is the case for the largest OSN, Facebook[6] will continue and that in the near future the majority of users will access OSNs via such devices.

4.5.3 Computational Overhead

We first study the computation time required to complete the share generating and decoding operations for the specific secret sharing scheme.

Figure 4.4 shows the average measured time for generating and decoding shares, for each of the profile types and for the case when the full profile is both shared and recovered. We note that this includes generating $n = 100$ shares for all schemes (the naive scheme actually generates this number of shares for each of the 10 profile attrib-

[4] http://en.wikipedia.org/wiki/File:Lenna.png

[5] http://www.lg.com/au/mobile-phones/lg-D821

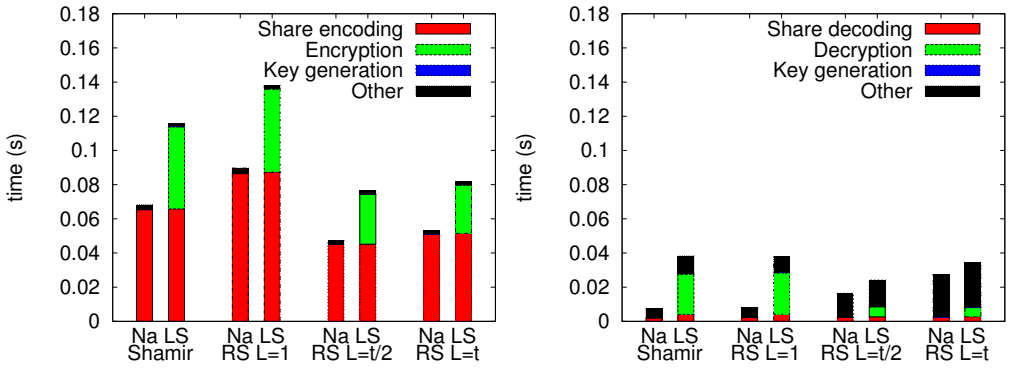
[6] Facebook statistics as of January 2014, from http://www.statisticbrain.com/facebook-statistics/

utes, while the Layered scheme deals with all attributes simultaneously). Similarly, with the selected maximum threshold $t = 50$, the same number of shares is combined in the decoding process to recover all profile attributes. We note that only the average values are shown in Figure 4.4 for clarity, and Tables 4.4 and 4.5 additionally list the corresponding values of standard deviation. For all results, the share generating and decoding measurement is an average of 2500 profile sharing or recovery experimental rounds.

We consider three cases: (a) when the underlying single secret sharing mechanism for Layered shares is Shamir secret sharing; (b) when each layer is based on a strong ramp scheme with $L = t$ (note the values of t vary); and (c) when the $L = \lceil \frac{t}{2} \rceil$ ramp scheme is used in each layer, where $\lceil \cdot \rceil$ is the ceiling function. The use of $L = t$ strong ramp scheme minimises the size of shares at the expense of minimum offered security. On the other hand, using the $L = \lceil \frac{t}{2} \rceil$ ramp scheme results in a trade-off between the improved security and larger share size.

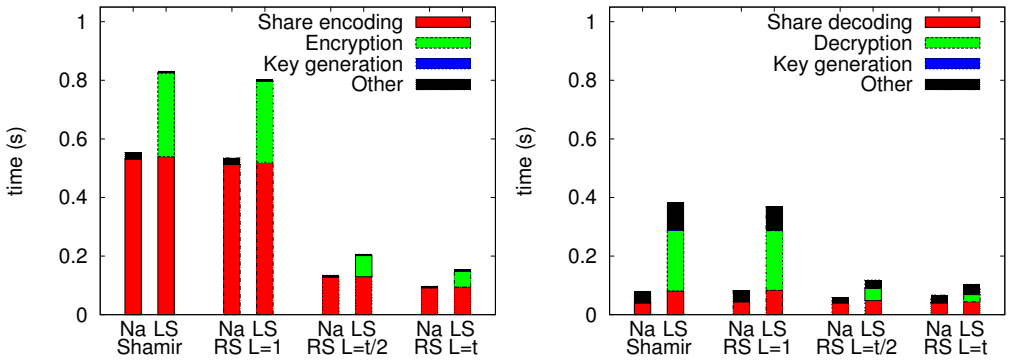
To aid the analysis, the share generating and decoding process for Layered shares is divided into four basic components, shown in Figure 4.4. First, the corresponding single secret sharing scheme based operations to generate or decode individual shares. For Layered shares, this includes the time to process the additional shares used for generating the key. Then, the time to construct the key, encryption (in the share generating process) and decryption (part of the decoding process) and finally other residual operations related to our implementation (e.g., BASE64 transcoding operations and data padding). We note that the first and the last components are also applicable to the naive scheme.

We first consider the difference in computation times for the Layered and naive schemes. We can observe that using the naive approach results in faster generating and decoding of shares, for all parameter choices. Considering the contributing components, the largest contributor are the additional cryptographic operations, that add, on the average, 32–62 % (35 ms–5 s) for the share generating and 160–265 % (15 ms–6 s)



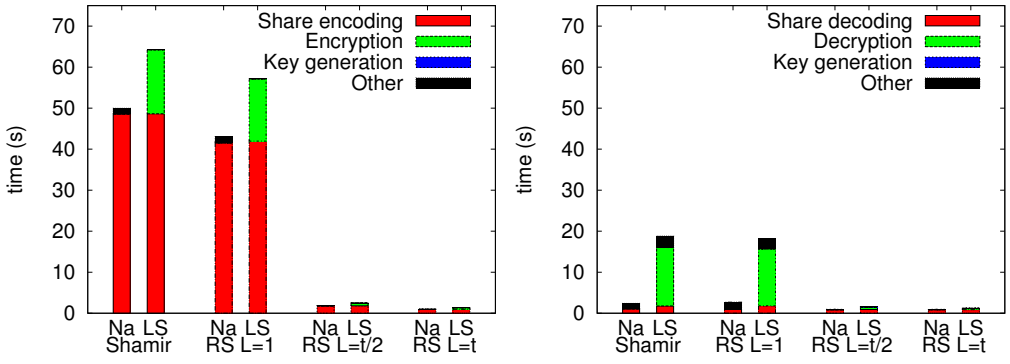
(a) Encoding for *Profile 1* (all text)

(b) Decoding for *Profile 1*



(c) Encoding for *Profile 2* (text and profile picture)

(d) Decoding for *Profile 2*



(e) Encoding for *Profile 3* (text, profile picture and bigger image)

(f) Decoding for *Profile 3*

Figure 4.4: Average times to generate 100 shares or to reconstruct secrets (from at least 50 shares). Results are shown for the naive approach (Na) and Layered scheme (LS) with selected secret-sharing schemes: Shamir's scheme and strong ramp scheme (RS) with $L \in [\lceil t/2 \rceil, t]$. Note that y-axis scales are the same for the encoding and decoding of the same profile, but not across different profiles.

Table 4.4: Average time and standard deviation (in milliseconds) required to generate shares and to decode the profile attributes using the naive approach, based on different single secret sharing schemes.

	Naive	Shamir	Ramp $\lceil L = \frac{t}{2} \rceil$	Ramp $L = t$
Profile 1	Encoding	68.26 (7.07)	47.26 (10.61)	53.11 (9.91)
	Decoding	7.55 (5.44)	16.39 (6.35)	27.39 (5.94)
Profile 2	Encoding	553.56 (16.25)	134.36 (20.13)	96.86 (17.42)
	Decoding	79.28 (17.52)	58.58 (13.38)	67.13 (13.86)
Profile 3	Encoding	49,966.38 (920.75)	1,859.33 (38.43)	1,040.2 (59.8)
	Decoding	2,394.90 (423.19)	936.89 (14.52)	946.89 (26.98)

for the share decoding process, for all three test profiles. There is also a difference in the time required to perform the single share operations, resulting from the introduction of two additional secrets (to verify the legitimacy of the Layered share and for the group ID) and from an additional share generated in every layer (for every secret).

Comparing the results for the different profiles, we can observe that the size of the secrets greatly impacts the duration of secret sharing operations, particularly when using Shamir scheme. This is the case for both Layered and naive schemes, where the share generating time varies between around 0.1 second and the decoding time is no larger than 0.04 seconds for the text-only *Profile 1* shown in Figures 4.4(a) and (b), to over 60 seconds for generating and 20 seconds for decoding for *Profile 3* that includes images, as shown in Figures 4.4(e) and (f).

When the size of secrets increases, the overhead introduced by the cryptographic operations in the Layered scheme decreases for the process of generating shares (from 70% for *Profile 1* to 31% for *Profile 3*), however it increases for the decoding process (from an already considerable 310% for *Profile 1* to a huge 594% for *Profile 3*).

Table 4.5: Average time and standard deviation (in milliseconds) required to generate shares and to decode the profile attributes using the Layered mechanism, based on different single secret sharing schemes.

	Layered	Shamir	Ramp $\lceil L = \frac{t}{2} \rceil$	Ramp $L = t$
Profile 1	Encryption	47.89 (50.45)	29.13 (51.13)	28.16 (49.22)
	Total encoding	115.78 (59.70)	76.33 (61.35)	81.72 (58.73)
	Decryption	23.42 (10.25)	5.92 (7.77)	5.36 (7.22)
	Total decoding	38.22 (21.74)	24.17 (14.95)	34.19 (13.59)
Profile 2	Encryption	285.36 (61.51)	72.15 (50.02)	54.04 (52.96)
	Total encoding	830.46 (75.75)	206.10 (68.89)	153.29 (70.50)
	Decryption	206.83 (19.45)	41.61 (10.35)	24.28 (11.05)
	Total decoding	382.20 (57.13)	118.11 (22.59)	103.28 (23.81)
Profile 3	Encryption	15,581.39 (745.44)	648.16 (56.02)	370.86 (78.67)
	Total encoding	64,267.80 (1,306.81)	2,481.95 (90.59)	1,368.77 (125.08)
	Decryption	14,248.87 (443.64)	586.07 (11.41)	309.44 (25.23)
	Total decoding	18,800.11 (939.92)	1,605.92 (24.13)	1,301.30 (59.40)

Ramp schemes consistently require less time to complete the share construction and decoding operations, consistent with the smaller share sizes when compared to Shamir scheme. Focusing on the choice of ramp scheme parameters, we consider *Profiles 2* and *3* (note the overall duration of the share generating and decoding processes is very low for *Profile 1* and, for decoding, dominated by the "other" component). We can observe that increasing L decreases both the time to generate and to encrypt the Layered shares. For the decoding process, increasing L slightly increases the time needed to decode the shares (by about 10 ms). The time to decrypt the shares is

however halved.

4.5.3.1 Usability of the Layered Scheme for Different Profile Types

We now consider the potential for real time use, as per the QoS requirements of a maximum acceptable delay of 4 seconds for the real-time OSN application, outlined in the introductory part of Section 4.5.

Considering each profile independently, we can conclude that *Profile 1* can practically be shared using any Layered scheme, as observed in Figures 4.4(a) and (b). As all schemes are usable for a profile containing only text attributes, Shamir scheme may therefore be the best solution amongst the three presented, maximizing the security without introducing unacceptable delays.

As previously noted, increasing the size of the profile (e.g., by adding images) makes the Shamir-based Layered scheme less practical for our purpose: it takes 50 s to encode *Profile 3* and 2.3 s to decode it, even without layering the shares, as shown in Figures 4.4(e) and (f). Strong ramp scheme based solutions however deliver the required efficiency: the duration of both the share generation and decoding processes and of cryptographic operations is greatly reduced in comparison to Shamir based scheme, and remains within acceptable limits for real-time use (well under 4 seconds).

There is, however, a resulting reduction in security, as the shares are not as well protected compared to the use of Shamir scheme, and an attacker capturing more than $t - L$ shares may, theoretically, still be able to learn *some* information about the secret. This motivates the use of heterogeneous single secret sharing schemes within the Layered secret sharing solution.

4.5.3.2 Enabling the Trade-off between Computational Overhead, Security and Versatility of the Supported Profiles

Considering the usability of the Layered scheme, it is evident that a ramp scheme needs to be utilised for sharing large secrets. This, however, should not impact the use

of the more secure Shamir scheme when the secret size can be practically supported. The design of Layered shares, as described in Section 4.3, allows the flexibility of using a different underlying single secret-sharing scheme. We therefore consider a *heterogeneous Layered scheme*, where the ramp schemes with a specific choice of L are utilised for large-size secrets and Shamir scheme is applied in layers that are generated from smaller secrets.

We present additional experimental results for the *heterogeneous Layered scheme*. Table 4.6 provides an example of the time required to generate shares and to decode attributes of *Profiles 2* and *3*, where Shamir scheme is used for all attributes with the exception of images, for which we select a strong ramp scheme with $L = t$. Similarly to other experiments, the average time and standard deviation are calculated based on 1500 experimental runs. We can observe that the measured computation times are comparable to the ones obtained with experiments using the Layered scheme with the underlying $L = t$ ramp scheme for each layer (around 1 second). The security for the text attributes is however maintained at the theoretical maximum offered by Shamir scheme.

4.5.4 Communication overhead

We now evaluate the communication overhead of the various schemes. We consider a cold-start scenario, where all users are yet to exchange their own Layered shares and communicate with each other to acquire the shares required to decode the profiles of other users.

We define the communication cost as the average volume of data that is downloaded and uploaded by an OSN user. We consider a user with n friends, denoted $F_1, \ldots, F_n$, who has c_i friends in common with friend $F_i, \forall i \in 1, \ldots, n$.

First, to share his own profile, the user has to send one Layered share to each of his friends. In parallel with sending his share, a user also receives one share from each of his friends. Then, the user endeavours to acquire shares related to the profiles of his

Table 4.6: Encoding and decoding times (in milliseconds) for the heterogeneous Layered scheme (using Shamir scheme for all text based secrets and a $L = t$ ramp scheme for images); we show average and standard deviation values (in brackets)

Encoding	Naive	Layered	*Decoding*	Naive	Layered
Profile 2 (incl. 128×128 profile picture)					
Share generation	125.28 (10.29)	126.66 (9.53)	Share decoding	39.90 (3.14)	45.32 (1.93)
Encryption	— —	67.72 (48.77)	Decryption	— —	40.61 (9.98)
Total encoding	130.56 (18.93)	198.31 (65.48)	Total decoding	47.39 (11.36)	100.66 (19.15)
Profile 3 (incl. profile picture & larger 520×520 image)					
Share generation	964.07 (15.11)	972.37 (14.67)	Share decoding	854.28 (1.55)	877.02 (8.31)
Encryption	— —	347.09 (56.55)	Decryption	— —	309.15 (11.00)
Total encoding	1,014.37 (36.39)	1,347.82 (90.55)	Total decoding	901.06 (12.44)	1,258.75 (22.56)

friends. Thus, for each friend F_i, the user exchanges F_i's Layered shares with friends that he has in common with F_i. This results in c_i exchanges (again, including both upload and download of shares). After contacting all mutual friends, a user transfers and receives a total of

$$N_s = n + \sum_{i=1}^{n} c_i \tag{4.11}$$

Layered shares.

We assume, for simplicity, that all users have a uniform profile size and that the profiles are shared in the same way (using identical underlying scheme parameters). The Layered shares will therefore be of the same size, that we denote as $|\mathscr{L}|$. Table 4.7 lists share size $|\mathscr{L}|$ measured in experiments, for the various combinations of parameters used in the previous sub-section. We can observe that the difference between the share sizes for the naive and Layered scheme varies quite widely. The proportion of additional bytes needed for the Layered schemes compared to the naive approach decreases with the size of the shares, starting from 180 % when the share size is less

Table 4.7: Size of a single share (kB) for the different profiles and schemes considered.

		Shamir	RS $L = \lceil \frac{t}{2} \rceil$	RS $L = t$	Heterogeneous
Profile 1	Naive	1.832	0.364	0.240	—
	Layered	1.964	0.528	0.432	—
Profile 2	Naive	36.460	7.288	3.700	5.180
	Layered	36.588	7.440	3.888	5.344
Profile 3	Naive	762.548	36.332	18.224	19.528
	Layered	762.684	36.480	18.400	19.680

than 1 kB, and a $L = 1$ ramp scheme is used, to 0.01 % when the share size is equal to 762 kB. However for the majority of cases, i.e., when the share size is greater than 1kB, the difference is in fact negligible.

To calculate a representative communications overhead, we use the Facebook example from Section 4.4.5. There, a Facebook user has, on the average, $n = 100$ friends and users have $c_i = 18$ friends in common with any of their friends. Applying these values to Equation (4.11), we calculate that a typical user will therefore have to transfer $N_s = 1900$ Layered shares. In our simplified uniform share-size scenario, each user will then have to download and upload (depending on the underlying secret sharing scheme) between 821 kB and 3.7 MB, to share their profiles for *Profile 1*, 7.4–69.5 MB for *Profile 2*, and 35.0 MB–1.5 GB for *Profile 3*.

As previously observed when evaluating the computational overhead, using Shamir scheme is only practical when all shared secrets are strings of characters. When larger secrets are shared, practical considerations necessitate a trade-off between the level of offered security and the share size. The heterogeneous Layered approach is again showing the best potential for real world use, as the communications cost is comparable to what can be achieved with sole use of the ramp scheme, around 10 MB for *Profile 2* and 37 MB for *Profile 3*.

We note that the relatively high communication cost in the presented example relates to the cold-start scenario, that assumes no prior profile sharing. In practice, following the cold start, Layered shares may be stored, and replaced only when a

user modifies their profile. For each modification, the user would have to generate and send only $n = 100$ Layered shares. This would be reflected in corresponding communications cost of between 24 and 196 kB for *Profile 1*, 389 kB–3.7 MB for *Profile 2*, and 1.8–76.3 MB for *Profile 3*). Each friend would then need to download $c_i + 1 = 19$ new Layered shares and upload $c_i = 18$ shares, resulting in a more acceptable overhead of between 16 kB and 28.2 MB.

Although we use a simplified scenario for evaluating the communication cost of the Layered scheme, we believe that the results are applicable to other more complex OSN profile sharing cases, as the basis for calculating the cost i.e., the individual Layered share size, will remain unchanged. Overall, our results show that the use of the Layered scheme does not introduce a significant overhead in the size of the shares when larger profiles, that include images, are shared within an OSN group. Moreover, the use of ramp schemes in place of Shamir secret-sharing enables a significant reduction of the size of each share (by 73–95 % using a reasonably secure $L = \lceil \frac{t}{2} \rceil$ ramp scheme parameters). Finally, a good trade-off between security and communication cost can be achieved by using a hybrid Layered scheme, that uses a secret-size based customised combination of Shamir secret-sharing and ramp schemes.

4.6 Discussion

4.6.1 There is no Free Lunch

Providing meaningful and automated access policies, while keeping all the information private and without relying on an OSN provider is a difficult problem. Our Layered scheme proposes one way to address this, based on the number of common friends that a dealer has with any user in a specific OSN circle. We acknowledge that this criteria may not be universally applicable to all OSN groups. However, we believe, as per [119], that the number of common friends represents a relevant and easy to understand metric for the OSN users (note that this is a common component of the

algorithms for automated clustering or the creation of communities).

The current proposal does not specifically address revocation (un-friending) and it is assumed that the Layered shares would need to be re-generated and redistributed to friends in a group when a user's group membership is revoked. A similar problem was identified in the *Persona* proposal [3], where upon revocation of the social link, the dealer needs to renew the cryptographic keys for all other friends in the OSN group. A possible way to address this is by using proxy re-encryption [53], although this solution introduces a third party to enable the revocation mechanism, which is not fully in line with our design goals.

In an alternative solution, the profile owner could provide all remaining participants with a way to compute new Layered shares from a small quantity of new information and their original Layered shares. An example would be for the profile owner to distribute a newly generated alternative of the first two protocol layers, along with the original and newly generated second-layer keys. Each remaining friend would thus be able to generate their new Layered share by removing the first two layers of their original Layered share, decoding the remainder using the original key, re-encoding it with the newly generated key, and concatenating the two new layers. The impact of this solution on system security needs to be studied in more depth. One issue is that the revoked participants still own real Layered shares containing information about the original profile, however this solution would ensure that this old share cannot be combined with the new ones. Another issue is that during the re-encoding, the remaining friends would directly acquire in the clear the first layer of the information contained in the profile, without having to exchange Layered shares to obtain the decoding key. An option would be for the profile owner to distribute a combination of the old and new decoding keys, allowing the re-encoding of layers without prior decoding. An additional encryption scheme may be needed to support this functionality.

The second drawback of the current proposal also relates to dynamicity of the

mechanism, *i.e.*, handling of attribute changes (*e.g.*, when a user moves to a new location, or changes their relationship status). This again necessitates refreshing of the Layered shares for all users. An extension of our scheme to handle this is a subject for future study.

4.6.2 Use of Other Secret-Sharing Schemes

Other secret-sharing schemes have been proposed in this very active research field, that may be suited for a specific purpose, *e.g.*, [51] which presents a strong ramp scheme with general access structures. The Layered secret-sharing may utilize any other secret-sharing scheme to build a (selected) layer, assuming that the scheme satisfies the following requirements. First, the chosen scheme should only rely on private shares, as public shares would leak information, *e.g.*, about the number of secrets. Second, the user who has decoded a specific secret should be able to generate (at least one) additional share, that is required to generate the encryption key and progress to the next layer.

4.7 Conclusion

Multiple secret-sharing schemes enable concurrent sharing of a number of secrets, however they still expose information about the strength of protection that may be utilized for re-identifying the dealer, and about the number of secrets shared. We have proposed a new Layered multiple secret-sharing scheme that fully preserves privacy of the data shared by the dealer, by protecting the number of secrets he is sharing and their thresholds. We consider the use of this scheme for profile sharing in OSNs, that would enable direct access to selected attributes of the profile by friends, with access levels determined by the number of friends in common with the profile owner. We evaluate the security of the proposed scheme and analyse the level of threat posed by OSN users who are on the social graph of the user sharing his profile. Finally, we evaluate the communication overhead and the complexity of critical components of

the scheme, by experimental evaluation of the time required to generate and decode Layered shares. Future work includes extensions to handle revocation of shares (unfriending) and changes to the relationship status, without the need to re-distribute new shares.

Chapter 5

Systematic MDS Code and Strong Ramp Schemes

5.1 Introduction

Data from both individuals and businesses is increasingly collected, aggregated and analysed to provide new services.[1] There is a corresponding research effort to enable both storage and processing of such data in a secure and privacy-preserving way, in line with the increasing public concerns and strict regulatory requirements for the protection of such data.[2] MPC [24] is a mechanism by which a number of parties can collaborate to compute an agreed function of their inputs, ensuring both confidentiality of the data and the integrity of the resulting output. Private computations over distributed data are applicable in many scenarios, allowing multiple organizations to jointly utilize their private or business confidential data to provide a service (e.g., Internet Service Providers troubleshooting network outages [30]), and enabling processing of personal data stored on individuals' mobile devices [49].

MPC is typically based on secret sharing or garbled circuits. In this chapter we

[1] McKinsey report on big data, http://www.mckinsey.com/insights/business_technology/big_data_the_next_frontier_for_innovation

[2] EU data protection rules reform, http://ec.europa.eu/justice/newsroom/data-protection/news/120125_en.htm

focus on MPC with secret sharing, which is better suited to scenarios with a number of parties (garbled circuit being predominantly used for two-party computation [81]). The confidentiality of the computation is guaranteed by the security of the underlying secret sharing scheme.

The choice between a perfect secret sharing scheme such as Shamir is and a strong ramp scheme is a trade-off between the information-theoretical security guarantee offered by a perfect scheme against the efficiency and decrease of communication cost provided by a strong ramp scheme. However a strong ramp scheme guarantees a higher level of security compared to any generic ramp scheme, as discussed in the Section 2.3.1. The security is measured using information theoretical metrics, relying on entropy properties.

As discussed in 2.3.1.3, despite the promise of strong ramp schemes, there is limited prior work on construction methods for such schemes [4]. Prior works have also studied links between generic ramp schemes and MDS codes [25] [22]. For example, the authors of [85] proposed a construction method for threshold schemes, which is based on MDS codes. However, such methods are not applicable to construction of strong ramp schemes. Altogether, only a few constructions for strong ramp schemes have been proposed and, to the best of our knowledge, there has been no evaluation of the performance of such schemes in MPC. In this chapter we aim to address these deficiencies. The summary of our contributions is as follows.

First, we present a novel method for constructing strong ramp schemes based on any (systematic or non-systematic) MDS code. To this end, we demonstrate that the encoded packets (the redundancy elements) from well-known systematic MDS error/erasure correcting codes [65] satisfy the definition of a strong ramp scheme. Utilizing results from [59], we provide a method to build a strong ramp scheme from any MDS code. For completeness, we also show how MDS codes may be constructed from any ramp scheme.

We then present a second novel construction for strong ramp schemes, derived from

the Shamir scheme. The method uses an alternative (but equivalent to the original [97]) construction of shares from points of a polynomial, rather than the polynomial coefficients.

Finally, we evaluate the practical benefits of strong ramp schemes in an MPC application for network outage monitoring. Our evaluation relies on implementations of (a) a strong ramp scheme based on the Reed-Solomon MDS code [90] and (b) a strong ramp scheme based on the Shamir secret sharing scheme, both integrated into the SEPIA MPC framework [17]. We consider a realistic setting with 20 input and 20 privacy peers processing Internet Service Provider (ISP) outage monitoring data, and show that strong ramp schemes can reduce the computation overhead (measured by CPU time) by around 44 times and the communication cost per MPC input peer by 20 times, compared to the performance of the baseline Shamir scheme.

5.2 Background and Related Work

In Section 2.3, we have presented the different notions of threshold secret sharing scheme, ramp scheme and strong ramp scheme. In Section 2.2.1, we provided the definitions of the MDS and systematic properties of block codes. In this section, we will present equivalent definitions based on their generator matrices which are better suited for the remainder of this chapter.

Error correction codes are used to recover information in lossy (error or erasure) environments. Linear codes are constructed using a linear combination of information elements. In a (t, N) code, a vector $\vec{S}$ of t information elements (packets) is encoded into a vector (codeword) of $N \geq t$ packets using a generator matrix G of size $N \times t$ (i.e., the codeword is obtained via $\vec{C} = G \cdot \vec{S}$). An MDS code is a (t, N) linear code that achieves the Singleton bound [88], with the pairwise difference d between vector elements of any two codewords $d = N - t + 1$. Therefore, it can correct up to $(N-t)/2$ errors and $(N - t)$ erasures. An alternative definition is as follows.

Definition 12 (MDS Code)**.** *A (t, N) code is MDS if any square sub-matrix of size t derived from its generator matrix $G_{N\times t}$ is non-singular.*

We note that all coding operations are defined on a Galois field $\mathbb{GF}$, based either on a prime number q, i.e., $GF(q)$, or a power of a prime $GF(q^p)$, with the latter being most commonly used in traditional coding applications. Reed-Solomon (RS) codes are the best known MDS codes, with a generator matrix that can be based on either Vandermonde or Cauchy matrices [90], [86].

Definition 13 (Systematic MDS Code)**.** *A (t, N) MDS code is* systematic *if every codeword of N elements includes the (non-transformed) t elements comprising the vector $\vec{S}$. I.e., the identity matrix of size t, I_t, is a submatrix of the generator matrix G.*

Systematic MDS codes have found the largest number of practical uses in a number of applications [86]. All MDS codes based on a Cauchy matrix are systematic. We note that MDS codes based on a Vandermonde matrix are not always systematic, however, using the algorithm proposed by Lacan and al. [59], non-systematic codes can be transformed into a systematic variant.

Theorem 1 ([59])**.** *Let A be a $t \times t$ matrix of rank t and B a $(N-t)\times t$ matrix of rank $min(N-t, t)$ such that any square submatrix of size t from $\binom{A}{B}$ has a rank equal to t (i.e., $\binom{A}{B}$ generates a MDS code). Then the matrix $\binom{I_t}{B\cdot A^{-1}}$ generates a (t, N) systematic MDS code.*

5.3 Links between Ramp Schemes and MDS Codes

Despite the promise of strong ramp schemes, we discussed in Section 2.3.1.4 that only few constructions of these schemes were available, and usually only theoretically. However, it is proven that ramp schemes and MDS codes are closely related. Noticing that strong ramp schemes are a sub-category of ramp schemes, we wonder to which

Table 5.1: Notations used to describe the ramp schemes and MDS codes

Notation	Description
n	Number of shares generated by a secret sharing scheme, equivalent to the number of privacy peers
t	Security threshold for secret sharing schemes, equivalent to the minimum number of packets required to decode an erasure code
L	$t - L$ being the second security threshold for ramp schemes
N	Codeword size (number of packets) for an erasure code
$G_{N\times t}$	Generator matrix of a (t, N) MDS code.
$\vec{S} = (S_1, \ldots, S_L)^\intercal$	Secret, a vector of L elements (packets)
$\vec{E} = (E_1, \ldots, E_n)^\intercal$	Vector of n encoded elements (packets), or a vector of n shares generated by a secret sharing scheme
$\vec{V} = (S_1, \ldots, S_L, r_1, \ldots, r_{t-L})^\intercal$	Vector secret padded with random elements (used in the process of generating the shares)
$M_{\{i_1,\ldots,i_j\}}$	The submatrix of any matrix M built from its rows $i_1, \ldots, i_j$
I_t	Identity matrix of size t
$0_{i,j}$	Zero matrix of size $i \times j$

sub-category of MDS codes they were related. It follows that in this section, we provide proofs for the links between ramp schemes and MDS codes. Our main goal is to to enable the practical use of strong ramp schemes in MPC, by providing construction methods that will utilize the systematic MDS codes and their (available) efficient implementations. We also derive a method to construct a strong ramp scheme from Shamir scheme. Table 5.1 lists the notations used throughout this section.

5.3.1 Deriving a Strong Ramp Scheme from a Systematic MDS Code

In this section, we will prove that strong ramp schemes can be derived from systematic MDS codes. To do so, we will first describe how a generator matrix $G_{N\times t}$ of a systematic (t, N) MDS code can be used to generate $n = N - L$ shares out of L secrets. According to Definition 13, we can assume without loss of generality that the matrix G is of the form $G = \binom{I_t}{A}$, where I_t is the $t \times t$ identity matrix. We will then prove that this construction defines a strong ramp scheme. In what follows we will use $0_{m,n}$ to denote a $m \times n$ zero matrix, and $M_{\{i\}}$ to denote the i'th row of a matrix M.

Algorithm 1 describes how a generator matrix $G_{N\times t}$ of a (t, N) systematic MDS code can be used to generate shares from a secret vector $(S_1, \ldots, S_L)$, where $L \leq min(t, N-t)$. To this end, a submatrix $R_{(N-L)\times t}$ is obtained by taking the last $N-L$ rows of G. Then, the secret vector $\vec{S}$ is extended to a vector $\vec{V}$ of length t by appending to it $t-L$ random values. Finally, the result of $R \cdot \vec{V}$ is a vector that constitutes the $N-L$ shares, where each share is associated with the respective row of R that generated it (i.e., $E_i = R_{\{i\}} \cdot \vec{V}$). The matrix R and the assignment of rows $R_{\{i\}}$ to participants and their shares are assumed to be public knowledge.

Any subset $\vec{E'} = \{E_{i_1}, \ldots, E_{i_t}\}$ of t shares, associated with a subset of the rows of R, is sufficient to reconstruct the vector $\vec{V}$: The rows $R_{\{i_1\}}, \ldots, R_{\{i_t\}}$ form a square matrix R', which is guaranteed to be non-singular since it is a submatrix of G, a generator matrix of MDS code. Therefore, there is only a single solution to $R' \cdot \vec{X} = \vec{E'}$, and since $\vec{V}$ is a valid solution, necessarily $\vec{X} = \vec{V}$, and the first L elements of this vector are the recovered shared secrets.

Algorithm 1: Share generation from a systematic MDS code

1: **procedure** SRS$(G, (S_1, \ldots, S_L)^\intercal)$ ▷ Generation of $n = N - L$ shares
2: $R \leftarrow G_{\{L+1,\ldots,N\}}$ ▷ R submatrix from G without the L first rows
3: Let $r_1, \ldots, r_{t-L}$ be $t-L$ random iid values.
4: $\vec{V} \leftarrow (S_1, \ldots, S_L, r_1, \ldots, r_{t-L})^\intercal$
5: $\vec{E} \leftarrow R \cdot \vec{V}$
6: **return** $n{=}N{-}L$ shares where share i is E_i and is associated with the row $R_{\{i\}}$.
7: **end procedure**

Theorem 2. *Let G be a generator matrix of a (t, N) systematic MDS code. Then, for any $L \leq min(t, N-t)$, the share generation algorithm $SRS(G, \cdot)$ (Algorithm 1) is a $(t, L, N-L)$ strong ramp scheme.*

We note that the addition of $t-L$ random values to the vector $\vec{V}$ matches the lower bound for the randomness that is required to obtain a ramp scheme [13].

Proof. Let $\vec{S} = (S_1, \ldots, S_L)^\intercal$ be a vector consisting of L secrets to share. Algorithm 1

extracts from the matrix G the submatrix $R = \left(\begin{smallmatrix} 0_{t-L,L} & | & I_{t-L} \\ & A & \end{smallmatrix}\right)$. Recall that $\vec{V} = (S_1, \ldots, S_L, r_1, \ldots, r_{t-L})^\intercal$. Therefore, the computation $\vec{E} = R \cdot \vec{V}$ in line 5 results in $N - L$ shares of the form $\vec{E} = (E_1, \ldots, E_{t-L}, \varepsilon_1, \ldots, \varepsilon_{N-t})^\intercal$, where $E_i = r_i$ and $\varepsilon_j = A_{\{j\}} \cdot \vec{V}$. We will next show that the described scheme maintains the three conditions of Definition 10.

1. For any $x < t - L$, $H(\vec{S}|E_{i_1}, \ldots, E_{i_x}) = H(\vec{S})$: According to entropy properties, $H(\vec{S}|E_{i_1}, \ldots, E_{i_x}) \leq H(\vec{S})$ always holds. We will show that for $x = t - L$, $H(\vec{S}|E_{i_1}, \ldots, E_{i_{t-L}}) = H(\vec{S})$. Since for any $x < t - L$, $H(\vec{S}|E_{i_1}, \ldots, E_{i_x}) \geq H(\vec{S}|E_{i_1}, \ldots, E_{i_{t-L}})$ (adding known information can only reduce entropy), this will also prove $H(\vec{S}|E_{i_1}, \ldots, E_{i_x}) \geq H(\vec{S})$.

To prove $H(\vec{S}|E_{i_1}, \ldots, E_{i_{t-L}}) = H(\vec{S})$, we will show that any possible set of secrets $\vec{S'}$ is consistent with the $t-L$ shares, i.e., we will find a vector $\vec{V'}$ that extends $\vec{S'}$ such that $E_i = R_{\{i\}} \cdot \vec{V'}$ holds for all the provided shares. To find this vector, we construct a square matrix B as a submatrix of G, in the following way: the first L rows of B will be the first rows of G, which are of the form $(I_L|0_{t-L,L})$. The remaining $t - L$ rows of B will be the rows of R corresponding to the given shares, i.e., $R_{\{i_1\}}, \ldots, R_{\{i_{t-L}\}}$. To summarize, the matrix B is of the form:

$$B = \begin{pmatrix} I_L|0_{t-L,L} \\ R_{\{i_1,\ldots,i_{t-L}\}} \end{pmatrix} \quad . \tag{5.1}$$

Next, consider the following problem:

$$B \cdot \vec{X} = (S'_1, \ldots, S'_L, E_{i_1}, \ldots, E_{i_{t-L}})^\intercal \quad . \tag{5.2}$$

Since matrix B is a square submatrix of size t of G, it is non-singular, and therefore there exists one and only one solution for $\vec{X}$, which we will denote $\vec{V'}$. The selection of the first rows of B ensures that the first elements in $\vec{V'}$ are $S'_1, \ldots, S'_L$, so $\vec{V'}$ extends $\vec{S'}$. The selection of the remaining rows of B ensures that $\vec{V'}$ is also consistent with

the shares: for example, for the share E_{i_1} (and any of the other given shares), there is a row j in B such that $B_{\{j\}} = R_{\{i_1\}}$, and therefore $R_{\{i_1\}} \cdot \vec{V'} = B_{\{j\}} \cdot \vec{V'} = E_{i_1}$ as required. To summarize, we showed that any vector of secrets is consistent with a subset of $x = t - L$ shares, so the entropy of the secret vector is not reduced given the shares. This extends also to any smaller subset of shares.

2. For any $t - L \leq x < t$, $H(\vec{S}|E_{i_1}, \ldots, E_{i_x}) = H(S_{j_1}, \ldots, S_{j_{t-x}}|E_{i_1}, \ldots, E_{i_x}) = \frac{t-x}{L} H(\vec{S})$ for any set of indices $j_1, \ldots, j_{t-x}$: Since all the elements of the vector $\vec{S}$ are independent,

$$H(\vec{S}) = \sum_{i=1}^{L} H(S_i) = L \cdot H(S_j) \text{ for any } j \in [1, L]. \tag{5.3}$$

Therefore, $H(S_{j_1}, \ldots, S_{j_{t-x}}) = \frac{t-x}{L} H(\vec{S})$. In addition, because of arguments similar to those used to show $H(\vec{S}|E_{i_1}, \ldots, E_{i_{t-L}}) = H(\vec{S})$ in the first part of the proof, it follows that

$$H(S_{j_1}, \ldots, S_{j_{t-x}}|E_{i_1}, \ldots, E_{i_x}) = H(S_{j_1}, \ldots, S_{j_{t-x}}) \ ,$$

and therefore $H(S_{j_1}, \ldots, S_{j_{t-x}}|E_{i_1}, \ldots, E_{i_x}) = \frac{t-x}{L} H(\vec{S})$.

It remains to show that $H(\vec{S}|E_{i_1}, \ldots, E_{i_x}) = \frac{t-x}{L} H(\vec{S})$. Based on the properties of conditional entropy:

$$\begin{aligned} H(\vec{S}|E_{i_1}, \ldots, E_{i_x}) = & H(S_1, \ldots, S_{t-x}|E_{i_1}, \ldots, E_{i_x}) \\ & + H(S_{t-x+1}, \ldots, S_L|E_{i_1}, \ldots, E_{i_x}, S_1, \ldots, S_{t-x}) \ . \end{aligned} \tag{5.4}$$

We have shown that $H(S_1, \ldots, S_{t-x}|E_{i_1}, \ldots, E_{i_x}) = \frac{t-x}{L} H(\vec{S})$. Consider the matrix $B = \binom{I_{t-x}|0_{t-x,t}}{R_{\{i_1,\ldots,i_x\}}}$. It is a square submatrix of G of size t, and therefore non-singular. Consequently, the linear system

$$B \cdot \vec{X} = (S_1, \ldots, S_{t-x}, E_{i_1}, \ldots, E_{i_x})^\intercal$$

has the unique solution $\vec{X} = \vec{V}$, i.e., the set $S_{t-x+1}, \ldots, S_L$ can be reconstructed with a probability of 1 given $S_1, \ldots, S_{t-x}, E_{i_1}, \ldots, E_{i_x}$, so

$$H(S_{t-x+1}, \ldots, S_L | E_{i_1}, \ldots, E_{i_x}, S_1, \ldots, S_{t-x}) = 0 \ .$$

We can finally conclude that $H(\vec{S}|E_{i_1}, \ldots, E_{i_x}) = \frac{t-x}{L} H(\vec{S})$.

3. For any $x \geq t$, $H(\vec{S}|E_{i_1}, \ldots, E_{i_x}) = 0$: This follows immediately from the property that any subset $\vec{E'} = \{E_{i_1}, \ldots, E_{i_t}\}$ of t shares is sufficient to reconstruct the vector $\vec{V}$ with probability 1.

□

Corollary 2. *More generally, from any (t, N) systematic MDS code, a (t, L, n) strong ramp scheme can be derived with $n = N - L$ and $L \leq min(t, N - t)$.*

5.3.2 Additional Links between Ramp Schemes and MDS Codes

Any MDS code can be used to derive a systematic MDS Code: according to Theorem 1 [59], if G is a generator matrix of a (t, N) MDS code, and $G_{\{1,\ldots,t\}}$ is the submatrix built from the first t rows of G, then the matrix $G' = G \cdot {G_{\{1,\ldots,t\}}}^{-1}$ is a generator matrix of a (t, N) systematic MDS code. Combining this with the construction described in the previous section leads to the following corollary:

Corollary 3. *A (t, L, n) strong ramp scheme with $n = N - L$ and $L \leq min(t, N - t)$ can be constructed from any (t, N) MDS code.*

Any strong ramp scheme is, by definition, a ramp scheme. In addition, setting $L = t$ for a (t, L, n) linear ramp scheme results in a (t, N) MDS code, as any t or more non-corrupted encoded elements (shares) allow recovery of the information elements (secrets). These observations, together with the former results, allow the construction of any of the schemes starting from any other reference point. For example, based on the previous results, it is easy to show that:

Corollary 4. *A (t, N) systematic MDS code with $N = n$ can be derived from a (t, L, n) strong ramp scheme with $L = t$.*

Corollary 5. *A (t, L', n') strong ramp scheme with $n' = n - L'$ and $L' \leq min(t, n - t)$ can be derived from a (t, L, n) ramp scheme with $L = t$.*

Figure 5.1 summarizes the links between the different schemes using the construction methods.

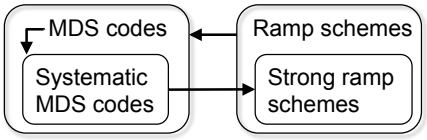


Figure 5.1: Links between ramp schemes and MDS codes

5.3.3 Deriving a Strong Ramp Scheme from Shamir Scheme

We present an additional method to construct a strong ramp scheme, starting from Shamir secret sharing. As we will show in the next section, this scheme enables additional computation capabilities, in line with Shamir scheme (i.e., not limited by Galois Field $GF(2^q)$ operations that are most commonly used for Reed Solomon codes).

We start by observing a different construction method for Shamir secret sharing, which is equivalent to the one presented in Section 2.3.1.2. A polynomial of degree $t - 1$ can be uniquely defined by its t coefficients (as described in Section 2.3.1.2), but it can also be uniquely defined by t of its points. In Shamir's scheme, only one secret is shared, thus only the constant coefficient, which is also the point of polynomial at the abscissa 0, is fixed. The other coefficients are then chosen randomly. But as mentioned in [51], a ramp scheme based on a polynomial defined by its coefficients is not a strong ramp scheme. Thus in the following we will explain the steps to construct a strong ramp scheme using a polynomial defined by its points (rather than its coefficients), and we will prove that it is a strong ramp scheme.

Let $\vec{S} = (S_1, \ldots, S_L)^\intercal$ be the secret vector to share and let $r_1, \ldots, r_{t-L}$ be $t - L$ random values. The polynomial Q of degree $t - 1$ is defined uniquely by the t points $(x_0, Q(x_0) = S_1), \ldots, (x_{L-1}, Q(x_{L-1}) = S_L), (x_L, Q(x_L) = r_1), \ldots, (x_{t-1}, Q(x_{t-1}) = r_{t-L})$, where $x_0, \ldots, x_{t-1}$ are all distinct. From these t points, the value of the polynomial Q can be interpolated for any x: $Q(x) = \sum_{i=0}^{t-1} Q(x_i) \prod_{j \neq i} \frac{x - x_j}{x_i - x_j}$. Then, as in Shamir's scheme, shares are derived from the polynomial by computing new points. Algorithm 2 summarizes the process in which n shares are created. Given any subset of t of these shares, the polynomial Q can be reconstructed and the secret vector is given by $Q(x_0), \ldots, Q(x_{L-1})$.

Algorithm 2: Strong ramp scheme from Shamir scheme

1: **procedure** SSRS($(S_1, \ldots, S_L)^\intercal$) ▷ Generation of n shares
2: Let $r_1, \ldots, r_{t-L}$ be $t - L$ random iid values.
3: Choose $n + L$ distinct values $x_0, \ldots, x_{n+L-1}$.
4: Let $Q(x_{i-1}) = S_i$ for $i \in [1, L]$, and $Q(x_{L+i-1}) = r_i$ for $i \in [1, t - L]$.
5: Interpolate $Q(x)$ for $x_t, \ldots, x_{n+L-1}$ with $Q(x) = \sum_{i=0}^{t-1} Q(x_i) \prod_{j \neq i} \frac{x - x_j}{x_i - x_j}$.
6: **return** the shares $(x_L, Q(x_L)), \ldots, (x_{n+L-1}, Q(x_{n+L-1}))$.
7: **end procedure**

The property that the scheme described in Algorithm 2 is a strong ramp scheme follows from Theorem 2: the linear scheme defined by the points $(x_0, Q(x_0) = S_1), \ldots, (x_{L-1}, Q(x_{L-1}) = S_L), (x_L, Q(x_L)), \ldots, (x_{n+L-1}, Q(x_{n+L-1}))$ is a (t, N) systematic MDS code with $N = n + L$, since the polynomial can be interpolated from any t points, and the original information packets are the first points $Q(x_0), \ldots, Q(x_{t-1})$. Thus, puncturing (i.e. removing) the L elements related to the secrets generates a strong ramp scheme.

5.4 MPC with Strong Ramp Schemes

Linear secret sharing schemes can be used for secure evaluation of both arithmetic operations (e.g., addition, multiplication, etc.) and logical operations like comparison. A number of practical MPC implementations based on Shamir secret sharing have

been developed in recent years, including [17] and [27]. We presented in Section 2.4 a background on MPC.

Strong ramp schemes theoretically have the same computational capabilities as the Shamir scheme, as both belong to the large family of linear schemes. However, in practice, and specifically for secret sharing based on MDS erasure codes, the type of Galois field that the codes are based on will determine the operations that may be applicable using a specific scheme. The schemes based on $GF(q)$, with q being a large prime number, will allow integer based arithmetic and logical operations, which, with an equivalent and sufficiently large value of q, will provide the same functionality as the Shamir scheme that is based on the same q. $GF(2^p)$, on the other hand, limits the applicable operations to bit-wise XOR. We note that the choice of $\mathbb{GF}$ does not impact the secret sharing capabilities of a scheme, as any secret can be represented as a bit string and secret sharing only provides the mechanism to recover this string by combining shares, rather than the ability to manipulate the shares derived from multiple secrets.

A large research effort in recent years has resulted in efficient implementations of not only traditional $GF(2^p)$ based error and erasure codes (mostly implemented in bespoke hardware), but also of more generic cryptographic mechanisms that are based on $GF(q)$ (with large q values), like elliptic curve signatures [94]. This provides a strong foundation for practical implementations of efficient strong ramp schemes.

5.5 Implementation in SEPIA

We implemented and evaluated some selected schemes using SEPIA [17], an efficient MPC library written in Java. SEPIA assumes a semi-honest adversary model, and uses a fully distributed architecture: it is optimized for parallel execution and specifically designed to aggregate network events and statistics from multiple domains.

Within SEPIA, the `mpc` library package contains an implementation of the Shamir secret sharing scheme and the corresponding operations. We extended SEPIA by

implementing two strong ramp schemes within the `mpc` package: the first based on the Shamir scheme and the second based on MDS codes. Furthermore, we added the implementation of the valid operations using the respective new secret sharing schemes. Finally, we also added new `Primitives` classes, each corresponding to the additional secret sharing scheme, and the abstract classes defining the MPC peer and protocol.

Our implementation of the ramp scheme that relies on MDS codes is based on the Reed-Solomon (RS) code implementation from Google ZXing [3], an open source 1D/2D barcode image processing software written in Java. We modified ZXing to include an erasure RS decoder (the original decoder could only handle errors), as an RS erasure correction code is required for constructing the strong ramp secret sharing scheme. The ramp scheme based on Shamir secret sharing was a bespoke implementation.

We note that the RS code implementation is based on $GF(2^8)$, thereby limiting the available MPC operations to those based on bit-wise XOR. However, we still consider this as a good basis for performance evaluation of an MDS code based strong ramp scheme, since the choice of $\mathbb{GF}$ does not impact the encoding and decoding mechanisms used in the implementation of secret sharing. Extension of the current implementation to include operations on $GF(q)$ is planned for future work.

5.6 Performance Evaluation

We evaluated the performance of the different secret sharing schemes in MPC, as implemented in SEPIA. As performance metrics, we use (a) the computation overhead of the different schemes, represented by the time needed to do a set of specific tasks (perform a computation) by a peer, and (b) the communication overhead, represented by the number of bytes received and/or sent by the MPC participants.

We simulate a network outage detection scenario from [30], where Internet service providers (ISPs) perform traffic data aggregation in a privacy-preserving way, to obtain additional information that can help determine the root cause of outages [31]. In MPC,

[3] ZXing (Zebra Crossing). https://github.com/zxing

such aggregation can be done using the multiset union operation, based on a Counting Bloom Filter (CBF) [30]. The process involves m ISPs (input peers) and n privacy peers, and proceeds as follows:

1. The input of each ISP (an input MPC peer) consists of a number of unreachable IP addresses.

2. The ISP creates a CBF based on the unreachable destinations, which will be the input for the multi-party computation.

3. The ISP generates n shares of the CBF and distributes them to the privacy peers.

4. Each privacy peer performs the multiset union by adding the corresponding array elements of the CBF in the shares it received and sends the combined share to all the other peers.

5. Each privacy peer reconstructs the aggregated CBF from the combined shares, and sends it to the input peers.

6. Each input peer can check the CBF locations corresponding to its inputs, and deduce whether the respective outage is local or global.

All our experiments were executed on an OpenStack cloud that includes six workers, with each worker being allocated 12 CPU cores of Intel Xeon X5650 2.67GHz processor and a Gigabit LAN connection. Due to system limitations, we only used ten virtual machines (VMs), where each machine has 2GB memory and one virtual CPU based on KVM virtualization. Each machine runs Ubuntu 12.10 clouding amd64. The input peers and privacy peers are installed on these virtual machines and their operation is distributed uniformly across the ten machines.

We evaluated the performance of three secret sharing schemes: Shamir secret sharing (the original scheme implemented in SEPIA), a strong ramp scheme based on Shamir scheme ($Ramp{-}Sh$) and a strong ramp scheme based on RS code ($Ramp{-}RS$), for a range of parameter values: the number of privacy peers n, number of input peers

m, the secret sharing threshold t and the second security parameter L for ramp schemes (which directly impacts the share size reduction compared to Shamir scheme). We note that the maximum value of n used in our experiments is 20, to ensure a manageable computation time. We also note that *Ramp-Sh* with $L = 1$ amounts to (more efficient) implementation of the standard Shamir scheme, which directly uses random numbers as shares (rather than polynomial coefficients), as detailed in Section 3.4.

For the experiments, we utilized the input data parameters from [30], where each input peer shares $2,114$ unreachable IP addresses; this corresponds to the highest number of unreachable IP addresses collected in the SWITCH network[4] during Hurricane Sandy. This input data is then converted to a CBF of size 131,072 (around $1.05MB$ of data) and split into manageable secret sizes ($4B$ for Shamir and $Ramp - Sh$ schemes and $1B$ for $Ramp - RS$). Furthermore, each experiment consists of 10 computation rounds using different (randomly generated) input data, to remove any data related variation in the processing requirements.

5.6.1 Computation Overhead

To evaluate the computation overhead, we measured the CPU time for the specific components of the overall computation: the input peers' share generation time (step 3), the privacy peers' multiset union computation time (step 4), and the privacy peers' secret reconstruction time (step 5). Table 5.2 presents the average CPU time per peer, for the three operation components, using different secret sharing schemes and for m, n and t value of 20. The share generation time is averaged over all input peers, while the secure computation time and the time to reconstruct the result are averaged over all privacy peers.

Considering the overall computation time (both for input and privacy peers), we observe that the original Shamir scheme is outperformed by all other schemes, including our improved implementation (*Ramp-Sh*, $L = 1$ scheme) that ensures identical security

[4] http://www.switch.ch

Table 5.2: The average CPU time per peer (in millisecond) to complete all the steps in the multiset union operation, with $m = 20$ input peers, $n = 20$ privacy peers and secret sharing threshold $t = 20$.

Scheme	Share Gen.	Computation	Reconstruction	Total
Shamir	15,649.29	96.24	1,093.19	16,838.72
Ramp-Sh, L=1	5,397.41	98.51	1,083.88	6,579.80
Ramp-Sh, L=5	1,352.92	18.06	1,003.75	2,374.73
Ramp-Sh, L=10	745.81	10.23	939.35	1,695.39
Ramp-Sh, L=15	433.16	6.47	940.22	1,379.85
Ramp-Sh, L=20	222.78	5.48	943.37	971.63
Ramp-RS, L=1	1,127.53	66.24	5,626.39	6,820.16
Ramp-RS, L=5	179.85	6.38	1,253.85	1,440.08
Ramp-RS, L=10	73.45	5.09	657.39	735.93
Ramp-RS, L=15	39.30	3.18	452.29	494.77
Ramp-RS, L=20	25.26	2.86	356.22	384.34

guarantees. For $L = 20$, the overall computation time using *Ramp-RS* scheme is nearly 40 times lower than the CPU time in the original Shamir scheme, reducing it from around $17s$ to less than $0.4s$. The *Ramp-Sh* scheme also provides a significant reduction in CPU time, with around $1s$ total computation time.

Increasing L results in a lower computation overhead for share generation in both ramp schemes, as this is equivalent to having a lower number of share generation rounds (as multiple secrets are used to generate shares). The secret reconstruction time is quite stable for different L values for the *Ramp-Sh* scheme, while it is significantly higher for lower L values for the *Ramp-RS* scheme. This is due to the overhead introduced in the *Ramp-RS* reconstruction (decoding) step, where both the useful data and padding (as described in the construction method in Section 3.2) need to be decoded. For the *Ramp-Sh* scheme, only the useful data needs to be reconstructed.

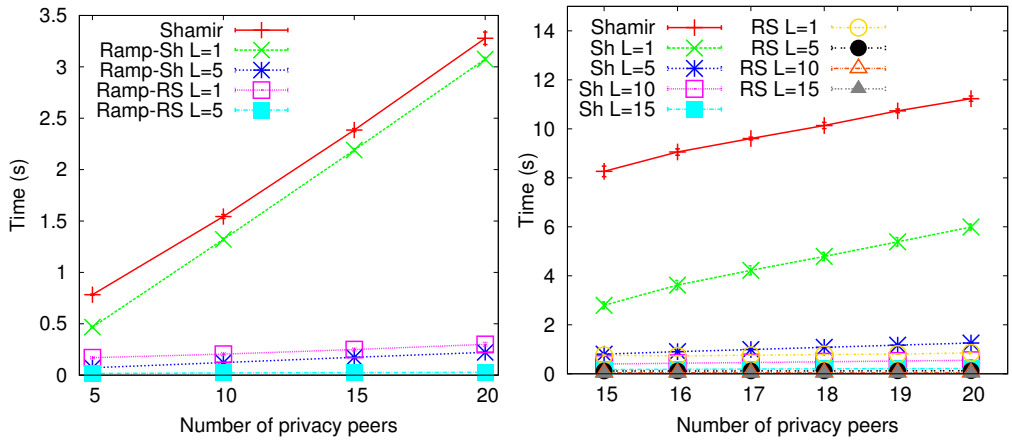
For all schemes, the duration of the secure computation component is significantly lower than the time required to generate the shares or to reconstruct the result. We therefore concentrate on the two latter metrics and further explore the impact of various parameters. We also note our observation, from extensive experiments, that varying the number of input peers has limited impact on the measured CPU time per

peer. Therefore, in the remainder of this section, the presented results are averaged over a varying number of input peers (for m between 5 and 20).

We now investigate how increasing the number of privacy peers n (and respectively the number of shares to generate and reconstruct) affects the computation time for different secret sharing schemes. Figures 5.2(a) and 5.2(b) show the average CPU time for an input peer to generate shares, while Figures 5.3(a) and 5.3(b) show the average CPU time that a privacy peer needs to reconstruct the result. The results are measured over n between 5 and 20, with $t = 5$ and $t = 15$, and for different values of L. As expected, the CPU time increases as the number of privacy peers increases, as a larger number of privacy peers corresponds to a higher number of shares to generate and reconstruct. There is a performance benefit from using ramp schemes, similar to that observed in Table 5.2, for both share generation and reconstruction time, which increases with the value of L.

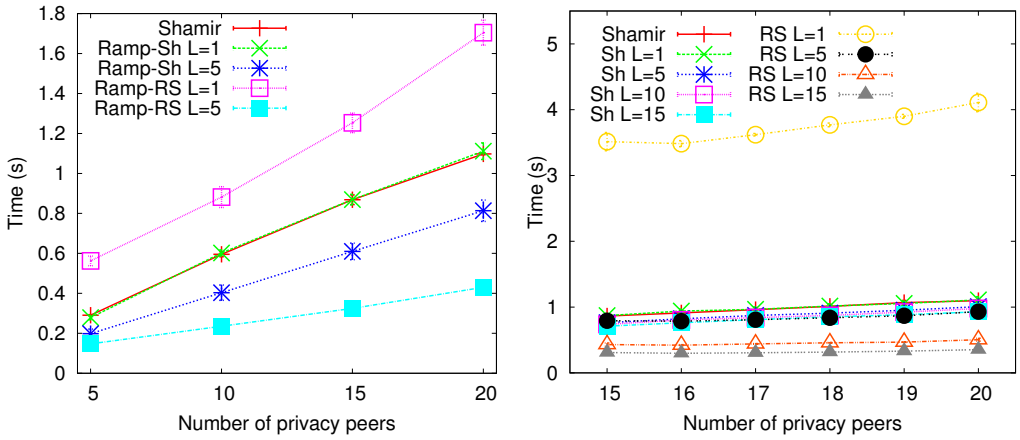
Finally, we show how increasing the threshold t affects the computation time for share generation and reconstruction, respectively, in Figures 5.4(a) and 5.4(b), with $n = 20$ privacy peers. The share generation CPU time in Figure 5.4(a) linearly increases with the threshold for the original Shamir scheme, while the optimized implementation of *Ramp-Sh*, $L = 1$ scheme varies slightly between $t = 10$ and $t = 20$.

The reconstruction CPU time shown in Figure 5.4(b) indicates a low impact of increased t, for both Shamir and the *Ramp-Sh* scheme. The impact of increased L is also not significant for the *Ramp-Sh* scheme, as there is a balance between the CPU time reduction resulting from handling multiple secrets simultaneously and the increased overhead required for generating the shares. We observed a negative effect of using lower L values (compared to the value of t) for the *Ramp-RS* scheme, similarly to that observed in Figure 5.3(b). However, we note the increased security that such a configuration would provide with respect to higher L values.



(a) $t = 5$ (b) $t = 15$

Figure 5.2: Average CPU time an input peer requires to generate n secrets, for $t = 5$ and $t = 15$.



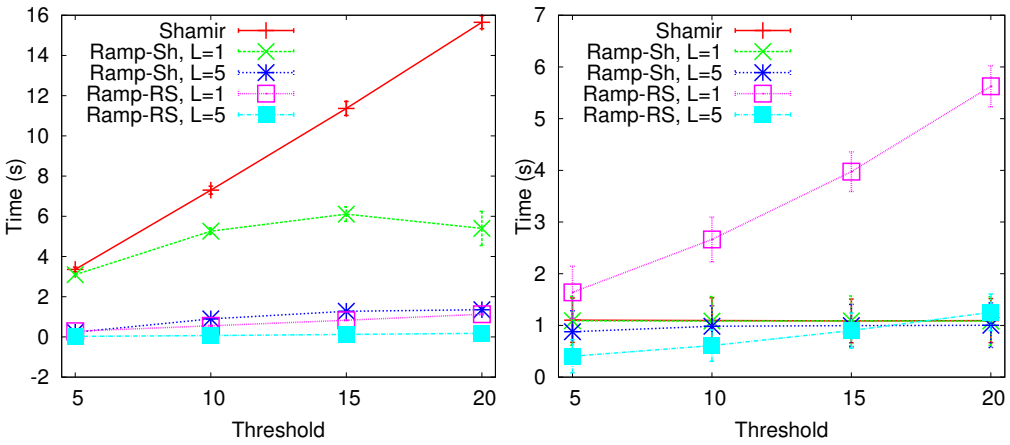
(a) $t = 5$ (b) $t = 15$

Figure 5.3: Average CPU time a privacy peer requires to reconstruct the result, for $t = 5$ and $t = 15$.

5.6.2 Communication Overhead

We now address the communication overhead of the various schemes. We define this cost as the average volume of data downloaded and uploaded by each peer.

Let C_u^i denote the average volume of data uploaded (sent) from all input privacy peers and C_d^i denote the average volume of data received by the same peers. Similarly, we denote the average volume of data uploaded and downloaded by privacy peers,



(a) Input peer share generation time (b) Privacy peer secret reconstruction time

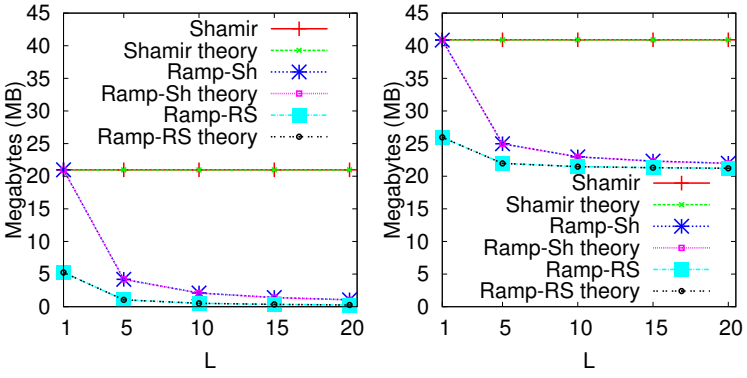
Figure 5.4: Average CPU time per peer to (a) generate shares and (b) reconstruct shares, averaged over $m = 20$ privacy peers and $n = 20$ input peers.

respectively, as C_u^p and C_d^p.

For each of the m input peers, theoretically, $C_u^i = \frac{|S|}{L} \cdot n$, where $|S|$ is the size of the input secret (all participants share secrets of the same size). The download data volume is given by $C_d^i = |S| \cdot n$, assuming (as is the case in SEPIA) that each of the n privacy peers communicates the result back to all input peers. Each privacy peer sends $C_u^p = \frac{|S|}{L}(n-1) + |S| \cdot m$, as they communicate both with the input peers and all other privacy peers. They receive $C_d^p = \frac{|S|}{L} \cdot m + \frac{|S|}{L} \cdot (n-1)$, comprising the computed shares they need to aggregate (from other privacy peers) and data from the input peers.

We conducted experiments to verify the validity of the simple theoretical estimates and to evaluate the communication overhead introduced by the SEPIA protocol. Figures 5.5(a) and 5.5(b) show example average upload volumes C_u^i and C_u^p as a function of L, with $m = 20$ input peers, $n = 20$ privacy peers and $t = 20$ (the threshold does not impact the communication costs, and we selected a high value to enable a wide range of L values). As estimated, the decrease in communication cost is proportional to $\frac{1}{L}$, with a small practical difference of $7kB$ on average (the input data being $1.05MB$ in our scenario), which may be due to peer synchronization. The privacy peer upload volume is lower bounded by $21MB$, as each peer needs to forward the result back to

input peers; this is obviously not impacted by L. We note that the download volumes C_d^i and C_d^p are similarly close to the theoretical estimates.



(a) Input peer (b) Privacy peer

Figure 5.5: Upload in MB from each input and privacy peer for $m = 20$ input peers, $n = 20$ privacy peers and $t = 20$.

5.7 Discussion

In our experimental evaluation, we investigated a wide range of parameter choices for strong ramp schemes. However, these choices were still limited by the experimental platform, particularly the RAM size of our virtual machines (2GB, shared by four peers). In a setting where individual devices would take the role of input and privacy peers, RAM of 1GB and a 1GHz CPU is quite common, even in mobile devices. Therefore, in accordance with reported SEPIA capabilities ([17] estimates that SEPIA could support up to 140 input peers), we envisage that having over a hundred input and privacy peers would be easily achievable. In general, increasing the value of the parameter L (up to the maximum possible size of t), for a fixed number of privacy peers n (that has to be greater than t), is beneficial for reducing both the computation and communication overhead. On the other hand, as per Definition 3, strong ramp schemes provide reduced security compared to Shamir scheme. The sensitivity of data used for specific secure computations will ultimately be the deciding factor in choosing the appropriate secret sharing scheme.

5.8 Conclusion

Overall, strong ramp schemes have a potential to provide significant benefits in regards to both communication costs and complexity and could be well suited to MPC applications in the emerging mobile network services that rely on private data. We proposed two construction methods for such schemes and experimentally evaluated their performance in a practical setting, using our implementation of ramp schemes within the MPC framework.

Chapter 6

Conclusion

6.1 Conclusion

This thesis addressed the challenge of sharing private information in real-time, on mobile devices that have resource constraints (memory, computation and bandwidth). Specifically, we address three research questions and provide related contributions. The first research question related to the complexity and memory requirement of two variants of on-the-fly coding scheme with an elastic encoding window designed for real-time applications. The second question addresses the problem of sharing, privately, a user profile in an OSN service, without leaking any information about this profile and without having to rely on a trusted third party. Finally we addressed the problem of the practicality of sharing, securely, a large volume of private data and performing secure computations on such data. We detail the three contributions below.

Practicality of the On-the-Fly Coding Scheme on Mobile Devices

On-the-fly coding scheme with an elastic encoding window, proposed and evaluated in prior studies, improves the performance of real-time applications in regards to lowering the delays [103]. In Chapter 3, we have investigated the difference between the systematic and non-systematic variants of this scheme in regards to in the required

memory size and computational constraints, in a wireless multicast scenario where the receivers are mobile devices.

We used simulations and we considered two types of the erasure channel: uniformly distributed losses and the Gilbert-Elliot model. Our results have shown that the systematic approach outperforms the non-systematic approach in regards to both the buffer sizes needed at the transmitting and the receiving side, and the computation complexity.

Layered Secret Sharing Scheme for Profile Sharing in OSN Services

Chapter 4 studied the usability of secrets sharing schemes for an OSN profile sharing application, where users wish to prevent leakage of any profile related information to either third parties (including the OSN service provider) or to malicious participants. The existing multi-secret sharing schemes do not satisfy the application requirements, as *e.g.,* they do not protect the number of profile attributes *ie.* secrets shared by a user.

We proposed a novel Layered secret sharing scheme that has the desired properties. We then analysed the security of this scheme against honest-but-curious attackers who, after receiving a set of Layered shares, attempt to recover a larger number of secrets than what they should be able to legitimately access. Then, we analysed the level of protection proided by the Layered secret sharing scheme in an OSN; using the results of this analysis, based on a real OSN graph, we provided guide-lines on how to parametrise the security level of our scheme. Finally we implemented our scheme and compared it with a naive secret sharing scheme in regards to computation time and communication costs. We have shown that the increased security of our scheme comes at a cost of additional computation time. However we have also shown that by carefully selecting the parameters of the Layered scheme (*ie.* by using a ramp scheme as a primitive when sharing large secrets), our scheme is practical for a profile sharing application, as the time to generate and decode the Layered shares is of the order of

magnitude of one second.

Strong Ramp Schemes and Systematic MDS Codes

In Chapter 5, we studied the links between ramp schemes, strong ramp schemes, MDS erasure codes and systematic MDS erasure codes. The main theoretical outcome of this work was proving that that the redundancy components, generated by a systematic MDS code, satisfy the conditions for a strong ramp scheme. We implemented and integrated two strong ramp schemes in the SEPIA MPC framework, first scheme derived from Shamir scheme and the second scheme using Reed-Solomon erasure codes. We have experimentally evaluate the different implementations of the strong ramp schemes and compared them with Shamir scheme (originally implemented in SEPIA) in a network outage detection application. The results have shown, first, that such strong ramp schemes deliver a practical trade-off between the information theoretic security offered by Shamir scheme and the computation time required to compute the selected MPC operation (multiset union). We have also shown that the communication cost of the MPC protocol is reduced by using strong ramp schemes.

6.2 Future Work

The research presented in this thesis may be extended in a number of directions:

- We have shown in Chapter 5 that a strong ramp scheme can be derived from any systematic MDS code. In our experimental evaluation, we used the Reed-Solomon code based on Vandermonde matrices over a binary Galois field. The implementation of these codes assumes a scenario where the erasures are not predictable. However, in a MPC scenario outlined in Chapter 5, the number of participants and connectivity between them is static. This may allow participants to pre-compute some elements required to generate and/or decode the shares. Therefore, investigating the optimised implementations of MDS codes in this

scenario could provide additional performance benefits.

- We have focused our secret sharing work on OSN and networking scenarios. A promising additional application area for future study could be distributed storage. In this field, there are a number of prior works combining cryptography and erasure codes *e.g.,* [88], [91] [63]. In Chapter 5, we have shown that the redundancy components of a systematic MDS code can be utilised as a strong ramp scheme. Therefore the next logical step could be to explore the potential to remove the cryptographic component used in related works, and only use the strong ramp scheme to provide secure distributed storage. Future work could examine the use of secret sharing in this field and the efficiency bounds that may be achieved.

- In Chapter 3, we address the efficiency of novel convolutional codes. Considering the results showing the close links between block codes and secret sharing schemes, an interesting diretcion for future study could be the potential extension of secret sharing schemes based on the concepts of elastic memory, used in on-the-fly codes.

Appendices

Appendix A

Fountain Multiple Description Coding

A.1 Introduction

Video content has become hugely popular on Internet [58] and the resulting video traffic adds new constraints to the network in terms of capacity while some video applications, *e.g.* streaming, have strong delay constraints. Furthermore, the network is composed of heterogeneous types of devices ranging from a smartphone with low computational capacity and small screen, to a powerful computer with a HD screen, therefore having different video quality requirements while sharing the same network.

The large number of end-user devices accessing video content can also overload the video server capacity. To address this problem, a proposed solution is to take benefit of a Peer-To-Peer (P2P) network to decrease the load on the server by sharing the data (referred to as content pieces) between all the nodes as in [96]. P2P networks are known to provide high throughput and ability to cope with failure, churn and heterogeneous node's capacity. In the particular context of video streaming, P2P solutions must also provide sequentiality to ensure that chunks which are due for playout are not incomplete or missing. Providing sequentiality limits re-buffering, or the potential to

abandon expired frames. In file sharing P2P such as BitTorrent, the high throughput and robustness are mainly due to the diversity of the chunks available on peers which is a consequence of the rarest-first chunk selection algorithm. The sequentiality prevents the use of rarest-first and reduce the diversity. As it was proven in [36], there is no system satisfying those three constraints and users must be able to cope with varying throughput and incomplete chunks.

To respond to the different quality of the receivers, a solution is to use Multiple Description Coding (MDC) codes [42] which split a video into n descriptions. Each description brings out information about the video, thus the more descriptions a user receives, the better the quality of the video is. A receiver is able to download the full quality video if the n descriptions are available, but high churn rate, congestion in the underlying network or link layer losses may prevent the receiver to complete the download of chunks in time. If a P2P streaming solution integrates an MDC code, partial chunks translate to less than n descriptions allowing a graceful quality degradation (*i.e.* lower quality of video without stopping or skipping frames [42]). If the use of MDC code allows a trade-off between video quality and buffering time, it remains that the low chunk diversity impacts on the throughput and robustness.

If MDC codes are able to produce an infinite amount of descriptions while allowing to reconstruct high quality video with n of those, seeders could increase the chunk diversity by generating different descriptions for each peer. Towards this goal, this paper studies the feasibility of such a Fountain MDC codes. We propose a practical scheme and assess the quality of reconstructed video compared to standard MDC. Finally we discuss the limitations of such a scheme, its implication in terms of complexity and its integration in a video codec.

Our contributions include the novel concept of combined MDC and Fountain codes for P2P video streaming, a proposal for a specific code and the evaluation of the performance of this code on a selected set of pictures.

This appendix is organised in the following way. In Section 2, we provide a back-

ground on P2P video streaming, Fountain codes and MDC codes. In Section 3, fountain MDC code are presented and then simulated in Section 4. Finally, we discuss and conclude this paper in Section 5.

While the topic of this work is closely related to the rest of this thesis, we preferred to have it in appendix because of the linear operations we are using to built the fountain MDC code which are not in a Galois field as the previous works.

A.2 Background on P2P Streaming

During the past few years, the adaptation of P2P networks to the context of streaming applications have received a significant interest from the research community. This section review the key design concept and the use of erasure coding and the adapted video codec.

A.2.1 Network Structure and Chunk Selection

In live P2P streaming, randomly connected mesh networks have been promoted [71] as they allow path diversity, churn resilience and a simple construction and maintenance of the topology. As in file-sharing P2P, the random mesh structure implies that chunks diversity impacts directly on the capacity of peers to help each other and achieve high throughput. The downside is that contrary to tree-structured networks, the limited availability of future content in live streaming applications as well as the need for sequentiality to achieve smooth playback affects the chunks diversity. This issue has been partially addressed by the use of playout buffers and by pushing missing chunks randomly with a probability proportional to the playback time of the chunk [14].

A.2.2 Rateless Erasure Code

Usage of source or peer-based rateless erasure codes[1] have been proposed to increase chunk diversity without increasing the delay [54, 112]. For instance, the authors of [109] propose a source-based encoding with Fountain codes. One or more sources creates and pushes encoded chunks built from k source chunks. These encoded chunks have the same size than the source ones. The characteristic of rateless codes is that they can generate an infinite amount of distinct encoded chunks [70]. A given block can be decoded when slightly more than k coded chunks are received. In this case, the diversity is increased as source chunks have multiple representations in the network and a peer can accept coded chunks describing the same block pushed by multiple source.

Random linear coding is used to perform peer-based coding (*i.e.* network coding) such as [112] where peers re-encode the various coded chunks available. This allows to increase a bit further the diversity and reduce the overhead and control messages. The downside of erasure codes is that chunks cannot be played until the full block has been decoded[2].

A.2.3 MDC Codes

As previously described, Multiple Description Coding is made to split a single video stream in multiple streams, called descriptions. Different kind of MDC codes exist: Scalable Video Coding (SVC) requires the first i descriptions to decode the $i + 1^{th}$ description and increase the quality of the video [95]. MDC codes that produce independent descriptions are furthermore interesting[3] as they tolerate the loss of any description [42]. This is an ideal solution for multiple-tree based P2P networks with

[1] The encoding part can use different types of operations as the XOR, linear combinations or operations on Galois fields.

[2] Obviously, encoding should be media aware *e.g.* if a block is as a function of the GOP (Group Of Pictures) size, it is sufficient to decode a block to ensure that the GOP will be played without errors (as in [109]).

[3] Until now, SVC codecs provide a better ratio between fidelity and compression than independent MDC codes.

the transmission of one description per tree. In random mesh networks, they cope well with churn [79] and the heterogeneity of peers connectivity [67].

A.2.4 Combinations of Techniques

Both rateless erasure codes and MDC video codecs improve the overall Quality of Experience (QoE) of a video streaming on a P2P network and many proposals use both schemes conjointly [109, 39, 82]. In these works, the video is encoded in multiple descriptions using an MDC codec. Then, on each description a rateless code is used. However in these papers, the techniques are used one after the other and not combined. In other words, a user first needs to decode the rateless code and then, use the MDC code to display the video.

The remaining of this paper investigates the design of a *Fountain MDC code* which inherits from the properties of both techniques, *i.e.*: to be able to create an infinite number of encoded packets (Fountain property) which are all useful on their own and improve each other (MDC property).

A.3 Our Proposal: the Fountain MDC Code

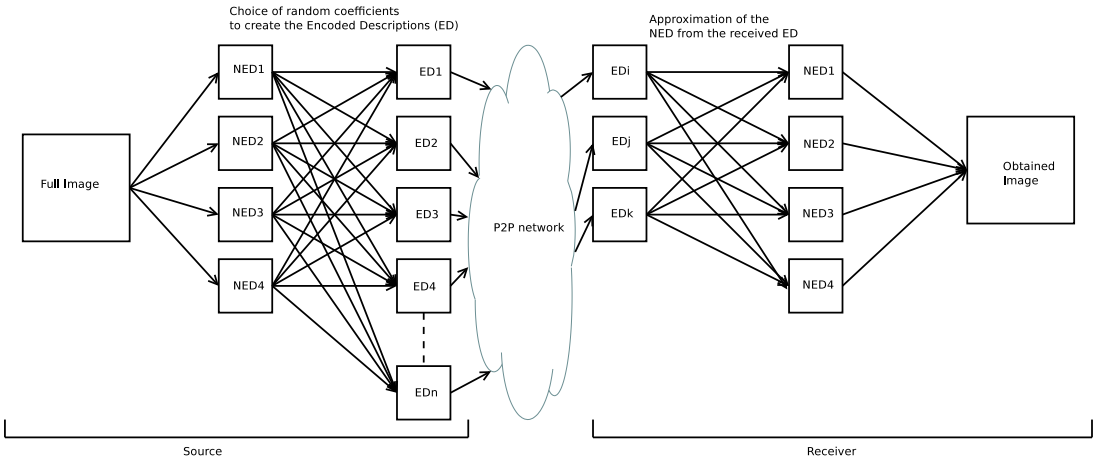


Figure A.1: The whole process when for example three ED are received.

The Fountain MDC code would enable the creation of an infinite number of descriptions of a video, creating these descriptions when needed, *i.e.* on-the-fly. From the receiver point of view, each received description would increase the quality of the video (as a classical MDC code would) and when a given number of them is received, the resulting video would have a sufficient quality (as a Fountain code, a file can be decoded when the receiver receives a little more information than the size of the file). In a P2P network, this code would allow a seeder to send new descriptions to each node which is in contact with this seeder. This new description is then useful to any other nodes in the network, which thus may increase the chunk diversity on the P2P network. Furthermore, a peer which does not have a sufficient amount of descriptions to decode the full quality may be able to make linear combination of the ones received creating another description (similarly to Network Coding [61]), also improving the chunk diversity. Finally as a standard MDC code, it would allow heterogeneous devices to play the same content with different quality. This would allow to adapt the content as a function of the screen resolution of the device or the network capacity, by downloading more or less descriptions.

To create one of this Fountain MDC codes, we focus our effort on creating a Fountain code which would have the MDC property to increase the video quality for each new received description. Thus, we have specifically worked on pictures, which would represent the different I frames created by a video codec as with *h.264* coding scheme [92].

Usually, Fountain codes are introduced at the network layer, so in this context, it would be after the codec at the source side. However in this case, a receiver would have to decode the Fountain code to use the information. As our purpose is to be able to use information not totally decoded, we create the Fountain code at the application layer to be able to use this encoded information, and more precisely before the compression which is not a linear process.

The whole process from the recorded picture to the displayed one at the receiver

side follows four steps as illustrated in Figure A.1. At the source side, the first step is to split the picture in a number of non-encoded descriptions (NEDs), then the creation of the encoded descriptions (EDs) which can be made on-the-fly by making linear combinations of the previous NEDs. The different streams are then sent to the users. At the receiver side, first a certain amount of streams are received. From the received EDs, the next step is to approximate the NEDs from the linear combination obtained to finally rebuild the whole picture. We precisely describe each step of the process in the following.

A.3.1 Creation of Four Non Encoded Descriptions

From the picture, four descriptions are created from a spatial subdivision by taking one pixel on four per description from all the $2 * 2$ blocks of pixels as in Figure A.2. These four descriptions will be referred to as non-encoded descriptions (NEDs). As explained in [107], this spatial subdivision allows to create four descriptions with a low computational power and without creating a whole new video codec.

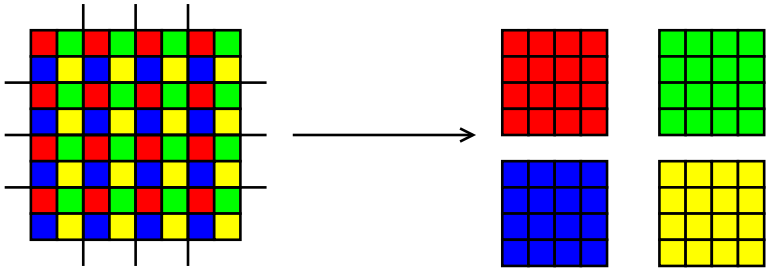


Figure A.2: Creating the four non-encoded descriptions from the picture

A.3.2 Creation of the Infinite Number of Descriptions

A.3.2.1 In Theory

The four NEDs are used to create the infinite number of encoded descriptions (EDs) by performing the Fountain process as in Figure A.1. The constraint at this stage is to have a useful description on its own after the encoding. For this purpose, we are doing

random linear combinations of the four NEDs with coefficients which sum is equal to 1:

$$ED_i = \alpha_i * NED_1 + \beta_i * NED_2 + \gamma_i * NED_3 + \phi_i * NED_4$$

with $\alpha_i + \beta_i + \gamma_i + \phi_i = 1$. With this process, all the pixels after the encoding still have a value between 0 and 255 and are a barycentre of the pixels from the NEDs. The choices of the coefficients will be explained in Section A.3.2.2.

When the EDs are created, the different pixels do not have integer values, but are float numbers (due to the coefficients which are not integers). To be sent on a network or to be used by a standard codec (as *JPEG* for pictures or *h.264* for video), the different pixels values are rounded to the closest integer. This is the first stage where errors appear in the process as the numbers are approximated.

A.3.2.2 Choice of the Coefficients

We choose two types of random coefficients to create the EDs from the NEDs. The first choice is to create the four random coefficients close to each other. This choice allows to have a good description of the full image when one ED is received, but because of the rounding to obtain integer, the different EDs created with this process can be close to each other. As the difference between them can be small, it is more complicated to obtain new information when a new ED is received. In practice, we choose to take randomly the coefficients between 1 and 4 and then normalise them so that the sum is one. In the following, this configuration is denoted *small coefficients* configuration.

Then we choose a second type of random coefficients: one coefficient is dominant and the others are small in comparison. This choice describes well one NED in an ED, while little information about the other NEDs is present. We denote in this case that an ED_i describes an NED_j if the coefficient used for the NED_j is the dominant one. In practice, the dominant coefficient is around 20 times higher than the small ones. Then, they are also normalized so that the sum totals one. For the case a peer receives four EDs describing the four original NEDs, we obtained nearly the same results as

receiving directly the four NEDs. Thus in the following, we only study the worst case for the receiver: each received ED describes the same NED_j. This corresponds to the worst case as every time, the same NED is well characterized while the receiver does not have precise information about the others. In the remainder of the paper, this configuration is denoted *large coefficient* configuration. In a P2P system, to increase the availability of chunks, a user should not be forced to differentiate between two descriptions built from the same frame. In the *large coefficient* case, this worst case scenario is quite likely, *i.e.* it is likely that two or more EDs will describe the same NED, as the receiver does not choose a description based on the way to encode it, but only on the frame it was built from.

Finally, we also introduce a threshold condition on the ED when it is created. When the coefficients are chosen, the average error due to the rounding is computed for each pixel and each color. If this value of the error is higher than:

- $\frac{3}{\sum coefficients}$ for the *small coefficients* configuration and
- $\frac{10}{\sum coefficients}$ for the *large coefficients*;

the ED created is replaced with a newly created one. Applying this threshold condition reduces the amount of errors introduced by the rounding when EDs are created.

A.3.3 Decoding the Streams

A.3.3.1 Overview

If a codec is used, the first step at the receiver side is to decompress the stream in a picture where all pixels are described by three bytes. Then, depending on the number of received EDs, the process to decode them can differ but after the decoding process, four NEDs are approximated to finally reconstruct the whole picture.

A.3.3.2 Operations Done to Decode the Fountain Code

In this section, we explain the operations done to approximate the full image as a function of the number of ED received:

1. If only one ED is received, we just use the value obtained for the four pixels;

2. If two EDs are received, the operation used for the approximation is the barycentre. Using the same notations as in Section A.3.2.1, we approximate the pixels in each NED as follows:

$$NED_1 = \frac{\alpha_1 * ED_1 + \alpha_2 * ED_2}{\alpha_1 + \alpha_2};$$

$$NED_2 = \frac{\beta_1 * ED_1 + \beta_2 * ED_2}{\beta_1 + \beta_2};$$

$$NED_3 = \frac{\gamma_1 * ED_1 + \gamma_2 * ED_2}{\gamma_1 + \gamma_2};$$

$$NED_4 = \frac{\phi_1 * ED_1 + \phi_2 * ED_2}{\phi_1 + \phi_2};$$

3. If three EDs are received, we are also doing the barycentre method as previously but with three coefficients instead of two. This solution is compared with a more complex one: knowing that the solutions are integers, we solve the Diophantine system with three equations and four unknowns by choosing the last value which gives the lowest difference with the received ED. Then, we use these solutions to determine directions. We do not use them directly as the rounding process creates to much inaccuracy to have a good solution. Finally, we add or subtract (depending on the direction obtained) the value $\frac{1}{max(\alpha_1,\alpha_2,\alpha_3)}$ to the barycentre of NED_1 for example. We add or subtract this fraction value as the pixel can be equal to both these values and still have the same rounding in the linear operation. This fraction corresponds to the imprecision of the rounding for this specific NED;

4. To finish, if four EDs are received, we are doing the barycentre process and as an alternative, we are also solving the linear system made by four equations with four unknowns by inverting the matrix built from the random coefficients.

A.4 Results

Simulations are done with the well-known Lena picture which is a 512*512 pixels size in color (RGB) (see Figure A.3). In Table A.1, the receiver receives the NEDs from the *JPEG* codec with a quality of 100% which still creates errors due to compression, as we can observe, if we compare them with the not compressed pictures (without codec). The results in both tables are close, the main difference is when the receiver gets four NEDs, the PSNR (Peak Signal-to-Noise Ratio) grows from $50dB$ to ∞. Note that for a video application, a PSNR equals to $50dB$ is already an excellent quality. These two tables are presented as references.

In Table A.2, we compute the average PSNR on 50 simulations, when the receiver gets one to four EDs which are not compressed by a codec, in order to analyse only the effect of the errors due to the rounding in the encoded process. The EDs are built with small or large coefficients, and with or without the threshold condition on the coefficients as explained in Section A.3.2.2. As explained in the part A.3.3.2, depending on the number of EDs received, different decoding algorithms are used.

First, we can see in Table A.2 that the *large coefficients* configurations (with or without the threshold condition on the coefficients) do not bring any improvements when new EDs are received. The configurations with *large coefficients* are worst cases, *i.e.* when all the EDs are describing the same NED. This result shows that the *large coefficients* is equivalent to receive directly the NED. Actually, receiving two or more EDs describing the same NED do not improve the PSNR, which implies that in this configuration, the receiver cannot download any ED built from the frame. It has to choose one which brings new information, exactly as without the Fountain process.

Then, with the *small coefficients* configuration, we can first observe that using the

(b)

Figure A.3: Pictures used for the simulations

threshold condition on the coefficients improves the average quality of the picture. Then the MDC property is verified: the more EDs are received, the better is the quality. But the improvement is quite low: from $31dB$ with one ED to $37.5dB$ with four EDs in the best case, furthermore the improvement is only equal to $0.5dB$ when three EDs are received compared to one.

Finally, when compared to the standard MDC (when the NEDs are directly received), the best configuration of the Fountain process brings out an improvement when only one ED is received (difference of $3dB$). However, receiving a new NED improves more the PSNR than receiving a new ED. Thus with three NEDs received, the PSNR obtained is equal to $35.5dB$ when the one obtained with the ED is equal to $31.5dB$. Finally if the four NEDs are received, the PSNR is infinite while with the EDs, it is only equal to $37.5dB$ on average.

On average, we observed that the *small coefficients* configuration with threshold condition brings out the best results compared to the other case. Thus, to have a better understanding of the results obtained, Figure A.4 shows the quartiles obtained with this configuration as a function of the method used and the number of received descriptions. The PSNR obtained for the NEDs are also plotted as a reference. Therefore we can see that the barycentre method has a very low variance, and in 100% of the case, it brings

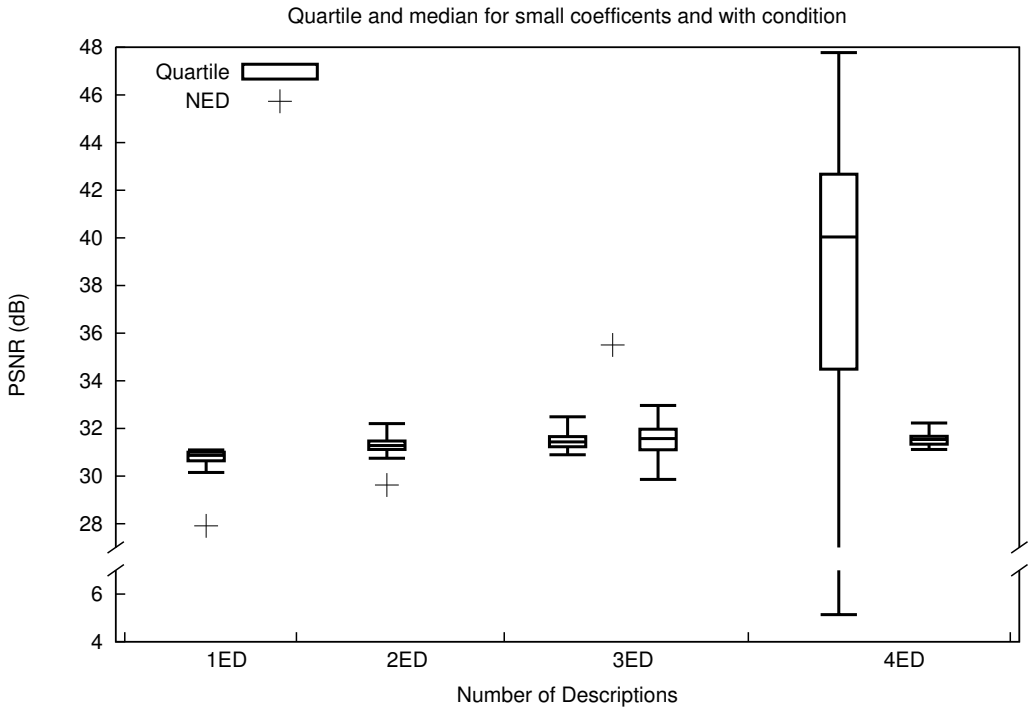


Figure A.4: Quartiles obtained when the small coefficients with the threshold condition configuration are used. When three EDs are received, the box on the left is obtained with the barycentre techniques, the one on the right with the Diophantine. When four EDs are received, the box on the left represent the matrix inversion, the one the right the barycentre. When four EDs are received, the value for the NEDs is infinite.

out better results when one or two EDs are received compared to NEDs receptions. But in all case it has a worse PSNR when three or four descriptions are received. Then when four EDs are obtained, to inverse the matrix shows a better PSNR than the barycentre method in more than 75% of cases, but in few cases, the decoded picture can be unusable (the minimum PSNR is equal to $5.1dB$).

As a final test, we use another picture of a table soccer which is composed by 3264*2448 pixels in color (see Figure A.3). We made only five simulations with one configuration which is the *small coefficients* without threshold condition and without using a codec. The algorithms used to recompose the full picture at the receiver side is the barycentre one when one to three EDs are received, or the matrix inversion

Table A.1: PSNR (dB) for the NEDs

	1 NED	2 NEDs	3 NEDs	4 NEDs
With Codec	27.88	29.56	35.37	50.30
Without Codec	27.91	29.62	35.50	∞

Table A.2: PSNR (dB) for Lena picture without codec, summary, barycentre except if noticed (the coefficients of variation are in parenthesis)

Number of EDs	Average without threshold condition	Average with threshold condition
1 ED large coeff	29.36 (0.011)	29.10 (0.008)
1 ED small coeff	30.92 (0.007)	30.80 (0.008)
2 EDs large coeff	29.47 (0.008)	29.18 (0.005)
2 EDs small coeff	31.26 (0.009)	31.30 (0.009)
3 EDs large coeff	29.45 (0.006)	29.16 (0.005)
3 EDs Diophantine chooses the direction large coeff	27.34 (0.095)	25.21 (0.155)
3 EDs small coeff	31.39 (0.007)	31.47 (0.010)
3 EDs Diophantine chooses the direction small coeff	31.76 (0.014)	31.54 (0.022)
4 EDs inversion large coeff	26.84 (0.268)	26.46 (0.239)
4 EDs large coeff	29.49 (0.005)	29.16 (0.004)
4 EDs inversion small coeff	34.97 (0.224)	37.52 (0.227)
4 ED small coeff	31.42 (0.006)	31.53 (0.007)

Table A.3: PSNR (dB) for Table Soccer picture without codec, the configuration tested is the small coefficients without threshold condition.

Number of EDs	Average	Worst case	Best case
1 ED	33.45	33.23	33.56
2 EDs	33.79	33.48	34.25
3 EDs	33.85	33.55	34.21
4 EDs (inversion)	35.73	26.31	43.83

when four EDs are received. The results presented in the Table A.3 are the average PSNR, the worst and the best case obtained on the different simulations. The results are similar to the one obtained from Lena's picture, which tends to prove that the simulated results are not linked to the picture used.

A.5 Conclusion, Discussion and Future Work

Streaming a video over a P2P network composed of heterogeneous devices with different calculation, bandwidth and display characteristics, is a complex problem that has been tackled in several studies. MDC codes and rateless codes are two possible solutions and are sometimes used conjointly in some studies. However, MDC codes are not flexible and when combined with a rateless code, this code still has to be decoded to be useful. The Fountain MDC code introduced in this article combined both the property of the MDC code to improve quality for each received descriptions, and the rateless property in order to create an infinite number of descriptions on-the-fly. This kind of code would allow to increase the chunk diversity over a P2P network, i.e., its global throughput and robustness. In this appendix, we assess the practical feasibility of the Fountain MDC code by proposing a low-computation one to estimate the gain in PSNR as a function of the number of received descriptions when used on pictures. Although the MDC and Fountain properties are achieved and the idea seems promising for the chunk diversity on a P2P network, we observe that the gain in PSNR obtained per new description is limited (denoising filtering recommended by MPEG-4 [89] is not used in this study to observe the capacity of the Fountain MDC code without artefact),

which tends to limit its deployment.

Then, as our main goal is to study the feasibility of the Fountain MDC code, we did not investigate details of a potential video implementation. However we can raise many questions for future works if improvement on the obtained PSNR are gained, as to understand how to create the motion vectors for this type of codec. Actually, random ED are created, but they are finally closed to the original NED. Thus, the study of the effect of the linear combination used to create the ED on the motion vector is still an open question, and more precisely how to assess whether we need to send the motion vector created from the NED or from the ED. Otherwise, as we are rebuilding the whole pictures, it could be relevant to use directly the motion vectors created from the whole video before the splitting process.

Finally concerning the improvement on a P2P network, we think that the Fountain MDC code could increase the availability of the different chunks on the network, but we did not push further the study to measure this improvement. Furthermore, with this type of codes, a node in the network could also utilize network coding on the different descriptions it receives, knowing that network coding can usually improve the global performance of a P2P network as in [111]. A peer in the P2P network (which also watches the video) can first obtained different ED, and then make a random normalized linear combination of them which becomes a new MDC encoded description. However it may imply new rounding, and consequently additional errors which have to corrected.

Appendix B

Calculating $P(|U_1^k| = x \mid x_y)$

To assist with the analysis of $P(|U_1^k| = x \mid x_y)$, the probability that the size of the union of any k sets of size x_y amongst n Layered shares is equal to x, we introduce some additional notations. Let S_y be a set of size x_y for $y \in [1, k]$. Denote $U_1^y = \cup_{i=1}^{y} S_y$ and let x_1^y be the size of U_1^y. We will calculate the number $R_k(x)$ of k sets S_y, verifying $|U_1^k| = x$, knowing the size x_y of each set. We prove by induction on $k \geq 1$ that for all $x = x_1^k \in [0, n]$:

$$R_k(x_1^k) = \sum_{x_1^{k-1}=\max_{l=1}^{k-1}(x_l)}^{min(\sum_{l=1}^{k-1} x_l, x_1^k)} \cdots \sum_{x_1^y=\max_{l=1}^{y}(x_l)}^{min(\sum_{l=1}^{y} x_l, x_1^{y+1})} \cdots \sum_{x_1^2=max(x_1,x_2)}^{min(x_1+x_2,x_1^3)} \prod_{z=1}^{k} \binom{n - x_1^{z-1}}{x_1^z - x_1^{z-1}} \binom{x_1^{z-1}}{x_z - (x_1^z - x_1^{z-1})} \tag{B.1}$$

Initialization: $k = 1$

The number of sets verifying $|U_1^1| = |S_1| = x_1^1$ with $|S_1| = x_1$ is equal to $\binom{n}{x_1^1}$, and $x_1^1 = x_1$. For $k = 1$, the previous formula is equal to $\binom{n-x_1^0}{x_1^1-x_1^0}\binom{x_1^0}{x_1-(x_1^1-x_1^0)} = \binom{n}{x_1^1}$ having $x_1^0 = 0$ and $\binom{0}{0} = 1$. Thus the formula is true for $k = 1$.

Iteration: $k \geq 1$

We assume that the formula is true for k and for all $x_1^k \in [0, n]$. We will show that it is also true for $k + 1$, for all $x_1^{k+1} \in [1, n]$.

We calculate $R_{k+1}(x_1^{k+1})$ for all $x_1^{k+1} \in [1, n]$. We consider all the possible unions

of size x_1^k made with k sets of respective sizes $x_1, ..., x_k$. Then, for each of these unions, S_{k+1} of size x_{k+1} needs $x_1^{k+1} - x_1^k$ elements in the remaining $n - x_1^k$ elements to complete the union, and the $x_{k+1} - (x_1^{k+1} - x_1^k)$ other elements in the x_1^k already chosen, should not add new elements in the union. Thus

$$R_{k+1}(x_1^{k+1}) = \sum_{x_1^k = \max_{l=1}^{k}(x_l)}^{min(\sum_{l=1}^{k} x_l, x_1^{k+1})} R_k(x_1^k) \binom{n - x_1^k}{x_1^{k+1} - x_1^k} \binom{x_1^k}{x_{k+1} - (x_1^{k+1} - x_1^k)} \tag{B.2}$$

By the recursion hypothesis, we obtain:

$$\begin{aligned} R_{k+1}(x_1^{k+1}) = & \sum_{x_1^k = \max_{l=1}^{k}(x_l)}^{min(\sum_{l=1}^{k} x_l, x_1^{k+1})} \binom{n - x_1^k}{x_1^{k+1} - x_1^k} \binom{x_1^k}{x_{k+1} - (x_1^{k+1} - x_1^k)} \times \\ & \left[\sum_{x_1^{k-1} = \max_{l=1}^{k-1}(x_l)}^{min(\sum_{l=1}^{k} x_l, x_1^k)} \cdots \sum_{x_1^y = \max_{l=1}^{y}(x_l)}^{min(\sum_{l=1}^{y} x_l, x_1^{y+1})} \cdots \sum_{x_1^2 = max(x_1, x_2)}^{min(x_1 + x_2, x_1^3)} \prod_{z=1}^{k} \binom{n - x_1^{z-1}}{x_1^z - x_1^{z-1}} \right. \\ & \left. \times \binom{x_1^{z-1}}{x_z - (x_1^z - x_1^{z-1})} \right] \\ = & \sum_{x_1^k = \max_{l=1}^{k}(x_l)}^{min(\sum_{l=1}^{k} x_l, x_1^{k+1})} \cdots \sum_{x_1^y = \max_{l=1}^{y}(x_l)}^{min(\sum_{l=1}^{y} x_l, x_1^{y+1})} \cdots \sum_{x_1^2 = max(x_1, x_2)}^{min(x_1 + x_2, x_1^3)} \prod_{z=1}^{k+1} \binom{n - x_1^{z-1}}{x_1^z - x_1^{z-1}} \\ & \times \binom{x_1^{z-1}}{x_z - (x_1^z - x_1^{z-1})} \end{aligned} \tag{B.3}$$

which is the expected result for $k + 1$.

Conclusion

The result is true for $k = 1$ and we have shown that if true for $k \geq 1$, then it is true for $k + 1$. By recursion the result is then true for any value of $k \geq 1$.

We can conclude that for all $k \geq 1$ and for all $x = x_1^k \in [0, n]$, the probability that for k sets, with respective sizes of $x_1, ..., x_k$, their union has a size of x_1^k, that is equal

to:

$$P(|U_k|{=}x_1^k \mid x_y) = R_{k+1}(x_1^{k+1})$$

$$= \sum_{x_1^{k-1}=\max_{l=1}^{k-1}(x_l)}^{min(\sum_{l=1}^{k-1} x_l, x_1^k)} \cdots \sum_{x_1^y=\max_{l=1}^{y}(x_l)}^{min(\sum_{l=1}^{y} x_l, x_1^{y+1})} \cdots \sum_{x_1^2=max(x_1,x_2)}^{min(x_1+x_2,x_1^3)} \prod_{z=1}^{k} \frac{\binom{n-x_1^{z-1}}{x_1^z-x_1^{z-1}}\binom{x_1^{z-1}}{x_z-(x_1^z-x_1^{z-1})}}{\binom{n}{x_z}} \tag{B.4}$$

Bibliography

[1] J. Anderson, C. Diaz, J. Bonneau, and F. Stajano, "Privacy-enabling social networking over untrusted networks," in *Proceedings of the 2nd ACM workshop on Online social networks*, ser. WOSN '09. New York, NY, USA: ACM, 2009, pp. 1–6. [Online]. Available: http://doi.acm.org/10.1145/1592665.1592667

[2] M. Arai, A. Yamamoto, A. Yamaguchi, S. Fukumoto, and K. Iwasaki, "Analysis of using convolutional codes to recover packet losses over burst erasure channels," in *Proceedings of the 2001 Pacific Rim International Symposium on Dependable Computing*, ser. PRDC '01. Washington, DC, USA: IEEE Computer Society, 2001, pp. 258–265. [Online]. Available: http://portal.acm.org/citation.cfm?id=882475.883534

[3] R. Baden, A. Bender, N. Spring, B. Bhattacharjee, and D. Starin, "Persona: an online social network with user-defined privacy," *SIGCOMM*, vol. 39, no. 4, pp. 135–146, Aug. 2009. [Online]. Available: http://doi.acm.org/10.1145/1594977.1592585

[4] L. Bai, "A strong ramp secret sharing scheme using matrix projection," in *Proceedings of the 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks*. IEEE Computer Society, 2006, pp. 652–656.

[5] D. Beaver, S. Micali, and P. Rogaway, "The round complexity of secure protocols," in *Proceedings of the Twenty-second Annual ACM Symposium on Theory of Computing*, ser. STOC '90. New York, NY, USA: ACM, 1990, pp. 503–513. [Online]. Available: http://doi.acm.org/10.1145/100216.100287

[6] A. Beimel, "Secret-sharing schemes: A survey," in *Coding and Cryptology*, ser. Lecture Notes in Computer Science, Y. Chee, Z. Guo, S. Ling, F. Shao, Y. Tang, H. Wang, and C. Xing, Eds. Springer Berlin Heidelberg, 2011, vol. 6639, pp. 11–46. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-20901-7_2

[7] M. Ben-Or, S. Goldwasser, and A. Wigderson, "Completeness theorems for non-cryptographic fault-tolerant distributed computation," in *Proceedings of the twentieth annual ACM symposium on Theory of computing*, ser. STOC '88. New York, NY, USA: ACM, 1988, pp. 1–10. [Online]. Available: http://doi.acm.org/10.1145/62212.62213

[8] G. R. Blakley, "Safeguarding cryptographic keys," *Managing Requirements Knowledge, International Workshop on*, p. 313, 1979.

[9] G. R. Blakley and C. Meadows, "Security of ramp schemes," in *Proceedings of CRYPTO 84 on Advances in cryptology*. Springer-Verlag, 1985, pp. 242–268. [Online]. Available: http://dl.acm.org/citation.cfm?id=19478.19498

[10] J. Blömer, M. Kalfane, M. Karpinski, R. Karp, M. Luby, and D. Zuckerman, "An XOR-based erasure-resilient coding scheme," International Computer Science Institute, Tech. Rep. TR-95-048, Aug. 1995.

[11] C. Blundo, A. Gaggia, and D. Stinson, "On the dealer's randomness required in secret sharing schemes," in *Advances in Cryptology EUROCRYPT'94*, ser. Lecture Notes in Computer Science, A. Santis, Ed. Springer Berlin Heidelberg, 1995, vol. 950, pp. 35–46. [Online]. Available: http://dx.doi.org/10.1007/BFb0053422

[12] C. Blundo, A. D. Santis, and U. Vaccaro, "Efficient sharing of many secrets," in *STACS*, ser. Lecture Notes in Computer Science, P. Enjalbert, A. Finkel, and K. W. Wagner, Eds., vol. 665. Springer, 1993, pp. 692–703.

[13] ——, "Randomness in distribution protocols," *Information and Computation*, vol. 131, no. 2, pp. 111–139, 1996. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0890540196900954

[14] Y. Borghol, S. Ardon, A. Mahanti, and N. Carlsson, "Toward efficient on-demand streaming with bittorrent," in *IFIP Networking 2010*, L. N. i. C. S. M. Crovella et al, Ed. Springer, May 2010, pp. 53–66.

[15] R. Braden, D. Borman, and C. Partridge, "Computing the internet checksum," *ACM SIGCOMM Computer Communication Review*, vol. 19, no. 2, pp. 86–94, 1989.

[16] S. Buchegger, D. Schiöberg, L.-H. Vu, and A. Datta, "Peerson: P2p social networking: early experiences and insights," in *Proceedings of the Second ACM EuroSys Workshop on Social Network Systems*, ser. SNS

'09. New York, NY, USA: ACM, 2009, pp. 46–52. [Online]. Available: http://doi.acm.org/10.1145/1578002.1578010

[17] M. Burkhart, M. Strasser, D. Many, and X. Dimitropoulos, "Sepia: privacy-preserving aggregation of multi-domain network events and statistics," in *Proceedings of the 19th USENIX conference on Security*, ser. USENIX Security'10. Berkeley, CA, USA: USENIX Association, 2010, pp. 223–240. [Online]. Available: http://dl.acm.org/citation.cfm?id=1929820.1929840

[18] C. Carlet, C. Ding, and J. Yuan, "Linear codes from perfect nonlinear mappings and their secret sharing schemes," *Information Theory, IEEE Transactions on*, vol. 51, no. 6, pp. 2089–2102, Jun. 2005.

[19] B. Carminati, E. Ferrari, and J. Girardi, "Trust and share: Trusted information sharing in online social networks," in *ICDE 2012*, Apr. 2012, pp. 1281 –1284.

[20] R. Cazabet, M. Leguistin, and F. Amblard, "Automated community detection on social networks: useful? efficient? asking the users," in *Proceedings of the 4th International Workshop on Web Intelligence & Communities*, ser. WI&C '12. New York, NY, USA: ACM, 2012, pp. 1–6. [Online]. Available: http://doi.acm.org/10.1145/2189736.2189745

[21] C.-W. Chan and C.-C. Chang, "A scheme for threshold multi-secret sharing," *Applied Mathematics and Computation*, vol. 166, no. 1, pp. 1–14, 2005. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0096300304004163

[22] H. Chen, R. Cramer, S. Goldwasser, R. Haan, and V. Vaikuntanathan, "Secure computation from random error correcting codes," in *Advances in Cryptology - EUROCRYPT 2007*, ser. Lecture Notes in Computer Science, M. Naor, Ed. Springer Berlin Heidelberg, 2007, vol. 4515, pp. 291–310. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-72540-4_17

[23] R. Cramer, I. Damgård, and R. Haan, "Atomic secure multi-party multiplication with low communication," in *Proceedings of the 26th annual international conference on Advances in Cryptology*, ser. EUROCRYPT '07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 329–346. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-72540-4_19

[24] R. Cramer and I. Damgård, "Multiparty computation, an introduction," in *Contemporary Cryptology*, ser. Advanced Courses in Mathematics -

CRM Barcelona. Birkhsuser Basel, 2005, pp. 41–87. [Online]. Available: http://dx.doi.org/10.1007/3-7643-7394-6_2

[25] R. Cramer, V. Daza, I. Gracia, J. J. Urroz, G. Leander, J. Martí-Farré, and C. Padró, "On codes, matroids, and secure multiparty computation from linear secret-sharing schemes," *IEEE Transactions on Information Theory*, vol. 54, no. 6, pp. 2644–2657, 2008.

[26] L. Cutillo, R. Molva, and T. Strufe, "Safebook: A privacy-preserving online social network leveraging on real-life trust," *Communications Magazine, IEEE*, vol. 47, no. 12, pp. 94–101, Dec. 2009.

[27] I. Damgård, M. Geisler, M. Krøigaard, and J. Nielsen, "Asynchronous multiparty computation: Theory and implementation," in *Public Key Cryptography PKC 2009*, ser. Lecture Notes in Computer Science, S. Jarecki and G. Tsudik, Eds. Springer Berlin Heidelberg, 2009, vol. 5443, pp. 160–179. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-00468-1_10

[28] F. Didier, "Efficient erasure decoding of Reed-Solomon codes," *CoRR*, vol. abs/0901.1886, 2009.

[29] C. Ding, T. Laihonen, and A. Renvall, "Linear multisecret-sharing schemes and error-correcting codes," *Journal of Universal Computer Science*, vol. 3, no. 9, pp. 1023–1036, Sep. 1997.

[30] M. Djatmiko, D. Schatzmann, X. Dimitropoulos, A. Friedman, and R. Boreli, "Federated flow-based approach for privacy preserving connectivity tracking," in *Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies*, ser. CoNEXT '13. New York, NY, USA: ACM, 2013, pp. 429–440. [Online]. Available: http://doi.acm.org/10.1145/2535372.2535388

[31] M. Djatmiko, D. Schatzmann, A. Friedman, X. Dimitropoulos, and R. Boreli, "Collaborative network outage troubleshooting with secure multiparty computation," *IEEE Communications Magazine*, Nov. 2013.

[32] T. Elgamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *Information Theory, IEEE Transactions on*, vol. 31, no. 4, pp. 469–472, Jul. 1985.

[33] P. Elias, "Coding for two noisy channels," in *Information Theory, The 3rd London Symposium*. Buttersworth's Scientific Publications, Sep. 1955, pp. 61–76.

[34] E. O. Elliott, "Estimates of error rates for codes on burst-noise channels," *Bell System Technical Journal*, vol. 42, pp. 1977–1997, Sep. 1963.

[35] ETSI, "Universal Mobile Telecommunications System (UMTS); Quality of Service (QoS) concept and architecture (3GPP TS 23.107 version 5.4.0 Release 5)," 2002.

[36] B. Fan, D. G. Andersen, M. Kaminsky, and K. Papagiannaki, "Balancing throughput, robustness, and in-order delivery in P2P VoD," in *Proc. CoNEXT*, Dec. 2010.

[37] M. Franklin and M. Yung, "Communication complexity of secure computation (extended abstract)," in *Proceedings of the twenty-fourth annual ACM symposium on Theory of computing*, ser. STOC '92. New York, NY, USA: ACM, 1992, pp. 699–710. [Online]. Available: http://doi.acm.org/10.1145/129712.129780

[38] R. Gallager, "Low-density parity-check codes," *Information Theory, IRE Transactions on*, vol. 8, no. 1, pp. 21–28, Jan. 1962.

[39] P. Garbacki, D. H. J. Epema, J. Pouwelse, and M. van Steen, "Offloading servers with collaborative video on demand," in *Proceedings of the 7th international conference on Peer-to-peer systems*, ser. IPTPS'08. Berkeley, CA, USA: USENIX Association, 2008, p. 6. [Online]. Available: http://dl.acm.org/citation.cfm?id=1855641.1855647

[40] E. N. Gilbert, "Capacity of a burst-noise channel," *Bell System Technical Journal*, vol. 39, pp. 1253–1265, Sep. 1960.

[41] O. Goldreich, S. Micali, and A. Wigderson, "How to play any mental game," in *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, ser. STOC '87. New York, NY, USA: ACM, 1987, pp. 218–229. [Online]. Available: http://doi.acm.org/10.1145/28395.28420

[42] V. Goyal, "Multiple description coding: compression meets the network," *Signal Processing Magazine, IEEE*, vol. 18, no. 5, pp. 74–93, Sep. 2001.

[43] S. Guha, K. Tang, and P. Francis, "Noyb: privacy in online social networks," in *Proceedings of the first workshop on Online social networks*, ser. WOSN '08. New York, NY, USA: ACM, 2008, pp. 49–54. [Online]. Available: http://doi.acm.org/10.1145/1397735.1397747

[44] F. Günther, M. Manulis, and T. Strufe, "Cryptographic treatment of private user profiles," in *Financial Cryptography Workshops*, ser. Lecture Notes in Computer

Science, G. Danezis, S. Dietrich, and K. Sako, Eds., vol. 7126. Springer, 2011, pp. 40–54.

[45] D. Hankerson, A. J. Menezes, and S. Vanstone, "Finite field arithmetic," in *Guide to Elliptic Curve Cryptography*, ser. Springer Professional Computing. Springer New York, 2004, pp. 25–73. [Online]. Available: http://dx.doi.org/10.1007/0-387-21846-7_2

[46] L. Harn, "Comment on "multistage secret sharing based on one-way function"," *Electronics Letters*, vol. 31, no. 4, p. 262, Feb. 1995.

[47] J. He and E. Dawson, "Multistage secret sharing based on one-way function," *Electronics Letters*, vol. 30, no. 19, pp. 1591–1592, Sep. 1994.

[48] ——, "Multisecret-sharing scheme based on one-way function," *Electronics Letters*, vol. 31, no. 2, pp. 93–95, Jan. 1995.

[49] Y. Huang, P. Chapman, and D. Evans, "Privacy-preserving applications on smartphones," in *Proceedings of the 6th USENIX Conference on Hot Topics in Security*, ser. HotSec'11. Berkeley, CA, USA: USENIX Association, 2011, p. 4. [Online]. Available: http://dl.acm.org/citation.cfm?id=2028040.2028044

[50] ITU, "T-rec-g.107-200503-i: The e-model, a computational model for use in transmission planning," Tech. Rep., 2005.

[51] M. Iwamoto and H. Yamamoto, "Strongly secure ramp secret sharing schemes for general access structures," *Inf. Process. Lett.*, vol. 97, no. 2, pp. 52–57, 2006.

[52] W.-A. Jackson and K. M. Martin, "A combinatorial interpretation of ramp schemes," *Australasian Journal of Combinatorics*, vol. 14, pp. 51–60, 1996.

[53] S. Jahid, P. Mittal, and N. Borisov, "Easier: encryption-based access control in social networks with efficient revocation," in *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*, ser. ASIACCS '11. New York, NY, USA: ACM, 2011, pp. 411–415. [Online]. Available: http://doi.acm.org/10.1145/1966913.1966970

[54] D. Jurca, J. Chakareski, J.-P. Wagner, and P. Frossard, "Enabling adaptive video streaming in P2P systems," *IEEE Communications Magazine*, vol. 45, no. 6, pp. 108–114, 2007.

[55] J. Korhonen, *Introduction to 3G mobile communications*, ser. Artech House mobile communications series. Artech House, 2001. [Online]. Available: http://books.google.com.au/books?id=bfxSAAAAMAAJ

[56] H. Krawczyk, "Secret sharing made short," in *CRYPTO*, ser. Lecture Notes in Computer Science, D. R. Stinson, Ed., vol. 773. Springer, 1993, pp. 136–146.

[57] J. Kurihara, S. Kiyomoto, K. Fukushima, and T. Tanaka, "A new (k,n)-threshold secret sharing scheme and its extension," in *Information Security*, ser. Lecture Notes in Computer Science, T.-C. Wu, C.-L. Lei, V. Rijmen, and D.-T. Lee, Eds. Springer Berlin Heidelberg, 2008, vol. 5222, pp. 455–470. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-85886-7_31

[58] C. Labovitz, S. Iekel-Johnson, D. McPherson, J. Oberheide, and F. Jahanian, "Internet inter-domain traffic," in *Proceedings of the ACM SIGCOMM 2010 Conference*, ser. SIGCOMM '10. New York, NY, USA: ACM, 2010, pp. 75–86. [Online]. Available: http://doi.acm.org/10.1145/1851182.1851194

[59] J. Lacan and J. Fimes, "A construction of matrices with no singular square submatrices," in *International Conference on Finite Fields and Applications*, 2003, pp. 145–147. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-24633-6_11

[60] D. A. Levin, Y. Peres, and E. L. Wilmer, *Markov chains and mixing times*. American Mathematical Society, 2008. [Online]. Available: http://pages.uoregon.edu/dlevin/MARKOV/

[61] S.-Y. Li, R. Yeung, and N. Cai, "Linear network coding," *Information Theory, IEEE Transactions on*, vol. 49, no. 2, pp. 371–381, Feb. 2003.

[62] Z. Li, J. Sun, and J. Li, "A novel secret sharing scheme based on minimal linear codes," *Wuhan University Journal of Natural Sciences*, vol. 18, no. 5, pp. 407–412, 2013. [Online]. Available: http://dx.doi.org/10.1007/s11859-013-0949-1

[63] H.-Y. Lin and W.-G. Tzeng, "A secure erasure code-based cloud storage system with secure data forwarding," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 23, no. 6, pp. 995–1003, Jun. 2012.

[64] S. Lin and D. Costello, Jr., *Error Control Coding: Fundamentals and Applications*. Prentice Hall, Englewood Cliffs, NJ., 1983.

[65] S. Lin and D. J. Costello, *Error control coding*. Prentice-hall Englewood Cliffs, 2004, vol. 123.

[66] Y. Liu, K. P. Gummadi, B. Krishnamurthy, and A. Mislove, "Analyzing Facebook privacy settings: user expectations vs. reality," in *Proceedings of the*

2011 ACM SIGCOMM conference on Internet measurement conference, ser. IMC '11. New York, NY, USA: ACM, 2011, pp. 61–70. [Online]. Available: http://doi.acm.org/10.1145/2068816.2068823

[67] Z. Liu, Y. Shen, S. S. Panwar, K. W. Ross, and Y. Wang, *P2P video live streaming with MDC: Providing incentives for redistribution.* IEEE, 2007, pp. 48–51. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4284583

[68] M. Lucas and N. Borisov, "flybyNight: mitigating the privacy risks of social networking," in *Proceedings of the 5th Symposium on Usable Privacy and Security*, ser. SOUPS '09. New York, NY, USA: ACM, 2009, pp. 37:1–37:1. [Online]. Available: http://doi.acm.org/10.1145/1572532.1572577

[69] W. Luo, Q. Xie, and U. Hengartner, "Facecloak: An architecture for user privacy on social networking sites," in *Proceedings of the 2009 International Conference on Computational Science and Engineering - Volume 03*, ser. CSE '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 26–33. [Online]. Available: http://dx.doi.org/10.1109/CSE.2009.387

[70] D. J. C. Mackay, "Fountain codes," *Communications, IEE Proceedings-*, vol. 152, no. 6, pp. 1062–1068, 2005. [Online]. Available: http://dx.doi.org/10.1049/ip-com:20050237

[71] N. Magharei and R. Rejaie, "Understanding mesh-based peer-to-peer streaming," in *Proceedings of the 2006 International Workshop on Network and Operating Systems Support for Digital Audio and Video*, ser. NOSSDAV '06. New York, NY, USA: ACM, 2006, pp. 1–10. [Online]. Available: http://doi.acm.org/10.1145/1378191.1378204

[72] D. Many, M. Burkhart, and X. Dimitropoulos, "Fast private set operations with sepia," ETZ G93; Gloriastrasse 35; 8092 Zrich, Mar. 2012.

[73] E. Martinian and C.-E. Sundberg, "Burst erasure correction codes with low decoding delay," *Information Theory, IEEE Transactions on*, vol. 50, no. 10, pp. 2494–2502, Oct. 2004.

[74] J. L. Massey, "Minimal codewords and secret sharing," in *Proceedings of the 6th Joint Swedish-Russian International Workshop on Information Theory*, 1993, pp. 276–279.

[75] ——, "Some applications of coding theory in cryptography," *Codes and Ciphers: Cryptography and Coding IV*, pp. 33–47, 1995.

[76] R. J. McEliece and D. V. Sarwate, "On sharing secrets and Reed-Solomon codes," *Commun. ACM*, vol. 24, no. 9, pp. 583–584, Sep. 1981. [Online]. Available: http://doi.acm.org/10.1145/358746.358762

[77] McGraw-Hill, *McGraw-Hill concise encyclopedia of engineering*, ser. Concise Encyclopedia Series. McGraw-Hill, 2002. [Online]. Available: http://books.google.com.au/books?id=zLBaAAAAYAAJ

[78] M. Mignotte, "How to share a secret," in *Cryptography*, ser. Lecture Notes in Computer Science, T. Beth, Ed. Springer Berlin Heidelberg, 1983, vol. 149, pp. 371–375. [Online]. Available: http://dx.doi.org/10.1007/3-540-39466-4_27

[79] S. Milani, S. Busato, and G. Calvagno, "Multiple description peer-to-peer video streaming using coalitional games," in *EURASIP*, ser. European Signal Processing Conference (EUSIPCO 2011), 2011.

[80] A. Narayanan and V. Shmatikov, "Myths and fallacies of "personally identifiable information"," *Commun. ACM*, vol. 53, no. 6, pp. 24–26, Jun. 2010. [Online]. Available: http://doi.acm.org/10.1145/1743546.1743558

[81] V. Nikolaenko, S. Ioannidis, U. Weinsberg, M. Joye, N. Taft, and D. Boneh, "Privacy-preserving matrix factorization," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, ser. CCS '13. New York, NY, USA: ACM, 2013, pp. 801–812. [Online]. Available: http://doi.acm.org/10.1145/2508859.2516751

[82] H. R. Oh and H. Song, "Mesh-pull-based p2p video streaming system using fountain codes," in *Computer Communications and Networks (ICCCN), 2011 Proceedings of 20th International Conference on*, Aug. 2011, pp. 1–6.

[83] L.-J. Pang and Y.-M. Wang, "A new (t, n) multi-secret sharing scheme based on shamir's secret sharing." *Applied Mathematics and Computation*, vol. 167, no. 2, pp. 840–848, 2005.

[84] M. B. Paterson and D. R. Stinson, "A simple combinatorial treatment of constructions and threshold gaps of ramp schemes," *Cryptography and Communications*, vol. 5, no. 4, pp. 229–240, 2013. [Online]. Available: http://dx.doi.org/10.1007/s12095-013-0082-1

[85] J. Pieprzyk and X.-M. Zhang, "Ideal threshold schemes from mds codes," in *Information Security and Cryptology ICISC 2002*, ser. Lecture Notes in Computer Science, P. Lee and C. Lim, Eds. Springer Berlin Heidelberg, 2003, vol. 2587, pp. 253–263. [Online]. Available: http://dx.doi.org/10.1007/3-540-36552-4_18

[86] J. Plank and L. Xu, "Optimizing Cauchy Reed-Solomon codes for fault-tolerant network storage applications," in *Network Computing and Applications, 2006. NCA 2006. Fifth IEEE International Symposium on*, 2006, pp. 173–180.

[87] S. Puducheri and T. Fuja, "Coding versus feedback: Hybrid arq protocols for the packet erasure channel," in *Information Theory Proceedings (ISIT), 2010 IEEE International Symposium on*, Jun. 2010, pp. 1988–1992.

[88] M. O. Rabin, "Efficient dispersal of information for security, load balancing, and fault tolerance," *J. ACM*, vol. 36, no. 2, pp. 335–348, Apr. 1989. [Online]. Available: http://doi.acm.org/10.1145/62044.62050

[89] G. Raja and M. J. Mirza, "In-loop deblocking filter for jvt h.264/avc," in *Proceedings of the 5th WSEAS International Conference on Signal Processing, Robotics and Automation*. Stevens Point, Wisconsin, USA: World Scientific and Engineering Academy and Society (WSEAS), 2006, pp. 235–240. [Online]. Available: http://dl.acm.org/citation.cfm?id=1365904.1365944

[90] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of the Society for Industrial and Applied Mathematics*, vol. 8, no. 2, pp. 300–304, 1960. [Online]. Available: http://dx.doi.org/10.1137/0108018

[91] J. K. Resch and J. S. Plank, "AONT-RS: blending security and performance in dispersed storage systems," in *FAST-2011: 9th Usenix Conference on File and Storage Technologies*, Feb. 2011, pp. 191–202.

[92] I. Richardson, *H.264 and MPEG-4 video compression: video coding for next-generation multimedia*. Wiley, 2003. [Online]. Available: http://books.google.com.au/books?id=ECVV_G_qsxUC

[93] L. Rizzo, "Effective erasure codes for reliable computer communication protocols," *SIGCOMM Comput. Commun. Rev.*, vol. 27, pp. 24–36, Apr. 1997. [Online]. Available: http://doi.acm.org/10.1145/263876.263881

[94] E. Sava and C. Ko, "Finite field arithmetic for cryptography," *Circuits and Systems Magazine, IEEE*, vol. 10, no. 2, pp. 40–56, 2010.

[95] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the h.264/avc standard," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 17, no. 9, pp. 1103–1120, Sep. 2007.

[96] E. Setton, P. Baccichet, and B. Girod, "Peer-to-peer live multicast: A video perspective," *Proceedings of the IEEE*, vol. 96, no. 1, pp. 25–38, Jan. 2008.

[97] A. Shamir, "How to share a secret," *C. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979. [Online]. Available: http://doi.acm.org/10.1145/359168.359176

[98] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656, Jul. 1948. [Online]. Available: http://cm.bell-labs.com/cm/ms/what/shannonday/shannon1948.pdf

[99] R. C. Singleton, "Maximum distance q -nary codes," *Information Theory, IEEE Transactions on*, vol. 10, no. 2, pp. 116–118, Apr. 1964.

[100] A. Soro, M. Cunche, J. Lacan, and V. Roca, "Erasure codes with a banded structure for hybrid iterative-ml decoding," in *Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE*, Dec. 2009, pp. 1–6.

[101] J. Sundararajan, D. Shah, and M. Medard, "Online network coding for optimal throughput and delay - the three-receiver case," in *Information Theory and Its Applications, 2008. ISITA 2008. International Symposium on*, Dec. 2008, pp. 1–6.

[102] J. Sundararajan, D. Shah, M. Medard, M. Mitzenmacher, and J. Barros, "Network coding meets tcp," in *INFOCOM 2009, IEEE*, Apr. 2009, pp. 280–288.

[103] P. Tournoux, E. Lochin, J. Lacan, A. Bouabdallah, and V. Roca, "On-the-fly erasure coding for real-time video applications," *Multimedia, IEEE Transactions on*, vol. 13, no. 4, pp. 797–812, Aug. 2011.

[104] P. Tournoux, E. Lochin, J. Leguay, and J. Lacan, "Robust streaming in delay tolerant networks," in *Communications (ICC), 2010 IEEE International Conference on*, May 2010, pp. 1–5.

[105] J. Ugander, B. Karrer, L. Backstrom, and C. Marlow, "The anatomy of the Facebook social graph," *CoRR*, vol. abs/1111.4503, 2011.

[106] B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi, "On the evolution of user interaction in Facebook," in *Proceedings of the 2nd ACM SIGCOMM Workshop on Social Networks (WOSN'09)*, Aug. 2009.

[107] A. Vitali, "Multiple description coding-a new technology for video streaming over the internet," *EBU Technical Review*, 2007.

[108] L.-H. Vu, K. Aberer, S. Buchegger, and A. Datta, "Enabling secure secret sharing in distributed online social networks," in *Proceedings of the 2009 Annual Computer Security Applications Conference*, ser. ACSAC '09. Washington, DC, USA: IEEE, 2009, pp. 419–428. [Online]. Available: http://dx.doi.org/10.1109/ACSAC.2009.46

[109] J.-P. Wagner, J. Chakareski, and P. Frossard, "Streaming of scalable video from multiple servers using rateless codes," in *Multimedia and Expo, 2006 IEEE International Conference on*, Jul. 2006, pp. 1501–1504.

[110] F. Wang, L. Gu, S. Zheng, Y. Yang, and Z. Hu, "A novel verifiable dynamic multi-policy secret sharing scheme," in *Proceedings of the 12th international conference on Advanced communication technology*, ser. ICACT'10. Piscataway, NJ, USA: IEEE Press, 2010, pp. 1474–1479. [Online]. Available: http://dl.acm.org/citation.cfm?id=1833006.1833117

[111] M. Wang and B. Li, "Network coding in live peer-to-peer streaming," *Multimedia, IEEE Transactions on*, vol. 9, no. 8, pp. 1554–1567, Dec. 2007.

[112] ——, "R2: Random push with random network coding in live peer-to-peer streaming," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 9, pp. 1655–1666, 2007.

[113] A. Waseda and M. Soshi, "Consideration for multi-threshold multi-secret sharing schemes," in *ISITA 2012*, Oct. 2012, pp. 265–269.

[114] M. Wernke, F. Dürr, and K. Rothermel, "Pshare: Position sharing for location privacy based on multi-secret sharing," in *PerCom*, S. Giordano, M. Langheinrich, and A. Schmidt, Eds. IEEE, 2012, pp. 153–161.

[115] S. Wicker and V. Bhargava, *Reed-Solomon codes and their applications*. Wiley, 1999. [Online]. Available: http://books.google.com.au/books?id=yws55Rx1orEC

[116] H. Yamamoto, "Secret sharing system using (k, l, n) threshold scheme," *Electronics and Communications in Japan (Part I: Communications)*, vol. 69, no. 9, pp. 46–54, 1986. [Online]. Available: http://dx.doi.org/10.1002/ecja.4410690906

[117] C.-C. Yang, T.-Y. Chang, and M.-S. Hwang, "A (t,n) multi-secret sharing scheme," *Applied Mathematics and Computation*, vol. 151, no. 2, pp. 483–490, 2004.

[118] A. C. Yao, A. C. Yao, A. C. Yao, and A. C. Yao, "Protocols for secure computations," in *Foundations of Computer Science, 1982. SFCS '08. 23rd Annual Symposium on*, Nov. 1982, pp. 160–164.

[119] H. Yildiz and C. Kruegel, "Detecting social cliques for automated privacy control in online social networks," *Pervasive Computing and Communications Workshops, IEEE International Conference on*, pp. 353–359, 2012.

[120] J. Yuan and C. Ding, "Secret sharing schemes from three classes of linear codes," *Information Theory, IEEE Transactions on*, vol. 52, no. 1, pp. 206–212, Jan. 2006.