

Secure Message Transmission and its Applications

Stelios Erotokritou

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
of
University College London.

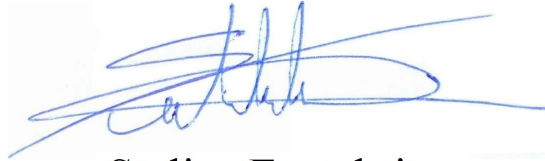
Department of Computer Science
University College London

June 2016

Declaration

I, Stelios Erotokritou hereby declare that the work presented in this dissertation submission entitled “Secure Message Transmission and its Applications” in partial fulfilment of the requirements for the degree of Doctor of Philosophy of University College London, to the best of my knowledge and belief, contains no material previously published or written by another person and is an authentic record and presentation of my own work during a period of seven and a half years from October 2007 to April 2015 under the supervision of Professor Yvo Desmedt, Department of Computer Science, University College London.

Where information has been derived from other sources, I confirm that this has been indicated in the thesis.



Stelios Erotokritou
June 2016

Abstract

In this thesis we focus on various aspects of secure message transmission protocols. Such protocols achieve the secure transmission of a message from a sender to a receiver - where the term “secure” encapsulates the notion of privacy and reliability of message transmission. These two parties are connected using an underlying network in which a static computationally unlimited active adversary able to corrupt up to t network nodes is assumed to be present. Such protocols are important to study as they are used extensively in various cryptographic protocols and are of interest to other research areas such as ad-hoc networks, military networks amongst others.

Optimal bounds for the number of phases (communication from sender to receiver or vice versa), connectivity requirements (number of node disjoint network paths connecting sender and receiver - denoted by n), communication complexity (complexity of the number of field elements sent - where \mathbb{F} is the finite field used and $|\mathbb{F}| = q$) and transmission complexity (proportion of communication complexity to complexity of secrets transmitted) for secure message transmission protocols have been proven in previous work. In the one-phase model it has been shown that $n \geq 3t + 1$ node disjoint paths are required to achieve perfect communication. In the two phase model only $n \geq 2t + 1$ node disjoint paths are necessary. This connectivity is also the required bound for almost perfectly secure one-phase protocols - protocols which achieve perfect privacy but with a negligible probability may fail to achieve reliability. In such cases the receiver accepts a different message to that transmitted by the sender or does not accept any message.

The main focus of recent research in secure message transmission protocols has been to present new protocols which achieve optimal transmission complexity. This has been achieved through the transmission of multiple messages. If a protocol has a communication complexity of $O(n^3)$ field elements, to achieve optimal transmission complexity $O(n^2)$ secrets will have to be communicated. This has somewhat ignored the simplification and improvement of protocols which securely transmit a *single* secret. Such improvements include constructing more efficient protocols with regards to communication complexity, computational complexity and the number of field elements sent throughout the whole protocol.

In the thesis we first consider one-phase almost perfectly secure message transmission and present two *new* protocols which improve on previous work. We present a polynomial time protocol of $O(n^2)$ communication complexity which at the time of writing this thesis, is computationally more efficient than any other protocol of similar communication complexity for the almost perfectly secure transmission of a single message.

Even though our first almost perfectly secure transmission protocol is of polynomial time, it is important to study other protocols also and improve previous work presented by other researchers. This is the idea behind the second one-phase almost perfectly secure message transmission protocol we present which requires an exponential complexity of field operations but lower ($O(n)$) communication complexity. This protocol also improves on previous protocols of similar communication complexity, requiring in the order of $O(\log q)$ less computation to complete - where q denotes the size of the finite field used. Even though this protocol is of exponential time, for small values of n (e.g. when $t = 1$, $t = 2$ or $t = 3$) it may be beneficial to use this protocol for almost perfectly secure communication as opposed to using the polynomial time protocol. This is because less field elements need to be transmitted over the whole network which connects a sender and a receiver. Furthermore, an optimal almost perfectly secure transmission protocol will be one with $O(n)$ communication complexity and with polynomial computational complexity. We hope that in the future, other researchers will be inspired by our proposed protocol, improve on our work and ideally achieve these optimal results.

We also consider multi-phase protocols. By combining various cryptographic schemes, we present a *new* two-phase perfectly secure single message transmission protocol. At the time of writing this thesis, the protocol is the most efficient protocol when considering communication complexity. Our protocol has a communication complexity of $O(n^2)$ compared to $O(n^3)$ of previous work thus improving on the communication complexity by an order of $O(n)$ for the perfectly secure message transmission of a single message.

This protocol is then extended to a three phase protocol where a multi-recipient broadcast end channel network setting is considered. As opposed to point to point networks where a path from a sender reaches a single receiver, this network model is *new* in the field of message transmission protocols. In this model each path from a sender reaches multiple receivers, with all receivers receiving the same information from their common network communication channel. We show how the use of this protocol upon such a network can lead to great savings in the transmission and computation carried out by a single sender. We also discuss the importance and relevance of such a multi-recipient setting to practical applications.

The first protocols in the field of perfectly secure message transmission with a human receiver are also presented. This is a topic proposed by my supervisor Professor Yvo Desmedt for which I constructed solutions. In such protocols, one of the communicating parties is considered to be a human who does not have access to a computational device. Because of this, solutions for such protocols need to be computationally efficient and computationally simple so that they can be executed by the human party. Experiments with human participants were carried out to assess how easily and accurately human parties used the proposed protocols. The experimental results are presented and these identify how well human participants used the protocols.

In addition to the security of messages, we also consider how one can achieve anonymity of message transmission protocols. For such protocols, considering a single-receiver multi-sender scenario, the presence of a t -threshold bounded adversary and the transmission of multiple secrets (as many as the number of sender), once the protocols ends one should not be able to identify the sender of a received message. Considering a passive and active adversary new protocols are presented which achieve the secure and anonymous transmission of messages in

the information-theoretic security model. Our proposed solutions can also be applied (with minor alterations) to the dual problem when a single-sender multi-recipient communication setting is considered.

The contributions of the thesis are primarily theoretical - thus no implementation of the proposed protocols was carried out. Despite this, we reflect on practical aspects of secure message transmission protocols. We review the feasibility of implementing secure message transmission protocols in general upon various networks - focusing on the Internet which can be considered as the most important communication network at this time. We also describe in theory how concepts of secure message transmission protocols could possibly be used in practical implementations for secure communication on various existing communication networks.

Open problems that remain unsolved in the research area of the proposed protocols are also discussed and we hope that these inspire research and future solutions for the design (and implementation) of better and more efficient secure message transmission protocols.

Acknowledgements

Primarily I would like to thank my thesis advisor, Professor Yvo Desmedt. I am most grateful to him both on a personal and academic level. It has been an absolute pleasure for me to know him as a person, to have been his student and to learn from him.

His work ethic, motivation, encouragement and demand for thoroughness has greatly improved me as a student and as a researcher. Through his teaching I have improved and developed essential skills required for a researcher to possess and I thank him for that.

I would also like to thank him for his help during my work - helping me in the identification of errors and providing me with ideas and references to help me find better and correct solutions while at the same time helping me increase my knowledge and become more cautious with my work.

Another person I would also like to thank is Dr. Paul Kearney who was my industrial contact with British Telecom (BT). He was always very helpful and was very active in the search for my primary field of research - both from a theoretical and practical viewpoint. He also helped me greatly with gaining access to various industrial contacts at BT labs - which aided me greatly with my work.

I would also like to thank British Telecom which partly funded my PhD thesis and allowed me to use its resources during my research period.

Special thanks must also be given to Theo Dimitrakos and Srijith Nair of BT labs with whom I worked with during my placement at BT. Working with them was both constructive and enjoyable. Additionally, I am also grateful to all other BT employees that helped me during my time at BT.

Thanks also go to Qiushi Yang with whom we worked together in certain aspects of my work and was always willing to help in various discussions concerning my work.

I would also like to thank Professor Nicolas Courtois and Professor Jens Groth who were always willing to help and give me advice whenever required.

Finally, I would like to thank my PhD examiners - Professor Brad Karp and Professor Matthias Fitzi, whose critical assessment of my thesis coupled with their invaluable advice and suggestions have allowed me to improve the quality of my thesis and myself as a research student. I am most grateful to them for their input and for the time they provided me with to make the necessary changes and additions.

Contents

1	Introduction	12
1.1	Secure Message Transmission Schemes	12
1.2	The Importance of Secure Message Transmission Protocols	13
1.2.1	Message Transmission Protocols as Fundamental Cryptographic Protocols	14
1.2.2	Message Transmission Protocols for Key Exchange Purposes	15
1.2.3	Message Transmission Protocols for Information-Theoretic Privacy	15
1.2.4	The Importance of Studying Secure Message Transmission Protocols	17
1.3	Previous Work	18
1.4	Critical Review	23
1.5	Context and Overview of Thesis Focus	24
1.6	Overview and Organization of Thesis	25
2	Background	28
2.1	Environment of Message Transmission Protocols	28
2.2	Network Models	28
2.2.1	Point-to-Point Networks	29
2.2.2	Network with Broadcast End Channels	31
2.3	Security Model	32
2.4	Adversary Presence	34
2.5	Efficiency of Protocols	35
2.6	Secret Sharing	37
2.6.1	Shamir Secret Sharing	37
2.6.2	Alternative Secret Sharing Scheme Friendly to Humans	39
2.7	Privacy Amplification	40
3	Almost Perfectly Secure One-Phase Transmission Protocols	42
3.1	Environment of Almost Perfectly Secure One-Phase Protocols	43
3.2	One-Phase Polynomial Time Protocol	43
3.2.1	Protocol Main Idea	44
3.2.2	Main Protocol Techniques	44
3.2.3	Protocol Description	46
3.2.4	Security and Complexity Analysis	47

3.3	One-Phase Exponential Time Protocol	58
3.3.1	Protocol Main Idea	58
3.3.2	Formal Protocol Description	58
3.3.3	Security and Complexity Analysis	60
3.4	Comparison to Previous Work and Future Work	64
3.4.1	Comparison of Polynomial Time Protocol to Srinathan et al. Protocol	64
3.4.2	Comparison of Exponential Time Protocol to Kurosawa-Suzuki Protocol	66
3.4.3	Future Work	69
4	Multi-Phase Perfectly Secure Message Transmission Protocols	70
4.1	Efficient Two-Phase Perfectly Secure Message Transmission Protocol	70
4.1.1	Protocol Main Idea	71
4.1.2	Main Protocol Techniques	71
4.1.3	Formal Protocol Description	73
4.1.4	Security and Complexity Analysis	74
4.1.5	Two-Phase PSMT with lower Communication Complexity	75
4.2	Three-Phase Multi-Recipient Perfectly Secure Message Transmission Protocol	78
4.2.1	Environment of Message Transmission Protocol	78
4.2.2	Relevance and Practical Applications	79
4.2.3	Protocol Specifics and Main Idea	82
4.2.4	Formal Protocol Description	83
4.2.5	Security, Complexity and Comparison Analysis	84
4.2.6	Transmitting More Secret Messages	86
4.3	Future Work	88
5	Perfectly Secure Message Transmission With Human Computation	90
5.1	Motivation for Perfectly Secure Message Transmission with Human Computation	91
5.1.1	Secure Human Computation	92
5.2	Set Systems	93
5.2.1	Generalized Verifier Set Systems and Covering Designs	93
5.2.2	Constructing Set Systems	94
5.3	One-Phase Mod 10 PSMT with a Human Receiver	95
5.3.1	One-Phase PSMT with a Human Using a Disjoint Set System	96
5.3.2	Optimally Connected One-Phase PSMT with a Human	96
5.4	Two-Phase Mod 10 PSMT with a Human Receiver	97
5.4.1	Two-Phase PSMT with a Human Using a Disjoint Set System	98
5.4.2	Optimally Connected Two-Phase PSMT with a Human	99
5.5	Practically Feasible Protocols for Small Values of t	99
5.6	Human Friendly Addition Mod 10	99
5.6.1	Addition Mod 10 Using Permutations	100
5.6.2	Human PSMT Permutation Protocol	101
5.6.3	Human PSMT Permutation Protocols Against an Active Adversary	104

5.7	Experimental Evaluation of the Human Friendly Addition	104
5.7.1	Demographics of Participants	104
5.7.2	Experimental Procedure	105
5.7.3	Experimental Results	106
5.7.4	Experimental Results Discussion	107
5.7.5	Relativity of Experiments to PSMT Protocols and Discussion of Results	108
5.8	Comparing Human Computation Protocols to other PSMT Protocols	109
5.9	Future Work	110
6	Perfectly Secure and Anonymous Message Transmission	112
6.1	Perfectly Secure and Anonymous Communication Protocols	112
6.2	Private and Anonymous Communication Protocol against a Passive Adversary .	114
6.2.1	Main Idea	114
6.2.2	Formal Protocol Description	114
6.2.3	Security Analysis	115
6.3	Private and Anonymous Communication Protocol against an Active Adversary	117
6.3.1	Active Adversary Deviation Methods	117
6.3.2	Formal Protocol Description	118
6.3.3	Security Analysis	118
6.4	Practical Applications of Anonymous and Perfectly Secure Message Transmis- sion Protocols	119
7	Implementing Secure Message Transmission Protocols	121
7.1	Threat Model	121
7.2	Scope and Importance	122
7.3	Implementation Issues	125
7.3.1	Implementations of Message Transmission Protocols	125
7.3.2	Past Internet	125
7.3.3	Present Internet	126
7.3.4	Alternative Internet Options and their Issues	130
7.3.5	Implementation Upon other Networks	131
7.4	Using Concepts of PSMT on the World Wide Web	132
7.5	Importance of Implementation	134
8	Conclusions and Future Work	136
8.1	Conclusions	136
8.2	Future Work	137
A	Other Security Definitions	156
A.1	Current Security Definition - Probabilistic Reliability	156
A.2	Current Security Definition - Probabilistic Failure	156
A.3	Comparison of the two Security Definitions	157

B	Adversaries in Secure Message Transmission Protocols	158
C	Identifying the Best Adversary Strategy	160
D	Reconstruction of Mod 10 Secret Shared Secrets	162
E	Set Systems and Their Constructions	165
E.1	Constructions from the Literature	165
E.2	Constructions from Covering Designs	166
F	Permutation voting experiments	168
G	Mod 10 Voting Experiments	172

List of Figures

2.1	Network topology with all network nodes of a transmission protocol.	30
2.2	Example of a non-directional communication network.	30
2.3	Example of a directional communication network.	31
2.4	Example of network with broadcast end channels.	32
5.1	Representing addition mod10 through diagrams of permutations.	101
5.2	Initial view of human receiver.	102
5.3	Two diagrams to be received by the human receiver.	102
5.4	The human receiver will place the first diagram in the appropriate position. . .	103
5.5	The human receiver will place the second diagram in the appropriate position. .	103
5.6	The human receiver traces the radio button corresponding to the start of the traced line. In the example, the secret digit is 2.	103
5.7	Demographics of participants.	105
5.8	Experimental results of the mod10 voting scheme	106
5.9	Experimental results of the permutation voting scheme	107
C.1	Evaluations of $AY + AX + BX > A$ presented using a graph.	161
D.1	Instructions on how to reconstruct a mod10 secret shared secret in figure format	164

Chapter 1

Introduction

1.1 Secure Message Transmission Schemes

The main research area studied in this thesis is that of secure message transmission schemes. Such schemes have been used throughout history such as when Julius Caesar obscured the meaning of vital military orders destined for his generals against untrusted messengers and when Sir Francis Walsingham - the founder of England's first secret service, used coded letters to foil plots against Elizabeth I. Such protocols have also been used in more recent history, from military communications during World War II and espionage during the Cold War, to secure communication which occurs over the Internet. All these examples highlight the fact that secure transmission of messages between two parties - a sender and a receiver, has been an important and vital objective to achieve throughout history.

In practise, most secure message transmission schemes have focused on using encryption algorithms. In such schemes, the sender encrypts a secret message using a chosen encryption algorithm to produce a ciphertext which is then transmitted to the receiver. Upon receiving the ciphertext, the receiver carries out decryption to obtain the secret message in unencrypted cleartext format. As good encryption algorithms produce ciphertext which should appear as a random string of bits, it is hoped that if during transmission any third adversarial party views the ciphertext, they will not be able to gain any knowledge about the secret message.

Encryption algorithms used in practise throughout history include amongst others the Caesar cipher - supposedly used by Julius Caesar, Enigma machines - used by Nazi Germany during World War II [98], the Data Encryption Standard (DES) - the first publicly available encryption algorithm used for encryption of data both in industry and government [140], and the Advanced Encryption Standard (AES) which has been adopted by the U.S. government [139] and is now used worldwide.

An important question to ask is how a secure message transmission scheme can remain secure throughout time? The answer should take into account that in the future, cryptanalytic attacks will improve and computational capabilities will be far greater than the current ones. The use of encryption algorithms which are based on computational difficulty *may* leak secret information earlier than the originally planned period of secrecy¹, or it may leak secrets that should *never* be learned. Using encryption algorithms which are based on computational

¹By period of secrecy we refer to the length of time a secret must remain secret. Beyond this period of secrecy it is irrelevant if the value/message of the secret is revealed or not.

difficulty, may not be the best solution for some practical applications.

The one-time pad is a type of encryption which (when implemented correctly) is secure from any form of cryptanalysis and also remains secure despite any technological advancements. This is because it achieves information-theoretic security even when considering an adversary with unlimited computing power. Given a secret message M , information-theoretic secrecy is achieved when the adversary is not able to recover any probabilistic information on M besides what the adversary already knew a priori to the execution of the protocol - and despite any observations the adversary carried out whilst the protocol was executing.

Secure message transmission protocols studied in this thesis achieve information-theoretic security of secrets transmitted between two parties. Because of this, they are important protocols to study and improve in research. This is because they are fundamental cryptographic protocols used to implement underlying cryptographic assumptions. Furthermore they can be used for key exchange purposes to initiate communication between two network parties.

In this thesis we present novel protocols in our study of secure message transmission schemes which achieve information-theoretic security. The protocols we present will maintain their security properties for all time and will *not* be affected by any forms of future computational advancements or cryptanalytic attacks. Our focus considers abstract and theoretical network models in which a sender and a receiver can be any two nodes in a network. In such networks, the two parties are connected by node-disjoint network paths comprised by network nodes of an underlying network topology. Such theoretical networks are different to the Internet where the presence of node-disjoint network paths cannot be guaranteed or selected - as will be discussed in Chapter 7. An additional presence in the network is a threshold bounded static and computationally unlimited active adversary. For some of our protocols a passive adversary is considered. Despite the presence of the adversary and any actions it may impose upon a protocol, the protocol should *always* achieve the security properties required by a sender and a receiver.

1.2 The Importance of Secure Message Transmission Protocols

We first identify the reasons why it is important to study secure message transmission protocols - expanding on each of these in the forthcoming subsections.

It is important to study message transmission protocols for the following reasons:

- Message transmission protocols are fundamental cryptographic protocols which find applications in various research areas.
- To enable key exchange between nodes in networks where we consider an adversary and where the infrastructure to enable key exchange (such as public key infrastructure) is not present or is not available.
- To allow for the transmission of secrets with information-theoretic privacy when other cryptographic schemes based on computational hardness and prone to future technological frailties or future attacks cannot or should not be used.

The importance of each of the above reasons is described in detail in the following sub-sections.

1.2.1 Message Transmission Protocols as Fundamental Cryptographic Protocols

Secure message transmission schemes as studied in this thesis are important cryptographic protocols. This is because they provide a guaranteed method of secure communication between a sender and any receiver in a theoretical network setting. Because of this, they are of interest in both a theoretical and a practical viewpoint.

In theory, such protocols find applications in various forms of distributed and secure computation. Such an example is that of secure multi-party computation (MPC). In such protocols, the assumption of a complete graph is made where each pair of protocol participants are connected to each other with a direct, reliable and private communication channel. This is a very strong assumption to make as it is very difficult to achieve a complete graph in a network - especially when the number of participants in a protocol is large. This will require a very large number of links - which will make such a network extremely expensive to set up and expand. Contrary to this, networks are in general built with an underlying network of nodes (routers or switches) which implement the correct routing and forwarding of data from a sender to the intended recipient(s). As complete graphs are rarely found, to implement secure multiparty computation - and indeed any other protocol which assumes a complete graph, the concept of a direct, reliable and private channel between any two participants needs to be *simulated* using secure message transmission protocols - which achieve the required security properties of complete graph edges. It should be pointed out that despite secure message transmission protocols using multiple disjoint paths, the overall effect for a participant using secure message transmission protocols is that of a *single* secure channel.

Secure message transmission protocols are thus fundamental cryptographic protocols used to implement underlying assumptions. It is thus important to simplify such protocols as much as possible so as to simplify protocols which in turn use them.

The importance of secure message transmission protocols is not confined to cryptography alone. When studying such protocols, abstract network topologies are considered. Because of this, such protocols can be deployed in many networks provided they allow for the use of a multiple number of node disjoint paths - an essential requirement for such protocols. Such protocols have found direct applications in various research fields which include sensor networks, ad hoc networks, military networks and have been used in various published papers also [47, 72, 111, 119, 171].

It is worth noting that despite the main motivation for such protocols being communication networks, they can be used in other types of networks, such as human networks [84]. *Concepts* of these protocols have also been used in other aspects of research where networks and secure properties are required, such as in critical infrastructure protection [48, 175].

In more practical scenarios, secure message transmission protocols can be used in different contexts on (real-world) networks. As an example, due to the use of multiple disjoint paths and given the reliability of message transmission such protocols guarantee, this could allow for such protocols to be used as a defence against Denial of Service attacks, directing the flow of information across networks paths free from any form of congestion or attack and thus providing higher reliability guarantees. This was used in [191] to allow two sensor nodes in a sensor network (which do not have a shared key from a key pre-distribution phase) to establish a

secure communication channel between them. This process considered an active adversary and was able to defend itself against denial of service attacks without the involvement of the base station.

1.2.2 Message Transmission Protocols for Key Exchange Purposes

Secure message transmission protocols can be used to initiate communication amongst two parties - a sender and a receiver, in a network. More specifically, they can be used for key exchange purposes.

The use of encryption algorithms (such as DES or AES or other similar symmetric key algorithms) by two parties requires some form of initial setup between them before any secret communication can occur. This not only includes the choice of encryption algorithm to be used, but the essential pre-shared knowledge of an encryption key which should be known by both of the communicating parties. The latter is otherwise known as “key agreement”. Although solutions for this kind of agreement exist, they generally require the use of pre-shared secrets or some form of initial setup - such as a public key infrastructure, to be available. Use of public key encryption methods also suffer from initial setup requirements (such methods require the existence of a *trusted* and secure public key infrastructure).

An important question to ask and examine is how two parties can achieve secure message transmission when key agreement infrastructure is not available to them.

This may be possible for some networks which may be *specifically designed* for secure message transmission and for a specific security parameters (specific values of t). This may be the case for military communication networks in a warzone. In such cases, the network graph, the value of t and the definition of the protocol will be pre-programmed in all network nodes which comprise a network. If this is the case, no initial setup (beyond the initial implementation of the network nodes and network) will be required for the execution of a secure message transmission protocol.

For other networks, it is *assumed* that knowledge about properties of the communication network (such as network graph, value of t) are commonly known by both sender and receiver of a communication.

1.2.3 Message Transmission Protocols for Information-Theoretic Privacy

In this section we describe how message transmission protocols compare to other cryptographic protocols concerning the transmission of secrets in the presence of an adversary.

The advantage of message transmission protocols studied in this thesis is that they achieve information-theoretic privacy. Because of this, such protocols are proven to be secure using a complete and correct mathematical proof. Assuming two communicating parties are connected with a sufficient number of node disjoint network paths (the number of which depends on the protocol used), such protocols will remain secure and will remain so throughout time. To borrow a term from Aumann, Ding and Rabin [8] such protocols achieve *everlasting privacy*. It is important to study such protocols and through research design solutions which as much and as close as possible reach the optimal bounds (for various complexities) identified by various work.

As history has shown in the breaking of Enigma [98], the use of an encryption algorithm

remains secure up until any weaknesses of the encryption algorithm are possibly found.

The technological demise of DES also highlights the fact that encryption algorithms are secure up to the point in time when technology catches up with the encryption algorithm. In the case of DES, this was designed and accepted as a standard in 1976 [193]. In 1998 a machine built by the Electronic Frontier Foundation (EFF) [57] was able to perform a brute force search upon the key space of DES. DES was thus overtaken by technological advancements which proved DES to be computationally insecure. This frailty was a result of DES's small key space (2^{56}) - even though DES as an encryption algorithm has a relatively high resistance to cryptanalysis.

The Advanced Encryption Standard is the latest encryption algorithm chosen by the National Institute of Standards and Technology as the main encryption algorithm to be used by the US government [139]. Despite AES currently being a secure algorithm, a number of different attacks on AES have been found. Two of these attacks which target the full version of AES include Related-Key Cryptanalysis (RKC) introduced at [25] and attacks exploiting Time-Memory-Key (TMK) trade-offs as presented in [26]. While the complexities of such attacks ($2^{99.5}$ AES operations for RKC, 2^{80} for TMK) are much faster than an exhaustive search attack, they are currently not technologically feasible. Despite this, the authors of [24] describe how the use of GPU like special purpose hardware in a high performance computer could theoretically break the full AES in a time frame of as little as one year when using RKC, or in merely one month when performing a TMK attack. Although the supercomputer they describe is very expensive (one trillion dollars) these costs should decrease with time. The importance of [24] is that it suggests that even AES is vulnerable to technological advancements.

Other cryptographic algorithms such as RSA, lattice-based cryptography and other such schemes are based on computationally hard problems. This means that the *assumption* that it is too hard for an adversary to decrypt an encrypted message in a reasonable time is made (but such assumptions are as yet unproven).

The RSA algorithm [157] (upon which some public key infrastructure implementations are based) assumes the presumed difficulty of factoring large integers. The advent of quantum computing though has the potential to "break" RSA through the implementation of Shores Algorithm on a quantum computer [166]. Although quantum computing is still years away from a practical implementation, things may change in the future through the progressive advancement in quantum computing research.

Currently, post-quantum cryptography (cryptographic systems that are not breakable using quantum computers and current quantum algorithms) [23] includes different approaches such as lattice-based cryptography, multivariate cryptography, hash-based signatures amongst others. These are also based on computationally hard problems - for which currently no known quantum algorithms which perform significantly better than the best known non-quantum algorithms have been found. Lattice based cryptographic schemes were first proposed in [5]. Most of these use mathematical problems based on lattices - problems that have a solution but are hard and (currently) computationally difficult to find. Hash-based signatures such as Lamport signatures [108] or Merkle signature schemes [122] require the use of a secure one-way function. For such functions it is assumed that it is easy to compute the result of the function on

every input, but this is hard to invert. Usually a cryptographically secure hash function will be used - but these are prone to breaking as demonstrated upon SHA-1 in [190].

When encryption algorithms are “broken” or overtaken by technology, they are deemed as insecure to use. These two factors highlight the fact that certain encryption algorithms which are based on computational difficulty and unproven assumptions are time-limited with regards to their usefulness in the transmission of secret messages.

In summary, cryptographic algorithms which do not achieve information-theoretic secrecy are based on computationally hard problems. These may be prone to various attacks which progressively build on each other and improve with time. This can decrease the search space of the ever improving computational capabilities of high performance computers, thus allowing for the security properties of such algorithms to be broken - something which cannot happen against secure message transmissions protocols which achieve information-theoretic privacy.

1.2.4 The Importance of Studying Secure Message Transmission Protocols

As outlined earlier, secure message transmission protocols consider and achieve information-theoretic secrecy - similar to the one-time pad. However, unlike the one-time pad, such protocols do not require the exchange of a key (whose size is equivalent to the size of the message) before any secure communication takes place. For certain networks some initial information (such as security parameters and network information) is needed to establish communication between two parties and the assumption that these are known by both parties is made.

Despite this, it is more expensive to communicate multiple messages using secure message transmission protocols than when using the one-time pad². This is because a greater amount of data needs to be transmitted across a network. Additionally, the computational demands of secure message transmission protocols may be greater for both parties (depending on the protocol used) when compared to the computational demands of the one-time pad.

It is thus important to study these protocols and improve on the current knowledge to design more efficient protocols.

In this thesis we mainly focus on almost perfectly secure one-phase message transmission schemes and multi-phase perfectly secure message transmission schemes. Almost perfectly secure one-phase message transmission schemes do not achieve perfect security as with a negligible probability the protocol may “fail”. When this occurs the receiver of the transmission scheme does not accept³ a message in the message space $\mathcal{M} \subseteq \mathbb{F}$ - where \mathbb{F} is a finite field and $|\mathbb{F}| = q$, but instead accepts the empty message - signified by \perp .

The main focus of the research for both types of transmission schemes is to design new, more efficient protocols when compared to existing protocols. We mainly focus on decreasing the computational and communication complexities of such protocols.

We also introduce a new aspect to secure message transmission protocols - considering one of the communicating parties to be a human participant, and present new protocols which are based on experiments with human participants. We also consider a different aspect to per-

²The transmission of multiple messages may be required for example in the implementation of a secure multi-party computation protocol where a party has to send a number of secrets to all other parties participating in the protocol in a secure manner. Such protocols assume the presence of a complete graph - which in fact may be implemented through the use of secure message transmission protocols.

³Here the term “accepts” is used to mean receive as the message of the transmission protocol.

fectly secure message transmission where in addition to the security of a transmission we also consider anonymity of transmission. In such protocols, the identity of the sender of a message should remain anonymous in a scenario where a single receiver receives multiple messages from many senders⁴. The importance of such protocols is described and applications requiring these properties are identified. We also address the importance and feasibility of implementing such protocols in practise upon networks which exist in the real world.

1.3 Previous Work

In this section we review research carried out in message transmission protocols. In the next section we critically assess this work and later specify in detail what was studied and what will be presented in this thesis.

During the Cold War, where the threat of nuclear attacks was a real danger, military networks required great survivability guarantees. This would ensure that command centers were always online and available for any necessary military operation. Similar to such networks, the design and implementation of the ARPANET, from which the Internet evolved, also required survivability and continuity of network operation guarantees [113]. The main focus of any disruption for ARPANET was against losses of large portions of the underlying network [113].

Almost all work in message transmission protocols consider a sender and a receiver connected to each other using node disjoint network paths. Early work on reliable communication [88] considered an adversary which is able to block transmission on a node disjoint path - otherwise known as a *wire*. The specific adversarial model is otherwise known as a fail-stop adversary, which has the ability to cause a network node to become non-responsive. Assuming the adversary is t -bounded, the adversary can control up to t nodes in the network causing them to become non-operational. As the adversary is assumed to know the definition of the protocol and is able to control any network node, he/she can disable the transmission of data on up to t wires connecting the sender and the receiver - effectively causing these wires to be *cut* from the network. Alternatively, one can consider this kind of attack to be equivalent to a network attack where the network paths controlled by the adversary are suffering from congestion or a Denial of Service attack and cannot deliver data effectively, if at all. It is easy to see that with a t -bounded adversary present, to guarantee reliability of message transmission between a sender and a receiver ($t + 1$) wires are needed. In this case, the same data will be sent over all wires. With at most t non-operational wires the data will be successfully delivered to the receiver from at least one wire.

Similarly, the work by Dolev [55] considered a t -bounded Byzantine adversary [109]. Such an adversary, in addition to the capabilities of a fail stop adversary can cause a corrupt node to behave in any manner the adversary chooses - causing it to deviate from the protocol specification and causing errors or alterations to data transmitted across corrupt nodes (and in effect across wires). The necessary and sufficient minimum number of wires n - connecting a sender and a receiver, to allow for reliable communication was shown to be $n > 2t$ [55]. In this case, the sender will transmit the same data over all n wires. The receiver can identify

⁴Similarly in the dual single-sender multiple-receiver scenario where the single sender sends multiple messages, anonymity of receivers who receive a message should be achieved.

and accept the correct data using a majority - data received at least $(t + 1)$ times is identified as the correct data to accept. This is because there are at least $(t + 1)$ honest (non-adversary controlled) wires which will correctly deliver the data transmitted by the sender. If a lower number of wires are used, then the active adversary can simulate the transmission of a different message. For example, if only $2t$ wires are used, the adversary can alter data transmitted on adversary controlled wires so that no single value is received as a majority. In this case the receiver cannot know which is the correct value to accept. Using less than $(2t + 1)$ wires therefore cannot guarantee reliability of message transmission.

Both of the earlier references considered the transmission of messages with reliability being the only required security property. The work of Dolev, Dwork, Waarts and Yung [56] enhanced message transmission protocols by additionally considering the concept of secrecy of the message to be transmitted. The main objective of this work was to consider security “without complexity theoretic assumptions and with perfect correctness”. This means that the protocols would achieve information-theoretic security, that the adversary would be computationally unbounded, that the adversary could not break the privacy of a secret or could not cause the protocol to fail in any way and that the receiver would always accept the same message as that transmitted by the sender. In other words, the work of [56] introduced perfectly secure message transmission (PSMT) protocols. These protocols mainly use the cryptographic concept of secret sharing - such as that of Shamir secret sharing [163], in combination with node disjoint network paths (or wires).

In their work Dolev et al. [56], showed that when data is sent from the sender to the receiver using one-way communication (also known as one-phase protocols), $(3t + 1)$ wires connecting the sender and the receiver are necessary and sufficient to achieve PSMT. If a lower number of wires are used then the receiver may be unable to uniquely decode a single message. This is a direct result obtained from coding theory combined with secret sharing. When considering a t -bounded active adversary, a secret sharing scheme requiring $t + 1$ (unaltered) shares will be required to reconstruct the secret message. Such a secret sharing scheme requires the use of an at most t -degree polynomial. Two t -degree polynomials can share t common points between them. Because of this, a t -bounded adversary can alter t shares and create an at most t -degree polynomial with *at most* $2t$ points defined by the polynomial “created” by the adversary (t shares altered by the adversary and t unaltered shares). To be able to uniquely decode a single message, the receiver would thus have to identify a polynomial with at least $2t + 1$ points - which obviously cannot be a wrong polynomial (as altered polynomials can include at most $2t$ points). Because of this, for one-way PSMT a $(t + 1)$ -out-of- $(3t + 1)$ secret sharing scheme has to be used and $3t + 1$ wires also (with each share transmitted *once* and on one wire only).

The necessary condition of at least $(3t + 1)$ disjoint paths can be rather restricting in applications of secure message transmission protocols. This is because the connectivity requirements are relatively high and may require a rather dense network to provide this number of disjoint paths for the protocol to be executed. The basis of message transmission protocols such as those presented in [68, 107, 167] is to use a lower number of wires - $(2t + 1)$ instead of $(3t + 1)$, to connect a sender and a receiver. This is also the minimum number of wires one can use for reliable communication against a t -bounded active adversary as shown in [55]. With a lower

number of wires, one-phase message transmission protocols could be executed over more sparse networks which cannot provide the required $(3t + 1)$ number of wires required for one-phase PSMT protocols.

Using a lower number of wires, such protocols [68, 107, 167] cannot achieve perfectly secure message transmission - instead they achieve a different form of security which is defined in Section 2.3 and throughout this thesis is referred to as $(\epsilon, \delta, \gamma)$ -security. These are briefly defined here (and with greater detail in Section 2.3). The term ϵ identifies ϵ -privacy if for every two messages $M_0, M_1 \in \mathcal{M}$, with c being a constant, $\mathcal{V}_{ADV}(M, r)$ denoting the view of the adversary when the message transmitted by the sender is M and r denoting the randomness used by the adversary we have:

$$\sum_c |Pr[\mathcal{V}_{ADV}(M_0, r) = c] - Pr[\mathcal{V}_{ADV}(M_1, r) = c]| \leq 2\epsilon$$

The probabilities are taken over the randomness of the honest parties and the sum is over all possible values of the adversary's view. The term δ identifies δ -authenticity if the sender transmits M^S and given that the receiver accepts $M^R \in \{\mathcal{M}\}$ (where $\{\mathcal{M}\}$ denotes the message space) $M^S \neq M^R$ with probability δ . The term γ identifies γ -availability if, with probability at least $1 - \gamma$, the receiver accepts a message $M^R \in \mathcal{M}$ and in all other cases accepts \perp which denotes the empty message. When executing such protocols, the probability of γ , δ and ϵ should be *negligible* - meaning that the probability should be so small that it can be safely ignored.

The main security property such protocols aim to achieve is that of perfect privacy. However, due the adversary's actions and the lower number of wires used, the receiver may decode more than one message - instead of a single one (as earlier described, only through the use of $3t + 1$ wires can a unique message be decoded correctly by the receiver). When this occurs, different protocols terminate in a different manner. For protocols presented in [68] [52, Section 3] the receiver will always accept a message and reliability of message transmission may fail with a given probability bound. In such cases, reliability is said to have failed as the receiver accepts a *different* message to that transmitted by the sender. In protocols presented in [107, 167], when the receiver decodes more than one message, the receiver will not accept a message but will accept \perp and the protocol will terminate. When this occurs the protocol is said to have failed.

In [107] Kurosawa and Suzuki considered almost perfectly secure (1-phase, n -channel) message transmission when $n = 2t + 1$. It was shown that almost perfectly secure message transmission can be achieved provided we are willing to accept a small (negligible) probability that the protocol will fail. This occurs when the receiver decodes more than one message and accepts \perp . The probability with which this occurs is identified by the variable γ of the security definition used in this thesis. A lower bound of $\Theta(n)$ communication complexity for this type of transmission was proven and an exponential time protocol achieving this bound was presented.

As exponential time protocols are impractical for large values of t , other research has focused on designing polynomial time almost perfectly secure (1-phase, n -channel) message transmission protocols. In [167] Srinathan et al. considered the same type of security - although different terms were used to refer to very similar security properties. Their work presented a polynomial time protocol achieving almost perfectly secure message transmission of $O(n)$

messages with a communication complexity of $O(n^2)$. Because of the complexity in the number of messages transmitted, the protocol achieved $O(n)$ transmission rate⁵ - which is the optimal transmission rate that any transmission protocol can achieve as proven in [169].

When two-way communication between the sender and the receiver is possible, Dolev et al. [56] proved that $n = 2t + 1$ wires connecting a sender and a receiver is a necessary and sufficient condition to achieve PSMT. Dolev et al. [56] also presented a three-phase protocol achieving PSMT - with a communication complexity of $O(n^5)$. It was shown that two phases are necessary and sufficient to achieve PSMT and presented a two-phase PSMT protocol. Nonetheless, the protocol was inefficient as it required exponential communication complexity - which is impractical for large values of n . Since then, work has sought to improve such protocols [4, 62, 105, 168, 170]. The main emphasis is on protocols with two phases which is the optimal number of phases a protocol can achieve PSMT when using $n < 3t + 1$ wires. These protocols initiate with the receiver sending information to the sender (first phase) and end after the sender replies back with data (in the second phase) containing the encrypted secret which the receiver can decode.

Srinathan et al. [169] proved that any two-phase reliable message transmission protocol, and hence any secure protocol, over $n \geq 2t + 1$ wires out of which at most t are faulty is required to transmit at least $b = \binom{nl}{n-2t}$ bits for the transmission of an l -bit message. A protocol claiming to achieve this lower bound was presented, but an error in their protocol was presented in [4].

Agarwal et al. [4] also presented a two-phase protocol able to achieve the optimal transmission rate of $O(n)$. This was achieved with exponential communication complexity which meant that the computational requirements of the two communicating parties were also exponential. This was improved somewhat in Fitzi et al. [62] where the presented work was able to achieve the optimal transmission rate of $O(n)$ with lower $O(l)$ communication complexity achieved - where l is the length of the message sent. Despite this, a greater number of wires $n \geq (2 + \epsilon)t$ (where $\epsilon > 0$ is a fixed but arbitrarily small constant) and not the optimal $(2t + 1)$.

Both the above results were improved in [105] where a polynomial time two-phase PSMT protocol able to achieve $O(n)$ transmission rate was presented. The communication complexity for the protocol was $O(n^3)$ and $O(n^2)$ messages were transmitted by the sender to the receiver.

Secure message transmission schemes have also been studied in other contexts. Considering a general adversary structure [96] is such an example. In [104] this adversarial model was considered for networks with bi-directional channels and proved necessary and sufficient conditions for PSMT for this network setting. In [53] the same adversary model was considered but with a different network model where communication between sender and receiver and vice versa was possible, but through the use of one-way channels only.

Other contexts in which PSMT protocols have been studied include asynchronous networks [161], mobile adversaries [143] and mixed adversaries [56, 145] amongst others.

Solutions for asynchronous networks have been studied by other researchers. The work of [35, 36] proved that in an asynchronous network, if all the n wires are directed from sender

⁵Transmission rate is the ratio of the protocol communication complexity divided by the complexity of the number of secret messages transmitted in a transmission protocol.

S to receiver R , then any PSMT protocol tolerating a static t -threshold adversary is possible iff $n > 3t$. Surprisingly, the same work proved that even if the n wires are bi-directional, then any PSMT protocol in an asynchronous network tolerating a static t -threshold adversary is possible iff $n > 3t$ (for synchronous networks bi-directional PSMT requires $n > 2t$)⁶.

Other work has considered secure message transmission protocols but with various assumptions made. In [18, 19] the work focused on decreasing the number of disjoint network paths required for the communication between the sender and receiver to take place. It was shown that lower connectivity requirements are required if some pairs of processors share authentication keys between them. The work of [18] showed that for reliable communication amongst any sender and any receiver against a coalition of at most t faulty processors, the network of channels defines a natural “communication graph”, with an edge between two vertices for every channel between two processors. The pairs of parties sharing authentication keys define a natural “authentication graph”, with an edge between two vertices for every shared key. It was shown that all-pairs reliable communication is possible if and only if the communication graph is $(t + 1)$ -connected and the union of the two graphs is $(2t + 1)$ -connected. In this case, disjoint network paths are replaced by authentication keys and their respective edges in the “authentication graph”.

In [73, 165] the problem of secure message transmission with public discussion (SMT-PD) - which was introduced in [74], was considered. Similar to PSMT protocols, such protocols also consider a t -threshold bounded adversary which can corrupt t of the n ($t < n$) channels connecting a sender and receiver. Additionally, SMT-PD protocols assume the presence of a reliable, but public channel, that is a broadcast channel the adversary can read but cannot alter. This public channel is implemented using the work of [22, 46, 187] where it was shown how broadcast functionality is feasible in partially connected settings for a subset of the nodes in the network. However, the implementation of the public channel in point-to-point networks is costly and highly non-trivial in terms of rounds of computation and communication [73]. The use of such a public channel allows for secure message transmission to occur even when the number of wires connecting the sender and receiver is less than $2t$ - something which is not possible in PSMT protocols. SMT-PD protocols are important as they allow for secure multi-party computation to be achieved on sparse networks where the number of node disjoint network paths connecting communicating parties is not enough to allow the use of PSMT protocols.

Further to the described work, different network models have been proposed in the theory of secure message transmission protocols. These include the directional model of communication - where communication can occur from the sender to the receiver, and the corresponding non-directional model of communication - where communication from sender to receiver and from receiver to sender is possible. Both of these models were first proposed in [56]. A third model was proposed in [52] where communication from sender to receiver and from receiver to sender was possible, but wires which allowed for communication from sender to receiver were different to the wires which allowed for communication from receiver to sender.

It is important for the research community to research all network models (in this thesis

⁶It should be noted that asynchronous networks will not be studied in this thesis as for all protocols to be presented synchronous networks will be considered.

we study two of these three models) as different networks which exist in practise can fall in any one of the three models. Network nodes which are part of a wired network will generally fall in the non-directional model. On the other hand, nodes of actuator networks may follow the directed network model. In such networks the sink of the network carries out most of the communication and actuator nodes forward or receive information. Satellite networks may also fall under this category, where the main form of communication occurs one way. This communication may be either from a control center on earth sending instructions to a satellite far out in space or from a satellite sending collected data to a control center. Additionally, the network model considered in [52] is equally important. As shown in the work of [101] which began with the attempt to implement geographic routing [99], idealized assumptions about radios and their resulting connectivity graphs were shown to be untrue. This in addition to connectivity problems highlighted in [102], show that for ad-hoc networks which use radio communication, the presumed network graph and connectivity is not necessarily the one that will be available to the network. For such networks, the implemented network may have a topology similar to the network model presented in [52]. On the contrary, the network connectivity graph may not allow for node-disjoint paths and in this worst case scenario secure message transmission protocols cannot be used.

1.4 Critical Review

In this section we critically assess previous work and specify in detail what will be studied in this thesis.

We first consider work which assumes certain conditions in the environment of message transmission protocols. The work of [18, 19] assumes the presence of authentication keys in a network and the work on secure message transmission with public discussion [73, 165] assumes the presence of a reliable channel. These may be impractical assumptions to make. Secure message transmission schemes are generally considered in the context of abstract theoretical networks. In this context, the presence of authentication keys requires prior placement and distribution of these keys before such a protocol is executed. Furthermore, the assumption of a reliable channel in the model of secure message transmission with public discussion is also an impractical assumption to make. As stated earlier, against a Byzantine adversary reliability of message transmission cannot be achieved when the number of disjoint paths n connecting a sender and a receiver does not satisfy $n \geq 2t + 1$. This raises the question of how one will implement the required reliable channel of public discussion protocols *efficiently* when such work states that the number of disjoint paths can be less than $(2t + 1)$. Implementing the public channel using the work of [22, 46, 187] when $n < 2t + 1$ is costly and highly non-trivial as [73] asserts. Such assumptions may be impractical to implement and such work *will not* be considered further in this thesis.

It seems that most recent research carried out in message transmission schemes has focused on increasing the number of messages sent in a protocol execution. This is done so that the transmission rate of a protocol reaches its optimal limit of $O(n)$. This ignores the cases when secure message transmission protocols will mainly be used by a sender and a receiver, to allow them to initiate communication in networks where no infrastructure to allow for key

exchange is available. In such cases, the only message that needs to be securely transmitted is the *single* value of an encryption key. Once this has occurred the two communicating parties can carry on with secure communication using some form of encryption algorithm. This will allow for the latter communication to occur in a more efficient manner - with regards to communication complexity and number of phases required. This is useful for applications where the planned age of secrets is short enough that it is secure to use current symmetric key encryption algorithms. Secure message transmission protocols are thus used for key exchange purposes when infrastructure which could allow for such a process is unavailable (but details of network topology and security parameter t may be assumed knowledge network nodes possess).

Alternatively, secure message transmission protocols may be used to implement the direct, reliable and private channel of an assumed complete graph for various protocols - for example MPC. For some of these protocols, it may be the case that each participant sends a single (possibly different) message to all other participants. Using protocols which transmit multiple messages is thus not required and it is best to use more efficient protocols which transmit only a single message.

Work which focuses on increasing the number of secrets transmitted is not that useful for the given scenarios. On the contrary, all that needs to be transmitted securely is a *single message* and it is important to use the most efficient algorithms with regards to communication and computational complexities. The main aim of the thesis will be to present new protocols which achieve this.

1.5 Context and Overview of Thesis Focus

In this thesis we study various forms of secure message transmission protocols which transmit a *single* message from a sender to the receiver⁷ considering synchronous networks and static t -threshold bounded and computationally unlimited adversaries.

The main focus of the work is to design new more efficient protocols - improving earlier work in the literature. The efficiency improvements sought are those for the communication and computational complexities of such protocols. This is important as secure message transmission protocols are fundamental cryptographic protocols (they are used by other cryptographic protocols to simulate a complete graph) and are also used in other research areas (for key exchange purposes, perfect privacy of exchanged secrets amongst others). Simplifying them as much as possible is important so that protocols which use secure transmission protocols can in turn become more efficient.

Our work focuses on the synchronous model of communication *only*. It is important to study synchronous networks as understanding how to solve a problem in the synchronous model can be a useful intermediate step towards understanding how to solve the same problem in other models. For synchronous networks, a maximum bound on the amount of time in which data is delivered is assumed. This of course cannot be guaranteed using an asynchronous network. Our results thus cannot be used in practise on asynchronous networks (such as the Internet) because when considering a t -bounded threshold adversary one cannot know or distinguish the

⁷This single message may be a secret value in the execution of a secure multi-party computation protocol or it may be the value of an encryption key for later communication which will occur through the use of an encryption algorithm. Its exact purpose depends on the application in which the message transmission protocol is used.

reason as to why a message is not delivered in time - even if some kind of synchronicity was to be enforced through timeouts. One could not know whether something was undelivered as a result of the adversary blocking the transmission of data or whether the network was slow in its delivery. Because of this, for asynchronous networks one will have to consider any possible delays in network delivery as well as the adversary's effect on the protocol⁸.

In the thesis we only consider the threat model of a static t -bounded computationally unlimited static adversary. As the adversary is computationally unlimited we cannot use cryptographic schemes based on computational difficulty or unproven assumptions in the construction of any protocols. In general an active adversary is considered but for some protocols a passive adversary is considered and in such cases this will be explicitly stated. The static nature of the adversary forces the adversary to choose the nodes to corrupt before the execution of the secure transmission protocol initiates. The t -bounded adversary threat model assumes that the adversary can take control of at most t nodes of a network. This is comparable to the adversary being comprised of at most t parties which have to be physically present at the location of a network node to take control of it.

The alternative adversary threat model which assumes the capability of corrupting an operating system and thus allowing the adversary to corrupt all network nodes running the *same* operating system is based on the general adversary structure model as described by [96]. More specifically, the colour adversary problem as studied in [54] considered adversaries which were able to carry out replicated attacks on similar systems (such as systems using the same operating system). This threat model is *not* considered in this thesis.

1.6 Overview and Organization of Thesis

The next chapter provides background material necessary in the understanding of work to be presented in later chapters. Despite being brief - as it assumes the reader possesses some general knowledge of message transmission protocols, readers are referred to the relevant more detailed appendix sections.

In Chapter 3, we consider almost perfectly secure one-phase message transmission protocols. Two new protocols are presented - both of which improve on previous protocols proposed by other authors. We present a polynomial time protocol which improves on the work of [167] and we also present an exponential time protocol which improves on the work of [107]. The following describes how the new polynomial time protocol compares to previous work.

Protocol	Computational Complexity	Communication Complexity
Protocol proposed in [167]	$O(n^3)$	$O(n^2)$
Protocol 1 (Section 3.2)	$O(n^3)$	$O(n^2)$
<p><u>Note:</u> Protocol proposed in [167] transmits more field elements (in number) than Protocol 1 (neglecting terms that are subquadratic the two protocols transmit $10t^2$ vs $6t^2$ field elements respectively) and requires more computation.</p>		

⁸Furthermore, as shown in [35, 36] asynchronous PSMT requires $n \geq 3t + 1$ wires where in the protocols we present consider $n = 2t + 1$ wires.

The following below describes how the new exponential time protocol compares to previous work.

Protocol	Computational Complexity	Communication Complexity
Protocol proposed in [107]	$O\left(\binom{2t+1}{t+1}n^2 \log(q)\right)$	$O(n)$
Protocol 2 (Section 3.3)	$O\left(\binom{2t+1}{t+1}n^2\right)$	$O(n)$
<u>Note:</u> Protocol proposed in [107] requires the use of a planar difference set. Protocol 2 can use any finite field.		

One might ask if exponential time protocols in general are necessary and practically relevant considering polynomial time protocols exist. This depends on a number of factors as we have to take into account that the two protocols also differ in their communication complexity. Given their different complexities for communication and computation, for small values of t (such as $t = 1$, $t = 2$ or even $t = 3$) it may be preferable to use the exponential time protocol, as the amount of computation that needs to be carried out does not amount to much. In such cases, for small amounts of t one may find that using an exponential time protocol uses less energy over the entire network (when considering energy required to transmit, forward and receive data as well as the required computation) than the polynomial time protocol.

Furthermore, the polynomial time protocol is theoretically not the most efficient protocol which can be designed. Such a protocol would be of polynomial time and will have a communication complexity of $O(n)$ - similar to the exponential time protocol we propose. It is thus important to study such protocols and try and improve on previous work as much as possible so that we can progressively work together towards designing the best possible solutions. Each new protocol which is introduced in the literature may help another researcher in improving on already published work.

In Chapter 4, multiphase protocols are considered. A new efficient two-phase protocol is primarily presented which achieves the perfectly secure message transmission of a single message between a sender and a single receiver with lower communication complexity than previous work. The new protocol achieves two-phase PSMT with $O(n^2)$ communication complexity (with $n = 2t + 1$) - improving on the $O(n^3)$ communication complexity of previous two-phase PSMT protocols. The way this protocol compares to previous work is summarized by the following.

Protocol	Number of Messages Transmitted	Communication Complexity
Protocol proposed in [162]	1	$O(n^3)$
Protocol proposed in [105]	n^2	$O(n^3)$
Protocol 3 (Section 4.1)	1	$O(n^2)$

This protocol is later extended to a three-phase protocol in the context of a different single sender multi-receiver network model. In this network model we consider the sender to be connected to the receivers through the use of broadcast channels, with each receiver accessing a number of these broadcast channels - which deliver the *same* information to all receivers who have access to the same broadcast channel. This model is similar to networks which can be

found in practise and for the first time is considered in the context of perfectly secure message transmission protocols. The work shows how the use of broadcast channels can allow for great savings for the single sender in the secure transmission of a message to multiple receivers when compared to multiple executions of single receiver protocols. The importance and relevance of such a protocol is also discussed.

Chapter 5 introduces a new research area in secure message transmission protocols - that of perfectly secure message transmission with a human. In such protocols, the receiver of the communication (which can be altered to the sender of the communication or both parties of a communication) is assumed to be a human participant. Because of this, the receiving party is at a computational disadvantage as it is assumed that a human does not have a trusted computational device available to them. The difficulty of this problem is that a human cannot carry out complex mathematical based algorithms to achieve a solution. Such protocols are thus limited to use only trivial maths such as addition over an abelian group and more specifically addition mod10 - which is similar to mathematical operations and addition of one digit numbers that humans are more familiar with. In addition to the requirement that such solutions should be easy for a human to use, they should also be computationally efficient to be executed by a human in a timely manner. We present protocols which achieve these requirements and discuss possible applications of such protocols. We also present the results of experiments carried out with human participants which show the correctness with which the participants used the protocols.

Chapter 6 studies a different aspect to perfectly secure message transmission when we consider anonymous secure message transmission protocols which achieve information-theoretic privacy. In such protocols, multiple senders each send a message to a single receiver and the identity of the sender of a message should remain anonymous to the receiver. The anonymity and security properties should be achieved despite the presence of an adversary. Two new protocols are presented when considering a passive and active adversary. The dual of this setting is when a single sender sends messages to multiple receivers for which a solution can also be designed using our protocols (with minor modifications) which is also described.

Chapter 7 considers message transmission protocols in general with a more practical based viewpoint. We address the importance of secure message transmission protocols - highlighting how their use could improve secure and reliable communication on networks such as the Internet. This is of great importance as it can allow us to defend against future network threats which can disrupt the reliable operation of networks we have become accustomed to and on whose correct and reliable operation we depend on. We also explore the feasibility of implementing such protocols upon various networks which exist in the real world.

This thesis concludes in Chapter 8 where open problems that could be carried out as future work are identified.

The thesis also includes a number of Appendix chapters which in general provide greater detail to background material and include the experimental tests given to human participants.

Chapter 2

Background

This chapter provides background material necessary for the understanding of work to be presented in later chapters. The content of this chapter assumes the reader possesses some general knowledge of secure message transmission protocols. A general overview of the background is therefore given. For further details the reader is referred to the appropriate Appendix sections.

2.1 Environment of Message Transmission Protocols

The main players of a secure message transmission protocol are a sender and a set of receivers¹ which are connected via an underlying synchronous network. The main objective for the sender is to select a message M^S drawn from a message space $\mathcal{M} \subseteq \mathbb{F}$ - where \mathbb{F} is a finite field and $|\mathbb{F}| = q$. The secret M^S is selected with respect to a certain probability distribution and should be transmitted securely to the set of receivers. Communication between the sender and the receivers occurs in phases - where a phase is a transmission from the sender to the receivers or vice-versa. At the end of the protocol, all receivers output a message M^R .

In the network environment, the presence of another player - known as an active adversary, is considered. Such an adversary is capable of corrupting up to $t \ll q$ nodes in the underlying network. The adversary is assumed to know the complete protocol specification, message space \mathcal{M} and the complete structure of the network graph. The adversary can take full control of any compromised network node and may aim to break the security of the transmission protocol. Further details on the adversary presence is given in Section 2.4 and in Section 2.3 the security model considered in the thesis is presented.

Throughout the thesis we assume that knowledge about properties of the communication network (such as network graph, value of t) are commonly known by both sender and the set of receivers of a communication.

2.2 Network Models

Two different network models will be considered in the thesis. The first considers a single sender and a single receiver scenario, where network edges are point to point channels (where two endpoints are connected together with a permanent link). The second network model is *new* in the field of message transmission protocols and considers a single sender multi-recipient scenario with the network containing both point to point and broadcast channels (upon which a message is delivered to all recipients who have access to the broadcast channel) which are

¹In the single receiver model the set of receivers consists of a single party only.

otherwise known as “hyperedges”². Both models are described in greater detail later in this section. We first describe characteristics of the network environment which are common to both models.

For both models, we consider a synchronous network for which there is a known upper bound on message transmission delays. The time period between the instant at which a message is sent by a process and the time at which the message is received by the destination process is thus less than this bound.

Secure message transmission protocols consider the presence of disjoint network paths in the underlying network which connect the sender and the set of receivers of the communication. This disjointness can be edge disjointness - where disjoint paths do not share common network edges between them. When considering an adversary presence with the ability to corrupt up to t network nodes, the main disjointness that is considered is that of node disjointness (also known as vertex disjointness). Node disjoint paths - otherwise known as *wires*, do not share any network nodes between them (besides the sender and the receiver). It is easy to see that node disjointness achieves edge disjointness also. As the adversary can corrupt up to t network nodes, the adversary can control up to t wires which connect the sender and the set of receivers.

Given a directed network graph $G = (V, E)$ and two vertices denoting the sender S and receiver R , node disjoint paths connecting the two parties can be found using a maximum-flow min-cut algorithm [75, 79, 112, 174]. Such algorithms may find a greater number of node disjoint paths (or wires) than the number which will be required in a transmission protocol. Throughout the thesis, n will denote the number of wires used in a communication network to connect the sender to the set of receivers in a transmission protocol.

It should be pointed out that we do not consider the length of these wires. The network nodes and the number of nodes which are used to compose the wires are abstracted away. Our only concern is that the required number of wires are available for the execution of a transmission protocol. Additionally, any network nodes which are not used in the construction of wires, are also abstracted away and do not affect the execution of transmission protocols in any way. As seen in Figure 2.1, the final network topology for a transmission protocol does not necessarily use all network nodes in the underlying network connecting a sender and a receiver. Figure 2.1 is an example of a network topology with all network nodes of a transmission protocol. The sender and the receiver are denoted with the respective S and R which appear in a circle. Node disjoint paths (wires) which connect S and R are represented by lines which join S , R and black coloured circles which represent network nodes used in the composition of wires. Grey coloured circles represent other network nodes (which do not compose wires and are not S or R). These nodes do not take part in transmission protocols and are abstracted away.

In the next two subsections, we describe in more detail the network models which will be considered.

2.2.1 Point-to-Point Networks

In this model, the communication network is modeled as a directed graph $G = G(V, E)$ whose nodes are the parties and edges are point-to-point reliable and private communication channels.

²These were formally introduced in [69] and were used to connect a sender to a subset of receivers.

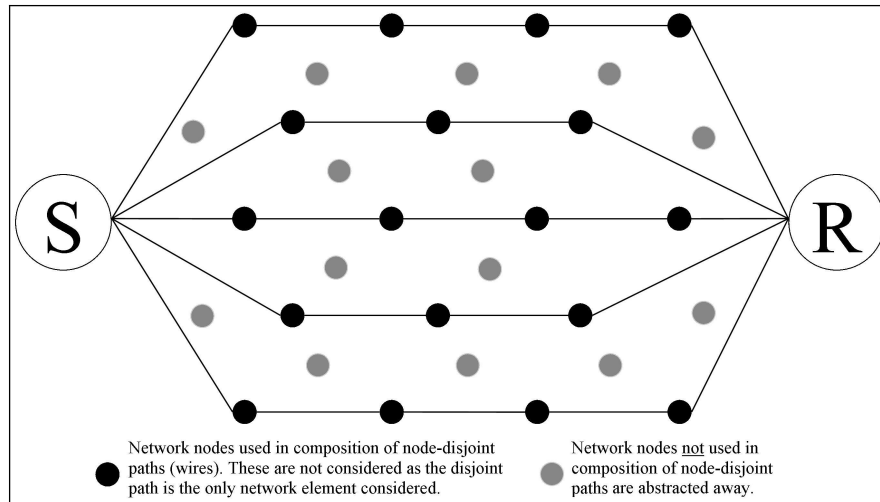


Figure 2.1: Network topology with all network nodes of a transmission protocol.

Wires connecting the sender and a receiver of a communication can be composed of a number of network nodes of the underlying network. As stated earlier, these are abstracted away and the communication channel as a whole is modelled as one single wire.

Two variations of this network model will be considered in this thesis - the non-directional network model and one variation of the directional network model. These vary relative to the directionality of the edges.

In the non-directional model, the communication channels which connect the sender and the receiver allow for communication to occur both from sender to receiver and from receiver to sender. An example of the non-directional model can be seen in Figure 2.2.



Figure 2.2: Example of a non-directional communication network.

The non-directional model can be considered as a general model of point-to-point networks. This is because of the non-directional communication channels which allow for bi-directional data transmission. The directional network model on the other hand can be considered as a special case of the general non-directional model - allowing for data transmission to occur *in one direction only* (from sender to receiver *or* from sender to receiver) for each communication channel.

Two variations of the directional network model have so far been considered in literature.

The term “directional model” will refer to a directional network model with communication channels which allow for communication to occur *only* from sender to receiver. This can be seen from the direction of the links in the directional model example given in Figure 2.3.

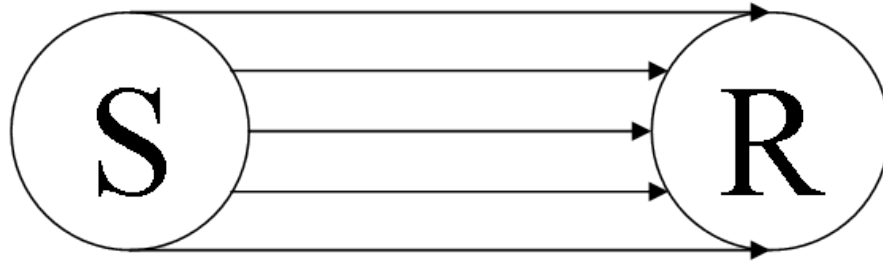


Figure 2.3: Example of a directional communication network.

In [52] the authors considered another special case of the non-directional model. In their model - similar to the directional model, Desmedt and Wang allowed for communication from sender to receiver and vice versa to occur. What was different about this model - compared to the non-directional model, was that the number of wires which allowed for communication from sender to receiver was different to the number of wires which allowed for communication from receiver to sender. Furthermore, the network model allowed for these two sets of wires to be disjoint between them. In this case, the network will be composed of directed channels only. This network model is not considered in this thesis.

2.2.1.1 Simulating Broadcast in Point-to-Point Networks

Broadcast will be used in all multiphase protocols presented in the thesis. To simulate broadcast in point-to-point networks, when a party (either sender or receiver) broadcasts information, the same information will be sent over all wires which connect the party to the receiving party of the communication. The number of wires connecting the two parties is denoted by n and throughout the thesis $n \geq 2t + 1$. As the adversary is able to corrupt at most t of these n wires, the receiving party correctly identifies and receives the information sent by the sending party using a majority vote (information received at least $t + 1$ times).

To identify the majority amongst $n = 2t + 1$ values received from n wires requires $n \log n$ computational complexity (using binary search).

2.2.2 Network with Broadcast End Channels

We now describe the second network model we consider which is a *new* network model in the research area of message transmission protocols. In this model, the communication network is modelled as a graph $G = G(V, E, B)$ whose nodes in V are the parties and edges in E are point-to-point reliable and private communication channels. Set B is a set of hyperedges which are reliable and private *broadcast* communication channels - which throughout the thesis will be referred to as *LAN multicast* channels. Such channels exist in practise - such as wired ethernet [123] and fibre optic broadcast [45]. It should be noted, that when referring to LAN multicast channels, these are not only confined to broadcast mediums of communication (i.e. radio or ethernet) but include any form of communication which could reliably and privately deliver *the same* information to multiple receivers³.

We identify four different types of nodes in V - a sender $S \in V$, a set of multiple receivers $R \subsetneq V \setminus \{S\}$, and a set of forwarding nodes $F \subseteq V \setminus (\{S\} \cup R)$. Forwarding nodes connect

³An example of such communication is through the communication of messages in an online closed community - such as forums where only registered users can see and send messages.

to the receivers with LAN multicast channels in B . The rest of the nodes (if any exist) are the intermediate nodes between the sender S and the forwarding nodes in F . These nodes either comprise node disjoint paths between S and nodes in F or are not used at all in the execution of a protocol.

Node disjoint paths between S and nodes in F are referred to as *wires* and will be termed so from now on. For this network model, we assume that the adversary can corrupt forwarding nodes only (or any network node which is present in the wire which connects the sender to each forwarding node) and can take control of at most t of these in a network. Because of this, the adversary has the capability to control t wires.

Communication channels between receivers in R and forwarding nodes in F are assumed to be LAN multicast channels in B . Information placed by forwarding nodes on such a channel, is thus transmitted and received with perfect authenticity (see the security definition defined in the next section) by receivers who have access to that LAN multicast channel.

As earlier defined, n denotes the number of wires in the communication network. This in turn implies that there are n number of nodes in F and n LAN multicast channels in the communication network.

Without loss of generality, we assume that receivers are connected to $n' \leq n$ number of LAN multicast channels. This also indicates the number of forwarding nodes each receiver is connected to.

Figure 2.4 is an example of the communication network we consider.

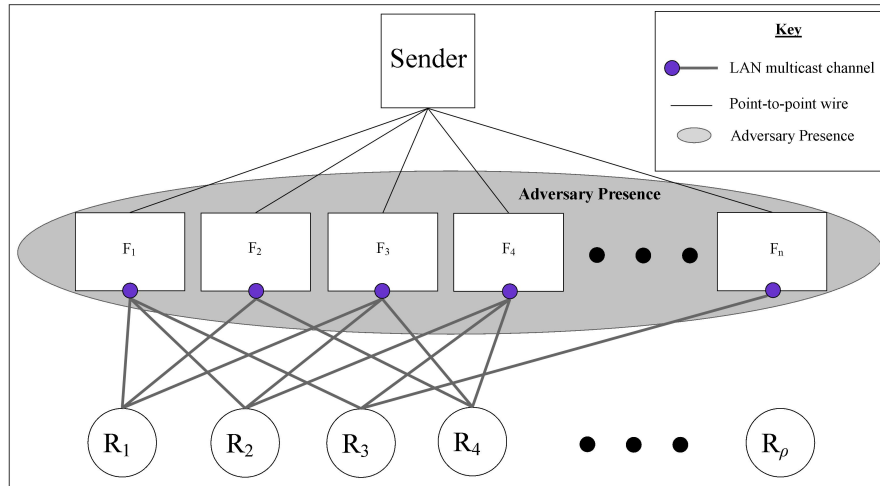


Figure 2.4: Example of network with broadcast end channels.

Some networks implemented in practise have a network topology with similar characteristics to the topology of the network model we consider. This is discussed further in Section 4.2. In Section 4.2.2.1 examples of how receivers can access more than one broadcast channel are provided - in the context of practical networks.

2.3 Security Model

In this section we provide details of the security definition that will be considered in the thesis. This will be the same as the security definition presented in [50]. For other security defini-

tions and how these compare to the one that will be used, the interested reader is referred to Appendix A.

For any protocol execution, we denote with \mathcal{V}_{ADV} the view of the adversary for the execution of the protocol. This includes the behavior of the corrupt nodes during the execution of the protocol, the initial state of the adversary and any randomness (coin flips) the protocol uses during its execution. We denote with $\mathcal{V}_{ADV}(M, r)$ the view of the adversary when the message transmitted by the sender is M , and r denotes the randomness used by the adversary. We also use \perp to denote the empty message.

Privacy: A message transmission protocol is ϵ -private if for every two messages $M_0, M_1 \in \mathcal{M}$, with c being a constant and every r we have:

$$\sum_c |Pr[\mathcal{V}_{ADV}(M_0, r) = c] - Pr[\mathcal{V}_{ADV}(M_1, r) = c]| \leq 2\epsilon$$

The probabilities are taken over the randomness of the honest parties and the sum is over all possible values of the adversary's view.

Perfect Privacy is achieved when $\epsilon = 0$ and thus despite \mathcal{V}_{ADV} the adversary gains no extra knowledge about the secret message. The adversary knowledge of M is therefore equivalent to when the adversary does not observe the execution of the protocol and instead guesses the value of M from the message space \mathcal{M} . The adversary - which for perfect privacy is assumed to have unlimited computing power, should thus not be able to recover any probabilistic information on M besides what the adversary already knew a priori to the execution of the protocol.

For the following definitions of authenticity and availability, the receiver *always* accepts a message $M^R \in \{\mathcal{M}, \perp\}$ as the message of a transmission protocol (the receiver thus never rejects a value).

Authenticity: Assuming the sender transmits M^S and the receiver accepts $M^R \in \{\mathcal{M}, \perp\}$ as the message of the transmission, δ -authenticity is achieved by the following conditional probability:

$$\delta \geq P(M^S \neq M^R | Receiver Accepts M^R \in \{\mathcal{M}\}).$$

Perfect Authenticity is achieved when given that the receiver accepts a message $M^R \in \{\mathcal{M}, \perp\}$ as the message of the transmission, $M^S = M^R$ or $M^R = \perp$ with probability 1. This means that the message accepted by the receiver is the same as that sent by the sender or is \perp . The value of δ is thus equal to 0.

$$\text{Perfect Authenticity: } \delta = 0 = P(M^S \neq M^R | Receiver Accepts M^R \in \{\mathcal{M}, \perp\}).$$

Authenticity is thus a measure of the probability with which a receiver accepts the same (in value) message to the message sent out by the sender. It should be noted, that the receiver *always* accepts a message M^R where either $M^R \in \mathcal{M}$ or $M^R = \perp$. Based on the above definitions, perfect authenticity is achieved when the receiver accepts $M^R = \perp$ or when $M^R = M^S$.

Availability: Let $\gamma \leq 1$. A message transmission protocol achieves γ -availability if, with probability at least $1 - \gamma$, the receiver accepts a message $M^R \in \mathcal{M}$ as the message of the transmission. In all other cases (with probability $\leq \gamma$) the receiver accepts \perp .

Perfect Availability is achieved when $\gamma = 0$, that is for all executions of a protocol the receiver accepts a message $M^R \in \mathcal{M}$ (and *never* accepts \perp).

A message transmission protocol is $(\epsilon, \delta, \gamma)$ -secure if it achieves γ -availability, δ -authenticity and ϵ -privacy.

Perfectly secure message transmission - or PSMT for short, refers to protocols which achieve perfect availability (where $\gamma = 0$), perfect authenticity (where $\delta = 0$) and perfect privacy (where $\epsilon = 0$). PSMT protocols therefore achieve $(0, 0, 0)$ -security.

Almost perfectly secure message transmission, refers to protocols which achieve perfect authenticity (where $\delta = 0$) and perfect privacy (where $\epsilon = 0$) but the availability of these protocols may fail with a bounded probability. Almost perfectly secure message transmission protocols therefore achieve $(0, 0, \gamma)$ -security.

In cryptography, a security scheme is considered secure if the probability of security failure is negligible. In the context of almost perfectly secure message transmission protocols, γ should be made negligible in the security parameters. From [32], a function μ is called negligible if it decreases faster than any inverse polynomial. Formally, it is negligible if, for any positive polynomial $poly(\cdot)$, there exists an N_p such that:

$$\forall n \geq N_p : \mu(n) < \frac{1}{poly(n)}$$

In this thesis we aim to construct protocols with negligible probability of failure (preferably the function which will define the probability of failure will be a negligible function) or zero probability of failure - for all three security properties described.

2.4 Adversary Presence

Different types of adversaries have been considered in the context of secure message transmission protocols. These vary from a threshold adversary, a generalized adversary structure [65, 96], a mobile adversary [90] amongst others. In this section we give a general outline of the adversary presence which will be considered throughout the thesis. For a more detailed description of the various adversaries considered in message transmission protocols, the interested reader is referred to Appendix B.

The adversary is assumed to be present in the *underlying network* which connects the sender and the receiver of a message transmission protocol. Based on the network example shown in Figure 2.1, the adversary can thus corrupt any network node which comprise the wires which connect the sender and the receiver(s). It should be noted that throughout the thesis the adversary is not able to corrupt any of the communicating parties (sender or receiver(s)) of a transmission protocol. The adversary is assumed to know the complete protocol specification, message space \mathcal{M} and the complete structure of the network graph. Because of this knowledge, the adversary is assumed to know the nodes which will compose the wires to be used in an execution of a transmission protocol.

Throughout the thesis, a static t -threshold bounded, computationally unlimited active adversary will be considered. A static adversary chooses the nodes to corrupt before the execution of the secure transmission protocol initiates. Despite considering a static adversary, protocols

presented in the thesis are also secure against an adaptive adversary - which can choose the nodes to corrupt anytime after the execution of a protocol. This is an observable result because once both static and adaptive adversaries choose the nodes to corrupt, these cannot change. Furthermore, the protocols have been designed to take into account the fact that the adversary can corrupt any wire connecting the sender and the receiver. With this in mind, all subsets of t corrupt wires will provide the adversary with the same amount of information of the protocol view. Because of this, the adaptive nature of an adversary does not present an advantage over a static adversary. Additionally as for all multiphase protocols presented in the thesis, broadcast is used as the main method of communication beyond the first phase and as broadcast achieves perfect authenticity of messages transmitted, an adaptive adversary cannot affect the correctness of the protocol if the nodes to corrupt are chosen after the first phase of communication.

The active adversary considered possesses the capabilities of a passive adversary and is able to view any data transmitted across corrupt nodes. Due to its active nature, it can also fully control corrupt nodes - causing them to deviate from the protocol specification in any way the adversary chooses. As the adversary is computationally unlimited, cryptographic schemes based on unproven assumptions or computational hardness such as those presented in [136, 137, 157] cannot be used. As the adversary is t -threshold bounded and can corrupt any network node in the underlying network, the adversary has the ability to control up to t node-disjoint network paths (or wires) connecting a sender and a receiver. Throughout the thesis, the term “honest wires” refers to wires which are *not* controlled by the adversary.

The adversary present during the execution of a secure message transmission protocol aims to break the security of the protocol. The adversary may do this by attempting to learn the value of the secret message - thus breaking the secrecy of the protocol. The adversary could also try to attack the authenticity of the protocol. This would be achieved by carrying out appropriate changes to make the receiver of the protocol accept a message different to that sent by the sender of the communication ($M^R \neq M^S$). Alternatively, the adversary may choose to attack the availability of the protocol - causing it to fail so that the receiver does not accept a message but instead accepts \perp . The adversary could of course aim to break the security of a protocol by attempting all of the above attacks against a protocol. The specific attack to be carried out depends on the adversary and its choices.

2.5 Efficiency of Protocols

There are a number of factors which decide the efficiency of message transmission protocols and a brief outline of these is given in this section.

Connectivity requirements of transmission protocols is one of the most important factors which decides the efficiency of protocols. Whether we are considering a threshold or general adversary structure, a protocol with lower connectivity requirements than one with greater connectivity requirements is considered more efficient. This is because a lower number of resources (lower number of wires which possibly could result in lower number of network nodes) are required to achieve security requirements. Additionally, an algorithm with lower connectivity requirements can possibly be used over a wider range of networks⁴.

⁴A protocol with greater connectivity requirements than another could possibly not be used in sparse networks

Communication complexity is another important factor for secure message transmission protocols. Communication complexity measures the complexity of the number of field elements communicated by the sender and receiver(s) in a protocol. This complexity is usually expressed as a function of n - which denotes the number of wires used in a communication network to connect the communicating parties. This will also be used throughout the thesis (sometimes this may be expressed in terms of t - where n is a function of t). It should be noted that for multi-phase protocols (protocols of more than one-phase but with a constant number of phases), the communication complexity of different phases may vary. In such cases, the greatest communication complexity of any phase is considered as the communication complexity of the protocol. Despite this, even decreasing the communication complexity of a phase but not of the protocol itself can be considered a slight improvement. Throughout the thesis, sender communication complexity will refer to the communication complexity of data a sender *transmits*. Similarly, receiver communication complexity will refer to the communication complexity of the data a receiver *transmits*.

Computational complexity is also an important factor. This measures the complexity of the number of field operations carried out by the sender and receiver(s) in a protocol. These include multiplication, addition and exponentiation of field elements. Similarly to communication complexity, computational complexity is also expressed as a function of n (or t) and the greatest complexity for any operation a sender or receiver carries out is considered as the computational complexity of the protocol. Obviously the lower the computational complexity of a protocol the better⁵ but where computational complexity cannot be improved, simplification of protocols (where the sender or receiver carry out less computation) should be sought.

It should be noted that we only consider the computational complexity of the sender and the receiver. We do not consider the computational complexity of nodes in the underlying network used to compose wires connecting the sender and receiver. This is not only because such nodes are abstracted away, but because they only execute the forwarding function - which is a relatively simple process. Similarly to communication complexity, when sender computational complexity or receiver computational complexity is used in the thesis, this will refer to the computational complexity carried out by the sender or receiver respectively.

Transmission complexity (also known as transmission rate) is another factor often considered in secure message transmission protocols which is defined as follows:

$$\frac{\text{Communication Complexity}}{\text{Complexity of Secrets}}$$

Where “Complexity of Secrets” expresses as a function of n the number of secret transmitted between a sender and a receiver. Various work [4, 56, 105, 107, 169] have proven or shown that the optimal transmission complexity of secure message transmission protocols is $O(n)$.

The importance of these factors depends on the application scenario for which secure message transmission protocols are used. For sparse networks - which possibly cannot guarantee the presence of a high number of disjoint network paths, connectivity requirements of algo-

whereas one with lower connectivity requirements possibly could.

⁵This implies that achieving polynomial time protocols is better than exponential time protocols.

gorithms may be the most important of these factors. For applications such as sensor networks - where the preservation of battery life for cheap and small components is very significant, communication complexity may actually be of greatest importance. For applications where computationally inefficient devices are used, perhaps keeping computational complexity as low as possible may be the most important factor.

As proven in various papers in previous work, there are certain limits to each of these factors. For example, as shown in [55], it is not possible to achieve perfect authenticity of message transmission against a t -bounded Byzantine adversary when the number of wires n connecting the sender and the receiver does not satisfy $n \geq 2t + 1$. This is because if a lower number of wires is used, a majority of the message transmitted by the sender cannot be achieved.

During the research process of the thesis we have sought to construct more efficient protocols than those which exist in the literature in terms of both communication and computational complexities using the optimal number of wires.

2.6 Secret Sharing

Different secret sharing schemes have been proposed in the literature. These include threshold secret sharing schemes such as [28, 163], as well as secret sharing schemes for general access structures [20, 29, 96]. Throughout the thesis we only consider threshold secret sharing schemes. To be more specific, Shamir secret sharing [163] will mainly be used.

2.6.1 Shamir Secret Sharing

An m -out-of- n threshold Shamir secret sharing scheme allows for a secret message M to be distributed as a selection of n shares $\{s_1, \dots, s_n\}$, so that the following properties are achieved:

Property 1: Any collection of m shares is able to reconstruct the secret message M .

Property 2: Any subset of $(m - 1)$ or less shares reveals no information about M .

Property 3: Using any collection of m (original) shares, one is able to reconstruct any of the other shares.

Property 4: Given any subset of $m - e$ shares, no information can be obtained about any disjoint subset of e shares.

The collection of shares is constructed using a random polynomial $p(x) = M + a_1x^1 + \dots + a_{m-1}x^{m-1}$ of degree at most $(m - 1)$ - where a_1, \dots, a_{m-1} are uniformly random elements of the finite field - $GF(q)$, used. Shares which are constructed are evaluations of the random polynomial $p(x)$ using various values for x . It is generally common practise for the x values of shares used to be public and for the private data of the shares sent to a party to be the y -coordinates of the points used. It is also common practise when n shares are needed to use points $\{1, p(1), (2, p(2)), \dots, (n, p(n))\}$ of polynomial $p(x)$. This means that the values of the n shares will be the set of values $\{p(1), p(2), \dots, p(n)\}$.

The secret of an m -out-of- n secret sharing scheme can be reconstructed using unaltered shares and Lagrange interpolation. The coefficients of an unknown polynomial $p(x)$ of degree

less than m , defined by points (x_i, y_i) , $1 \leq i \leq m$, are given by the Lagrange interpolation formula:

$$p(x) = \sum_{i=1}^m y_i \prod_{1 \leq j \leq m, j \neq i} \frac{x - x_j}{x_i - x_j}.$$

Since in Shamir secret sharing the secret is $p(0) = a_0 = M$, the shared secret may be expressed as:

$$M = \sum_{i=1}^m c_i y_i, \text{ where } c_i = \prod_{1 \leq j \leq m, j \neq i} \frac{x_j}{x_j - x_i}.$$

The secret message M can therefore be calculated as a linear combination of m shares.

The process of sharing secrets using Shamir secret sharing is similar to constructing Reed Solomon (RS) codes - which were introduced in [153]. Comparisons of these two schemes has been carried out in [121] and in the context of sharing a secret, they are equivalent. Algorithms used in the decoding of Reed-Solomon codes could thus be used in the decoding process of Shamir secret sharing schemes. As using Shamir secret sharing produces codes which are equivalent to Reed Solomon codes [121], the set of all n possible shares in a secret sharing scheme (s_1, \dots, s_n) is called a code and its elements codewords. Similar to Reed Solomon codes, an m -out-of- n secret sharing scheme has a Hamming distance of $(n - m + 1)$ [121]. This distance allows for $(n - m)/2$ error shares to be corrected and still produce the original secret. This could be carried out using Reed-Solomon error correcting decoding algorithms such as [21] which has a computational complexity of $O(d^3 \log q)$ where d denotes the degree of the encoding polynomial and q denotes the size of the finite field used.

When considering an m -out-of- n Shamir secret sharing scheme with n being a factor of m - for example $n = 2m$, it is easy to see that the computational complexity of sharing a secret M is $O(n^2)$. This because a random polynomial of degree at most $(m - 1)$ has to be created. This polynomial consists of $m - 1$ random elements and the value of M - thus overall $m = n/2$ coefficients. To create the n shares, polynomial $p(x)$ has to be evaluated n times. With each one of its evaluation having a complexity of $O(n)$ (using efficient algorithms such as Horner's Rule), the computational complexity of an m -out-of- n Shamir secret sharing scheme is clearly $O(n^2)$.

The computational complexity of the secret reconstruction process depends on the polynomial interpolation algorithm used. Gaussian elimination as shown in [27] requires a computational complexity of $O(n^3)$. Neville's algorithm though can achieve this with a computational complexity of $O(n^2)$ as shown in [150].

2.6.1.1 Variant of Shamir Secret Sharing Scheme

Shamir secret sharing [163] allows for the reconstruction of a secret using polynomial interpolation. When using Shamir secret sharing in message transmission protocols - denoting as p the polynomial from which the shares are constructed, it is usually the norm to transmit share $p(i)$ across wire w_i - where $1 \leq i \leq n$ and n denotes the number of wires connecting a sender and a receiver. In this case, the values of i are common knowledge to all parties.

For the protocol presented in Section 3.2 a variant of the above will be used. The only thing that will vary is that the x -coordinates of points to be used will also remain private. Shares sent

to the receiver will thus consist of both the (x, y) -coordinates of points of the polynomial p used to share a secret.

2.6.2 Alternative Secret Sharing Scheme Friendly to Humans

In Chapter 5 where we consider secure message transmission protocols with a human receiver, an alternative secret sharing scheme will be used. As one of the communicating parties is a human, the secret sharing scheme is designed to be friendlier for humans to use. When using this scheme, a secret will always be shared in an n -out-of- n manner. The number of shares required to reconstruct a secret are consequently equivalent to the number of shares constructed when sharing a secret.

An n -out-of- n secret sharing scheme allows for a secret message M to be distributed as a selection of n number of shares $\{s_1, \dots, s_n\}$ so that the following properties are achieved:

- From the collection of n shares one is able to reconstruct the secret message M .
- Any subset of $(n - 1)$ or less shares reveals no information about M .

Various secret sharing schemes exist in the literature such as those presented in [96, 163]. These schemes though require computation which is difficult for a human receiver (as we consider in Chapter 5) to carry out. Any secret sharing schemes that will be used in the protocols to be presented in Chapter 5 should thus be easy for humans to use. For these protocols, secret sharing mod10 will be the only computation that will be used. We now describe this scheme.

In secret sharing mod10, when a secret (which is quantified by a number composed of a number of digits) is shared to a number of shares (each of which is also a number of equal length to the secret), the sum of all respective digits from all shares mod10 will be equal to the respective digit of the original secret that was shared.

We explain this secret sharing scheme through an example. Supposing that the secret “2597” has to be shared into five shares, initially four shares are *randomly* created - each with the same number of digits as the secret to be shared. As an example, we assume the four random shares are $s_1 = 7291$, $s_2 = 1658$, $s_3 = 9202$ and $s_4 = 7484$. The fifth share is then constructed by first identifying its units, so that when summing the *units* over *all* shares mod10, this will equal the units of the secret to share. In a similar manner, the number of tens/hundreds/thousands of the fifth share are identified so that when summing the tens/hundreds/thousands over *all* shares mod 10, this will equal the tens/hundreds/thousands of the secret to share respectively. With the example random shares given, the value of the fifth share is thus $s_5 = 8172$.

Similar actions to the above description will have to be carried out when sharing a secret in an n -out-of- n manner.

The above description of human friendly secret sharing mod10 can be explained by the following formula:

$$\forall i \in \{1, \dots, k\} : D_i(s) = \sum_{j=1}^n D_i(s_j) \text{ mod } 10$$

Where s denotes the secret, s_j denotes one of the n shares ($1 \leq j \leq n$), k denotes the

length in digits of the secret s (and subsequently the length in digits of all shares both of which are elements of $(Z_{10})^k$), and D_i denotes digit i ($1 \leq i \leq k$) of the secret or share (D_2 would thus identify the second digit of the secret or share). The above can be read as “For every i , the sum of digit i over all shares, mod 10, results in the same value as digit i of the secret”.

It is easy to see that the described scheme is a secret sharing scheme. This is because for one to know the value of M , one must know all n shares. This means that this alternative secret sharing scheme is an n -out-of- n secret sharing scheme - which of course cannot detect or correct any errors. As all shares are random, not knowing the value of at least one share achieves perfect secrecy. This is because the probability of correctly guessing the unknown share is equivalent to the probability of correctly guessing the value of the secret message M itself. Additionally, all messages in \mathcal{M} are possible when less than n shares are known. This is equivalent to information-theoretic secrecy and Shannon’s proof of the one time pad as described in [164].

Constructing an n -out-of- n code using this alternative secret sharing scheme is computationally more efficient than when using Shamir secret sharing. This is because this alternative scheme has a computational complexity of $O(n)$ as opposed to $O(n^2)$ of Shamir secret sharing. The computational complexity is $O(n)$ because $n - 1$ random shares are selected and the digits of the n^{th} share are found through addition mod 10.

Reconstruction of a secret from the shares that were created using this secret sharing scheme can be carried out in a simple to use manner - as described in Appendix D. Ways with which human participants interacted with this alternative secret sharing scheme friendly to humans and how accurately they used it are discussed in Chapter 5. The computational complexity for reconstruction of a secret using this secret sharing scheme is clearly $O(n)$ - as addition of $O(n)$ field elements are the only operations required.

2.6.2.1 Generalization of Secret Sharing Scheme

The secret sharing scheme presented in this section can be generalized to a more abstract use of an abelian group where addition is the operation (\cdot) associated with the set A of the group.

In such a case, the sharing of a secret from the group will be explained by the following:

$$\forall i \in \{1, \dots, k\} : D_i(s) = \sum_{j=1}^n D_i(s_j) \text{ mod } |A|$$

For the above, the definitions of s , s_j , n , k and D_i are similar to those of when human friendly secret sharing mod 10 were explained in the previous section.

When $A = \{0, 1\}$, the generalization of the secret sharing scheme describes XOR-sharing. When using the alternative secret sharing scheme friendly to humans, we assume the case of when $A = \{0, \dots, 9\}$ as it has the favorable property of being easy to evaluate when used by humans - since base 10 is a more familiar base for humans to use.

2.7 Privacy Amplification

Privacy amplification achieves the following for a sender and a receiver - as described in [4].

Suppose the sender and the receiver share b uniformly random finite field elements and that

it is guaranteed that $a < b$ of these elements are completely unknown to the adversary. Then there is a simple technique that allows the sender and the receiver to non-interactively generate a random elements about which the adversary has no information.

Assuming that $q > a + b$, then we can view the b shared random elements as the first b shares in a Shamir's b -out-of- $(a + b)$ -threshold secret sharing scheme. From Property 3 of Shamir Secret Sharing (see Section 2.6.1), these b shares fix all the other shares in the selection of shares. From Property 4 of Shamir Secret Sharing, the adversary has no information about the values of the a 'new' shares. These shares can therefore be taken as the outcome of the privacy amplification.

Chapter 3

Almost Perfectly Secure One-Phase Transmission Protocols

In this chapter we consider almost perfectly secure transmission protocols. Such protocols do not achieve perfect security of message transmission as the availability of these protocols may fail with a negligible probability. When this occurs the receiver of the transmission protocol does not accept a message, but instead accepts the empty message - signified by \perp . These protocols always achieve perfect privacy when executed and whenever the receiver accepts a message from a message space \mathcal{M} , perfect authenticity of message transmission is achieved. Such protocols therefore achieve $(0, 0, \gamma)$ -security - as this has been defined in Section 2.3.

Almost perfectly secure message transmission protocols can occur over any number of phases. In [50, 52, 68] almost perfectly secure protocols were presented for two, three and more phase protocols¹. In this chapter we only consider almost perfectly secure one-phase message transmission protocols.

In such protocols the only communication that occurs does so over a single phase and data is transmitted by a sender to a receiver. A necessary and sufficient condition for one-phase perfectly secure message transmission between a sender and a receiver is $n \geq 3t + 1$ - where n denotes the number of wires² connecting a sender and a receiver and t represents the static active adversary threshold bound. Almost perfectly secure one-phase message transmission protocols consider a lower number of n wires (with $3t + 1 > n \geq 2t + 1$) connecting the sender and the receiver of a communication.

This chapter presents two *new* almost perfectly secure one-phase message transmission protocols. Primarily, the environment for these protocols is described in Section 3.1 providing details of the network and security model considered. In Section 3.2 we present a new polynomial time almost perfectly secure one-phase message transmission protocol and in Section 3.3 we present one of exponential time. The importance of studying exponential time protocols (when polynomial time protocols already exist) was explained in Section 1.5. In both cases, the main idea of the protocol is initially given and this is followed by the formal description of the protocol and its security and complexity analysis. Both protocols *improve* on previous work

¹In the cited work of [52, 68] the security definition used for almost perfectly secure transmission protocols was comparable to that given earlier. The only difference between the two security definitions is that for protocols presented in [52, 68] the receiver would *always* accept a message from the protocol message space and with a negligible probability the accepted message would be different to that sent by the sender.

²The reader is reminded that a wire refers to a node disjoint path.

and the way this is achieved is presented in Section 3.4. We also identify the unsolved open problems for such protocols which could be considered in future work.

3.1 Environment of Almost Perfectly Secure One-Phase Protocols

The network setting considered in one-phase message transmission protocols is that of a directed graph. Additionally, the underlying network which connects a sender and a receiver has node disjoint network paths which allow for the transmission of data from the sender to the receiver. On the contrary, no communication channels exist for the reverse communication (from receiver to sender) to be possible. Because of this, transmission of data can only occur from the sender to the receiver. This network setting is the same as the second network model described in Section 2.2.1.

As opposed to one-phase PSMT protocols which require at least $(3t + 1)$ wires to connect a sender and a receiver, almost perfectly secure one-phase transmission protocols consider a lower number - $n \geq (2t + 1)$, of wires. As shown in [55] $n = (2t + 1)$ is the minimum number of wires required for perfect authenticity in message transmission to be possible. Due to the lower number of wires used, the transmission protocol cannot achieve perfect security. This is because with a small (negligible) probability, the actions of the adversary may cause such protocols to fail.

Almost perfectly secure message transmission protocols should *always* guarantee that the message transmitted by the sender can be decoded by the receiver of the communication. Despite this, actions carried out by the active adversary may result in the receiver decoding more than one (different value) message(s). When this occurs, the receiver cannot know which is the correct message transmitted by the sender of the communication. The receiver thus does not accept a message but instead accepts \perp - signifying the empty message, and in such cases the protocol is said to have failed. The probability with which this occurs is modeled by the variable γ which denotes the availability of a protocol.

Almost perfectly secure message transmission protocols though always achieve perfect privacy when executed. Additionally, when a single message is decoded by the receiver (when the protocol does not fail) this message is accepted by the receiver as the secret of the transmission. As this is guaranteed to be the same message transmitted by the sender, in such cases the receiver accepts the decoded message and perfect authenticity is achieved.

We consider a static t -threshold bounded computationally unlimited active adversary and denote with $n \geq 2t + 1$ (more precisely $n = 2t + 1$) the number of wires which connect the sender and the receiver of a message transmission.

3.2 One-Phase Polynomial Time Protocol

In this section we present a *new* 1-phase polynomial time almost perfectly secure message transmission protocol. The protocol improves upon previous work as it is more efficient with regards to the amount of computation that needs to be carried out and the number of field elements which need to be transmitted. We first present the main idea of our protocol and in Section 3.2.2 describe the main techniques used in the protocol. We then formally present the protocol and its security and complexity proof - showing that it is a polynomial algorithm

regarding both computation and communication complexities.

3.2.1 Protocol Main Idea

As the protocol is a 1-phase protocol, communication can only occur one way from the sender to the receiver. Denoting with M^S the secret message of the transmission, the sender will share M^S using a $(t + 1)$ -out-of- n Shamir secret sharing scheme. This will provide the sender with n shares (s_1, \dots, s_n) of M^S . For each one of the n shares the sender will then carry out a $(t + 1)$ -out-of- n Shamir secret sharing (these new shares are termed as sub-shares for clarity) and will transmit these sub-shares to the receiver - the way this is done is outlined in Section 3.2.2.1. The receiver will then check the correctness of the shares of M^S . Considering only the correct shares, the receiver will proceed to carry out error *detection*. These two schemes are described in Section 3.2.2.2 and Section 3.2.2.3 respectively. If no error is detected the receiver accepts the message that can be reconstructed from these shares. If at least one error is detected, the receiver accepts \perp . When the receiver accepts a value ($M^R \in \mathcal{M} \cup \{\perp\}$) the protocol terminates. As in some cases the receiver accepts \perp and in all other cases the receiver accepts the correct message with perfect secrecy and authenticity, the protocol achieves $(0, 0, \gamma)$ -security.

3.2.2 Main Protocol Techniques

In this section we outline three main techniques used in the protocol. The first is the encoding and transmission of the secret message which is executed by the sender. The latter two are carried out by the receiver and are the identification of faulty wires and error detection schemes.

3.2.2.1 Message Encoding and Transmission

Denoting as M^S the secret message of the transmission, the sender will carry out a $(t + 1)$ -out-of- n Shamir secret sharing of M^S - obtaining shares (s_1, \dots, s_n) . The sender does this by choosing a random polynomial p of degree at most t over $GF(q)$ such that $p(x) = M^S + a_1x^1 + \dots + a_t x^t$ - where a_1, \dots, a_t are uniformly random elements of $GF(q)$. The n shares (s_1, \dots, s_n) are obtained by evaluating $(p(1), \dots, p(n))$.

The sender chooses n random, distinct, non-zero elements (r_1, \dots, r_n) over the finite field - with random element r_i corresponding to wire w_i where $1 \leq i \leq n$. The sender proceeds to construct a $(t + 1)$ -out-of- n Shamir secret sharing scheme of share s_i and transmit the constructed shares in the following manner:

1. The sender chooses a random polynomial of degree at most t which we define as p_i such that $p_i(x) = s_i + a_{i1}x^1 + \dots + a_{it}x^t$ where a_{i1}, \dots, a_{it} are uniformly random elements of $GF(q)$.
2. The n sub-shares of s_i are computed by evaluating p_i at (r_1, \dots, r_n) to obtain $(s_{i1} = p_i(r_1), \dots, s_{in} = p_i(r_n))$.
3. The random elements with the corresponding sub-shares are coupled together to obtain $((r_1, s_{11}, s_{21}, \dots, s_{n1}), (r_2, s_{12}, s_{22}, \dots, s_{n2}), \dots, (r_n, s_{1n}, s_{2n}, \dots, s_{nn}))$.
4. Random element r_i and the definition of polynomial p_i are transmitted over wire w_i .
5. The sender transmits sub-share s_{ij} (where $1 \leq i \leq n$) over wire w_j - where $1 \leq j \leq n$.

Let us start with a preliminary analysis. It is easy to see that the computational complexity of this scheme is polynomial in field operations. The sender carries out Shamir secret sharing of $(n + 1)$ values - the secret and the n shares of the secret. As a $(t + 1)$ -out-of- n Shamir secret sharing of a value requires the sender to choose t random values (to construct the random polynomial) and then evaluate the value of n shares, secret sharing of a value requires $O(n^2)$ field operations and thus has a computational complexity of $O(n^2)$ field operations. As $(n + 1)$ values are secret shared, the computational demands of the sender are $O(n^3)$ field operations.

It is easy to see that the number of field elements transmitted in this scheme are $n * (n + t + 2) = n^2 + nt + 2n$. This is because upon each wire, a polynomial definition (composed of at most $t+1$ field elements) a random element and n sub-shares are transmitted. The communication complexity of this scheme is thus $O(n^2)$ field elements.

3.2.2.2 Faulty Wire Detection

This technique is carried out by the receiver at the end of the protocol phase. From wire w_j the receiver receives a random element r'_j , a polynomial definition p'_j and n sub-shares $(s'_{1j}, \dots, s'_{nj})$. Identification of faulty wires is carried out in the following manner:

Step 1 Initialize set $FAULTY := \emptyset$, $REPEAT_{FLAG} := TRUE$.

Step 2 Identify all faulty sub-shares. If $p'_i(r'_j) \neq s'_{ij}$ - where $1 \leq i \leq n$ and $1 \leq j \leq n$, then share s'_{ij} is identified as a faulty sub-share.

Step 3 Do the following while ($REPEAT_{FLAG} = TRUE$):

a $REPEAT_{FLAG} := FALSE$.

b For $i := 1, \dots, n$

I **IF** wire $w_i \in FAULTY$ GOTO Step IV.

II **IF** there are at least $(t + 1)$ sub-shares from $(s'_{i1}, \dots, s'_{in})$ found to be correct in Step 2 and *not* in $FAULTY$, then do nothing.

III **ELSE** wire $w_i \notin FAULTY$ is identified as a faulty wire. Add w_i to $FAULTY$. Set $REPEAT_{FLAG} := TRUE$.

IV End of loop.

The faulty wire detection scheme described above identifies faulty wires. However, it cannot guarantee that wires identified by the scheme as non-faulty are not controlled by the adversary. It also cannot guarantee that changes were not carried out on data transmitted upon such wires. As will be outlined later in Section 3.2.4, adversary controlled wires can still pass the test of this scheme - even if changes were carried out. In short, adversary controlled wires can achieve this if any alterations carried out on polynomial definitions still result in the algorithm finding at least $(t + 1)$ pair of values (sub-shares) passing the test of Step II.

It is easy to see that the computational complexity of the above scheme is polynomial in field operations. Step 2 is the most computationally expensive step as the correctness of n^2 sub-shares has to be verified. Each of the n random elements have to be evaluated for each of the n (at most) t -degree polynomials. One of these evaluations requires $O(n)$ field operations

thus the computational complexity of Step 2 is $O(n^3)$ field operations. The while loop of Step 3 can be repeated at most t times (as overall at most t faulty wires can be identified in different instances of the while loop). The inner for-loop repeats at most n times each time it is called. The computational complexity of Step 3 is $O(n^3)$ field operations. Overall, the computational complexity of the above scheme is $O(n^3)$ field operations.

3.2.2.3 Error Detection

Error detection is carried out in the following manner. Considering only wires $w_i \notin FAULTY$, share s'_i is the coefficient of polynomial p'_i received from wire w_i which corresponds to the y -intercept of p'_i - thus $s'_i = p'_i(0)$. We denote with $m = n - |FAULTY|$ the resulting number of shares - which are shares of the secret message. Error detection on the m shares is then carried out in the following manner. From the m shares, $(t + 1)$ (random) shares are selected and polynomial p' is obtained from these shares using polynomial interpolation. The remaining $m - (t + 1)$ shares are then checked to see if they lie on p' (i.e. if they evaluate to p'). A share s_i is said to lie on p' if $s_i = p'(i)$. If for any of these shares this is not the case, then an error has been detected and the receiver outputs \perp . If no error is detected, the receiver accepts $M^R = p'(0)$ as the message of the transmission.

The computational complexity of this scheme depends on the polynomial interpolation algorithm used. Gaussian elimination as shown in [27] requires a computational complexity of $O(n^3)$. Neville's algorithm though can achieve this with a computational complexity of $O(n^2)$ as shown in [150]. Checking the correctness of the remaining $m - (t + 1)$ shares (which can be at most t) requires a computational complexity of $O(n^2)$ - as at most t shares are evaluated on an at most t degree polynomial.

Using Neville's algorithm for polynomial interpolation, the computational requirements of this error detection scheme is $O(n^2)$.

3.2.3 Protocol Description

We now present the protocol with $n = 2t + 1$ and for message M^S .

Protocol 1. *Polynomial Time Almost Perfectly Secure (1-phase, n -channel) Message Transmission.*

Start of Phase 1 *The sender does the following:*

1. *The sender uses the scheme of Section 3.2.2.1 to share the secret message M^S and transmit the shares over all wires.*

End of Phase 1 *The receiver does the following:*

1. *The receiver receives n shares and a polynomial definition from each wire.*
2. *The receiver carries out faulty wire detection as described in Section 3.2.2.2 to detect the "correct" shares received from all wires.*
3. *The receiver carries out error detection as described in Section 3.2.2.3 on the m "correct" shares and accepts $M^S = M^R$ or \perp .*

3.2.4 Security and Complexity Analysis

In this section we present the security analysis of the protocol which consists of two parts, the first being the analysis of the protocol's security properties of privacy, authenticity and availability and the second being the analysis of the adversary's best strategy with regards to causing the protocol to fail (thus causing the receiver to accept \perp). We conclude with the protocol's complexity analysis.

3.2.4.1 Security Analysis

Theorem 1. *The above protocol achieves $(0, 0, 1 - (\frac{q-3t}{q-t-1})^t)$ security.*

Proof. We first prove the perfect privacy of the protocol. The secret message M^S is secret shared using a $(t + 1)$ -out-of- n Shamir secret sharing scheme. The adversary can only learn t of these shares - those whose polynomial definitions are sent on adversary controlled wires. For the secret sharing of the remaining shares the adversary only learns t sub-shares of each and thus does not learn any of these shares. As the adversary learns t shares, no information is obtained about the secret message. The protocol therefore achieves perfect privacy.

Perfect authenticity is achieved as the receiver accepts a message only when no errors are detected. This is proven by the following lemma.

Lemma 1. *The faulty wire detection scheme of Section 3.2.2.2 will always identify as "correct" wires, the set of $(t + 1)$ non-faulty wires.*

Proof. Note first that the faulty wire detection scheme identifies as a "correct" wire those whose polynomial definitions correspond to at least $(t + 1)$ received sub-shares. As honest (not adversary controlled) wires do not alter any data transmitted upon them and as there are at least $(t + 1)$ honest wires, honest wires will always be identified as "correct" wires. \square

Due to the above lemma, at least $(t + 1)$ original shares of M^S will be considered by the receiver in the error detection scheme. Because of this, whatever the changes the adversary carries out, the receiver will *never* accept a wrong message. This is because the degree of the polynomial used in the secret sharing of M^S is at most t . This means that any alterations the adversary carries out cannot result in a different t -degree polynomial which includes all the honest $(t + 1)$ shares (corresponding to honest wires) of M^S . Therefore, no matter what alterations the adversary carries out to share values, at least one error will *always* be detected. The receiver will thus never accept a message different to the message sent by the sender. Perfect authenticity is therefore achieved.

We now calculate the availability. Failure of message transmission (i.e. when the receiver outputs \perp) occurs when an error is detected in the error detection scheme of Section 3.2.2. An error is detected by this technique only when the adversary successfully alters at least one share of the secret message. This can be achieved only when a polynomial definition transmitted over a faulty wire is altered and after the execution of the faulty wire detection scheme, the specific wire *does not* belong in set *FAULTY*. For this to occur, the adversary must ensure that at least $(t + 1)$ sub-shares received over all wires lie on the altered polynomial. Assuming that the adversary controls t wires (all of which do not belong in the set *FAULTY*) - and in turn controls t sub-shares, any strategy followed by the adversary must ensure that *at least one*

sub-share not controlled by the adversary lies on the polynomial definition to be altered by the adversary.

The reader is reminded that sub-shares in the scheme correspond to a pair (r_x, s_y) where r_x is a random field element representing the x -value of a point on a 2-dimensional plane and s_y is the evaluation of r_x for a particular polynomial. As the adversary knows the polynomial definition transmitted on adversary controlled wires, the adversary knows the value of all possible sub-shares $((r_x, s_y)$ pairs) that can be constructed using the polynomial. What the adversary *does not know* is which *specific* $(t + 1)$ sub-shares were transmitted over the honest wires.

We denote as p_i the polynomial transmitted by the sender on adversary controlled wire w_i and with p'_i the altered - by the adversary, polynomial. As the adversary controls t wires and alters the polynomial to p'_i , the adversary can alter all shares transmitted on adversary controlled wires to reflect shares of the altered polynomial p'_i .

We denote with r_j the random non-zero x -value transmitted on an honest wire w_j . As the adversary requires $(t + 1)$ sub-shares received over all wires to lie on the altered polynomial (and we assume that the adversary alters shares transmitted on adversary controlled wires correctly so that they lie on p'_i). We denote with event A, the event when at least one of the $(t + 1)$ sub-shares transmitted over honest wires lie on the altered polynomial. In other words:

$$\text{Event A : } p_i(r_j) = p'_i(r_j).$$

Additionally, we denote with event B the event when the adversary alters the polynomial transmitted on an adversary controlled wire different to that transmitted by the sender, so that a different (to the original) share is reconstructed from this altered polynomial:

$$\text{Event B : } p_i(0) \neq p'_i(0).$$

The above implies that $p_i \neq p'_i$.

Given the above, the adversary succeeds with an attack with the following conditional probability:

$$\text{prob}(A|B) = \frac{\text{prob}(A, B)}{\text{prob}(B)}.$$

Since both polynomials p_i and p'_i are polynomials of degree at most t and *since they are different*, they can share at most t common points between them - this is a result from coding theory³. For these t points $p_i(x) = p'_i(x)$ - where x is distinct for each of the t points.

We now analyze the probability $\text{prob}(A, B)$. This is when the adversary carries out an alteration to polynomial p'_i and at least one of the shares transmitted on honest wires lie on this altered polynomial. It should be pointed out that the adversary should not achieve all $t + 1$ shares transmitted on honest wires to lie on p'_i (otherwise no change has occurred as $p'_i = p_i$).

Since polynomial p_i is of degree at most t , there can be up to t points which have a specific (and same) value for their $p_i(r_{x_j})$ evaluation. For example, for a specific value $y \in GF(q)$

³This is essentially the same as the fact that any two lines - which is a one degree polynomial, on a two dimensional plane can share at most one point between them. The same idea also applies for t degree polynomial which can share at most t points between them and follows from the fact that $\text{degree} \leq t$ and that $t + 1$ points define any polynomial.

there can be t points such that $y = p_i(r_{x_j})$ where $1 \leq j \leq t$. (It should be noted that when polynomial p_i is of degree zero all x -values will evaluate to the same y value.)

We assume that for p_i there is such a value y . We also assume that **none** of the shares transmitted on adversary controlled wires with x_i value correspond to (x_i, y) . The adversary can then assume that this y -value was transmitted as a share for each of the honest wires. The adversary does this as this value is the most popular and thus the most likely value to have been transmitted. With the adversary knowing the definition of p_i , the adversary also knows the r_j values sent on these wires. Overall, one way with which the adversary can alter to polynomial p_i to polynomial p'_i is in the following manner:

1. The adversary will look at p_i and the evaluations $p_i(x)$ over all x in the finite field.
2. The adversary will list the evaluations $y = p_i(x)$ in order of popularity - determined by the number of different values of x which evaluate to the same value of y .
3. As polynomial p_i is of degree at most t , there can be up to t different x -values which appear to evaluate to the same y value⁴.
4. In the context of calculating an upper bound we assume that there is one such value y' .
5. As the adversary knows the definition of p_i the adversary can calculate the t x -values which evaluate to y' .
6. We assume that upon adversary controlled wires *none* of these t x -values were transmitted.
7. The adversary alters polynomial p_i to polynomial p'_i in the following manner. Using these t x -values and their evaluation to y' the adversary simply alters the shared secret from s_i to a different s'_i . This gives a new polynomial p'_i and by calculating it (using polynomial interpolation) the adversary is able to alter the shares transmitted on adversary controlled wires to correspond to $((r_{A_1}, y'_{A_1}), (r_{A_2}, y'_{A_2}), \dots, (r_{A_t}, y'_{A_t}))$ (as opposed to their original (r_{A_j}, y_{A_j}) form - where $1 \leq j \leq t$ and where r_{A_j} represent random elements transmitted on adversary controlled wires) so that they constitute shares of p'_i .

The adversary is successful with this attack when at least one of the t x -values corresponding to the most popular y -value was in fact transmitted upon an honest wire. So, the probability of the adversary getting a share transmitted on an honest wire correct with a single guess is less than or equal to $\frac{t}{q-t-1}$.⁵

Over t guesses, and when the (intelligent) adversary uses sampling without replacement (thus choosing t *different* non-zero finite field elements which are also different to those transmitted on adversary controlled wires), the probability of the adversary getting none of the t

⁴When polynomial p_i is of degree zero all x -values will evaluate to the same y value.

⁵Here $q - t - 1$ and not q is used in the denominator as the x values are *non-zero* and randomly selected over the finite field of size q . An intelligent adversary will also take into account the t x -values transmitted on adversary controlled wires - thus the probability being $\frac{t}{q-t-1}$. Thus all x values are different and the adversary knows t of these, so the values could only come from $q - t - 1$ possibilities.

points correct - signified by P_{F_1} (probability of failure for one polynomial alteration), can be given by the hypergeometric distribution using the following:

$$P_{F_1} = \frac{\binom{t}{0} \binom{(q-t-1)-t}{t}}{\binom{q-t-1}{t}} = \frac{\binom{q-2t-1}{t}}{\binom{q-t-1}{t}} = \frac{\frac{(q-2t-1)!}{t!(q-3t-1)!}}{\frac{(q-t-1)!}{t!(q-2t-1)!}} = \frac{(q-2t-1)!(q-2t-1)!}{(q-3t-1)!(q-t-1)!}$$

This can be simplified to $\frac{(q-2t-1)(q-2t-2)\dots(q-3t)(q-3t-1)\dots(q-2t-1)}{(q-3t-1)(q-t-1)(q-t-2)\dots(q-2t)(q-2t-1)}$.

The above can be simplified further to $P_{F_1} = \frac{(q-2t-1)(q-2t-2)\dots(q-3t)}{(q-t-1)(q-t-2)\dots(q-2t)}$.

The probability of the adversary getting at least one point correct is equal to 1 minus the probability of the adversary getting none of the t points correct. The probability thus equals to $1 - P_{F_1} = 1 - \frac{(q-2t-1)(q-2t-2)\dots(q-3t)}{(q-t-1)(q-t-2)\dots(q-2t)}$.

So:

$$\text{prob}(A|B) = 1 - \frac{(q-2t-1)(q-2t-2)\dots(q-3t)}{(q-t-1)(q-t-2)\dots(q-2t)}.$$

We now calculate the probability of event B . As polynomials of degree at most t are used, there are clearly q^{t+1} number of polynomials of this degree. As for event B to occur, the adversary has to alter to a different polynomial which interpolates a different share value, all the adversary has to do is to change to a polynomial p'_i whose y -intercept is different to the original share value (different to the y -intercept of the original polynomial p_i). The adversary thus has to avoid changing p'_i to one of q^t polynomials - these are the polynomials which share the y -intercept of the original polynomial p_i but all other t coefficients of the polynomial are either the same or different. So the probability of event B occurring is $\text{prob}(B) = \frac{q^{t+1}-q^t}{q^{t+1}} = 1 - \frac{1}{q}$. It is clear that $\text{prob}(B) \geq 1 - \frac{t}{q}$ and when $t \ll q$ this is close to 1. In fact, when considering an intelligent adversary we can assume $\text{prob}(B) = 1$ and that the adversary will not make such a trivial error.

We thus have the following:

$$\text{prob}(A|B) \leq 1 - \frac{(q-2t-1)(q-2t-2)\dots(q-3t)}{(q-t-1)(q-t-2)\dots(q-2t)}$$

This can be simplified to:

$$\text{prob}(A|B) \leq 1 - \frac{(q-2t-1)(q-2t-2)\dots(q-3t)}{(q-t-1)(q-t-2)\dots(q-2t)} \leq 1 - \frac{(q-3t)^t}{(q-t-1)^t}$$

The above is valid as $\frac{(q-3t)^t}{(q-t-1)^t} < \frac{(q-2t-1)(q-2t-2)\dots(q-3t)}{(q-t-1)(q-t-2)\dots(q-2t)}$.

γ -availability is therefore achieved if the following condition is true:

$$1 - \frac{(q-3t)^t}{(q-t-1)^t} \equiv 1 - \left(\frac{q-3t}{q-t-1}\right)^t \leq \gamma$$

Therefore, probability of Protocol 1 failure $\gamma \geq 1 - \left(\frac{q-3t}{q-t-1}\right)^t$.

When one wants to run the protocol of this section with a specific required bound on protocol availability, they will have to select an appropriate value of q - taking into account the

value of t , so the above inequality applies to the acceptable protocol probability of failure (as chosen by the users of the protocol).

Given that $\gamma \geq 1 - \left(\frac{q-3t}{q-t-1}\right)^t$, the following expansions and simplifications give us the inequality which describes this appropriate value of q :

$$\begin{aligned} \gamma &\geq 1 - \left(\frac{q-3t}{q-t-1}\right)^t \equiv \left(\frac{q-3t}{q-t-1}\right)^t \geq 1 - \gamma \equiv \frac{q-3t}{q-t-1} \geq (1-\gamma)^{\frac{1}{t}} \\ &\equiv q-3t \geq (1-\gamma)^{\frac{1}{t}} q - t - 1 \equiv q-3t \geq q(1-\gamma)^{\frac{1}{t}} - t(1-\gamma)^{\frac{1}{t}} - (1-\gamma)^{\frac{1}{t}} \\ &\equiv q - q(1-\gamma)^{\frac{1}{t}} \geq 3t - t(1-\gamma)^{\frac{1}{t}} - (1-\gamma)^{\frac{1}{t}} \equiv q(1 - (1-\gamma)^{\frac{1}{t}}) \geq 3t - t(1-\gamma)^{\frac{1}{t}} - (1-\gamma)^{\frac{1}{t}} \end{aligned}$$

The above is finally simplified to: $q \geq \frac{3t - (t+1)(1-\gamma)^{\frac{1}{t}}}{1 - (1-\gamma)^{\frac{1}{t}}}$.

It is easy to see that γ can be made *negligible* in the security parameters given a sufficiently large $\mathbb{F}(q)$.

Considering $q = 2^x$ (where x is a variable) the probability of protocol availability is given by:

$$1 - \left(\frac{q-3t}{q-t-1}\right)^t \equiv 1 - \left(\frac{2^x - t - 1 - 2t + 1}{2^x - t - 1}\right)^t \equiv 1 - \left(1 - \frac{2t-1}{2^x - t - 1}\right)^t$$

$$\text{This can be re-written as: } 1 - \sum_{i=0}^t \binom{t}{i} (-1)^i \left(\frac{2t-1}{2^x - t - 1}\right)^i$$

$$\text{Which is equivalent to: } 1 - 1 - \sum_{i=1}^t \binom{t}{i} (-1)^i \left(\frac{2t-1}{2^x - t - 1}\right)^i$$

$$\text{Which is simplified to: } \sum_{i=1}^t \binom{t}{i} \left(\frac{2t-1}{2^x - t - 1}\right)^i$$

Given that t is a constant, this is clearly an exponential function, meaning that γ can be made *negligible* in the security parameters given a sufficiently large $\mathbb{F}(q)$ - such as $q = 2^x$ which is a protocol drawback. □

3.2.4.2 Analysis of best Adversary Strategy towards Protocol Failure

We now show that given the above probability with which the adversary can cause the availability of the protocol to fail, the best strategy for the adversary to follow is to change the polynomial definition transmitted upon a *single* adversary controlled wire.

Probability of success P_1 when adversary alters a *single* wire

We denote with P_1 the probability of the adversary successfully altering the polynomial transmitted on a single wire.

This means that the adversary will alter the polynomial definition transmitted on an adversary controlled wire from p' to p'' where $p'(0) \neq p''(0)$. A successful alteration refers to when despite the alteration, the checks carried out by the receiver still find at least $(t+1)$ sub-shares which lie on the altered polynomial p'' . This probability has been calculated earlier and is equal

to the following:

$$1 - P_{F_1} = 1 - \frac{\binom{q-2t-1}{t}}{\binom{q-t-1}{t}} = 1 - \frac{(q-2t-1)(q-2t-2)\dots(q-3t)}{(q-t-1)(q-t-2)\dots(q-2t)}$$

Probability of success P_2 when adversary alters two wires

For the evaluation of P_2 , we have to take into account the two different strategies the adversary can follow when it comes to choosing the two sets of t x-values for the two wires.

The first strategy is when the two sets of t x-values have common points between them whilst the second is when these two sets are disjoint to each other. For each of the two strategies we will show that altering a single wire is the best strategy the adversary can follow.

Strategy 1: Common points between two sets of t x-values

In our analysis, we use w_{A_1} and w_{A_2} to identify the two adversary controlled wires upon which changes will be carried out and use $w_{A_1} \cap w_{A_2}$ to denote the common points between them.

We consider $|w_{A_1} \cap w_{A_2}| = t - 1$ and identify the following three cases when the adversary fails with an attack on protocol availability:

- CASE 1: None of the x-values in $w_{A_1} \cap w_{A_2}$ are transmitted on honest wires and when the remaining x-value of both w_{A_1} and w_{A_2} are different to those transmitted on honest wires.
- CASE 2: None of the x-values selected for w_{A_1} (and in effect all x-values in $w_{A_1} \cap w_{A_2}$) are transmitted on honest wires and only the remaining x-value for w_{A_2} is the same as a value transmitted on an honest wire⁶.
- CASE 3: None of the x-values selected for w_{A_2} are transmitted on honest wires and only the remaining x-value for w_{A_1} is the same as a value transmitted on an honest wire.

It should be noted that the case of when $w_{A_1} \cap w_{A_2}$ is not empty, but both $w_{A_1} \setminus w_{A_2}$ and $w_{A_2} \setminus w_{A_1}$ are empty implies that the two sets of t x-values have the same points between them. This strategy is in effect equivalent to the strategy of altering the polynomial upon a single wire as considered and analysed in the previous sub-section.

We now calculate the probabilities for each case:

- CASE 1: This is equivalent to the adversary guessing $t + 1$ points, all of which are different to those transmitted on honest wires. Similar to the earlier calculation of P_{F_1} , it is easy to see that this probability is equivalent to $\frac{(q-2t-1)(q-2t-2)\dots(q-3t)(q-3t-1)}{(q-t-1)(q-t-2)\dots(q-2t)(q-2t-1)}$
- CASE 2: This is equivalent to the adversary guessing $t + 1$ points, t of which are different to those transmitted on honest wires and only one being the same. It is easy to see that this probability is equal to P_{F_1} times the probability of guessing one correct value (using sampling without replacement). Thus this probability is equal to $\frac{(q-2t-1)(q-2t-2)\dots(q-3t)(t)}{(q-t-1)(q-t-2)\dots(q-2t)(q-2t-1)}$

⁶Given that w_{A_1} will be identified as an adversary controlled wire and thus ignored, at least two sub-shares from w_{A_2} are required to lie on an altered polynomial

CASE 3: It is easy to see that this probability is equivalent to that of Case 2.

For this strategy, the combined probability of the adversary failing with an attack on protocol availability is given by $\left(\frac{(q-2t-1)(q-2t-2)\dots(q-3t)}{(q-t-1)(q-t-2)\dots(q-2t)} \times \frac{(q-3t)+t+t}{(q-2t-1)}\right)$ which is equivalent to $\left(\frac{(q-2t-1)(q-2t-2)\dots(q-3t)(q-t-1)}{(q-t-1)(q-t-2)\dots(q-2t)(q-2t-1)}\right)$.

P_2 for this scenario is thus equal to: $P_2 = 1 - \left(\frac{(q-2t-1)(q-2t-2)\dots(q-3t)(q-t-1)}{(q-t-1)(q-t-2)\dots(q-2t)(q-2t-1)}\right)$.

Given the value of P_1 and P_2 , it is apparent that the following inequality $P_1 \geq P_2$ is valid (as the last fractional term of P_2 is greater than 1):

$$1 - \frac{(q-2t-1)(q-2t-2)\dots(q-3t)}{(q-t-1)(q-t-2)\dots(q-2t)} \geq 1 - \frac{(q-2t-1)(q-2t-2)\dots(q-3t)(q-t-1)}{(q-t-1)(q-t-2)\dots(q-2t)(q-2t-1)}$$

This means that when it comes to comparing the current strategy, the best strategy for the adversary to follow is to change the polynomial definition transmitted upon a *single* adversary controlled wire.⁷

Strategy 2: Two disjoint sets of t x-values

Before the evaluation of P_2 we first identify the two cases when an adversary attack involving the alteration of polynomial definitions upon two adversary controlled wires (denoted as w_1 and w_2) will be successful:

CASE 1: When both⁸ w_1 and w_2 pass the test of the faulty wire detection scheme (as described in Section 3.2.2.2).

CASE 2: In the event when one of the two adversary controlled wires *does not* pass the test of the faulty wire detection scheme⁹ the other wire still passes this test¹⁰.

We now analyze the probability P_2 . When the adversary carries out this attack (alters polynomials transmitted on two wires), the following six different events can occur as presented in Table 3.1.

The combined probability of these events sums to 1 as follows:

$$1 = P_{C1_{w_1}} P_{C1_{w_2}} + P_{F1_{w_1}} P_{C2_{w_2}} + P_{C2_{w_1}} P_{F1_{w_2}} + P_{F1_{w_1}} P_{F1_{w_2}} + P_{F1_{w_1}} P_{F2_{w_2}} + P_{F2_{w_1}} P_{F1_{w_2}}$$

Where all the following combined probabilities are independent because the adversary selects the two sets of t x-values before the protocol begins and as the adversary has no knowledge

⁷Using a similar analysis, it is easy to see that when the difference between the two sets of t x-values increases, the inequality $P_1 \geq P_2$ will still remain valid, especially when using a large finite field size, such as for example a cubic field size in the security parameter t ($q = t^3$).

⁸This means that for *both* w_1 and w_2 at least one sub-share not controlled by the adversary lies on the polynomial definition(s) altered by the adversary.

⁹This will occur when *none* of the sub-shares not controlled by the adversary lie on the polynomial definition altered by the adversary.

¹⁰This will occur when *at least two* sub-shares not controlled by the adversary lie on the polynomial definition altered by the adversary. This is a requirement as for the wire to pass the test of the faulty wire detection scheme, at least $t + 1$ sub-shares need to lie on the polynomial corresponding to the adversary controlled wire. Assuming that the adversary alters all shares on adversary controlled wires to reflect shares of an altered polynomial, and given that one adversary controlled wire is not considered, at least two sub-shares from non-adversary controlled wires are required to lie on this altered polynomial.

Event	Correct points for w_1	Correct points for w_2	Adversary Attack	Denoted by
1	At least 1 (P_{C1w_1})	At least 1 (P_{C1w_2})	Successful	$P_{C1w_1} P_{C1w_2}$
2	0 (P_{F1w_1})	At least 2 (P_{C2w_2})	Successful	$P_{F1w_1} P_{C2w_2}$
3	At least 2 (P_{C2w_1})	0 (P_{F1w_2})	Successful	$P_{C2w_1} P_{F1w_2}$
4	0 (P_{F1w_1})	0 (P_{F1w_2})	Fails	$P_{F1w_1} P_{F1w_2}$
5	0 (P_{F1w_1})	1 (P_{F2w_2})	Fails	$P_{F1w_1} P_{F2w_2}$
6	1 (P_{F2w_1})	0 (P_{F1w_2})	Fails	$P_{F2w_1} P_{F1w_2}$

Table 3.1: Events which can occur when the adversary alters polynomials transmitted on two wires.

on the success or not of one of the two sets, they are both chosen at the same time independently of each other:

- $P_{C1w_1} P_{C1w_2}$ denotes the combined probabilities P_{C1w_1} - when the adversary gets at least one point not controlled by the adversary correct for w_1 , and P_{C1w_2} - when the adversary gets at least one point not controlled by the adversary correct for w_2 .
- $P_{F1w_1} P_{C2w_2}$ denotes the combined probabilities P_{F1w_1} - when the adversary fails to get any point not controlled by the adversary correct for w_1 , and P_{C2w_2} - when the adversary gets at least two points not controlled by the adversary correct for w_2 .
- $P_{C2w_1} P_{F1w_2}$ denotes the combined probabilities P_{C2w_1} - when the adversary gets at least two points not controlled by the adversary correct for w_1 , and P_{F1w_2} - when the adversary fails to get any point not controlled by the adversary correct for w_2 .
- $P_{F1w_1} P_{F1w_2}$ denotes the combined probabilities P_{F1w_1} - when the adversary fails to get any point not controlled by the adversary correct for w_1 , and P_{F1w_2} - when the adversary fails to get any point not controlled by the adversary correct for w_2 .
- $P_{F1w_1} P_{F2w_2}$ denotes the combined probabilities P_{F1w_1} - when the adversary fails to get any point not controlled by the adversary correct for w_1 , and P_{F2w_2} - when the adversary fails to get at least two points not controlled by the adversary correct for w_2 .
- $P_{F2w_1} P_{F1w_2}$ denotes the combined probabilities P_{F2w_1} - when the adversary fails to get at least two points not controlled by the adversary correct for w_1 , and P_{F1w_2} - when the adversary fails to get any point not controlled by the adversary correct for w_2 .

The probability of adversary success P_2 when the polynomials on two wires are altered is given by:

$$P_2 = P_{C1w_1} P_{C1w_2} + P_{F1w_1} P_{C2w_2} + P_{C2w_1} P_{F1w_2}$$

where:

- P_{C1w_1} is the same as P_1 calculated earlier, thus $P_{C1w_1} = 1 - \frac{\binom{q-2t-1}{t}}{\binom{q-t-1}{t}}$ and from now on is denoted by $P_{C1w_1} = 1 - A$.

- P_{C1w_2} is similar to the calculation of P_1 except that an intelligent adversary will *not* select the t x -values selected for w_1 (to increase the probability of success). Thus $P_{C1w_2} = 1 - \frac{\binom{q-3t-1}{t}}{\binom{q-2t-1}{t}}$ and from now on is denoted by $P_{C1w_1} = 1 - X$.
- P_{F1w_1} is equivalent to A .
- P_{C2w_2} is given by 1 - Probability adversary getting no points right - Probability adversary getting one point right for w_2 . This is equivalent to $P_{C2w_2} = 1 - \frac{\binom{q-3t-1}{t}}{\binom{q-2t-1}{t}} - \frac{\binom{t}{1} \binom{q-3t-1}{t-1}}{\binom{q-2t-1}{t}}$ and from now on is denoted by $P_{C1w_1} = 1 - X - Y$.
- P_{C2w_1} is given by 1 - Probability adversary getting no points right - Probability adversary getting one point right for w_1 . This is equivalent to $P_{C2w_2} = 1 - \frac{\binom{q-2t-1}{t}}{\binom{q-t-1}{t}} - \frac{\binom{t}{1} \binom{q-2t-1}{t-1}}{\binom{q-t-1}{t}}$ and from now on is denoted by $P_{C1w_1} = 1 - A - B$. It is easy to see¹¹ that $A < B$.
- P_{F1w_2} is equivalent to X .

With all the above we have:

$$P_2 = (1 - A)(1 - X) + A(1 - X - Y) + X(1 - A - B)$$

This evaluates to:

$$P_2 = 1 - A - X + AX + A - AX - AY + X - AX - BX$$

Which is simplified to:

$$P_2 = 1 - AY - AX - BX$$

Finding the best adversary strategy

We now use the above calculations for the evaluation of probabilities P_1 and P_2 to find the best adversary strategy. By this we mean the strategy (of whether to alter the polynomial definition upon a single or two wires) which will provide the adversary with the highest probability of carrying out a successful attack on the availability of the protocol.

¹¹As calculated earlier, $A = \frac{(q-2t-1)(q-2t-2)\dots(q-3t)}{(q-t-1)(q-t-2)\dots(q-2t)}$. Similar analysis will show that $B = t * \frac{t*(q-2t-1)^2}{q-t-1!q-3t-2!}$ and that $B = t^2 * \frac{(q-2t-1)(q-2t-2)\dots(q-3t)(q-3t-1)}{(q-t-1)(q-t-2)\dots(q-2t)}$

Theorem 2. *Altering a single wire is the best strategy the adversary can follow given a large enough q - such as $q = t^3 + 4t$.*

Claim 1. *If $P_1 > P_2$ is a valid inequality, then altering a single wire is the best adversary strategy.*

We now carry out simplifications on the inequality $P_1 > P_2$.

$$1 - A > 1 - AY - AX - BX$$

$$\Leftrightarrow$$

$$AY + AX + BX > A$$

The reader is referred to Appendix C for a graphical representation of the above inequality which shows it to be a valid inequality.

Given that $B > A$

$$AY + AX + BX > AY + AX + AX > A$$

And we can proceed with the validity proof of the following $AY + AX + AX > A$ inequality which will also suggest the correctness of the $P_1 > P_2$ inequality.

$$AY + AX + AX > A \equiv A(Y + X + X) > A \equiv Y + 2X > 1$$

We proceed with the evaluations of X and Y .

$$2 * \frac{\binom{q-3t-1}{t}}{\binom{q-2t-1}{t}} + t * \frac{\binom{q-3t-1}{t-1}}{\binom{q-2t-1}{t}} > 1$$

$$\Leftrightarrow$$

$$2 * \frac{\frac{(q-3t-1)!}{t!(q-4t-1)!}}{\frac{(q-2t-1)!}{t!(q-3t-1)!}} + t * \frac{\frac{(q-3t-1)!}{(t-1)!(q-4t-2)!}}{\frac{(q-2t-1)!}{t!(q-3t-1)!}} > 1$$

$$\Leftrightarrow$$

$$\frac{2(q-3t-1)!(q-3t-1)!}{(q-4t-1)!(q-2t-1)!} + \frac{t^2(q-3t-1)!(q-3t-1)!}{(q-4t-2)!(q-2t-1)!} > 1$$

$$\Leftrightarrow$$

$$\frac{2(q-3t-1)!(q-3t-1)! + t^2(q-4t-1)(q-3t-1)!(q-3t-1)!}{(q-4t-1)!(q-2t-1)!} > 1$$

$$\Leftrightarrow$$

$$2 + t^2(q-4t-1) > \frac{(q-4t-1)!(q-2t-1)!}{(q-3t-1)!(q-3t-1)!}$$

$$\Leftrightarrow$$

$$2 + t^2(q - 4t - 1) > \frac{(q - 2t - 1) \dots (q - 3t)}{(q - 3t - 1) \dots (q - 4t)}$$

It is easy to see that $\frac{(q-2t-1)^t}{(q-4t)^t} > \frac{(q-2t-1)\dots(q-3t)}{(q-3t-1)\dots(q-4t)}$ and we will continue with the proof using $\left(\frac{q-2t-1}{q-4t}\right)^t$. Thus:

$$2 + t^2(q - 4t - 1) > \frac{(q - 2t - 1)^t}{(q - 4t)^t} > \frac{(q - 2t - 1) \dots (q - 3t)}{(q - 3t - 1) \dots (q - 4t)}$$

$$\Downarrow$$

$$2 + t^2(q - 4t - 1) > \left(\frac{q - 4t + 2t - 1}{q - 4t}\right)^t$$

$$\Downarrow$$

$$2 + t^2(q - 4t - 1) > \left(1 + \frac{2t - 1}{q - 4t}\right)^t$$

Given a cubic field size in the security parameter t - such as $q = t^3 + 4t$ the above becomes¹²:

$$2 + t^2(t^3 + 4t - 4t - 1) > \left(1 + \frac{2t - 1}{t^3 + 4t - 4t}\right)^t$$

$$\Downarrow$$

$$2 + t^2(t^3 - 1) > \left(1 + \frac{2t - 1}{t^3}\right)^t$$

As $\left(1 + \frac{2t}{t^3}\right)^t > \left(1 + \frac{2t-1}{t^3}\right)^t$ we will proceed with using the $\left(1 + \frac{2t}{t^3}\right)^t$ term.

$$2 + t^2(t^3 - 1) > \left(1 + \frac{2t}{t^3}\right)^t > \left(1 + \frac{2t - 1}{t^3}\right)^t$$

$$\Downarrow$$

$$2 + t^2(t^3 - 1) > \left(1 + \frac{2}{t^2}\right)^t$$

$$\Downarrow$$

$$2 + t^2(t^3 - 1) > \left(1 + \frac{2}{t}\right)^t$$

We now show that the above inequality is valid. The case of when $t = 1$ where the inequality does not apply is irrelevant for this analysis where we consider the alteration of polynomials upon at least two wires, thus when $t \geq 2$. When $t = 2$, it is easy to see that the inequality is also satisfied. Similarly, as t increases the left hand side of the inequality increases in value whereas the right hand side decreases towards 1.

Overall, the above analysis shows that the best strategy for the adversary to follow is to

¹²It should be noted that when using a quadratic field size such as $q = t^2 + 1$ the proof of the inequality $P_1 > P_2$ is not valid for all values of t and thus a cubic field size in the security parameter t should be used.

change the polynomial definition transmitted upon a *single* adversary controlled wire only.

3.2.4.3 Complexity Analysis

We now analyze the complexity of the protocol. Following on from the analysis carried out in Section 3.2.2.1, the communication complexity of the protocol is $O(n^2)$.

The computational complexities of both sender and receiver are polynomial. This is because both the sender and the receiver execute polynomial time schemes as presented in Section 3.2.2. The sender executes the scheme of Section 3.2.2.1 which as analyzed has a computational complexity of $O(n^3)$. The receiver executes the schemes of Section 3.2.2.2 and Section 3.2.2.3 which as analyzed have computational complexities of $O(n^3)$ and $O(n^2)$ respectively. The computational complexity of the receiver is thus $O(n^3)$.

3.3 One-Phase Exponential Time Protocol

In this section we present a *new* one-phase exponential time almost perfectly secure message transmission protocol. This protocol achieves this type of security with optimal $O(n)$ communication complexity. Furthermore, the protocol achieves optimal $O(n)$ transmission rate. This protocol improves on previous protocols with similar communication and computational complexities as it is more general in its use. The importance of studying exponential time protocols (when polynomial time protocols already exist) was explained in Section 1.5.

Initially we provide the main idea of our protocol. We then formally present our protocol and the formal security and complexity proofs.

3.3.1 Protocol Main Idea

The secret message M^S will be secret shared using an $(mt + 2)$ -out-of- mn Shamir secret sharing scheme - where $m > 2$ is a constant. m consecutive shares are grouped together and transmitted across a wire to the receiver - with each m -tuple transmitted only once¹³.

At the end of the first phase, the receiver will receive m shares from each wire. Shares received from some wires may have been altered by the adversary during the transmission of the protocol phase. The receiver proceeds to check all possible $(t + 1)$ subset of wires to see if *all* shares received from these wires lie on the same polynomial.

The protocol fails when due to the adversary actions, the receiver finds more than one subset of $(t + 1)$ wires whose received shares all lie on a polynomial. An additional requirement for the protocol to fail is for more than one different value to be reconstructed from these polynomials.

3.3.2 Formal Protocol Description

We now formally present the protocol with $n = 2t + 1$ and for message $M^S \in \mathcal{M}$.

Protocol 2. *Exponential Time Almost Perfectly Secure (1-phase, n -channel) Message Transmission.*

Start of Phase 1 *The sender does the following for $i := 1, \dots, n$.*

¹³The way shares of the message are transmitted over wires is in effect an m -folded Reed Solomon code as described in [87].

1. The sender chooses a random polynomial p of degree at most $(mt + 1)$ over $GF(q)$ such that:

$$p(x) = M^S + a_1x^1 + \dots + a_{mt+1}x^{mt+1}$$

where a_1, \dots, a_{mt+1} are random elements of $GF(q)$.

2. mn shares are computed by evaluating $p(x)$ at x_1, x_2, \dots, x_{mn} to obtain shares $(s_1, s_2, \dots, s_{mn})$.
3. The sender transmits shares $\{s_{((i-1)m+1)}, \dots, s_{im}\}$ upon wire w_i .

End of Phase 1 The receiver does the following:

1. The receiver receives m shares from each wire.
2. The receiver initializes the set $MESSAGES = \emptyset$.
3. For all $\binom{n}{t+1}$ subsets of wires the receiver checks if **all** shares received from the same subset of $(t+1)$ wires lie on the same polynomial p' .¹⁴ If this is so, the receiver calculates $v = p'(0)$ and adds v in set $MESSAGES$ if $v \notin MESSAGES$. If $|MESSAGES| = 2$ the receiver accepts \perp and the protocol terminates.
4. If the protocol has not terminated prematurely and $|MESSAGES| = 1$ the receiver accepts $v \in MESSAGES$ as the message of the transmission.

Technical Comment Before we present the security proof we first explain why the degree of polynomial p used to share M^S is at most $(mt + 1)$ and not at most mt . To achieve a contradictory statement we assume the degree of the polynomial to be at most mt .

The adversary has the ability to learn mt shares transmitted on t adversary controlled wires. This is one less than the number of shares required to learn the secret. For a non-uniform message distribution the adversary can assume the secret of the transmission to be the most likely message in the distribution. By doing this, the adversary now knows $(mt + 1)$ shares of the polynomial and thus can learn all other shares transmitted by the sender. The adversary can do this by interpolating polynomial p using the $(mt + 1)$ shares and then evaluating the polynomial to learn the other shares.

The adversary can then alter the shares transmitted on adversary controlled wires in such a way so that at least a second value (different to the secret) will be decoded by the receiver. This is easy for the adversary to achieve and it can be done in the following manner. The adversary can consider all the shares transmitted upon $1 \leq k \leq t$ honest (not adversary controlled) wires. The adversary then chooses a polynomial p' such that $p'(0) \neq p(0)$ which also shares the points corresponding to shares transmitted on the k honest wires. Once p' is selected, the adversary can evaluate p' to calculate values of shares to be altered on adversary controlled wires to ensure that shares from at least $(t + 1)$ wires lie on p' . This attack is successful when the secret of the transmission is guessed correctly by the adversary - which can be done with a relative probability for non-uniform message distributions.

¹⁴The receiver does this by randomly selecting $(mt + 2)$ of the $(mt + m)$ shares received from the subset of $(t + 1)$ wires, constructing polynomial p' interpolated by these shares and checking that the remaining $m - 2$ shares all lie on p' . A remaining share s' which corresponds to x' lies on p' iff $p'(x') = s'$.

This attack can be prevented by increasing the degree of the polynomial from at most mt to at most $(mt + 1)$. In this way, even if the adversary guesses the secret of the transmission, the adversary only learns $(mt + 1)$ shares of the polynomial. This is one less than the number of shares required to define the polynomial which constructed all the shares. The adversary thus cannot carry out alterations to present a second message with certainty. Instead, the adversary can only achieve this with a specific probability for which an upper bound will be proven in the next section.

It should be pointed out, that if considering a uniform message distribution, sharing the secret using an at most mt degree polynomial with $m \geq 2$ is sufficient.

3.3.3 Security and Complexity Analysis

3.3.3.1 Security Analysis

Theorem 3. *The above protocol achieves $\left(0, 0, \frac{\binom{2t+1}{t+1}-1}{q}\right)$ security.*

Proof. We first prove the perfect privacy of the protocol. Since the polynomial used to construct the shares of the message is of degree at most $(mt + 1)$, m number of shares are transmitted on each wire and the adversary can control at most t wires, the adversary can learn at most mt shares. This is two less than what are required to interpolate the secret of the transmission and thus perfect privacy is achieved.

The protocol achieves perfect authenticity as the receiver accepts a message only when one value is correctly decoded - when $|MESSAGES| = 1$. It is easy to see that the value sent by the sender will always be correctly decoded using the shares received from the set of $(t + 1)$ honest wires. If only one value is decoded, as this corresponds to the same message transmitted by the sender, the receiver always accepts the correct message. If more than one value is decoded, the receiver cannot know which value is the correct one to accept. In this case the receiver does not accept a message but outputs \perp . Perfect authenticity is therefore achieved.

Estimate on the upper bound on the probability γ of protocol failure.

We now calculate an estimate on the *upper bound* on the probability γ with which the receiver accepts \perp and the adversary causes the protocol to fail. For this to occur, any changes that may be carried out should result in (at least) a second valid polynomial. We term as *valid* polynomials those polynomials of degree at most $(mt + 1)$ for which *all* $(mt + m)$ shares received from $(t + 1)$ wires lie on the polynomial. A valid polynomial is defined in this way because of the decoding process carried out by the receiver.

For any strategy the adversary may follow, the protocol fails when the adversary alters all or a subset of the mt shares under its control so that a valid polynomial p' - which outputs a different (to the secret) value $p'(0) \neq M^S$, is decoded by the receiver¹⁵. We now prove an *upper bound* on the probability with which any adversary strategy can cause the protocol to fail.

As the adversary learns mt shares and a polynomial of degree at most $(mt + 1)$ was used to share the message, the adversary learns two shares less than what are required to learn the

¹⁵The case when two valid polynomials - p_1 and p_2 , are decoded by the receiver and $p_1(0)=p_2(0)$ is not considered a successful attack by the adversary. This is because despite a valid second polynomial, it has not resulted in a second valid *different* value.

secret of the communication. Using its infinite computing power to look at all polynomials of degree at most $(mt + 1)$, the adversary can identify q^2 polynomials which could have created the shares seen by the adversary - where q denotes the size of the finite field \mathbb{F} .

We now assume the following scenario. Assuming changes were carried out by the adversary which led to a successful attack, we denote with \mathcal{A} the event that a successful attack was carried out by the adversary and with \mathcal{B} we denote the set of all possible wires and their corresponding shares which resulted in this attack. More specifically, \mathcal{B} is a triple $\{i, A, H\}$ with i indicating the number of honest wires (wires not controlled by the adversary) involved in the successful attack and A and H identifying the set of Adversary/Honest controlled wires involved in the successful attack.

We now analyze the following conditional probability:

$$p(\mathcal{A}) = \sum_0^{|\mathcal{B}|} p(\mathcal{A})$$

The probability of having a successful attack corresponds to guessing the encoding polynomial used in the sharing of the message. This equates to the adversary guessing the correct polynomial from a search space of q^2 possible polynomials. However, when considering the use of non-uniform message distributions, the adversary can guess the value of the secret with a relative probability. The best strategy for the adversary to follow is to guess the most likely message in the message space and in this way, the adversary knows $(mt + 1)$ shares from the encoding polynomial - reducing the search space of likely encoding polynomials to only q . The probability of the adversary guessing the correct polynomial is now $\frac{1}{q}$.

We still have not found the upper bound to a successful attack. What remains is to sum over the set of different \mathcal{B} values. To do this we need to count how many different \mathcal{B} values there are.

As the i value of \mathcal{B} indicates the number of honest wires used in a successful adversary attack, it is easy to see that $1 \leq i \leq t$. This is because each successful attack should result in a valid polynomial constructed from the shares of at least $(t + 1)$ wires and as there are only t adversary controlled wires, at least one honest wire must be included in a successful attack¹⁶.

As for a successful attack to occur all the shares from $(t + 1)$ wires are needed, assuming the adversary alters shares upon all adversary controlled wires, \mathcal{B} can take upon the following number of different values - with $C(n, k)$ denoting the number of k -combinations from a given set of n elements :

$$\sum_{i=1}^t C(t + 1, i), C(t, t + 1 - i)$$

The above sums over all the values that i can take, whilst considering the different combinations of i honest wires and the different combinations of the remaining number of adversary controlled wires so that their total is $(t + 1)$.

Using the Vandermonde convolution this can be simplified to $\binom{2t+1}{t+1} - 1$. The -1 appears as $\binom{2t+1}{t+1}$ represents all possible sets of $(t + 1)$ wires and as we are considering an adversary

¹⁶No more than t honest wires can be included in an attack as this will result in the original encoding polynomial.

attack which obviously cannot include all $(t + 1)$ honest wires, we have to remove this honest set.

Taking all of the above into account, we can now sum over all the different values of \mathcal{B} and obtain an upper bound on the probability of protocol failure to be $\gamma = \frac{\binom{2t+1}{t+1}-1}{q}$.^{17, 18}

We expand $\gamma = \frac{\binom{2t+1}{t+1}-1}{q}$ to identify if the function defining the probability of protocol failure is a negligible function.

We use Stirling's approximation for $n!$ which from [173] is described by the following:

$$\frac{n^n}{e^{n+\frac{1}{12n+1}}}\sqrt{2\pi n} < n! < \frac{n^n}{e^{n+\frac{1}{12n}}}\sqrt{2\pi n}$$

Given $\binom{2t+1}{t+1} = \frac{(2t+1)!}{t!(t+1)!}$ we use Stirling's approximation to *approximate* γ - using the Stirling's approximation upper bound for the factorial in the numerator and the Stirling's approximation lower bound for the factorials in the denominator:

$$\gamma \leq \frac{(2t+1)^{2t+1} e^{t+1+\frac{1}{12(t+1)+1}} e^{t+\frac{1}{12t+1}} \sqrt{2\pi(2t+1)}}{e^{2t+1+\frac{1}{12(2t+1)}} t^t (t+1)^{t+1} q \sqrt{2\pi(t+1)} \sqrt{2\pi t}}$$

This can be simplified to the following:

$$\gamma \leq \frac{(2t+1)^{2t+1} e^{t+1+\frac{1}{12t+13}} e^{t+\frac{1}{12t+1}} \sqrt{2\pi(2t+1)}}{e^{2t+1+\frac{1}{24t+12}} t^t (t+1)^{t+1} q \sqrt{2\pi(t+1)} \sqrt{2\pi t}}$$

The above can otherwise be written as follows:

$$\gamma \leq \frac{\sqrt{2\pi}}{2\pi} \sqrt{\frac{2t+1}{t(t+1)}} \left(\frac{2t+1}{t+1}\right)^{t+1} \left(\frac{2t+1}{t}\right)^t \left(\frac{1}{q}\right) \left(\frac{e^{t+1+\frac{1}{12t+13}} e^{t+\frac{1}{12t+1}}}{e^{2t+1+\frac{1}{24t+12}}}\right)$$

As $t \rightarrow \infty$ the term $\left(\frac{e^{t+1+\frac{1}{12t+13}} e^{t+\frac{1}{12t+1}}}{e^{2t+1+\frac{1}{24t+12}}}\right) \rightarrow 1$. This simplifies the above to the following:

$$\gamma \leq \frac{\sqrt{2\pi}}{2\pi} \sqrt{\frac{2t+1}{t(t+1)}} \left(\frac{2t+1}{t+1}\right)^{t+1} \left(\frac{2t+1}{t}\right)^t \left(\frac{1}{q}\right)$$

¹⁷The reader is reminded that this probability is so when the guess of the likely secret message carried out by the adversary is correct. For uniform message distributions or for strategies where the adversary does not guess the value of the message this probability is $\gamma = \frac{\binom{2t+1}{t+1}-1}{q^2}$. For a uniform distribution and when the secret is shared using a $mt+1$ -out-of- mn secret sharing (using an at most mt degree polynomial), the probability of failure remains $\gamma = \frac{\binom{2t+1}{t+1}-1}{q}$.

¹⁸An alternative argument for the probability of failure presented is as follows. Assuming the secret is shared using an at most $mt+1$ degree polynomial with $m=3$. Then any $mt+2$ shares define a valid at most $mt+1$ degree polynomial. We assume the adversary alters at least one share for all adversary controlled wires. The probability that any combination of $t+1$ wires (which includes at least one adversary controlled wire) with all shares lying on the same at most $mt+1$ degree polynomial is $\frac{1}{q}$. This because $mt+2$ shares interpolate an at most $mt+1$ degree polynomial p' and the probability of the $(mt+3)^{rd}$ share having the correct $p'((mt+3)^{rd} \text{ value})$ is $\frac{1}{q}$. Furthermore, as there are $\binom{2t+1}{t+1} - 1$ possible combination of $t+1$ wires - which include at least one adversary controlled wire, the probability of protocol failure is $\frac{\binom{2t+1}{t+1}-1}{q}$.

Given $\left(\frac{2t+1}{t}\right)^t \leq 4^t$ the above is equal to the following:

$$\gamma \leq \frac{\sqrt{2\pi}}{2\pi} \sqrt{\frac{2t+1}{t(t+1)}} \left(\frac{2^{t+1}4^t}{q}\right)$$

Taking the size of the finite field q to be equal to the following function of t , $q \geq 2^{4t+1}$ - where t is variable, an estimation of the above probability becomes:

$$\gamma \leq \frac{\sqrt{2\pi}}{2\pi} \sqrt{\frac{2t+1}{t(t+1)}} \left(\frac{2^{t+1}4^t}{2^{4t+1}}\right)$$

Which is simplified to:

$$\gamma \leq \frac{\sqrt{2\pi}}{2\pi} \sqrt{\frac{2t+1}{t(t+1)}} \left(\frac{1}{2^t}\right)$$

It is easy to see that the above function which describes γ is a negligible function as its value decreases faster than any positive polynomial $poly(\cdot)$.

The probability γ of the protocol failing can be made negligible when referring to security parameters if a *sufficiently large* $\mathbb{F}(q)$ is considered. This completes the security proof.

The protocol is thus a $\left(0, 0, \frac{\binom{2t+1}{t+1}-1}{q}\right)$ almost perfectly secure one-phase transmission protocol. \square

It is easy to see that the above arguments are all valid for any constant value of $m > 2$ (for a uniform message distribution $m \geq 2$ is sufficient). Additionally, the probability of protocol failure can be decreased further by employing the variant of Shamir secret sharing as used in the protocol of Section 3.2.

It should be noted that the protocol achieves the same security as the one-phase almost perfectly secure message transmission protocol presented in [107] - including similar probability of failure as described in [158] (in that as the value of t increases, so does the probability of the adversary causing the protocol to fail).

For the $\gamma = \frac{\binom{2t+1}{t+1}-1}{q}$ probability with which the protocol fails, it is easy to see that as the value of t increases, the probability with which the adversary can cause the protocol to fail (for a *fixed* value of q) also increases. When one wants to run the protocol of this section with a specific required bound on protocol availability, they will have to select an appropriate value of q so that the acceptable protocol probability of failure (as chosen by the users) is achieved within the range of values of t (or fixed value of t) under which the protocol will be executed.

3.3.3.2 Complexity Analysis

The communication complexity of the protocol is clearly $O(nm)$ as m field elements are transmitted on each wire. As m is a constant the communication complexity is $O(n)$. As only one secret is sent in the execution of the protocol, the transmission rate of the protocol is $O(n)$ which is optimal as shown in [107].

The computational complexity of the sender is polynomial as all the sender executes is a Shamir secret sharing of M^S . To do this the sender has to select $mt + 1$ random elements to choose the at most $mt + 1$ degree random polynomial p with which shares of M^S will be

created. Furthermore, to construct the mn shares the sender has to evaluate p mn times - which requires $O(n^2)$ field operations. The computational complexity of the sender is thus $O(n^2)$.

The computational complexity of the receiver on the other hand is clearly exponential. This is because in the worst case scenario the receiver has to check all possible sets of $(t + 1)$ from $(2t + 1)$ sets of wires - which is of exponential growth. To do this, the receiver has to interpolate the polynomial which passes through $mt + 2$ shares and then check the correctness of the remaining $m - 2$ shares. The polynomial interpolation (as shown in Section 3.2.2.3) when using Neville's algorithm requires a computational complexity of $O(n^2)$ (when using Gaussian elimination this complexity is $O(n^3)$). The amount of computation carried out by the receiver is thus $O(\binom{2t+1}{t+1}n^2)$ which is of exponential computational complexity.

3.4 Comparison to Previous Work and Future Work

In this section we compare the two new protocols presented in this chapter to previous one-phase almost perfectly secure message transmission protocol. We compare the protocol of Section 3.2 to the polynomial time protocol of Srinathan et al. presented in [167]. We compare the exponential time protocol presented in Section 3.3 to the exponential time protocol of Kurosawa and Suzuki presented in [107]. In both cases we argue why each protocol improves on the previous protocols.

We conclude the chapter with future work and open problems regarding almost perfectly secure one-phase protocols.

3.4.1 Comparison of Polynomial Time Protocol to Srinathan et al. Protocol

The work presented by Srinathan et al. in PODC 2008 [167] was important as it presented a polynomial time almost perfectly secure message transmission protocol which achieves the optimal transmission rate of $O(n)$ with a communication complexity of $O(n^2)$. For the protocol to achieve this transmission rate $O(n)$ messages are transmitted. Protocol 1 also has a communication complexity of $O(n^2)$. As only a single message is transmitted the transmission rate of the protocol is $O(n^2)$ and thus not optimal.

When these two protocols are compared it may seem that the Srinathan et al. protocol is the best protocol to use as more messages can be sent with the same communication complexity.

Even though this is true, if the two protocols were to be compared, it will be found that Protocol 1 transmits less field elements and requires less computation to complete than the Srinathan et al. protocol.

For comparison purposes, the Srinathan et al. protocol is briefly described below - with the relative complexities of each step identified:

Sender Actions The sender carries out the following steps:

1. A random non-zero array A of size $(t + 1) \times n$ is created. Computational complexity (n^2) .
2. Considering the column elements of array A as the coefficients of an at most t degree polynomial, for each one of these n polynomial evaluate a further t points. Extend array A to an $n \times n$ array using these evaluated points. Computational complexity (n^3) .

3. Choose n random elements - one corresponding to each wire. Considering each row of the now modified array A as an at most $n - 1$ degree polynomial, evaluate each of the n random elements for each of the n at most $n - 1$ degree polynomial. Computational complexity (n^3).
4. Execute algorithm **EXTRAND** [144] to obtain $t + 1$ encryption pads for the $t + 1$ secret messages which are then encrypted. Computational complexity (n^2).

Protocol Transmission The sender sends the following on each wire:

1. An at most $n - 1$ degree polynomial definition.
2. A random element.
3. The n evaluations of the random element over the n at most $n - 1$ degree polynomials
4. The $t + 1$ encrypted messages are broadcast over all wires.
5. The total number of field elements sent is n^2 from point 1, n from point 2, n^2 from point 3 and $n(t + 1)$ from point 4. Overall, $2n^2 + nt + 2n$ field elements are transmitted.

Receive Actions The receiver carries out the following steps:

1. Accepts using majority the correct set of encrypted shares. Computational complexity $O(n^2)$.
2. Checks the correctness of all shares (evaluates each random element for each polynomial). Computational complexity $O(n^3)$.
3. Identifies and ignores faulty wires contradicted by $t + 1$ wires. And (partially or completely) reconstructs a form of the extended array A similar to what the sender did. Computational complexity $O(n^2)$.
4. For each of the n columns of the extended array A , error detection is carried out to check if all values lie on an at most t -degree polynomial. Computational complexity $O(n^3)$ when using Neville's algorithm ($O(n^4)$ when using Gaussian elimination).
5. If no error is detected, algorithm **EXTRAND** is executed to obtain the $t + 1$ encryption pads and decrypt the secret messages. Otherwise \perp is accepted. Computational complexity $O(n^2)$.

Comparing the two protocols in terms of the number of field elements they transmit, Protocol 1 as shown in Section 3.2.2.1 transmits $n^2 + nt + 2n$ field elements which with $n = 2t + 1$ equals $6t^2 + 9t + 3$ field elements. The Srinathan et al. protocol transmits $2n^2 + nt + 2n$ field elements which with $n = 2t + 1$ equals $10t^2 + 13t + 4$ field elements¹⁹. Protocol 1 therefore transmits about a *third less* number of the field elements than what the Srinathan et al. protocol

¹⁹ $2n^2 + nt + 2n$ with $n = 2t + 1$ becomes $2(2t + 1)(2t + 1) + (2t + 1)t + 2(2t + 1) = 2(4t^2 + 4t + 1) + (2t^2 + t) + (4t + 1) = (8t^2 + 8t + 2) + (2t^2 + t) + (4t + 1) = 10t^2 + 13t + 3$

transmits (neglecting terms that are subquadratic the two protocols transmit $6t^2$ vs $10t^2$ field elements respectively)²⁰.

Comparing the two protocols in terms of the computations the sender and the receiver have to carry out, the computational requirements of Protocol 1 are lower.

For the sender of this protocol, the sender has to secret share $n + 1$ values - of which the computational requirements are equivalent to the first two steps the sender carries out for the Srinathan et al. protocol. As more steps are required in the Srinathan et al. protocol, the computational demands for the sender are greater (but of equal $O(n^3)$ computational complexity), in fact they are more than double the sender computational demands of Protocol 1.

The receiver of Protocol 1 executes schemes which are similar to Step 2 and Step 3 of the Srinathan et al. protocol receiver with similar computational demands and computational complexities. Furthermore, the receiver of Protocol 1 executes only one error detection. The receiver of the Srinathan et al. protocol though has to execute more steps and furthermore, n error detections instead of one are carried out. As identified in Section 3.2.2.3, the computational demands for error detection are $O(n^2)$ when using Neville's algorithm. When this algorithm is used, the receiver of both protocols have the same computational complexity of $O(n^3)$ but despite this, the computational demands of the Srinathan et al. protocol receiver are clearly n times greater (as n instead of one error detections have to be executed). On the other hand, if error detection was carried out using Gaussian elimination which requires $O(n^3)$ computational complexity, Protocol 1 in this case improves on the Srinathan et al. protocol by an order of n lower computational complexity for the receiver and indeed the protocol (Protocol 1 computational complexity would remain $O(n^3)$ whereas Srinathan et al. protocol would be $O(n^4)$).

Overall, Protocol 1 is a simpler protocol which transmits less field elements and requires less computation by both communicating parties. This makes Protocol 1 very useful for situations where the almost perfectly secure transmission of a *single* message is required. This is may be the case in networks built with battery powered nodes where the conservation of node battery life is of great importance in such application scenarios. As in general only a single value will need to be exchanged, it makes sense to use the simplest and most energy efficient protocol available - which due to computational and communication requirements is the protocol presented in Section 3.2. Furthermore, in multi-party computation protocols it may be the case that each participant sends a single (possibly different) message to all other participants. Using protocols which transmit multiple messages is thus not required and it is best to use more efficient protocols which transmit only a single message.

3.4.2 Comparison of Exponential Time Protocol to Kurosawa-Suzuki Protocol

Even though the idea of one-phase almost perfectly secure message transmission was first proposed in [169] the work presented by Kurosawa-Suzuki (KZ) in ICITS 2007 [107] was important as it was the first paper which formalized the idea of almost perfectly secure message transmission²¹. It also proved optimal bounds on the communication and transmission com-

²⁰This can amount to a large number of field elements for large values of t . For networks which are built using resource constrained battery powered nodes the transmission and forwarding of larger amounts of data (than are necessary) could affect the lifetime of nodes being operational.

²¹The idea of one-phase protocols which terminate with a high probability and otherwise fail was introduced in [169] but due to faults in the protocol that was presented their idea was somewhat overlooked.

plexities of such protocols.

The protocol presented in [107] achieves almost perfectly secure message transmission with exponential computation time. However, it is confined to its use only over a planar difference set. The protocol presented in Section 3.3 is similar to the KZ protocol in the sense that it meets the lower communication complexity bound and optimal transmission rate proven in [107]. Unlike the protocol of [107], Protocol 2 is not confined to specific finite fields - for example over a planar difference set, but can be used over any finite field.

To understand the reason why Protocol 2 not being confined to specific finite fields is an advantage over the KZ protocol presented in [107], one must look at their protocol in detail. We first present the definition of a planar difference as given and used in [107] then comment on the availability of such difference sets. We also comment on specifics of the KZ protocol which concern its computational complexity.

We now present the definition of a planar difference set (from [107]).

Definition 1. *A planar difference set modulo $N = l(l-1)+1$ is a set $B = \{d_0, d_1, \dots, d_{l-1}\} \subseteq \mathbb{Z}_N$ with the property that the $l(l-1)$ differences $d_i - d_j$ ($d_i \neq d_j$), when reduced modulo N , are exactly the numbers $1, 2, \dots, N-1$ in some order.*

Further to this, the KZ protocol requires $\{d_0, \dots, d_q\}$ to be a planar difference set modulo $p = q^2 + q + 1$ where both p and q are prime, $\{0, 1, \dots, q\}$ is the message space of the message transmission protocol (although this is not always necessarily the case as $\{d_0, \dots, d_q\}$ may include other elements other than the complete list of elements in $\{0, 1, \dots, q\}$) and Shamir secret sharing will be carried out over $GF(p)$.

Given the fact that both q and $p = q^2 + q + 1$ have to be prime, an analysis was carried out on the distribution of such p and q between 1 and 45 billion. Surprisingly only 1315 of these were found and these can be found online [59].

Further analysis of the primes p and q identifies that for most of primes q when assuming $\{0, 1, \dots, q\}$ as the message space of the message transmission protocol, the respective prime p requires double the number of bits to encode an element in $GF(p)$ when compared to an element in $GF(q)$ (99 per cent of the primes require at least 1.9 times more bits). As the KZ protocol uses secret sharing over $GF(p)$ but the message space is only between $\{0, 1, \dots, q\}$, double the number of bits have to be transmitted to transmit a message than are actually necessary to encode it. Because of this reason, the KZ protocol and Protocol 2 can have the same *communication cost*²².

Another advantage Protocol 2 has over the KZ protocol is in the initial setup of the protocol. The KZ protocol as stated earlier requires a planar difference set. This requires that given $N = l(l-1)+1$ one has to find a set $B = \{d_0, d_1, \dots, d_{l-1}\} \subseteq \mathbb{Z}_N$ with the property that the $l(l-1)$ differences $d_i - d_j$ ($d_i \neq d_j$), when reduced modulo N , are exactly the numbers $1, 2, \dots, N-1$ in some order. Given the sparse distribution of primes q and p as required by the

²²Protocol 2 has the advantage of being able to use any finite field thus the actual number of bits used to encode a secret message will be transmitted. However, even though the computational complexity of the two protocols is the same ($O(n)$), the exponential time protocol presented in this chapter requires for $m \geq 2$ field elements to be transmitted on each wire. Because of this, when assuming a uniform message distribution and $m = 2$ the communication cost of the two protocols (in terms of bits) is equivalent.

KZ protocol, it is apparent that as the bit length of the message space of a transmission protocol increases, the number of initial calculations required to identify set $B = \{d_0, d_1, \dots, d_{l-1}\}$ (to identify all differences $d_i - d_j (d_i \neq d_j)$) increases greatly - as N (or p) number of subtractions will have to be carried out. On the contrary, Protocol 2 beyond the choice of any finite field to use does not require any further initial steps (and computation) before execution.

Protocol 2 also performs better with regards to computational complexity. Both protocols carry out reconstruction of a message from the shares received from all $t + 1$ subsets of wires. Thus both protocols have to perform the reconstruction phase of the Shamir secret sharing scheme (as described in Section 2.6.1). The KZ protocol though has to check that for each of these subsets, the reconstructed message $y_i = d_s \in \{d_0, \dots, d_q\}$ (in such case d_s should be a single value) or $y_i \notin \{d_0, \dots, d_q\}$ (in such a case, the adversary clearly altered a share and y_i is ignored). If the receiver identifies a single d_s this is accepted as the secret message of the transmission protocol, otherwise \perp is accepted. However, checking whether $y'_i \neq d_s$ belongs in $\{d_0, \dots, d_q\}$ or not requires $\log(q)$ computation²³. This is especially the case when the elements of the planar difference set used in KZ protocol are not ordered elements of a set (and thus one cannot check that a reconstructed value belongs in the planar difference set easily). Thus the computational complexity of the KZ protocol is $O(\binom{2t+1}{t+1} n^2 \log(q))$ as opposed to the $O(\binom{2t+1}{t+1} n^2)$ complexity of Protocol 2.

Because of all the above, Protocol 2 is a more efficient protocol to use when compared to the KZ protocol.

3.4.2.1 Comparison of Exponential Time Protocol to Polynomial Time Protocols

In this subsection we also compare how the exponential time almost perfectly secure protocol proposed in this chapter compares against polynomial time almost perfectly secure protocols - such as the protocol proposed by Srinathan et al. [167] or indeed Protocol 1 which was also proposed in this chapter.

Both the exponential and polynomial time protocols use the optimal number of wires ($n = 2t + 1$) for almost perfectly secure message transmission.

The exponential time protocol requires optimal communication complexity of $O(n)$ for message transmission to occur - as opposed to the polynomial time protocols which require $O(n^2)$ communication complexity²⁴.

When considering computation, polynomial time protocols are obviously better to use. But when one wishes to use a protocol that transmits the least amount of data and where computation time on the receiver side is not an issue²⁵, then the exponential time protocol proposed in this chapter is the best one to use. This is especially the case for small values of t where the computational complexity does not amount to much.

Furthermore, the ideal almost perfectly secure one-phase transmission protocol will be one which requires the optimal number of wires ($n = 2t + 1$), requires optimal $O(n)$ communication

²³On the contrary Protocol 2 compares the reconstructed messages to the first reconstructed share and if any are different \perp is accepted. This only requires $O(1)$ computation.

²⁴The exponential time protocol also achieves optimal transmission rate - as also is the case of the Srinathan et al. [167] protocol (but not so for Protocol 1).

²⁵This may be the case in a network where a battery powered node transmits a secret to the base station which is computationally more capable.

complexity and runs in polynomial time. The exponential time protocol proposed in this chapter is the closest protocol to this ideal protocol - and whether it can be improved so that polynomial time is achieved (if this is even possible) is something that could be carried out in future work.

3.4.3 Future Work

What remains to solve in the area of almost perfectly secure message transmission protocols is to design a polynomial time protocol with optimal communication complexity of $O(n)$ - with $n = 2t + 1$. It is important to try and solve this open problem as almost perfectly secure message transmission protocols are very similar to perfectly secure message transmission schemes. This is because such protocols achieve perfect authenticity and perfect privacy of the transmitted message. Admittedly almost perfectly secure message transmission protocols have a probability of failure. But this probability can be made *as small as possible* - preferably negligible, by altering the size of the finite field - or by repeating the protocol a number of times²⁶.

Achieving this optimal open problem will lead to an efficient secure protocol with a negligible probability of failure which will simplify and optimize those protocols - such as multiparty computation, which use transmission protocols. The higher number of wires ($3t + 1$) required for the perfectly secure one-phase protocol and the extra phase in two-phase perfectly secure protocols will no longer be used - greatly simplifying such protocols.

If this open problem cannot be solved, an alternative question to answer is whether a polynomial time almost perfectly secure transmission protocol with communication complexity of $O(n)$ but with $2t + 1 \leq n < 3t + 1$ can be designed. In this case a tradeoff between polynomial time and low communication complexity against a greater number of wires is made.

An additional open problem is whether the protocol of Section 3.2 can be improved so that the almost perfectly secure transmission of a single message can occur in polynomial time with lower communication complexity - e.g. $O(n \log n)$ as opposed to the current $O(n^2)$ communication complexity.

It is important to study the above two open problems and design protocols which achieve these targets. Not only because they will improve on current protocols but because they may provide ideas to solve the initial and main open problem of almost perfectly secure one-phase transmission protocols - which is to design a polynomial time almost perfectly secure message transmission protocol with optimal $O(n)$ communication complexity.

²⁶For all one-phase almost perfectly secure message transmission protocols, there is a small probability $p \ll 1$ that the protocol will fail - meaning that the receiver will not accept a message from the message space \mathcal{M} but will instead accept \perp . With probability $1 - p$ the receiver will accept the correct message - which is the same message (in value) as sent out by the sender of the communication. By repeating one-phase almost perfectly secure protocols m number of times, the probability with which all executions will fail decreases to p^m . Based on p , selecting an appropriate value of m allows one to select an acceptable (and preferably negligible) probability of protocol failure.

Chapter 4

Multi-Phase Perfectly Secure Message Transmission Protocols

In this chapter, we consider perfectly secure message transmission protocols that require multiple phases to complete. Such protocols achieve perfectly secure message transmission when transmission of information between the two communicating parties occurs over more than one phase¹. This chapter considers two different network and recipient models - one of which has never before been considered in research of message transmission protocols.

The first model, is that of a single sender and a single receiver connected via an underlying network with undirected edges. This allows for transmission of information from a sender to a receiver and vice-versa also. As shown in [56], a necessary and sufficient condition for perfectly secure message transmission in multi phase protocols, is for the number of wires n connecting a sender and a receiver to equal $n = 2t + 1$. This is also the number of wires we consider. Additionally, it was shown that for PSMT and $n = 2t + 1$, two phases are necessary and sufficient. We also consider two phases and in Section 4.1 present a *new* protocol which is the most efficient for the perfectly secure transmission of a secret from a sender to a receiver.

This protocol is later extended to a three-phase protocol in Section 4.2 for the new network model we consider. In this model a single sender multi-recipient setting is considered. Additionally, a different network model with broadcast end channels is considered. We describe applications where such a protocol could be useful and argue how close to real world scenarios and networks the model we consider actually is. The protocol we present allows for the perfectly secure transmission of a message (later extended to multiple messages) to multiple receivers. We show how this protocol can bring about huge savings in the communication and computational complexities of the single sender.

4.1 Efficient Two-Phase Perfectly Secure Message Transmission Protocol

In this section we focus on two-phase perfectly secure message transmission. The new protocol we present achieves perfectly secure message transmission of a single message with $O(n^2)$ communication complexity and transmission rate with $O(n^3)$ computational complexity - where $n = 2t + 1$. This greatly improves on previous protocols [105, 162] which achieve

¹This does not imply that perfectly secure message transmission is not possible over one-phase. As shown in [56], when $n \geq 3t + 1$ PSMT over one phase is possible.

this with $\theta(n^3)$ communication complexity and transmission rate. This makes the protocol of this section the most efficient protocol for the perfectly secure message transmission of a *single* message between a sender and a receiver. It should be noted that for protocols where a greater number of messages need to be transmitted (such as $O(n)$ or $O(n^2)$ number of messages) the protocol presented in [105] is the best protocol to use.

We first describe the network setting of the protocol and present the main idea of the initial protocol. We proceed to describe the main techniques that will be used in the protocol. In Section 4.1.3 we formally present the initial protocol and in Section 4.1.4 present the security and complexity proof. We conclude in Section 4.1.5 where the protocol is further improved and compared to previous 2-phase perfectly secure message transmission protocols.

4.1.1 Protocol Main Idea

The network model we consider is composed of undirected point to point network edges and is the same as the first network model described in Section 2.2.1.

As the protocol is a two-phase protocol - like most two-phase protocols, in the first phase the receiver will send random elements of the finite field to the sender.

At the end of the first phase, the sender will observe the received data and identify possible errors which may have occurred in the transmission of the first phase. Different types of errors could occur and these will be outlined in Section 4.1.2.3. These errors will be sent - using broadcast, to the receiver in the transmission step of the second phase. The sender will also send via broadcast correcting information so that the random elements sent in the first phase will constitute shares of the secret shared message of the transmission.

At the end of the second phase, using the identified errors, the receiver is able to identify all wires which were active during the transmission of the first phase. Using the correcting information, the receiver is able to securely obtain the secret message of the communication. The receiver is able to do this as it can ignore shares that correspond to the identified faulty wires.

4.1.2 Main Protocol Techniques

In this section we describe the main techniques that will be used in the protocol and prove their correctness.

4.1.2.1 Broadcast

Broadcast will be used in the second phase of the protocol. When the sender broadcasts information, the sender will send the same information over all $n = 2t + 1$ wires which connect the sender and the receiver. As the adversary is able to corrupt at most t of these n wires, the receiver correctly receives the information using a majority vote (information received at least $t + 1$ times).

4.1.2.2 Transmission of Random Elements

We now describe how random elements are sent from the receiver to the sender in the first phase of the protocol². For simplicity, we first describe the transmission of one random element r .

The receiver first constructs shares of r using a $(t + 1)$ -out-of- n Shamir secret sharing scheme. The receiver thus chooses a random polynomial p of degree at most t such that $p(x) =$

²This scheme is similar to the encoding scheme of Section 3.2.2. It also resembles techniques used in [169].

$r + a_1x^1 + \dots + a_tx^t$ - where a_1, \dots, a_t are uniformly random elements of the finite field. The n shares (s_1, s_2, \dots, s_n) are computed by evaluating $p(x)$ at x_1, x_2, \dots, x_n .

The receiver proceeds to send share s_i on wire w_i ($1 \leq i \leq n$). The receiver also transmits the t coefficients (a_1, \dots, a_t) and r - which define p , across a single wire (the specific wire will be identified in the more specific explanation which follows).

In the protocol, n parallel executions of the above will be carried out. So, n random elements (r_1, \dots, r_n) will be selected. The corresponding n random polynomials (p_1, \dots, p_n) will also be constructed. For each random element, using the corresponding polynomial, n shares will be constructed. For reasons of clarity we denote as (s_{i1}, \dots, s_{in}) the n shares for the i^{th} random element and with p_i the polynomial used to construct these shares. Upon each of the n wires, n shares will be transmitted as will the definition of a single at most t degree polynomial (so at most $n + t + 1$ field elements are transmitted on each wire). The definition of polynomial p_i will be transmitted on wire w_i ($1 \leq i, j \leq n$). Share s_{ij} constructed from this polynomial will be sent on wire w_j . The communication complexity of this scheme is clearly $O(n^2)$.

This scheme is similar to that used in Protocol 1 and described in Section 3.2.2.1. In this scheme, the receiver selects n random values and secret shares them using a $(t + 1)$ -out-of- n Shamir secret sharing scheme. Following on from the analysis of Section 3.2.2.1, the computational demands of the receiver are clearly $O(n^3)$.

4.1.2.3 Error Detection and Identification of Faulty Wires

The above technique describes what will occur and what will be transmitted in the first phase of the protocol. At the end of the first phase, the sender will receive n shares and a definition of a polynomial from each wire. Using this, the sender carries out error detection as follows.

For $i, j := 1, \dots, n$ and for polynomial p_i received from wire w_i the sender considers the n shares received as the j^{th} share from each wire. The sender checks each of the shares and identifies as error shares those shares whose values do not match the definition of polynomial p_i . Share s_{ij} received from wire w_j is identified as an error share if $s_{ij} \neq p_i(x_j)$. The sender proceeds to broadcast the identified error shares to the receiver. This requires the sender to send a triple (j, i, v) to the receiver for each error share - indicating the i^{th} share of wire w_j with its value v . The i value also indicates that the share was constructed from polynomial p_i .

We now show that with this information, the receiver can identify wires that were active in the first phase of the protocol. For clarity we assume that each error share is denoted as es_{ij} - with j indicating the wire w_j and i indicating the position from which the share was received by the sender ($1 \leq i, j \leq n$). The i position of the share indicates that the share corresponds to the i^{th} polynomial received by the sender (from wire w_i).

The receiver checks the following cases to identify faulty wires:

Case 1: If the value of error share es_{ij} is different to the corresponding share sent out by the receiver in Phase 1, then wire w_j is identified as a faulty wire.

Case 2: If the value of error share es_{ij} is equal to the corresponding share sent out by the receiver in Phase 1, then wire w_i is identified as a faulty wire.

Lemma 2. *The above cases correctly identify faulty wires of the first phase.*

Proof. In Case 1, the error value received by the sender at the end of the first phase is identified to be different to the value sent by the receiver at the start of the phase. The only way this could have occurred is if the wire upon which the share was transmitted was actively controlled and the share was altered. The specific wire is thus correctly identified as a faulty wire.

In Case 2, the value of the error received by the sender at the end of the first phase is identified to be the same as that sent by the receiver. The only way that a correct value of a share could be identified as an error by the sender, is if it does not evaluate correctly to the corresponding polynomial. The only way this could occur is if the polynomial had been altered from its original form. The wire upon which the specific polynomial was sent is thus identified as a faulty wire. \square

Following on from this we also prove the following lemma.

Lemma 3. *If the adversary alters the polynomial transmitted across an adversary controlled wire, the specific wire will always be identified as faulty.*

Proof. As all polynomials are of degree at most t and as shares sent on honest wires cannot be altered, if the adversary alters a polynomial, the maximum number of shares transmitted on honest wires that can be included on the altered polynomial is t .³ This is a direct result from coding theory which states that polynomials of degree at most t can share at most t common points between them. Because of this, there will always be at least one share transmitted on an honest wire which will be identified as an error by the sender at the end of the first phase. Following on from this, the adversary controlled wire will be identified as earlier described. \square

As the sender checks the correctness of n^2 shares (by evaluating an at most t degree polynomial for each), the computational demands of the sender are $O(n^3)$. The maximum number of errors shares which the sender can identify are $O(n^2)$. This will occur if the adversary alters the value of all shares and all polynomials transmitted on adversary controlled wires. As the sender broadcast the information of the error shares, the communication complexity of this scheme is clearly $O(n^3)$.

4.1.3 Formal Protocol Description

We now formally present the protocol assuming $n = 2t + 1$ and the message of the transmission to be $M^S \in \mathcal{M} \subseteq \mathbb{F}$.

Protocol 3. *Efficient Two-Phase Protocol.*

Step 1 *The receiver does the following for $i, j := 1, \dots, n$:*

1. *The receiver selects random element r_i .*
2. *The receiver constructs a $(t + 1)$ -out-of- n Shamir secret sharing scheme of r_i using the random polynomial p_i of degree at most t , to obtain n shares $(s_{i1}, s_{i2}, \dots, s_{in})$.*

³The adversary can trivially carry out this alteration as the adversary knows the polynomial definition and thus all the shares.

3. The receiver sends polynomial p_i on wire w_i and share s_{ij} is sent on wire w_j .

Step 2 The sender does the following

1. The sender constructs a $(t + 1)$ -out-of- n Shamir secret sharing scheme of M^S to obtain n shares $(s_{m_1}, s_{m_2}, \dots, s_{m_n})$.
2. For $i := 1, \dots, n$ the sender receives polynomial p_i from wire w_i . The sender evaluates $p_i(0)$ as r_i . The sender calculates the value $d_i := r_i \oplus s_{m_i}$. These are termed as correcting information.
3. For $i := 1, \dots, n$ using the i^{th} shares received from each wire, error shares are identified. Share s_{ij} received from wire w_j is an error share if $s_{ij} \neq p_i(x_j)$.
4. The set of all identified error shares is sent to the receiver using broadcast.
5. The set of correcting information - (d_1, d_2, \dots, d_n) , is sent to the receiver using broadcast.

Step 3 The receiver does the following:

1. The receiver uses the technique of Section 4.1.2.3 to identify the set of active wires of the first phase. The set of honest wires (indicated as *HONEST*) is also constructed.
2. Using *HONEST* the receiver computes shares of the secret message M^S . This is done by computing $s_{m_{w_i}} := r_{w_i} \oplus d_{w_i}$ where $w_i \in \text{HONEST}$.
3. Using the shares computed from the previous step, the receiver uses Lagrange interpolation to obtain the secret message.

4.1.4 Security and Complexity Analysis

Theorem 4. *The above protocol achieves perfectly secure message transmission.*

Proof. We first prove the perfect privacy of the protocol. As the secret message is shared using a $(t + 1)$ -out-of- n secret sharing scheme and as the adversary is t -bounded, the adversary can learn at most t shares. This is because only t of the random elements received by the sender in Step 2 become known to the adversary. These are the random elements whose polynomial definitions were transmitted on adversary controlled wires (these may have been altered by the adversary). The remaining $(t + 1)$ random elements are not learned by the adversary. This is because all random elements are shared using a $(t + 1)$ -out-of- n secret sharing scheme and the adversary only learns t shares of each one. As a result, the adversary can only learn t shares of the secret shared message. Perfect privacy is therefore achieved.

Perfect authenticity of the protocol is achieved as the receiver only considers shares of the secret message whose corresponding random element was correctly received by the sender. This is achieved using the technique of identifying faulty wires described in Section 4.1.2.3. As shown, if the adversary alters the polynomial transmitted on an adversary controlled wire, the wire will always be identified as a faulty wire. Because of this, only correct shares are used in the reconstruction of the secret and thus perfect authenticity is achieved.

Perfect availability is achieved because the receiver always accepts a message. This because the receiver only considers correct shares when reconstructing the secret. Because of this, the adversary cannot cause the protocol to fail.

The protocol is thus a perfectly secure message transmission protocol and achieves $(0, 0, 0)$ -security. \square

We now analyze the complexity of the protocol.

As the receiver in the first phase of the protocol sends n shares and a polynomial (defined by $(t + 1)$ field elements) across each wire, the communication complexity of the first phase is $O(n^2)$.

The most expensive part of phase two in terms of communication complexity, is the broadcast of the error shares identified in Step 2 of the protocol. As there are only $(t + 1)$ honest wires, the minimum number of shares not identified as error shares by the sender will always be $(t + 1)^2$. The maximum number of error shares is $n^2 - (t + 1)^2$. This is $O(n^2)$. Therefore, for the broadcast of the error shares $O(n^3)$ communication complexity is required. The communication complexity of the second phase is thus $O(n^3)$ (this complexity will be decreased to $O(n^2)$ in the next section). As only one message is sent, the transmission rate of the protocol is $O(n^3)$.

It is easy to see that the computational costs of both sender and receiver are polynomial with a complexity of $O(n^3)$ - as identified in the earlier sections describing the schemes each party executes. The receiver carries out the secret sharing of n random elements in Step 1 - which can be done with computational complexity of $O(n^3)$. In Step 3, the receiver checks the validity of the error shares which were sent using broadcast by the sender. As there are at most $O(n^2)$ number of error shares to identify using majority, the receiver has to receive and check the correctness of $O(n^3)$ field elements. This requires a computational complexity of $O(n^3)$. The receiver also compares values sent in Step 1 to values received in Step 3 and reconstructs the secret message - both of which requires $O(n^2)$ time. Overall, the computational complexity of the receiver is $O(n^3)$.

The sender in Step 2 of the protocol has to check the validity of all received shares whether they lie on an at most t degree polynomial. As there are $O(n^2)$ shares, the computational demands for the sender are $O(n^3)$. Furthermore, the sender has to broadcast at most $O(n^2)$ number of error shares to the receiver. This also requires a computational complexity of $O(n^3)$.

Overall, the communication complexity and computational complexity of the protocol are $O(n^3)$.

4.1.5 Two-Phase PSMT with lower Communication Complexity

The protocol of Section 4.1.3 in its current form achieves $O(n^3)$ transmission rate. This is because of the $O(n^3)$ communication complexity of $COM(2)$ for the transmission of only one secret message. We now describe how to decrease the communication complexity and transmission rate of the protocol to $O(n^2)$. The most expensive step in the protocol is the broadcast of the error shares by the sender to the receiver. The protocol is optimized by using the technique of generalized broadcast which allows for the transmission of the error shares to occur with perfect authenticity and $O(n^2)$ communication complexity (as opposed to its current

$O(n^3)$ communication complexity when using broadcast).

4.1.5.1 Generalized Broadcast

The technique of generalized broadcast was first presented in [169] and later used in [4, 105]. The technique assumes the receiver knows the location of a number f of faulty wires - where $f < t$. Generalized broadcast is then able to authentically transmit up to $(f + 1)$ field elements between a sender and a receiver using codes that can correct any (remaining) errors that may occur. Generalized broadcast thus combines broadcast and error correcting codes to achieve a more efficient broadcast of the error shares. This enables the authentic transmission of $(f + 1)$ field elements between a sender and a receiver with a communication complexity of $O(n)$ instead of $O(n^2)$. This allows for the design of more efficient protocols. In the context of the protocol presented in this section, generalized broadcast is used to decrease the communication complexity of phase two from $O(n^3)$ to $O(n^2)$.

As generalized broadcast only concerns the second phase of the protocol, we assume that phase one of the protocol has completed and the sender has identified all error shares that may have occurred. We now describe the additional steps the sender carries out to transmit the error shares more efficiently.

The sender first defines the undirected graph G_e . An edge of G_e represents an error share that has been identified by the sender. The two vertices of an edge are the two wires involved with the error share - the wire from which the share was received and the wire whose polynomial definition the share corresponds to. What is important to note here is that each edge of G_e always involves at least one faulty wire. This is because two honest wires can never cause an error share as no alterations occur on honest wires. We now provide the following definition as it is necessary for understanding the remaining description of general broadcast.

Definition 2. *A matching M of a graph $G = (V, E)$ is a set of pairwise non-adjacent edges. This means that no two edges in M share a common vertex. A maximum matching of a graph G is a matching that contains the largest possible number of edges.*

The sender proceeds to compute a maximum matching M_{G_e} of G_e . From [106] which describes an algorithm to construct a maximum matching, the computational complexity for this is $O(n)$. Denoting as M_s the size of M_{G_e} , this indicates that there are M_s number of edges in M_{G_e} . It should be noted that $M_s \leq t$. Because of this, the sender is able to broadcast the maximum matching M_{G_e} of G_e to the receiver with $O(n \log n)$ communication complexity. For each edge of M_{G_e} the sender will broadcast the received value es_{ij} of the error share, the wire w_j from which the share was received and the wire w_i whose corresponding polynomial p_i the share es_{ij} is associated with. As every edge in G_e (and thus in M_{G_e}) always involves at least one faulty wire, this allows the receiver to identify M_s number of faulty wires (in the same way as described in Section 4.1.2.3). What is important for the encoding and transmission of all the error shares is that the sender is aware of this.

Suppose the sender wants to send M_s number of elements (e_1, \dots, e_{M_s}) with perfect authenticity (and without using broadcast) to the receiver. The sender finds a polynomial p of degree at most M_s such that $p(1) = e_1, \dots, p(M_s) = e_{M_s}$. The sender computes $p(M_s + i)$ and transmits the value on wire w_i where $1 \leq i \leq n$. Using Neville's algorithm, the computa-

tional complexity of the sender for this step is clearly $O(n^2)$. The receiver in turn will receive n shares of an $(M_s + 1)$ -out-of- n secret sharing scheme. This kind of code has a minimum Hamming distance of $n - M_s = 2t + 1 - M_s$. As $M_s \leq t$ this code does not allow the receiver to correct the maximum number of errors that may occur. However, as the receiver knows M_s number of faulty wires - through the broadcast of M_{G_e} , the receiver can ignore all shares received from these wires. The shortened code the receiver now considers is an $(M_s + 1)$ -out-of- $(n - M_s)$ secret sharing scheme with a minimum Hamming distance of $n - M_s - M_s = 2t + 1 - M_s - M_s = 2(t - M_s) + 1$. This kind of code allows the receiver to correct $t - M_s$ number of errors which also equates to the number of faulty wires that have yet to be identified. The use of this shortened code thus allows the receiver to correct all remaining errors that may occur, reconstruct the same polynomial p and with perfect authenticity obtain the M_s elements (e_1, \dots, e_{M_s}) . With the transmission of n elements (one share per wire) the sender is able to communicate M_s elements to the receiver with perfect authenticity.

Following on from the above, as the size of the maximum matching is M_s , this means that $2M_s$ vertices (which correspond to wires) appear in M_{G_e} and G_e . As at most n error shares can be associated with each wire, the number of error shares that will need to be transmitted to the receiver are at most $2M_s n$. As shown, M_s elements can be transmitted using n elements and as there are at most $2M_s n$ error shares to transmit, this can be carried out in $2n$ independent executions of the generalized broadcast method.

All of the error shares can therefore be transmitted from the sender to the receiver with perfect authenticity and $O(n^2)$ communication complexity. Because of this, the communication complexity of the second phase of the protocol is now reduced from $O(n^3)$ to $O(n^2)$.

4.1.5.2 Lower Communication Complexity of Protocol

Apart from the use of generalized broadcast for the transmission of the error shares from sender to receiver with perfect authenticity, everything else in the protocol remains the same - including the security proof. In the second phase of the protocol the sender uses generalized broadcast to transmit the error shares and broadcasts the n correcting information to allow for secret message recovery on the receiver side. The second phase communication complexity - similar to the first phase, is now $O(n^2)$ and as only one message is transmitted, the transmission rate of the protocol is also $O(n^2)$. The computational complexity of the receiver increases to $O(n^4 \log q)$. This is because $O(n)$ instances of generalised broadcast will be carried out - with each instance authentically transmitting $O(n)$ field elements of the set of $O(n^2)$ field elements which need to be transmitted via broadcast. Each instance may require error correction to be carried out by the receiver - with $O(n^3 \log q)$ computational complexity for each error correction. The computational complexity of the sender remains at $O(n^3)$.

Previous efficient two-phase perfectly secure message transmission protocols included [162] and the protocol of Section 4 of [105]. These have a communication complexity of $O(n^3)$. This makes the presented protocol the most efficient 2-phase polynomial perfectly secure transmission protocol for the transmission of a *single* message in the literature. The protocol presented in Section 6 of [105] is another efficient 2-phase PSMT protocol which can transmit multiple ($O(t^2)$) messages with a communication complexity of $O(n^3)$. Despite this - as mentioned in Chapter 3, for applications where a single secret needs to be securely

transmitted (as opposed to multiple secrets), the protocol of this section is the best one to use - as a lower communication complexity ($O(n^2)$ vs $O(n^3)$) is required.

4.2 Three-Phase Multi-Recipient Perfectly Secure Message Transmission Protocol

In this section, the two-phase protocol of the previous section is extended to a three-phase protocol and used in the context of a new and different network and recipient model - which for the first time is considered in PSMT protocols.

Despite this, the main idea of the protocol remains the same. We consider a single sender multi-receiver setting and a network model with broadcast end channels - as described in Section 2.2.2. In this model, a single sender needs to securely transmit with perfect security a message to multiple receivers. This is later extended to allow the sender to securely transmit multiple messages - when certain recipient conditions apply. This allows for all receivers to securely receive messages transmitted by the sender. Huge savings in the communication and computational complexities of the sender are achieved in this model, when compared to multiple executions of single receiver protocols.

Overall, the main benefit of the network model and protocol we present is the lower amount of computation carried out by a single sender and lower amount of data (bits) per receiver which need to be transmitted by the sender. The amount of data the sender needs to transmit in the protocol to be presented is comparable to the data the sender transmits in the protocol of the previous section. But when a single sender wants to communicate with $O(\rho)$ number of multiple receivers (where $\rho = \binom{n}{n'}$ with $n > n'$), by executing the protocol presented in this section, the communication complexity of the sender is in the order of $O(\rho)$ lower when compared to multiple executions of other single sender-single receiver protocols (as will be shown in Section 4.2.5).

We describe applications where such a protocol could be useful and argue how close to real world scenarios and networks the model we consider is.

We primarily give details of the network environment considered and in Section 4.2.2 discuss the relevance of this network model to networks which exist in the real world. We also provide examples of practical based scenarios where this kind of protocol - in which a single sender securely transmits messages to multiple receivers, could be useful. In Section 4.2.3, we give details about the protocol and its main idea. We formally present the protocol in Section 4.2.4 and then present its security, complexity and comparison analysis. In Section 4.2.6 we extend the protocol so that for certain conditions, a greater number of secrets can be transmitted.

4.2.1 Environment of Message Transmission Protocol

The network model considered in this section is the same as that described in Section 2.2.2. This consists of a single receiver, connected to forwarding nodes which connect receivers using *LAN multicast* channels. The t -threshold bounded active adversary is assumed to be able to take control of forwarding nodes *only* in a network and thus all receivers are free from any adversary presence. Because of this, the adversary has the capability to control t wires connecting the sender to the network which in turn means that the adversary controls t LAN multicast channels

which connect receivers to the network.

We consider the presence of n number of wires, n forwarding nodes and n LAN multicast channels available in a network - assuming that $n = O(t)$. Forwarding node F_i services the i^{th} LAN multicast channel and delivers data to all receivers who access the particular LAN multicast channel. As in all secure message transmission schemes considered in the thesis, receivers are connected to the network using disjoint network paths. We denote with $n' \leq n$ the number of LAN multicast channels each receiver has access to.

We provide the following definition to differentiate between the different receivers in this network model.

Definition 3. We define as non-similar receivers, receivers who are connected to the network with different subsets of LAN multicast channels.

Similar receivers are thus identified as receivers connected to the network with the same subset of LAN multicast channels. This implies that similar/non-similar receivers are connected to the same or different subset of forwarding nodes respectively. When we refer to receivers we will be referring to non-similar receivers. We thus generalize the view of multiple similar receivers to that of a single receiver. It is easy to see that the number ρ of non-similar receivers in such a network can be up to $\rho = \binom{n}{n'}$. In general, ρ represents all possible subsets of n' from the n LAN multicast channels available in a network.

Note that if $n = n'$, then there can only be one non-similar receiver. As we are considering a multi-receiver scenario, we assume $n > n'$. The protocol we present thus securely transmit a secret message from a single sender to $\rho > 1$ non-similar receivers. It is easy to see that $\rho = O((n')^{n-n'})$.

As an example of a network let $t = 1$, $n' = 3t + 1 = 4$ and $n = 5$. We denote the five forwarding nodes present in a network as $F_1, \dots, F_5 \in F$. Then there can be at most $\rho = \binom{5}{4} = 5$ non-similar receivers $R_1, \dots, R_5 \in R$ such that for each $1 \leq i \leq 5$, R_i is connected to all forwarding nodes except F_i .

4.2.2 Relevance and Practical Applications

In contrast to modern communication, perfectly secure message transmission (PSMT) is expensive since multiple paths are required. Despite this, we outline in what type of real world practical applications PSMT might be needed in a broadcast environment. The typical scenario is broadcast in case of emergency, for example after a natural or man-made disaster. Government authorities might need to give recommendations to the population, which are natural to broadcast. Moreover, authorities may need to mobilize troops or civilian authorities after such a disaster. If this is the case, it is important that this communication should be carried out in a secure (authentic and secret) manner. Indeed, after the earthquake in Haiti [13], lack of immediate deployment of an emergency law enforcement force created an atmosphere which allowed for uncontrolled looting. After such a disaster, several of the communication media may have been accidentally or even maliciously destroyed by adversaries who want to take advantage of the chaos.

In real world emergency scenarios, such as terrorist attacks, before/during/after war, natural disaster events, or in general any type of practical mobilization scenario, a single central authority - such as a central government of a country, will need to communicate information to a number of people. This number could possibly be very large - depending on the application. Nonetheless, this communication should possibly be secure against any disruptions, eavesdropping or alterations.

As there is only one central entity transmitting information, it is important that the transmission of the information to all parties who need to receive it be carried out in the most efficient manner. In general, this means that the amount of information the single central authority should send, receive and process, be kept to a minimum. This will aid in the speedy and easy distribution of information to all receiving parties.

In this section, we assume a network setting where the sender is connected to the network with point-to-point channels. On the other hand, receivers are connected to the network via broadcast channels. In between these two types of parties, are forwarding nodes which broadcast information sent by the sender to the receivers.

The LAN multicast *channels* considered in this network model are comparable to the network channels of *some* networks which exist in the real world. The Internet for example is composed of Internet Service Provider (ISP) networks. Some of these networks are mainly built up of a core network (often using optical fiber). The rest of the network can be described as follows:

- “Provider edge nodes” are the terminal nodes of the core network. These provider edge nodes connect many home users to the network. In one approach, the Internet is provided using telephone wires [71].
- The provider edge nodes connect the customers to the Internet with a LAN, using broadcast. This scenario occurs when the Internet provider is, for example, a TV cable company which uses broadcast to deliver its services to its customers⁴. Customers may of course have access to more than one such channel.

Other examples in this category include radio networks where radio antennae are connected to networks using point to point links, but transmit information using broadcast over different radio frequencies. Practical applications of this scenario include commercial TV and radio.

The cellular network, which in general uses the Global System for Mobile Communications (GSM) standard (another similar standard include the CDMA2000 family of 3G mobile technology standards), is similar to the network model considered in this section. In such a network, cellular antennae are connected to a fixed wired network using point to point links, but transmit information using radio waves - a broadcast medium. At any one time, a cellular device (for a example a mobile phone) is within the range of a number of antennae - although one in general is used to handle the service of the device. In such networks, cellular devices serviced by the same antennae could all receive the same information from the same antennae

⁴Other technologies which do not use broadcast are available for companies to use - but in this chapter we focus on broadcast technologies to reflect the similarities of our proposed network model.

- such as a text from a cellular network provider which provides local (to the location of the antennae) commercial information.

Modeling the receivers of the network scenario to users who own a cell phone device, the single sender of the scenario as a central sender (government agency for example) and forwarding nodes with cellular antennae one can see that the cellular network and the network model considered in this section have similar characteristics to each other. In the mobile phone network, non-similar receivers can be modeled as cell phones users who are within the reach of a different subset of cellular antennae. In the next sub-section we describe who a cell phone user can access multiple channels at the same time.

4.2.2.1 Accessing Multiple Broadcast Channels

As seen from the above examples, various network technologies exist in practise which connect end users to a network using a broadcast channel. Such channels are comparable to the channels we consider in our network model.

The difference between the model we consider and such practical networks is that mainly in practise, end users have access to only one such broadcast channel. In this subsection we provide examples of how end users in practical networks can possibly access more than one broadcast channel.

In the cellular network, end users in possession of more than one cell phone (SIM card) can access the cellular network using more than one cellular antenna - provided each of their phones are locked onto the networks of different providers.

More practical for cellular network end users are cell phones which can hold more than one SIM card. Examples of such phones are the Samsung B7722 [146], the Nokia C2-00 [138] amongst others. Such cell phones allow an end user to have access to two different cellular network providers and can change between different networks with a touch of a button.

Some cell phones have more advanced capabilities than the two given examples. The HTC Desire VC [93] not only accepts two SIM cards, but one can be a CDMA SIM card whilst the other being a GSM SIM card. There are even some cell phones which accept four different SIM cards - one such example being [66]. Furthermore, the HTC Desire VC is also a smartphone (other examples of dual SIM smartphones include [159, 188] amongst others) which runs the (programmable) Android operating system. Based on the above examples, one can see the possibility of a smart phone application being developed which could use data from two (or more) different cell phone provider networks to implement the protocol to be presented in the next section.

One example highlighting the required presence of secure message transmission amongst a single sender and multiple receivers was during President Obama's visit to Oakland July 2012 [160]. During this visit, a major portion of Oakland's police radio system failed leaving many of the 100 officers assigned to handle presidential security unable to communicate as protesters roamed the streets. This was considered to be a serious issue, as the radio failure could have possibly put police officers and even President Obama at risk. In this kind of situation, a practical implementation of perfect secure message transmission between a single sender and multiple receivers in a similar manner as described above with smart phones would have provided an alternative method of communication. If a diversity of networks were used in such

an implementation, then a failure in a threshold of these (as will be described in the description of the protocol in the next section) will not only have allowed for communication between the president's security detail (single sender) and the police officers (multiple receivers) to continue in a reliable manner, but additionally (due to the nature of the police force operations) would have been secret.

Similarly, subscribers to more than one broadcast service provider (such as commercial television, Sky TV, Internet TV) can also have access to more than one of such broadcast channels.

Alternatively, the network model we consider is similar to communication networks which can be implemented on the Internet. Using blogs with restricted access (i.e. blogs with a limited number of users which require a username and password for access) one can model the LAN multicast channels of our network. Assuming users (receivers) have access to a number of such blogs, this simulates receivers having access to more than one LAN multicast channel of our network model. Non-similar receivers of this communication network can be modeled by users which have access (login details) to different sets of these blogs. If a single sender wants to securely transmit a message to multiple receivers, this sender can carry this out using the protocol presented in the next section. In such a scenario the adversary (as in our network model), will be able to corrupt a number of the blogs (broadcast end channels) which connect the receivers to the communication network. This corruption can be carried by an attacker on the Internet by hacking a number of these blogs.

Other examples which are similar to communication networks include mailing lists - where each receiver is a member of a mailing list, can be a member of more than one mailing list and all members of a mailing list receive the same information.

Obviously the adversary model for the above two examples is different which assumes the adversary can take control of communication methods instead of network nodes.

4.2.3 Protocol Specifics and Main Idea

We now give details of the protocol to be presented in the next section.

For simplicity we let $n \geq n' = 2t + 1$ which is the optimally resilient case and assume that $n = cn$ - where c is a constant. The protocol thus transmits a single message to at most $\rho = \binom{n}{2t+1}$ non-similar receivers with perfect security. We compare the multi-receiver protocol to multiple executions of the 3-phase PSMT protocol with a single receiver [105, Section 3], and show that the protocol leads to great savings. In Section 4.2.6 we adapt the protocol slightly so that one can transmit multiple messages with similar communication complexity.

The main idea of the protocol is similar to that of the protocol presented in the previous section. The protocol also uses the same techniques presented earlier.

In the first phase of the protocol, the sender will send random elements drawn from the finite field to the receivers. The encoding and transmission of these random elements is carried out in the same way as the protocol in the previous section.

At the end of the first phase, the receivers will observe the received data and identify possible errors that may have occurred in the transmission of the first phase. Different types of errors may have occurred and these are the same as outlined in Section 4.1.2.2. These errors

will be sent by the receivers using broadcast to the forwarding nodes. Forwarding nodes will in turn forward this information to the sender.

Using the identified errors, the sender is able to identify all wires which were active during the transmission of the first phase and will inform all receivers using broadcast. The sender will also compute and send using broadcast, correcting information so that the random elements sent in the first phase will constitute shares of a secret message M^S .

Using the correcting information, the receivers are then able to securely obtain the secret message as they can ignore shares which correspond to identified faulty wires.

4.2.4 Formal Protocol Description

We now formally present the protocol, in which we let $M^S \in \mathcal{M}$ be the message of the transmission.

Protocol 4. *Three-Phase Multi-Recipient Protocol.*

Phase 1 *The sender does the following for $1 \leq i, j \leq n$:*

1. *The sender selects n random elements $r_1, \dots, r_n \in \mathbb{F}$.*
2. *The sender constructs a $(t+1)$ -out-of- n secret sharing scheme of r_i using a random polynomial p_i of degree at most t , to obtain n shares $(s_{i1}, s_{i2}, \dots, s_{in})$.*
3. *The sender sends polynomial p_i on wire w_i and share s_{ij} on wire w_j .*
4. *Forwarding nodes forward any information they receive to all receivers connected to their LAN multicast channel.*

Phase 2 *Each receiver does the following:*

1. *We denote as $WIRE S_{R_i}$ the set of wires with which receiver R_i ($1 \leq i \leq \rho$) is connected to the sender. This in effect indicates the forwarding nodes from which receiver R_i receives information.*
2. *From each wire $w_k \in WIRE S_{R_i}$, R_i receives polynomial p_k . The receiver evaluates $p_k(0)$ as r_k .*
3. *For wires $w_k, w_j \in WIRE S_{R_i}$ using the k^{th} shares received from each wire, error shares are identified. Share s_{kj} received from wire w_j is an error share if $s_{kj} \neq p_k(x_j)$.*
4. *The set of all identified error shares is sent to the sender using broadcast.*
5. *For $1 \leq i \leq n$, each forwarding node F_i receives a set of error shares from each receiver that it is connected to. Forwarding node F_i performs data compression (see Section 4.2.4.1) and sends the error shares to the sender via wire w_i .*

Phase 3 *The sender does the following:*

1. *The sender uses the technique of Section 4.1.2.3 to identify the set of active wires of Phase 1. The set of honest wires – indicated as $HONEST$, is also constructed and sent to all receivers using broadcast.*

2. The sender constructs a $(t+1)$ -out-of- n secret sharing scheme of the secret message M^S to obtain n shares $(s_{m_1}, s_{m_2}, \dots, s_{m_n})$.
3. The sender calculates the value $d_i := r_i \oplus s_{m_i}$ for all wires $w_i \in \text{HONEST}$. These are termed correcting information and are sent to all receivers using broadcast.

Message reconstruction Each receiver does the following:

1. Using *HONEST* each receiver computes shares of the secret message M^S . This is done by computing $s_{m_i} := r_i \oplus d_i$ where $w_i \in \text{HONEST}$. Each receiver uses only the $w_i \in \text{HONEST}$ to which they are connected to.
2. Using the computed shares from the step above, each receiver interpolates and obtains the secret message.

4.2.4.1 Forwarding Node Data Compression

We now describe a scheme which could be carried out by the forwarding nodes to improve the efficiency of the above protocol. At the end of Phase 2, receivers broadcast error shares to the sender. As the number of receivers can be very large ($\rho = O((n')^{n-n'})$), this could result in huge amounts of data transmitted toward the sender.

The incoming communication cost of the sender can be decreased if each forwarding node sends each error share it receives only once. Error shares which may occur in the transmission of random elements are identified as a triple (j, i, v) as described in Section 4.1.2.3. If for example two receivers R_1 and R_2 are connected to a forwarding node $F_{i'}$, and both R_1 and R_2 send (j', i', v') as information of an error share⁵, then the forwarding node $F_{i'}$ only needs to forward this information to the sender once.

4.2.5 Security, Complexity and Comparison Analysis

Theorem 5. *The above protocol achieves perfectly secure message transmission of M^S for up to $\rho = \binom{n}{2t+1}$ non-similar receivers.*

Proof. (Similar to proof of Theorem 4) We first prove the perfect privacy of the protocol. As the secret message is shared using a $(t+1)$ -out-of- n Shamir secret sharing scheme and as the adversary is t -bounded, the adversary can only learn at most t shares. This is because only t of the random elements received by receivers in Step 2 are learned by the adversary. These are the random elements whose polynomial definitions (used for the secret sharing of random elements) were transmitted on adversary controlled wires. None of the remaining $(n-t)$ random elements are learned by the adversary. This is because these random elements are shared using a $(t+1)$ -out-of- n secret sharing scheme and the adversary only learns at most t shares of each one. Therefore, the adversary can only learn at most t shares of the secret shared message. Perfect privacy is therefore achieved.

The protocol achieves perfect authenticity as each receiver only considers the shares of the secret message whose corresponding random element was correctly received in Phase 1 –

⁵Receivers R_1 and R_2 will send the same information (j', i', v') about an error share if both are connected to the LAN multicast channels (or wires) $w_{i'}$ and $w_{j'}$. This implies that they are both connected to forwarding nodes $F_{i'}$ and $F_{j'}$.

as indicated by the sender. This is achieved using the technique of identifying faulty wires described in Section 4.1.2.3. As shown, if the adversary alters the polynomial transmitted on an adversary controlled wire, the wire will always be identified as a faulty wire. Additionally, the set of all faulty wires is identified by the broadcast of *HONEST*. Because of this, only correct shares are used in the reconstruction of the secret by receivers and thus perfect authenticity is achieved.

It is easy to see that the protocol achieves perfect availability as the adversary cannot make the protocol fail in any way. This because receivers only consider correct shares when reconstructing the secret message.

The protocol is thus a perfectly secure message transmission protocol achieving $(0, 0, 0)$ -security. \square

We now analyze the complexity of the protocol. In Phase 1, the sender transmits n shares and a polynomial (defined by $(t + 1)$ field elements) across each wire - thus the communication complexity is $O(n^2)$.

The most expensive part of Phase 2, is the broadcast of the error shares identified in Step 2 of the protocol. The maximum number of error shares that a receiver can transmit is $O(n^2)$. This is because each receiver receives $O(n^2)$ shares (as shown in Section 4.1.2.3). As all error shares are broadcast, the communication complexity for each receiver in Phase 2 of the protocol is $O(n^3)$.

Given that there are ρ receivers each transmitting the same information on all n' channels they are connected to (to implement broadcast of error shares), the incoming communication complexity of the sender at the end of Phase 2 of the protocol will be in the order of $O(n^3\rho)$ if data compression (Section 4.2.4.1) was *not* carried out by forwarding nodes. Using the data compression scheme, forwarding nodes do not send the same information more than once. Therefore the field elements that each forwarding node sends in this phase is $O(n^2)$. As a result, the incoming communication complexity of the sender at the end of Phase 2 decreases by an order of $O(\rho)$ to a more manageable and practical complexity of $O(n^3)$.

In Phase 3 the sender broadcasts the set of honest wires (at most n) and also broadcasts n correcting information. The communication complexity of the sender in Phase 3 is thus $O(n^2)$.

The computational complexities of the sender and the receivers are similar to those of Protocol 3.

In Phase 1 the sender has to secret share n random elements using a $(t + 1)$ -out-of- n secret sharing scheme. The computational complexity of this is $O(n^3)$ as described in the previous section. In Phase 3 the sender has to receive and check the correctness of $O(n^3)$ field elements which are the at most $O(n^2)$ number of error shares identified by the receivers. This requires a computational complexity of $O(n^3)$.

In Phase 2 receivers have to check the correctness of the $O(n^2)$ shares they receive and broadcast the at most $O(n^2)$ error shares they may identify back to the sender. Both these processes require a computational complexity of $O(n^3)$. The most expensive steps of Phase 4 for receivers are to receive the broadcast set of *HONEST* wires and reconstruct the secret - both of which require a computational complexity of $O(n^2)$.

The computational complexity of the protocol is thus $O(n^3)$.

In a network setting similar to the one considered in Section 4.1, to securely transmit a secret message to ρ different receivers, a sender will have to perform multiple executions of multi-phase PSMT protocols - such as the three-phase protocol of [105, Section 3] or the two-phase protocols of [105, Section 4] or [162].

Considering the three-phase protocol of [105, Section 3], for ρ receivers the sender will have to execute this protocol ρ number of times. For the transmission of a secret to ρ different receivers the overall communication complexity for each phase of the [105, Section 3] protocol will increase by an order of ρ . The overall communication complexity for Phase 1 will be $(n^2\rho)$, for Phase 2 will be $(n^3\rho)$ and for Phase 3 will be $(n^2\rho)$.

Considering the network model of this section, significant savings over multiple executions of the [105, Section 3] protocol can clearly be obtained for the sender of the transmission⁶.

Primarily, it is easy to see that the *computational* requirements of the sender are much lower when the network model we consider is available and the protocol presented in this section is used. This is because the sender executes a single algorithm, as opposed to multiple executions of a protocol.

When analyzing the outgoing and incoming communication complexity of the sender⁷ is when the savings achieved can be identified. By executing the protocol presented in this section, both the outgoing and incoming communication complexity of the sender is in the order of $O(\rho)$ lower when compared to multiple executions of other protocols. This is of particular importance in a central single sender emergency scenario.

The network scenario also requires a network with a lower number of wires. It does however require that the network is specially built with LAN multicast capability - but as discussed earlier, communication networks with similar architectures exist in practise.

4.2.6 Transmitting More Secret Messages

The protocol of the previous section in its current form only allows for the secure transmission of a single message. As in [105], we now show how the protocol presented in Section 4.2.4 can be transformed so that the sender can securely send $(n - t)$ messages to all receivers. It should be noted that this can only occur when the number of non-similar receivers ρ in a network is at its maximum capacity - when $\rho = \binom{n}{n'}$.

4.2.6.1 Transmitting More Secret Messages Using Privacy Amplification

To send a greater number of secret messages, the protocol presented in Section 4.2.4 remains the same in the first two phases. Given that the adversary does not learn random elements whose polynomial definitions are sent on honest wires, use of this in combination with privacy amplification (see Section 2.7) allows the sender to send $(n - t)$ messages securely to all receivers. The third phase and message reconstruction step of the protocol now both change as follows:

Phase 3 The sender does the following:

⁶The communication and computational complexities for each receiver are equivalent.

⁷We define as outgoing communication complexity of the sender the communication complexity of the data the sender sends. Similarly, incoming communication complexity is the complexity of the data the sender receives.

1. The sender uses the technique of Section 4.1.2.3 to identify the set of active wires of Phase 1. The set of honest wires – indicated as *HONEST*, is also constructed and broadcast to all receivers.
2. Denoting as $h = |HONEST|$, the sender constructs a $h - 1$ degree polynomial p_h , with points (i, r_i) for wires $w_i \in HONEST$.
3. The sender calculates the next $n - t$ points of polynomial p_h by evaluating p_h at $p_h(n + 1), \dots, p_h(n + n - t)$ to obtain values d_1, \dots, d_{n-t} .
4. The sender encrypts secret message m_j ($1 \leq j \leq n - t$) by carrying out $c_j = m_j \oplus d_j$ and using broadcast transmits c_j to the receivers.

Message reconstruction Each receiver does the following:

1. Using *HONEST*, each receiver computes all r_i for wires $w_i \in HONEST$ - the way this is done is described in Section 4.2.6.2.
2. Each receiver reconstructs polynomial p_h and calculates values d_1, \dots, d_{n-t} in the same manner as carried out by the sender in Step 2 and Step 3 of Phase 3 (shown above).
3. Each receiver decrypts secret message m_j ($1 \leq j \leq n - t$) by carrying out $m_j = c_j \oplus d_j$.

4.2.6.2 How Receivers Obtain All Random Elements

We now clarify a note of the protocol described in the previous subsection. For the receivers to obtain the secret messages securely, all receivers need to know the h random elements to construct polynomial p_h – which is used for the secure transmission of $(n - t)$ messages as described in Section 4.2.6.1.

The problem is that each receiver only receives $(2t + 1)$ polynomial definitions as described in the protocol of Section 4.2.4. For all receivers to obtain all secret messages, receivers will have to obtain random values whose polynomial definition they did not receive. This can easily be done as all receivers receive $(2t + 1)$ shares of each secret shared random value. All receivers also know the identities of all faulty wires – as these are globally broadcast by the sender in phase three. By ignoring all shares received from faulty wires, correct random values whose polynomial definitions were not received could be obtained. Overall, receivers obtain all random values to obtain p_h , by ignoring random values which correspond to faulty wires.

The above is correct because LAN multicast channels are used. This ensures that receivers connected to the same forwarding node receive data in the same form. Because of this, all receivers which receive error shares will identify them and correctly broadcast these back to the sender. As broadcast is used - using $(2t + 1)$ wires, this ensures that the sender will correctly identify the set of error wires and broadcast the set to the receivers. Additionally, the fact that all receivers are connected to the sender by $n' \geq 2t + 1$ different wires, guarantees that there are at least $(t + 1)$ correct shares of random elements – enough to reconstruct all random elements correctly.

4.2.6.3 Keeping the Communication Complexity of Receivers the Same

The above alteration of the protocol allows for $(n - t)$ messages to be transmitted to receivers from the sender. If a greater number $m \geq n - t$ number of messages need to be transmitted, the protocol will have to be repeated $\lceil \frac{m}{n-t} \rceil$ number of times. We now describe a scheme which allows for the identification of error wires with the same communication complexity - from the receivers viewpoint as the protocol of Section 4.2.4. It also allows for the incoming communication complexity for the sender at the end of Phase 2 to remain the same. This scheme only regards Step 4 of Phase 2 for the protocol of Section 4.2.4.

Inconsistencies which can occur in the transmission of random elements are identified as a triple (j, i, v) as described in Section 4.1.2.3. The (j, i) values identify the two wires related to the inconsistency.

Despite the fact that many random elements are sent, if more than one inconsistency is found for the same (j, i) value, there is no need for all inconsistencies to be sent to the sender. This is because only one - any one, of the inconsistencies will identify the same faulty wire(s). Because of this, it is sufficient for receivers to send only one inconsistency for every (j, i) wire-pair.

For the above, a default inconsistency (for example the first one which occurs) should be transmitted by all receivers. This will prevent an increase in the incoming communication complexity for the sender after the forwarding node data compression occurs. Additionally, from a receiver's viewpoint the communication complexity remains the same - as analyzed in the previous section, when many secret messages are transmitted.

4.3 Future Work

We conclude this chapter with a discussion of open problems and future work that could be carried out in multi-phase protocols.

As shown in [169] the optimal communication complexity for 2-phase perfectly secure message transmission protocols is $O(n)$. At the moment, the most efficient protocol is that presented in Section 4.1 which has a communication complexity of $O(n^2)$ for both phases. Even though it seems somewhat unlikely that both phases of a two-phase protocol could achieve the optimal $O(n)$ communication complexity for both phases, it would be a good improvement if just one of these phases achieves this optimal complexity. It seems that the first phase is most likely to achieve this limit - if it is possible.

Alternatively, decreasing the communication complexity of either or both phases below $O(n^2)$ to $O(n \log n)$ for example would also be an obvious improvement.

In [105] the authors presented the first polynomial time protocol to achieve the optimal $O(n)$ transmission rate with a communication complexity of $O(n^3)$ for both phases. Even though achieving $O(n)$ communication complexity for both phases of a two-phase protocol will also achieve this optimal transmission rate, designing such a protocol seems a very difficult problem to solve. One could thus alternatively try and improve the work of [105] by decreasing the communication complexity of either phase from $O(n^3)$ to a lower complexity - such as $O(n^2)$. If this is achieved, this would be a great improvement and a considerable step towards designing a protocol which achieves optimal communication complexity.

With regards to multi-recipient protocols, work that could be carried out could include trying to design more efficient protocols with regards to both communication and computational complexities. As the network model considered for this protocol is similar to networks which exist in the real world, and as described in Section 4.2.2 such protocols can have a wide range of practical real-world applications, it is also important for such protocols to be deployed and readily available in such networks. Various historical incidents as described in Section 4.2.2 emphasize the importance and necessity for the availability of such protocols. It is thus of great importance that industry - mainly ISPs and other communication service providers (telephone, cellular), be convinced that such kinds of protocols are important and necessary so that they can be available for use - wherever and whenever needed. The importance of this, is discussed further in Section 7.2.

Chapter 5

Perfectly Secure Message Transmission With Human Computation

In this chapter we present a *new* scope of research - that of perfectly secure message transmission with human computation. This is not solely a different and new aspect to perfectly secure message transmission protocols, as at the same time it introduces a new research area in the field of (secure) human computation.

For such protocols, one of the two communicating parties - either the sender or the receiver, in a message transmission protocol is assumed to be human and the other party is a computational device - any form of Turing machine. The human participant on the other hand is at a computational disadvantage, as it is assumed that a computational device is unavailable to them - or ones that they have access to cannot be trusted. Because of this, any message transmission protocol that needs to be executed should not only be secure, but should also be computationally efficient and computationally simple so that it can be executed by the human party. The computational simplicity in this type of computation limits the difficulty of such solutions to trivial maths such as addition over an abelian group. This is because such math seems easy and natural for a human party to carry out without much effort.

Taking the above into consideration, PSMT protocols with human computation are proposed using two different constructions. These have been designed to be secure and usable - so as to be easy and accurate when human parties use them. One of these constructions is based on mod10 arithmetic while the second construction - identified by my supervisor Professor Yvo Desmedt, is based on using permutations to perform such modulo addition. The constructions are based on experiments with human participants.

This chapter is structured as follows. In Section 5.1 we describe the motivation behind the research of PSMT with human computation. We also outline the field of secure human computation with examples where such computation is used in the design of secure systems. In Section 5.2 we provide some preliminary background on set systems. These are the key combinatorial structures used in the construction of one-phase and two-phase mod10 arithmetic PSMT protocols with a human receiver - which are presented in Section 5.3 and Section 5.4 respectively. In Section 5.5 we describe how alternative constructions for the one and two-phase protocols can be used for small values of t . In Section 5.6, a PSMT protocol with human computation designed using our second construction is presented. In Section 5.7 we assess the way human participants used a variant of the proposed protocols through the results of experimental

evaluation that was carried out. In Section 5.8 the proposed protocols are compared to other PSMT protocols proposed in previous research with regards to their communication and computational requirements. Future work that could be carried out in the context of this chapter is described in Section 5.9.

5.1 Motivation for Perfectly Secure Message Transmission with Human Computation

The research described in this chapter was mainly motivated by work carried out in the design of an electronic code voting scheme. Despite this, such research can also find applications in other practical scenarios as we describe in what follows.

Nearly all of PSMT protocols proposed in various research papers consider the sender and receiver of a communication to be a Turing machine capable of complex mathematical operations such as Lagrange interpolation for the reconstruction of shared secrets [163], Reed Solomon error correction techniques [21] amongst others.

But what if one of the two communicating parties is not capable of such operations? What if the receiver of the communication is a human who cannot execute such computationally intensive and hard (for humans) to execute operations? This could arise when a computational device is not available or indeed cannot be trusted.

A computational device may not be trusted when perhaps it is infected with some form of malware. As seen in [14], when computer systems were infected by malware, key facilities had to be taken offline to prevent any problems that could arise from this. When such a situation arises, it is important that operations of a plant still continue. To allow for this, it is important that human operatives continue with key required operations in a correct and precise manner.

Alternatively, computational devices may not be trusted as a result of malfunctions. In the nuclear disaster of the Fukushima Daiichi plant, one of the issues that had to be dealt with were faulty equipment [15, 194]. Had a system been in place which would have allowed for human operatives to securely receive correct and precise readings from within the damaged areas of the plant, perhaps the accident will have been handled and contained in a more correct and effective manner.

The main motivation behind human PSMT protocols was for electronic code voting purposes. Code voting was proposed by Chaum in [33] and in such a voting scheme each voter receives a unique PIN per candidate by *postal mail*. By making the PINs, considered over all different voters and all different candidates, unique, the PIN can be used to vote. To vote, the voter just enters the PIN received which corresponds to the candidate of his/her choice. The breakthrough of Chaum's approach is that one can use a possibly hacked computer to perform a secure operation. Furthermore, it has been shown to be user friendly [7]. But what if contrary to Chaum's assumptions the postal mail cannot be trusted? If that is the case, adversarial collaborations can violate key election properties such as identifying a voter's vote, ballot stuffing, amongst others. In such a case, to make electronic code voting secure, human voters would have to receive shares of their voting codes from a number of possibly corrupt (by malware) computational devices. Reconstruction of the voting codes in such a scenario would have to be carried out by the human voters themselves as their devices are not necessarily trusted. For this

to occur correctly, schemes used would have to be simple and easy enough for humans to carry out the required computations with a very high percentage of accuracy.

Despite the main motivation of the presented protocols being that of a human receiver, the scope of application for such PSMT protocols is not limited to human parties only. The human party of these protocols could just as well be replaced by a computational device. The main advantage of such protocols is that they are computationally efficient and simple (as will be shown in Section 5.8). The only operations required are equivalence - for majority purposes, and addition over an abelian group. The computational simplicity of these protocols could allow their use in other applications of PSMT protocols which use computational devices which are either energy constrained, are computationally slow, or are computationally limited (or a combination of these and other factors). The main difficulty of such protocols is the limitation of the high number of required disjoint paths which may make their practical use difficult. Beyond this though, as will be shown in Section 5.8, the two-phase protocol presented in Section 5.4 transmits a lower number of field elements when compared to other two-phase protocols - making this a more energy efficient protocol one can use in such application scenarios.

5.1.1 Secure Human Computation

Secure human computation is an interesting challenge for the cryptographic community. Its goal is to provide secure cryptographic solutions to humans, when conventional cryptographic solutions are not adequate or cannot be used. This may be the case because of an unavailability of computational power for a number of reasons. These include scenarios where users (humans) may have resource constrained devices, or computational devices may be corrupt and thus not trusted, or because computational devices are not available. Because of this, solutions need to be designed so that they are not only secure but are also computationally efficient and computationally simple so that they can be executed by a human. As stated earlier, such solutions usually employ trivial maths such as addition over an abelian group and mod10 arithmetic.

Previous work has considered secure human computation in different cryptographic contexts. In [118] it was considered in an electronic voting scenario and was used to securely and correctly provide voters with a receipt confirming the correctness of votes cast. Other work which considers secure human computation includes the work of [77] which considered large-scale distributed computation to solve difficult problems, but where humans could act as agents to provide candidate solutions. In such a system, humans would aid in problem classes that appear difficult for computers to solve but are relatively easy for humans to carry out. These include image analysis (for example captchas [189]) and speech recognition. In [92] the authors considered secure authentication and identification for humans when trusted hardware or software - such as smart cards, are unavailable or cannot be used. Other work [120] considers secure human computation also.

This chapter introduces the notion of PSMT protocols with a human receiver - a new research problem which looks at security and computation with a human party. Protocols are presented which achieve PSMT and are computationally efficient and computationally simple so as to be decoded by a human receiver.

It should be noted that despite the protocols presented for the case where the receiver is a

human, they can easily be adapted for the case where the sender is a human and the receiver a computer. Furthermore, both the sender and the receiver can be human parties also.

5.2 Set Systems

In this section we introduce set systems which are the key combinatorial structures used in the construction of PSMT protocols with a human receiver.

In [51] Desmedt-Kurosawa defined the following terms:

Definition 4 ([41]). *A set system is a pair (X, \mathcal{B}) , where $X \triangleq \{1, 2, \dots, n\}$ and \mathcal{B} is a collection of blocks $B_i \subset X$ with $i = 1, 2, \dots, b$.*

In the above definition the symbol \triangleq means “is equal to by definition”.

Definition 5 ([51]). *We say that (X, \mathcal{B}) is an (n, b, t) -verifiers set system if:*

1. $|X| = n$,
2. $|B_i| = t + 1$ for $i = 1, 2, \dots, b$, and
3. For any subset $F \subset X$ with $|F| \leq t$, there exists a $B_i \in \mathcal{B}$ such that $F \cap B_i = \emptyset$.

Similar to the definition of (n, b, t) -verifiers set systems is the following:

Definition 6. *We say that (X, \mathcal{B}) is a generalized (n, b, t', t) -verifiers set system if the following conditions are satisfied:*

1. $|X| = n$,
2. $|B_i| = t' + 1$ for $i = 1, 2, \dots, b$, and
3. For any subset $F \subset X$ with $|F| \leq t$, there exists a $B_i \in \mathcal{B}$ such that $F \cap B_i = \emptyset$.

The definitions of both verifiers set systems and generalized verifiers set systems ensure the *key* property that for a set of blocks (each block of size $(t + 1)$ or $(t' + 1)$ respectively), at least one block does not contain any elements of *any* (at most) t sized subset of elements.

This key property will be important in the construction of PSMT protocols with a human receiver. We will assume X represents the set of wires connecting the sender to the human receiver and \mathcal{B} will be different subsets of these wires. As the adversary is assumed to be t -bounded, this ensures that at least one block will be free from any adversary presence.

In the next two subsections we identify how set systems can be constructed using cover designs and will also identify the example set systems we will use in the description of the protocols to be presented in Section 5.3 and Section 5.4.

5.2.1 Generalized Verifier Set Systems and Covering Designs

We now explain the connection between generalized verifier set systems and covering designs.

Definition 7. *A collection \mathcal{C} of k -subsets of $\{1, \dots, n\}$ called blocks is an (n, k, t) -cover design if every t -subset of $\{1, \dots, n\}$ is contained in at least one block.*

The definitions of generalized verifier set systems and covering designs identify them to essentially be equivalent by taking complements. In other words, one can use cover designs to construct generalized verifier set systems (and vice versa). The following lemma from [51] and corollary show this - the proof of which is clear.

Lemma 4. (X, \mathcal{B}) is a (v, b, t) -verifiers set system if and only if the set system (X, \mathcal{B}^c) is a $(v, v - t - 1, t)$ -covering, where $\mathcal{B}^c \triangleq \{X \setminus B_i \mid B_i \in \mathcal{B}\}$.

The above basically states that one can construct the blocks of the set system by removing each of the blocks of the covering designs from the set of points X . As one block of the covering design contains all the elements of any subset F it is ensured that one block of the resultant set system *will not* contain any elements of F .

Collary 1. Similarly to Lemma 4, (X, \mathcal{B}) is a generalized (n, b, t', t) -verifiers set system if and only if the set system (X, \mathcal{B}^c) is a $(v, v - t' - 1, t)$ -covering, where $\mathcal{B}^c \triangleq \{X \setminus B_i \mid B_i \in \mathcal{B}\}$.

Similarly, one can construct the blocks of a cover design by removing each of the blocks of a set system from the set of points X . As one block of the set system does not contain any of the elements of any subset F it is ensured that one block of the resultant covering design *will contain all* elements of F .

5.2.2 Constructing Set Systems

We now describe the simplest construction of generalized verifier sets which will be used as a reference for protocols to be presented in the next two sections. We will refer to this construction as “Disjoint Set System”.

Lemma 5. When $n = b \times (t' + 1)$ and $b \geq t + 1$, a generalized (n, b, t', t) -verifier set system (X, \mathcal{B}) can be constructed when each block is disjoint to all others.

Proof. From [126, p. 221], as \mathcal{B} is composed of $(t + 1)$ disjoint $(t' + 1)$ -sized blocks, it is easy to see that a t -threshold bounded adversary can be present in at most t blocks - satisfying the properties of generalized verifier sets. \square

Similarly, the following is a construction for an (n, b, t) -verifiers set system.

Lemma 6. When $n = b \times (t + 1)$ and $b = t + 1$, an (n, b, t) -verifiers set system (X, \mathcal{B}) can be constructed when each block is disjoint to all others.

The proof of this is similar to that of Lemma 5.

We refer to verifiers and generalized verifier set systems constructed as in Lemma 5 and Lemma 6 as *Disjoint Set Systems*.

Other ways which can be used to construct generalized verifier set systems (and verifier set systems) follow from [154] and two of these are described in Appendix E. Further constructions result from the extensive work in covering designs such as those of [1, 2, 81, 82, 126, 154] amongst others.

When choosing any of these constructions for the protocols presented in Section 5.3 and Section 5.4, the complexity on the number of wires required (size of X) without requiring exponentially many blocks¹ is $O(t^2)$. As future work, the protocols to be presented could be made

¹An exponential number of blocks make such a solution impractical to use with a human party.

more efficient regarding communication and computational complexities - where less data need to be sent and less computation needs to be carried out by the communicating parties. To be able to achieve this, more efficient constructions of set systems - which require a lower than $O(t^2)$ complexity on the number of wires required (size of X), whilst still achieving secure and polynomial time protocols, is an interesting question to solve. This is important to study further as currently all known constructions have a complexity of $O(t^2)$ in the size of X . Because of this, the communication complexity of such protocols is $O(t^2)$ and the communication complexity is close to this bound also. Future work should thus aim to decrease the communication and computational complexities below this bound.

5.3 One-Phase Mod 10 PSMT with a Human Receiver

In this section we present a one-phase mod10 perfectly secure message transmission protocol with a human receiver.

In the one-phase protocol the human receiver cannot communicate with the sender. The only transmission of data which occurs is from the sender to the receiver. The protocol we present is based on the concept of generalized (n, b, t', t) -verifier set systems and mod 10 arithmetic. As shown in [56], one-phase PSMT can only occur when $n \geq 3t + 1$ wires are used. We assume that wires connecting the sender and the receiver correspond to points in a generalized $(n, b, 2t + 1, t)$ -verifier set system to be used. For the protocol we present we consider an active adversary.

The main idea of the protocol is as follows. The secret message will be secret shared in a similar manner as described in Section 2.6.2 - but instead a b -out-of- b secret sharing scheme will be used (where b denotes the number of blocks to be used). Each share will then be transmitted over a block of wires with each share transmitted over one block only. Each block of wires will be of size at least $(2t + 1)$ so that a majority can always be received correctly by the receiver. By sending shares over blocks of wires chosen so that the adversary is not present in at least one of these blocks, the correctness of the protocol is achieved.

The protocol is formally presented as follows - denoting with M^S the secret of the communication:

Protocol 5. *One-phase Transmission Protocol with a Human.*

Protocol Setup *A generalized (n, b, t', t) -verifiers set system with b number of blocks B_1, \dots, B_b all of which are of size $|B_i| \geq 2t + 1$ and which satisfy the key property that at least one block does not contain elements of any (at most) t sized subset of elements will have to be constructed. We will assume the case of when all these sets are truly disjoint between them - as shown in the construction of Lemma 5, as this construction is the simplest and easiest construction for a human party to use. Using this construction, the parameters of the protocol are as follows - there are $b = t + 1$ blocks with the size of each block $|B_i| \geq 2t + 1$ (although $|B_i| = 2t + 1$ suffices).*

Phase 1 *The sender generates secret shares of M^S using a b -out-of- b secret sharing scheme as described in Section 2.6.2 and sends share s_i ($1 \leq i \leq b$) over all wires which correspond to points for block B_i .*

End of Phase 1 *The receiver obtains the secret message M^S by using all correct shares and using the reconstruction process described in Section 2.6.2. The receiver identifies correct shares using a majority vote - correct shares are received at least $(t + 1)$ times. By summing the digits over all correct shares mod10 the receiver will reconstruct the secret.*

Theorem 6. *The One-phase Transmission Protocol with a human receiver is a perfectly secure message transmission protocol.*

Proof. The protocol achieves perfect privacy as due to Property 3 of generalized verifier set systems, there will exist at least one block of wires free of the adversary. This is achieved as the adversary is t -bounded. In this way, the adversary does not learn at least one share transmitted by the sender and perfect privacy is achieved.

Perfect authenticity is achieved as the protocol only uses blocks of size at least $(2t + 1)$. As the adversary is t -bounded, this ensures that the human receiver accepts a majority of the correct share transmitted by the sender for all blocks. The human receiver thus correctly accepts the secret by reconstructing the secret using majority received (correct) shares.

As the concept of majority is used to confirm the correctness of a share (or random element) and as this is always achieved, the adversary cannot cause the protocol to fail in any way. The protocol is thus a perfectly secure message transmission protocol. \square

5.3.1 One-Phase PSMT with a Human Using a Disjoint Set System

We may assume the case of when all blocks are truly disjoint between them - as described by Lemma 5, as this construction is the simplest and easiest construction for a human party to use. Using this construction, the parameters of the protocol are as follows - there are $b = t + 1$ blocks with the size of each block $|B_i| \geq 2t + 1$ (although $|B_i| = 2t + 1$ suffices). As each wire is used only once over all blocks, it is easy to see that this construction will contain at least one block without an adversary presence.

Assuming the use of the construction described in Lemma 5, the sender's computational complexity is $O(t^2)$. This is because the sender has to transmit a field element in $(Z_{10})^k$ on each of the $O(t^2)$ wires connecting sender and receiver². The computational complexity of the receiver is $O(t^2 \log t)$, because the receiver at the end of Phase 1 accepts $O(t^2)$ field elements in $(Z_{10})^k$ and because the correctness of these elements needs to be checked to identify the correct $t + 1$ shares of the secret. It is easy to see that the communication complexity of the protocol is also $O(t^2)$ - as the sender transmit a single field element in $(Z_{10})^k$ over each of the $O(t^2)$ wires³.

5.3.2 Optimally Connected One-Phase PSMT with a Human

As shown in [56] the optimal number of wires for one-phase PSMT is $n = 3t + 1$. Using this number, there exist a trivial generalized $(3t + 1, \binom{3t + 1}{2t + 1}, 2t + 1, t)$ -verifier set. A protocol

²The sender also has to secret share the secret message, but using this secret sharing scheme described, it is easy to see that this requires $O(t)$ computation.

³It is important to note that using any of the known set systems constructions will always result in the same communication and computational complexities - as all state of the art constructions will require $O(t^2)$ wires connecting sender and receiver.

initiated with these parameters is identical to a protocol presented in [54] when using a threshold adversary structure. In such a protocol, all possible $(2t + 1)$ subsets of wires from $(3t + 1)$ wires are used as blocks. Such a protocol though is impractical for use when a human party is involved as the number of required blocks increases greatly as the value of t increases.

It should be noted that for small values of t (say $t = 1, 2$) the number of subsets of wires which will be used as blocks will be relatively low, allowing for such a protocol to be simple and practically feasible for a human party to use.

5.4 Two-Phase Mod 10 PSMT with a Human Receiver

In this section we present a two-phase mod10 perfectly secure message transmission protocol with a human receiver.

For two-phase protocols, the receiver and sender are able to interact over a network with $n \geq 2t + 1$ number of wires connecting the two parties. The solution we present is based on the concept of (n, b, t) -verifier sets. We assume that wires connecting the sender and receiver correspond to points in the (n, b, t) -verifier set system to be used.

The main idea of the protocol is as follows. The protocol is initiated by the receiver who transmits different random values to the sender upon different sets of wires (blocks) of size at least $(t + 1)$ - transmitting the same value over wires in the same set. This ensures that if the adversary is active, the sender is able to identify sets of wires for which *different* values are received. Such sets are blacklisted. The sender uses the values received from non-blacklisted sets to privately send the secret of the communication.

The protocol is formally presented as follows with M^S denoting the secret message:

Protocol 6. *Two-Phase Transmission Protocol with a Human.*

Protocol Setup *An (n, b, t) -verifier set system with b number of blocks B_1, \dots, B_b all of which are of size $|B_i| \geq t + 1 = t'$ and which satisfy the key property that at least one block does not contain elements of any (at most) t sized subset of elements will have to constructed. We use \mathcal{B} to denote the set of all blocks used. We will assume the case of when all these sets are truly disjoint between them - as shown in the construction of Lemma 6, as this construction is the simplest and easiest construction for a human party to use. Using this construction, the parameters of the protocol are as follows - there are $b = t + 1$ blocks with the size of each block $|B_i| = t + 1$.*

Step 1 *For each block B_1, \dots, B_b , the receiver chooses a uniformly random field element r_i from $(\mathbb{Z}_{10})^k$. When $B_i = \{x_{i_1}, x_{i_2}, \dots, x_{i_{t'}}\}$, the sender sends the value of r_i on wire x_{i_j} (for all x_{i_j} in B_i where $1 \leq j \leq t'$).*

Step 2 *If the received values that should correspond to r_i on wires $B_i = \{x_{i_1}, x_{i_2}, \dots, x_{i_{t'}}\}$ are different, the sender black lists B_i (and checks this for all $1 \leq i \leq b$). The sender broadcasts the complete black list of such B_i to the receiver. We call this list \mathcal{B}' .*

The sender then computes $\mathcal{B}'' = \mathcal{B} \setminus \mathcal{B}'$. The sender adds the random values r_i corresponding with the blocks in \mathcal{B}'' . So, if $\mathcal{B}'' = \{B_{j_1}, B_{j_2}, \dots, B_{j_l}\}$, the sender computes $r = r_{j_1} + r_{j_2} + \dots + r_{j_l}$ and sends to the receiver $V = r + M^S$ using broadcast. (The

addition of the random values to calculate r occurs using mod10 arithmetic for each digit and likewise for the evaluation of V .)

Step 3 The receiver is able to decode M^S in a similar manner to the actions of the sender in Step 2. B'' will first be found using the received broadcast value of B' and then the value of r will be calculated. The secret message M^S can then be decoded as $M^S = V - r$. (The addition of the random values to calculate r occurs using mod10 arithmetic and likewise for the subtraction which occurs for the evaluation of M^S .)

Note: It should be noted that the broadcast which occurs in Step 2 does not have to be carried out by transmitting the information to broadcast over all wires connecting the sender and the receiver. Instead, this information can be sent on any $2t + 1$ wires. As there can be at most t faulty wires, this ensures that the correct information will be correctly received by the receiver at the end of Step 2.

Theorem 7. *The Two-phase Transmission Protocol with a human receiver is a perfectly secure message transmission protocol.*

Proof. The protocol achieves perfect privacy as due to Property 3 of verifier set systems, there exists at least one block with wires free of the adversary. Because of this, the adversary does not learn at least one random value sent by the receiver in the first phase. As this block is free of the adversary, a single value will be received from this block. This value will be used in the calculation of r and as the adversary does not learn it, perfect privacy is achieved.

Perfect authenticity is achieved as the sender only considers blocks from which a single value was received. As blocks are of size $(t + 1)$, a t -bounded adversary cannot make the sender consider a value the receiver did not send. This in addition to the broadcast of the list of blacklisted blocks B' , allows for perfect authenticity to be achieved and for perfect availability also. \square

5.4.1 Two-Phase PSMT with a Human Using a Disjoint Set System

We may assume the case of when all these sets are truly disjoint between them - as shown in the construction of Lemma 6, as this construction is the simplest and easiest construction for a human party to use. Using this construction, the parameters of the protocol are as follows - there are $b = t + 1$ blocks with the size of each block $|B_i| = t + 1$.

Assuming the use of the construction described in Lemma 6, the receiver's computational complexity is $O(t^2)$. This is because the receiver has to transmit a field element in $(Z_{10})^k$ on each of the $O(t^2)$ wires connecting the two parties in the first phase⁴. In the second phase of the protocol, the receiver may have to accept $O(t^2)$ field elements in $(Z_{10})^k$ - from the broadcast of the black list B' of 'faulty' blocks which can be of maximum size t . The receiver has to identify the majority of the black list and reconstruct the secret - which requires less computation. The computational complexity of the sender is $O(t^2 \log t)$ as the sender accepts $O(t^2)$ field elements in $(Z_{10})^k$ at the end of the first step and has to evaluate and broadcast black list B' . The communication complexity of the protocol is $O(t^2)$ because of the broadcast of the $O(t)$ sized set of black-listed blocks over any $O(2t + 1)$ wires.

⁴The receiver also has to select $t + 1$ random elements and evaluate B'' .

5.4.2 Optimally Connected Two-Phase PSMT with a Human

As shown in [56] the optimal number of wires for two-phase PSMT is $n = 2t + 1$. Similar to one-phase transmission protocols with a human, there also exists a trivial $(2t + 1, \binom{2t + 1}{t + 1}, t)$ -verifier set. In such a protocol, all possible $(t + 1)$ subsets of wires from $(2t + 1)$ wires are used as blocks. Such a protocol though is impractical for use when a human party is involved as the number of required blocks increases greatly as the value of t increases.

It should be noted that for small values of t (say $t = 1, 2, 3$) the number of subsets of wires which will be used as blocks will be relatively low, allowing for such a protocol to be simple and practically feasible for a human party to use.

5.5 Practically Feasible Protocols for Small Values of t

A slight weakness of the one and two-phase protocols presented in the previous two sections is the rather sub-optimal number of wires required to tolerate a t -bounded adversary. In this section we identify parameters that could be used for natural practical use case scenarios where the value of t is relatively small - up to $t = 4$.

For both the one and two-phase protocols we identified an alternative set system comprised of all $(2t + 1)$ and $(t + 1)$ subset of wires from the respective optimal number of wires. We now comment on the set systems which could be used for these alternative protocols for small values of t .

For the one-phase case, $n = 3t + 1$ number of wires will be required. The generalized verifier set system to be used will consist of all possible $(2t + 1)$ sized subsets. For $t = 1$ and $n = 4$ the number of blocks will be $C(4, 3) = 4$. For $t = 2$ and $n = 7$ the number of blocks will be $C(7, 5) = 21$. Beyond $t = 2$ the number of blocks for the human receiver to use becomes impractical (for $t = 3$ $C(10, 7) = 120$).

Similarly, for the two-phase case, $n = 2t + 1$ number of wires will be required. The verifier set system to be used will consist of all possible $(t + 1)$ sized subsets. For $t = 1$ and $n = 3$ the number of blocks will be $C(3, 2) = 3$. For $t = 2$ and $n = 5$ the number of blocks will be $C(5, 3) = 10$ and for $t = 3$ and $n = 7$ the number of blocks will be $C(7, 4) = 35$. Beyond $t = 3$ the number of blocks for the human receiver to use becomes impractical (for $t = 4$ $C(9, 5) = 126$).

Further to the above, in Table 5.1 and Table 5.2 we also present examples of practically feasible set systems for small values of t which could be used by the protocols presented in the earlier two sections. We identify the number of required blocks and the size of set X of the set system. For the one-phase protocol, we require the blocks to be of size at least $2t+1$. Similarly, for the two-phase protocol, blocks size must be at least $t+1$. Specifics on how such set systems can be constructed follow from the source of the data which is the La Jolla Covering Repository [1] (please note that other similar examples found in the repository could also be used).

5.6 Human Friendly Addition Mod 10

The two protocols presented in Section 5.3 and Section 5.4 required the human receiver to carry out some sort of computation. Despite the computational and seemingly mathematical simplic-

Value of t	Size of Blocks	Number of Required Blocks	$ X = n$	$n = 3t + 1$
1	3	4	4	Yes
2	5	11	8	No
2	5	21	7	Yes
3	7	21	13	No
3	7	15	14	No
4	9	28	20	No
4	9	25	21	No

Table 5.1: Examples of various generalized verifier sets which could be used in the one-phase protocol for small values of t using Corollary 1.

Value of t	Size of Blocks	Number of Required Blocks	$ X = n$	$n = 2t + 1$
1	2	3	3	Yes
2	3	6	6	No
2	3	10	5	Yes
3	4	12	9	No
3	4	35	7	Yes
4	5	24	12	No

Table 5.2: Examples of various verifier sets which could be used in the two-phase protocol for small values of t using Lemma 4.

ity, some human users may find these operations confusing and difficult to execute correctly.

In this section we present an alternative PSMT protocol with a human receiver when considering a *passive* adversary. The protocol is more user friendly and easier for human receivers to use as no *apparent* form of mathematical operations need to be carried out. Instead, all receivers have to do is to follow a path traced by joined lines over some figures - as will be described in Section 5.6.2.

This alternative protocol though is just another representation of addition mod10 which uses permutations to make the addition mod 10 operation friendlier to humans. We explain this with more detail in the next section. We regard $Z_{10}(+)$ as a subgroup of the symmetric group S_{10} .

5.6.1 Addition Mod 10 Using Permutations

Supposing that one wants to carry out the addition mod10 of $x + s$ (where $0 \leq s, x, \leq 9$). Furthermore, lets assume that this addition should use secret sharing - as s is a secret that should not be transmitted over a single channel. This in effect means that when carrying out addition mod10 and provided that $s_1 + s_2 + s_3 = s$, $x + s \equiv x + s_1 + s_2 + s_3$.

In effect, the shares s_1, s_2 and s_3 denote how much the value of x should progressively shift (upwards due to addition) mod10. We now describe how this can be represented as permutations and more importantly (to achieve user friendliness) pictures of shifts.

Suppose that the value of $x = 6$ and that the value of $s = 5$. This means that after the addition mod10 of $x + s$, the result should be $(6 + 5)\%10 = 1$ (where $\%10$ denotes addition mod10). A picture of the shift should thus point from 6 to 5.

Furthermore, lets assume that s is secret shared to $s_1 = 8, s_2 = 4$ and $s_3 = 3$. This means that the value of x should shift from its original value of 6 to $((6 + 8)\%10 =) 4$ when added

mod10 with s_1 . When this result is added mod10 to s_2 it should then shift to $((4 + 4)\%10 =) 8$ and when this second result is added mod10 to s_3 is should result to $((8 + 3)\%10 =) 1$.

Based on this description, we present through the following diagram how addition mod 10 can be represented as permutations and diagrams.

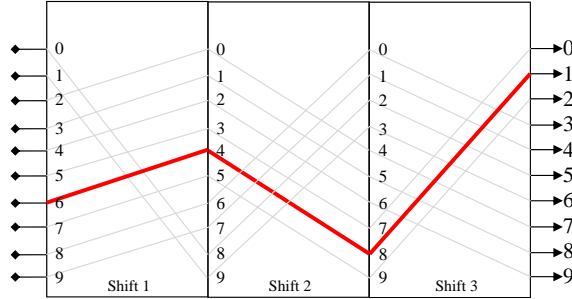


Figure 5.1: Representing addition mod10 through diagrams of permutations.

Looking at the diagram it is easy to see how addition mod10 can be represented as diagrams. The three different diagrams labeled “Shift 1”, “Shift 2” and “Shift 3” represent the addition mod10 of s_1 , s_2 and s_3 to x in consecutive resultant order as described in the given example. Each of these diagrams is in effect also a permutation as different numbers are mapped elsewhere by each diagram as shown by the light grey lines of each shift diagram.⁵

5.6.2 Human PSMT Permutation Protocol

We now present the more user friendly PSMT protocol with a human receiver considering a t -threshold bounded passive adversary. In the background the scheme uses permutations to add modulo a number as described in the earlier section. Despite this, all human receivers have to carry out is the simple task of tracing a line amongst other lines in a diagram.

For the protocol to be presented we consider a t -threshold bounded *passive* adversary, thus the number of network disjoint paths (or wires) n required to connect the sender and a human receiver equal $n = t + 1$. In Section 5.6.3 we outline how such protocols can be used against an active adversary.

We now present the general description of the protocol. In the protocol the sender transmits $\pi_i \in_R S_c$ over wire w_i ($1 \leq i \leq t+1$). Each π_i will be a *diagram* which is part of a permutation. So instead of receiving shares of a secret (as for the protocols in Section 5.3 and Section 5.4), the human receiver receives shares of one permutation. Each share of the permutation will be a diagram and when the human receiver brings all diagrams (corresponding to all the shares of a permutation) together in the correct order, the human receiver will be able to view the complete permutation.

We now present the protocol more formally.

Protocol 7. *Human Permutation Protocol.*

Protocol Setup *The sender constructs a random permutation π and shares this permutation over n other random permutations - π_1, \dots, π_n , such that these permutation placed in order reconstruct π . Each π_i ($1 \leq i \leq n$) will be a diagram which is part of π .*

⁵In effect all of them are added mod10 by the diagram shift amount.

Phase 1 The sender transmits $\pi_i \in_R S_c$ over wire w_i ($1 \leq i \leq n$).

End of Phase 1 The human receiver receives shares of a permutation - each of which is a diagram. The human receiver brings all diagrams (all the shares of a permutation) together in their right order (π_1 first, π_2 second, \dots , π_n last), and is able to view the complete permutation.

Theorem 8. The Human Permutation Protocol is a PSMT protocol.

Proof. Perfect privacy is achieved as the adversary will not learn the permutation transmitted on the wire which the adversary does not control. As these permutations are randomly constructed, perfect privacy is achieved. Perfect reliability is achieved as a passive adversary is considered. \square

For clarity, we also present and explain the protocol through the use of diagrams. We assume the value of $t = 1$ and that the secret to be transmitted from the sender to the receiver is a single digit.

The human receiver starts of with the same initial handout of instructions similar to that shown in Figure 5.2. Such a handout could be made readily available before any such protocol execution is required.

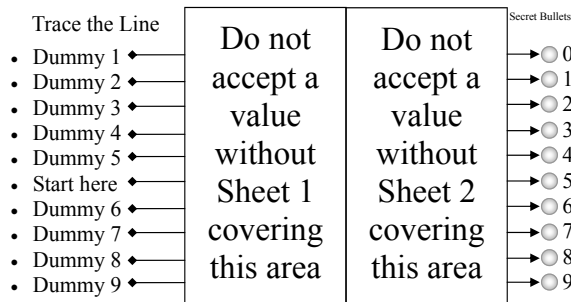


Figure 5.2: Initial view of human receiver.

We assume that the two diagrams to be transmitted by the sender will be the those shown in Figure 5.3.

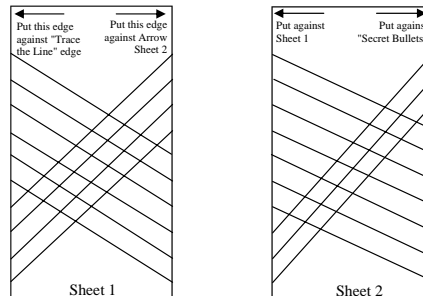


Figure 5.3: Two diagrams to be received by the human receiver.

The human receiver will receive the first diagram from one of the two wires which connect the sender to the receiver. This will be placed in the position indicated by the handout as shown in Figure 5.4.

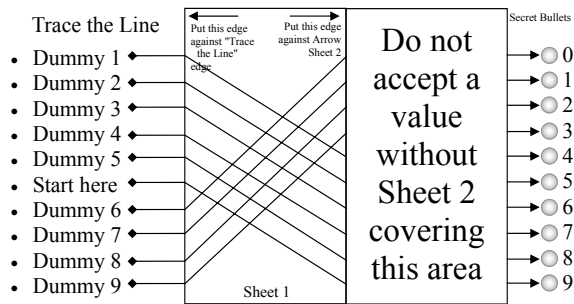


Figure 5.4: The human receiver will place the first diagram in the appropriate position.

In a similar manner, the human receiver will receive the second diagram from the second wire and place it in the position indicated by the handout as shown in Figure 5.5.

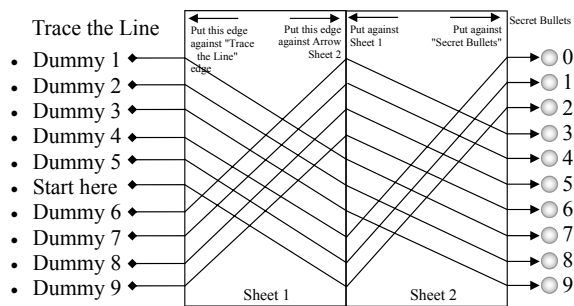


Figure 5.5: The human receiver will place the second diagram in the appropriate position.

The human receiver will then identify the one digit secret. They can do this by tracing the line over the two permutation sheets and identifying the radio button number which corresponds to the start of the line that was traced. The radio button number will correspond to the digit secret. This is shown in Figure 5.6.

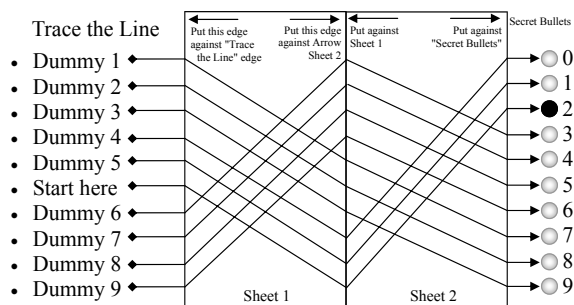


Figure 5.6: The human receiver traces the radio button corresponding to the start of the traced line. In the example, the secret digit is 2.

Once the human receiver identifies the radio button which corresponds to the start of the traced line, they can then accept the secret digit of the communication as the number which corresponds to the final radio button.

From the description, this protocol seems easy for a human receiver to use without errors. In Section 5.7 we assess the way human participants used a variant of this protocol by discussing

the results of experimental evaluation that was carried out.

The protocol is a perfectly secure message transmission as a t -bounded passive adversary can learn at most t of the diagrams which are sent to the human receiver. *Given that the generation of these diagrams are carried out in a random manner by the sender*, then by not knowing one of these $t + 1$ in number diagrams, it is easy to see that the adversary cannot know the value of the secret.

Note: If a secret to be sent to a human receiver is a number with more than one digit, then the above process will have to be carried out in independent executions for each of the digits which correspond to the secret.

5.6.3 Human PSMT Permutation Protocols Against an Active Adversary

It is easy to see that the constructions used for the protocols presented in Section 5.3 and Section 5.4) can also be used for human PSMT permutation protocols against an active adversary. The only way in which the protocols will be different, is that instead of the transmission of shares, transmission of diagrams will take place.

For the one-phase active adversary protocol, the required changes are very apparent (just transmit diagrams in the place of shares). For the two-phase case, the changes would be very similar but some extra steps will be required⁶.

5.7 Experimental Evaluation of the Human Friendly Addition

Before perfectly secure message transmission protocols with human computation are deployed to be used by human parties in secure message transmission (whether it is for code voting schemes or for any other application), it is important that an empirical study be carried out to evaluate the *correctness* with which people can use and evaluate the correct result (output) of such protocols. This kind of study may provide insight into how people respond to such a protocol and could give useful information on the best way to express instructions of their use.

To assess the human usability of the proposed human PSMT protocols we carried out experiments with participants. It should be noted that these experiments were carried out in the context of a paper [49] which focused on how humans interact with electronic voting protocols.

Although our original experiments [49] were directed to check how user friendly our alternatives are to Chaum's code voting [33], the same experiments allow us to evaluate how correctly the participants used the concepts of PSMT with a human as proposed in this chapter.

Details of the experiments and the experimental evaluation that was carried out are presented in this section.

5.7.1 Demographics of Participants

One hundred different people participated in the experimental evaluation of the proposed protocols. We tried to target adults across various age groups and of different educational backgrounds. Participants were classified by their age groups and whether they held a university degree or not.

⁶This will include ordering the diagrams which are not blacklisted in some order and possibly sending a correction deviance value to precisely define the secret digit of the communication. All of this information can be sent via broadcast. As set systems are used and as the adversary will not know at least one diagram the security of the protocols will still be achieved.

More specifically, the latter identified people with a university degree as those who either were currently attending university (3rd year or beyond) or had completed university and obtained a degree (in various subjects and not only confined to scientific ones). People identified in the group that *did not* hold a university degree included people that left school at the age of fifteen, people that did not proceed with further education beyond high school and people that attended a college (at most two years further education).⁷

The combined demographics (based on age and level of education) of the people that participated in the experiments can be seen in the figure below.

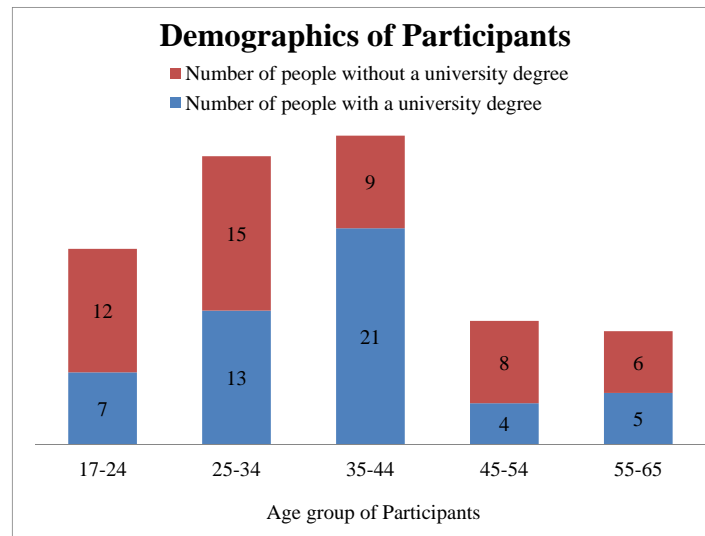


Figure 5.7: Demographics of participants.

5.7.2 Experimental Procedure

Before we describe the experimental procedure we first must thank the participants that took part in the experiments and gave us their consent to use the results of their participation for this experimental assessment. The participants cannot be named (not only because there were many of them) as their results were anonymized to maintain participant privacy. This was done to conform to UCL ethics rules on aptitude tests which state that:

The following types of human participant research DO NOT require ethics approval:

Research involving the use of educational tests (cognitive, diagnostic, aptitude, achievement), survey procedures, interview procedures or observation of public behavior UNLESS information obtained is recorded in such a manner that human participants can be identified ...

For this reason, experimental results were not marked in any way which could identify the person who created them. Furthermore, the results were placed in an envelope which contained all results papers (complete or not) and these were mixed to prevent identification. Once all the experiments were finished, the results were taken out of the envelope to be analyzed.

⁷It should be noted that none of our participants that attended a college followed scientific/mathematical related education.

The experiments were paper based. Participants were given the same experimental sheets of paper as those which can be found in Appendix F for the permutation experiments and in Appendix G for the mod10 experiments.

The permutation vote experiments were the first ones to be carried out and when participants were finished, they progressed to the mod10 voting experiments.

For the permutation vote experiments, the two sets of sheets corresponding to Sheet 1, Sheet 2 and Sheet 3 were provided to participants by the person conducting the experiment in a random shuffled order. The sheets for the first method were first provided to participants and once they had completed the required tasks, the sheets for the second method were given to them.

For all participants of the experiment, participants were *not* given any instructions further to those that could be found on the experiment sheets which can be found in the appendices. The only form of interaction participants had with the person conducting the experiment was the exchange of sheets for the two versions of the permutation vote experiments.

Furthermore, before they began, participants only knew that the experiment they were participating in was only to test “The way people use two voting protocols”. Further details about the experiments and the scope of their research was given to them once they had completed all the required experimental tasks.

5.7.3 Experimental Results

The following figure identifies the experimental results of the mod10 voting schemes.

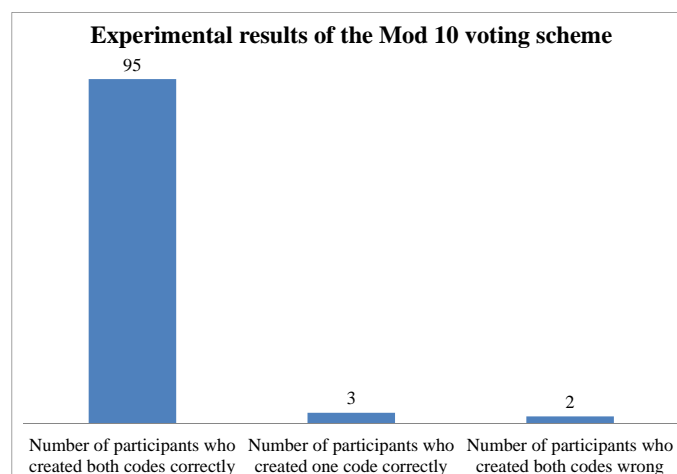


Figure 5.8: Experimental results of the mod10 voting scheme

The following figure identifies the experimental results of the permutation voting schemes.

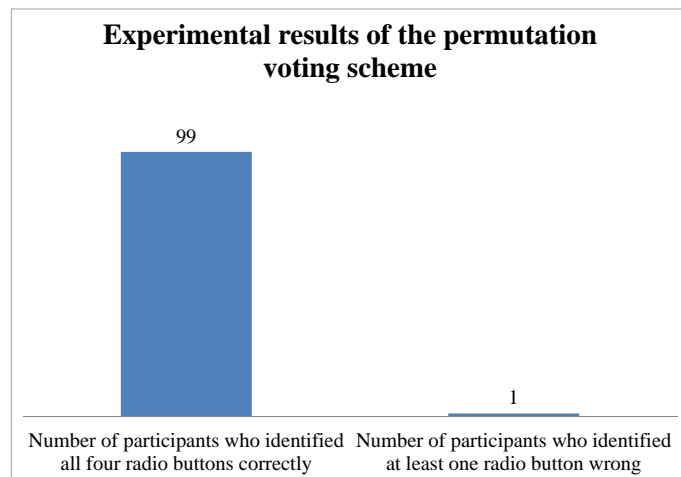


Figure 5.9: Experimental results of the permutation voting scheme

Based on the above results, it is easy to see that the mod10 voting system has a 95% success rate. Similarly, the permutation voting system has a 99% success rate.

However, it is important to realize that even though there were 100 participants, the number of *experiment instances* were far greater.

For the mod 10 voting scheme, each participant was asked to create the code of two different candidates, thus we can consider that there were 200 test instances - although not independent. Similarly, for the permutation based scheme, each participant was asked to identify the radio button corresponding to candidates in four different cases. Thus we consider that overall there were 400 test instances among both versions of this scheme - although not independent. We believe this should be taken into account as there is a distinction between the different types of errors that were created. Some participants - especially in mod 10 voting scheme, made only one error when during the experiment they were asked to calculate two codes.

Based on this, we could argue the success rate of the mod10 voting system is close to 96.5% and for the permutation voting system is close to 99.75%.

5.7.4 Experimental Results Discussion

In this section we discuss the results of our experiments.

Both schemes performed extremely well and this is seen by the very high percentage of people that used them correctly to find a correct answer.

This was especially the case for the permutation voting protocol. For this protocol, only a single error (out of 400 experiment instances) was made. It should be noted that this error only came about because the participant made a mistake and not because the participant did not understand how to use the protocol (as the participant was correct for the other three experiment instances). During discussion with participants after the experiments, many participants commented that they found the permutation voting experiment extremely easy. Some participants also said that they found the second version of the experiment (using the second set of sheets) slightly easier than the first version of the experiment - as there was a lower amount of untidiness between the lines which allowed for the required tracing to be easier.

For the mod10 voting experiments only five people (out of one hundred) made errors.

Two of these people got both experimental instances (calculation of code for candidate A and D) wrong. One of these participants had a university degree and it seems that they over thought the experiment and instead of following the experiment instructions they sort of came up with their own alternative version of the experiment. The other participant that made errors in both experiment instances clearly did not understand the instructions and as a result made errors. Three different participants made errors on just one of the experiment instances. One's mistake was that for the first instance (code for candidate A), they copied the given example exactly whereas they did everything correctly for the second instance. Another's mistake was that instead of evaluating the code for candidate D they evaluated the code for candidate B (but did so correctly). The third participant to get one of the instances wrong did so as they got the summations for candidate D wrong for some reason - maybe due to human error.

Different people interacted with the mod10 voting experiments in different ways. Some people found it extremely easy and did not even have to look at or use the code generation form example to complete the experiment correctly. This was noticed for some people that both had and did not have a university degree. In general, people found this experiment more challenging than the permutation based scheme. One comment that came up several times during discussion with participants after the experiments had finished was that the initial instructions of the code generation were slightly confusing but once they saw the filled in code generation example things were easier to understand. Furthermore, some participants also stated that if these forms were not provided (the example form and the forms they used to create the codes), they would have most likely got things wrong.

This highlights the importance of clear and concise instructions and forms when dealing with people, usability and how people use security protocols.

Despite the errors that occurred, it should be taken into account that if such schemes were to be used by people, then education of users on how to use the schemes would occur before any human PSMT protocols are used. This would include more examples for people to correctly understand how to use protocols and perhaps clearer, shorter and more concise instructions. Even though errors would most likely still occur, they may (but not necessarily) occur at a lower rate.

5.7.5 Relativity of Experiments to PSMT Protocols and Discussion of Results

In this section we explain how the presented experiments carried out in the context of an electronic voting paper are also valid in the context of the human PSMT protocols presented in this chapter.

It is easy to see that the experiments carried out in the context of the mod10 voting schemes are very relative to the protocols presented in Section 5.3 and Section 5.4. This because both the mod10 voting scheme and the protocols presented in the two sections use the same secret sharing scheme as described in Section 2.6.2. The experiments carried out showed that human participants were able to use the secret sharing scheme correctly with a fairly high success rate. This indicates that human receivers will also use the protocols presented in Section 5.3 and Section 5.4 correctly. Identifying a majority of data received is something that human parties will have to carry out in the protocols - something which was not tested in the

experiments that were carried out. But this seems a relatively easy operation that humans can perform. So *perhaps* human receivers will be able to use protocols similar to those presented in Section 5.3 and Section 5.4 with high accuracy.

It is also clear to see the relationship between the experiments referred to as permutation voting schemes with the PSMT protocol with a human receiver presented in Section 5.6. In the experiments carried out, participants were asked to identify bullets corresponding to identified candidates in a similar way as the description of the protocol presented in Section 5.6.2. Based on the very high accuracy of correctness for the experiments, this suggests that human receivers will use the protocol presented in Section 5.6 correctly in most cases also.

When comparing the two techniques between them, it is apparent from the results of the experiments that the human participants had a greater success rate with the permutation based mod10 addition than with the regular addition mod10. In fact, it was a common comment from the participants that the permutation based mod 10 addition was extremely easy - whereas the other experiment was rather challenging for some people.

5.8 Comparing Human Computation Protocols to other PSMT Protocols

In this section we briefly compare the PSMT protocols with human computation presented in this chapter to other PSMT protocols which do not use this type of computation.

When considering one-phase protocols, we compare Protocol 5 presented in Section 5.3 to the one-phase protocol presented in [56]⁸.

Between the two, the protocol of [56] is more efficient in various aspects. It requires a lower number of wires ($3t + 1$ vs $O(t^2)$) and transmits a lower number of field elements also ($3t + 1$ vs $O(t^2)$). Computationally though, Protocol 5 is a simpler protocol. Both protocols have the same computational complexity of $O(t^2)$ for the sender of the communication. The sender computational complexity of Protocol 5 is $O(t^2)$ only because the sender has to transmit a field element on each of the wires connecting the sender and the receiver. The actual computation carried out by the sender of Protocol 5 is to secret share the secret message which requires a computational complexity of $O(t)$. This is computationally simpler than Shamir secret which is executed by the sender of the protocol presented in [56] and has a computational complexity of $O(t^2)$. Comparing the computational complexities of the receivers, for Protocol 5 the complexity is $O(t^2)$ but for the protocol presented in [56] is $O(t^3)$ - due to the error correction which may be required. The one-phase human computation protocol is thus a computationally simpler protocol.

Considering two-phase protocols, we compare Protocol 6 presented in Section 5.4 to the final (most efficient) version of Protocol 3 presented in Section 4.1 - which is the most efficient two-phase PSMT protocol in the literature.

As in the one-phase case, Protocol 3 is more efficient with regards to the number of wires it requires - $2t + 1$ compared to $O(t^2)$ wires required for Protocol 6.

⁸For this protocol $3t + 1$ wires connecting the sender and the receiver are required. The sender secret shares the secret message using a $t + 1$ -out-of- $3t + 1$ Shamir secret sharing scheme and transmits one share per wire. Using error correction techniques, the receiver is able to uniquely recover the secret message with perfect security.

The communication complexity of both protocols is $O(t^2)$. Detailed analysis of the two protocols though shows that a *lower number* of field elements are transmitted by Protocol 6. Considering the example set construction given in Section 5.4, in Step 1 of Protocol 6, the receiver will transmit $(t+1)^2$ number of field elements. The number of field elements transmitted in the first phase of Protocol 3 are $(2t+1) \times ((2t+1) + (t+1))$ - as n shares and the definition of an at most t degree polynomial is transmitted on each wire. The number of field elements transmitted are $t^2 + 2t + 1$ for Protocol 6 versus $6t^2 + 7t + 2$ for Protocol 3. In Step 2 of Protocol 6 the set of black listed blocks \mathcal{B}' - which is of size at most t , as well as the encrypted secret message is broadcast. By implementing broadcast using only $2t + 1$ wires the number of field elements transmitted is at most $(2t + 1) \times (t + 1)$. In the second phase of Protocol 3 the generalized broadcast of the error shares is transmitted. As the number of error shares can be a large proportion of n^2 shares, the number of field elements which will have to be transmitted in this generalized broadcast are greater than the number of field elements transmitted in the equivalent step of Protocol 6.

Considering the computational complexity of the two protocols, Protocol 6 is computationally more efficient. This because the computational complexity of both sender and receiver is $O(t^2)$ whereas for Protocol 3 the computational complexity for both parties is $O(t^3)$.

It should be noted that the human computation protocols presented in this chapter are not only computationally more efficient, but they also use computational operations (comparisons, addition) which are simpler to execute in most computational devices (in an arithmetic logical unit of a central processing unit (CPU)) compared to the additional computational operations (multiplication, division) required by the other protocols. This is because such operations require a lower number of clock cycles to complete when considering a CPU architecture which uses pipelining of instructions⁹.

5.9 Future Work

Before using protocols with human parties, ways of overcoming the limitation of the high number of required disjoint paths in practise should be sought.

Additionally, the protocols that were presented could still be improved. The protocols presented in Section 5.3 and Section 5.4 are based on set systems. When choosing the values given as parameters for the proposed protocols, the complexity on the number of wires required (size of X) without requiring exponentially many blocks is $O(t^2)$. As future work, the protocols presented could be made more efficient regarding communication and computational complexities - where less data need to be transmitted and less computation needs to be carried out by the communicating parties. To achieve this, more efficient constructions of set systems - which require a lower than $O(t^2)$ complexity on the number of wires required (size of X), whilst still achieving secure and polynomial time protocols, is an interesting question to solve. This is important to study further as currently all known constructions have a complexity of $O(t^2)$ in the size of X .

The work on the proposed protocols and the outcomes of their initial experiments are very

⁹Such protocols could thus be executed faster than previous PSMT protocols - due to the simplicity of their required computational operations.

encouraging. But despite this, further work that could be done would be to carry out a more extensive experimental study with further testing on the proposed protocols. This could include a greater proportion of the population. For example, even though we had 100 participants, their ages did not surpass 65. Thus further experiments of the proposed protocols should include older participants to assess how older human receivers could use the proposed human PSMT protocols.

Chapter 6

Perfectly Secure and Anonymous Message Transmission

This chapter combines the concept of anonymity in communication with perfectly secure message transmission. Contrary to previous work which has also considered anonymity with secure message transmission, the security (and more specifically privacy) achieved by the protocols we present is information-theoretic. This is a stronger security model than the computational security model which has been considered in previous work. This is because our constructions are resilient towards a computationally unlimited adversary and always output the correct and expected result with no probability of failure.

The content of this chapter was also motivated by work carried out in the theoretical design of an electronic code voting scheme. In such a scheme voting codes would be transmitted to voters over a network. However, to retain the secrecy of a vote (to keep private the candidate each voter voted for all voters) - which is one of the most important requirements of any election, the authority which creates the voting codes *should not* be able to correlate a voter with a specific vote (except through a random guess). Therefore, when voting codes are to be sent to the voters, it is very important to achieve anonymity of message transmission. Furthermore, as the transmission network may be corrupted by an adversary presence, it is also important that these codes are transmitted in a secure manner - otherwise the adversary could learn and use voting codes and prevent some voters from voting - thus affecting the true result of an election.

The chapter defines anonymity, the required security properties, and background to the new perfectly secure and anonymous communication protocols in Section 6.1. New protocols which achieve these properties against a passive and an active adversary are presented in Section 6.2 and Section 6.3 respectively.

Various applications in which such protocols could be used are also identified.

6.1 Perfectly Secure and Anonymous Communication Protocols

For the perfectly secure and anonymous communication protocols to be presented, a MIX network [34, 97] communication setting will be considered.

MIX-networks were first introduced by Chaum in [34] and can be used in applications - such as e-voting, which require anonymity. MIX-networks allow for a sender to input a number of (usually encrypted) messages to a mix-network, which then outputs and delivers each message to a random recipient without the sender being able to identify which message

was delivered to a specific receiver.

The dual problem to this is when a single receiver multi-sender communication setting is considered. In this setting, multiple senders input a message to a mix-network and a single recipient receives all messages - unable to identify the sender of any message that was received. This is the setting that will be considered in this chapter. Throughout the chapter we will assume that the two sets of communicating parties (sender(s) and receiver(s)) are connected via an underlying possibly corrupt MIX network.

Solutions to these two problems are very similar to each other and thus a solution to one problem implies a solution to its dual. Because of this, in this chapter we will only focus on one of the two settings.

MIX-networks can be constructed using different tools. One such tool is through the use of a shuffle which can be implemented by using different tools - as described in various work such as [70, 85, 196], which only achieve conditional privacy and anonymity. In this chapter we will not consider this kind of security but will instead present protocols which achieve *unconditional* security against a static t -threshold bounded computationally unlimited adversary in a synchronous network model.

For the protocols to be presented, the same security properties of perfectly secure message transmission protocols - as these have been defined in Section 2.3, are required.

Additionally, the protocols should also be perfectly anonymous. This security property is achieved when for any coalition of t parties, their probability of correctly determining a sender and/or receiver of a message is the same whether the adversary is given the protocol view or only the restricted view. Sender anonymity prevents adversaries from identifying the party which sent a message and applies to a single receiver multi-sender communication scenario. Receiver anonymity prevents adversaries from identifying the party which receives a message and this is considered in the dual single sender multi-recipient scenario.

The term restricted view as used in the above definition refers to the view of the protocol seen by the adversary had no parties been corrupted. Protocol view refers to the adversary's combined view of the protocol as a result of the t parties corrupted by the adversary. Thus even though at most t mix servers may be corrupted, the adversary will never learn (beyond guessing) the value of a secret message and the recipient of a message in a single-sender multi-receiver scenario.

Further to the above, the adversary is assumed to be able to corrupt *any* t parties. This can include the sender(s), the receiver(s) or any nodes of the underlying MIX network. Because of this, in the single sender multi-recipient setting, each receiver has to receive a secret message from the single sender with the requirement that the sender (if corrupted) should not be able to identify the message received by each receiver¹. Similarly for its dual single-receiver multi-sender setting, the receiver should not be able to identify the message transmitted by a specific sender.

In this chapter protocols which assume a passive and active adversary are presented in

¹More precisely, the (possibly corrupt) sender should not increase the probability of determining message receivers beyond that of a restricted view and outcomes which can logically be made. Similar anonymity requirements can also be made in the dual single receiver multi-sender scenario.

Section 6.2 and Section 6.3 respectively. Our solutions - similar to the protocols presented in Section 5.3 and Section 5.4 are based on set systems.

As in [51] the total number of required MIX operations is b . The set of MIX servers is denoted with X and \mathcal{B} will be different subsets of these MIX servers. For the two solutions considering a passive or active adversary, we assume we have an (X, \mathcal{B}) set system, which is an (n, b, t) -verifiers set system in the passive case and in the active case is an (n, b, t', t) -generalized verifiers set system (as these have been defined in Section 5.2). In both cases $B_k = \{MIX_{k,1}, MIX_{k,2}, \dots, MIX_{k,t+1}\}$ and $MIX_{k,1}$ is referred to as “ B_k ’s leader”.

Both protocols we present require the following *new* property to be applied over all blocks \mathcal{B} of MIX servers.

Definition 8. *Set T of t MIX servers is said to be under t -Confinement if all members of set T appear in at most t positions over all blocks of MIX servers used, and this for all $T \subseteq X$.*

Both protocols we present consider the single receiver multi-sender communication setting. Despite this - as stated earlier, the solution for the dual case of the single sender multi-recipient scenario is very similar and this is briefly described for each protocol we present.

6.2 Private and Anonymous Communication Protocol against a Passive Adversary

In this section, we present a perfectly secure and anonymous message transmission protocol for a single receiver multi-sender communication setting when considering a passive adversary. It should be noted that the protocol to be presented is based on an (n, b, t) -verifiers set system which achieves the *t-Confinement* property as this has been defined in Definition 8.

Before formally presenting the protocol, the main idea of the protocol is given.

6.2.1 Main Idea

Each sender will secret share the message they want to transmit as the sum of random shares. Each of these shares will be given to a MIX server in the first block of MIX servers. As shares of messages are transmitted within the network of MIX server blocks, they are permuted and altered - through addition of random shares from the secret sharing of zero (as the sum of random shares). The final block of MIX servers delivers the shares of messages to the receiver that reconstructs the secrets sent by senders.

6.2.2 Formal Protocol Description

For the formal description of the protocol which follows, we assume that sender S_i wants to privately and anonymously send message m_i .

Protocol 8. *Protocol for the Passive Adversary Case*

1. Each sender S_i ($1 \leq i \leq v$) uses a $(t + 1)$ -out-of- $(t + 1)$ Shamir secret sharing scheme to make shares $s_{i,j}^1$ from m_i ($1 \leq j \leq t + 1$) and privately gives share $s_{i,j}^1$ to $MIX_{1,j}$.
2. For $k = 1, \dots, b$:

(a) For each i ($1 \leq i \leq v$), the leader, i.e. $MIX_{k,1}$, chooses shares $s'_{i,j}$ of 0 using a $(t+1)$ -out-of- $(t+1)$ secret sharing scheme and privately sends $s'_{i,j}$ to $MIX_{k,j}$ ($1 \leq j \leq t+1$), and this for each i ($1 \leq i \leq v$). The leader, i.e. $MIX_{k,1}$, also chooses a permutation $\pi_k \in_R S_v$, which it privately sends to all $MIX_{k,j}$ ($1 \leq j \leq t+1$).

(b) Each $MIX_{k,j}$ ($1 \leq j \leq t+1$) computes:

$$s_{i,j}^{k+1} := s_{\pi_k(i),j}^k + s'_{\pi_k(i),j}$$

and if $k < b$, then privately sends $s_{i,j}^{k+1}$ to $MIX_{k+1,j}$.

3. $MIX_{b,j}$ privately sends $(s_{1,j}^b, s_{2,j}^b, \dots, s_{v,j}^b)$ to the receiver.

4. For each i , the receiver computes the message m_i which corresponds to shares $(s_{i,1}^b, s_{i,2}^b, \dots, s_{i,t+1}^b)$.

6.2.3 Security Analysis

We now present the security proof of the passive adversary perfectly secure and anonymous communication protocol.

Theorem 9. *The protocol for the passive-only case is perfectly anonymous and achieves perfectly secure message transmission against any coalition of t passive MIX servers.*

Proof. As the adversary is passive and as shares of zero are added to the original secret shared messages throughout the MIX network, the receiver is able to obtain the original messages correctly. Perfect authenticity is therefore achieved as the receiver can reconstruct all messages correctly. As we are considering a passive adversary, there is no way with which the protocol can fail and thus perfect availability of the communication protocol is always achieved.

It is easy to see that due to the property of t -confinement, the adversary can learn at most t shares of each message. As the remaining shares are unknown to the adversary, the adversary is in no better position to learn the message than without knowing these shares and thus perfect privacy of messages is achieved.

We now prove the perfect anonymity of the protocol, i.e. the receiver cannot tell which sender sent each of the received messages, despite allowing for the co-operation of the receiver with any t adversary controlled MIX-servers. For simplicity of the proof we assume there are only two senders - each transmitting only one message.

Due to Property 3 from the definition of verifier set systems, there exists a block b_k - $1 \leq k \leq b$, free from adversary controlled MIX servers. Because of this, the adversary is unable to learn the secret sharing of 0 which is carried out by the leader of this block of MIX servers. Additionally, the adversary does not learn the mixing of shares carried out within a block - which is also decided by the leader of the block.

We now analyze the view of the adversary before - denoted as $view_{\text{bef}}$, and after - denoted as $view_{\text{aft}}$, this block of MIX servers, assuming that for each of these two blocks the adversary is able to view t shares of each of the two messages. We denote

$view_{\text{bef}} = \{(s_{1,1}^{k-1}, s_{2,1}^{k-1}), (s_{1,2}^{k-1}, s_{2,2}^{k-1}), \dots, (s_{1,t}^{k-1}, s_{2,t}^{k-1})\}$ - with $s_{i,j}^{k-1}$ denoting a share for message $1 \leq i \leq 2$ in the $k - 1^{\text{th}}$ block of MIX servers. In $view_{\text{bef}}$ the adversary is able to identify the message to which shares correspond as we assume that the adversary is aware of any mixing of shares carried out in any previous MIX blocks.

We denote $view_{\text{aft}} = \{(s_{a,1}^{k+1}, s_{b,1}^{k+1}), (s_{a,2}^{k+1}, s_{b,2}^{k+1}), \dots, (s_{a,t}^{k+1}, s_{b,t}^{k+1})\}$ - with $s_{x,j}^{k+1}$ denoting a share for one of the two messages in the $(k + 1)^{\text{th}}$ block of MIX servers where x can take either one of the two values a or b . The view $view_{\text{aft}}$ is equal to one of the following two values - where $1 \leq j \leq t$, depending on the permutation used:

- (a) $s_{a,j}^{k+1} = s_{1,j}^{k+1} + (s_{a,j}^{k+1} - s_{1,j}^{k+1})$ and $s_{b,j}^{k+1} = s_{2,j}^{k+1} + (s_{b,j}^{k+1} - s_{2,j}^{k+1})$, **or**
 (b) $s_{a,j}^{k+1} = s_{2,j}^{k+1} + (s_{a,j}^{k+1} - s_{2,j}^{k+1})$ and $s_{b,j}^{k+1} = s_{1,j}^{k+1} + (s_{b,j}^{k+1} - s_{1,j}^{k+1})$.

As the secret sharing of 0 and the permutation of the MIX block where the adversary is not present is *random*, the adversary cannot distinguish between the two cases and thus perfect anonymity is achieved. Without loss of generality, the same proof can be extended to any number v of messages sent by v different senders. \square

Collary 2. *A protocol and proof similar to the one presented will be valid for the dual case of receiver anonymity where a multiple-receivers single sender scenario is considered.*

The differences in the dual protocol is that the single sender will secret share all messages in the same manner as carried out by the multiple senders in the presented protocol. The number of these messages will equal to the number of receivers. The sender will submit these shares to the first block of MIX servers. The mixing and alteration of shares occurs in the same manner within the MIX network as described in the presented protocol. An additional difference is that the MIX network will deliver a single message to each receiver with each message delivered only once (as opposed to a single receiver accepting all messages).

Note: Protocol 8 cannot guarantee privacy of messages if we do not use the t -confinement property².

In Appendix E, a general discussion for (n, b, t) -verifiers set systems which satisfy the required t -confinement property is given. However, as in the previous chapter, we provide an example construction. Primarily from the available constructions of (n, b, t) -verifiers set systems $O(t^2)$ number of MIX servers are required to ensure a polynomial number of MIX server blocks. Additionally, at least $t + 1$ blocks are required to ensure that anonymity is achieved and the size of each block should be at least $t + 1$ number of MIX servers to ensure that privacy of the message transmitted is achieved.

An example of such construction is one where there are $t + 1$ blocks each of size $t + 1$ MIX servers with each server used only once. This construction is equivalent to the construction

²If t -confinement is not used, secrecy of messages is broken in the following scenario. The adversary is present in t MIX servers of the first block of MIX servers, occupies the first position of this block and also occupies in the second block of MIX servers the position not occupied by the adversary in the first block of MIX servers. Secrecy is broken as the adversary learns the mixing of the message shares and the secret sharing of 0 as described in the protocol. The adversary thus learns t shares of each message from the first block and learns the remaining share in the second block of MIX servers. It is easy to see that secrecy can be broken in similar scenarios also.

described in Lemma 6³.

6.3 Private and Anonymous Communication Protocol against an Active Adversary

The solution for a perfectly secure and anonymous communication protocol when considering an active adversary, is very similar to the protocol presented in the earlier section where a passive adversary was considered. However, it should be noted that the protocol we present is based on a generalized $(n, b, 3t, t)$ -verifiers set system which achieves the t -Confinement property as this has been defined in Definition 8.

Before the protocol is presented and as the adversary is active, we first identify ways in which adversary controlled MIX servers can deviate from the definition of the protocol.

6.3.1 Active Adversary Deviation Methods

These deviations can occur in one (or more) of the following ways:

1. If the adversary controls the leader of a MIX block - by being present in the first position of that MIX block, the adversary may secret share a value different to 0.
2. If the adversary controls the leader of a MIX block, different permutations may be sent to MIX servers.
3. Adversary controlled MIX servers can permute the shares of messages using a different permutation to that sent by the first MIX server of a block.

We now describe tools which will be used to overcome the *respective* deviations given above.

1. To overcome this, the leader will send the secret sharing of zero (all of the constructed shares) to all MIX servers of a block. Executing Byzantine agreement (such as one of schemes described in [63, 64, 109]), MIX servers of the same block will ensure that the *same secret sharing* was sent to all MIX servers and that 0 was *the secret shared value*. If this was not the case, the block of MIX servers will *not* be used in the MIX network.
2. To overcome this, Byzantine agreement needs to be carried out to ensure that the *same permutation* is sent to all MIX servers of the same block. As before, if this was not so, the block of MIX servers will *not* be used in the MIX network.
3. We do not know of an existing cryptographic scheme which allows for such a check to be carried whilst retaining privacy of shares. Containing the adversary and thus the number of errors the adversary can create seems to be the best option. Using the property of t -confinement upon the MIX network, restricts the adversary to create at most t errors to message shares over the MIX network. In such a case, our proposed protocol should be able to correct these errors.

³As in the previous chapter, a construction using a $(2t + 1, \binom{2t + 1}{t + 1}, t)$ -verifier set can be used for the above protocol, but an exponential number of MIX server blocks will be required.

Before presenting the protocol, we inform the reader that the main idea of the protocol is very similar to that of Protocol 8.

The only thing different in this protocol, is that as before, shares of messages are altered through the addition of random shares from the secret sharing of zero - only for the protocol to be presented zero is secret shared using a Shamir secret sharing scheme[163].

6.3.2 Formal Protocol Description

We denote with m_i the message sender S_i wants to send privately and anonymously. As in Protocol 8, the concept of t -Confinement is also required. We use $B \geq 3t + 1$ to denote the size of MIX server blocks. We now formally present the protocol.

Protocol 9. *Protocol for the Active Adversary Case*

Initial setup - For each block of MIX servers, the leader of each block secret shares zero using a $(t + 1)$ -out-of- (B) Shamir secret sharing scheme - where B denotes the size of blocks, selects a permutation $\pi \in_R S_v$ and sends both to all MIX servers of the block. Byzantine agreement decides the correctness of the secret sharing and the permutation and whether the block of MIX servers will be used in the MIX network. Blocks where Byzantine agreement fails will not be used. We denote as b the remaining number of MIX blocks. The permutation and secret sharing of zero verified through Byzantine agreement for all blocks will be used in the following description.

1. Each sender S_i ($1 \leq i \leq v$) uses a $(t + 1)$ -out-of- (B) Shamir secret sharing scheme to make shares $s_{i,j}$ from m_i and privately gives share $s_{i,j}^1$ to $MIX_{1,j}$.

2. For $k = 1, \dots, b$:

(a) Each $MIX_{k,j}$ ($1 \leq j \leq B$) computes:

$$s_{i,j}^{k+1} := s_{\pi_k(i),j}^k + s'_{\pi_k(i),j}$$

using the definition of π_k and $s'_{\pi_k(i),j}$ initially verified by Byzantine agreement. If $k < b$, $MIX_{k,j}$ privately sends $s_{i,j}^{k+1}$ to $MIX_{k+1,j}$.

3. $MIX_{b,j}$ privately sends $(s_{1,j}^b, s_{2,j}^b, \dots, s_{v,j}^b)$ to the receiver.

4. For each i , the receiver uses error correction (if required) to compute the message corresponding to the shares $(s_{i,1}^b, s_{i,2}^b, \dots, s_{i,B}^b)$.

6.3.3 Security Analysis

Theorem 10. *The protocol for the active-only case is perfectly anonymous and achieves perfectly secure message transmission against any coalition of t passive MIX servers.*

Proof. As the property of t -confinement is used, this ensures that the adversary can learn at most t shares of each secret message. As each message is secret shared using a $(t + 1)$ -out-of- (B) secret sharing scheme perfect privacy for all messages is achieved (a direct consequence of t -confinement). Due to Property 3 from the definition of generalized verifier set systems, there exists a block free from adversary controlled MIX servers. The adversary does not learn

the mixing for this block and thus anonymity is achieved in a similar manner as in Protocol 8. The protocol achieves perfect reliability as perfect authenticity and perfect availability of all messages is achieved through the use of t -confinement which limits the number of errors the adversary can create to at most t . Using results from [56] and a block size $B \geq 3t + 1$ ensures the correction of any errors (at most t) the adversary creates and the unique decoding of the secret messages. \square

Collary 3. *A protocol and proof similar to the one presented will be valid for the dual case of receiver anonymity where a multiple-receivers single sender scenario is considered.*

The differences in the dual protocol are exactly the same as the dual case considering a passive adversary. The single sender will secret share all messages, submit them to the MIX network, which will mix and alteration shares in the same described manner and a single message will be delivered to each receiver with each message delivered only once (as opposed to a single receiver accepting all messages).

As in the earlier protocol, we provide an example of a generalized (n, b, t', t) -verifiers set systems which satisfies the required t -confinement property. Primarily from the available constructions of generalized (n, b, t', t) -verifiers set systems $O(t^2)$ number of MIX servers are required to ensure a polynomial number of MIX servers blocks. Additionally, at least $t + 1$ blocks are required to ensure that anonymity is achieved and the size of each block should be at least $3t + 1$ number of MIX servers to ensure that privacy of message transmitted is achieved as well as the unique re-construction of the secret message by the receiver - using error correction.

An example of such construction is one where there are $t + 1$ blocks each of size $3t + 1$ MIX servers with each server used only once. This construction is equivalent to the construction described in Lemma 5.

6.4 Practical Applications of Anonymous and Perfectly Secure Message Transmission Protocols

The protocols presented could be used in various applications where the secrecy of messages as well as the anonymity of senders or receivers (or both) is required. During the research period of the thesis, the protocols were used to achieve anonymity for electronic code voting purposes as described in the introduction of the chapter.

Beyond theoretical applications⁴ work should be carried out to achieve the implementation of such protocols upon real world networks - such as the Internet. This would achieve the anonymity of Internet users and could result in various useful applications - such as anonymous peer to peer communication systems, secure and anonymous email, instant messaging or voice over IP applications, amongst others.

What is interesting to note about the anonymous and perfectly secure message transmission protocols proposed in this chapter is the fact that the proposed protocols make no use of expensive public key operations or even symmetric key operations and thus the mixes are merely network bound when it comes to how fast they can perform mixing. The advantage of

⁴Such protocols could be of use to other cryptographic protocols also.

this is that some network devices (such as routers), may be turned into mixes in a system which uses the proposed protocols without being endowed with more powerful CPUs⁵.

⁵It should be noted that the proposed protocols assume that the mixes are connected between them across different blocks using private channels. However if these are not available on a network, then these can be simulated using various perfectly secure message transmission protocols such as those presented in Chapter 3 and Chapter 4

Chapter 7

Implementing Secure Message Transmission Protocols

This chapter studies whether secure message transmission protocols could possibly be implemented and used upon networks which exist in practise. We refer to message transmission protocols in general and not only the protocols proposed in this thesis¹. The theoretical network models considered in cryptography and previous chapters differ from networks which exist in practise, as the diversity and abundance of node disjoint paths may not be present as assumed in theory. Furthermore, some of these models consider synchronous networks which are different to asynchronous networks used in practise. We therefore explore the possibility in which secure message transmission protocols can be implemented upon practical networks which operate under such constraints.

Such networks include the Internet and other networks such as ad-hoc networks, sensor networks and military networks. The scope in which secure message transmission protocols could be used and their importance is discussed. Issues which exist and avert the implementation of such protocols for some of these networks are identified and described. Since the Internet can be considered as the most important network today, the main focus is on the implementation of such protocols on the Internet. Despite these issues, we identify possible alternative ways with which secure message transmission protocols can be implemented upon various other networks.

7.1 Threat Model

For this chapter, we assume that the goal of the adversary is to break the privacy and/or authenticity and/or availability of a secure message transmission protocol - therefore any secure message transmission protocols should explicitly aim to defend against these attacks.

We assume that the routing protocol is a trusted process which cannot be affected by the behaviour of the adversary. This may be because the routing protocol code cannot be altered or because any false actions carried out by the adversary (such as advertising false paths) will be identified and corrected through the collective execution of the routing protocol by non-compromised nodes.

We also ignore any other attacks the adversary may carry out - such as flooding the network

¹Protocols proposed in this thesis assume synchronous networks which cannot be used for certain networks (such as the Internet) used in practise.

with traffic and causing congestion throughout the whole network.

It is important to note, that in this chapter we also consider a similar adversary model to that of the theoretical work presented in the earlier chapters. To be more precise, we consider threshold bounded adversaries - in other words adversaries which are bounded in the number of routing nodes they can corrupt. When considering the Internet and other networks which exist in practise, this could equate to the number of routers an adversary can take control of. This limit may be because of a lack of manpower (if physical access to a node is required), cost (either financial when considering bribing or computational if considering attacks such as password cracking) or risk of prosecution (taking control of too many nodes may trigger security mechanisms which identify the location and identity of the adversary).

Taking control of nodes can be carried out using various attacks. One is through breaking in to a server room, altering the Linux root password (by booting in single user mode) and taking control of a machine². Another attack is through the use of rootkits - when an adversary is able to obtain privileged access to a node through the exploitation of a known (and not secured) vulnerability or by obtaining/guessing the password³. This kind of attack has been reportedly used in the "Greek Watergate" [176, 182] and the Sony BMG copy protection rootkit scandal [11, 30]. The case which assumes that when an adversary corrupts one node also gives the adversary the ability to control other nodes of similar implementation (i.e. nodes of the same model/company or running the same operating system) equates to the general adversary model [96] and the colour adversary problem as studied in [54] which is not considered in this thesis.

7.2 Scope and Importance

Protecting the secrecy of a message to be transmitted from a sender to a receiver has been an important goal for a very long time. Equally important in the transmission of messages is the authenticity as well as the availability of protocols themselves.

In our current technologically dependent civilization, most communication occurs over networks. The public switched telephone network (PSTN) [110] was one of the first networks allowing for communication to easily and swiftly occur between people in a simple and accessible manner. Nowadays, most of the communication which takes place does so over the Internet. From email to voice over IP, data transfer to secure bank transactions, the Internet can be considered as the most important network today. In fact, the Internet has taken over PSTNs as nowadays many phone companies only use Internet protocols [61, 76, 94].

Securing communication over the Internet is thus of great importance in the field of network security. The initial and main emphasis has been to use cryptography to protect the secrecy and integrity of communication between two communicating parties - a sender and a receiver.

²Similar attacks which allow an adversary to take full control of a node if they are physical present to where the systems are located can also be carried out on systems running Windows or OS X operating systems and even Cisco systems.

³Even though network nodes may be running the same operating system and thus prone to the same vulnerabilities, such attacks may allow an adversary to take control of a certain number of network nodes instead of all as passwords may (and should) be different, or because of security updates which patches vulnerabilities for some but not all nodes - mainly because these may not have been carried out because of time constraints or because security updates on a specific node were inadvertently overlooked.

This has mainly been achieved through an encryption tunnel - implemented using virtual private networks (VPNs) or Secure Socket Layer (SSL) among other solutions, to connect the two parties.

Cryptography alone cannot guarantee network security and security of communication. Two of the most severe network threats are those of Denial of Service (DoS) attacks [127, 130] and its distributed version, distributed Denial of Service (DDoS) [128]. Such attacks affect the availability of networks - thwarting the network from successfully delivering data to their intended recipients.

DoS is not a new attack and is not unique to the Internet either. This is because it appears in other networks also. According to [180] on the 5th of August 1914, the cable ship *Telconia* lifted from the bed of the North Sea, German overseas telegraph cables which connected Germany to the outside world. After this, German diplomatic communications and other military signals had to be transmitted using wireless signals - which could be intercepted. These wireless signals were later encrypted, but due to their wireless nature, they could still be jammed - thus extending the DoS attack in this alternative mode of communication. This was a very important event which highlights the use of information for strategic political objectives during World War I, where the British used a DoS attack against the German telegraph cable network. As Kuehl puts it in [103] "This strategic information operation not only cut Germany's military C3 links to its forces worldwide (at sea and in its colonies) but also-and more importantly meant that the neutral countries of the world, most especially the United States, saw the war through London's filter."

As seen by more recent DoS incidents such as [10, 86, 186, 199], this kind of attack can be very disruptive. In the DDoS attack reported in [131], over several weeks, websites of Georgia's government were the target of an organized number of Denial of Service attacks. The unavailability of crucial government websites severed communication from the Georgian government in the early days of the Georgian-Russian conflict [12]. This was a period of time critical in the events that were transpiring and where the Georgian government had a vital interest in keeping information flowing to both the international public and to its own residents. The same attack also targeted the financial sector of Georgia, when key Georgian commercial Internet servers [183] also suffered from DoS attacks. As a result, these servers could not accommodate legitimate visitors. In fact, during the attack, in an effort to protect against theft of data, banks shut down their servers [43]. The attacks on these financial servers caused a crippling effect on Georgia's economy [37].

A similar DoS attack upon the infrastructure of a country was that which occurred in April 2007 in Estonia [16, 192] reportedly linked to Russia. The DoS attack was rumored to have occurred in retaliation to Estonian authorities removing a bronze statue of a World War II-era Soviet soldier from a park [156]. The attack targeted Estonian organizations such as the Estonian parliament, banks, ministries, newspapers and broadcasters. It was so intense that it has been described as the first war in cyberspace, which came close to shutting down the country's digital infrastructure [181].

DoS attacks are thus very serious and such an attack has the potential to cripple entire organizations - whether that is a website, company or country. Indeed, they are considered as

one of the main forms of future cyberwarfare attacks which could occur [83]. Ways in which responsible parties could be held liable for their actions is something difficult to define but is being discussed [198].

The importance of defending against such attacks has only been highlighted by events that occurred towards the end of 2010. The publication of US government diplomatic transcripts by the website Wikileaks in late November 2010 triggered a series of network attacks. These events were mainly Denial of Service attacks and came in two related incidents. In the first case, Denial of Service attacks were carried out by an organization - most likely a country's government, against any Internet Service Provider (ISP) which hosted the Wikileaks website [115, 177]. This resulted in the specific ISP to not only be unable to service the Wikileaks website (in such cases the Wikileaks site was unavailable to Internet users) but additionally, such ISPs found it difficult to service their other customers. Because of this, various ISPs decided to terminate the hosting of the Wikileaks website and thus avoid the DoS attack against Wikileaks which affected their own networks so much. Because of this, the Wikileaks website had to change to a different ISP every time this occurred. The other wave of Denial of Service attacks related to this incident were the attacks carried out by group "Anonymous". This group of hackers carried out Denial of Service attacks against online transaction payment firms such as Paypal and Mastercard which (possibly under government influence) stopped processing online donations made by Internet users for the Wikileaks website. Because of these attacks, these firms had difficulties in running a regular service and servicing their customers [10].

In February 2012, DoS attacks were carried out - again by group "Anonymous", against the Department of Justice, the Recording Industry Association of America, Motion Pictures Association of America and Universal Music. This was in response to when the file sharing website Megaupload.com was taken offline and charged with copyright infringement by a U.S. grand jury [86, 132, 172].

Overall, the effects of DoS attacks emphasize the importance of proactive reliability of networks - for which organizations depend on their reliable and resilient operation to conduct normal business operations.

Secure message transmission protocols are a candidate tool to use so as to defend against Denial of Service attacks. This because of the multitude and diversity of paths they use. This could possibly ensure that at least one of the paths (which connect a sender and a receiver) is not affected by such an attack - thus allowing for communication to be successful. Secure message transmission protocols could also be useful in various other applications such as in emergency scenarios. As discussed in Section 4.2.2 such scenarios can include after a natural or man-made disaster, during a conflict/warfare amongst others. In such situations, it is imperative that information sent by a sender should reach a receiver(s). In the example scenarios given, the sender can be considered to be a central government of a country and the receivers will be its citizens. In Section 4.2.2 specific examples such as the Haiti earthquake in 2010 were given. These in addition to the denial of service examples highlight the importance of secure message transmission protocols and their availability for use over networks such as the Internet.

7.3 Implementation Issues

The goal of work which studies secure message transmission protocols is not solely of theoretical interest. One of the objectives for such work was to design efficient protocols which could possibly be implemented on networks. Such networks include the Internet, and these protocols could increase the reliability, resilience and resistance guarantees of the Internet against DoS attacks - as well as achieving secrecy of communication in some cases. Indeed, the results of this work seem to be one of the most effective ways to ensure continuity of operation whilst a network attack is taking place. This is because multiple paths are used and this could ensure that enough paths are free from network areas which are either suffering from congestion or some kind of attack (such as a DoS attack) to allow for communication to take place effectively.

In this section, we explore the feasibility of achieving these results on current networks - mainly focusing on the Internet which in the history of the civilized world can be considered as the most important network. We first overview work which has been carried out to implement message transmission protocols. We then identify differences between past and present routing methods available on the Internet and identify how these affect the possibility of achieving node-disjoint paths across ISP networks.

7.3.1 Implementations of Message Transmission Protocols

We first review work which has carried out implementation of concepts used in message transmission protocols.

In [100] the authors used the concept of disjoint paths to achieve anonymous communication upon a peer-to-peer overlay network. Contrary to PSMT protocols, the disjoint paths that were used were vertex-disjoint paths. Additionally, the disjointness of the paths only considered edges of an overlay peer-to-peer network. This did not take into account the actual physical network upon which the overlay network was based. Because of this, disjoint paths in this implementation could in fact share common nodes and edges of the underlying network. As the adversary model considered was able to corrupt edges of the overlay network it was considered secure in the implementation. Despite this, the implementation was later shown to be insecure in [184].

In the context of PSMT protocols, the implementation of [100] is not a secure implementation. This is because PSMT protocols require node-disjoint paths. Additionally, if any sort of overlay network is considered, it should take into account the underlying network of this overlay and node-disjoint paths should be truly disjoint with regards to the overall network used.

7.3.2 Past Internet

Up until the Internet explosion (mid 1990s) different useful network commands were valid on the Internet.

The IP options of loose/strict source routing [42] were such options. These options allowed a sender of a packet to partially or completely specify the network path a packet should traverse through the network. By providing a list of network nodes (routers which are identified by their IP addresses) data packets transmitted by a sender should pass through, this allowed for data to bypass routing protocols and cross the specified network path.

Using this capability, could possibly have allowed senders to create disjoint network paths

upon which data could be sent. As the use of node disjoint network paths is a necessary condition for PSMT protocols, this could possibly have allowed for PSMT protocols to be deployed and successfully used upon this past configuration of the Internet - provided users know which IP's to specify⁴.

7.3.3 Present Internet

The current structure and operation of the Internet as a whole is different to that of the past. It seems that most networks comprising the Internet seem to have the following properties:

1. The use of any IP options is prohibited.
2. ISPs use specific routing policies.
3. The evolution and growth of the Internet is driven by economic factors.
4. Border Gateway Protocol (BGP) routing policies affect the availability of disjoint paths.

7.3.3.1 The Use of Any IP Options is Prohibited

The use of any IP options (including loose/strict source routing) is prohibited and IP packets with any IP options are either dropped or do not have their options processed. A number of reasons are behind this decision.

One of these is because using IP options, one can carry out various security attacks. As stated in [39], a hacker can carry out a successful denial of service attack upon a router by sending large streams of packets with IP options. In this way, the router under attack will need to process a large number of packets with IP options. Because of this, much of the router's processing capacity will be consumed with packets originating from the attacker. This will result in the limited ability of the router to perform the forwarding process of packets which do not originate from an attacker.

Similarly, the processing of any IP options requires router processing time. This decreases the throughput of the forwarding process carried out by routers. As one of the main competitive aims of ISPs is to offer ever greater transmission speeds to clients - in the quest for a faster Internet, ISP networks are generally configured so that routers do not process the IP options of any packets. This leaves the processing power of routers to solely focus on the forwarding process - thus making networks faster.

Additionally, it was found that the use of IP options - specifically the use of loose/strict source routing, could lead to various security threats [38] such as session hijacking [195]. This could be achieved when a hacker is able to intercept the stream of data packets from a sender S to a receiver R . The hacker could then add his/her own address to the set of network nodes which comprise the network path of the communication. Without checking for this attack, this would allow a hacker to participate in the communication between S and R - as IP packets pass through a machine controlled by the hacker.

⁴In the past, various tools could be used to help users identify part of the structure of a network and the IP addresses of the comprising nodes. Such tools include ping, traceroute or other tools such as PathPing [124] which may combine ping and traceroute for network discovery purposes. It should be noted that some of these commands/tools may no longer function as expected due to security steps implemented by network administrators to prevent network details from being identified.

Another attack which is possible - through the use of loose source routing, is that of source address spoofing [125]. The purpose of this attack is to either gain access to resources that only accept requests from specific source addresses, or to hide the source of an attack. This could be implemented in a similar manner to the previous attack. All the attacker has to do, is to spoof one of the permitted source addresses and to set the loose source routing option with an address that the attacker has access to. This will force the response to return to the attacker's network and thus a successful attack will occur.

Discussions with network administration staff at British Telecom, confirm that packets with IP options are either dropped or not processed by network routers. This for reasons that have been described.

7.3.3.2 ISPs Use Specific Routing Policies

Most large ISPs which own and operate their own network infrastructure implement traffic shaping policies within their networks. In traffic shaping [58] network traffic is controlled and managed in various ways relative to the policies implemented by an ISP. This is done to optimize the performance of the network, to improve latency and to increase the usable bandwidth of the overall network. Additionally, traffic shaping allows for ISPs to identify what their subscribers use or view on the Internet. This could be used to identify services that in turn the ISP can provide to their subscribers.

Traffic shaping policies are also used to provide Quality of Service guarantees for applications [58]. Alternatively, they can be used to discourage use of certain applications - such as peer-to-peer applications [197]. In the case of peer-to-peer applications, the volume of traffic that can be generated from their use can amount to a large percentage of the total network traffic on an ISP network. This makes it difficult for ISPs to cope with the network demand and provide the same bandwidth guarantees to all users [9]. Additionally, some users use peer-to-peer applications for the illegal file-sharing of copyright material and some ISPs prefer to discourage such activity from occurring upon their networks [95].

Within the current Internet, large ISPs also use traffic shaping to implement their own specific routing policies. This is done so that traffic within an ISP network is routed in a specific and efficient manner. One such policy is to use multi-protocol label switching (MPLS) and a central core of routers [71] which implement the routing process in a very efficient manner. This policy directs most traffic flows within this central network core of routers - making these routers very important for the operation of an ISP network.

Because ISPs use such traffic shaping policies, it is very difficult for a network programmer who requires node-disjoint paths to by-pass the routing of traffic carried out by ISPs. This because the routing policies within the central core network of ISPs cannot guarantee that data streams transmitted by the sender and intended to be transmitted on disjoint network paths will not converge to a single common node within the central core of routers. This goes against the key requirement of node-disjoint network paths in the implementation of perfectly secure message transmission protocols. Indeed, the work carried out in [178] looked into path diversity of Internet topologies. One of its key conclusions was that within an ISP network, there is a high potential for path diversity. A further comment was that the utilization of this path diversity is limited and this is due to routing policies implemented by ISPs.

7.3.3.3 Economic Reasons affecting Internet Evolution

The evolution of the Internet has mainly been driven through economic factors. ISPs design and build their networks in the most economic way with which they will be able to offer a required quality of service to their customers. These choices, most likely affect (limit) the diversity of node disjoint paths amongst ISP networks - and the Internet in general. An example of such choices is specific routing policies ISPs use - as outlined in the earlier section.

Further to this, ISPs cooperate with each other under competitive pressure. BGP routing policy and ISPs which border each other negotiate contracts between them which defines how traffic is forwarded amongst networks of different ISPs. Policies implemented by ISPs - mainly driven by revenue, may thus configure their routers in such a way that they prune away most of the path diversity which may be possible on their networks. This will mainly be done because ISPs have decided that this is the best policy to use which will make the most money by forwarding traffic on specific links. This may be the case when data originating in the network of an ISP has to be transmitted on another ISP's networks. In most cases, ISPs will choose to use as few ISPs as possible and mainly select the cheapest options. In such a case, path diversity is lost on the Internet as network links between ISPs are lost due to economic reasons. More details on BGP routing policy and how this can affect the abundance of path diversity are also given in the next subsection.

Other economic reasons which may affect the diversity of network paths can include (amongst others) the wealth of nations. Countries which do not have the financial capability to invest in their digital infrastructure will have a lower number of nodes present in their country. Because of this, any required paths that have to traverse such a country may not be node disjoint - as required, due to the bottleneck of the limited number of nodes present in such a country. Despite this, the effects of this reason may possibly change in the future. As time passes and countries realize the importance of a large and efficient digital infrastructure more money may be invested in enhancing this. Thus the effects of the bottleneck due to limited nodes in poor nations may not be so apparent.

Furthermore, as people use the Internet and demand more from their use of the Internet, economic incentives for ISPs to provide multiple paths are likely to grow stronger in the future as the demand for performance and robustness increase, and customers are willing to pay for value-added Internet services as described in [89].

7.3.3.4 BGP Routing Policies Affect the Availability of Disjoint Paths

The Border Gateway Protocol (BGP) [155] is a routing protocol used to exchange reachability information across autonomous systems which comprise the Internet. In this section we describe how the way ISPs use BGP can affect the number of disjoint paths in the middle of a network which connects a sender and a receiver.

As described in [31, 201], when BGP was first introduced, it was a fairly simple protocol where routers would choose the route to forward data packets based on the minimum path length. As the Internet grew in size becoming more and more commercialized, ISPs wanted a way to control the way traffic was routed within their networks for various reasons - such as economic reasons amongst others. Overtime, alterations were added to BGP to allow ISPs to

control route selection (where packets should be forwarded) and propagation (who to export routes to). Eventually, this allowed for an ordered list of attributes to be created by network administrators. Route advertisements and router forwarding decisions were now chosen based on the most desirable attribute(s) of this list. As a result of all these modifications, BGP has evolved into a complex protocol with an array of tunable knobs and complex cross protocol interactions which make it highly subject to a variety of problems such as misconfigurations amongst others.

The range of BGP policies used by operators can in general be classified into four main categories - all of which can affect the availability of disjoint network paths between autonomous systems. We give an overview of these below.

The first of these is a business relationship policy [44] based on the economic and/or political relationships ISPs share amongst their neighbouring autonomous systems. Forwarding of data packets mostly reflects agreements or relationships ISPs have with their neighbours, with an ISP preferring to forward data through their customers (ISPs which pay money to ISP to forward their data) as this generates income for them - as opposed to forwarding data on their provider's networks which costs an ISP money. Furthermore, ISPs will in general choose to not export routes learned from a provider to other providers as there is no economic incentive to do so. Because of such policies, not all possible routes are available and known to routers in their routing tables - affecting the availability of disjoint network paths which could connect a sender and a receiver.

The second of these categories is traffic engineering policies [151]. As discussed earlier, ISPs use their own specific routing policies to route traffic within their networks in the manner they deem to be most effective for various reasons. These include to avoid congestion, to implement load balancing policies, to provide good quality of service amongst others. Because of this, ISPs will prune away disjoint paths within their own networks and only use the paths they decide and choose to best implement their policies.

Scalability is also a factor which plays a role in BGP policies. Such policies aim to protect routers from failing through overloading a router's processing capability or overloading a router's memory (this may be more relevant to smaller ISPs with less capable systems). To ensure this, ISPs may limit the router's table size through various methods such as using aggregation to filter long prefixes. Because of this and other similar actions used to achieve scalability, not all available routes are known to routers and thus are not used in the BGP forwarding process. This further limits the availability of disjoint network paths to what their actual potential number could be.

Security reasons also play a role in BGP policies. ISPs may choose to not advertise certain routes to some ISPs for reliability or quality of service purposes - such as to avoid excessive traffic entering their network. Misconfigurations of BGP attributes - possibly by inexperienced network administration staff, can cause alterations to the router forwarding process [116]. Furthermore, security policies are there to protect the network during malicious or accidental attacks - such as during a Denial of Service attack. As an example, if a DoS attack is occurring on an ISP network, the ISP can dynamically change the way they advertise their network in an effort to avoid such an attack from continuing on their networks [185]. Because of this dy-

dynamic nature of BGP, a path which exists and which a person may be using for secure message transmission purposes may not be available when such updates occur.

Based on this brief description of BGP routing policies, it seems very difficult (if not impossible) to obtain the required number of node disjoint paths within networks for the purposes of secure message transmission protocols. To achieve the multiple paths required for secure message transmission protocols, investment in infrastructure will have to be carried out by some ISPs (to purchase systems with more memory which will be able to handle the required routing table data). Furthermore, ISPs will have to change their routing policies and remove the ordered list of attributes routers currently being used for the selection of their forwarding process.

It is obvious that given the heavily privatized and commercialized nature of ISPs, most (if not all) of the above will unlikely be carried out without any profitable economic incentives for ISPs. ISPs will possibly have to charge their clients for the availability and use of node disjoint paths - which will require a different economic model to that under which ISPs currently operate.

7.3.4 Alternative Internet Options and their Issues

Due to the current structure and configuration of most networks which comprise the Internet, achieving disjoint network paths on the Internet is extremely hard.

One way with which disjoint network paths could possibly be achieved is by using multi-homing [178]. In multi-homing, an Internet user is connected to the Internet by more than one ISP. This could possibly allow for the presence of disjoint network paths as different (large/major) ISPs have their own networks with their own infrastructure. If a network programmer requires a number of disjoint network paths, they should have an equivalent number of different connections to the Internet serviced by different ISPs. By sending data over these different connections, it is expected that data transmitted over a connection serviced by a specific ISP will *only* be forwarded by network nodes (routers) which belong to the network of the ISP.

Despite this, as described in the BGP routing policy section, ISPs cooperate between them [80] and it is the norm for data originating from one ISP to at some point possibly be carried over infrastructure of another ISP. Because of this, even when using multi-homing, this cannot guarantee that data transmitted on different links (and intended for disjoint paths) will not converge upon the same network node of an ISP's network. Multi-homing will thus not always allow for use of *truly* disjoint network paths mainly because multi-homing cannot guarantee a solution to the path diversity problem. This is because multi-homing only accounts for edges of the network when bottlenecks of common nodes amongst the "different" paths may occur in the core of a communication network - due to BGP routing policy, network topology and other possible reasons.

Similar to multi-homing, one can use different ways to access the Internet - such as using computers, 3G enabled mobile phones amongst other solutions. This uses technologies with different connectivity and access methods to the Internet to achieve disjoint network paths. However, similar to the case of using multi-homing, this cannot guarantee that within the core

network of ISPs, traffic shaping policies will not converge data streams - intended to be sent upon disjoint network paths, to pass through and be forwarded by a common node.

One way with which Internet users can check for path diversity, is by using commands such as traceroute (and its associated parameters). When trying to infer router-level topology by software tools such as traceroute, routers may not reply to ICMP packages or may reply with different misnamed domain names [202]. Reasons for this not only include performance purposes (to prevent excessive router processing) but could also include security reasons - ISP topological data are in general kept confidential [129, 179].

It is of the author's opinion that one further reason why this security may be imposed is to possibly prevent network information - such as network topology, from being learned by competitors, for example other ISPs. Additionally, this could be done to possibly ensure the secrecy of important network information to Internet users and possibly malicious parties. Due to the rising threat of network attacks and cyber terrorism this is very important. Keeping this information as secret as possible, prevents malicious users from targeting key and important network nodes in the topology of an ISP's network.

Taking all of the above into account, it seems that for the Internet, secure message transmission protocols are a remote practical possibility. This is because the way the Internet has evolved and is configured makes it very difficult to achieve node disjoint network paths - a key requirement for such protocols.

To implement such protocols in the Internet one may have to consider a more relaxed security model which would allow for specific nodes to be shared and to be common between the "node disjoint" network paths connecting a sender and receiver. The need for this may be due to the requirement of a greater number of links (edges) in a sparse network. These "common" nodes though would have to be nodes that are deemed to be secure and which cannot be corrupted by an adversary. Examples of such nodes can include core network nodes upon which most security is provided. Such nodes will thus be housed in highly physically secure server rooms, have regular maintenance windows for update of security patches, run behind a secure firewall and possibly an intrusion detection system to keep them secure from any remote threats.

7.3.5 Implementation Upon other Networks

Our focus so far has been on the Internet and regular Internet users. From the above, it seems virtually impossible for regular Internet users to achieve node disjoint paths across ISP networks. Contrary to this, ISPs can achieve node disjoint paths in different ways. One of these is using Multi-Layer Protocol Switching (MPLS) [78, 142]. Using MPLS, ISPs can control the routing of data packets and can ensure that node-disjoint network paths can be achieved within their networks. Taking advantage of this capability can result in new secure services which ISPs can offer to their customers - with such services being more resilient and less prone to network attacks. Implementation of such services could follow a model similar to that described in [6] which describes sending information (shares of data) across multiple paths which are implemented using MPLS.

In conversations the author held with British Telecom (BT) staff, examples of network

capabilities similar to the above were described as being available to BT clients. To be more specific, for clients who required communication between the United Kingdom and the United States of America, BT had the capability to offer them two node disjoint network paths (at a premium of course) for reliability of message transmission purposes. The way this was implemented was through the virtual “colouring” of network nodes of the underlying network of the cross Atlantic link(s). All nodes were coloured (labelled as) white, but half were also coloured red whilst the other half were also coloured blue. If a BT client did not want to use the service of disjoint paths, then their data could be forwarded by any white coloured node (any node in the network). For BT clients who requested the use of disjoint paths, their data would be transmitted both on blue network nodes and on red network nodes. Their data was thus transmitted on a blue wire and on a red wire - with both wires being node disjoint to each other. With this form of transmission, greater reliability guarantees were offered to clients.

Other networks besides the Internet also exist, upon which secure transmission protocols could possibly be implemented. Networks such as sensor networks, ad-hoc networks and military networks implement their own routing protocols. Relative to the network and application setting, such networks could be configured so that the routing could allow for multiple node disjoint paths to connect communicating parties. Work that uses this capability includes [111, 119, 171] for ad-hoc networks and [47, 72] for sensor networks.

Given that multiple disjoint paths are possible for certain network topologies (including topologies which do not have bottlenecks and thus do not prevent the presence of the required number of disjoint paths/wires) this allows for secure message transmission protocols to be implemented in various application settings upon such networks.

7.4 Using Concepts of PSMT on the World Wide Web

In this section we describe alternative ways with which *concepts* of PSMT protocols can be used on the World Wide Web for secure message transmission purposes. The following do not assume the classical model of PSMT protocols which require node disjoint network paths but instead consider alternative communication channels which are different to each other.

For this model, one considers a different set of adversary capabilities. Instead of the adversary being capable of compromising network nodes, the assumption that the adversary can compromise different online communication methods is made. For the purposes of this section *only*, we assume a passive adversary who is only able to view data transmitted using an online communication method and whose main objective is to break the privacy of a secret. Similarly to previous sections of the thesis, in this section we also consider a t -bounded adversary.

Nowadays, there exist many different ways with which people can communicate online. Examples of these include (secure) email, online chat systems, short message services, VoIP and many other methods. But can Internet users know that the communication methods they are using are truly secure? Can users know for sure that their communication is not under surveillance by third parties such as malware, criminals, foreign agents, even their own government [117, 135, 149]?

An example of an online communication method being compromised by a third party (a party who is not the sender or receiver of the communication) is the alleged snooping capabil-

ity of Microsoft upon Skype conversations - as reported in [134]. Indeed, by Microsoft's own admission, it could pass on messages to law enforcement agencies when "appropriate" - confirming its surveillance capabilities upon Skype conversations. Similarly, as reported in various online articles [67, 133], it is common practise for Internet based companies such as ISPs or even cellphone service providers to provide the communication details of their customers when they are asked to do so - for example as part of ongoing criminal investigations. Indeed in January 2013 it was reported in [141] that due to the Foreign Intelligence Surveillance Act (FISA), US federal agencies will be able to spy on the data of foreign nationals (stored in cloud servers such as those of Google, Amazon and Apple) without warning.

So how can people - and more specifically regular Internet users, communicate in a secure and private manner online? Considering the given skype example and various other security issues with online communication systems - such as privacy concerns on Facebook [40], the mass hacking of Hotmail accounts [114] amongst other similar examples, it seems very difficult for regular Internet users to find a secure and trusted online communication service. Encryption is a possible tool with which this could be achieved. But is encryption available to regular users in an easy and secure to use manner? (Free) Software which users may download to use for encrypting their messages could be infected with malware. Furthermore, can we really know that government agencies (or other powers as described in [200]) with their immense computational power and intellect are not able to break the encryption algorithms used by such software which users consider secure? Other attacks such as side-channel attacks, attacks against the key generation systems amongst others, could also be carried out by such agencies upon the encryption process of such messages.

With the vast array (and vast number of each type) of communication methods which are available to users on today's World Wide Web (email accounts, Facebook profile, twitter, skype accounts, blogs, online chat systems, online video calling systems) there are bound to be many ways of communicating in a secret manner.

Further to the many ways with which users can communicate online, one should also take into account the concept of mobility of communication. Nowadays, using a laptop, smartphone, or any other similar device capable of online mobile communication, users can communicate beyond a fixed location. Furthermore, people can connect online in many different locations - such as in coffee shops, airports, restaurants even bars, using free and publicly available wi-fi.

Considering all of the above, it seems possible to communicate online and make it very difficult for someone who wants to completely track your communication - given the appropriate tools and methodology.

As an example, Google documents allows one to create a document which can be accessed and viewed by a number of people at the same time. Such documents can be interactively edited online without Google saving any changes that take place. At the same time, Google documents allows other people to see and view what is being written - which of course can be erased within a fraction of a second (people may record their screen and view it later for better comprehension). Similar to Google docs, other online systems have similar capabilities [3, 60, 203]. Similar to the above is the reported applications Facebook is working on which can delete

messages, photos or videos once a user (the receiver) views them [17]⁵.

Based on the given examples, it is easy to see that there are many “secure” online communication channels. It is easy to see, that when considering a t bounded passive adversary one can find $t + 1$ different online communication channels which can be used without much effort (assuming the value of t to be small where $t \leq 5$ for example).

It therefore seems possible for someone to use secret sharing when considering this alternative adversary model and communication over the World Wide Web. Secret sharing schemes which could be used include Shamir secret sharing (implementations of which can be found online [147, 148]), the friendly to humans secret sharing schemes described in previous chapters of the thesis, or a very basic form of the one-time pad which uses 29 characters (26 letters of alphabet, space, full stop, question mark where 1 stands for a, 2 for b, this carries on for the other letters of the alphabet and 27 for space, 28 for full stop and 29 for question mark).

Furthermore, it is possible for someone with programming experience to code such software. Indeed, the source code for such software can be found online, downloaded and run locally on someone’s computer whilst it is offline. The different shares of the communication could then be transmitted over the World Wide Web using different devices (laptop, cell phone, smartphone, tablet), using different communication methods (sms, phone call, VoIP, online chat systems, google documents) and in different locations (home, pub, cafe, shopping mall).

The receiver of the intended messages could then check the various communication channels (which would have been agreed with the sender beforehand) and use appropriate code for the secure reconstruction of secrets.

Everything that has been described so far in this section are *concepts* which exist between theory and practice. One example of a practical implementation similar to the above examples was the implementation carried out by Jack Cheney in 2010 for his MSc thesis under the supervision of Professor Yvo Desmedt⁶. In his work, considering a passive adversary, a Thunderbird extension was implemented such that sending an e-mail to Alice@shared implies sending an e-mail to Alice@hotmail.com and to Alice@gmail.com. Sending an e-mail to Alice@shared thus creates a 2-out-of-2 secret sharing scheme of the message where one share is sent via hotmail and the other via gmail. The general $(t + 1)$ -out-of- n case was also programmed. The Thunderbird extension worked for sending and receiving emails.

7.5 Importance of Implementation

From the examples given in both Section 4.2.2 and Section 7.2, secure message transmission protocols are important - as seen in various applications. Whether they are used to defend against ongoing denial of service attacks, to ensure business continuity or used in emergency scenarios, the importance for them to be available cannot be disputed.

As outlined in previous sections, such protocols are not currently available in the real world. With the increasing number of cyber attacks - mainly Denial of Service attacks, and the devastation they can cause - as seen in the attacks against Estonia and Georgia, it seems that the Internet is not prepared for the future against such attacks. With the constant threat of terrorism

⁵It should be noted than the context of secure message transmission schemes, the photos themselves may contain encoded messages or shares of messages.

⁶This was described to the author by Professor Desmedt in email correspondence.

and crime syndicates in the world, the future seems very ominous if terrorist cells and criminals which currently resort to violating physical security alone add cyber attacks to their arsenal.

Indeed the National Security Strategy published by the British government in October 2010 [83] realizes how “cyberspace is woven in to the fabric of our society” and that it is “integral to our economy and our security and access to the Internet”. Because of this, the Security Strategy identifies cyber attacks as one of the most important types of attacks that Britain should invest and defend against - to protect people’s cyber security and the British economy also.

It is thus important to be prepared for the future and allow secure message transmission protocols to be implemented and become available on the Internet or any other important communication networks that exists or which will be built. This requires the construction and use of network disjoint paths - something that used to be available on the Internet but currently is not so for regular Internet users. Furthermore, ISPs who own and operate their networks and have the capability to do this do not make such protocols available - mainly due to cost purposes⁷.

⁷To be able to make this fully available on the Internet, Internet protocol changes may be required and the addition of new nodes and interconnections in the Internet will most probably be required. For private networks which an ISP has setup (or will setup) for a specific client this may be easier.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

This thesis has focused on secure message transmission protocols. As described throughout the thesis, such protocols are very important from a theoretical viewpoint.

They are important in theory as they are fundamental cryptographic protocols used to implement underlying assumptions of various cryptographic protocols. Additionally, such protocols can be employed in other security oriented research where the secure transmission of messages is required. Because of this, it is of outmost importance to study such protocols and to improve on the current knowledge and existing algorithms.

On a more practical aspect, such protocols also find application for the transmission of secrets with information-theoretic security. To borrow a term from Aumann, Ding and Rabin [8] such protocols achieve *everlasting privacy*.

The thesis has studied almost perfectly secure one-phase transmission protocols, presenting two new protocols which improve on previous work.

The following describes how the new polynomial time protocol presented compares to previous work.

Protocol	Computational Complexity	Communication Complexity
Protocol proposed in [167]	$O(n^3)$	$O(n^2)$
<p><u>Note:</u> Protocol proposed in [167] transmits more field elements (in number) than Protocol 1 (neglecting terms that are subquadratic the two protocols transmit $10t^2$ vs $6t^2$ field elements respectively) and requires more computation.</p>		

The following describes how the new exponential time protocol presented compares to previous work.

Protocol	Computational Complexity	Communication Complexity
Protocol proposed in [107]	$O\left(\binom{2t+1}{t+1}n^2 \log(q)\right)$	$O(n)$
Protocol 2 (Section 3.3)	$O\left(\binom{2t+1}{t+1}n^2\right)$	$O(n)$
<p><u>Note:</u> Protocol proposed in [107] requires the use of a planar difference set. Protocol 2 can use any finite field.</p>		

Two-phase perfectly secure message transmission protocols were also studied and a new efficient protocol was presented. The way this protocol compares to previous work is summarized by the following.

Protocol	Number of Messages Transmitted	Communication Complexity
Protocol proposed in [162]	1	$O(n^3)$
Protocol proposed in [105]	n^2	$O(n^3)$
Protocol 3 (Section 4.1)	1	$O(n^2)$

This two-phase protocol was later adapted to work in the context of a new network and recipient model - which for the first time is considered in PSMT protocols. The importance of such a protocol was discussed with regards to various application scenarios.

Protocols in a new area of secure message transmission where a human receiver is considered were also presented. These were based on two different constructions and the way with which people interact with the proposed protocols was evaluated through experiments carried out with human participants. The main aim of the experiments was to identify how easily and accurately the participants would use the proposed protocols. The results of the experiments showed that the experiment participants used the constructions of the proposed protocols with a high degree of accuracy. Furthermore, when compared to other PSMT protocols it was shown that such protocols require less (and simpler) computation to complete. The two-phase protocol presented also transmitted a lower number of field elements compared to other two-phase PSMT protocols.

New protocols were also presented in a single-receiver multiple-sender scenario where additionally to secure message transmission, anonymity should also be achieved by the proposed protocols. In the proposed protocols, both the security of message transmission and the anonymity of a sender was obtained when considering the information theoretic model of security.

The contributions of the thesis are primarily theoretical in nature - but the question of whether secure message transmission protocols could be implemented and used upon networks which exist in practise was reviewed. Through examples, the importance of why such a practical implementation is necessary was identified. Considering the Internet, differences between past and current configurations of the Internet which prevent implementation of such protocols nowadays were examined and analysed. The way secure message transmission protocols - or concepts of these, could be used upon other networks was also described in theory.

8.2 Future Work

The work presented in this thesis has added to the knowledge of secure message transmission protocols. Despite this, open questions still exist in this field. Due to the importance of message transmission protocols, it is imperative that the study of such protocols continues so that solutions to any open problems can be found.

Much of the remaining open problems have been discussed in detail at the end of the corresponding thesis chapters. Here we provide an outline of what remains to be solved.

The open question for one-phase almost perfectly secure message transmission protocols is to design a *polynomial* time protocol which will have an optimal communication complexity of $O(n)$ - where $n = 2t + 1$ and denotes the number of wires connecting the sender and a receiver when considering a t -threshold bounded and computationally unlimited active adversary. The protocol presented in Section 3.3 achieves the optimal communication complexity, but at exponential computational cost. It is important to solve this open problem as almost perfectly secure message transmission protocols are almost equivalent to perfectly secure message transmission protocols. This is because the probability with which the protocol may fail can be decreased to a negligible probability by either repeating the protocol¹ or by varying parameters of the protocol (such as the size q of the finite field).

If this open problem cannot be solved - as it is a hard problem, a polynomial time protocol with a communication complexity lower than $O(n^2)$ should be sought - thus improving the polynomial time protocol presented in Section 3.2.

A summary of the described one-phase almost perfectly secure open problems and how they compare to protocols presented in the thesis is presented in the following - with $n = 2t + 1$.

Protocols Presented in Thesis	Future Work
Protocol 1 (Section 3.2) Computational Complexity: $O(n^3)$ Communication Complexity: $O(n^2)$	<u>Desired Future Protocol</u> Polynomial Computational Complexity $O(n) < \text{Communication Complexity} < O(n^2)$ An example of such communication complexity is $O(n \log n)$
Protocol 2 (Section 3.3) Exponential Computational Complexity Communication Complexity: $O(n)$	<u>Desired Future Protocol</u> Polynomial Computational Complexity Communication Complexity: $O(n)$ Obviously this second case is the most ideal scenario for a future protocol.

Further to the above, one further future work problem that could be studied is to consider one-phase almost perfectly secure message transmission protocols in the asynchronous model of communication. Connectivity requirements in the asynchronous model have been found for both one and two phase perfectly secure message transmission protocols but this has yet to be considered for almost perfectly secure message transmission protocols. One can explore if the protocols proposed in this thesis will work on asynchronous networks and if they do not, connectivity requirements should be found and respective protocols should be constructed.

When considering two-phase PSMT protocols one key open problem is to design a communication protocol that is able to improve the work of [105] and achieve the optimal transmis-

¹For all one-phase almost perfectly secure message transmission protocols, there is a small probability $p \ll 1$ that the protocol will fail - meaning that the receiver will not accept a message from the message space \mathcal{M} but will instead accept \perp . With probability $1 - p$ the receiver will accept the correct message - which is the same message (in value) as sent out by the sender of the communication. By repeating one-phase almost perfectly secure protocols m number of times, the probability with which all executions will fail decreases to p^m . Based on p , selecting an appropriate value of m allows one to select an acceptable probability of protocol failure - preferably a negligible probability.

sion rate of $O(n)$ with lower than $O(n^3)$ communication complexity (CC) for one or (preferably) both phases of the protocol.

Improving the protocol presented in Section 4.1 for the PSMT of a *single message* is also an open problem to solve. One way to improve this protocol is to decrease the communication complexity of either one (or preferably both) of the two phases.

The following compares the current two-phase protocols mentioned and how future work could improve these.

Protocol	Future Work
Protocol presented in [105] Polynomial Computational Complexity Communication Complexity: $O(n^3)$ Transmission Rate: $O(n)$	Desired Future Protocol Polynomial Computational Time Desired Communication Complexity: $O(n^2)$ Transmission Rate: $O(n)$ <u>Alternative CC for one phase:</u> $O(n^2) \leq \text{Communication Complexity} < O(n^3)$
Protocol 3 (Section 4.1) Polynomial Computational Complexity Communication Complexity: $O(n^2)$ Number of Messages Transmitted: 1	Desired Future Protocol Polynomial Computational Time Desired Communication Complexity: $O(n)$ Number of Messages Transmitted: 1 <u>Alternative CC for one phase:</u> $O(n) \leq \text{Communication Complexity} < O(n^2)$

On a more theoretical and combinatorial viewpoint, the solutions to the work of Chapter 5 (for the protocols considering an active adversary) and Chapter 6 were based on either verifier set or general verifier set systems. As outlined in Appendix E, constructions of such required set systems are known, but require a complexity of t^2 points to be present in set X of the set system. It will be interesting to study such set systems and seek constructions which will require a lower complexity on the size of set X - whilst retaining a usable (non-exponential) number of blocks. This will in turn lead to more efficient protocols of the aforementioned chapters. This however is a hard problem to solve. It should be noted that this open question is outside the field of message transmission protocols but instead is a combinatorics problem - which has found applications in cryptography.

Considering message transmission protocols with a human participant, the following questions are of interest to study and answer. Can these be improved - both in terms of the communication complexity of the protocols and the simplicity of the computational operations that need to be carried out by the human participants. Currently, the proposed protocols are based on set systems. Future work could look into the design of alternative more efficient protocols (in terms of communication complexity) which are not based on set systems - for both passive and active adversaries. Any future protocols should also undergo experimental evaluation with human participants - to assess the ease and accuracy with which the protocols are used by humans. Despite the experiments carried out for the proposed protocols including a high number of participants, perhaps a greater range of age groups (which should include people over sixty

years old) should also be included. This is important as the human message transmission protocols were proposed for their use in an electronic voting scheme - and thus should include human participants across the whole breadth of voting ages.

Furthermore, before using such protocols in practise, ways of overcoming the limitation of the high number of required disjoint paths should be sought. More efficient set systems (which do not require t^2 points in set X) or protocols not based on set systems would be a good step towards resolving this.

Additionally, one could look into the question of whether message transmission protocols with a human participant - as introduced in Chapter 5, are equivalent to earlier presented PSMT protocols and if they could replace them. If they are indeed equivalent, one should consider whether with a human participant, efficient one-phase almost perfectly secure message transmission protocols can be designed.

When considering anonymous secure communication protocols, the importance of improving set systems appears once more. Alternatively, one could also consider ways to design information-theoretic anonymous secure communication protocols which are not based on set systems. Additionally, applications of information-theoretic anonymous secure communication protocols on real world networks such as the Internet (as opposed to using protocols based on conventional cryptography and unproven assumptions) should be explored. The proposed protocols could also be used in other cryptographic protocols which require anonymous and secure MIX networks.

The importance of the work presented in this thesis should also be addressed and taken into serious consideration from a practical viewpoint. Chapter 7 explored the practical difficulties which prevent secure message transmission protocols from being implemented on the current configuration and development of the Internet. As discussed, the use of secure message transmission protocols on the Internet could provide one of the best ways for an organization to defend against Denial of Service attacks. On the scale of the Internet, each “wire” (as defined in message transmission protocols) could in some cases be modelled as a path from a sender to a receiver which passes through a number of different autonomous systems (networks under a single administrative control) - with each autonomous system being present in only one of the wires which connect the sender and the receiver. In this case, network nodes in the theoretical abstract network model are similar to autonomous systems of the Internet.

The importance of defending companies/organizations/states against such attacks has only been highlighted by the constant threat of Denial of Service attacks. This could possibly be achieved by using secure message transmission protocols (as studied in this thesis) on the Internet.

It is thus important for industry to work with researchers involved in secure message transmission protocols so the practical implementation of such protocols is made available on industrial networks should they be required.

Bibliography

- [1] La Jolla Covering Repository. <http://www.ccrwest.org/cover.html>.
- [2] R. Julian R. Abel, Ahmed M. Assaf, Frank E. Bennett, Iliya Bluskov, and Malcolm Greig. Pair Covering Designs with Block Size 5. *Discrete Mathematics*, 307(14):1776–1791, 2007.
- [3] Acrobat. Buzzword. <https://www.acrobat.com/main/en/home.html>.
- [4] Saurabh Agarwal, Ronald Cramer, and Robbert de Haan. Asymptotically Optimal Two-Round Perfectly Secure Message Transmission. In *CRYPTO*, volume 4117 of *Lecture Notes in Computer Science*, pages 394–408. Springer, 2006. August 20-24, 2006, Santa Barbara, California, USA.
- [5] Miklos Ajtai. Generating Hard Instances of Lattice Problems (Extended Abstract). In *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing*, pages 99–108. ACM, 1996.
- [6] Sahel Alouneh, Anjali Agarwal, and Abdeslam En-Nouaary. A Novel Path Protection Scheme for MPLS Networks Using Multi-Path Routing. *Computer Networks*, 53(9):1530–1545, 2009.
- [7] Arne Ansper, Sven Heiberg, Helger Lipmaa, Tom André Øverland, and Filip van Laenen. Security and Trust for the Norwegian E-Voting Pilot Project E-valg 2011. In *NordSec*, volume 5838 of *LNCS*, pages 207–222. Springer, 2009. Oslo, Norway, 14-16 October 2009.
- [8] Yonatan Aumann, Yan Zong Ding, and Michael O. Rabin. Everlasting Security in the Bounded Storage Model. *IEEE Transactions on Information Theory*, 48(6):1668–1680, 2002.
- [9] BBC News. A bit of BitTorrent bother. <http://news.bbc.co.uk/1/hi/programmes/newsnight/4758636.stm>.
- [10] BBC News. Anonymous hackers say Wikileaks war to continue. <http://www.bbc.co.uk/news/technology-11935539>.
- [11] BBC News. Anti-piracy CD problems vex Sony. <http://news.bbc.co.uk/2/hi/technology/4511042.stm>.

- [12] BBC News. Georgia-Russia conflict. http://news.bbc.co.uk/1/hi/in_depth/europe/2008/georgia_russia_conflict/default.stm.
- [13] BBC News. Haiti devastated by massive earthquake. <http://news.bbc.co.uk/1/hi/8455629.stm>.
- [14] BBC News. Iranian oil terminal 'offline' after 'malware attack'. <http://www.bbc.com/news/technology-17811565>.
- [15] BBC News. Japan's damaged Fukushima nuclear plant one year on. <http://www.bbc.co.uk/news/world-asia-17192215>.
- [16] BBC News. The cyber raiders hitting Estonia. <http://news.bbc.co.uk/1/hi/world/europe/6665195.stm>.
- [17] Venture Beat. Facebook to go all Mission Impossible with self-destructing photo messaging app. <http://venturebeat.com/2012/12/17/facebook-to-go-all-mission-impossible-with-self-destructing-photo-messaging-app/>.
- [18] Amos Beimel and Matthew K. Franklin. Reliable Communication over Partially Authenticated Networks. *Theoretical Computer Science*, 220(1):185–210, 1999.
- [19] Amos Beimel and Lior Malka. Efficient Reliable Communication over Partially Authenticated Networks. In *PODC 2003*, pages 233–242. ACM, 2003. July 13-16, 2003, Boston, Massachusetts, USA.
- [20] Josh Cohen Benaloh and Jerry Leichter. Generalized Secret Sharing and Monotone Functions. In *CRYPTO*, volume 403 of *Lecture Notes in Computer Science*, pages 27–35. Springer, 1988. August 21-25, Santa Barbara, California, USA.
- [21] Elwyn Berlekamp. Factoring Polynomials over Large Finite Fields. In *SYMSAC '71: Proceedings of the second ACM symposium on Symbolic and algebraic manipulation*, page 223. ACM, 1971. Los Angeles, California, United States.
- [22] Piotr Berman and Juan A. Garay. Fast Consensus in Networks of Bounded Degree. *Distributed Computing*, 7(2):67–73, 1993.
- [23] Daniel J. Bernstein, Johannes Buchmann, and Erik Dahmen. *Post Quantum Cryptography*. Springer Publishing Company, Incorporated, 1st edition, 2008.
- [24] Alex Biryukov and Johann Großschädl. Cryptanalysis of the Full AES Using GPU-Like Special-Purpose Hardware. *Fundam. Inform.*, 114(3-4):221–237, 2012.
- [25] Alex Biryukov and Dmitry Khovratovich. Related-Key Cryptanalysis of the Full AES-192 and AES-256. In *ASIACRYPT*, volume 5912 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2009. December 6-10, Tokyo, Japan.

- [26] Alex Biryukov, Sourav Mukhopadhyay, and Palash Sarkar. Improved Time-Memory Trade-Offs with Multiple Data. In *Selected Areas in Cryptography*, volume 3897 of *Lecture Notes in Computer Science*, pages 110–127. Springer, 2005.
- [27] Ake Bjorck and Victor Pereyra. Solution of Vandermonde Systems of Equations. volume 24, pages 893–903, 1970.
- [28] George Robert Blakley. Safeguarding Cryptographic Keys. In *Proceedings of the National Computer Conference*, volume 48, pages 313–317. AFIPS Press, 1979.
- [29] Ernest F. Brickell. Some Ideal Secret Sharing Schemes. In *EUROCRYPT*, volume 434 of *Lecture Notes in Computer Science*, pages 468–475. Springer, 1989. April 10-13, Houthalen, Belgium.
- [30] CA Global Security Advisor. XCP Sony Rootkit - Spyware Detail. <http://gsa.ca.com/pest/pest.aspx?ID=453096362>.
- [31] Matthew Caesar and Jennifer Rexford. BGP Routing Policies in ISP Networks. *IEEE Network*, 19(6):5–11, 2005.
- [32] Ran Canetti and Ronny Ramzi Dakdouk. Obfuscating Point Functions with Multibit Output. In *EUROCRYPT*, volume 4965 of *Lecture Notes in Computer Science*, pages 489–508. Springer, 2008. Istanbul, Turkey, April 13-17, 2008.
- [33] David Chaum. SureVote: Technical Overview. Proceedings of the Workshop on Trustworthy Elections (WOTE '01). August 26-29 2001. Tomales Bay, CA, USA.
- [34] David Chaum. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Commun. ACM*, 24(2):84–88, February 1981.
- [35] Ashish Choudhary, Arpita Patra, B. V. Ashwinkumar, Kannan Srinathan, and Chandrasekharan Pandu Rangan. On Minimal Connectivity Requirement for Secure Message Transmission in Asynchronous Networks. In *ICDCN*, volume 5408 of *Lecture Notes in Computer Science*, pages 148–162. Springer, 2009.
- [36] Ashish Choudhury, Arpita Patra, B. V. Ashwinkumar, Kannan Srinathan, and Chandrasekharan Pandu Rangan. Secure Message Transmission in Asynchronous Networks. *Journal Parallel Distributed Computing*, 71(8):1067–1074, 2011.
- [37] CIO. Could a Cyber Blockade Happen to the U.S.? http://www.cio.com/article/490796/Could_a_Cyber_Blockade_Happen_to_the_U.S._
- [38] Cisco. Cisco Security Advisory: Crafted IP Option Vulnerability. http://www.cisco.com/en/US/products/products_security_advisory09186a00807cb157.shtml.
- [39] Cisco. Protecting the Router from DoS Attacks. Cisco 10000 Series Router Software Configuration Guide.

- [40] CNET. How to avoid making one of the 10 worst Facebook mistakes. http://howto.cnet.com/8301-11310_39-57556686-285/how-to-avoid-making-one-of-the-10-worst-facebook-mistakes/.
- [41] Charles J. Colbourn and Jeffrey H. Dinitz. *The CRC Handbook of Combinatorial Designs*. CRC Press, Inc., New York, 1996.
- [42] Douglas E. Comer. *Internetworking with TCP/IP, Volume 1: Principles, Protocols, and Architectures, Fourth Edition*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2000.
- [43] COMPUTERWORLD. Russia's cyber blockade of Georgia worked. Could it happen here? http://www.computerworld.com/s/article/9131270/Russia_s_cyber_blockade_of_Georgia_worked._Could_it_happen_here_.
- [44] Costas Courcoubetis and Richard Weber. *Pricing Communication Networks: Economics, Technology and Modelling (Wiley Interscience Series in Systems and Optimization)*. John Wiley & Sons, 2003.
- [45] Albert Cutler. Broadcasting Systems with Fibre Optic Transmission Lines, 1979. Patent US 4135202, <http://www.freepatentsonline.com/4135202.html>.
- [46] Cynthia Dwork and David Peleg and Nicholas Pippenger and Eli Upfal. Fault Tolerance in Networks of Bounded Degree. In *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, STOC '86, pages 370–379. ACM, 1986.
- [47] Budhaditya Deb, Sudeept Bhatnagar, and Badri Nath. ReInForM: Reliable Information Forwarding Using Multiple Paths in Sensor Networks. In *LCN*, pages 406–415. IEEE Computer Society, 2003. 20-24 October, Bonn/Königswinter, Germany.
- [48] Yvo Desmedt. Analyzing Survivable Computation in Critical Infrastructures. In *Managing Cyber Threats*, volume 5 of *Massive Computing*, pages 267–280. Springer.
- [49] Yvo Desmedt and Stelios Erotokritou. Towards Usable and Secure Internet Voting. <http://www.cyi.ac.cy/images/ResearchProjects/SteliosE/towUsSecIntVoting.pdf>.
- [50] Yvo Desmedt, Stelios Erotokritou, and Reihaneh Safavi-Naini. Simple and Communication Complexity Efficient Almost Secure and Perfectly Secure Message Transmission Schemes. In *AFRICACRYPT*, volume 6055 of *Lecture Notes in Computer Science*, pages 166–183, 2010. Stellenbosch, South Africa, May 3-6, 2010.
- [51] Yvo Desmedt and Kaoru Kurosawa. How to Break a Practical MIX and Design a New One. In *Advances in Cryptology — Eurocrypt 2000, Proceedings Lecture Notes in Computer Science 1807*, pages 557–572. Springer-Verlag, 2000. May 14-18, Bruges, Belgium.

- [52] Yvo Desmedt and Yongge Wang. Perfectly Secure Message Transmission Revisited. In *EUROCRYPT*, volume 2332 of *Lecture Notes in Computer Science*, pages 502–517. Springer, 2002. April 28 - May 2, Amsterdam, The Netherlands.
- [53] Yvo Desmedt, Yongge Wang, and Mike Burmester. A Complete Characterization of Tolerable Adversary Structures for Secure Point-to-Point Transmissions Without Feedback. In *ISAAC*, volume 3827 of *Lecture Notes in Computer Science*, pages 277–287. Springer, 2005. December 19-21, Sanya, Hainan, China.
- [54] Yvo Desmedt, Yongge Wang, and Mike Burmester. A Complete Characterization of Tolerable Adversary Structures for Secure Point-to-Point Transmissions without Feedback. In *Algorithms and Computation, 16th Annual International Conference, ISAAC 2005, Lecture Notes in Computer Science 3827*, pages 277–287, 2005. December 19 - 21, 2005, Sanya, Hainan, China.
- [55] Danny Dolev. The Byzantine Generals Strike Again. *J. Algorithms*, 3(1):14–30, 1982.
- [56] Danny Dolev, Cynthia Dwork, Orli Waarts, and Moti Yung. Perfectly Secure Message Transmission. *Journal of the ACM*, 40(1):17–47, January 1993.
- [57] Electronic Frontier Foundation. Cracking DES. https://w2.eff.org/Privacy/Crypto/Crypto_misc/DESCracker/.
- [58] Anwar Elwalid and Debasis Mitra. Traffic Shaping at a Network Node: Theory, Optimum Design, Admission Control. In *INFOCOM '97*, page 444. IEEE Computer Society, 1997. April 7-12, 1997, Kobe, Japan.
- [59] Stelios Erotokritou. Planar Primes between 1 and 45 Billion. <http://www.cyi.ac.cy/planarPrimes2.pdf>.
- [60] Etherpad. Etherpad. <http://etherpad.org/>.
- [61] Igor Faynberg, Lawrence Gabuzda, and Hui Lan Lu. *Converged Networks and Services: Internetworking IP and the PSTN*. John Wiley & Sons, Inc., New York, NY, USA, 2000.
- [62] Matthias Fitzi, Matthew K. Franklin, Juan A. Garay, and Harsha Vardhan Simhadri. Towards Optimal and Efficient Perfectly Secure Message Transmission. In *TCC*, volume 4392 of *Lecture Notes in Computer Science*, pages 311–322. Springer, 2007. February 21-24, 2007, Amsterdam, The Netherlands.
- [63] Matthias Fitzi, Nicolas Gisin, Ueli M. Maurer, and Oliver von Rotz. Unconditional Byzantine Agreement and Multi-party Computation Secure against Dishonest Minorities from Scratch. In *EUROCRYPT*, volume 2332 of *Lecture Notes in Computer Science*, pages 482–501. Springer, 2002.
- [64] Matthias Fitzi, Daniel Gottesman, Martin Hirt, Thomas Holenstein, and Adam Smith. Detectable Byzantine Agreement Secure Against Faulty Majorities. In *PODC*, pages 118–126, 2002.

- [65] Matthias Fitzi and Ueli M. Maurer. Efficient Byzantine Agreement Secure Against General Adversaries. In *DISC*, volume 1499 of *Lecture Notes in Computer Science*, pages 134–148. Springer, 1998.
- [66] Focalprice. SANMO SMM4. <http://www.focalprice.com/MHO40B/>.
- [67] Electronic Frontier Foundation. Australian Government Moves to Expand Surveillance Powers. <https://www.eff.org/deeplinks/2012/07/australian-government-moves-expand-surveillance-powers>.
- [68] Matthew Franklin and Rebecca N. Wright. Secure Communication in Minimal Connectivity Models. *Journal of Cryptology: The Journal of the International Association for Cryptologic Research*, 13(1):9–30, 2000. July 21-24, 2002, Monterey, California, USA.
- [69] Matthew K. Franklin and Moti Yung. Secure Hypergraphs: Privacy from Partial Broadcast (Extended Abstract). In *STOC*, pages 36–44. ACM, 1995. 29 May-1 June 1995, Las Vegas, Nevada, USA.
- [70] Jun Furukawa. Efficient and Verifiable Shuffling and Shuffle-Decryption. *IEICE Transactions*, 88-A(1):172–188, 2005.
- [71] Ian Gallagher, Matthew Robinson, Sharam Semnani, Adrian Smith, and Jim Mackenzie. Multi-Protocol Label Switching as the Basis for a Converged Core Network. *BT Technology Journal*, 22(2):95–103, 2004.
- [72] Deepak Ganesan, Ramesh Govindan, Scott Shenker, and Deborah Estrin. Highly-Resilient, Energy-Efficient Multipath Routing in Wireless Sensor Networks. In *MobiHoc*, pages 251–254. ACM, 2001. October 4-5, Long Beach, CA, USA.
- [73] Juan A. Garay, Clint Givens, and Rafail Ostrovsky. Secure Message Transmission with Small Public Discussion. In *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 177–196. Springer, 2010. May 30 - June 3, 2010, French Riviera.
- [74] Juan A. Garay and Rafail Ostrovsky. Almost-Everywhere Secure Computation. In *EUROCRYPT*, volume 4965 of *Lecture Notes in Computer Science*, pages 307–323. Springer, 2008. April 13-17, 2008, Istanbul, Turkey.
- [75] Naveen Garg, Vijay Vazirani, and Mihalis Yannakakis. Approximate Max-Flow Min-(Multi)Cut Theorems and Their Applications. In *STOC '93*, pages 698–707. ACM, 1993.
- [76] Constant Gbaguidi, Jean Pierre Hubaux, Giovanni Pacifici, and Asser N. Tantawi. An Architecture for the Integration of Internet and Telecommunication Services. In *Workshop on Open Architectures and Network Programming (OPENARCH)*, 1999.
- [77] Craig Gentry, Zulfikar Ramzan, and Stuart G. Stubblebine. Secure Distributed Human Computation. In *Security Protocols Workshop*, volume 5087 of *Lecture Notes in Computer Science*, pages 177–180. Springer, 2006. March 27-29, Cambridge, UK.

- [78] Luc De Ghein. *MPLS Fundamentals: A Comprehensive Introduction to MPLS Theory and Practice*. Cisco Press, 2006.
- [79] Andrew V. Goldberg and Robert E. Tarjan. A new Approach to the maximum-flow Problem. *Journal of the ACM*, 35(4):921–940, 1988.
- [80] Sharon Goldberg, Shai Halevi, Aaron D. Jaggard, Vijay Ramachandran, and Rebecca N. Wright. Rationality and Traffic Attraction: Incentives for Honest Path Announcements in BGP. In *SIGCOMM*, pages 267–278, 2008. August 17-22, 2008, Seattle, WA, USA.
- [81] Daniel M. Gordon, Greg Kuperberg, and Oren Patashnik. New Constructions for Covering Designs. *J. Combin. Designs*, 3:269–284, 1995.
- [82] Daniel M. Gordon, Greg Kuperberg, Oren Patashnik, and Joel H. Spencer. Asymptotically Optimal Covering Designs. *Journal of Combinatorial Theory, Series A*.
- [83] Directgov HM Government. Securing Britain in an Age of Uncertainty: The Strategic Defence and Security Review. 19 Oct 2010.
- [84] Kerry Grosser. Human Networks in Organizational Information Processing. *Annual Review of Information Science and Technology*, 26:349–402, 1991.
- [85] Jens Groth. A Verifiable Secret Shuffle of Homomorphic Encryptions. *J. Cryptology*, 23(4):546–579, 2010.
- [86] British Sky Broadcasting Group. Hacking Backlash As Top ‘Piracy’ Site Shut Down. <http://news.sky.com/home/world-news/article/16152992>.
- [87] Venkatesan Guruswami and Atri Rudra. Explicit Codes Achieving List Decoding Capacity: Error-Correction With Optimal Redundancy. *IEEE Transactions on Information Theory*, 54(1):135–150, 2008.
- [88] Vassos Hadzilacos. *Issues of Fault Tolerance in Concurrent Computations (databases, reliability, transactions, agreement protocols, distributed computing)*. PhD thesis, Harvard University, Cambridge, MA, USA, 1985.
- [89] Jiayue He and Jennifer Rexford. Toward Internet Wide Multipath Routing. *IEEE Network*, 22(2):16–21, 2008.
- [90] Amir Herzberg, Stanislaw Jarecki, Hugo Krawczyk, and Moti Yung. Proactive Secret Sharing Or: How to Cope With Perpetual Leakage. In *CRYPTO*, volume 963 of *Lecture Notes in Computer Science*, pages 339–352. Springer, 1995.
- [91] Martin Hirt and Ueli Maurer. Player Simulation and General Adversary Structures in Perfect Multiparty Computation. *Journal of Cryptology*, 13(1):31–60, 2000.
- [92] Nicholas J. Hopper and Manuel Blum. Secure Human Identification Protocols. In *ASIACRYPT*, volume 2248 of *Lecture Notes in Computer Science*, pages 52–66. Springer, 2001.

- [93] HTC. HTC Desire VC. <http://www.htc.com/sea/smartphones/htc-desire-vc/>.
- [94] Jean Pierre Hubaux, Constant Gbaguidi, Shawn Koppenhoefer, and Jean Yves Le Boudec. The Impact of the Internet on Telecommunication Architectures. *Computer Networks*, 31(3):257–273, 1999.
- [95] ISPreview. ISP TalkTalk UK Warns Music Fans Will Sidestep P2P Filesharing Clampdown. <http://www.ispreview.co.uk/story/2010/03/12/isp-talktalk-uk-warns-music-fans-will-sidestep-p2p-filesharing-clampdown.html>.
- [96] Mitsuru Ito, Akira Saito, and Takao Nishizeki. Secret Sharing Schemes Realizing General Access Structures. In *Proc. IEEE Global Telecommunications Conf., Globecom'87*, pages 99–102. IEEE Communications Soc. Press, 1987.
- [97] Markus Jakobsson, Ari Juels, and Ronald L. Rivest. Making Mix Nets Robust for Electronic Voting by Randomized Partial Checking. In *Proceedings of the 11th USENIX Security Symposium*, pages 339–353, Berkeley, CA, USA, 2002. USENIX Association.
- [98] David Kahn. *Seizing the Enigma: The Race to Break the German U-boat Codes, 1939-1943*. Frontline Books.
- [99] Brad Karp and H. T. Kung. GPSR: Greedy Perimeter Stateless Routing for Wireless Networks. In *MOBICOM*, pages 243–254. ACM, 2000. August 6-11, Boston, MA, USA.
- [100] Sachin Katti, Jeff Cohen, and Dina Katabi. Information Slicing: Anonymity Using Unreliable Overlays. In *NSDI*. USENIX, 2007. April 11-13, 2007, Cambridge, Massachusetts, USA.
- [101] Young-Jin Kim, Ramesh Govindan, Brad Karp, and Scott Shenker. Geographic Routing Made Practical. In *NSDI*. USENIX, 2005. May 2-4, Boston, Massachusetts, USA.
- [102] David Kotz, Calvin C. Newport, Robert S. Gray, Jason Liu, Yougu Yuan, and Chip Elliott. Experimental Evaluation of Wireless Simulation Assumptions. In *MSWiM*. ACM, 2004. October 4-6, Venice, Italy.
- [103] Dan Kuehl. Defining Information Power. In *Strategic Forum*, (115), June 1997.
- [104] Ashwin Kumar, Pranava Goundan, Kannan Srinathan, and Chandrasekharan Pandu Rangan. On Perfectly Secure Communication over Arbitrary Networks. In *PODC '02: Proceedings of the twenty-first annual symposium on Principles of distributed computing*, pages 193–202. ACM, 2002. July 21-24, Monterey, California, USA.
- [105] Kaoru Kurosawa and Kazuhiro Suzuki. Truly Efficient 2-Round Perfectly Secure Message Transmission Scheme. In *EUROCRYPT*, volume 4965 of *Lecture Notes in Computer Science*, pages 324–340. Springer, 2008. April 13-17, Istanbul, Turkey.

- [106] Kaoru Kurosawa and Kazuhiro Suzuki. Truly Efficient 2-Round Perfectly Secure Message Transmission Scheme. *IACR Cryptology ePrint Archive*, 2008:421, 2008.
- [107] Kaoru Kurosawa and Kazuhiro Suzuki. Almost Secure (1-Round, n-Channel) Message Transmission Scheme. *ICITS 2007, (Revised Selected Papers)*, pages 99–112, 2009. Madrid, Spain, May 25-29, 2007.
- [108] Leslie Lamport. Constructing Digital Signatures from a One-Way Function. Technical report, 1979.
- [109] Leslie Lamport, Robert E. Shostak, and Marshall C. Pease. The Byzantine Generals Problem. *ACM Trans. Program. Lang. Syst.*, 4(3):382–401, 1982.
- [110] Harte Lawrence and Ofrane Avi. *Telecom Systems, PSTN, PBX, Datacom, IP Telephony, IPTV, Wireless and Billing*. Althos, 2006.
- [111] Sung-Ju Lee and Mario Gerla. Split Multipath Routing with Maximally Disjoint Paths in ad-hoc Networks. *IEEE International Conference on Communications, 2001. ICC 2001*, 10:3201–3205, 2002.
- [112] Tom Leighton and Satish Rao. An Approximate max-flow min-cut Theorem for Uniform Multicommodity Flow Problems with Applications to Approximation Algorithms. *Foundations of Computer Science, Annual IEEE Symposium on*, 0:422–431, 1988.
- [113] Barry M. Leiner, Vinton G. Cerf, David D. Clark, Robert E. Kahn, Leonard Kleinrock, Daniel C. Lynch, Jon Postel, Larry G. Roberts, and Stephen Wolff. A Brief History of the Internet. *SIGCOMM Comput. Commun. Rev.*, 39(5):22–31, 2009.
- [114] LETSbyteCODE. Microsoft Mail has Undergond Massice Hacking of Accounts. <http://letsbytecode.com/security/microsoft-mail-has-undergone-massive-hacking-of-accounts/>.
- [115] Lilian Edwards, Professor of e-Governance, University of Strathclyde. Wikileaks, DDOS and UK Criminal Law: The Key Issues. <http://ipandit.practicallaw.com/1-504-3391>.
- [116] Ratul Mahajan, David Wetherall, and Thomas E. Anderson. Understanding BGP Misconfiguration. In *SIGCOMM*, pages 3–16. ACM, 2002. August 19-23, Pittsburgh, PA, USA.
- [117] Daily Mail. Blogger faces five years in Chinese jail after tweeting a joke about Communist Party delegates dying. <http://www.dailymail.co.uk/news/article-2236305/>.
- [118] Dahlia Malkhi, Ofer Margo, and Elan Pavlov. E-Voting Without ‘Cryptography’. In *Financial Cryptography*, volume 2357 of *Lecture Notes in Computer Science*, pages 1–15. Springer, 2002. March 11-14, Southampton, Bermuda.

- [119] Mahesh K. Marina and Mahesh K. Das. On-Demand Multi Path Distance Vector Routing in Ad Hoc Networks. In *ICNP*, pages 14–23. IEEE Computer Society, 2001. 11-14 November 2001, Riverside, CA, USA.
- [120] Tsutomu Matsumoto. Human-Computer Cryptography: An Attempt. *Journal of Computer Security*, 6(3):129–150, 1998.
- [121] Robert McEliece and Dilip Sarwate. On Sharing Secrets and Reed-Solomon Codes. *Commun. ACM*, 24(9):583–584, 1981.
- [122] Ralf C. Merkle. *Secrecy, Authentication, and Public Key Systems*. PhD thesis, 1979.
- [123] Robert M. Metcalfe and David R. Boggs. Ethernet: Distributed Packet Switching for Local Computer Networks. *Commun. ACM*, 19:395–404, July 1976.
- [124] Microsoft Technet. PathPing. <http://technet.microsoft.com/library/Cc958876>.
- [125] Microsoft TechNet. Source Address Spoofing. <http://technet.microsoft.com/en-gb/library/cc723706.aspx>.
- [126] W. H. Mills. Covering Designs I: Coverings by a Small Number of Subsets. In *Ars Combinatoria*, 8, pages 199–315, 1979.
- [127] Jelena Mirkovic, Sven Dietrich, David Dittrich, and Peter Reiher. *Internet Denial of Service: Attack and Defense Mechanisms (Radia Perlman Computer Networking and Security)*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2004.
- [128] Jelena Mirkovic and Peter Reiher. A Taxonomy of DDoS attack and DDoS Defense Mechanisms. *SIGCOMM Comput. Commun. Rev.*, 34(2):39–53, 2004.
- [129] Edmar Mota-Garca and Rogelio Hasimoto-Beltran. Fast Geographic Internet Mapping System for Location Based Services. In *International Magazine on Advances in Computer Science and Telecommunications*, volume 1, pages 37–43, 2010.
- [130] Roger M. Needham. Denial of Service. In *CCS '93: Proceedings of the 1st ACM conference on Computer and communications security*, pages 151–153. ACM, 1993. November 3-5, Fairfax, Virginia, USA.
- [131] New York Times. Before the Gunfire, Cyberattacks. <http://www.nytimes.com/2008/08/13/technology/13cyber.html>.
- [132] BBC News. Hackers retaliate over Megaupload website shutdown. <http://www.bbc.co.uk/news/technology-16646023>.
- [133] BBC News. How will the proposed surveillance laws work? <http://www.bbc.co.uk/news/uk-18434232>.

- [134] BBC News. Skype denies police surveillance policy change. <http://www.bbc.co.uk/news/technology-19012415>.
- [135] The Buffalo News. Cleveland Hill student charged with making bomb threat on Twitter. <http://www.buffalonews.com/apps/pbcs.dll/article?AID=/20121220/CITYANDREGION/121229924/1010>.
- [136] NIST. DATA ENCRYPTION STANDARD (DES). Federal Information Processing Standard, FIPS-46-3, October 1999.
- [137] NIST. Secure Hash Standard. Federal Information Processing Standard. Federal Information Processing Standard, FIPS-180-1, April 1995.
- [138] Nokia. Nokia C2-00. <http://www.nokia.com/mea-en/products/phone/c2-00/>.
- [139] National Institute of Standards and Technology. Advanced Encryption Standard. <http://csrc.nist.gov/archive/aes/index.html>.
- [140] National Institute of Standards and Technology. DATA ENCRYPTION STANDARD (DES). <http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf>.
- [141] Mail Online. Britons with data on Amazon, Apple and Google cloud servers in America can be snooped on in secret by U.S. government. <http://www.dailymail.co.uk/news/article-2270608/>.
- [142] Eric Osborne. *Traffic Engineering with MPLS*. Cisco Press, 2002.
- [143] Arpita Patra, Ashish Choudhary, Chandrasekharan Pandu Rangan, Kannan Srinathan, and Prasad Raghavendra. Perfectly Reliable and Secure Message Transmission Tolerating Mobile Adversary. *IJACT*, 1(3):200–224, 2009.
- [144] Arpita Patra, Ashish Choudhary, Kannan Srinathan, and Chandrasekharan Pandu Rangan. Unconditionally Reliable and Secure Message Transmission in Undirected Synchronous Networks: Possibility, Feasibility and Optimality. Cryptology ePrint Archive, Report 2008/141.
- [145] Arpita Patra, Ashish Choudhary, Kannan Srinathan, and Chandrasekharan Pandu Rangan. Perfectly Reliable and Secure Communication in Directed Networks Tolerating Mixed Adversary. In *DISC*, volume 4731 of *Lecture Notes in Computer Science*, pages 496–498. Springer, 2007. September 24–26, Limassol, Cyprus.
- [146] phoneArena.com. Samsung B7722 Review. http://www.phonearena.com/reviews/Samsung-B7722-Review_id2488.
- [147] point-at infinity.org. Shamir's Secret Sharing Scheme. <http://point-at-infinity.org/ssss/>.

- [148] point-at infinity.org. The SSSS demo page. <http://point-at-infinity.org/ssss/demo.html>.
- [149] Huff Post. India Facebook Arrests In Palghar Over Posts Raise Free Speech Concerns. http://www.huffingtonpost.com/2012/11/28/india-facebook-arrests-palghar-free-speech_n_2204879.html.
- [150] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes: The Art of Scientific Computing*. Cambridge University Press, 1992.
- [151] Bruno Quoitin, Steve Uhlig, Cristel Pelsser, Louis Swinnen, and Olivier Bonaventure. Interdomain Traffic Engineering with BGP. *IEEE Communications Magazine*, 41, 2003.
- [152] Tal Rabin and Michael Ben-Or. Verifiable Secret Sharing and Multiparty Protocols with Honest Majority. In *STOC '89: Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 73–85. ACM Press, 1989.
- [153] Irving Reed and Gustave Solomon. Polynomial Codes Over Certain Finite Fields. *Journal of the Society for Industrial and Applied Mathematics*, 8(2):300–304, 1960.
- [154] Rolf Rees, Douglas R. Stinson, Ruizhong Wei, and G. H. John van Rees. An Application of Covering Designs: Determining the Maximum Consistent Set of Shares in a Threshold Scheme. *Ars Comb. Volume 53*, pages 225–237, 1999.
- [155] Yakov Rekhter and T. Li. A Border Gateway Protocol 4 (BGP-4). IETF RFC 1771, March 1995.
- [156] Reuters. Kremlin loyalist says launched Estonia cyber-attack. <http://www.reuters.com/article/idUSTRE52B4D820090313>.
- [157] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.
- [158] Reihaneh Safavi-Naini, Mohammed Ashraful Tuhin, and Pengwei Wang. A general construction for 1-round δ -rmt and $(0, \delta)$ -smt. In *ACNS*, volume 7341 of *Lecture Notes in Computer Science*, pages 344–362. Springer, 2012. Singapore, June 26-29, 2012.
- [159] Samsung. Galaxy Ace Duos. <http://www.samsung.com/ph/consumer/mobile-devices/smartphones/android/GT-S6802CWAXTC>.
- [160] San Francisco Chronicle. Oakland police radios fail during Obama visit. <http://www.sfgate.com/bayarea/article/Oakland-police-radios-fail-during-Obama-visit-3736022.php>.
- [161] Hasan Md. Sayeed and Hosame Abu-Amara. Perfectly Secure Message Transmission in Asynchronous Networks. In *SPDP '95: Proceedings of the 7th IEEE Symposium on Parallel and Distributed Processing*, page 100, Washington, DC, USA, 1995. IEEE Computer Society.

- [162] Hasan Md. Sayeed and Hosame Abu-Amara. Efficient Perfectly Secure Message Transmission in Synchronous Networks. *Information and Computation*, 126(1):53–61, 1996.
- [163] Adi Shamir. How to Share a Secret. *Communications of the ACM*, 22(11):612–613, 1979.
- [164] Claude E. Shannon. Communication Theory of Secrecy Systems. *Bell Systems Technical Journal*, 28:656–715, 1949.
- [165] Hongsong Shi, Shaoquan Jiang, Reihaneh Safavi-Naini, and Mohammed Ashraful Tuhin. Optimal Secure Message Transmission by Public Discussion. *CoRR*, abs/0901.2192, 2009.
- [166] Peter Shor. Algorithms for Quantum Computation: Discrete Logarithms and Factoring. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science, SFCS '94*, pages 124–134. IEEE Computer Society, 1994.
- [167] Kannan Srinathan, Ashish Choudhary, Arpita Patra, and Chandrasekharan Pandu Rangan. Efficient Single Phase Unconditionally Secure Message Transmission with Optimum Communication Complexity. In *PODC 2008*, page 457. ACM, 2008. August 18-21, Toronto, Canada.
- [168] Kannan Srinathan, M. V. N. Ashwin Kumar, and Chandrasekharan Pandu Rangan. Asynchronous Secure Communication Tolerating Mixed Adversaries. In *ASIACRYPT*, volume 2501 of *Lecture Notes in Computer Science*, pages 224–242. Springer, 2002. December 1-5, 2002, Queenstown, New Zealand.
- [169] Kannan Srinathan, Arvind Narayanan, and Chandrasekharan Pandu Rangan. Optimal Perfectly Secure Message Transmission. In *CRYPTO*, volume 3152 of *Lecture Notes in Computer Science*, pages 545–561. Springer, 2004. August 15-19, California, USA.
- [170] Kannan Srinathan, Prasad Raghavendra, and Chandrasekharan Pandu Rangan. On Proactive Perfectly Secure Message Transmission. In *ACISP*, volume 4586 of *Lecture Notes in Computer Science*, pages 461–473. Springer, 2007. July 2-4, 2007, Townsville, Australia.
- [171] Anand Srinivas and Eytan Modiano. Finding Minimum Energy Disjoint Paths in Wireless Ad-Hoc Networks. *Wireless Networks*, 11(4):401–417, 2005.
- [172] The CNN Wire Staff. Anonymous strikes back after feds shut piracy hub Megaupload. <http://edition.cnn.com/2012/01/19/business/megaupload-shutdown/index.html>.
- [173] Pantelimon Stanica. Good Lower and Upper Bounds on Binomial Coefficients. *Journal of Inequalities in Pure and Applied Mathematics*, 2(3), 2001. Article 30.
- [174] J. W. Suurballe. Disjoint Paths in a Network. *Networks*, 4(2):125–145, 1974.

- [175] Nils Svendsen and Stephen Wolthusen. An Analysis of Cyclical Interdependencies in Critical Infrastructures. 5141:25–36, 2008.
- [176] IEEE Spectrum Inside Technology. The Athens Affair. <http://spectrum.ieee.org/telecom/security/the-athens-affair/0>.
- [177] Technology Review - MIT Magazine. Wikileaks Isn't Going Anywhere. <http://www.technologyreview.com/blog/editors/26085/>.
- [178] Renata Teixeira, Keith Marzullo, Stefan Savage, and Geoffrey M. Voelker. Characterizing and Measuring Path Diversity of Internet Topologies. In *SIGMETRICS '03*, pages 304–305. ACM, 2003. June 9-14, 2003, San Diego, CA, USA.
- [179] Renata Teixeira, Keith Marzullo, Stefan Savage, and Geoffrey M. Voelker. In Search of Path Diversity in ISP Networks. pages 313–318. ACM, 2003. Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement, Miami Beach, FL, USA, October 27-29, 2003.
- [180] The National Archives. Intelligence Records in The National Archives. <http://www.nationalarchives.gov.uk/catalogue/rdleaflet.asp?sLeafletID=32>.
- [181] The New York Times. Digital Fears Emerge After Data Siege in Estonia. <http://www.nytimes.com/2007/05/29/technology/29estonia.html>.
- [182] The Register. Greek mobile wiretap scandal unpicked. http://www.theregister.co.uk/2007/07/11/greek_mobile_wiretap_latest/page2.html.
- [183] The Washington Post. Georgian Web Sites Under Attack. http://voices.washingtonpost.com/securityfix/2008/08/georgian_web_sites_under_attac.html.
- [184] Andrew Tran, Nicholas Hopper, and Yongdae Kim. Hashing it out in Public: Common Failure Modes of DHT-based Anonymity Schemes. In *WPES '09: Proceedings of the 8th ACM workshop on Privacy in the electronic society*, pages 71–80. ACM, 2009.
- [185] D Turk. Configuring BGP to block Denial-of-Service attacks. IETF RFC 3882, September 2004.
- [186] Twitter. DoS Attack. <http://status.twitter.com/post/157191978/ongoing-denial-of-service-attack>.
- [187] Eli Upfal. Tolerating Linear Number of Faults in Networks of Bounded Degree. In *Proceedings of the eleventh annual ACM symposium on Principles of distributed computing, PODC '92*, pages 83–89. ACM, 1992.
- [188] ViewSonic. V350. <http://www.viewsoniceurope.com/uk/products/v350.htm>.

- [189] Luis von Ahn, Manuel Blum, Nicholas J. Hopper, and John Langford. The CAPTCHA Web Page, 2000. <http://www.captcha.net/>.
- [190] Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu. Finding Collisions in the Full SHA-1. In *CRYPTO*, volume 3621 of *Lecture Notes in Computer Science*, pages 17–36. Springer, 2005. August 14-18, Santa Barbara, California, USA.
- [191] Yongge Wang. Robust Key Establishment in Sensor Networks. *SIGMOD Record*, 33(1):14–19, 2004.
- [192] Wikipedia. 2007 cyberattacks on Estonia. http://en.wikipedia.org/wiki/2007_cyberattacks_on_Estonia.
- [193] Wikipedia. Data Encryption Standard. http://en.wikipedia.org/wiki/Data_Encryption_Standard.
- [194] Wikipedia. Fukushima Daiichi nuclear disaster. http://en.wikipedia.org/wiki/Fukushima_Daiichi_nuclear_disaster.
- [195] WikiPedia. Session hijacking. http://en.wikipedia.org/wiki/Session_hijacking.
- [196] Douglas Wikström. The Security of a Mix-Center Based on a Semantically Secure Cryptosystem. In *INDOCRYPT*, volume 2551 of *Lecture Notes in Computer Science*, pages 368–381. Springer, 2002. Hyderabad, India, December 16-18, 2002.
- [197] Wired. Comcast Sued Over BitTorrent Blocking. <http://www.wired.com/threatlevel/2007/11/comcast-sued-ov/>.
- [198] Wired. Former NSA Director: Countries Spewing Cyberattacks Should Be Held Responsible. <http://www.wired.com/threatlevel/2010/07/hayden-at-blackhat/>.
- [199] Wired. Hack Attacks. <http://www.wired.com/dangerroom/2009/06/activists-launch-hack-attacks-on-tehran-regime>.
- [200] James Bamford Wired.com. The NSA Is Building the Country’s Biggest Spy Center (Watch What You Say). http://www.wired.com/threatlevel/2012/03/ff_nsadatacenter/all/1.
- [201] Marcelo Yannuzzi, Xavier Masip-Bruin, and Olivier Bonaventure. Open Issues in Inter-domain Routing: A Survey. *IEEE Network*, 19(6):49–56, 2005.
- [202] Ming Zhang, Yaoping Ruan, Vivek Pai, and Jennifer Rexford. How DNS Misnaming Distorts Internet Topology Mapping. In *Proc. USENIX Annual Technical Conference, May/June*, 2006.
- [203] Zoho. Zoho. <http://www.zoho.com/>.

Appendix A

Other Security Definitions

In this appendix we give details of security definitions for message transmission protocols which exist in the literature. We argue why we think these definitions are incomplete and why a new security definition which includes new security parameters was proposed and formalized during research carried out in the thesis.

A.1 Current Security Definition - Probabilistic Reliability

The following definition is taken from Franklin and Wright [68] and defines (ϵ, δ) -security.

1. Let $\delta < \frac{1}{2}$. A message transmission protocol is δ -reliable if, with probability at least $1 - \delta$, B terminates with $M^B = M^A$. The probability is for any choice of M^A and the coin flips of all nodes.
2. A message transmission protocol is perfectly reliable if it is 0-reliable.
3. ϵ refers to the privacy that is achieved. A message transmission protocol is ϵ -private if for every two messages $M_0, M_1 \in \mathcal{M}$ and every r , $\sum_c |Pr[\mathcal{V}_{ADV}(M_0, r) = c] - Pr[\mathcal{V}_{ADV}(M_1, r) = c]| \leq 2\epsilon$. The probabilities are taken over the randomness of the honest parties and the sum is over all possible values of the adversary's view.
4. A message transmission protocol is (ϵ, δ) -secure if it is ϵ -private and δ -reliable.

A.2 Current Security Definition - Probabilistic Failure

Almost perfectly secure message transmission was considered in [107] with the following security definition:

Definition 9. *We say that a message transmission scheme is (t, δ) -secure if the following conditions are satisfied for any adversary **A** who can corrupt at most t out of n channels. **A** learns no information on M^A . More precisely when R denotes any randomness (coin flips) the protocol uses during the execution of the protocol:*

$$Pr(R = M^A | X_{i_1} = x_{i_1}, \dots, X_{i_t} = x_{i_t}) = Pr(R = M^A)$$

for any $M^A \in \mathcal{M}$ and any possible x_{i_1}, \dots, x_{i_t} messages observed by **A** on adversary controlled wires.

General Reliability. *The receiver outputs $M^B = M^A$ or \perp (failure). The receiver thus never*

outputs a wrong secret.

Failure. $Pr(\text{Receiver outputs } \perp) < \delta$.

It should be noted that the first work to consider one-phase protocols which terminate with a high probability was in [169] - however due to errors in the protocols presented it was somewhat overlooked until it was re-addressed in [107].

A.3 Comparison of the two Security Definitions

The above security definitions refer to two different security properties. The definition of reliability from [68] only considers executions that terminate and always output a message. When executed, protocols output the correct message with a bounded probability. However, the reliability definition of [107] identifies two types of executions. The first of these are executions that terminate but do not produce an output message (but instead output \perp). The second of these are executions that terminate and always produce the correct result. Reliability in these definitions refers to both types of executions.

As the concept of reliability was used in the two security definitions to refer to different security properties, a new security definition was introduced that reconciled the above definitions. This was done by introducing new security parameters which capture the *authenticity* of the received message and the *availability* of the message transmission protocol. Executions that terminate but do not produce a correct output can be seen as an attack on the authenticity of the message. Executions that output failure can be seen as an attack on the availability of the transmission protocol. This lead us to propose the security definition presented in Section 2.3. The importance of this was to introduce new security properties which could be used to resolve any ambiguities which could come up due to the vagueness of the previous security definitions.

Appendix B

Adversaries in Secure Message Transmission Protocols

Throughout the thesis a static t -threshold bounded computationally unlimited active adversary is considered. In this appendix we give details of other adversary models which are considered in research of secure message transmission protocols.

The adversary present in the interconnecting network which connects the sender and receiver of a message transmission protocol is assumed to know the complete protocol specification, message space \mathcal{M} and the complete structure of the network graph.

Adversaries differ in the way they can affect and interact with a protocol. A passive adversary is only able to view communication transmitted across adversary controlled nodes. On the other hand, in addition to the capabilities of a passive adversary, an active adversary is able to fully control adversary controlled nodes - causing them to deviate from the protocol specification in any way the adversary chooses. Other types of adversaries also exist. These include a fail-stop adversary which can cause adversary controlled nodes to fail at any time. This in effect is equivalent to a disjoint network path composed of the failed node to be “cut” and thus no longer connect the sender and the receiver. Variations of the above also exist - for example a mixed adversary has the ability to corrupt a subset of nodes in an active manner and corrupt the remainder of the nodes in a passive manner.

Adversaries can also vary with regards to their computational capabilities. For a computationally limited adversary it is assumed that the amount of computation that can be carried out by such an adversary is bounded. Against such an adversary cryptographic schemes based on computationally hard problems - such as RSA, can be used. A computationally unlimited adversary on the other hand is assumed to have infinite computing power. Against such an adversary, cryptographic schemes based on unproven assumptions and computational hardness cannot be used - as it is assumed that the adversary can break their security.

Adversaries can also differ depending on the set of network nodes they can corrupt. When considering a generalized adversary structure [96], the adversary is assumed to be able to corrupt only a subset of network nodes in the adversary structure. An adversary structure \mathcal{A} is a monotone subset of the power set of all network nodes V except the sender S and receiver R of the communication. Monotonicity is thus defined as follows. Given set $P = V \setminus \{S, R\}$, an adversary structure \mathcal{A} is a family of subsets $\mathcal{A} \subset 2^P$ such that if $A \in \mathcal{A}$ and $A' \subseteq A \subseteq P$ then $A' \in \mathcal{A}$ (see [91, 96]). A t -threshold bounded adversary (which is a special case of the

general adversary structure model) has the ability to corrupt *any* t nodes in a network (where t is a constant).

Additionally, adversaries may differ with the way they can corrupt nodes. A static adversary chooses the nodes to corrupt before the execution of the secure transmission protocol initiates whilst an adaptive can choose these nodes any time after the protocol begins. Certain protocols which are secure against a static adversary are also secure against an adaptive adversary - however as shown in [152] this is not always the case. A mobile adversary on the other hand is different to static and adaptive adversaries as it has the ability to control a different subset of network nodes in each phase of a protocol execution.

Appendix C

Identifying the Best Adversary Strategy

In this appendix we show using a graph that the best strategy an adversary can follow for the protocol presented in Section 3.2 is to alter the polynomial of a single wire.

We provide evaluations and a graphical representation of the $AY + AX + BX > A$ inequality in the proof of Theorem 2 which shows it to be a valid inequality and thus confirming that the best strategy an adversary can follow is to alter the polynomial of a single adversary controlled wire.

The following table provides the evaluations (rounded to three decimal points) of the terms which appear in the inequality given values of t between 2 and 20 and when a cubic field size $q = t^3 + 1$ is used.

t	q	A	B	X	Y	AY+AX+BX
2	9	0.4	0.533	0.167	0.667	0.422
3	28	0.657	0.311	0.614	0.345	0.821
4	65	0.753	0.227	0.737	0.241	0.904
5	126	0.805	0.181	0.797	0.188	0.938
6	217	0.839	0.152	0.834	0.156	0.956
7	344	0.862	0.131	0.859	0.133	0.967
8	513	0.879	0.115	0.877	0.117	0.975
9	730	0.892	0.103	0.891	0.104	0.980
10	1001	0.903	0.093	0.902	0.094	0.983
11	1332	0.912	0.085	0.911	0.086	0.986
12	1729	0.919	0.078	0.918	0.079	0.988
13	2198	0.925	0.072	0.924	0.073	0.990
14	2745	0.930	0.067	0.930	0.068	0.991
15	3376	0.935	0.063	0.935	0.063	0.992
16	4097	0.939	0.059	0.939	0.060	0.993
17	4914	0.942	0.056	0.942	0.056	0.994
18	5833	0.946	0.053	0.945	0.053	0.994
19	6860	0.948	0.050	0.948	0.050	0.995
20	8001	0.951	0.047	0.951	0.048	0.995

The above evaluations are presented using a graph in the figure below:

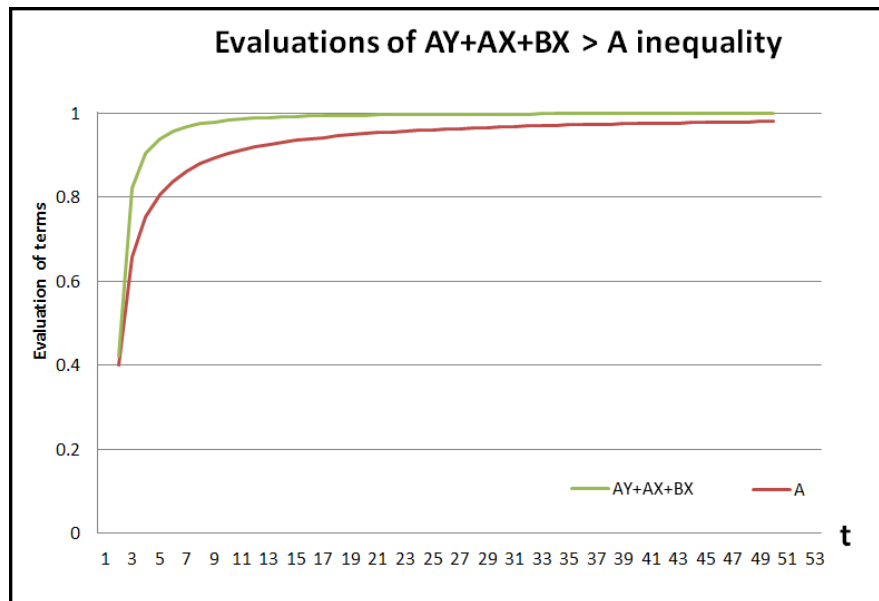


Figure C.1: Evaluations of $AY + AX + BX > A$ presented using a graph.

It is clear to see that when a cubic field size $q = t^3 + 1$ is used the $AY + AX + BX > A$ inequality is a valid inequality meaning proving that the best strategy and adversary can follow is to alter the polynomial transmitted on a single wire.

It should be noted that when using a quadratic field size such as $q = t^2 + 1$ the inequality is not valid for all values of t and thus a cubic field size in the security parameter t should be used.

Appendix D

Reconstruction of Mod 10 Secret Shared Secrets

We provide the instructions and their associated diagrammatic interpretation on how to reconstruct a secret shared using the alternative secret sharing scheme friendly to humans described in Section 2.6.2.

Instructions: From the set of shares, you will reconstruct the secret in the following manner. You will have to add each of the digits of the five numbers mod 10. This means that you will add together all the numbers corresponding to **units** and note down the **number of units** in the sum. Similarly, you will add together all the numbers corresponding to the **number of tens** and note down the **number of units** in the sum. Similarly you will do the same for the numbers which corresponds to the number of hundreds and thousands and for any other unit (if the length of the shares is greater).

We explain how to reconstruct a secret through an example. Supposing that five 4-digit shares will be used in the reconstruction of the secret and that these are the following:

$$7291 \quad 1658 \quad 9202 \quad 7484 \quad 8172$$

To reconstruct the secret you have to:

- Add all digits corresponding to units for the five numbers. In the example, these are all *emphasized* digits. Please note down the digit which corresponds to the number of units of the sum.

$$1 + 8 + 2 + 4 + 2 = 17. \text{ Here we note down } 7$$

- Add all digits corresponding to tens for the five numbers. In the example, these are all underlined digits. Please note down the digit which corresponds to the number of units of the sum.

$$9 + 5 + 0 + 8 + 7 = 29. \text{ Here we note down } 9$$

- Add all digits corresponding to hundreds for the five numbers. In the example, these are all **bold** digits. Please note down the digit which corresponds to the number of units of the sum.

$2 + 6 + 2 + 4 + 1 = 15$. Here we note down **5**

- Add all digits corresponding to thousands for the five numbers. In the example, these are all digits with no styling. Please note down the digit which corresponds to the number of units of the sum.

$7 + 1 + 9 + 7 + 8 = 32$. Here we note down **2**

We then reconstruct the secret by putting the numbers we noted down in their correct order (first digit written down as the units of the four digit code, second digit written down as the number of tens of the four digit code, third digit written down as the number of hundreds of the four digit code and the fourth digit written down as the number of thousands of the four digit code). The secret is reconstructed by correctly ordering the noted numbers. For the example, we would reconstruct the secret to be equal to 2597.

The following is a diagrammatic interpretation of the above instructions:

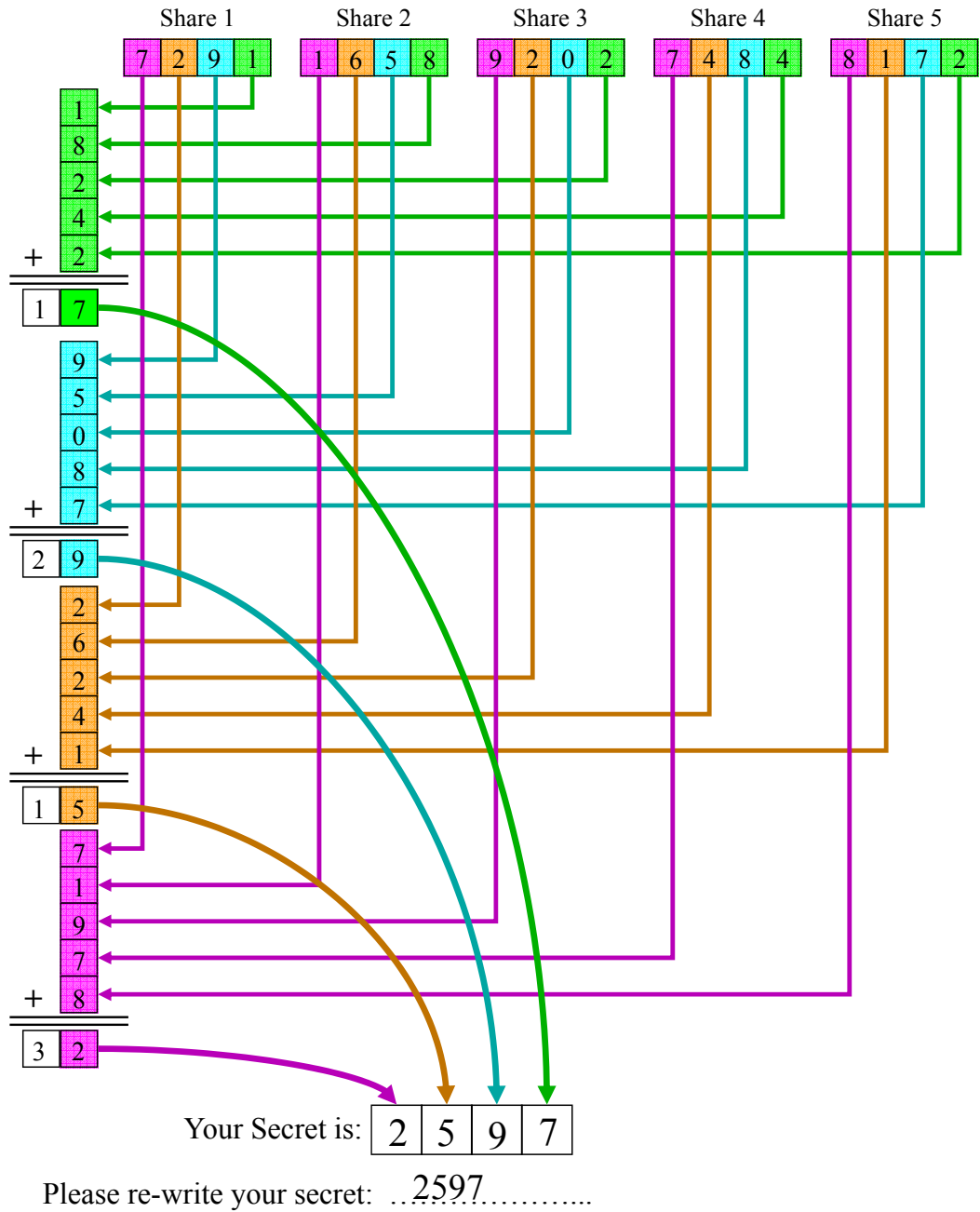


Figure D.1: Instructions on how to reconstruct a mod10 secret shared secret in figure format

Appendix E

Set Systems and Their Constructions

In this appendix details of how set systems can be constructed are given.

Set systems were used in the protocols for perfectly secure message transmission with human computation presented in Chapter 5 and were also used in the construction of perfectly secure and anonymous message transmission protocols presented in Chapter 6. In both chapters verifier set systems and generalized verifier set systems were used - with their definitions given in Chapter 5.

Ways in which such set systems can be constructed are briefly described.

E.1 Constructions from the Literature

The work by Rees et al. [154] (and other work it references) which were later used for verifiers set systems in [51] and generalized verifier sets, describes different ways in which set systems required for protocols of presented in Chapter 5 and Chapter 6 can be constructed. In this section we describe some of these constructions.

An example of a construction for a generalized (n, b, k, t) -verifiers set system is from [126] - where $k > t$, where $t + 1$ disjoint k -subsets of $\{1, \dots, n\}$ are used. For this construction, each element of X is used only once. It is easy to see that this collection of subsets satisfies properties of generalized verifier set systems. It is also easy to see that this construction achieves the property of t -confinement which is required in the protocols presented in Chapter 6. This construction is equivalent to the *Disjoint Set Systems* described in Section 5.2.2.

We now describe a more efficient construction from [126]:

Lemma 7. *When $t'(t - \frac{s-3}{2}) \leq n \leq t'(t - \frac{s-4}{2})$ for $3 \leq s \leq (t + 3)/2$ and b is even, a generalized (n, b, t', t) -verifier set system (X, \mathcal{B}) can be constructed by assuming k to be even, and constructing $t - 2s + 3$ pairwise disjoint k -subsets - denoted as A_1, \dots, A_{t-2s+3} and for $1 \leq i \leq s - 1$ constructs three sets of $(k/2)$ -subsets - denoted as B_i, C_i, D_i . A further $3(s - 1)$ k -sized blocks are constructed from the unions $B_i \cup C_i, C_i \cup D_i$ and $B_i \cup D_i$ to construct a total of $t - 2s + 3 + 3(s - 1) = t + s$ k -sized blocks. A similar construction is also presented in [126] for when b is odd.*

Proof. This construction satisfies properties of generalized verifier set systems because it ensures that a t -threshold bounded adversary cannot be present in all blocks which are created. This because for the adversary to be present in all A_1, \dots, A_{t-2s+3} blocks a presence of $t - 2s + 3$ is required (as the blocks are pairwise disjoint). For the adversary to be present

in each of the three blocks created from the union of $k/2$ -sized subsets the adversary needs to be present in two of the three subsets - meaning that the required adversary presence for these sets is $2(s - 1)$. For the adversary to be present in all sets, an adversary presence of $t - 2s + 3 + 2(s - 1) = t + 1$ is required. In this way, a t -threshold bounded adversary is not present in at least one block. This construction produces generalized verifier set systems (and in a similar manner can construct verifier set systems also), but cannot guarantee the property of t -confinement (required by protocols of Chapter 6) due to the union of sets (B_i, C_i, D_i) which are used in the construction. \square

Lemma 8. *When $n \geq t' + t + 1$, a trivial generalized (n, b, t', t) -verifier set system (X, \mathcal{B}) can be constructed when $b = \binom{n}{t'+1}$.*

Proof. \mathcal{B} is constructed by choosing all possible $(t' + 1)$ -subsets of X . As $n \geq t' + t + 1$, this guarantees that one block is free from any subset $|F| \leq t$. \square

Additional ways with which set systems can be constructed can be found in [154] and its references.

E.2 Constructions from Covering Designs

As described in Section 5.2 set systems (either generalized (n, b, t', t) -verifier set system or (v, b, t) -verifiers set system can be constructed from covering designs.

Fortunately, extensive work has been carried out in identifying various covering designs and these are publicly available through the online La Jolla Covering Repository [1] for various values of t .

When using the tables in the repository one must note that the constructions are for covering designs and that the parameters for set systems have to be altered appropriately. In the repository they use v to denote the size of set X which we denote with n . In the repository k is used to denote the size of a cover design block. When constructing a set system from a cover design, the size of a generalized verifier set system block t'^1 will equate to $t' = v - k$ and the number of block b will be the same as the number written in the tables corresponding to v and k written in the tables.

We now describe the construction of a generalized $(9, 12, 6, 2)$ -verifier set system using one of the constructions from the repository. The generalized verifier set system to be presented also achieves the property of t -confinement (see Definition 8).

We use the construction of the $(9,3,2)$ covering design whose 12 blocks are presented in the following table - using numbers to represent each of the 9 elements of set X . We also present the corresponding blocks of the generalized $(9, 12, 6, 2)$ -verifier set system constructed using this cover design. Close inspection of the constructed blocks identify them to part of a generalized verifier set system. Furthermore, they also satisfy the t -confinement property.

Further constructions of set systems can be constructed from the extensive examples available from the La Jolla Covering Repository [1]. It should be noted that the repository currently

¹It should be noted that the size of blocks for verifier set systems as per Lemma 4 is always $t + 1$ thus v and k will have to be selected accordingly.

Block Number	Cover design blocks	Generalized verifier set system block
1	1 4 7	2 3 5 6 8 9
2	2 5 8	1 3 4 6 7 9
3	3 6 9	1 2 4 5 7 8
4	1 6 8	2 3 4 5 7 9
5	2 4 9	1 3 5 6 7 8
6	3 5 7	1 2 4 6 8 9
7	1 5 9	2 3 4 6 7 8
8	2 6 7	1 3 4 5 8 9
9	3 4 8	1 2 5 6 7 9
10	1 2 3	4 5 6 7 8 9
11	4 5 6	1 2 3 7 8 9
12	7 8 9	1 2 3 4 5 6

Table E.1: Blocks of the cover design and the corresponding generalized verifier set system blocks constructed using the cover design.

only displays constructions for up to $t = 8$. For greater values of t it is recommended to use constructions from the literature as outlined in the previous section.

Appendix F

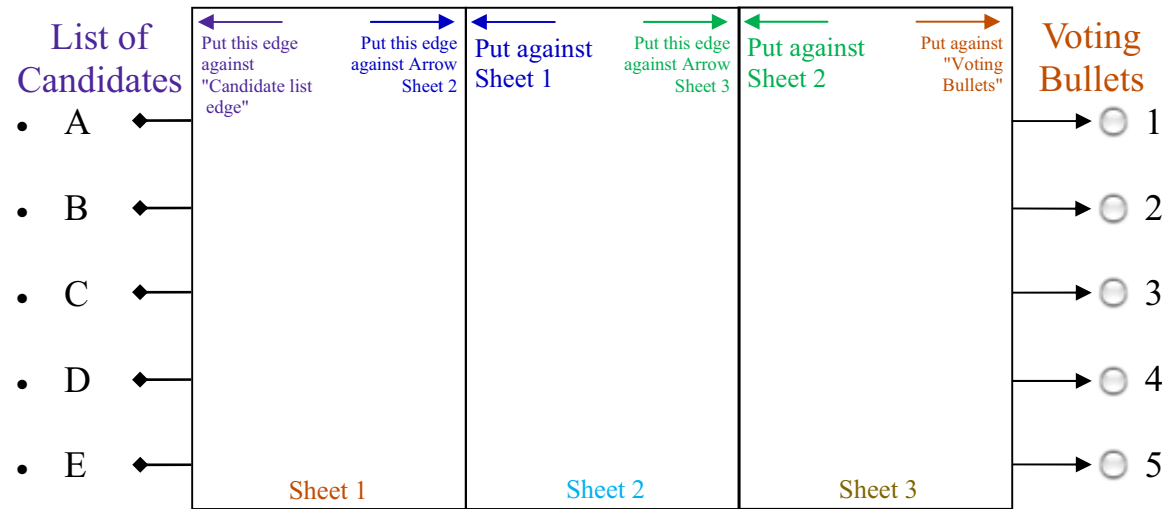
Permutation voting experiments

In the following pages, the sheets of paper given to participants for the permutation experiments are presented.

Appendix B: Permutation voting experiments

You will be given two sets of three sheets that contain various lines.

For each of the sets please place them in the right location relative to the diagram below:



For each of the sets, by following the lines identify the voting bullet number which corresponds to candidates A and D.

First Method

Second Method

Candidate **A** voting bullet number:

Candidate **A** voting bullet number:

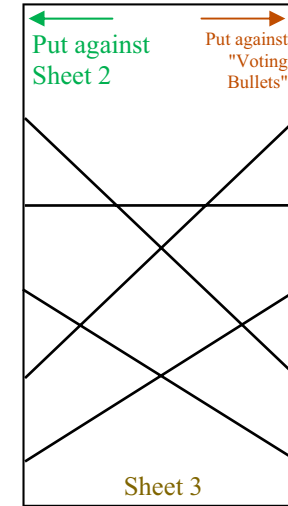
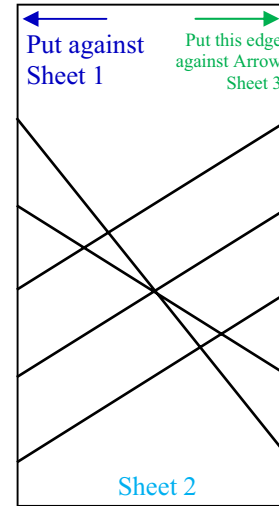
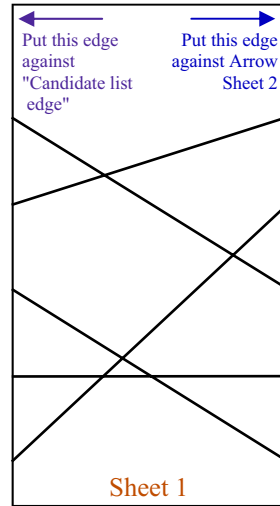
Candidate **D** voting bullet number:

Candidate **D** voting bullet number:

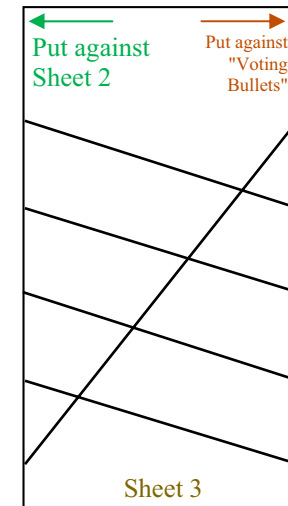
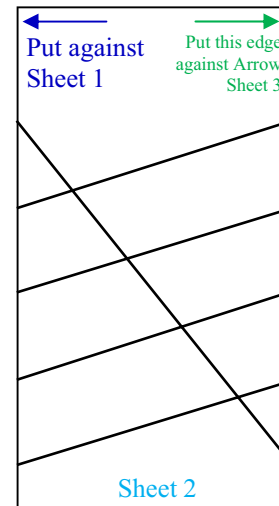
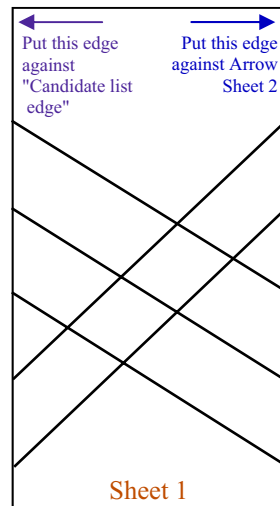
Do you have a university degree?

Please state your age:

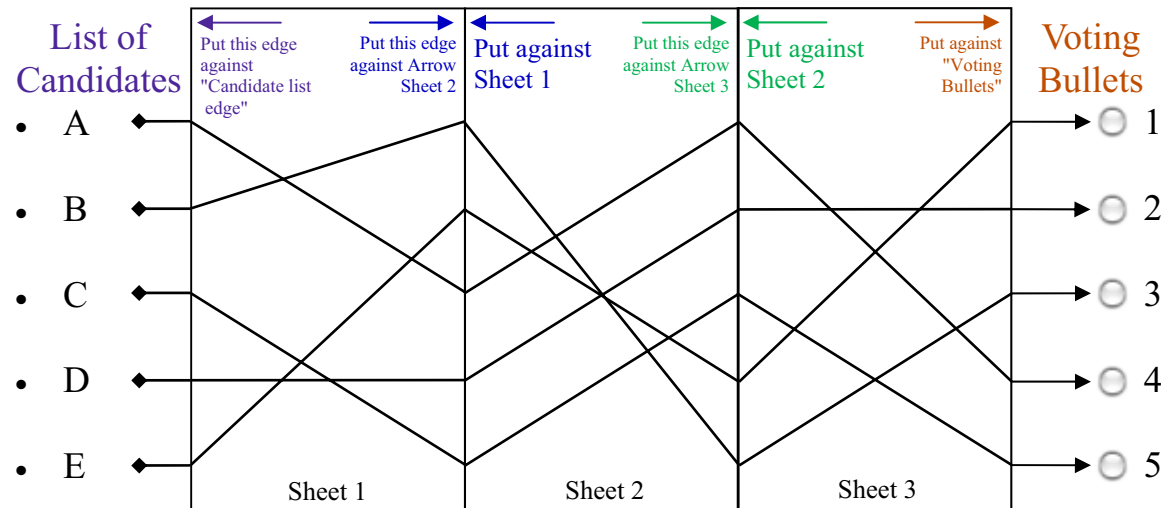
First set of sheets
given to participants
(given as three small
pieces of paper)



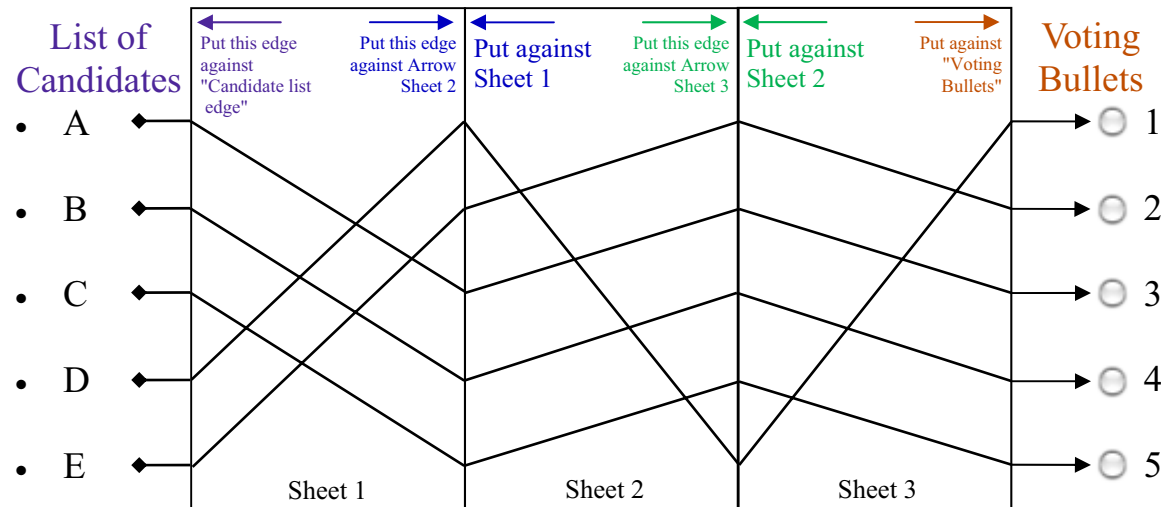
Second set of sheets
given to participants
(given as three small
pieces of paper)



Participant view
when sheets were
placed correctly for
first set of sheets



Participant view
when sheets were
placed correctly for
second set of sheets



Appendix G

Mod 10 Voting Experiments

In the following pages, the sheets of paper given to participants for the mod 10 experiments are presented.

Appendix C: Mod 10 voting experiments

You will be given five 4-digit numbers. From these, you will create the code with which to vote with. You can do this by adding each of the digits of the five numbers mod 10 (there is no need to know what this is as it is explained). This means that you will add together all the numbers corresponding to units and note down the number of **units in the sum**. Similarly, you will add together all the numbers corresponding to the number of tens and note down the number of **units in the sum**. Similarly you will do the above for the numbers which corresponds to the number of hundreds and thousands.

We explain the above process through an example.

Supposing the five 4-digit numbers are the following:

- | | | | |
|---|---|---|---|
| 7 | 2 | 9 | 1 |
| 1 | 6 | 5 | 8 |
| 9 | 2 | 0 | 2 |
| 7 | 4 | 8 | 4 |
| 8 | 1 | 7 | 2 |

To create your code you have to:

- Add all digits corresponding to units for the five numbers. In the example, these are all **green** highlighted digits. Please note down the digit which corresponds to the number of units of the sum.

$$1 + 8 + 2 + 4 + 2 = 17 \rightarrow \text{Here we note down } 7$$

- Add all digits corresponding to tens for the five numbers. In the example, these are all **blue** highlighted digits. Please note down the digit which corresponds to the number of units of the sum.

$$9 + 5 + 0 + 8 + 7 = 29 \rightarrow \text{Here we note down } 9$$

- Add all digits corresponding to hundreds for the five numbers. In the example, these are all **orange** highlighted digits. Please note down the digit which corresponds to the number of units of the sum.

$$2 + 6 + 2 + 4 + 1 = 15 \rightarrow \text{Here we note down } 5$$

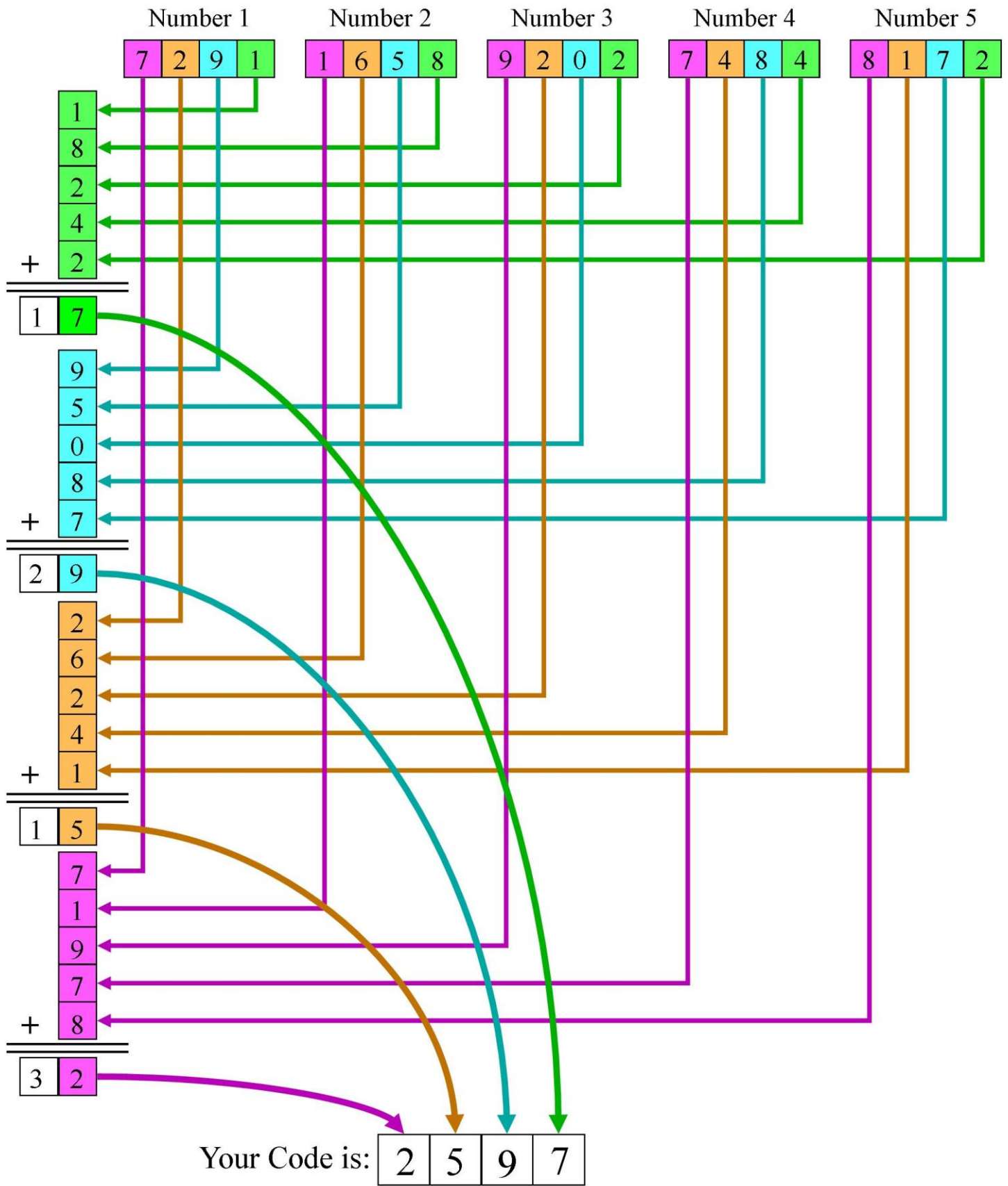
- Add all digits corresponding to thousands for the five numbers. In the example, these are all **pink** highlighted digits. Please note down the digit which corresponds to the number of units of the sum.

$$7 + 1 + 9 + 7 + 8 = 32 \rightarrow \text{Here we note down } 2$$

We then create a four digit code by putting the numbers we noted down in their correct order (first digit written down as the units of the four digit code, second digit written down as the number of tens of the four digit code, third digit written down as the number of hundreds of the four digit code and the fourth digit written down as the number of thousands of the four digit code). For the example we would get the code to equal 2597.

On the next page we also present this example in the code generation form supplied.

EXAMPLE – How to use the code generation form



Please re-write your code:2597.....

Based on the example given above and the code generation form example, create your own code from the five 4-digit numbers below for Candidates A and D.

You can either calculate the code yourself (as in the above example given) or use the empty code generation form supplied on pages 4 and 5.

Number 1

Number 1	Candidate A				Candidate B				Candidate C				Candidate D				Candidate E			
	6	9	7	3	6	1	4	2	4	2	5	8	4	8	7	8	0	7	8	9

Number 2

Number 2	Candidate A				Candidate B				Candidate C				Candidate D				Candidate E			
	7	4	6	5	0	2	2	4	4	2	5	8	2	9	1	5	7	4	6	5

Number 3

Number 3	Candidate A				Candidate B				Candidate C				Candidate D				Candidate E			
	0	6	7	4	6	5	9	6	7	9	5	3	6	8	4	4	7	5	3	8

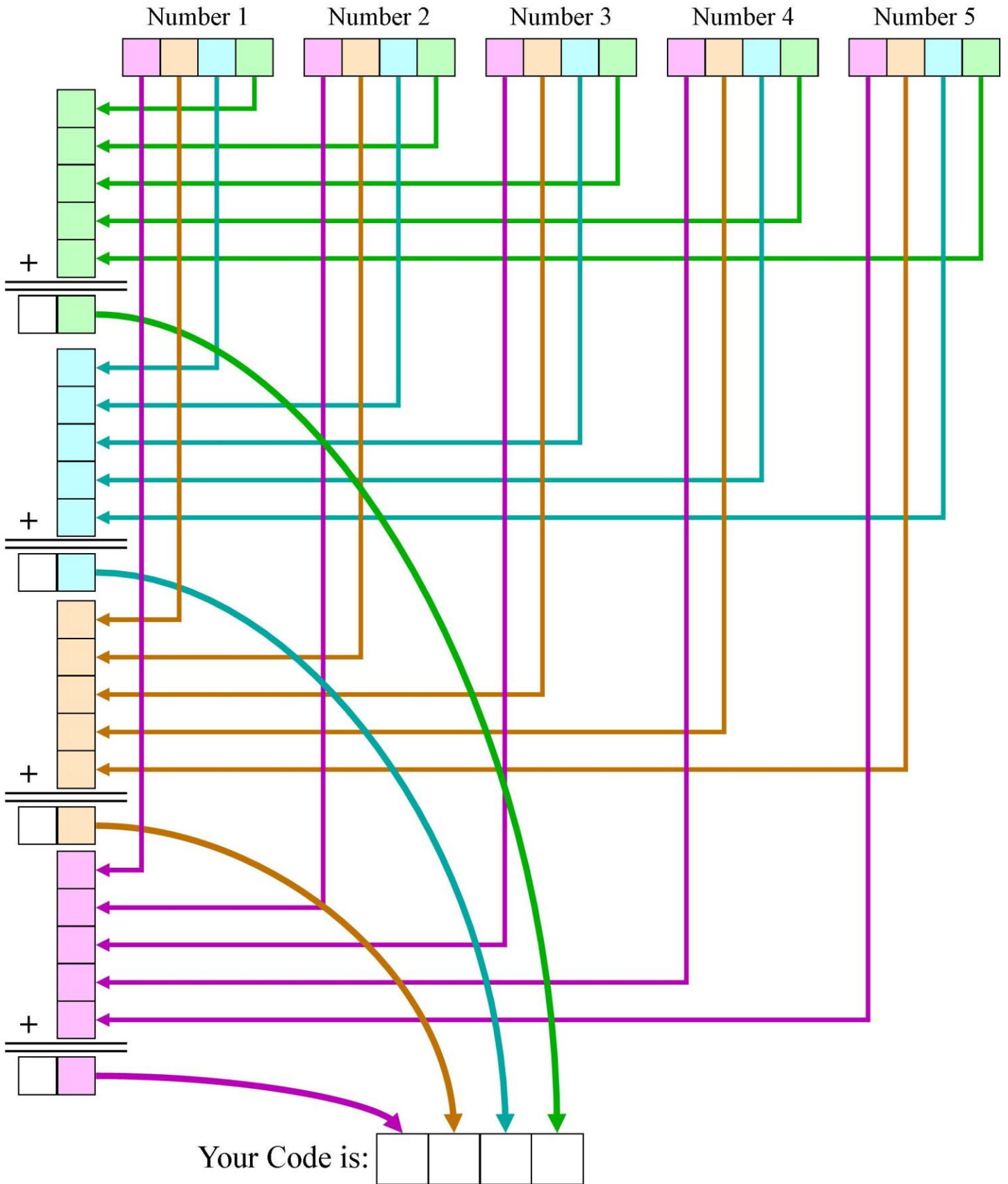
Number 4

Number 4	Candidate A				Candidate B				Candidate C				Candidate D				Candidate E			
	5	3	2	8	8	5	7	2	7	3	0	7	0	3	6	3	1	9	2	2

Number 5

Number 5	Candidate A				Candidate B				Candidate C				Candidate D				Candidate E			
	1	4	9	1	9	8	7	9	8	8	1	7	0	0	9	2	2	8	7	4

Please create your own code for **Candidate A**

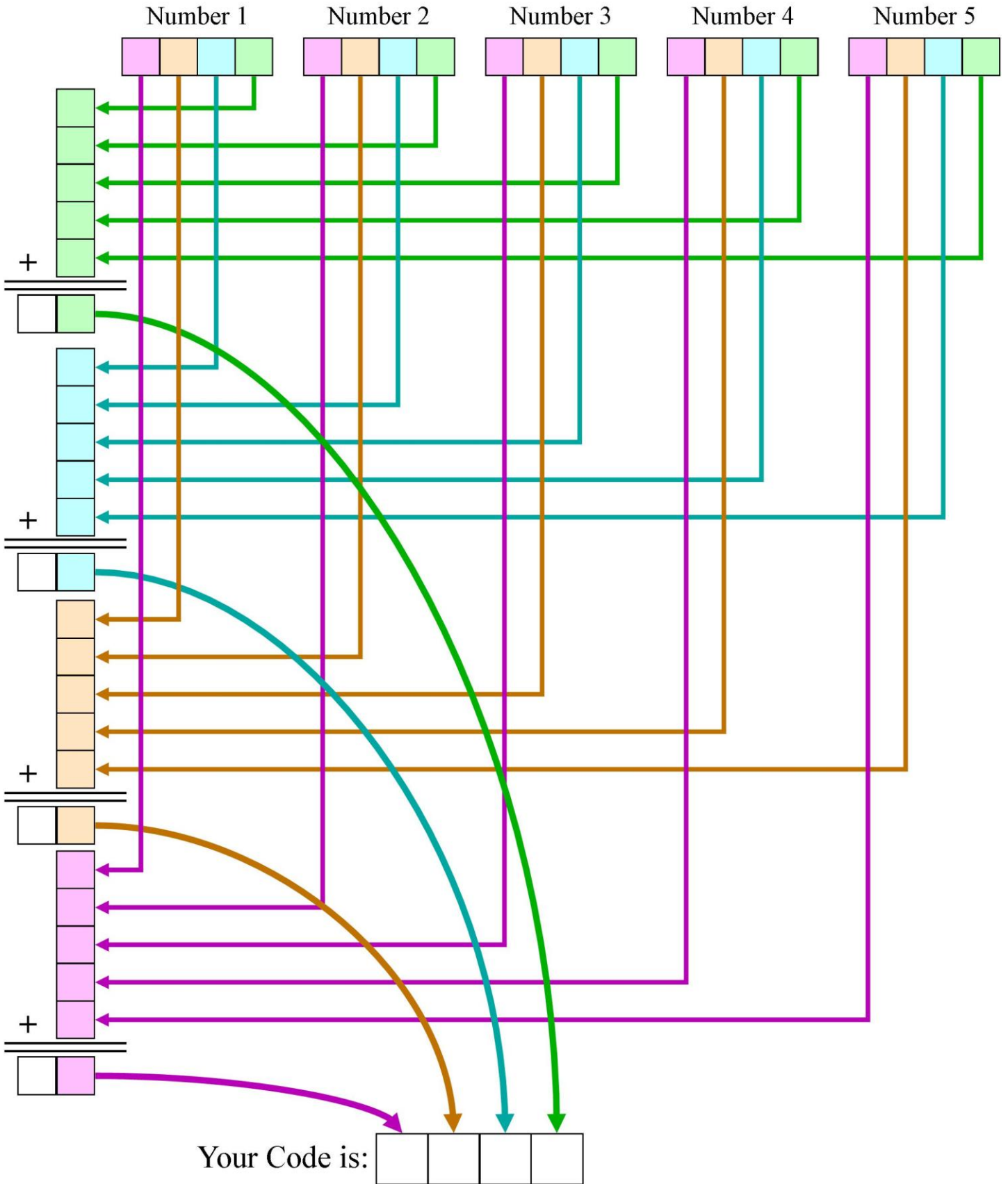


Please re-write your code:

Do you have a university degree?

Please state your age:

Please create your own code for **Candidate D**



Please re-write your code: