Efficient Protocols For Secure Message Transmission Over Directed Networks

Thesis submitted in partial fulfilment
of the requirements for the degree of

Doctor of Philosophy
in
Computer Science and Engineering

by

Ravikishore Vasala
201099006
ravikishore.vasala@research.iiit.ac.in

International Institute of Information Technology
(Deemed to be University)
Hyderabad - 500 032, INDIA
March, 2018

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

This is to certify that the thesis entitled **"Efficient Protocols For Secure Message Transmission Over Directed Networks"** submitted by **Ravikishore Vasala** to the International Institute of Information Technology, Hyderabad, for the award of the Degree of **Doctor of Philosophy** is a record of bona-fide research work carried out by him under my supervision and guidance. The contents of this thesis have not been submitted to any other university or institute for the award of any degree or diploma.

Date	Adviser: Dr. Kannan Srinathan

To my parents, my wife and my lovely daughters

Acknowledgements

I am fortunate for having Dr. Kannan Srinathan as my doctoral advisor. Without him this thesis would have never existed. Apart from being a great scientist and constant source of inspiration, he is a good teacher, dear friend and very good philosopher. A simple conversation with him enlightens anyone both academically and spiritually. If I did not meet him, I doubt if I would have ever understand the purpose of research. He has tremendous influence on my professional as well as personal development. The way he understands the essence of any complex topic is amazing; and coming up with the fundamental relation (or philosophical aspects) that connects different concepts (in no time) is beyond one's imagination. He is "the" role model everyone wants to mimic and I am no exception to that. It is an absolute bliss working under his able guidance. I want to "thank him" for being my advisor, though words are not enough to express my gratitude.

I would like to thank my doctoral thesis committee members Dr. Ashok Kumar Das, Dr. Kishore Kothapalli and Dr. Susmita Ruj along with my advisor for their invaluable suggestions and critical remarks offered on my earlier drafts of the thesis. Their remarks were crucial in finishing and presenting the final draft of the thesis in a better way. Special thanks to Dr. Kishore Kothapalli for helping me in writing my very first research article. Also, I would like to thank Dr. Indranil Chakrabarty, Dr. Prasad Krishnan, and Prof. C N Kaul for their encouragement and support. Further, I wish to express my sincere thanks to my friend, senior and well-wisher Dr. Sumit Kumar Pandey for his positive words and critical feedback on my PhD research work.

I wish to thank my friends (from IIIT-H) Anand, Anandaswarup, Arun, Chiranjeevi, Deepthi, Gopi, Jatin, Karthik, Kaushik, Mohan, Mulu, Nagender, Nikhil, Niraj, Peeyush, Prabhu, Pranav, Ramakrishna, Ramesh, Ravi Prasad, Shashank, Siddartha, Subhasree and Vasu. They made my days at IIIT-H beautiful and memorable. Special mention to my close friends Chiranjeevi and Nagender for being with me all these years through all my ups and downs. They both made my days at IIIT-H even more beautiful. Further, I wish to extend my thanks to my friends (outside of IIIT-H) Ambika, Harish, Mamatha, Krishna, Rakesh, Sreenu and Sudarshan for their company and positive words.

Besides my advisor, I would like to thank my collaborators and students who played an important role in my professional development; namely, Anupriya, Ashutosh, Bhanu, Chiranjeevi, Durgesh, Kavya, Meghana and Tushant. Further, I wish to express my special thanks to prof. Alfred Menezes and anonymous reviewers for their critical reviews and excellent feedback on my various research articles, that helped me in improving my writings. Without them, none of my research articles/results would have been possible.

I am grateful to IIIT-H and C-STAR for providing me wonderful research atmosphere. Moreover, having an association with IIIT-H in various capacities, starting from PhD student to teaching assistant to lecturer and finally to visiting faculty, is always an honour to me. Further, I wish to express my sincere thanks to Tata Consultancy Services (TCS) for financially supporting my PhD as well as conference visits abroad. Also, I wish to thank Prabhakar, Manohar and Phani for their help in dealing with various administerial works without any hassles.

This list is incomplete without mention of the most important people in my life: my parents, my wife and my lovely daughters. Amma and Nanna always encouraged and stood with me to pursue my goals. I am blessed to have them as my parents, especially for their unconditional love. Special mention to my wife Rajitha, I owe a lot to her. She is the backbone for me and my family. It is an understatement to say that she has equally contributed to this thesis; without her, finishing this PhD would have remained as a dream. I am lucky to have her as my life partner. Also, I must thank my lovely young daughters Sreshta and Swechha. These two angels always bring me loads of happiness and joy. I have had and will continue to have wonderful moments with them to cherish in life.

Please forgive me if I have missed naming anyone explicitly.

Abstract

In a large distributed *network* of interconnected nodes, the goal of any `Secure Message Transmission` (`SMT`) protocol is to *securely* deliver the sender's message at the receiver's end in the presence of a computationally unbounded *adversary*, that can partially control the network by *corrupting* some of its nodes (except the sender and the receiver).

In the literature, different variants of `SMT` problem have been studied depending on the type of network model and/or the types of node corruptions. Undirected graph model and directed graph model are two of the most well-studied network models. Different types of node corruptions studied are passive, fail-stop, omission and Byzantine. In passive corruption, the adversary is only allowed to read/eavesdrop the corrupted node. In fail-stop corruption, the adversary can crash the corrupted node at its will. In omission corruption, the adversary can eavesdrop and/or crash the corrupted node. In Byzantine corruption, the adversary fully controls the actions of the corrupted nodes.

In an undirected graph setting, for most variants of `SMT` problem whether the corresponding `SMT` protocols exist or not is known in the literature. Moreover, whenever a protocol exists, communication efficient protocols have been designed too. Whereas, in directed graph setting very little is known about the existence of protocols (for different variants of `SMT` problem) as well as the design of efficient protocols. This thesis primarily focuses on bridging this gap between undirected and directed graph settings.

It is usually difficult to design efficient protocols in arbitrary directed graphs due to the plethora of different cases/possibilities that usually need to be dealt with. However, in the literature due to the simplicity and ease of protocol design, *wires* model is considered which is a special case of directed graph setting. In *wires* model, paths (known as *wires*) from the sender to the receiver and vice-versa are considered whereas the weak paths are discarded. On the other hand, if the intermediate nodes can generate random-coins and/or perform computations, it is known that the directed wires-based abstraction is inadequate to capture arbitrary directed graphs. In this thesis, we design efficient protocols for different variants of `SMT` problem in both *wires* model as well as arbitrary directed graph model. Moreover, we introduce a new model namely *routing* model to answer the following fundamental question. If each intermediate node is a mere router (can not do computations but can route message to its neighbours) then under what condition(s) is given variant of `SMT` (im)possible? Furthermore, in arbitrary directed graph setting, we design a protocol which is both communication as well as round/hop optimum when the adversary is passive.

Contents

List of Figures

Chapter 1

Introduction

1.1 Overview of Secure Message Transmission in distributed environment

Consider the following problem: In a large network of interconnected nodes, two specially designated nodes, namely the sender and receiver, are connected by *multiple* channels. The network is under the influence of the adversary, that can partially control the network by corrupting some of its nodes. The sender wishes to deliver its message to the receiver such that the adversary learns nothing about the message. In reality this can be achieved because it may be *hard* for the attacker to gain complete control over the entire network as (s)he may not possess enough resources. Therefore, by designing appropriate protocols, one can achieve *security* relying on the inability of the attacker controlling the whole network at any given instant. In this approach, the sender encodes its message as a function of sub-messages; and, the sender skilfully distributes these sub-messages across the network such that enough information is always present in the network to reconstruct the message. The receiver reconstructs the message once it receives a sufficient number of sub-messages. At the same time, it is difficult for the attacker to learn any information about the message as (s)he can not control the whole network, thus bound to miss a substantial number of sub-messages. This is the study of *Secure Message Transmission*(`SMT`) in a distributed environment [1]. An informal definition of `SMT` in a distributed network is defined as follows.

Definition 1.1. *In a distributed* **network** *of interconnected nodes, the goal of any* `SMT` *protocol is to* **securely** *deliver the sender's message at the receiver's end in the presence of a computationally unbounded* **adversary***, that can partially control the network by* ***corrupting*** *some of its nodes (except the sender and the receiver).*

Below, we briefly explain each of the items in bold text, namely, **network, security, adversary, node corruption** and the formal definitions are deferred to the next chapter.

1. **Network**: We model the underlying distributed network as either undirected graph or directed graph. In undirected graph setting, a node A can communicate with a node B if and only if B can communicate with A. The same is not true in directed graph setting, where if A can communicate with B then it necessarily does not mean that B also can communicate with A.

2. **Security**: Each secure message transmission protocol studied in this thesis is either perfectly secure or unconditionally secure. For example, loosely speaking, a message transmission protocol is perfectly reliable if the message received by the receiver is identical with the sender's message without any error. Similarly, in an unconditionally reliable message transmission protocol, the receiver must receive the message with very high probability (error must be arbitrary low/negligible). A message transmission protocol is perfectly secure if it is perfectly reliable and additionally satisfies the condition that no information is leaked to the adversary (information theoretic security) about the message. Similarly, a message transmission protocol is unconditionally secure if it is unconditionally reliable and the adversary can learn information about the message only with negligible/zero probability.

3. **The adversary**: The adversary is either threshold or non-threshold. If the adversary is threshold, then the adversary can corrupt up to a fixed threshold number of nodes. In non-threshold setting, the adversary can corrupt any one set of nodes from the given collection of sets of nodes. Moreover, the adversary can be either static or mobile. If the adversary is static, the nodes once corrupted remain corrupted subsequently throughout the entire protocol execution. And, if the adversary is mobile, then the adversary can choose to corrupt different set of nodes in different rounds (see Definition 2.15) of the protocol execution.

4. **Various types of node corruption**: The adversary can corrupt the nodes in any of the following ways.

   (a) **Passive**: In passive corruption, the adversary can just eavesdrop the corrupted node.

(b) **Fail-Stop**: In fail-stop corruption, as long as the node is alive it works properly, otherwise the node fails to work further. Here, the adversary cannot even eavesdrop the node.

(c) **Omission**: In omission corruption, the adversary can corrupt the node passively or in fail-stop fashion or both.

(d) **Byzantine**: In Byzantine corruption, the adversary has full access to the corrupted node, can eavesdrop and/or modify the protocol code of the corrupted node.

(e) **Mixed**: In mixed corruption, the adversary can corrupt some of the nodes passively, some of the nodes in fail-stop fashion, some of the nodes in omission fashion and some of the nodes in Byzantine fashion.

1.2 Why to study SMT in distributed environment?

One of the fundamental goals of cryptography is to *securely* deliver the sender's message at the receiver's end in the presence of the *adversary.* And, the typical assumption is that the sender and receiver are connected by a *single* channel for which the adversary may possess access. The adversary's capacity to access the channel can be limited to just *eavesdropping* or can be allowed to have complete access, including modifying the messages passing through the channel. The required degree of *security* can be (1) perfect (2) unconditional or (3) computational. "Informally, the meaning of computational security is that the adversary should not learn (any function on) the sender's message in polynomial/reasonable time".

We have a straight-forward solution for this secure communication problem if both the sender and receiver have a *shared key* (which is unknown to the adversary), that is agreed in advance. The sender simply *encrypts* the message using the *pre-shared key* and sends it to the receiver. The receiver, upon receiving the encrypted message, using the same *pre-shared key*, *decrypts* the encrypted message and gets the original message. This is exactly the study of the *symmetric key cryptography.* The symmetric key encryption systems like *one-time pad, AES* and *DES* are popular among others [2].

The *one-time pad* cryptosystem is one of the finest examples which achieves perfect security. In this cryptosystem, the set $\{0, 1\}$ is both the message space as well as the key space. The sender sends a bit $b_s \oplus b_k$ to the receiver, where b_s is the sender's message and b_k is the shared key. Upon receiving $b_s \oplus b_k$, the receiver computes $(b_s \oplus b_k) \oplus b_k$ which is the intended message b_s. Though *one-time pad* cryptosystem achieves perfect security, its inherent limitation is that the necessity of the existence of another (equivalently fast) communication channel between the sender and receiver by which they can exchange the *key* securely, before the *actual* communication takes place. This is a big assumption. And, in fact, in the legendary work of Shannon [3], it is proved that perfect security is impossible to achieve in practice. More elaborately, the following conditions should be satisfied simultaneously to achieve perfect security against a *computationally unbounded* eavesdropping adversary.

1. The entropy of the key-space should be at least equal to the entropy of the message-space.

2. The channel used for shared key establishment can not be used for sending the *actual* message.

3. The key can not be reused to send *another* message securely.

Once this limitation is known to be inherent, researchers came up with the new paradigm called *computational security* as an alternative to perfect security. The motivation behind this paradigm is that, in practice it is unrealistic that the adversary possesses *unbounded* computational power and the amount of time the message needs to be safeguarded is not *infinite*. Moreover, we may allow negligible error in reliability and/or security. These two *reasonable* assumptions open the new line of research, called *public key cryptography (PKC)*, which completely changed the face of cryptography. However, the *computational security* of any such system depends on the (conjectured) *hardness* of solving a particular problem. In fact, those systems wherein the adversary must solve the corresponding *hard* problem in order to break the system in reasonable/polynomial time, are called *provably* secure systems. However, design of proven (instead of just "provable") secure systems is believed to be out-of-reach of contemporary mathematics because it necessarily answers the famous question of whether the *class* of P is same as the *class* of NP or not.

Given the limitation(s) of achieving perfect security in practice, the other distinguished approach followed to achieve perfect security is through the distribution of information. That is exactly the study of `SMT` in distributed environment.

The other motivation for studying `SMT` in distributed network is that some of the main cryptographic primitives, like Byzantine Agreement (BA) [4–6], Secure Multi-Party Computation (SMPC) [7–10], Verifiable Secret Sharing (VSS) [8, 11], assume that the network is complete, i.e., every pair of parties/players directly connected by a secure channel. In general, it is not economical to have a secure channel between every pair of players. However, when they are part of a large network, in the absence of a direct channel, sometimes it is possible to virtually *simulate* the corresponding secure channel with the help of *real* channels. The study of `SMT` explores the (im)possibility of simulating such virtual channels.

1.3 Different variants of SMT problem

One can study many (different) variants of `SMT` problem. To name a few for example:

1. In a given (un)directed graph, under what condition(s) is secure message transmission (im)possible tolerating a static threshold adversary which can corrupt up to t nodes, where nodes corruption can be passive, fail-stop, omission or Byzantine and security requirement can be either perfect or unconditional?

2. Similarly, we can ask: In a given (un)directed graph, under what condition(s) an r-round secure message transmission protocol exists tolerating a static threshold adversary that can corrupt up to t nodes, where nodes corruption can be passive, fail-stop, omission or Byzantine and security requirement can be either perfect or unconditional?

3. In a given (un)directed graph, under what condition(s) is secure message transmission (im)possible tolerating a static threshold mixed adversary which can corrupt up to t_p nodes passively, t_f nodes in fail-stop fashion, t_o nodes in omission fashion and t_b nodes in Byzantine fashion, where security requirement is either perfect or unconditional?

In any variant of the `SMT` problem, the following parameters play an important role.

1. **Characterization/Possibility**: Given a particular variant of `SMT`, what are the necessary and sufficient conditions under which the corresponding `SMT` is (im)possible?

2. **Feasibility**: Given the necessary and sufficient condition(s) under which a particular variant of `SMT` is possible, does there exists any efficient algorithm which tests whether or not the network meets the corresponding condition(s)?

3. **Round complexity**: If a particular variant of `SMT` is possible then is it possible to design round efficient/optimal `SMT` protocols?

4. **Communication Complexity**: If a particular variant of `SMT` is possible then is it possible to design communication efficient/optimal `SMT` protocols?

1.4 Brief overview of the existing results

1. Dolev et al. [1] in their pioneering work, introduced the Secure Message Transmission(`SMT`) problem and, studied the trade-off between the network connectivity and the existence of `SMT` protocols, tolerating a threshold adversary. Since then, `SMT` problem received considerable attention [12–16]. Researchers have worked on interesting generalizations in different dimensions, including hypergraphs (e.g., [17–19]), non-threshold adversaries (e.g., [20, 21]), mobile faults (e.g., [22–26]), mixed/hybrid faults (e.g., [15, 27–31]), asynchronous networks (e.g., [28, 32–37]), to name a few.

2. Dolev et al., without loss of generality abstracted the given undirected network as a collection of vertex-disjoint paths (called wires) between the sender and the receiver. The wires based abstraction is justified by Menger's theorem [38], which states that in an undirected graph, the minimum number of vertices required to disconnect two nodes S and R is the maximum number of pair-wise vertex-disjoint paths exist between S and R. More precisely, suppose an intermediate node between S and R is corrupted by the adversary then every path, irrespective of its length, passing through that node is also corrupted. Therefore, all the paths between S and R passing through that node considered as a single wire between S and R. Later, Wang et al. [14] were motivated by the fact that sometimes, `forward` channel, a path (see Definition 2.11) from the sender to the receiver, may be economical and `feedback` channel, a path from the receiver to the sender, may be expensive but not impossible. Accordingly, they modelled the network as a directed graph and considered forward channels as well as feedback channels. In this network model, Wang et al. followed the directed wires based approach – no pair of forward (resp. feedback) channels share a common vertex (except the sender and receiver). Moreover, no pair of a forward channel and feedback channel share a common vertex. This directed wires model received considerable attention and is well studied in the literature [39–42].

3. However, if the intermediate nodes can generate random-coins or perform computations, it is known that the directed wires-based abstraction is inadequate to capture general digraphs [43, 44]. In the following, we elaborate more on directed graph setting.

1.4.1 SMT problem in directed graph setting

Unlike `SMT` in undirected graphs that include elegant characterizations [1], optimal protocols [45], and sometimes with constant overheads [46], the existing literature on `SMT` in *digraphs* typically have at least one of the following limitations:

1. *Lack of Generality*: Due to Menger's theorem [38], it is noted that with respect to `SMT` protocols, undirected graphs may be, with no loss in generality, abstracted as a set of disjoint paths (also called *wires*) between S and R. However, it is proved in [47] that there is a *loss* in generality if the wires-model is used for digraphs. Notwithstanding, due to the simplicity and ease of protocol design, the wires-model has been used in [14, 21, 29, 39, 40, 48], trading-off generality.

2. *Characterization is Not Efficiently Testable*: The characterization of the published theorems of [47,49] are not (yet) efficiently testable, that is, given a digraph G, deciding if the (necessary and sufficient) condition is satisfied, is not known to be in the class **P**. This is a direct consequence of the fact that digraphs are a strict generalization of the undirected case and are devoid of symmetries present in undirected graphs, reducing the elegance of the resultant characterizations.

3. *Protocol is Inefficient*: It is usually difficult to design efficient protocols in digraphs due to the plethora of different cases/possibilities that usually need to be dealt with. Specifically, one can easily relate the flow of time with directed arcs and hence techniques for dealing with protocols in digraphs are often (at least) as difficult as the techniques for the design of round optimal protocols in undirected graphs (which of-course is typically hard). Likewise, extant protocols designed for arbitrary digraphs have typically been recursive with super-polynomial depth in the worst-case, leading to impracticality. It is therefore important to study less-general settings that facilitate efficient protocols.

Natural question is, then why to study `SMT` in directed graph setting knowing these limitations?: Despite these limitations, it is interesting and worthwhile to study distributed algorithms in digraphs for multi-fold reasons including (a) digraphs are *theoretical* generalizations and theorems proved for digraphs are far-reaching, (b) *practical* settings, where A can send data to B but not vice-versa, do exist and requires the design of appropriate protocols, and (c) interestingly, *round-optimal* protocols in undirected graphs are easily designed if some protocol (not necessarily optimal) is discovered for digraphs [47].

1.5 Our motivation and contributions

- **Motivation and contribution #1**: There are settings where answers to none of the aforementioned four questions (Characterization, Feasibility, Round complexity and Communication complexity) are known yet. For instance, we do not know of a necessary and sufficient condition on digraphs influenced by a Byzantine adversary corrupting up to any t nodes for the existence of protocols for perfectly secure message transmission(`PSMT`) from S to R [50]; not to mention, the design of optimal protocols for the same are still far-fetched. Researchers have therefore attacked the `PSMT` problem in scenarios that are not as general as mentioned above – the harder the inquiry, the more specific the chosen setting.

  It is proved that `PSMT` tolerating up to t Byzantine faults is possible *if and only if* there are at least $(2t+1)$ vertex-disjoint paths between S and R. Further, the protocols are efficient too. However, designing round optimal protocols for `PSMT` (even in undirected graphs) still remains a hard open problem. Consequently, results are known only with further restrictions.

  A setting where round-optimal protocols have been designed (on arbitrary digraphs) is when a small probability of error is permitted [47] (that is, perfectness is negligibly traded-off). However, the design of communication optimal solutions is still open as mentioned in [49].

  A particular setting where communication optimum protocols for `PSMT` are designed is the following: applying Menger's theorem [38], the undirected graph can be abstracted as a collection of wires (vertex-disjoint paths) between S and R, up to t among which are corrupted by the adversary. In this setting, a two phase[1] protocol for `PSMT` that is optimal in communication complexity is known [45]. While the notion of phase complexity has been studied in the works of [15,45,51–53], we stress that round complexity (e.g., [24,25]) is markedly different from phase complexity, even in the case of undirected graphs (see Chapter 3).

  Recently, restricting to passive adversaries, Renault *et al.* [50] characterized the digraphs that enable `PSMT`. In fact, *Renault et al.* in [50] use a more general non-threshold adversary model, characterized via an adversary structure, which is a collection of subsets of nodes in the graph, wherein the adversary may choose to corrupt (passively in this case) the nodes in any one subset from the collection.

[1] A phase is a send from the sender to the receiver or vice-versa.

The protocols of [50] are, therefore, not always efficient (that is, may be super-polynomial in n) as discussed in [49].

In summary, all the four questions in our inquiry, with respect to the problem of PSMT, have remained open in the general case of digraphs influenced by a Byzantine adversary characterized via an adversary structure. However, (im)possibility results are known if one restricts the setting to either undirected graphs [43] or passive adversary or security with error (e.g., [49, 50]). Nevertheless, efficient protocols are still elusive. To design efficient protocols using contemporary techniques, further restriction (apart from moving to undirected graphs) is required, namely, *threshold* adversary. For instance, Dolev *et al.* in [1] have given one such efficient protocol, which, however, is neither round optimal nor bit-optimal.

Round-optimal protocols are known only in the case of weaker (not perfect) security models like statistical [47] or computational security [54]. Bit-optimal protocols have been designed in the wires-based abstraction of the undirected graph in [45]. While a similar wires-based approach has been used for digraphs too in [14], it is known to be inadequate to capture all digraphs on which protocols exist as shown in [47].

We ask: *does restricting to the setting of passive threshold adversaries lead to the design of efficient and round-optimal and/or communication optimal protocols?* Or, are further restrictions like wires-based abstractions still required?

Interestingly, we design communication efficient and round optimal protocols, with no further restrictions beyond assuming that the adversary passively corrupt up to t nodes in the digraph. Incidentally, it turns out that our techniques for designing round-optimal protocols are orthogonal to those that entail linear communication complexity – therefore, when applied together, we obtain protocols that are *simultaneously* round optimal as well as communication optimal. Further, the *simplicity* of our protocol ensures the implementability of highly scalable perfectly secret message transmission. Surprisingly, as proved in Section 3.8, it turns out that most of our protocols can be adapted to work for the mobile adversary case too. In a nutshell, we address the PSMT problem in such a way that all the four questions, namely, characterization, feasibility, communication and round optimality, are answered in one-shot.

- **Motivation and contribution #2**: As discussed earlier, in directed graph setting, recently in [44] it is proved that *all* of the above mentioned limitations can be circumvented while tolerating *passive* adversaries. However, the problem of what is a necessary and sufficient condition for the existence of `PSMT` protocols in arbitrary digraphs tolerating a Byzantine adversary that corrupts up to any t nodes appears to be a notoriously hard open problem. Many of the aforementioned limitations may actually turn out to be inherent and insurmountable. Consequently, we focus on the adversary which can corrupt up to t_f nodes in a `fail-stop` fashion, in addition to `passively` corrupting up to any t_p nodes. And, We give (in Section‘ 4.5) an elegant necessary and sufficient condition for the possibility of `PSMT` from S to R in the given digraph. However, unlike the passive-only case, it is evident that the generality is perhaps achieved at the cost of retaining the other two limitations – namely, our characterization is probably hard to test as well as, in the worst-case, no efficient protocols may exist. Subsequently, we address the same problem in a less general, yet popular, model of abstracting the graph as a collection of directed wires (like in [14,21,29,39,40,48]). In the wires-based model, we present (in **Theorem 4.3**) a necessary and sufficient condition for the existence of a `PSMT` protocol from S to R. Not surprisingly, the protocols we design are efficient too. Finally, sandwiching the above two results, we ask if there are any special classes of graphs wherein the protocols are efficient, however, they fail to meet the necessary conditions of the wires-based model. In Section 4.6, we show that this is indeed the case – in other words, the loss in generality required to achieve efficient protocols is strictly less than that of the prevalent wires-based model. Interestingly, we use the above "sandwiched" result to prove (in **Theorem 4.11**) that for `PSMT` to be possible between all-pairs in the network, the necessary and sufficient condition is *simultaneously* (a) general, (b) admits efficient protocols and (c) is efficiently testable too! Summarizing, our results show that while the trade-off among the three limitations probably exists for `PSMT` among a chosen pair of nodes, no such concern remains in the case of all-pair `PSMT`.

- **Motivation and contribution #3**: We draw our motivation from the following fundamental question. Given a network of interconnected nodes with the sender S and the receiver R, under what conditions is `SMT` (im)possible from S to R tolerating a given adversary if each intermediate node is a mere router (cannot generate random coins and cannot perform any computations)?

  As discussed earlier, if the intermediate nodes can generate random-coins or perform computations, it is known that the directed wires-based abstraction is inadequate to capture general digraphs [43, 44]. On the other hand, in the case where intermediate nodes are only allowed to store-and-forward information, folklore suggests that directed wires based abstraction might be adequate. In this thesis, we show that the folklore is false. To show the same, we introduce a new model namely the routing model. The routing model considered in this thesis is a subtle variant of the directed wires model considered in [14, 39–41]. In all these papers, every pair of channels must be vertex-disjoint. That is, an overlap is not allowed between any two channels. In our routing model, an overlap is also allowed. The significance of allowing such an overlap is visible from the following implication (as elaborately illustrated in chapters 3 and 4) – there are networks (say G is one of them) under adversarial influence such that SMT from S to R is efficiently possible in G but existing results in the literature either: (1) prove the impossibility of SMT by modelling G without the 'overlap' [14, 39–41] or (2) give complicated protocols that work on arbitrary directed graphs and hence potentially inefficient in G too [43, 49]. This clearly shows that the current models are either too specific or too general with respect to capturing the natural setting, namely the scenario where all the intermediate nodes are mere routers. We show that the routing model that allows overlap between channels adequately captures this setting. Specifically, unlike in undirected graphs, fixating on the disjoint directed wires between S and R is not without loss of generality – the directed edges connecting these disjoint directed wires are often-times necessary too!

In particular, we contribute to the literature by answering the following questions. If each intermediate node is a mere router then, what are the necessary and sufficient conditions for the existence of a:

1. PSMT protocol tolerating up to t_p passive faults?
2. PSMT protocol tolerating mixed adversary - up to t_f fail-stop faults in addition to t_p passive faults.
3. PSMT protocol tolerating up to t_o omission faults.
4. USMT protocol tolerating up to t_b Byzantine faults?

For each of the above problems, we design communication efficient protocols if the given network admits corresponding SMT.

1.6 Thesis Organization

This thesis is organized in seven chapters as follows.

1. In Chapter 1 we briefly introduced SMT problem and motivations for studying SMT problem in distributed environment.

2. In Chapter 2 we formally introduce various definitions, and network models used in this thesis.

3. We dedicate Chapter 3 to passive faults. In passive threshold adversary setting, we study necessary and sufficient condition(s) for the existence of PSMT protocols in undirected graph setting as well as directed graph setting. Moreover, if the given network admits PSMT then correspondingly we design communication efficient protocols. Furthermore, we design a protocol which is round as well as communication optimum.

4. We dedicate Chapter 4 to mixed faults (passive+fail-stop). In threshold mixed adversary setting, in each of the network model, we study the trade-off between the network connectivity requirements and the existence of PSMT protocols. Moreover, we design efficient protocols in wires model setting as well as routing model setting. In arbitrary directed graph setting, we characterize networks in which PSMT is (im)possible. However, the protocol we present (if the network admits PSMT) is not communication efficient.

5. In Chapter 5, we study omission faults. In particular, we study network connectivity requirement(s) for the existence of PSMT protocols tolerating t_o omission faults under different kinds of network models. Moreover, we design efficient protocols in wires model as well as routing model, if the given network admits corresponding PSMT.

6. In Chapters 6 we study Byzantine faults. In particular, we study the necessary and sufficient condition(s) for the existence of SMT protocols tolerating t_b Byzantine faults. Moreover, in routing model setting, we design communication efficient USMT protocols if the given network admits USMT.

7. In Chapter 7, we briefly discuss the summary of this thesis and future work to carry on.

Chapter 2

Definitions and Preliminaries

2.1 Introduction

In this chapter, we present formal definitions used in this thesis. Specifically, we stress on the following definitions:

1. Network Model
2. Types of Node Corruption
3. The Adversary
4. Security Notions
5. Definitions of Graph and Paths

Also, we present the Shamir's secret sharing scheme which we extensively use in designing various `SMT` protocols.

2.2 Network Model

The underlying communication network of interconnected nodes can be modelled as an undirected graph or directed graph. The communication network can be *Synchronous* [1, 12, 14, 15, 55] or *Asynchronous* [32, 36, 37]. In a synchronous network, the maximum time delay for a message sent by one node to reach the other is bounded by a *fixed* finite number. In other words, a global clock access is available to every node in the network and a message sent at this *tick* of the clock will be delivered by the next *tick* of the clock. Unlike in synchronous network, there is no such global clock available to refer to in an asynchronous network. And, an arbitrary time delay is possible in delivering a message. We can also assume that either every node in the network knows the complete topology of the network or knows only its neighbours [56]. Below, we elaborate more on undirected and directed graph modellings.

1. **Undirected Graph** [1,15,52,57] - In this modelling, the communication network is abstracted as an undirected graph G with vertex set V and edge set E. Each edge $(S, R) \in E$ represents a private, authentic and reliable channel between S and R. Using the channel (S, R), nodes S and R can communicate each other. This model is best suitable in real life scenarios where all the nodes have similar communication capabilities. In an undirected graph, a path from S to R is also a path from R to S and vice-versa (see Figure 2.1). Therefore, `SMT` is possible from S to R if and only if corresponding `SMT` is possible from R to S. And, this is true for all variants of `SMT` problem.

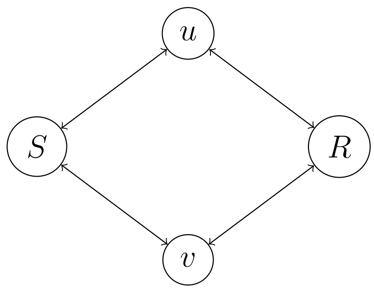

Figure 2.1: Graph G with two undirected disjoint wires

2. **Directed Graph** [14, 47, 48, 58]- In undirected graphs, S can communicate with R if and only if R can communicate with S as every path from S to R is also a path from R to S and vice-versa. However, in general, S may be able to communicate with R but the other way around may not be possible. The reason could be, the channel from S to R may be economical whereas the channel from R to S may be too expensive. Or, as mentioned in [40], the base station may be able to communicate with the far off hand held device whereas the other way around may not be possible. Therefore, when different nodes have different communication capabilities, undirected graph network modelling do not capture all real life scenarios. In such scenarios, the best fit is modelling the communication network as a directed graph. Each edge (A, B) represents a private, authentic and reliable channel from A to B. We consider two special cases of directed graph setting in this thesis along with the arbitrary directed graph setting.

(a) **Wires Model** [14, 29, 39, 40, 48]: In wires model, the given network is abstracted as a collection of *disjoint* wires, where each wire is either a path from S to R or vice versa. By disjoint we mean, no two wires share a common vertex except S and R (see Figure 2.2). A wire/path from S to R is known as a *forward channel* and a wire from R to S is known as a *feedback channel* [14]. A set of disjoint wires from S to R is known as *top band* and a set of disjoint wires from R to S (which are disjoint with each wire in top band) is known as *bottom band* [40].

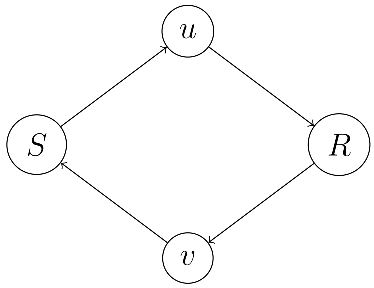

Figure 2.2: Graph G with two disjoint wires

(b) **Routing Model**: In routing model, the given network is abstracted as a collection of paths from S to R as well as paths from R to S. The main difference comparing with the wires model is that, the paths from S to R need not be disjoint with the paths from R to S and vice-versa. That is, a path from S to R can share a vertex with a path from R to S and vice-versa – overlap is allowed. For example, consider the graph G given in Figure 2.3, G has only one forward channel, namely, $W_1 : \langle S, u, v, R \rangle$, which exhausts all the nodes in the network. Therefore, there is no wire from R to S which is disjoint with the wire W_1. However, there are two disjoint wires from R to S, namely, $W_2 : \langle R, u, S \rangle$ and $W_3 : \langle R, v, S \rangle$ and a wire from S to R, namely, W_1. Unlike in wires model, for designing `SMT` protocols we use each and every wire; in this case, we use three wires W_1, W_2 and W_3.

(c) **Arbitrary Directed Graph Model** [31, 47, 50, 58]: In arbitrary directed graph model, we fully consider the given network. That is, we consider each and every edge of the given graph. For example consider the graph G given in Figure 2.4, in wires/routing model we do not consider the weak path $\langle S, v, R \rangle$ for designing protocols, whereas in arbitrary directed we consider such weak paths as well.

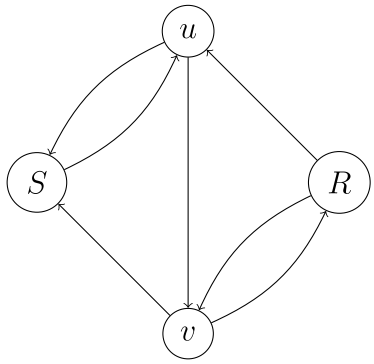

Figure 2.3: Graph G with only one forward channel.

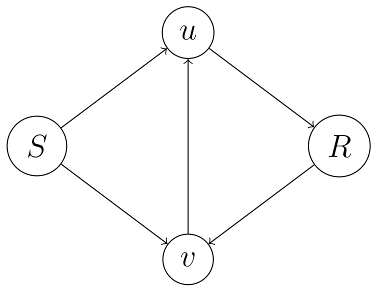

Figure 2.4: Graph G with only one wire

As each edge acts as a secure channel between the corresponding nodes, we assume that the ordered pair (S, R) is not an edge in the network. And, we aim to *simulate* the edge (S, R).

2.3 Types of Node Corruption

Different types of node corruptions studied in the literature are passive corruption, fail-stop corruption, omission corruption and Byzantine corruption. The formal definitions are given below.

Definition 2.1 (Passive Corruption [27,59])**.** *A node P is said to be passively corrupted if the adversary has full access to the information and the internal state of P. But P honestly follows the protocol execution.*

Definition 2.2 (Fail-Stop Corruption [27, 59])**.** *A node P is said to be fail-stop corrupted if the adversary can crash P at its will at any time during the execution of the protocol. But as long as P is alive, P will honestly follow the protocol and the adversary will have no access to any information or internal state of P. Once P is crashed, then it will remain inactive for the rest of the protocol execution.*

Definition 2.3 (Omission Corruption [59, 60]). *We say that a node P is omission corrupted, if the adversary can crash P at its will at any time during the execution of the protocol. But as long as P is alive, it will follow the instructions of the protocol honestly. The adversary can eavesdrop the internal data of P but cannot make P to deviate from the proper execution of the protocol. A blocked node P can again become alive at some later stage of the protocol and start following the protocol honestly.*

Definition 2.4 ([27, 59]). *Byzantine Corruption - A node P is said to be Byzantine corrupted if the adversary fully controls the actions of P. The adversary will have full access to the computation and communication of P and can force P to deviate from the protocol and behave arbitrarily.*

2.4 The Adversary

In the literature, faults/distrust in a distributed system is usually captured via a (fictitious) *adversary* that can *corrupt* some of the nodes in the network.

1. The adversary can be *threshold* [1, 14] or *non-threshold* [43, 47]. The adversary is threshold, if the number of nodes the adversary can corrupt is no more than a fixed threshold number. Unlike in threshold adversary, non-threshold adversary is characterized via an adversary structure, which is a collection of subsets of nodes in the network, wherein the adversary may choose to corrupt the nodes from any one of the subsets of the collection. Notice that, the threshold adversary is a special case of the non-threshold adversary. The reason is, for a given a threshold number t, the corresponding adversary structure is nothing but the set of all possible subsets of cardinality less than or equal to t.

2. We can also model the adversary as either *static* [1, 14] or *mobile* [22, 24, 61]. In static adversary model, the adversary is not allowed to corrupt different set of nodes in different rounds (see Definition 2.15). That is, the adversary corrupts the same set of nodes throughout the protocol execution. On contrary, the mobile adversary can corrupt different sets of nodes in different rounds – However, in any given round the adversary can corrupt only one subset of nodes from the adversary structure. The static adversary model is more suitable where protocols run for very short time. Whereas, for protocols that last long, modelling the adversary as mobile is more suitable. This is because, when protocol runs for very long time, the errors/faults occurred in earlier rounds can be identified and cured;

meanwhile the adversary may corrupt some other nodes. The notion of mobile adversary exactly captures this scenario.

3. Furthermore, we can consider mixed adversary as well [28–30, 60, 62]. The mixed adversary is the one who can corrupt some of the nodes in a mixed way during the course of the protocol. In detail, mixed adversary can simultaneously corrupt up to any $t_p(\geq 0)$ nodes in passive fashion, up to any $t_f(\geq 0)$ nodes in fail-stop fashion, up to any $t_o(\geq 0)$ nodes in omission fashion and up to any $t_b(\geq 0)$ nodes in Byzantine fashion. This adversarial model is more realistic than assuming that the adversary corrupts each and every node in the same fashion. This is due to the fact that, certain nodes in the network may be strongly protected and some other nodes may be weakly protected. In that case, the adversary may be able to corrupt the strongly protected nodes only in fail-stop/passive fashion but not in Byzantine fashion. Whereas, it may be possible for the adversary to corrupt the weakly protected nodes in Byzantine fashion.

2.5 Security Notions

In this section, we formally define various kinds of security notions. As discussed earlier, loosely speaking, a message transmission protocol is:

1. perfectly reliable if the message received by the receiver is identical to the sender's message without any error.

2. unconditionally reliable if the receiver receives the message with arbitrary high probability (only arbitrary low/negligible error is allowed in reliability).

3. perfectly secure if it is perfectly reliable and additionally satisfies the condition that no information is leaked to the adversary (information theoretic security) about the message.

4. unconditionally secure if it is unconditionally reliable and no information is leaked to the adversary.

Now, we move to the formal definitions. Let the sender and the receiver be denoted by S and R respectively. And, M^S be the message of the sender and M^R be the message received by the receiver at the end of an SMT protocol. Following [12, 14], the adversary's view is denoted by $adv(M, r)$ when the message transmitted by S is M and r is the sequence of coin flips used by the adversary during the course of the protocol.

Definition 2.5 (Reliable Message Transmission [12, 14]). *1. A message transmission protocol is δ-reliable, for $\delta < 1/2$, if the message M^R received by the receiver is identical to the sender's message M^S with probability at least $1 - \delta$. The probability is over the coin flips of all nodes and the choices of M^S.*

2. A message transmission protocol is perfectly reliable if it is 0-reliable.

Definition 2.6 (Secure Message Transmission [12, 14]). *1. A message transmission protocol is ϵ-secure, if for every two messages M_0 and M_1 and for every r, $\sum_c |Pr[adv(M_0, r) = c] - Pr[adv(M_1, r) = c]| \leq 2\epsilon$. The probabilities are taken over the coin flips of the honest parties, and the sum is over all possible values of the adversary's view.*

2. A message transmission protocol is perfectly secure if it is 0-secure.

Definition 2.7 ([12, 14]). *A message transmission protocol is (ϵ, δ)-secure if it is ϵ-secure and δ-reliable.*

Definition 2.8. *In literature, depending on the values of ϵ and δ, different names are given for (ϵ, δ)-secure protocols as follows.*

1. $(0, 0)$-SMT *protocols are known as Perfectly Secure Message Transmission* - PSMT *- protocols [1, 12].*

2. $(0, \delta)$-SMT *protocols are known as Almost Perfectly Secure Message Transmission protocols [12] as well as Unconditionally Secure Message Transmission* - USMT *- protocols [31, 41, 49, 59].*

3. $(1, \delta)$-SMT *protocols are also known as Almost Perfectly Reliable as well as Unconditionally Reliable* - URMT *- protocols [31, 41, 49, 59].*

4. $(1, 0)$-SMT *protocols are known as Perfect Reliable Message Transmission* - PRMT *- protocols [1, 12].*

Definition 2.9 (Notations). *1. We use $[l, u]$ to denote the set $\{i \in \mathbb{Z} \mid l \leq i \leq u\}$.*

2. We use $[N]$ to denote the set $\{x : 1 \leq x \leq N, x \in \mathbb{N}\}$.

2.6 Graph and Paths

Definition 2.10 (Underlying Undirected Graph). *The underlying undirected graph of a directed graph $G(V, E)$ is denoted by $G_u(V, E_u)$, where $E_u = \{(u, v) \mid (u, v) \in E \text{ or } (v, u) \in E\}$.*

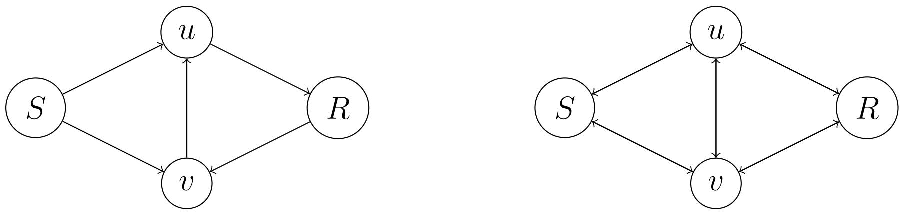

Figure 2.5: Directed Graph G and the underlying undirected graph G_u.

Definition 2.11 (Path). *In a directed graph $G(V, E)$, we say that a sequence of nodes $p : \langle v_0(= u), v_1, v_2, \ldots, v_k, v_{k+1}(= v)\rangle$ is a path from u to v, if and only if $(v_j, v_{j+1}) \in E$, $\forall j \in [0, k]$.*

$$S \longrightarrow v \longrightarrow u \longrightarrow R$$

Figure 2.6: Example of a path in the graph G given in Figure 2.5

Definition 2.12 (Weak Path). *In a directed graph $G(V, E)$, we say that a sequence of nodes $p : \langle v_0(= u), v_1, v_2, \ldots, v_k, v_{k+1}(= v)\rangle$ is a weak path from u to v if and only if $\forall j \in [0, k]$, either $(v_j, v_{j+1}) \in E$ or $(v_{j+1}, v_j) \in E$. $V(p)$ denotes the set of vertices of p. Note that by definition, every path is a weak path.*

$$S \longrightarrow u \longleftarrow v \longleftarrow R$$

Figure 2.7: Example of a weak path in the graph G given in Figure 2.5

Definition 2.13 (Corresponding Path of a Weak Path). *Let $G_u(V, E_u)$ be the underlying undirected graph of the directed graph $G(V, E)$ and $p : \langle v_0(= u), v_1, v_2, \ldots, v_k, v_{k+1}(= v)\rangle$ be a weak path in G. We say that the path $p' : \langle v_0(= u), v_1, v_2, \ldots, v_k, v_{k+1}(= v)\rangle$ in G_u is the corresponding path of the weak path p.*

$$S \longleftrightarrow u \longleftrightarrow v \longleftrightarrow R$$

Figure 2.8: Example of the corresponding path of the weak given in the Figure 2.7

Definition 2.14 (Induced Subgraph)**.** *Let $G(V, E)$ be a graph. For any subset U of V, we use $G[U]$ to denote the induced subgraph of G induced on vertices of U.*

Definition 2.15 (Round [63])**.** *In any synchronous network, every node has access to a global clock and the communication proceeds in rounds (time-steps) according to this global clock. From the communication point of view, it takes exactly one round (one time-step) to transmit field elements using any link (edge) of the network. More formally, in any* ***round****, a player can execute commands in the following order:*

1. *Perform local computations.*

2. *Send messages to its out-neighbour(s).*

3. *Receive all the messages sent earlier in this round by its in-neighbour(s).*

4. *Perform local computations on the received messages.*

Definition 2.16 (Round Complexity)**.** *The round complexity of any synchronous protocol is defined as the total number of rounds required to execute the protocol before its termination.*

Definition 2.17 (Communication complexity)**.** *The communication complexity of any protocol is defined as the total number of field elements communicated through all the edges in the network during the execution of the protocol.*

2.7 Shamir's Secret Sharing Scheme

In this thesis, we extensively use Shamir's secret sharing scheme for designing various `SMT` protocols. This scheme works as follows. Let $p(x)$ be a t-degree polynomial over a finite field such that the constant term $p(0)$ is the *secret*. As per this scheme, each participant gets a point on $p(x)$ as his/her share of the *secret*. And, to reconstruct the *secret* at least $t+1$ shares are required, whereas t or fewer shares together reveal nothing about the constant term which is the *secret*. This directly follows from the well-known fact that t or fewer points on a t-degree polynomial reveal nothing about its constant term whereas, with $t+1$ or more points, the corresponding t-degree polynomial can be reconstructed uniquely [64].

2.7.1 Adapting Shamir's Secret Sharing Scheme to SMT Protocols

This scheme is well adapted to design `SMT` protocols. The following is one such example. Let us consider the t-threshold passive static adversary $\mathcal{A}$ over the network $\mathcal{N}$ having $t+1$ vertex-disjoint paths from the sender S to the receiver R. One can design a `SMT` protocol tolerating $\mathcal{A}$ over $\mathcal{N}$ using Shamir's secret sharing scheme as follows. The sender chooses a random t-degree polynomial $p(x)$ over the field $\mathbb{F}$ such that the constant term $p(0)$ is the message M. And, for each i, S sends point $p(i)$ to R along the i^{th} disjoint path. Upon receiving $(t+1)$ points on $p(x)$, R reconstructs $p(x)$ and computes $p(0)$ which is the message M. *Security* is guaranteed as t or fewer points on a t-degree polynomial reveal nothing about its constant term. And, *reliability* is guaranteed as $t+1$ points are enough to reconstruct the t-degree polynomial.

2.8 Assumptions

Unless explicitly mentioned, all the results given/stated in this thesis are based on the following assumptions. The message space is large enough finite field $\langle \mathbb{F}, +, \star \rangle$ and all the calculations are performed in this field $\mathbb{F}$ only. By "a number r is chosen randomly" we mean that "r is chosen uniformly at random from the field $\mathbb{F}$". The sender and receiver are denoted by S and R respectively. We also assume that S and R do not share any key a priori. Security is vacuously achieved if either of the sender or receiver is corrupted by the adversary. Hence, we assume that neither the sender nor the receiver can be corrupted by the adversary. Also, we assume that the network is synchronous and the adversary is threshold. Furthermore, the protocol code/specifications, as well as the complete topology of the network is known to each player/node of the network including the adversary.

Chapter 3

SMT Tolerating Passive Faults

3.1 Introduction

In this chapter, we study SMT problem tolerating threshold passive adversary in each of the following network models, namely, undirected graph model, wires model, routing model and arbitrary directed graph model. In particular, we study the trade-off between the network connectivity and the existence of PSMT protocols in each of the aforementioned network models and concentrate on designing efficient protocols. Moreover, we study round optimality issues, like if PSMT is possible then (1) is it possible to design an r-round PSMT protocol, for any given $r > 0$ (2) what is the optimal number of rounds required for PSMT (3) how to design round optimal protocol(s). Specifically, we prove a necessary and sufficient condition on the synchronous network for the existence of r-round PSMT protocols, for any given $r > 0$; further, we show that round-optimality is achieved without trading-off the communication complexity; that is, our protocols have an overall communication complexity of $\mathcal{O}(|V|)$ elements of a finite field to perfectly transmit one field element.

Interestingly, optimality (of protocols) also implies: (a) when the shortest path between S and R has $\Omega(|V|)$ nodes, *perfect secrecy is achieved for "free"* because any (insecure routing) protocol also take $\mathcal{O}(|V|)$ rounds and send $\mathcal{O}(|V|)$ messages (one message along each edge in the shortest path) for transmission and (b) For a protocol to exist, it is well-known that $(t+1)$ vertex-disjoint paths from S to R are necessary; a consequent folklore is that the length of the $(t+1)^{th}$ ranked (disjoint shortest) path would dictate the round complexity of protocols; we show that the folklore is false; round-optimal protocols can be substantially faster than the aforementioned length.

Apart from optimality/scalability, a couple of interesting implications of our results are: (a) *adversarial mobility does not affect its tolerability:* PSMT tolerating a static t-adversary is possible *if and only if* PSMT tolerating mobile t-adversary is possible; and (b) *mobility does not affect the round optimality:* the fastest PSMT protocol tolerating a static t-adversary is *not faster* than the one tolerating a mobile t-adversary. Throughout this chapter, by a "faulty node", we mean that the node is "passively corrupted by the adversary" and by "secure" we mean "perfectly secret". For brevity, by "PSMT is possible" we mean "PSMT tolerating t-threshold passive adversary is possible". Moreover, unless we explicitly mention, by "the adversary" we mean " the static adversary".

By definition, passive faults do not disrupt the communication. As a result, in the presence of passive faults, the receiver always receives the information without any error sent by the sender along *any* channel connecting the sender and receiver. Though the reliability is guaranteed, the sender can not send the message/information in plain form as the adversary can eavesdrop the message/information. Thus, achieving security is not trivial. We start with PSMT in undirected network setting.

3.2 PSMT in undirected graph model

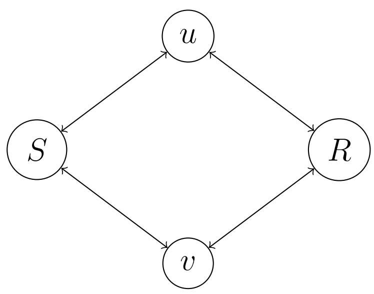

Figure 3.1: Graph G_1 with two undirected paths

Dolev et al. first introduced the SMT problem and studied threshold adversary setting in undirected graph model [1]. Consider the graph G_1 given in Figure 3.1. G_1 has two disjoint paths, namely $\langle S, u, R\rangle$ and $\langle S, v, R\rangle$. Notice that, if adversary corrupts both u and v then the adversary can listen to all the communication which takes place between S and R. Thus, the adversary can reconstruct the message as long as the receiver can reconstruct the message, as the protocol code is known to everyone including the adversary. Therefore, PSMT is impossible in G_1 if both the nodes u and v are corrupted. Whereas, PSMT is possible in G_1 tolerating one passive fault. And the corresponding Protocol Π_{G_1} works as follows.

The Protocol Π_{G_1}:

1. The sender S picks a random one-degree polynomial $p(x)$ such that the constant term $p(0)$ is the message m that S intends to send to R.

2. S sends $p(1)$ to u and $p(2)$ to v.

3. Upon receiving them, u and v forwards $p(1)$ and $p(2)$ to R respectively.

4. Once R receives both $p(1)$ and $p(2)$, R reconstructs $p(x)$ thus gets the message m.

The existing result for `PSMT` tolerating t-threshold static adversary is the following.

Theorem 3.1 (Dolev *et al.* [1]). *In an undirected graph G,* `PSMT` *from S to R tolerating up to t passive faults is possible if and only if there exist at least $t+1$ vertex-disjoint paths between S and R.*

Proof. **Necessity**: Suppose if there are at most t vertex-disjoint paths from S to R, then, by Menger's theorem [38] there exists a vertex-cut of size t between S and R. Therefore, by corrupting every node in the vertex-cut, the adversary corrupts each of these t paths and gets the information identical to what the receiver would receive from the sender.

Sufficiency: The sufficiency is achieved using Shamir's secret sharing scheme. The sender S chooses a random t-degree polynomial $p(x)$ such that $p(0)$ is the message m. The sender S sends $p(i)$ to the receiver R along the i^{th} disjoint path. We know that $t+1$ points on $p(x)$ are enough to reconstruct the polynomial $p(x)$ and t or fewer points reveal nothing about m [64]. □

3.3 PSMT in wires model

Recall that, in wires model we abstract the network as a collection of paths from S to R and R to S. A path from S to R (resp. R to S) is known as `forward` (resp. `feedback`) channel [14]. A set of disjoint wires from S to R is known as *top band* and a set of disjoint wires from R to S (which are disjoint with each wire in top band) is known as *bottom band* [39, 40, 42].

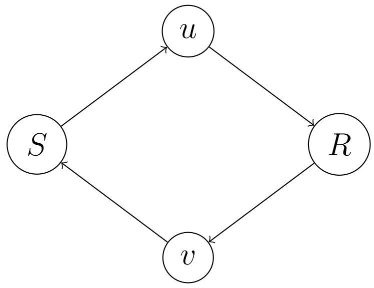

Figure 3.2: Graph G_2 with two disjoint wires

Consider the graph G_2 given in Figure 3.2. If PSMT is not possible in G_1 then clearly PSMT is not possible in G_2 too as G_2 is a subgraph of G_1. Also, we have seen that PSMT is not possible in G_1 if both u and v are corrupted by the adversary. Therefore, if both u and v are corrupted then PSMT is not possible in G_2 as well. Thus, we start with the assumption that the adversary can corrupt at most one of the two nodes u and v. We show that PSMT is possible in G_2 tolerating one passive fault. Notice that, unlike in G_1, the sequence $\langle S, v, R\rangle$ is not a path in G_2. Thus, the protocol Π_{G_1} do not work. We design a new protocol Π_{G_2}, which achieves PSMT in G_2 tolerating one passive fault. The protocol Π_{G_2} is presented below.

The Protocol Π_{G_2}:

1. The sender S picks a random one-degree polynomial $p(x)$ such that the constant term $p(0)$ is the message m that S intends to send to R.
2. S sends $p(1)$ to u and u forward the same to R.
3. R picks a random number r and sends it to S along the path $\langle R, v, S\rangle$.
4. S upon receiving r, computes $p(2) + r$ and sends it to R along the path $\langle S, u, R\rangle$.
5. Once R receives $p(2) + r$, gets the point $p(2)$ by subtracting r from $p(2) + r$.
6. Finally, R gets the message m by reconstructing $p(x)$ using $p(1)$ and $p(2)$.

Notice that, on corrupting the node v, the adversary gets the random number r and clearly it reveals nothing about the message m. In case of corrupting the node u, the adversary gets the point $p(1)$ and a field element $p(2) + r$. However, $p(2) + r$ reveals nothing about $p(2)$ (see Lemma 3.1) as long as r is unknown to the adversary (as in this case). This implies, the adversary gets at most one point in any case, which guarantees the security of the protocol.

Lemma 3.1. *In a field* $\langle \mathbb{F}, +, * \rangle$*, for given* $x, z \in \mathbb{F}$*,* $\exists$ *unique* $y \in \mathbb{F}$ *such that* $x+y = z$*.*

Proof. Assume that $x + y = z$ and $x + u = z$, for $u, y \in \mathbb{F}$. This implies $x + u = x + y$. As $\langle \mathbb{F}, + \rangle$ is a group, cancellation laws hold in $\mathbb{F}$. Therefore, we get $u = y$. □

Now we move to the characterization in wires model tolerating threshold passive adversary. As passive faults do not disrupt the communication, a single path from S to R is necessary and sufficient for reliable communication from S to R. However, to keep the message *secret*, we need at least $t + 1$ disjoint paths. On combining these two basic observations, we get the following simple and elegant characterization in the wires model.

Theorem 3.2 ([42]). `PSMT` *from* S *to* R *tolerating up to* t *passive faults is possible if and only if there exist* $t + 1$ *disjoint wires between* S *and* R *such that at least one of them is from* S *to* R*.*

Proof. **Necessity**: From **Theorem 3.1** we know that, in an undirected graph $t + 1$ vertex-disjoint paths between S and R are necessary and sufficient for `PSMT`. Since directed graph is a subgraph of its corresponding undirected graph, in wires model too $t + 1$ disjoint wires (each one either from S to R or vice-versa) are necessary for `PSMT`.

Sufficiency: We extend the idea used in designing the protocol Π_{G_2} to design an efficient protocol Π_{WP}. The protocol Π_{WP} achieves `PSMT` if the given graph meets the sufficiency condition of the theorem 3.2. Informally, the protocol Π_{WP} works as follows. S starts with a random degree-t polynomial $p(x)$ such that $p(0)$ is the message m. If the i^{th} wire W_i is a wire from S to R then S sends point $p(i)$ along W_i to R. If W_i is a wire from R to S then R sends a random number r_i to S along W to S. S upon receiving r_i, masks point $p(i)$ with r_i as $p(i) + r_i$ and sends it to R via some forward channel. Upon receiving $p(i)+r_i$, R subtracts r_i from $p(i)+r_i$ and gets the point $p(i)$. Formally, the protocol Π_{WP} achieves `PSMT` as follows, assuming that we have $t+1$ disjoint wires. Let us assume that T wires are from S to R, namely W_i for each $i \in [1, T]$ and B wires are from R to S namely, W_i for each $i \in [T+1, T+B]$ such that $T+B = t+1$ and $T \geq 1$.

The Protocol Π_{WP}:

1. The sender S picks a random t-degree polynomial $p(x)$ such that the constant term $p(0)$ is the message m.
2. The sender S sends $p(i)$ along the wire W_i to the receiver R, for each $i \in [1, T]$.
3. The receiver R chooses a random number r_i, for each $i \in [T+1, T+B]$ and sends r_i along the wire W_i to the sender S.
4. Once the sender S gets the random number r_i for each $i \in [T+1, T+B]$, S computes $p(i) + r_i$ and sends the same to R along a wire W_j, for some $j \in [1, T]$.
5. The receiver R upon receiving $p(i) + r_i$ gets the point $p(i)$ by subtracting r_i from $p(i) + r_i$, for each $i \in [T+1, T+B]$.
6. Finally, R gets the message m by reconstructing $p(x)$ using $t+1$ points on $p(x)$, namely, $p(i)$ for each $i \in [1, T+B]$.

It is easy to see that the proposed protocol Π_{WP} is secure because $p(i) + r_i$ reveals nothing about $p(i)$ (see Lemma 3.1) unless the adversary has some information about r_i. However, to get r_i the adversary must corrupt the wire W_i. Therefore, on corrupting t wires the adversary gets at most t points on $p(x)$. Therefore, these points reveal nothing about the message m. □

3.4 PSMT in routing model

In this section, we study the trade-off between the network connectivity and the existence of PSMT protocols in the presence of t passive faults in routing model. Moreover, we design an efficient protocol which achieves PSMT in the given graph if it meets the sufficiency conditions of the corresponding theorem. Also, we show that there are networks (graph G_3 given in Figure. 3.3 is one such) which admits PSMT if we abstract the network as in routing model, whereas if we abstract the given network as in wires model then no PSMT protocol exists.

In earlier sections, we have seen that if *all* the channels are either forward or both forward and feedback then $t+1$ disjoint channels are necessary and sufficient as shown by Dolev et al. in [1]. In the wires model, Ravi Kishore et al. shown that if there are

k (≥ 1) disjoint forward channels then $t-k+1$ disjoint feedback channels (which are disjoint with the k forward channels) are necessary and sufficient [42].

Recall that, in routing model, we abstract the network as a collection of paths from S to R and R to S as in the wires model, following along the lines of Wang et al. [14]. The only difference is, an overlap is allowed between channels. We call our model *routing* model and the reason is as follows. If each intermediate node is a mere router – can store-and-forward information but can neither do computations nor generate random coins – then weak paths (which are not paths) are of no use. This is easy to see from the fact that every piece of information originates at either sender's end or receiver's end.

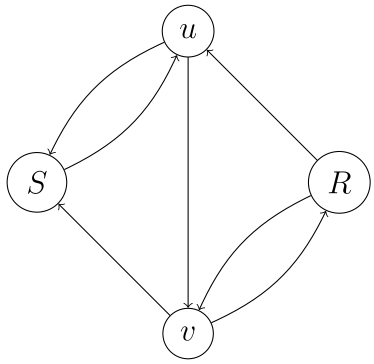

Figure 3.3: Graph G_3 with only one forward channel.

Consider the graph G_3 given in Figure 3.3. As discussed in previous section(s), PSMT is not possible in G_3 if both u and v are corrupted by the adversary. So we assume that the adversary can corrupt at most one node. We observe that there is only one forward channel from S to R, namely, $p_1 : \langle S, u, v, R \rangle$. Furthermore, this forward channel exhausts all the nodes of G_3. Therefore, G_3 fails to meet the conditions of the **Theorem 3.2**, tolerating one passive fault, as there in no extra required feedback channel.

As intermediate nodes are mere routers, at first glance we may believe that PSMT is not possible in G_3 if either u or v is corrupted. However, notice that there are two disjoint paths from R to S, namely $p_2 : \langle R, u, S \rangle$ and $p_3 : \langle R, v, S \rangle$. Using these two disjoint paths p_2 and p_3, S and R together establish a shared secret key k between them. Then, S blinds the message m using the key k as $m+k$ and sends it to R using the path p_1. Upon receiving $m+k$, R subtracts k from $m+k$ and gets the message m. More formally, the following protocol Π_{G_3} achieves PSMT in G_3 tolerating one fault. The correctness of the protocol is guaranteed by the **Theorem 3.3**.

Protocol Π_{G_3}:

1. The receiver R picks two random numbers, say k_1 and k_2. And, sends both to the sender S, k_1 along the path $\langle R, u, S\rangle$ and k_2 along the path $\langle R, v, S\rangle$.

2. The sender S upon receiving both k_1 and k_2, masks the message as $m+k_1+k_2$ and sends it to the receiver R along the path $\langle S, u, v, R\rangle$.

3. Once the receiver R receives $m + k_1 + k_2$, gets the message m by subtracting $k_1 + k_2$ from $m + k_1 + k_2$.

we present a fundamental observation, which explains why the existing results fail to characterize networks like G_3.

Core Idea: By definition any (ϵ, δ)-SMT protocol must be δ-reliable. In other words, reliability is necessary for any SMT protocol. Therefore, the existing results used the following approach. Given a reliable forward channel (α), they are interested in the minimum number of *extra* feedback channels (disjoint with α) required for the existence of a SMT protocol. However, observe that if there is a reliable forward channel (α) and there is a secure feedback channel (β) - β *need not* be disjoint with α and *need not* be two-way connected - *then also* we can simulate a secure forward channel. This is because, first the sender receives the *key* securely from the receiver through the secure channel β and then signs/blinds the message with the *key* before sending it to the receiver over a reliable channel α. The existing results fail to capture the same - the reliable forward channel α *need not* be disjoint with the secure feedback channel β.

In each case, as discussed above, we show that two *independent* conditions together, one for *reliability* and one for *security*, are necessary and sufficient for the existence of corresponding SMT protocols.

Theorem 3.3. *In a directed graph G,* PSMT *is possible from S to R, tolerating up to t passive faults if and only if G satisfies the following two conditions:*

1. *There exists at least one forward channel - for reliability.*

2. *There exist at least $t + 1$ disjoint channels - for security.*

Proof. **Necessity**: The necessity of the first condition is trivial to understand as if there is no `forwards` channel, then no communication is possible from S to R. Therefore, G must have at least one `forward` channel, which we denote by α.

The necessity of the second condition follows from the fact that, if each channel is two-way connected (i.e., both forward and feedback) then also $t + 1$ channels are necessary for the existence of a PSMT protocol as we have seen in **Theorem 3.1**.

Sufficiency: We present an elegant PSMT protocol Π_{RP} whenever G satisfies the conditions of the **Theorem 3.3**. Let us assume that there exists a `forward` channel α. Also, assume that there exist $T + B$ $(\geq t + 1)$ disjoint channels in total, of which T (≥ 0) of them $\beta_1, \beta_2, \ldots \beta_T$ are `forward` channels and B $(\geq t + 1 - T)$ of them $\beta_{T+1}, \beta_{T+2}, \ldots \beta_{T+B}$ are `feedback` channels.

The protocol Π_{RP}

1. The sender S generates T random numbers, say r_i, $\forall i \in [T]$ and sends r_i to R along the forward channel β_i.
2. The receiver R generates B random numbers, say r_{T+i}, $\forall i \in [B]$, and sends r_{T+i} to S along the feedback channel β_{T+i}.
3. Both the sender S and receiver R agree on the random numbers r_i, $\forall i \in [T + B]$ and consequently, the shared key $k = \sum_{i=1}^{T+B} r_i$ is established between S and R.
4. The sender S sends the message $m'(= m + k)$ to the receiver R along the forward channel α.
5. The receiver R simply subtracts k from m' and gets the message m.

The protocol Π_{RP} is a PSMT protocol. This is because the shared k is completely random/unknown to the adversary due to the fact that at least one random number r_i is unknown to the adversary as $T + B > t$. Therefore, the message m is secure. □

We have already noticed that the network G_3 given in Figure 3.3 fails to meet the conditions of the **Theorem 3.2** tolerating one passive fault. Whereas, the graph G_3 has one forward channel $\langle S, u, v, R\rangle$ and two disjoint channels $\langle R, u, S\rangle$, $\langle R, v, S\rangle$ - thus meets the conditions of the **Theorem 3.3** tolerating one passive fault.

3.5 PSMT in arbitrary directed graph model

In arbitrary directed setting, our inquiry includes: (a) *characterization*: under what conditions is a solution possible? (b) *feasibility*: is the characterization efficiently testable and is there an efficient protocol? (c) *round complexity*: what is the *fastest* solution? and (d) *communication complexity*: what is the *cheapest* solution? Intuitively, the above questions are in increasing order of difficulty. Consequently, question '(a)' has been answered in settings that are far more general than those where optimal solutions are known yet.

Although literature on information theoretically secure message transmission is rich (e.g., [1, 12, 14, 15, 32, 49, 65]), there are settings where answers to none of the aforementioned four questions are known yet. For instance, we do not know of a necessary and sufficient condition on digraphs influenced by a Byzantine adversary corrupting up to any t nodes for the existence of protocols for perfectly secure message transmission from S to R [50]; not to mention, the design of optimal protocols for the same are still far-fetched. Researchers have therefore attacked the PSMT problem in scenarios that are not as general as mentioned above – the harder the inquiry, the more specific the chosen setting.

As depicted in Figure 3.4, all the four questions in our inquiry, with respect to the problem of PSMT, have remained open in the general case of digraphs influenced by a Byzantine adversary characterized via an adversary structure. However, (im)possibility results are known if one restricts the setting to either undirected graphs [43] or passive adversary or security with error (e.g., [49,50]). Nevertheless, efficient protocols are still elusive. To design efficient protocols using contemporary techniques, further restriction (apart from moving to undirected graphs) is required, namely, *threshold* adversary. For instance, Dolev *et al.* in [1] have given one such efficient protocol, which, however, is neither round optimal nor bit-optimal.

Round-optimal protocols are known only in the case of weaker (not perfect) security models like statistical [47] or computational security [54]. Bit-optimal protocols have been designed in the wires-based abstraction of the undirected graph in [45]. While a similar wires-based approach has been used for digraphs too in [14], it is known to be inadequate to capture all digraphs on which protocols exist as shown in [47].

Our Contributions

As depicted in Figure 3.4, we ask: *does restricting to the setting of passive threshold adversaries lead to the design of efficient and round-optimal and/or communica-*

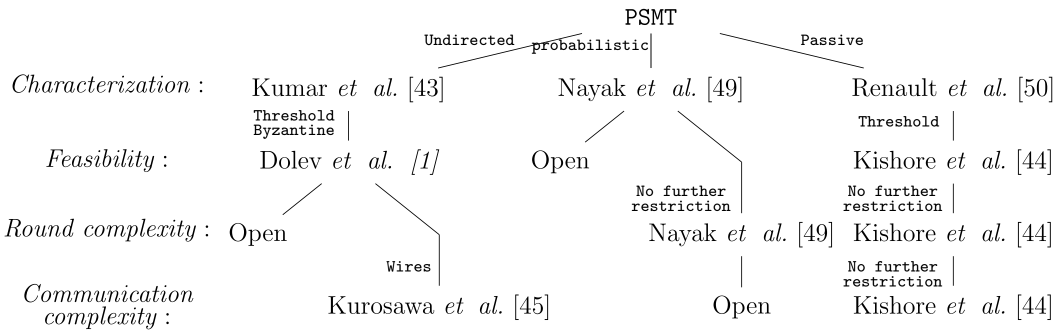

Figure 3.4: Restrictions based solutions.

tion optimal protocols? (or, are further restrictions like wires-based abstractions still required?)

Interestingly, we design communication efficient and round optimal protocols, with no further restrictions beyond assuming that the adversary passively corrupt up to t nodes in the digraph. Incidentally, it turns out that our techniques for designing round-optimal protocols are orthogonal to those that entail linear communication complexity – therefore, when applied together, we obtain protocols that are *simultaneously* round optimal as well as communication optimal. Further, the *simplicity* of our protocol ensures the implementability of highly scalable perfectly secret message transmission. Surprisingly, as proved in Section 3.8, it turns out that most of our protocols can be adapted to work for the mobile adversary case too. In a nutshell, we address the PSMT problem in such a way that all the four questions, namely, characterization, feasibility, communication and round optimality, are answered in one-shot. Below, we briefly describe our results and their significance.

1. **Complete characterization of networks wherein an r-round secret communication protocol tolerating static adversary is (im)possible**: In [1] Dolev et al. proved that $(t+1)$ vertex-disjoint paths are necessary and sufficient for PSMT from S to R in undirected graphs to tolerate passive t-threshold static adversary. Consequently, as noted in [1] too, without loss of generality, any network (undirected graph) may be abstracted as a set of wires (vertex-disjoint paths) between S and R. However, in the design of round optimal PSMT protocols, such an abstraction is inadequate even if the length of the wires is recorded. Specifically, using the edges connecting across these wires (or practically every edge in the network) it is possible to design *faster* protocols. For example, consider the graph in Figure 3.5. The two wires corresponding to two vertex-disjoint paths $\langle S, v, R\rangle$ and $\langle S(= v_0), v_1, v_2, v_3, \ldots v_{n-1}, R(= v_n)\rangle$ have lengths of 2 and n respectively.

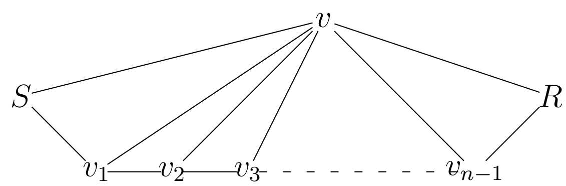

Figure 3.5: An undirected graph tolerating one passive fault.

Following Dolev's protocol, S sends two points on a linear polynomial whose constant term is the secret m, individually through these two wires. The receiver R gets the two points and hence the message after n rounds. Does a faster protocol exist? Our answer: *Yes. In fact, a 3-round protocol exists irrespective of how large n is.* Perhaps it is not conspicuous at first glance and certainly not if we continue to use the wires-based abstraction of the network. As a corollary to our **Theorem 3.13**, we know that *three* rounds are necessary and sufficient for S to R `PSMT` in the graph given in Figure 3.5. Thus, extant techniques are insufficient to design round optimal protocols and new techniques are necessary to design, and more importantly, prove round optimality. To summarize, the problem of characterizing round optimal protocols in directed networks is a non-trivial and interesting problem.

2. **Communication Complexity**: Folklore suggests that optimizing the number of rounds for a distributed protocol typically increases the communication complexity. In rare cases, round optimality can co-exist with communication-optimality – `PSMT` is indeed one such case! Specifically, we prove that the number of edges used by our protocol can be brought down to linear in the number of nodes (see Section 3.7.1). We also ensure that an edge is used to send at most one field element (or in general, bits equivalent to the size of the message).

   Thus, we arrive at a surprising protocol for secret communication which is round optimal and at the same time has linear communication complexity. Even more interesting is the case when the shortest path from S to R has $\Omega(n)$ nodes. In such cases, *perfect secrecy is achieved for "free"* because any (insecure routing) protocol would also take $\mathcal{O}(n)$ rounds and send $\mathcal{O}(n)$ messages for transmission – one message along each edge in the shortest path.

3. **Discriminant Algorithms**: Succinctly specifying the necessary and sufficient condition does not necessarily imply that there exists an efficient algorithm for checking the same. Indeed, the literature on the possibility of `PSMT` protocols in

directed graphs is replete with several problem specific characterizations, none of which are known to be efficiently testable. For instance, the possibility of reliable/secure message transmission in Byzantine adversarial setting in digraphs is characterized in [47,49]. However, no efficient algorithms to test these conditions are known. In fact, they may be NP-hard too as mentioned in [16] though no such study has been carried out. In contrast, for each of the results in this paper, we have a polynomial time algorithm for testing the same. *Algorithm 3.6.4* is a polynomial-time algorithm for testing the existence of an r-round secret communication protocol in a given network (and if yes, for obtaining a round optimal one).

4. **Mobile adversary**: Typically, mobile adversaries are notoriously difficult to withstand due to their dynamic movements across the network at a scorching pace. If the problem/protocol requires sustained long-distance collaboration for the task at hand, it is very easy for the mobile adversary to breach any kind of purported defences in-built in the protocol. And, we notice that in `PSMT` protocols it appears that the messages/packets need to travel across the network and therefore is easily susceptible to mobile adversarial attacks. A key ingredient in our solution tolerating mobile faults is the following: we address the problem by generating randomness across the network within a short span of time (say within one round) so that even a mobile adversary is bound to miss a substantial part of the random coins used by the protocol. More importantly, if the random-coins are locally deleted by the respective generators *before* the adversary can spy on them, there is ample scope for the protocol to withstand adversarial mobility as easily as its static counterpart. The challenge here is: what can be accomplished by random-coins that are ephemeral and have a very short life-span? We show that the answer isn't nothing; in particular, `PSMT` protocols can be designed with such short-lived randomness. In Section 3.8 we show how to use ephemeral random-coins and modify our static protocol to tolerate mobile faults.

3.5.1 Communication Efficient PSMT Protocol

Consider the graph G_4 given in Figure 3.6. We have already seen that `PSMT` is not possible in G_4 if both u and v are corrupted. Notice that, in G_4, the only node which can directly *talk* to R is u. Therefore, at first glance intuitively one feels that if u is corrupted then `PSMT` is impossible in G_4. But, counter-intuitively we show that `PSMT` is possible in G_4 tolerating one passive fault. Before presenting protocol, we note the

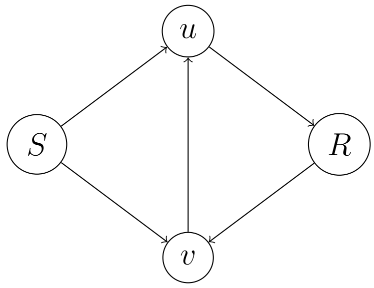

Figure 3.6: Graph G_4 with only one path

following observations. Firstly, both S and R can directly *talk* to v. Therefore R can send a key to v and S can send a message to v. In turn, v can blind the message received from S with the key sent by R. Secondly, as there is a path from v to R via u, namely, $p : \langle v, u, R \rangle$, v can send the blinded message to R using the path p. We achieve PSMT even if u is corrupted as the blinded message reveals nothing about the original message. More formally, the protocol works as follows.

Protocol Π_{G_4}:

1. The sender S picks a random one-degree polynomial $p(x)$ such that $p(0)$ is the message m.
2. The sender S sends $p(1)$ to the node u and $p(2)$ to the node v.
3. The receiver R picks a random number r and sends it to v.
4. The node v upon receiving both $p(2)$ and r computes $p(2) - r$ and sends it to the node u.
5. The node u forwards both $p(1)$ and $p(2) - r$ to the receiver R.
6. The receiver R upon receiving $p(2) - r$ computes $p(2)$ by adding r to $p(2) - r$.
7. The receiver R finally reconstructs $p(x)$ using the received points $p(1)$ and $p(2)$, consequently gets the message m.

In general, arbitrary directed graphs are much more complex than the graph G_4 given in Figure 3.6. Therefore, we should design a generic (graph independent) protocol which achieves PSMT in any given arbitrary directed graph, if at all one such protocol exists. This section contributes to the design of such a PSMT protocol $\boldsymbol{\Pi_{\text{Eff}}}$. Later, we illustrate the same with a more complex graph G given in Figure 3.8.

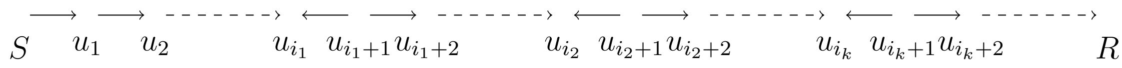

Figure 3.7: Weak path p.

We notice that, in any message transmission protocol, if there is no path from a node v to the receiver R, then v cannot convey any information to R. **Therefore, we assume that each node has at least one path to R.** In **Theorem 3.7** we show that PSMT is possible in G if and only if PSMT is possible in its underlying undirected graph G_u by designing a communication efficient PSMT protocol $\mathbf{\Pi_{Eff}}$ in G

In undirected graphs, we have seen a simple protocol, where, each disjoint path carries one point on t-degree polynomial. And, the honest path guarantees the security of the protocol. In directed graphs, we achieve the same effect with the protocol $\mathbf{\Pi_{Eff}}$. The core of the protocol $\mathbf{\Pi_{Eff}}$ is the sub-protocol $\mathbf{\Pi_{Sim}}$, which simulates the corresponding path p' of a given weak path p. Simulation we mean, for any given weak path p, the protocol $\mathbf{\Pi_{Sim}}$ always *reliably* transmits the message m from S to R using each node of p, as if p is a path. Moreover, if every node of p is honest then the adversary learns nothing about the message being transmitted using p. Thus, executing $\mathbf{\Pi_{Sim}}$ on $t+1$ disjoint weak paths results in the PSMT protocol $\mathbf{\Pi_{Eff}}$. Before presenting the protocol $\mathbf{\Pi_{Sim}}$, we first show that such a simulation is possible. Recall that, we use $[l, u]$ to denote the set $\{i \in \mathbb{Z} \mid l \leq i \leq u\}$.

Let $p : \langle S(= u_0), u_1, \ldots, u_l, u_{l+1}(= R)\rangle$ be a weak path in G. If p is a path in G then the simulation is trivial – S simply sends the message to u_1 and u_1 forwards it to u_2, u_2 in turn forwards it to u_3 and so on until it reaches R. If p is not a path in G, then there exist at least one u_i such that the forward edge $(u_i, u_{i+1}) \notin E$. Let $\{u_{i_1}, u_{i_2}, \ldots, u_{i_k}\}$ be the set of all nodes on the weak path p such that $(u_{i_j}, u_{i_j+1}) \notin E$ for $j \in [1, k]$. Without loss of generality, assume that $i_p < i_q$ for $p < q$ (see Figure 3.7). Also, from the context, it is clear that $u_{i_k} \neq R$.

Lemma 3.2. *In a graph G, if u, v, w are three honest nodes such that* PSMT *is possible from w to u and from w to v, then* PSMT *is possible from u to v as long as there is a path from u to v.*

Proof. Let m be the field element that u wants to secretly transmit to v. First, the node w chooses a random field element r and sends it to both u and v *secretly*, as PSMT is possible from w to both u and v. Now u masks the message m using the received number r as $m - r$ and sends it to the destination node v along a path from u to v, as there exists such a path. Finally, v obtains the message m by adding r to $m - r$. This protocol is perfectly secure even if the adversary corrupts the path from u to v, which carries $m-r$. Since, in a field $\langle \mathbb{F}, +, * \rangle$ for a given $x, z \in \mathbb{F}$, there exists a unique $y \in \mathbb{F}$ such that $x - y = z$. In other words, if the adversary corrupts the path from u to v then it learns $m - r$, which reveals nothing about m. □

Lemma 3.3. *In a directed graph G, let $p : \langle S(= u_0), u_1, \ldots, u_l, u_{l+1}(= R) \rangle$ be a weak path such that there exists a path from every node u_i (of the weak path p) to R. Then,* PSMT *from S to R is possible in G if no node of the weak path p is corrupted.*

Proof. Recall that, if p is a path in G then S simply sends the message to R along p. Therefore, PSMT from S to R is trivially possible in G (as no node of p is corrupted). If p is not a path in G, then recall that $\{u_{i_1}, u_{i_2}, \ldots, u_{i_k}\}$ is the set of all nodes that do not have a forward edge on the weak path p, where $i_k < l + 1$ (that is, $u_{i_k} \neq R$). As the node u_{i_k} is the last one satisfying $(u_{i_k}, u_{i_k+1}) \notin E$, there is a *secure* backward edge $(u_{i_k+1}, u_{i_k}) \in E$. For u_{i_k+1}, we have two cases:
Case (1): If $u_{i_k+1} = R$, then PSMT from u_{i_k+1} to R is trivially possible in G (as R can securely communicate with itself).
Case (2): If $u_{i_k+1} \neq R$ then (as no node of p is corrupted) there is a *secure path* from u_{i_k+1} to R along the nodes of the weak path p itself, which implies PSMT from u_{i_k+1} to R is possible in G.
Therefore, in any case, PSMT from u_{i_k+1} to R is possible in G. This implies, by applying the *Lemma* 3.2, we get that PSMT from u_{i_k} to R is possible in G. Now, we iteratively apply the above idea in reverse direction and show that PSMT from S to R is possible in G.

We notice that, for $j = k-1, k-2, \ldots, 1$:

1. We have a secure sub-path of p from u_{i_j+1} to $u_{i_{(j+1)}}$ in G (see Fig. 3.7). [1]

2. We have already shown that PSMT from $u_{i_{(j+1)}}$ to R is possible in G.

3. The above two steps (step 1 and 2) together ensure that PSMT from u_{i_j+1} to R is possible in G.

4. We have a secure backward edge $(u_{i_j+1}, u_{i_j}) \in E$.

5. The above two steps together (step 3 and 4), on applying *Lemma* 3.2, ensure that PSMT from u_{i_j} to R is possible in G.

[1] In case if $u_{i_j+1} = u_{i_{(j+1)}}$, then we trivially assume that there is a path from u_{i_j+1} to $u_{i_{(j+1)}}$ in G (as u_{i_j+1} can (securely) communicate with itself).

In particular, when $j = 1$, PSMT from u_{i_1} to R is possible in G. And, we have a secure sub-path of p from S to u_{i_1}, therefore, PSMT from S to R is possible in G. And, we have a secure sub-path of p from S to u_{i_1}, therefore, PSMT is possible from S to R. □

Communication Efficient Simulation

We use the above idea to design the protocol $\mathbf{\Pi_{Sim}}$ which simulates the corresponding path p' of a given weak path p. For each $j \in [1, k]$, we have $(u_{i_j}, u_{i_j+1}) \notin E$. This implies, there exist (i) a backward edge $(u_{i_j+1}, u_{i_j}) \in E$ and (ii) a sub-path of p, say p_{i_j+1}, from u_{i_j+1} to $u_{i_{(j+1)}}$ in G, where $u_{i_{(k+1)}}$ is R.

The Protocol $\mathbf{\Pi_{Sim}}$

1. The node u_{i_j+1} chooses a random number r_{i_j+1} and sends it to the node $u_{i_{(j+1)}}$ along the path p_{i_j+1}. Also, the node u_{i_j+1} sends r_{i_j+1} to the node u_{i_j} along the edge (u_{i_j+1}, u_{i_j}).
2. The node $u_{i_j}(j \neq 1)$ calculates $r_{i_{(j-1)}+1} - r_{i_j+1}$ and sends it to R along a path from u_{i_j} to R.
3. The sender S sends the message m to the node u_{i_1} along the path $\langle S(= u_0), u_1, u_2, \ldots, u_{i_1}\rangle$.
4. The node u_{i_1} calculates $m - r_{i_1+1}$ and sends it to R along a path from u_{i_1} to R.
5. `for` $j = k-1, k-2, \ldots, 1$: R recursively computes $r_{i_j+1} = (r_{i_j+1} - r_{i_{(j+1)}+1}) + r_{i_{(j+1)}+1}$.
6. Once R gets r_{i_1+1} for $j = 1$, it finally computes $m = (m - r_{i_1+1}) + r_{i_1+1}$.

Now with an example, we illustrate how the protocol $\mathbf{\Pi_{Sim}}$ works. Consider the graph G given in Figure 3.8 which has a maximum of three vertex-disjoint weak paths. Therefore, this graph can tolerate up to two faulty nodes. Let three weak paths be $p_1 : \langle S, v_1, v_2, R\rangle$, $p_2 : \langle S, v_3, v_4, R\rangle$ and $p_3 : \langle S, v_5, v_6, R\rangle$. The simulation of the corresponding paths of the weak paths p_1, p_2, p_3 are depicted in Figure 3.9, Figure 3.10 and Figure 3.11 respectively. The protocol $\mathbf{\Pi_{Sim}}$ for simulating p'_3 works as follows:

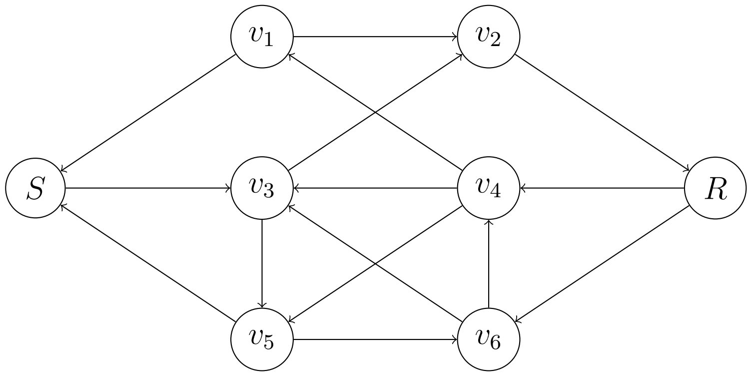

Figure 3.8: Graph G with three vertex-disjoint weak paths.

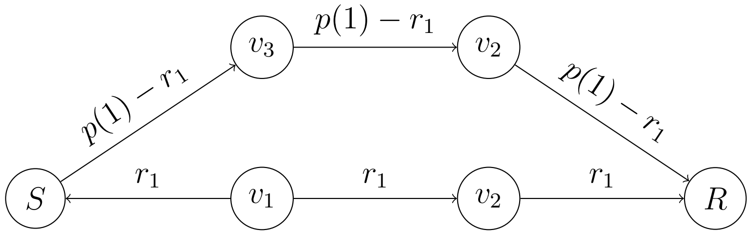

Figure 3.9: Simulation of the corresponding path p'_1.

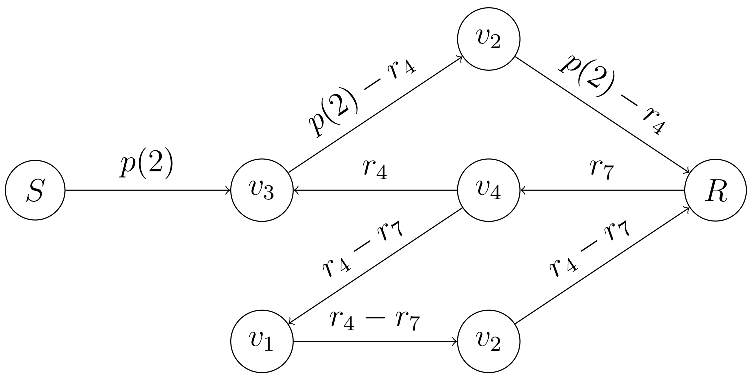

Figure 3.10: Simulation of the corresponding path p'_2.

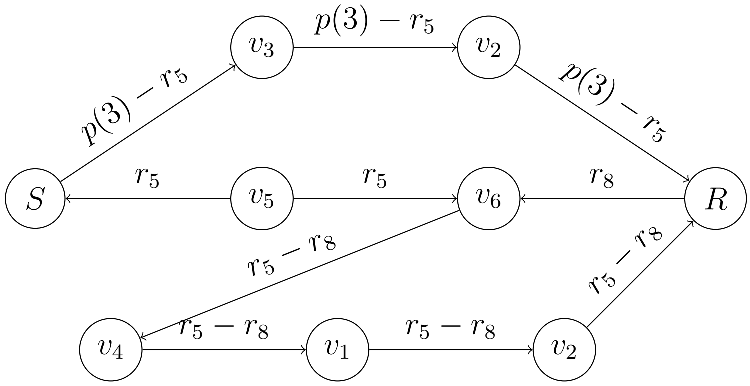

Figure 3.11: Simulation of the corresponding path p'_3.

1. R chooses a random number r_8 and sends it to v_6.
2. v_5 chooses a random number r_5 and sends it to both S and v_6.
3. v_6 masks r_5 with r_8 as $r_5 - r_8$ and sends it to R using the path $\langle v_6, v_4, v_1, v_2, R\rangle$.
4. S sends the masked value $p(3) - r_5$ to R using the path $\langle S, v_3, v_2, R\rangle$.
5. R first unmasks r_5 by adding r_8 to $r_5 - r_8$ and then unmasks $p(3)$ using r_5.

The correctness of the protocol $\mathbf{\Pi_{Sim}}$ is proved in the following theorem.

Theorem 3.4. *Let $G(V, E)$ be a directed graph in which S and R are two special nodes and $p : \langle S(= u_0), u_1, \ldots, u_l, u_{l+1}(= R)\rangle$ be a weak path such that there exists a path from every node u_i (of the weak path p) to R. Then, the protocol $\mathbf{\Pi_{Sim}}$ secretly transmits the message m from S to R in G if no node of the weak path p is corrupted.*

Proof. Let p be the path as given in the theorem statement and m be the message being transmitted by the protocol $\mathbf{\Pi_{Sim}}$. We know that the adversary cannot eavesdrop on any of these nodes as no node u_{i_j} is corrupted. However, for each $j \in [1, k]$, node u_{i_j} sends $r_{i_{(j-1)}+1} - r_{i_j+1}$ to R, where $r_{i_0+1} = m$. In the worst case, the adversary may intercept each of these values, in which case the view of the adversary is $\{r_{i_{(j-1)}+1} - r_{i_j+1} | j \in [1, k]\}$. We show that the view of the adversary is independent of the message being transmitted. In other words, we show that, for each view v of the adversary, there is exactly one valid execution of the protocol for every message m', and all these executions are equally likely.

Consider the following *valid* execution of the protocol $\mathbf{\Pi_{Sim}}$. Let m' be a message that is different from m, and define $r = m' - m$. Suppose each node u_{i_j+1} actually generates the random number $r_{i_j+1} + r$, for $j \in [1, k]$. Then, as per the protocol code, for each $j \in [1, k]$, node u_{i_j} sends $(r_{i_{(j-1)}+1} + r) - (r_{i_j+1} + r)$ to R. This implies, the view of the adversary is $\{(r_{i_{(j-1)}+1} + r) - (r_{i_j+1} + r) | j \in [1, k]\}$, which is nothing but $\{r_{i_{(j-1)}+1} - r_{i_j+1} | j \in [1, k]\}$. This shows that, the view of the adversary when the sender's message is m is the same as the view of the adversary when the sender's message is m', albeit for a different set of random coins of uncorrupted players. As m' is independent of m, the adversary's view is independent of the message being transmitted.

To prove the same mathematically, we individually compute $P[VIEW = v|M = m]$ and $P[VIEW = v|M = m']$ and show that these two probabilities are same.

Let m be the message being transmitted and $v = \{v_1, v_2, \ldots, v_k\}$ be the view of the adversary. Then, for each $j \in [1, k]$, $v_j = r_{i_{(j-1)}+1} - r_{i_j+1}$ if r_{i_j+1} is the random number generated by u_{i_j+1} for each $j \in [1, k]$, and $r_{i_0+1} = m$. This implies:

$$
\begin{aligned}
& P\big[VIEW = v|M = m\big] \\
= \;& P\big[(v_1 = r_{i_0+1} - r_{i_1+1}) \text{ and } (v_2 = r_{i_1+1} - r_{i_2+1}) \\
& \text{and } \ldots \text{ and } (v_k = r_{i_{(k-1)}+1} - r_{i_k+1}) \big| r_{i_0+1} = m\big] \\
= \;& P\big[(v_1 = m - r_{i_1+1}) \text{ and } (v_2 = r_{i_1+1} - r_{i_2+1}) \\
& \text{and } \ldots \text{ and } (v_k = r_{i_{(k-1)}+1} - r_{i_k+1})\big] \\
= \;& P\big[(r_{i_1+1} = m - v_1) \text{ and } (r_{i_2+1} = r_{i_1+1} - v_2) \\
& \text{and } \ldots \text{ and } (r_{i_k+1} = r_{i_{(k-1)}+1} - v_k)\big] \\
= \;& \frac{1}{|\mathbb{F}|^k}
\end{aligned}
$$

where the last step is because of k independent events, each one is occurring with probability of $\frac{1}{|\mathbb{F}|}$.

Similarly, let m' be the message being transmitted and $v = \{v_1, v_2, \ldots, v_k\}$ be the view of the adversary. Then, for each $j \in [1, k]$, $v_j = \mu_{i_{(j-1)}+1} - \mu_{i_j+1}$ if μ_{i_j+1} is the random number generated by u_{i_j+1} for each $j \in [1, k]$, and $\mu_{i_0+1} = m'$. This implies:

$$
\begin{aligned}
& P\big[VIEW = v|M = m'\big] \\
= \;& P\big[(v_1 = \mu_{i_0+1} - \mu_{i_1+1}) \text{ and } \\
& (v_2 = \mu_{i_1+1} - \mu_{i_2+1}) \text{ and } \ldots \text{ and } \\
& (v_k = \mu_{i_{(k-1)}+1} - \mu_{i_k+1}) \big| \mu_{i_0+1} = m'\big]
\end{aligned}
$$

$$
\begin{aligned}
&= P\big[(v_1 = m' - \mu_{i_1+1}) \text{ and } (v_2 = \mu_{i_1+1} - \mu_{i_2+1}) \\
&\quad \text{and } \dots \text{ and } (v_k = \mu_{i_{(k-1)}+1} - \mu_{i_k+1})\big] \\
&= P\big[(\mu_{i_1+1} = m' - v_1) \text{ and } (\mu_{i_2+1} = \mu_{i_1+1} - v_2) \\
&\quad \text{and } \dots \text{ and } (\mu_{i_k+1} = \mu_{i_{(k-1)}+1} - v_k)\big] \\
&= \frac{1}{|\mathbb{F}|^k}
\end{aligned}
$$

In other words, for every probability distribution on the message space, for every two distinct messages m, m' and every possible view v of the adversary, $P[VIEW = v|M = m] = P[VIEW = v|M = m']$. Therefore the protocol $\mathbf{\Pi_{Sim}}$ is perfectly secure. □

Efficient Protocol

We now present a communication efficient PSMT protocol $\mathbf{\Pi_{Eff}}$ in G if and whenever one exists. Recall that, in **Theorem 3.1** Dolev *et al.* [1] have shown that PSMT from S to R is possible *only if* there exist $(t+1)$ vertex-disjoint paths between S and R in G_u. This implies, $t+1$ vertex-disjoint weak paths from S to R are necessary for PSMT in G as well. Accordingly, let us assume that there are $t+1$ vertex-disjoint weak paths, namely p_i for each $i \in [1, t+1]$. The protocol is as follows.

The Protocol $\mathbf{\Pi_{Eff}}$

1. S chooses a random t-degree polynomial $p(x)$ such that the constant term $p(0)$ is the message m being transmitted to R.
2. S sends $p(i)$ to R by simulating the corresponding path p'_i of the weak path p_i using the protocol $\mathbf{\Pi_{Sim}}$, for each $i \in [1, t+1]$.
3. R reconstructs $p(x)$ once it receives all $t+1$ points and gets the message m.

Corollary 3.5. *The protocol* $\mathbf{\Pi_{Eff}}$ *is reliable.*

Proof. The reliability of the protocol $\mathbf{\Pi_{Sim}}$ assures that the receiver gets $t+1$ points on $p(x)$. And, Shamir's secret sharing scheme guarantees that these $t+1$ points are enough to reconstruct the polynomial and so is for the message m. □

Corollary 3.6. *The protocol* $\mathbf{\Pi_{Eff}}$ *is secure.*

Proof. We have $t+1$ vertex-disjoint weak paths and the adversary can corrupt at most t nodes. Therefore, there exist some $i \in [1, t+1]$ such that every node of the weak path

p_i is honest. This guarantees (from the **Theorem 3.4**) that the receiver R receives the point $p(i)$ reliably, whereas the adversary learns nothing about $p(i)$. This implies, in the worst case, the adversary learns at most t points on $p(x)$. And, the rest of the proof directly follows from the Shamir's secret sharing scheme [64]. □

The communication complexity of the protocol $\mathbf{\Pi_{Eff}}$ is $\mathcal{O}(|V|^2)$. This follows from the fact that, $t+1$ weak paths together may contain all the $|V|$ nodes and each of these nodes may need to send a masked value to the receiver R along some path, which in turn may contain $\mathcal{O}(|V|)$ nodes.

Theorem 3.7. `PSMT` *from S to R is possible in G if and only if* `PSMT` *is possible in G_u.*

Proof. **Necessity**: If `PSMT` is not possible in G_u, then clearly `PSMT` is not possible in G as G is a subgraph of G_u.

Sufficiency: If `PSMT` is possible in G_u, then run the protocol $\mathbf{\Pi_{Eff}}$ to achieve `PSMT` in G. □

Polynomial time algorithm to check if PSMT is possible in G

1. `If` edge $(R, S) \in E$ and there is a path from S to R in G or edge $(S, R) \in E$, then return *true*.
2. `Else`: create an auxiliary graph $G^{aux}(V^{aux}, E^{aux})$ of G as follows:
   (a) Split each vertex $v_i \in V$ except S and R into two vertices v_{i1} and v_{i2} and add an edge from v_{i1} to v_{i2}. $V^{aux} = \{S, R\} \cup_{i=1}^{n} \{v_{i1}, v_{i2}\}$.
   (b) Point all incoming edges of v_i to v_{i1} as incoming edges of v_{i1}.
   (c) Point all outgoing edges of v_i as outgoing edges of v_{i2}.
   (d) For every edge, add uniform edge capacity of 1.
3. In G_u^{aux} run the *Max-flow* algorithm to find the maximum flow, say f, from S to R.
4. If $f \geq t + 1$, then return *true* else return *false*.

This is a polynomial time algorithm, as the construction of graph G^{aux} requires $\mathcal{O}(|V|^2)$ time and *Max-flow* runs in $\mathcal{O}(|V|^3)$ time (see [66]). Also, note that $t+1$ vertex-disjoint paths can be found easily from the *Max-flow* algorithm.

3.6 Round optimality

This section contributes to the design of a round optimal protocol for perfectly secret message transmission. At first, it appears that the longest among the $t+1$ disjoint paths from S to R would act as a lower bound for the round complexity of PSMT. This is mainly because, to execute a protocol like $\mathbf{\Pi_{Sim}}$, each node needs to wait for the simulation to iteratively reach it, so that it can securely communicate a random number to R. However, recall the Figure 3.5 where it is noted that the length of the $(t+1)^{th}$ shortest path is not necessarily related to the minimum number of rounds required for PSMT. Intriguingly, constant-round protocols can sometimes exist in very large sparse graphs. This is because the (intermediate) nodes that need to send data to R, need not wait ($\mathbf{\Pi_{Sim}}$-like protocols) to iteratively simulate a secure channel to R – as what is being sent by them is just a random number. Specifically, in $\mathbf{\Pi_{Sim}}$, the receiver R receives the message masked by another random number, which yet again is masked by another random number and so on. R also receives securely (and iteratively) all these random numbers to successively unmask the message. Note that the message can be kept secret as long as none of these secondary/tertiary masks are unmasked. Therefore, all the randomness required for unmasking need not reach R in plain – in fact, it would suffice if (some sort of) a linear combination of them reaches R. This is exactly what we achieve through our protocol $\mathbf{\Pi_{Rnd\_Eff\_Sim}}$ in Section 3.6.1.

Note that once the bottleneck-of-iteration is circumvented, it is easy to apply the protocol $\mathbf{\Pi_{Rnd\_Eff\_Sim}}$ to obtain a round-efficient PSMT protocol $\mathbf{\Pi^{Static}_{Rnd\_Eff}}$ (see Section 3.6.2) in a manner exactly analogous to how the protocol $\mathbf{\Pi_{Eff}}$ designed using $t+1$ instances of $\mathbf{\Pi_{Sim}}$.

We remark that our round-efficient protocol is perhaps improvable further; thus the question of round-optimal protocols for PSMT is still yet to be fully addressed with the ideas discussed so far and new ideas are needed. Towards that end, we introduce in Section 3.6.3, the notion of a round evolution graph, a subgraph of G which evolves as the number of rounds increases. That is, the round evolution graph of order i is a subgraph of the round evolution graph of order $i+1$. Further, the full graph G evolves when the order number is $|V|$.

Crucially, we prove in **Theorem 3.12** that for any round evolution graph of order i, say H_i, if at all any PSMT protocol exists in H_i then our round efficient protocol $\mathbf{\Pi^{Static}_{Rnd\_Eff}}$ is an i-round PSMT protocol in H_i. Thus the smallest i for which our protocol $\mathbf{\Pi^{Static}_{Rnd\_Eff}}$ succeeds in securely transmitting the message in H_i is a *round optimal* PSMT protocol. We show that the search for such an i can be easily accomplished via the standard binary-search method. Note that a linear-search would also suffice for our purpose. However, we highlight that the setting is tailor-made for the much faster binary search method. We illustrate our round optimal protocol for the ongoing example.

3.6.1 Round Efficient Simulation Protocol $\mathbf{\Pi_{Rnd\_Eff\_Sim}}$

As discussed earlier, the protocol $\mathbf{\Pi_{Rnd\_Eff\_Sim}}$ simulates the corresponding path p' of a weak path p in the least number of rounds. In protocol $\mathbf{\Pi_{Rnd\_Eff\_Sim}}$, each node starts its computation and/or communication from the first round itself. And, if anything needs to be sent to R, it sends directly using a shortest path. This allows each node to convey the required information to R in the least number of rounds. Technical details are as follows. Let $p : \langle S(= u_0), u_1, \ldots, u_l, u_{l+1}(= R)\rangle$ be a weak path in G and m be the sender's message. Also, let p_{u_i} be a shortest path from u_i to R.

The Protocol $\mathbf{\Pi_{Rnd\_Eff\_Sim}}$

At the beginning of the first round:

1. Every node $u_i(\neq u_0)$ chooses a random number r_i, for each $i \in [1, l+1]$.
2. $S(= u_0)$ initializes $r_0 = m$ as well as $L[u_0] = m$.
3. For each $i \in [0, l]$:
   (a) *if* $(u_i, u_{i+1}) \in E$, then u_i sends r_i to u_{i+1} and initializes $R[u_i] = r_i$.
   (b) *else if* $(u_i, u_{i+1}) \notin E$, then u_{i+1} sends r_{i+1} to u_i and initializes $L[u_{i+1}] = r_{i+1}$.[a]

At the end of the first round:

1. For each $i \in [0, l]$:
   (a) *if* $(u_i, u_{i+1}) \in E$, then u_{i+1} receives r_i from u_i sent at the beginning of the first round and initializes $L[u_{i+1}] = r_i$.
   (b) *else if* $(u_i, u_{i+1}) \notin E$, then u_i receives r_{i+1} from u_{i+1} sent at the beginning of the first round and initializes $R[u_i] = r_{i+1}$.
   (c) every node u_i calculates its *value*, $Val[u_i] = L[u_i] - R[u_i]$.

Second round onwards:

1. For each $i \in [0, l]$: If $Val[u_i]$ is non-zero (i.e. $L[u_i] \neq R[u_i]$), then at the beginning of the second round, u_i sends $Val[u_i]$ to its out-neighbour of the shortest path p_{u_i}. In turn, at the beginning of the third round, the out-neighbour of u_i forwards $Val[u_i]$ to its out-neighbour of p_{u_i}. This process continues till the receiver receives $Val[u_i]$ from its in-neighbour of p_{u_i}.

2. Finally, the receiver R computes $m = (\sum_{i=0}^{l} Val[u_i]) + L[u_{l+1}]$.

[a] On any weak path, if u and v are two adjacent vertices such that $(u, v) \notin E$ then by definition $(v, u) \in E$.

3.6.2 Round Efficient Protocol $\Pi^{\text{Static}}_{\text{Rnd\_Eff}}$

1. S chooses a random t-degree polynomial $p(x)$ and replaces the constant term $p(0)$ with the message m.

2. S sends $p(i)$ to R by simulating the corresponding path p'_i of the weak path p_i using the protocol $\Pi_{\text{Rnd\_Eff\_Sim}}$, for each $i \in [1, t+1]$.

3. R reconstructs $p(x)$ once it receives all $t+1$ points to get the message m.

This protocol terminates in at most $|V|$ rounds. This is because, after sharing random numbers with their neighbours in the first round as per the protocol code, each node u sends $Val[u]$ (if it is non-zero) to R using the shortest path p_u. In any graph, as the length of every shortest path is trivially bounded by $|V|-1$, overall our protocol can take up to $|V|$ rounds.

Now we prove the correctness of the protocols $\Pi_{\text{Rnd\_Eff\_Sim}}$ and $\Pi^{\text{Static}}_{\text{Rnd\_Eff}}$.

Theorem 3.8. *The protocol $\Pi_{\text{Rnd\_Eff\_Sim}}$ for sending message m from S to R is reliable.*

Proof. By our protocol design, we have $R[u_i] = L[u_{i+1}]$ for each node u_i (except R) on the weak path p. As R finally computes the $Sum = (\sum_{i=0}^{l} Val[u_i]) + L[u_{l+1}]$, we show that the Sum is nothing but m.

$$\begin{aligned} Sum &= (\sum_{i=0}^{l}(L[u_i] - R[u_i])) + L[u_{l+1}] \\ &= (\sum_{i=0}^{l}(L[u_i] - L[u_{i+1}])) + L[u_{l+1}] \\ &= L[u_0] - L[u_{l+1}] + L[u_{l+1}] = L[u_0] = m \end{aligned}$$

□

Corollary 3.9. *The protocol* $\mathbf{\Pi^{Static}_{Rnd\_Eff}}$ *for sending message* m *from* S *to* R *is reliable.*

Proof. Reliability of the protocol $\mathbf{\Pi_{Rnd\_Eff\_Sim}}$ assures that the receiver gets $t+1$ points on t-degree polynomial $p(x)$. And, Shamir's secret sharing scheme [64] ensures that the message m can be reconstructed from these $t+1$ points. □

Theorem 3.10. *The protocol* $\mathbf{\Pi_{Rnd\_Eff\_Sim}}$ *for simulating the corresponding path* p' *of a weak path* $p : \langle S(= u_0), u_1, \ldots, u_l, u_{l+1}(= R)\rangle$*, secretly transmits message* m *from* S *to* R *if all the nodes on* p *are honest.*

Proof. The proof is analogous to the proof given in **Theorem 3.4**. We notice that, other than R, each node u_i on the weak path p sends $Val[u_i]$ (if it is non-zero) to the receiver R using the shortest path p_{u_i}. In the worst case, the adversary may learn $Val[u_i]$, for each $i \in [0, l]$. In this case too, we show that the adversary learns nothing about m by showing that the *view* of the adversary is independent of the message being transmitted.

In the execution of the protocol $\mathbf{\Pi_{Rnd\_Eff\_Sim}}$ for the sender's message m, the *view* of the adversary is $\{Val[u_i] | i \in [0, l]\}$, where $L[u_0] = m$ and $Val[u_i] = L[u_i] - R[u_i] = L[u_i] - L[u_{i+1}]$. Let us denote $L[u_i] = r'_i$ for $i \in [0, l+1]$, thus the *view* of the adversary is $\{r'_i - r'_{i+1} | i \in [0, l]\}$.

Now consider another *valid* execution of the protocol $\mathbf{\Pi^{Static}_{Rnd\_Eff}}$. Let $m + r$ be the sender's message, for an arbitrary fixed random r. And, suppose the node u_i actually generates the random number $r_i + r$, for each $i \in [1, l+1]$. In this execution of the protocol, the *view* of the adversary is $\{Val[u_i] | i \in [0, l]\}$, where $Val[u_i] = L[u_i] - L[u_{i+1}] = (r'_i + r) - (r'_{i+1} + r) = r'_i - r'_{i+1}$. The rest of the proof follows exactly as in the proof of the **Theorem 3.4**. Therefore, the protocol $\mathbf{\Pi_{Rnd\_Eff\_Sim}}$ is secure.

□

Corollary 3.11. *The protocol* $\mathbf{\Pi^{Static}_{Rnd\_Eff}}$ *for sending message* m *from* S *to* R *is secure.*

Proof. As the adversary can corrupt at most t nodes, there exists $i \in [1, t+1]$, such that every node of the weak path p_i is honest. And, the protocol $\mathbf{\Pi_{Rnd\_Eff\_Sim}}$ assures that $p(i)$ is secure. We have from Shamir's secret sharing scheme that t or fewer points on a t-degree polynomial reveals nothing about the constant term, which is the message. $\square$

3.6.3 PSMT in Round Evolution Graphs

Graphs have been used as a very powerful abstraction of the network by modelling the physical link from one player to another as a directed edge between the corresponding vertices of the graph. However, in this kind of modelling of the network, the edges of the graph only indicate the link between two spatial locations. It does not contain any temporal information. To incorporate the notion of time (rounds) in our graph, we propose a representation named *round evolution graph* that contains both spatial and temporal information.

Definition 3.1. *Given a round number r and a network represented by a directed graph $G(V, E)$ with the receiver R, the round evolution graph of order r, $G^{(r)}(V, E^{(r)})$ is defined as the subgraph of G, where edge set $E^{(r)} = E \setminus \{(u, v) \in E \mid d_v \geq r\}$, where d_v denotes the length of the shortest path from v to R. In other words, remove those edges from which R can't receive any information in r rounds.*

Theorem 3.12. `PSMT` *is possible in $G^{(r)}$ if and only if an r-round* `PSMT` *protocol exists in $G^{(r)}$.*

Proof. **Sufficiency**: If an r-round `PSMT` protocol exists in $G^{(r)}$, then `PSMT` is trivially possible in $G^{(r)}$.
Necessity: Suppose `PSMT` is possible in $G^{(r)}$, then we show that the *round efficient protocol* $\mathbf{\Pi^{Static}_{Rnd\_Eff}}$ given in Section 3.6.2 achieves `PSMT` in r rounds. As the protocol $\mathbf{\Pi^{Static}_{Rnd\_Eff}}$ is nothing but executing $t+1$ times the protocol $\mathbf{\Pi_{Rnd\_Eff\_Sim}}$, it is enough to show that the protocol $\mathbf{\Pi_{Rnd\_Eff\_Sim}}$ succeeds in r-rounds.

We observe that each node u_i on the weak path $p : \langle S(= u_0), u_1, \ldots, u_l, u_{l+1}(= R)\rangle$, sends the chosen random number r_i to its neighbour(s) in the first round as per the protocol $\mathbf{\Pi_{Rnd\_Eff\_Sim}}$. We have three cases for each node u_i of the weak path p:

1. $(u_{i-1}, u_i) \in E^{(r)}$. By our construction of $G^{(r)}$, $d_{u_i} \leq r - 1$, therefore, even u_i receives random numbers from its neighbour(s) only at the end of the first round, it can send $Val[u_i]$ to R in a total of r-rounds.

2. $(u_i, u_{i+1}) \notin E^{(r)}$. This implies $(u_{i+1}, u_i) \in E^{(r)}$. By our construction of $G^{(r)}$, $d_{u_i} \leq r - 1$. The rest follows as in previous case.

3. $(u_{i-1}, u_i) \notin E^{(r)}$ but $(u_i, u_{i+1}) \in E^{(r)}$. In this case it is trivial, as $Val[u_i] = L[u_i] - R[u_i] = r_i - r_i = 0$, u_i is not required to send its *value* to the receiver R.

□

Theorem 3.13. *An r-round* PSMT *protocol exists in G if and only if* PSMT *is possible in the* round evolution graph $G^{(r)}$ *of order r.*

Proof. **Sufficiency**: If PSMT is possible in $G^{(r)}$, then, the theorem directly follows from **Theorem 3.12** as $G^{(r)}$ is a subgraph of G.
Necessity: Assume that an r-round PSMT protocol Π exists in G. We show that for the same protocol Π, the *extra* edges which are present in E but not in $E^{(r)}$ never convey *any* information to R. This implies, at the end of the protocol Π, the *view* of the receiver R remains same whether these edges are present or not. Therefore, any such r-round protocol Π achieves PSMT in $G^{(r)}$.

Let (u, v) be an edge in E but not in $E^{(r)}$. This implies, by definition of $E^{(r)}$, $d_v \geq r$. As the shortest distance from v to R is at least r, any message sent by v takes at least r rounds to reach R. Also we know that, when u sends a message to v using edge (u, v) it does so in one round. Therefore, a total of at least $r + 1$ rounds are required for any message to reach R from u via edge (u, v). Therefore, these edges are of no use in any r-round protocol. This concludes the proof. □

Corollary 3.14. *An r-round* PSMT *protocol exists in G if and only if an r-round* PSMT *protocol exists in* $G^{(r)}$.

3.6.4 Polynomial time algorithm for identifying the optimal number of rounds

We have from *Corollary* 3.14 that the optimal number of rounds required for PSMT in G is nothing but the least r for which PSMT is possible in $G^{(r)}$. Also, it is easy to see that if PSMT is possible in $G^{(r)}$, then PSMT is trivially possible in $G^{(r+1)}$. Combining these two together, we get the *optimal* r for which PSMT is possible in G if we perform the standard binary search over $r \in [1, |V|]$ while we check for PSMT possibility in $G^{(r)}$. Since the overhead of binary search is logarithmic, we just need to show that each iteration of binary search takes only polynomial time. In each iteration, we are constructing the subgraph $G^{(r)}$ of G and checking if PSMT is possible in $G^{(r)}$. Constructing a subgraph $G^{(r)}$ of G requires only quadratic (polynomial) time. And, in Section 3.5.1, we have already shown that we can efficiently check if PSMT is possible in any given graph.

3.6.5 An Example of Round Optimal Protocol

In this section, we illustrate the protocol $\mathbf{\Pi^{Static}_{Rnd\_Eff}}$ with an example. Let us consider the graph G given in Figure 3.8. We have already seen that, in G PSMT is possible tolerating two faults but not three.

To execute our round optimal protocol, we need a shortest path from each node to R. Also, we have to find the least r for which PSMT is possible in $G^{(r)}$ tolerating two faults. For quick reference, the shortest distance and a shortest path from each node to R is shown in Figure 3.12. And, *round evolution graphs* $G^{(3)}$ of order three and $G^{(4)}$ of order four are depicted in Figure 3.13 and Figure 3.14 respectively.

We notice that the shortest distance from S to R is three. Therefore, any protocol will take at least three rounds. But, in $G^{(3)}_u$ there is only one (vertex-disjoint) path from S to R. Thus, it fails to meet the necessary conditions of **Theorem 3.7** tolerating two faults. This implies that PSMT is impossible in $G^{(3)}$. On the other hand, there are three vertex-disjoint paths from S to R in $G^{(4)}_u$, and every node on these paths has a path to R in G. Thus, PSMT is possible in $G^{(4)}$ tolerating two faults. Therefore, minimum number of rounds required for PSMT in G is four. And, our four round protocol works as follows.

An execution of four round protocol in $\mathbf{G^{(4)}}$

At the beginning of the first round:

1. The sender S chooses a random two-degree polynomial $p(x)$ and replaces the constant term $p(0)$ with the message m.
2. Every node v, except S and R, chooses a random number r_v. And, R chooses three random numbers $r_{R_1}, r_{R_2}, r_{R_3}$.
3. Node v_1 sends r_{v_1} to the node v_2 as well as the sender $S(= S_1)$.
4. Node v_2 sends r_{v_2} to the receiver R.
5. Node $S(= S_2)$ sends $p(2)$ to the node v_3.
6. Node v_4 sends r_{v_4} to the node v_3.
7. Node R sends r_{R_2} to the node v_4.

Node	Shortest path to R	Shortest distance
S	$p_S : \langle S, v_3, v_2, R \rangle$	3
v_1	$p_{v_1} : \langle v_1, v_2, R \rangle$	2
v_2	$p_{v_2} : \langle v_2, R \rangle$	1
v_3	$p_{v_3} : \langle v_3, v_2, R \rangle$	2
v_4	$p_{v_4} : \langle v_4, v_3, v_2, R \rangle$	3
v_5	$p_{v_5} : \langle v_5, v_6, v_3, v_2, R \rangle$	4
v_6	$p_{v_6} : \langle v_6, v_3, v_2, R \rangle$	3

Figure 3.12: Shortest paths from each node to the receiver R for the graph given in Figure 3.8.

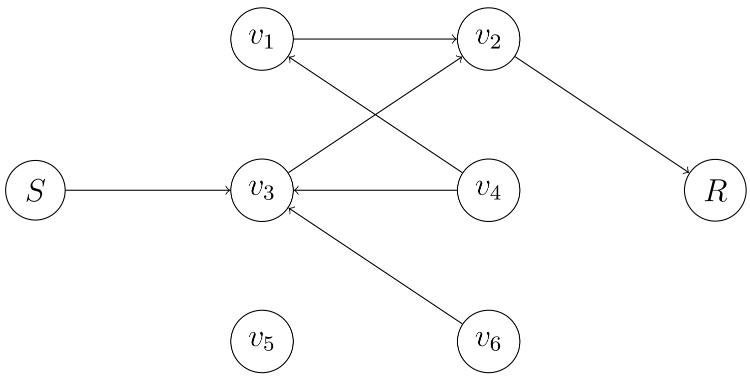

Figure 3.13: Round Evolution Graph $G^{(3)}$ of order three for the graph given in Figure 3.8.

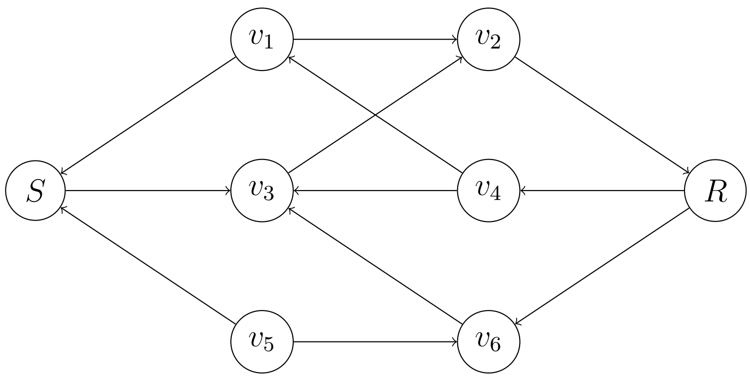

Figure 3.14: Round Evolution Graph $G^{(4)}$ of order four for the graph given in Figure 3.8.

8. Node v_5 sends r_{v_5} to both node v_6 and the sender $S(= S_3)$.

9. Node R sends r_{R_3} to the node v_6.

At the end of the first round:

1. Every node v, except R, calculates its *value*, $Val[v]$:

   (a) $Val[S_1] = p(1) - r_{v_1}$, $Val[v_1] = r_{v_1} - r_{v_1}$, $Val[v_2] = r_{v_1} - r_{v_2}$

   (b) $Val[S_2] = p(2) - p(2)$, $Val[v_3] = p(2) - r_{v_4}$, $Val[v_4] = r_{v_4} - r_{R_2}$

   (c) $Val[S_3] = p(3) - r_{v_5}$, $Val[v_5] = r_{v_5} - r_{v_5}$, $Val[v_6] = r_{v_5} - r_{R_3}$

round two, round three and round four:

1. Every node $v \notin \{v_1, S_2, v_5, R\}$ sends its *value* $Val[v]$ to R using a shortest path p_v from v to R(see Figure 3.12).

2. R calculates:

   (a) $p(1) = Val[S_1]+Val[v_1]+Val[v_2]+r_{v_2} = p(1)-r_{v_1}+0+r_{v_1}-r_{v_2}+r_{v_2} = p(1)$

   (b) $p(2) = Val[S_2]+Val[v_3]+Val[v_4]+r_{R_2} = 0+p(2)-r_{v_4}+r_{v_4}-r_{R_2}+r_{R_2} = p(2)$

   (c) $p(3) = Val[S_3]+Val[v_5]+Val[v_6]+r_{R_3} = p(3)-r_{v_5}+0+r_{v_5}-r_{R_3}+r_{R_3} = p(3)$

The shortest distance from each node (except v_5) to the receiver R is less than or equal to three. And, each node in G may have to receive random number(s) from its neighbour(s) in the first round. Therefore, each node (except v_5) can send its *value* to the receiver R in at most four rounds. As per the protocol code, v_5 does not send anything to R since the *value* of the node v_5 is zero. Therefore, this protocol terminates in four rounds.

3.7 Linear communication complexity

This section contributes to the design of a round optimal `PSMT` protocol, whose communication complexity is *linear* in the number of *vertices* of the graph. In Section. 3.5.1, we have seen that the communication complexity of the protocol $\mathbf{\Pi_{Eff}}$ is $\mathcal{O}(|V|^2)$ due to the following reason. Once sharing is done in the first round, each node sends its *value* to the receiver using a shortest path. In particular, each of these shortest paths may contain $\mathcal{O}(|V|)$ nodes, leading to quadratic complexity. However, we notice that many of these shortest paths may have several edges in common. And, each such edge has to carry k field elements if it is part of k shortest paths. We make sure that such edges carry only one field element, leading us to the design of a linear-communication protocol. More details are as follows. We construct a subgraph H of G such that `PSMT` is possible in G *iff* `PSMT` is possible in H. And, H contains only $\mathcal{O}(|V|)$ edges. Therefore, if we design a protocol in H such that each edge in H carries *at most one* field element then trivially we get a linear communication protocol in G. As this technique can be adapted to any graph, we work with the round evaluation graph $G^{(r)}$ of order r, where r is the optimal number of rounds required for `PSMT`; to get a round optimal protocol with linear communication complexity. We construct H as follows:

1. H contains only $\mathcal{O}(|V|)$ edges – The edge set of H is the union of two sets of edges. First one is the set of edges of $t+1$ disjoint weak paths. The other one is the set of edges of a tree with R as its root. More elaborately, suppose the shortest distance from a node u to R is d. Then, to send any information to R possibly in the least number of rounds, the node u must use another node whose shortest distance to R is $d-1$. We realize this by constructing a tree T of G with R as its root such that each node has exactly one path to R in the tree T. That is, a node in the i^{th} level connected to only one of its parent which is in the $(i-1)^{th}$ level, assuming R is at 0^{th} level. We already know that, no tree can have more than $|V|-1$ edges. Therefore, H contain only $\mathcal{O}(|V|)$ edges.

2. `PSMT` is possible in H whenever `PSMT` is possible in G – If `PSMT` is possible in G then G contains $t+1$ disjoint weak paths. As H contains every edge of these $t+1$ disjoint weak paths, nodes can share their random numbers with neighbours in the first round as per the protocol $\mathbf{\Pi^{Static}_{Rnd\_Eff}}$. Also, each node can send its *value* to R as it has a *shortest* path to R in the tree.

3. Each edge in H carries at most one field element – Instead of working with random t-degree polynomial, the sender randomly choose $t+1$ field elements, say m_i for $i \in [1, t+1]$, such that their sum is the message m. To get the message m, it is *not necessary* for R to know each individual m_i but it is enough if R gets the corresponding sum. Therefore, instead of pushing the calculation to R at the end, each node locally adds all the *values* it received from its children with its *value* and sends as a single field element to its parent in the tree T. In other words,

if an edge is part of multiple shortest paths, then instead of carrying multiple messages, it carries only one field element which is the sum of the corresponding multiple messages.

Now we are ready to formally introduce required definitions.

Definition 3.2. *Let $G(V, E)$ be a directed graph such that every node in V has at least one path to the fixed node $R \in V$. A* ***Reverse Directed Rooted Tree*** *of G rooted at R is a digraph $T_G(V, E_T; R)$ such that a node u is at the i^{th} level (R is at level 0) if and only if the shortest distance from u to R is exactly i in G.*

Note on Reverse Directed Rooted Tree: Every node in the tree has exactly one parent, else we would get cycles in T_G. Moreover, as there are no cycles, the maximum number of edges present in tree T_G is $|V| - 1$.

Definition 3.3. *Let $G(V, E)$ be a directed graph with k vertex-disjoint weak paths from S to R, namely p_i for each $i \in [1, k]$ such that every node in these k weak paths has at least one path to R. Let $T_G(V, E_T; R)$ be a Reverse Directed Rooted Tree of G. A communication graph of the digraph $G(V, E)$ of order k is denoted by $\mathcal{G}^k(V, \mathcal{E})$ and defined as $\mathcal{E} = E_p \cup E_T$, where $E_p = \bigcup_{i=1}^{k} E(p_i)$ and $E(p_i)$ is the set of all edges in the weak path p_i.*

Theorem 3.15. `PSMT` *from S to R is possible in graph G if and only if* `PSMT` *is possible in communication graph $\mathcal{G}^{(t+1)}$ of order $t + 1$.*

Proof. **Sufficiency**: Suppose `PSMT` is possible from S to R in G. Then from **Theorem 3.7**, we know that there exist at least $t + 1$ vertex-disjoint weak paths from S to R such that every node on these weak paths has a path to R in G. Observe that, by definition of $\mathcal{G}^{(t+1)}$, every edge of these $t+1$ weak paths is present in $\mathcal{G}^{(t+1)}$. Therefore, nodes can share their random numbers with their neighbours in the first round using these edges as per the protocol $\mathbf{\Pi^{Static}_{Rnd\_Eff}}$. Also, by the definition of Reverse Directed Rooted Tree of G, each node on these $t + 1$ weak paths has a unique shortest path to R in T_G. Therefore, each node on these $t + 1$ weak paths can send its *value* to R as per protocol $\mathbf{\Pi^{Static}_{Rnd\_Eff}}$.

Necessity: If `PSMT` is impossible in G then trivially `PSMT` is impossible in the subgraph $\mathcal{G}^{(t+1)}$ as well. □

3.7.1 Round optimal protocol with linear communication complexity

In this section, we present a round optimal linear communication protocol $\mathbf{\Pi^{Static}_{Rnd\_Opt\_Lin}}$ if PSMT is possible in G. As discussed earlier, the protocol $\mathbf{\Pi^{Static}_{Rnd\_Opt\_Lin}}$ is same as the protocol $\mathbf{\Pi^{Static}_{Rnd\_Eff}}$ except that (1) we run the protocol in $\mathcal{G}^{(t+1)}$ (2) for each $i \in [1, t+1]$, $p(i)$ is replaced with m_i, where the sum of these m_i's is the message m and (3) if an edge (u, v) carries more than one field element then u adds corresponding field elements and sends to its parent v in T as a single field element.

Let r be the optimal number of rounds required for PSMT possibility in G tolerating t-threshold static adversary. Then, we have from *Corollary* 3.14 that, PSMT is possible in $G^{(r)}$. This implies, combing with **Theorem 3.15**, PSMT is possible in communication graph $\mathcal{G}^{(t+1)}$ of the digraph $G^{(r)}(V, E)$. Therefore, $(t+1)$ vertex-disjoint weak paths from S to R exist in $\mathcal{G}^{(t+1)}$, namely $p_i : \langle u_{i0}(= S), u_{i1}, \dots, u_{ik_i}, u_{i(k_i+1)}(= R)\rangle$, for each $i \in [1, t+1]$. Let the height of a Reverse Directed Rooted Tree $T_{G^{(r)}}$ of $G^{(r)}$ be h with root R is at the 0^{th} level. Here notice that, each u_{i0} is S and each $u_{i(k_i+1)}$ is R.

The Protocol $\mathbf{\Pi^{Static}_{Rnd\_Opt\_Lin}}$

At the beginning of the first round:

1. Each node u_{ij}, except $u_{(t+1)0}$, picks a random number $r_{ij} \in \mathbb{F}$, for $i \in [1, t+1]$ and $j \in [0, k_i + 1]$.
2. $S(= u_{(t+1)0})$ computes $r_{(t+1)0} = m - \sum_{i=1}^{t} r_{i0}$.
3. For each $i \in [1, t+1]$: $S(= u_{i0})$ initializes $L[u_{i0}] = r_{i0}$.
4. For each $i \in [1, t+1]$ and $j \in [0, k_i]$:
   (a) if $(u_{ij}, u_{i(j+1)}) \in E^{(r)}$, then u_{ij} sends r_{ij} to $u_{i(j+1)}$ and initializes $R[u_{ij}] = r_{ij}$.
   (b) if $(u_{ij}, u_{i(j+1)}) \notin E^{(r)}$, then $u_{i(j+1)}$ sends $r_{i(j+1)}$ to u_{ij} and initializes $L[u_{i(j+1)}] = r_{i(j+1)}$.

At the end of the first round:

1. For each $i \in [1, t+1]$ and $j \in [0, k_i]$:

(a) if $(u_{ij}, u_{i(j+1)}) \in E^{(r)}$, then $u_{i(j+1)}$ receives r_{ij} from u_{ij} sent at the beginning of the first round and initializes $L[u_{i(j+1)}] = r_{ij}$.

(b) if $(u_{ij}, u_{i(j+1)}) \notin E^{(r)}$, then u_{ij} receives $r_{i(j+1)}$ from $u_{i(j+1)}$ sent at the beginning of the first round and initializes $R[u_{ij}] = r_{i(j+1)}$.

(c) Each node u_{ij} calculates its *value*, $Val[u_{ij}] = L[u_{ij}] - R[u_{ij}]$.

2. The sender S computes its *grand value*, $Val[S] = \sum_{i=1}^{t+1} Val[u_{i0}]$

Second round onwards:

1. `If` S is a leaf node (at level h) in $T_{G^{(r)}}$ and its *grand value*, $Val[S]$, is non-zero then S sends $Val[S]$ to its parent at $(h-1)^{th}$ level.

2. `Else`, if S is at k^{th} ($k \in [1, h-1]$) level then S receives *values*, which are non-zero, from its children at $(k+1)^{th}$ level. Subsequently, S *adds* all the received *values* to its *grand value* $Val[S]$ and sends to its parent which is at $(k-1)^{th}$ level.

3. For each $i \in [1, t+1]$ and $j \in [1, k_i]$:

   (a) `If` u_{ij} is a leaf node (at level h) in $T_{G^{(r)}}$ and its *value*, $Val[u_{ij}]$, is non-zero then u_{ij} sends $Val[u_{ij}]$ to its parent at $(h-1)^{th}$ level.

   (b) `Else`, each u_{ij} which is at k^{th} ($k \in [1, h-1]$) level receives *values*, which are non-zero, from its children at $(k+1)^{th}$ level. Subsequently, u_{ij} *adds* all the received values to its *value* $Val[u_{ij}]$ and sends to its parent which is at $(k-1)^{th}$ level.

4. In the last round, the receiver R adds the sum of all the values it received from its children with the sum of all its *Left Values* (i.e. $\sum_{i=1}^{t+1} L[u_{i(k_i+1)}]$) to get the message m.

Theorem 3.16. *The protocol* $\mathbf{\Pi^{Static}_{Rnd\_Opt\_Lin}}$ *is reliable.*

Proof. It is clear that the receiver R eventually gets the sum of the *grand value* of S and the sum of the *values* of each u_{ij}, for $i \in [1, t+1]$ and $j \in [1, k_i]$. As R computes the $Sum = Val[S] + (\sum_{i=1}^{t+1}\sum_{j=1}^{k_i} Val[u_{ij}]) + \sum_{i=1}^{t+1} L[u_{i(k_i+1)}]$ to get the message, we should show that the Sum is nothing but the message m. Recall that, on each weak path p_i, we have $R[u_{ij}] = L[u_{i(j+1)}]$ for each u_{ij}, where $j \in [0, k_i]$.

$$Sum = Val[S] + (\sum_{i=1}^{t+1}\sum_{j=1}^{k_i} Val[u_{ij}]) + \sum_{i=1}^{t+1} L[u_{i(k_i+1)}]$$

$$Sum = \sum_{i=1}^{t+1} Val[u_{i0}] + (\sum_{i=1}^{t+1}\sum_{j=1}^{k_i} Val[u_{ij}]) + \sum_{i=1}^{t+1} L[u_{i(k_i+1)}]$$

$$Sum = \sum_{i=1}^{t+1}(L[u_{i0}] - R[u_{i0}]) + (\sum_{i=1}^{t+1}\sum_{j=1}^{k_i}(L[u_{ij}] - R[u_{ij}])) + \sum_{i=1}^{t+1} L[u_{i(k_i+1)}]$$

$$Sum = \sum_{i=1}^{t+1}(L[u_{i0}] - L[u_{i1}]) + (\sum_{i=1}^{t+1}\sum_{j=1}^{k_i}(L[u_{ij}] - L[u_{i(j+1)}])) + \sum_{i=1}^{t+1} L[u_{i(k_i+1)}]$$

$$Sum = \sum_{i=1}^{t+1}(L[u_{i0}] - L[u_{i1}]) + (\sum_{i=1}^{t+1}(L[u_{i1}] - L[u_{i(k_i+1)}])) + \sum_{i=1}^{t+1} L[u_{i(k_i+1)}]$$

$$Sum = \sum_{i=1}^{t+1} L[u_{i0}] - \sum_{i=1}^{t+1} L[u_{i1}] + \sum_{i=1}^{t+1} L[u_{i1}] - \sum_{i=1}^{t+1} L[u_{i(k_i+1)}] + \sum_{i=1}^{t+1} L[u_{i(k_i+1)}]$$

$$Sum = \sum_{i=1}^{t+1} L[u_{i0}] = \sum_{i=1}^{t+1} r_{i0} = m.$$

□

Theorem 3.17. *The protocol* $\mathbf{\Pi^{Static}_{Rnd\_Opt\_Lin}}$ *is secure.*

Proof. The proof directly follows from the security proof of the protocol $\mathbf{\Pi_{Rnd\_Eff\_Sim}}$ given in **Theorem 3.10**. In **Theorem 3.10**, we showed that, once sharing of random numbers is done in the first round, in subsequent rounds even if the adversary gets $Val[u_i]$, for each u_i of the honest weak path p, the adversary learns nothing about the message m being transmitted to R. The protocol code of the first round of the current protocol $\mathbf{\Pi^{Static}_{Rnd\_Opt\_Lin}}$ is same as that of the protocol $\mathbf{\Pi_{Rnd\_Eff\_Sim}}$. Also, we have $t+1$ vertex-disjoint weak paths from S to R. Therefore, at least one weak path p_i is honest, for some $i \in [1, t+1]$. Combining all together, we get, the adversary learns nothing about r_{i0}. Thus, the adversary learns nothing about the message $m = \sum_{j=1}^{t+1} r_{j0}$. □

The communication complexity of the protocol $\mathbf{\Pi^{Static}_{Rnd\_Opt\_Lin}}$ is linear. The reason is as follows. In the first round, each edge of the weak paths carries exactly one field element r_{ij}, for some $i \in [1, t+1]$ and $j \in [0, k_i+1]$. As the weak paths are disjoint, the number of edges is bounded by $|V|$. Also, each edge of the Reverse Directed Rooted Tree $T_{G^{(r)}}$ carries at most one field element. And, the number of edges in $T_{G^{(r)}}$ is also bounded by $|V|-1$. Therefore, to transmit a single field element secretly, all the edges together carry at most $\mathcal{O}(|V|)$ field elements.

An interesting implication of this protocol is the following. If the shortest distance from S to R is $\Omega(|V|)$, then we achieve perfect secrecy for *free*. Because any reliable but insecure routing protocol would also takes $\mathcal{O}(|V|)$ rounds and send $\mathcal{O}(|V|)$ messages (one message along each edge in the shortest path) for transmission.

3.7.2 An example of the round optimal protocol with linear communication complexity

In this section, we illustrate the protocol $\mathbf{\Pi^{Static}_{Rnd\_Opt\_Lin}}$ with an example. Consider the graph G given in Figure 3.8. In earlier section, we have seen that the minimum number of rounds required for `PSMT` possibility in G is four. Accordingly, we execute the protocol $\mathbf{\Pi^{Static}_{Rnd\_Opt\_Lin}}$ on communication graph $\mathcal{G}^{(3)}$ of the graph $G^{(4)}$ given in Figure 3.14. We represent $\mathcal{G}^{(3)}$ with three vertex-disjoint weak paths and a Reverse Directed Rooted Tree $T_{G^{(4)}}$ rooted at R in Figure 3.15. Furthermore, a value sent by a node u to a node v as per the protocol code, is depicted over an edge $(u, v) \in E$. The first round computation of the protocol, that is, sharing of random numbers and calculating corresponding *values* is depicted at the top of the Figure 3.15 using three disjoint weak paths. Whereas, the computations of the second and subsequent rounds are depicted at the bottom using the tree $T_{G^{(4)}}$. The formal protocol is presented below.

An execution of the protocol $\mathbf{\Pi^{Static}_{Rnd\_Opt\_Lin}}$ on $\mathcal{G}^{(3)}$

At the beginning of the first round:

1. The sender S chooses two random numbers r_{S_1}, r_{S_2} and initializes $r_{S_3} = m - (r_{S_1} + r_{S_2})$.
2. Every node v, except S and R, chooses a random number r_v.
3. The receiver R chooses three random numbers $r_{R_1}, r_{R_2}, r_{R_3}$.
4. The node v_1 sends r_{v_1} to the node v_2 and the sender S.
5. The node v_2 sends r_{v_2} to the receiver R.

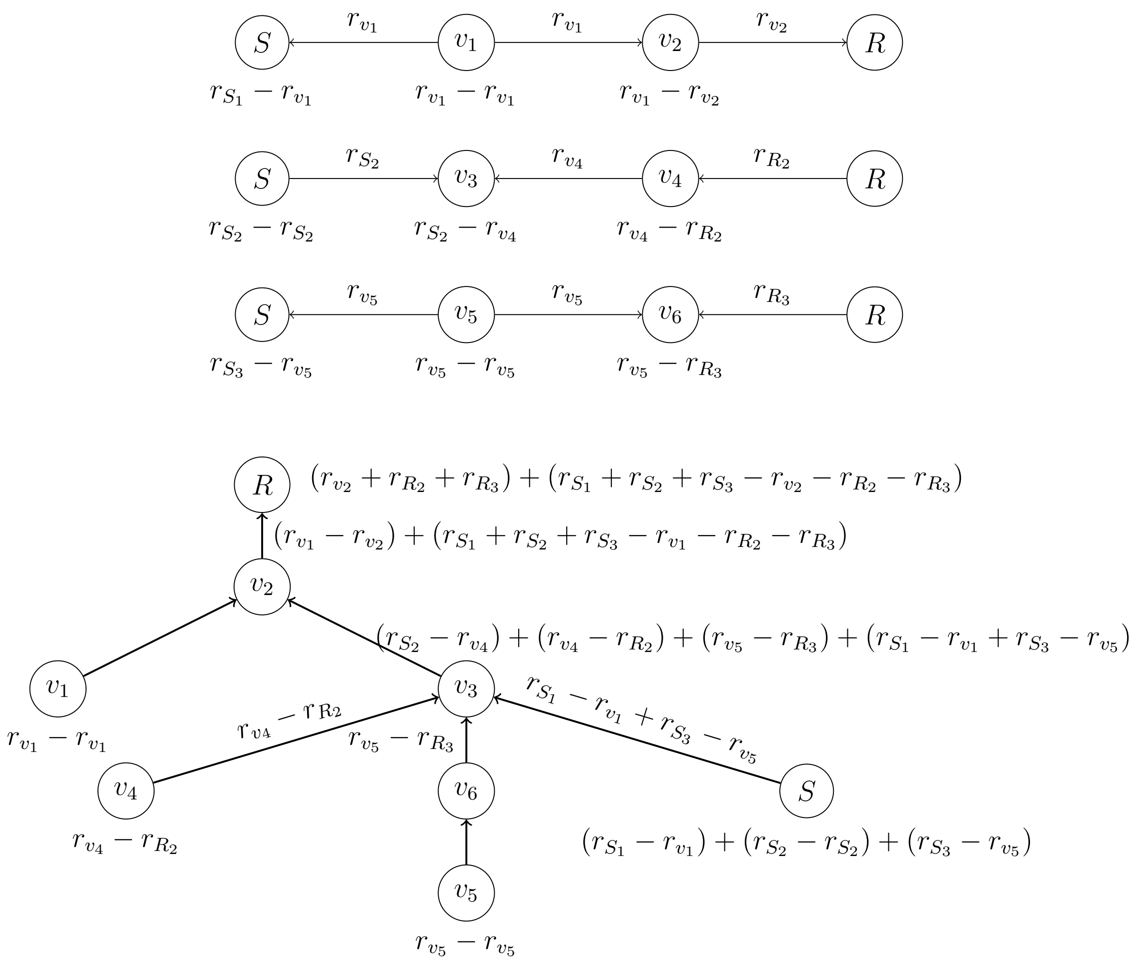

Figure 3.15: An execution of the protocol $\mathbf{\Pi}^{\textbf{Static}}_{\textbf{Rnd\_Opt\_Lin}}$ on $\mathcal{G}^{(3)}$ of the graph $G^{(4)}$ given in Figure 3.14

6. The node S sends r_{S_2} to the node v_3.

7. The node v_4 sends r_{v_4} to the node v_3.

8. The receiver R sends r_{R_2} to the node v_4.

9. The node v_5 sends r_{v_5} to the node v_6 and the sender S.

10. The receiver R sends r_{R_3} to the node v_6.

At the end of the first round:

1. Every node v, calculates its *value* $Val[v]$:

   (a) $Val[S] = (r_{S_1} - r_{v_1}) + (r_{S_2} - r_{S_2}) + (r_{S_3} - r_{v_5}) = r_{S_1} - r_{v_1} + r_{S_3} - r_{v_5}$.

   (b) $Val[v_1] = r_{v_1} - r_{v_1}$, $Val[v_2] = r_{v_1} - r_{v_2}$ and $Val[v_3] = r_{S_2} - r_{v_4}$.

   (c) $Val[v_4] = r_{v_4} - r_{R_2}$, $Val[v_5] = r_{v_5} - r_{v_5}$ and $Val[v_6] = r_{v_5} - r_{R_3}$

At the beginning of the second round:

1. The node v_4 sends $Val[v_4] = r_{v_4} - r_{R_2}$ to the node v_3.

2. The node v_6 sends $Val[v_6] = r_{v_5} - r_{R_3}$ to the node v_3.

3. The sender S sends $Val[S] = r_{S_1} - r_{v_1} + r_{S_3} - r_{v_5}$ to the node v_3.

At the end of the second round:

1. The node v_3 calculates: $Sum(v_3) = Val[v_3] + Val[v_4] + Val[v_6] + Val[S] = (r_{S_2} - r_{v_4}) + (r_{v_4} - r_{R_2}) + (r_{v_5} - r_{R_3}) + (r_{S_1} - r_{v_1} + r_{S_3} - r_{v_5}) = r_{S_2} - r_{R_2} - r_{R_3} + r_{S_1} - r_{v_1} + r_{S_3}$.

At the beginning of the third round:

1. The node v_3 sends $Sum(v_3) = r_{S_1} + r_{S_2} + r_{S_3} - r_{v_1} - r_{R_2} - r_{R_3}$ to the node v_2.

At the end of the third round:

1. The node v_2 calculates $Sum(v_2) = Val[v_2] + Sum(v_3) = (r_{v_1} - r_{v_2}) + (r_{S_1} + r_{S_2} + r_{S_3} - r_{v_1} - r_{R_2} - r_{R_3})$.

At the beginning of the fourth round:

1. The node v_2 sends $Sum(v_2) = r_{S_1} + r_{S_2} + r_{S_3} - r_{v_2} - r_{R_2} - r_{R_3}$ to the receiver R.

At the end of the fourth round:

1. The receiver R calculates the message $m = (r_{v_2} + r_{R_2} + r_{R_3}) + Sum(v_2) = (r_{v_2} + r_{R_2} + r_{R_3}) + (r_{S_1} + r_{S_2} + r_{S_3} - r_{v_2} - r_{R_2} - r_{R_3}) = r_{S_1} + r_{S_2} + r_{S_3}$.

3.8 PSMT tolerating mobile adversary

We have been considering the static adversary so far. In static adversary, a node once corrupted remains corrupted subsequently. Thus, the static adversary can corrupt only one fixed set of t-nodes throughout the protocol execution. Here, we relax this requirement. That is, the adversary can corrupt any set of t nodes (of its choice) in any round. In other words, in different rounds, the adversary can corrupt different set of t nodes.

For fast protocols, the adversary may be assumed to be *static*, that is, the same set of nodes are corrupt (in every round) throughout the protocol execution. However, for protocols that last long, a more suitable model is that of a *mobile* adversary which corrupts different set of t nodes in different rounds (catering to an equilibrium between (a) curing/replacing faulty machines and (b) breaking-in to new machines during the protocol execution). Evidently, protocols tolerating mobile t-adversary are likely to be far more cumbersome and complex than the ones tolerating static t-adversaries. Counter-intuitively, we show that protocols for perfectly secret message transmission (`PSMT`) can withstand adversarial mobility for *free*. Specifically, for `PSMT` in any directed graph influenced by a passive/eavesdropping adversary, we show that: (a) *adversarial mobility does not affect its tolerability:* `PSMT` tolerating a static t-adversary is possible *if and only if* `PSMT` tolerating mobile t-adversary is possible; (b) *mobility does not affect the round optimality:* the fastest `PSMT` protocol tolerating static t-adversary is *not faster* than the fastest one tolerating mobile t-adversary; and (c) *mobility does not affect communication complexity too:* we design `PSMT` protocols that have linear communication complexity in both static as well as mobile adversarial settings.

We notice that, if nodes cannot *wipe/delete* the data from their memory, then given sufficient time (rounds) the adversary can eavesdrop the required number of nodes to get the secret (in the worst case the adversary can eavesdrop each node of the network after certain time units though the protocol might have terminated). Therefore, it is necessary for the nodes to have the *data deletion* capability to tolerate the mobile adversary. Also, we assume that once data is deleted it cannot be recovered by any means. We use the notation DEL[U] to denote the *deletion* of every element from the subset U of field F.

We now present the intuition behind the protocol $\mathbf{\Pi^{Mobile}_{Rnd\_Opt\_Lin}}$, which tolerates the mobile t-adversary if PSMT is possible tolerating the static t-adversary. If there exist $t+1$ disjoint weak paths from S to R then at least one of them, say p, is honest in the first round. We make sure that, before the adversary corrupts any node from the weak path p in subsequent rounds, each pair of adjacent nodes of p exchange information, which eventually guarantees PSMT from S to R tolerating mobile adversary. Once adjacent nodes are done with exchanging information, each node u: (1) locally computes a function f on its local information (2) stores the output of f and (3) completely *deletes* the local information. The function f has the property that looking at the output of the function the adversary learns nothing about the corresponding inputs. More precisely, let u be a node from the uncorrupted weak path p. Let us consider the following two cases.

Case 1: Suppose node u is an in-neighbour of R (i.e., $(u, R) \in E$) and wants to send the message m to R. u simply forwards the message m to R and *deletes* m from its memory. The receiver R gets the message by the end of the round but the adversary learns nothing about the message even if it eavesdrops the node u in subsequent rounds as the message is deleted by node u.

Case 2: Suppose u is not directly connected to R but u is an out-neighbour of R (i.e., $(R, u) \in E$) and the node u wishes to send a message m to R. The protocol works as follows: R sends a random key K_R to u at the beginning of the first round and by the end of the first round u receives K_R. Now, the node u calculates $m - K_R$ and *deletes* both m and K_R from its memory. Observe that by corrupting node u in any subsequent rounds, the adversary gets $m - K_R$ which reveals nothing about either m or K_R. Therefore, in subsequent rounds node u can send $m - K_R$ to R using any path, which may be eavesdropped by the adversary. Once R receives $m - K_R$, R adds K_R to $m - K_R$ to get message m.

We use this simple idea to design the protocol $\mathbf{\Pi^{Mobile}_{Rnd\_Opt\_Lin}}$ to tolerate the mobile t-adversary. This protocol is same as the protocol $\mathbf{\Pi^{Static}_{Rnd\_Opt\_Lin}}$ except that at the end of the first round each node deletes its *Left Value* and *Right Value* after calculating its *value*.

Consider a communication graph $\mathcal{G}^{(t+1)}$ of the digraph $G^{(r)}(V,E)$, where r be the optimal number of rounds required for PSMT possibility in G tolerating t-threshold static adversary. Let $(t+1)$ vertex-disjoint weak paths of $\mathcal{G}^{(t+1)}$ are, namely $p_i : \langle u_{i0}(= S), u_{i1}, \ldots, u_{ik_i}, u_{i(k_i+1)}(= R)\rangle$, for each $i \in [1, t+1]$. And, the height of a Reverse Directed Rooted Tree $T_{G^{(r)}}$ of $G^{(r)}$ be h with root R is at the 0^{th} level. The protocol code is as follows.

3.8.1 The Protocol $\Pi^{\textbf{Mobile}}_{\textbf{Rnd\_Opt\_Lin}}$

At the beginning of the first round:

1. Each node u_{ij}, except $u_{(t+1)0}$, picks a random number $r_{ij} \in \mathbb{F}$, for every $i \in [1, t+1]$ and $j \in [0, k_i + 1]$.
2. $S(= u_{(t+1)0})$ computes $r_{(t+1)0} = m - \sum_{i=1}^{t} r_{i0}$.
3. For each $i \in [1, t+1]$: $S(= u_{i0})$ initializes $L[u_{i0}] = r_{i0}$.
4. For each $i \in [1, t+1]$ and $j \in [0, k_i]$:
   (a) if $(u_{ij}, u_{i(j+1)}) \in E^{(r)}$, then u_{ij} sends r_{ij} to $u_{i(j+1)}$ and initializes $R[u_{ij}] = r_{ij}$.
   (b) if $(u_{ij}, u_{i(j+1)}) \notin E^{(r)}$, then $u_{i(j+1)}$ sends $r_{i(j+1)}$ to u_{ij} and initializes $L[u_{i(j+1)}] = r_{i(j+1)}$.

At the end of the first round:

1. For each $i \in [1, t+1]$ and $j \in [0, k_i]$:
   (a) if $(u_{ij}, u_{i(j+1)}) \in E^{(r)}$, then $u_{i(j+1)}$ receives r_{ij} from u_{ij} sent at the beginning of the first round and initializes $L[u_{i(j+1)}] = r_{ij}$.
   (b) if $(u_{ij}, u_{i(j+1)}) \notin E^{(r)}$, then u_{ij} receives $r_{i(j+1)}$ from $u_{i(j+1)}$ sent at the beginning of the first round and initializes $R[u_{ij}] = r_{i(j+1)}$.
   (c) each node u_{ij} calculates its *value*, $Val[u_{ij}] = L[u_{ij}] - R[u_{ij}]$.
   (d) every node u_{ij} performs the deletion operation, $\text{DEL}[\{L[u_{ij}], R[u_{ij}]\}]$.
2. The sender S computes its *grand value*, $Val[S] = \sum_{i=1}^{t+1} Val[u_{i0}]$.

Second round onwards:

1. `If` S is a leaf node (at level h) in $T_{G^{(r)}}$ and its *grand value*, $Val[S]$, is non-zero then S sends $Val[S]$ to its parent at $(h-1)^{th}$ level.

2. `Else`, if S is at k^{th} ($k \in [1, h-1]$) level then S receives values from its children at $(k+1)^{th}$ level. Subsequently, S *adds* all the received values to its *grand value* $Val[S]$ and sends to its parent which is at $(k-1)^{th}$ level.

3. For each $i \in [1, t+1]$ and $j \in [1, k_i]$:

   (a) `If` u_{ij} is a leaf node (at level h) in $T_{G^{(r)}}$ and its *value*, $Val[u_{ij}]$, is non-zero then u_{ij} sends $Val[u_{ij}]$ to its parent at $(h-1)^{th}$ level.

   (b) `Else`, each u_{ij} which is at k^{th} ($k \in [1, h-1]$) level receives values from its children at $(k+1)^{th}$ level. Subsequently, u_{ij} *adds* all the received values to its *value* $Val[u_{ij}]$ and sends to its parent which is at $(k-1)^{th}$ level.

4. In the last round, the receiver R adds the sum of all the values it received from its children with the sum of all its *Left Values* (i.e. $\sum_{i=1}^{t+1} L[u_{i(k_i+1)}]$) to get the message m.

Theorem 3.18. *The protocol* $\mathbf{\Pi^{Mobile}_{Rnd\_Opt\_Lin}}$ *is reliable and secure.*

Proof. **Reliability**: The protocol $\mathbf{\Pi^{Mobile}_{Rnd\_Opt\_Lin}}$ is same as the protocol $\mathbf{\Pi^{Static}_{Rnd\_Opt\_Lin}}$, except that, each node u_{ij}, after computing $Val[u_{ij}]$ performs an extra *delete* operation, $\mathrm{DEL}[\{L[u_{ij}], R[u_{ij}]\}]$. However, in the protocol $\mathbf{\Pi^{Static}_{Rnd\_Opt\_Lin}}$, each individual $L[u_{ij}]$ and $R[u_{ij}]$ is not necessary for R to reconstruct the message. Therefore, reliability is guaranteed.
Security: We know that the adversary cannot corrupt each of the $t+1$ weak paths in the first round. Let p_i be one such uncorrupted weak path in the first round. Notice that, by the end of the first round, each node u_{ij} of p_i calculates its *value*, $Val[u_{ij}] = L[u_{ij}] - R[u_{ij}]$ and performs the delete operation, $\mathrm{DEL}[\{L[u_{ij}], R[u_{ij}]\}]$. Therefore by corrupting p_i in subsequent rounds, the adversary gets $Val[u_{ij}] = L[u_{ij}] - R[u_{ij}]$ but nothing about $\{L[u_{ij}], R[u_{ij}]\}$. In **Theorem 3.17**, we already showed that even if the adversary gets $Val[u_{ij}]$ for each u_{ij}, it learns nothing about r_{i0} and thus nothing about the message m. □

3.9 Multicast

Although point to point transmission is common, there are numerous applications in which the same message needs to be delivered to many receivers. To encompass this natural generalization we define SECRET MULTICAST, from the sender S to the set of receivers $\hat{\mathbf{R}} = \{R_1, R_2, \ldots, R_k\}$, $k \geq 1$, as follows: The sender S wishes to secretly communicate a message m to each receiver $R_i \in \hat{\mathbf{R}}$ such that the adversary, who can passively corrupt up to t nodes other than sender S and any receiver in $\hat{\mathbf{R}}$, learns nothing about the message m.

The simple idea of achieving SECRET MULTICAST by doing separate `PSMT` from the sender to each of the receivers does not work. Suppose `PSMT` is not possible from the sender S to a single receiver R_1 in Graph G. After making node R_2 (of the same graph) the second receiver, SECRET MULTICAST from S to $\{R_1, R_2\}$ may become possible. For example, consider a graph in which, the sender S can securely send the message to the second receiver R_2, who in turn, can securely send it to the first receiver R_1. The main idea in characterizing multicast is realizing the fact that the receivers can never be corrupted by the adversary and hence can be used as intermediate senders. Now, we define the following notion to help us model this transitive communication.

Definition 3.4. *Let $V_1, V_2, \ldots, V_k$ and W be any subsets of V. We say that $V_1, V_2, \ldots, V_k$ are pair-wise disjoint* modulo *W if $V_i \cap V_j \subseteq W$ for every $i, j(\neq i) \in [1, k]$. By extending this definition to weak paths in G, we say that any k weak paths $p_1, p_2, \ldots, p_k$ are pair-wise vertex-disjoint* modulo *a set W, if $V(p_1), V(p_2), \ldots, V(p_k)$ are pair-wise disjoint* modulo *W, where $V(p_i)$ is the set of all vertices of the weak path p_i. In other words, no two distinct weak paths can share a node except the nodes from W.*

Theorem 3.19. *In a digraph G,* SECRET MULTICAST *from S to $\hat{\mathbf{R}}$ tolerating up to t passive faults is possible if and only if at least one of the following two conditions hold for each $R_i \in \hat{\mathbf{R}}$:*

1. *There exists a path from S to R_i containing nodes only from $\hat{\mathbf{R}} \cup \{S\}$.*

2. *There exist at least $t + 1$ vertex-disjoint weak paths modulo $\hat{\mathbf{R}} \cup \{S\}$ from S to R_i such that each node on these weak paths must have a path to R_i in G.*

Proof. **Necessity**: Consider a weak path p from S to R_i such that some node u of p has no path to R_i in G. Then clearly, the sender S can never convey any information to R_i along p. At best, node u may receive message from the sender S but would not be able to forward to the receiver R_i, making the weak path p useless for S to R_i communication. Hence, we consider only the weak paths in which every node has a path to R_i. Let us assume on contrary that, for some $R_i \in \hat{\mathbf{R}}$, there exist only t vertex-disjoint weak paths modulo $\hat{\mathbf{R}} \cup \{S\}$ from S to R_i such that each node on these t weak paths has a path to R_i and there is no path containing only nodes from $\hat{\mathbf{R}} \cup \{S\}$. Then, there exist a vertex-cut of size t between S and R_i. Thus, by corrupting each node from vertex-cut, the adversary learns each piece of information exchanged between S and R_i. Therefore, the *view* of the adversary is the same as the *view* of the receiver.

Sufficiency: We give a *modified* `PSMT` protocol for secretly transmitting the message m to each R_i. Suppose, there exists a path p from S to R_i containing nodes only from $\hat{\mathbf{R}} \cup \{S\}$. Then, S simply forwards the message to R_i using the nodes on the path p. As all the nodes on p are honest, reliability and security are guaranteed.

Otherwise, assume that there exist at least $t+1$ vertex-disjoint weak paths modulo $\hat{\mathbf{R}} \cup \{S\}$ from S to R_i, namely p_i for each $i \in [1, t+1]$ such that each node on these $t+1$ weak paths has a path to R_i. The sender S simply runs the linear communication protocol $\mathbf{\Pi^{Mobile}_{Rnd\_Opt\_Lin}}$ given in section 3.8.1.

Notice that these $t+1$ weak paths are pairwise vertex-disjoint modulo $\hat{R} \cup \{S\}$. Therefore the nodes which are common to any of these weak paths must be from $\hat{R} \cup \{S\}$. Moreover, the adversary cannot corrupt the nodes from $\hat{R} \cup \{S\}$, which implies that for corrupting any two weak paths, the adversary must corrupt at least two nodes, one from each of the two paths. Therefore by corrupting t nodes, the adversary can corrupt maximum of t weak paths. Also, the protocol $\mathbf{\Pi^{Mobile}_{Rnd\_Opt\_Lin}}$ reveals nothing about the message m to the adversary which can corrupt up to t different weak paths in each round. Therefore, this *modified* `PSMT` protocol is both reliable and secure, completing the proof. □

3.9.1 Communication Complexity

Multicast can be achieved by executing *modified* `PSMT` protocol from S to each of the receivers. For this, we use the protocol $\mathbf{\Pi^{Mobile}_{Rnd\_Opt\_Lin}}$ from Section 3.8.1, which has $\mathcal{O}(|V|)$ communication complexity. As the number of receivers can be $\mathcal{O}(|V|)$, the communication complexity of our protocol becomes $\mathcal{O}(|V|^2)$. To show that this is asymptotically the best we can achieve, we present a graph which has $\Omega(|V|^2)$ critical edges. Consider the digraph $G(V, E)$ represented in Figure 3.16, which has vertex set $V = \{S, v_1, v_2, \ldots, v_n, R_1, R_2, \ldots, R_n\}$ and edge set $E = \Big\{\{S\} \times \{v_1, \ldots, v_n\}\Big\} \cup$

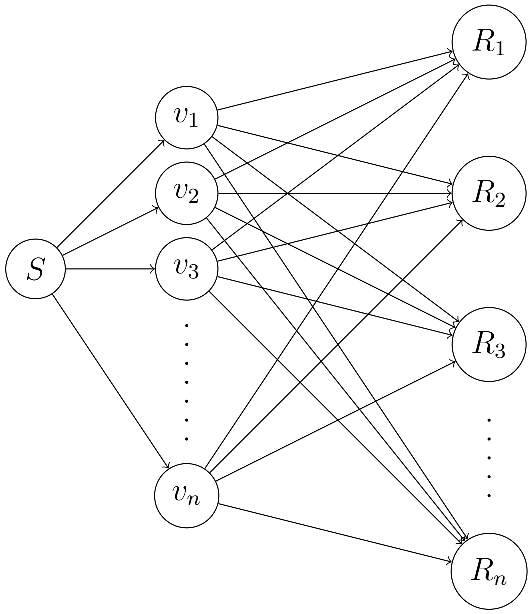

Figure 3.16: An example graph in which all the edges are critical for SECRET MULTICAST

$\Big\{\{v_1, \ldots, v_n\} \times \{R_1, \ldots, R_n\}\Big\}$. Here the sender S wishes to SECRET MULTICAST to $\hat{\mathbf{R}} = \{R_1, R_2, \ldots, R_n\}$ tolerating up to $n-1$ passive faults. As this graph satisfies the necessary conditions for **Theorem 3.19**, S to $\hat{\mathbf{R}}$ SECRET MULTICAST is possible, however, removing even a single edge makes it impossible.

3.9.2 Round Complexity

As the *modified* `PSMT` protocol for each of the receivers can be executed concurrently, the optimal protocol for multicast can be as fast as the slowest among these protocols. More formally, if r_i is the optimal number of rounds required for `PSMT` from S to R_i, then the optimal number of rounds for SECRET MULTICAST from S to $\hat{\mathbf{R}}$ is $r = Max\{r_1, r_2, \ldots, r_{|\hat{R}|}\}$.

3.10 Summary

In this chapter, we studied about characterizing networks and designing efficient protocols tolerating threshold passive adversary, in each of the four network models, namely, undirected graph model, wires model, routing model and arbitrary directed graph model. In undirected graph setting, we presented one of the simplest `PSMT` protocols exist in the literature. In wires, routing, and arbitrary directed graph models, we characterized networks in which PSMT is (im)possible and designed efficient protocols. Also, we designed a protocol that achieves PSMT in the least possible number of rounds. Moreover, without trading-off round efficiency, we designed a round optimal PSMT protocol whose communication complexity is linear. Furthermore, we showed that the designed round-optimal linear communication protocol tolerating t_p-threshold static adversary also achieves PSMT tolerating t_p-threshold mobile adversary neither affecting round optimality nor linear communication complexity. In the last section, we studied multicast and characterized networks in which multicast is (im)possible.

Chapter 4

SMT Tolerating Mixed Faults: Passive+Fail-Stop

4.1 Introduction

In this chapter, we consider a synchronous distributed network which is partially controlled by the mixed adversary. Here mixed adversary controls the network by corrupting up to t_p nodes passively in addition to corrupting up to t_f nodes in fail-stop fashion. Accordingly, we characterize the set of networks (under different network models) that admit PSMT protocols tolerating such mixed adversary. We assume that neither the sender nor the receiver can be corrupted by the adversary, otherwise secrecy is vacuously achieved. As mentioned earlier, every node in the network knows the complete topology of the network but no node has information about the part of the network which is compromised by the adversary. We also assume that S and R do not share any key a priori. Recall that, we use $[l, u]$ to denote the set $\{i \in \mathbb{Z} \mid l \leq i \leq u\}$.

4.2 PSMT in undirected graph model

By definition, fail-stop faults honestly follow the protocol code as long as they are alive. Moreover, in fail-stop corruption, the adversary can not read the data of the corrupted node but can crash it at its will. Therefore, it is enough to design reliable protocols as there is no issue of secrecy involved here. Also, it is easy to see that, $t_f + 1$ disjoint paths between S and R are necessary and sufficient for PRMT between S and R. If there are maximum of t_f disjoint paths then on corrupting one node from each of these t_f paths, the adversary cuts all the communication channels between S and R, thus make sure that no communication is possible between the sender and the receiver. Therefore, $t_f + 1$ disjoint paths are necessary for reliability. Suppose there are $t_f + 1$ disjoint paths between S and R then S simply sends the message to R along each of these $t_f + 1$ paths. As t_f be the maximum number of fail-stop faults can occur, the receiver is guaranteed to receive the message along at least one of the paths. Formally, the existing result in the literature is the following.

Theorem 4.1 ([60]). *In an undirected graph G,* `PSMT` *between S and R tolerating up to t_f fail-stop faults is possible if and only if there exist at least $t_f + 1$ vertex-disjoint paths between S and R in G.* □

In undirected graph setting, we now present the existing result for `PSMT` possibility tolerating up to t_f fail-stop faults and up to t_p passive faults.

Theorem 4.2 ([60]). *In an undirected graph G,* `PSMT` *tolerating up to t_f fail-stop faults and up to t_p passive faults is possible* `if and only if` *there exist at least $t_p + t_f + 1$ vertex-disjoint paths between S and R.*

Proof. **Necessity**: If there exist a maximum of $t_p + t_f$ disjoint paths between S and R then from Menger's theorem [38] we know that there exist a vertex-cut of size $t_p + t_f$ between S and R. Therefore, by corrupting each and every node from the vertex-cut, t_p of them passively and t_f of them in fail-stop fashion, the adversary learns everything that the receiver would learn from the information sent by the sender.

Sufficiency: If there exist $t_p + t_f + 1$ disjoint paths from S to R, namely W_i, for each $i \in [1, t_p + t_f + 1]$, then we can present a simple one-phase protocol $\mathbf{\Pi_{One-Phase}}$ as follows: S chooses a t_p-degree polynomial $p(x)$ such that the constant term $p(0)$ is the message m. Once polynomial is fixed, S sends point $p(i)$ to R along the path W_i, for each $i \in [1, t_p + t_f + 1]$. This protocol is reliable and secure since at most t_f paths can be corrupted in fail-stop fashion, the receiver R is guaranteed to receive at least $t_p + 1$ points. We know that $t_p + 1$ points are enough to reconstruct the t_p-degree polynomial uniquely [64]. Since the adversary can not eavesdrop on more than t_p points, the adversary learns nothing about message m as t_p or fewer points on t_p-degree polynomial reveal nothing about the message/constant term [64].

The protocol $\mathbf{\Pi_{One-Phase}}$ communication complexity is $\mathcal{O}(|V|)$. This is because each node in these disjoint paths only forwards the corresponding point to its out-neighbour. In any graph G, We can find whether there exist $t_p + t_f + 1$ disjoint paths or not in $\mathcal{O}(|V|^3)$ time using max-flow algorithms [67, 68]. Therefore, if there exist $t_p + t_f + 1$ paths between S and R then PSMT is guaranteed by the protocol $\mathbf{\Pi_{One-Phase}}$ which is efficient in both `communication complexity` and `testing the characterization`. We can further minimize the communication complexity if we use first t_p+t_f+1 shortest disjoint paths instead of arbitrary t_p+t_f+1 disjoint paths. □

4.3 PSMT in wires model

In this section, we abstract the network as a collection of disjoint wires where each wire is either a path from S to R or R to S and study the necessary and sufficient condition for PSMT possibility tolerating up to t_f fail-stop faults in addition to up to any t_p passive faults. In the literature, the collection of disjoint wires from S to R is known as `top band` and the collection of disjoint wires (disjoint from `top band`) from R to S is known as `bottom band`. The motivation(s) for studying this kind of abstraction is given in the works [14, 21, 29, 39, 40, 48].

If there exist $t_p + t_f + 1$ wires in `top band` then the simple protocol $\mathbf{\Pi_{One-Phase}}$ presented in earlier section achieves PSMT. Therefore, we give a necessary and sufficient condition for PSMT possibility when there exist at most $t_p + t_f$ wires in `top band`. We show that a total of $t_p + t_f + 1$ wires together in `top band` and `bottom band` are necessary and sufficient as long as `top band` contains at least $t_f + 1$ wires. Before presenting the theorem, we first illustrate the protocol with an example.

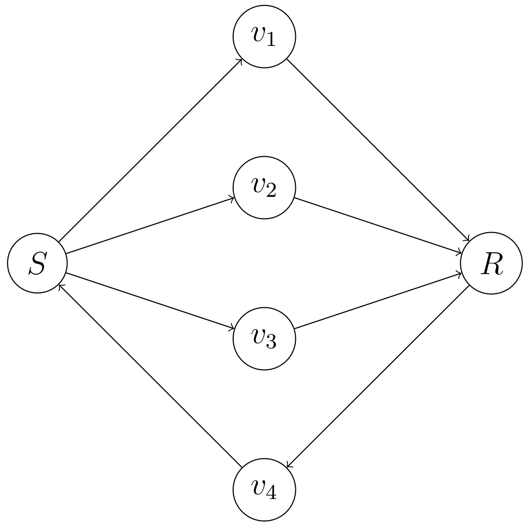

Figure 4.1: Graph G_1 with four wires.

Consider the graph G_1 given in Figure 4.1 which contains total of four wires out of which three are from S to R. Therefore PSMT protocols exist in graph G_2 tolerating maximum of two fail-stop faults and one passive fault. Let W_i be the wire containing the node v_i for each $i \in [1, 4]$. A two-phase protocol tolerating one passive fault and two fail-stop faults works as follows:

1. S starts with a random degree-1 polynomial $p(x)$ such that the constant term $p(0)$ is the message m.
2. S sends $p(i)$ to R along the wire W_i, $\forall i \in [1, 3]$.
3. R chooses a random number r_4 and sends it to S along the wire W_4.
4. If S receives r_4 then it sends $p(4) - r_4$ to R along each wire W_i, for $i \in [1, 3]$.

The above protocol is reliable and secure. We prove this in two cases depending on whether S receives r_4 or not.

Case(1): If S receives the random number r_4 then R definitely receives $p(4) - r_4$ as at least one v_i for some $i \in [1, 3]$ is non fail-stop faulty node. Upon receiving $p(4) - r_4$, R adds r_4 to $p(4) - r_4$ and gets $p(4)$. Also, R receives a point $p(i)$ along the wire W_i, for some $i \in [1, 3]$. We know that, these two points are enough to construct degree-1 polynomial uniquely. We get security from the fact that if v_i for $i \in [1, 3]$ is passively corrupted then the adversary gets one point $p(i)$ and one field element $p(4) - r_4$ but $p(4) - r_4$ reveals nothing about $p(4)$, since r_4 is unknown to the adversary.

Case(2): If S did not receive the random number r_4 then S knows that the wire W_4 is fail-stop faulty. And, as the node v_4 is already corrupted in fail-stop fashion, only one node, say v_i for some $i \in [1, 3]$, can be fail-stop faulty. Therefore, R may not receive the point $p(i)$ but R definitely receives the other two points. Rest follows as in Case(1).

The main idea is, even if there exist only $t_f + 1$ disjoint wires in `top band` and out of which the adversary corrupts t_f wires in fail-stop fashion and eavesdrops the remaining wire, we make sure that the adversary learns nothing about the message whereas the receiver receives the message. We achieve this as follows: Assume that there exist $t_f + k$ ($k \geq 1$) wires in `top band` and $t_p - k + 1$ wires in `bottom band`. First, S sends $t_f + k$ points to R along $t_f + k$ wires in top band, one point along one wire and, R sends $t_p - k + 1$ random numbers to S along $t_p - k + 1$ wires in bottom band, one random number along one wire. Upon receiving a random number r_i, S sends $p(i) - r_i$ to R along each wire in top band. As R knows the random number r_i, R gets $p(i)$ by simply adding r_i to $p(i) - r_i$. Moreover, R is guaranteed to receive $p(i) - r_i$, since the adversary can not crash every wire in top band. Therefore R receives at least $t_p + 1$ points on degree-t_p polynomial. It is clear that the adversary gets nothing about

$p(i)$ by knowing $p(i) - r_i$, unless the adversary knows r_i. The main theorem for PSMT possibility in the wires model is the following:

Theorem 4.3 ([42]). PSMT *is possible in* G if and only if *there exist at least* $t_f + k$ *wires in* top band *and* $t_p - k + 1$ *wires in* bottom band, *where* $k \geq 1$.

Proof. **Necessity**: Suppose there exist at most t_f wires in top band then the adversary can crash every wire in top band to make sure that no communication is possible from S to R. Thus, there should exist at least $t_f + k$ ($k \geq 1$) wires from S to R. Let us assume that there exist $t_f + k$ ($k \geq 1$) wires in top band but at most $t_p - k$ wires in bottom band. Then, the adversary corrupts up to t_f wires in fail-stop fashion from top band and each of the remaining wires (including wires in bottom band) in passive fashion. This implies that the adversary gets every piece of information exchanged between S and R. In other words, the *views* of the adversary and the receiver becomes identical. Therefore, $t_p - k + 1$ wires are necessary in bottom band.

Sufficiency: The protocol $\mathbf{\Pi_{One-Phase}}$ guarantees PSMT if $k \geq t_p + 1$, so we start with assuming that $k \leq t_p$ and there exist $t_f + k$ wires in top band, namely W_i for each $i \in [1, t_f + k]$ and $t_p - k + 1$ wires in bottom band, namely W_i for each $i \in [t_f + k + 1, t_p + t_f + 1]$. W.L.G let us assume that, from top band, the adversary choose to corrupt first T_f wires in fail-stop fashion, namely W_i for $i \in [1, T_f]$ and next T_p wires in passive fashion, namely W_i for $i \in [T_f + 1, T_f + T_p]$ and similarly from bottom band, the adversary choose to corrupt first B_f wires in fail-stop fashion, namely W_i for $i \in [t_f + k + 1, t_f + k + B_f]$ and next B_p wires in passive fashion, namely W_i for $i \in [t_f + k + B_f + 1, t_f + k + B_f + B_p]$. It is clear that $T_p + B_p \leq t_p$ and $T_f + B_f \leq t_f$. We now present a two-phase protocol $\mathbf{\Pi_{Two-Phase}}$ which ensures PSMT from S to R.

Phase 1: For each $i \in [t_f + k + 1, t_p + t_f + 1]$, R chooses a random number r_i and sends it to S along the wire W_i. And, S receives the random number r_i along the wire W_i, for each $i \in [t_f + k + B_f + 1, t_p + t_f + 1]$.
Phase 2: Once first phase is finished, S receives $t_p - k + 1 - B_f$ random numbers, one from the wire W_j, for each $j \in [t_f + k + B_f + 1, t_p + t_f + 1]$. And, S gets the identities of the B_f fail-stop faulty wires in bottom band. Then, S informs R the identities of these B_f fail-stop faulty wires. Also, S upon receiving a random number r_j from the wire W_j, computes $p(j) - r_j$ and sends it to R along each wire W_i, for $i \in [1, t_f + k - B_f]$. Furthermore, the sender S sends $t_f + k$ points to R, one point along the wire W_i, namely $p(i)$, for each $i \in [1, t_f + k]$.

In next two lemmas, we prove the reliability and secrecy of the protocol, which completes the proof. □

Lemma 4.1. *The protocol* $\mathbf{\Pi_{Two-Phase}}$ *is reliable.*

Proof. We show that, irrespective of the adversary strategy R always receives at least $t_p + 1$ points. Notice that, R receives the point $p(i)$ along the wire W_i, for each $i \in [T_f + 1, t_f + k]$ and R also receives $p(j) - r_j$, for each $j \in [t_f + k + B_f + 1, t_p + t_f + 1]$. Upon receiving $p(j) - r_j$, R calculates $p(j) = p(j) - r_j + r_j$ as R knows r_j. Therefore R receives, in total of $(t_f + k - T_f) + (t_p - k + 1 - B_f)$ points on t_p-degree polynomial. We have $T_f + B_f \leq t_f$, which implies $t_f - (T_f + B_f) \geq 0$, therefore $(t_f + k - T_f) + (t_p - k + 1 - B_f) = t_p + 1 + t_f - (T_f + B_f) \geq t_p + 1$. □

Lemma 4.2. *The protocol* $\mathbf{\Pi_{Two-Phase}}$ *is secure.*

Proof. We have, in a field $\langle \mathbb{F}, +, * \rangle$; for given $x, z \in \mathbb{F}$, $\exists$ unique $y \in \mathbb{F}$ such that $x - y = z$. Which implies, without knowing random number r_i, the adversary gets nothing about $p(i)$ by learning just $p(i) - r_i$. This ensures that the adversary gets no more than $T_p + B_p (\leq t_p)$ points. We complete the proof from the fact that t_p or fewer points reveal nothing about message/constant term of a degree-t_p polynomial [64]. □

The protocol $\mathbf{\Pi_{Two-Phase}}$ communication complexity is $\mathcal{O}(|V|^2)$. **Phase 1** communication complexity is the total number of nodes in the bottom band as each node just forwards a random number to its out-neighbour in the corresponding path. In **Phase 2**, each wire in top band carries at most $t_p - k + 1 - B_f$ field elements and a point. This implies no edge carries more than $(t_p - k + 1 - B_f) + 1$ field elements which is bounded by $|V|$. Therefore the communication complexity is $\mathcal{O}(|V|^2)$.

4.4 PSMT in routing model

In this section, we focus on `PSMT` tolerating up to t_f fail-stop faults in addition to t_p passive faults in routing model. Recall that, in routing model, we separately consider forward and feedback channels. And, a forward channel can share a vertex with a feedback channel – overlap is allowed, which is not the case in wires model.

Consider the graph G_2 given in Figure 4.2. We notice that G_2 fails to meet the conditions of the **Theorem 4.3** for the existence of a `PSMT` protocol tolerating one fail-stop fault in addition to one passive fault. This is because, there exist two disjoint forward channels, namely, $\langle S, u, R \rangle$ and $\langle S, v, w, R \rangle$. And, these two forward channels

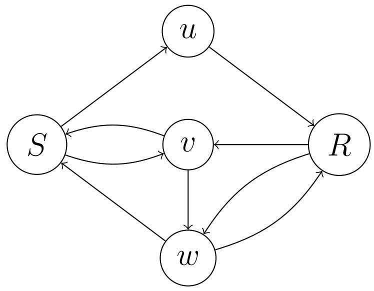

Figure 4.2: Graph G_2 with three disjoint channels.

exhaust all the nodes of G_2. Therefore, as required in the wires model, there exists no feedback channel which is disjoint with these two forward channels. However, if we allow overlap, there exist two disjoint forward channels, namely $p_1 : \langle S, u, R\rangle$ and $p_2 : \langle S, v, w, R\rangle$; and two disjoint feedback channels, namely, $\langle R, v, S\rangle$ and $\langle R, w, S\rangle$. And, as a Corollary to **Theorem 4.4**, we show that these channels are sufficient to exist a `PSMT` protocol in G_2 tolerating one fail-stop fault in addition to one passive fault in routing model. We now proceed to the main result. This result is analogous to the passive adversary case in the routing model.

Theorem 4.4. *In a directed graph G,* `PSMT` *is possible from S to R, tolerating up to t_f fail-stop faults in addition to up to t_p passive faults if and only if G satisfies the following two conditions:*

1. *There exist at least $t_f + 1$ disjoint forward channels - for reliability.*

2. *There exist at least $t_p + t_f + 1$ disjoint channels - for security.*

Proof. **Necessity**: If there are at most t_f `forward` channels, then communication itself is not possible from S to R. The necessity of the second condition follows from the fact that, if each channel is two-way connected then also $t_p + t_f + 1$ channels are necessary (see **Theorem 4.2**).

Sufficiency: Let us assume that the conditions of the theorem are satisfied. Also, let T and B denote the number of forward and feedback channels respectively, among the $t_p + t_f + 1$ disjoint channels. Like before, for each $i \in [T]$, S chooses a random number r_i and sends to R through the forward channel β_i. Similarly, for each $i \in [B]$, R chooses a random number r_{T+i} and sends it to S along the feedback channel β_{T+i}. We also observe, by first condition, that S can always reliably communicate with R. Now let us consider two cases: (1) $B \geq t_f + 1$ and (2) $B < t_f + 1$.

Case(1): If $B \geq t_f+1$, in this case, it is easy to see that the receiver can always *reliably* communicate with the sender without any error. The protocol works as follows:

1. Establishing the shared key: We have, $T + B \geq t_p + t_f + 1$, therefore, both the sender and receiver can share at least $t_p + 1$ random numbers. The only issue is, the sender (resp. receiver) may not know the exact set of random numbers received by the receiver (resp. sender) as crash failures can occur. Since reliable communication without any error is possible from S to R and vice-versa, both S and R inform each other, the *indices* of the channels/random numbers they received. Therefore, both S and R agree on the corresponding random numbers and hence the shared key k, which is the sum of the agreed random numbers. The shared key k is perfectly secure as at least one random number r_i is unknown to the adversary.

2. Transmitting the message: As there exist at least t_f+1 disjoint forward channels, S reliably sends the message $m'(= m + k)$ to R. Finally, R subtracts k from $m'(= m + k)$ and gets the message m. The message m is perfectly secure as the key k is perfectly secure.

Case(2): If $B < t_f + 1$, in this case, S and R try the above protocol as if $B \geq t_f + 1$. And, the above protocol fails only when all of the B feedback channels are corrupted in fail-stop fashion/crashed, since S may not know the exact random numbers received by R. And, hence can not establish the shared key. However, if that happens, S learns that all the feedback channels are crashed and S *reliably* informs the same to R. As a result, R also learns that each feedback channel is crashed. Subsequently, both S and R completely discards the previous information and runs the following new protocol.

Being each feedback channel is crashed, now at most $t_f - B$ fail-stop faults can occur in the rest of the graph. As we have at least $t_p + t_f - B + 1$ disjoint `forward` channels, S will generate a random degree-t_p polynomial, such that the constant term is the message m. And, S sends t_p+t_f-B+1 points to R, one along one distinct forward channel. We observe, of these, R will receive at least $t_p + 1$ points. The reliability and security comes from the fact that t_p or fewer points reveal nothing about the constant term - which is the message - and $t_p + 1$ or more points on a t_p-degree polynomial are enough to reconstruct the polynomial uniquely [64] □

Corollary 4.5. `PSMT` *is possible in the graph G_2 (given in the Figure 4.2) tolerating one passive and one fail-stop fault.*

Proof. The graph G_2 has two disjoint forward channels p_1, p_2 and three disjoint channels p_1, p_3, p_4 - thus meets the conditions of the above theorem tolerating one passive and one fail-stop fault. □

4.5 PSMT in arbitrary directed graph model

In this section, we model the network as an arbitrary directed graph and study the necessary and sufficient condition for `PSMT` possibility from S to R tolerating mixed adversary – up to t_f fail-stop faults in addition to up to any t_p passive faults. We assume that the network is synchronous and the adversary is static.

Recall that, in the absence of fail-stop faults (i.e., $t_f = 0$), the known result for `PSMT` possibility is the following:

Theorem 4.6 ([44]). `PSMT` *from S to R tolerating up to t_p passive faults in digraph G is possible* `if and only if` *there exist at least $t_p + 1$ vertex-disjoint weak paths from S to R, such that every node in these $t_p + 1$ weak paths must have a path to R.*

Remark on **Theorem 4.6**: Notice that, `PSMT` is not possible implies that at least one of the following two conditions must hold:

1. S to R perfect reliability is not possible.

2. S to R perfect reliability is possible but not perfect secrecy. That is, the receiver learns the message but the adversary also learns some information about the message.

   It is clear that, in the presence of only passive adversary, perfect reliability is always guaranteed as long as there is a path from S to R - the sender S simply sends the message along the corresponding path to the receiver R. As in latter case, perfect reliability is possible but not perfect secrecy implies that, the adversary gets each and every piece of information by eavesdropping each weak path between S and R. That is, the *views* of the receiver and the adversary are identical.

We have seen that, in passive case, each node should have at least one path to R, otherwise communication itself is not possible from those (particular) nodes to the receiver. Analogous to the passive case, at first glance one intuitively feels that each node must have $t_f + 1$ disjoint paths to R to communicate with R. However, in general it is sufficient but not necessary. This is due to the following reason. As the adversary can corrupt only one *fixed* set of (at most) t_f nodes of its choice, it may not be possible for the adversary to corrupt each and every path (from each and every node) to R. In other words, the actual set of t_f (fail-stop) corrupted nodes may not be omnipresent, thus each node need not have $t_f + 1$ disjoint paths to R to communicate with R. Consequently, we have the following result for mixed adversary case.

Theorem 4.7. *In a digraph $G(V, E)$,* `PSMT` *tolerating up to t_f fail-stop faults and up to t_p passive faults is possible* `if and only if` *for every subset F of V with cardinality t_f,* `PSMT` *tolerating up to t_p passive faults is possible in $G[V \setminus F]$.*

Proof. **Necessity**: On removal of some set $F(\subset V)$ with cardinality t_f from G, if `PSMT` is not possible in $G[V \setminus F]$ tolerating a set P of t_p passive faults then either perfect reliability is not possible in $G[V \setminus F]$ or perfect reliability is possible but not perfect secrecy as discussed in the above remark.

1. **Case(1)**: Suppose there is no path from S to R in $G[V \setminus F]$ then clearly perfect reliability itself is not possible in G if the adversary corrupts the nodes from the same set F in fail-stop fashion.

2. **Case(2)**: Suppose perfect reliability is possible but not perfect secrecy then the *view* of the adversary is identical with the *view* of the receiver in any protocol Π, if the adversary $\mathbb{A}$ corrupts the same set of t_f nodes of F in fail-stop fashion and the same set P of t_p nodes in passive fashion from $V \setminus F$ (see remark on **Theorem 4.6**). More precisely, graph G can be partitioned it to 4 sub graphs as depicted in Figure 4.3, where each edge from $G[U_1]$ to $G[U_2]$ represents the set of edges $E \cap (U_1 \times U_2)$.

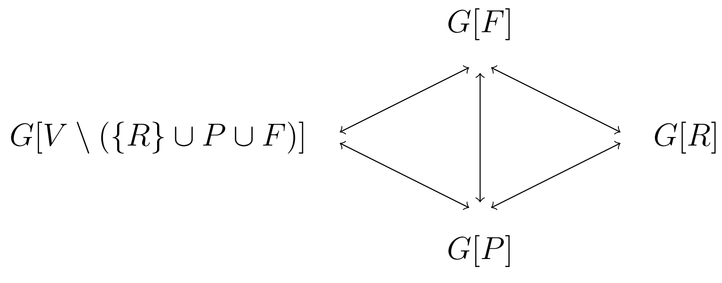

Figure 4.3: Partitioned Graph G.

Sufficiency: Notice that if the sender knows a priori the actual set $F(\subseteq V)$ of t_f fail-stop faults then the sender S can remove the same set F of nodes from the graph

G and can run a PSMT protocol for sending the message m from S to R, tolerating up to t_p passive faults in $G[V \setminus F]$. But the actual faulty information is not known to the sender except that up to t_f fail-stop faults and up to t_p passive faults can occur. The sender uses this knowledge and finds the possible sets of t_f fail-stop faulty nodes, namely F_i for each $i \in [1, \binom{n}{t_f}]$. Now the sender S will run $\binom{n}{t_f}$ **independent** PSMT protocols (chooses independent random polynomial for each sub protocol), each one sends the same message m from S to R tolerating up to t_p passive faults, namely Π_i in $G[V \setminus F_i]$ for each $i \in [1, \binom{n}{t_f}]$. This guarantees the secrecy as well as reliability because we know that only one set of t_f nodes can be corrupted by the adversary in fail-stop fashion, call it F_i. In fail-stop fault-free graph $G[V \setminus F_i]$, sub protocol Π_i ensures the reliability as well as secrecy tolerating up to t_p passive faults. Therefore the receiver is guaranteed to receive the message m by sub protocol Π_i. Also, we have every pair of PSMT sub protocols Π_j and Π_k are **independent** of each other which ensures that one protocol reveals nothing about the other protocol as each protocol uses an independent polynomial. In particular, no other protocol Π_j affects the secrecy of the protocol Π_i. Efficient PSMT protocol (if one exists) tolerating up to t_p passive faults in arbitrary/directed networks is given in [44]. The reason for running an exponential number of protocols is that when the adversary corrupts F_i, graph $G[V \setminus F_j]$ may or may not be fail-stop fault-free for $j \neq i$. If the graph $G[V \setminus F_j]$ actually contains a fail-stop faulty node say v then we can't guarantee reliability of message m by sub protocol Π_j even though it meets the sufficiency condition of **Theorem 4.6**, i.e. there exist $t_p + 1$ weak paths from S to R such that every node in these $t_p + 1$ weak paths has a path to R. This is because one of the nodes from these weak paths, say u may have only one path to R which passes through fail-stop faulty node v and node v may disconnect the communication from u to R. When v disconnects the communication from u to R, the weak path that the node u belongs to becomes obsolete. □

The protocol given in the sufficiency part of the proof of **Theorem 4.7** is a non-efficient communication protocol. This is because S is running an exponential number of sub protocols and the communication complexity of each sub protocol is $\mathcal{O}(|V|)$ [44]. **Remark** on **Theorem 4.7**: The **Theorem 4.7** statement is not equivalent to the following statement: On removal of any set P of t_p nodes from the graph $G(V, E)$, PSMT should be possible tolerating up to t_f fail-stop faults in $G[V \setminus P]$. In other words, there are graphs (say G is one of them) such that on removal of some set P of t_p nodes from G, the remaining graph $G[V \setminus P]$ fails to have $t_f + 1$ vertex-disjoint paths from S to R, however, protocols exist in G tolerating t_p passive faults and t_f fail-stop faults. For example consider the graph G given in Figure 4.4 which tolerates two passive faults and one fail-stop fault. This is evident because on removal of any one arbitrary node, the remaining graph G' will have three weak paths such that every node in these three weak paths has a path to R in G', which meets the sufficiency condition of **Theorem 4.7**. Now observe that on removal of two nodes v_1, v_2, the remaining graph has **only one** disjoint path from S to R namely either $\langle S, v_3, v_4, R \rangle$ or $\langle S, v_4, R \rangle$.

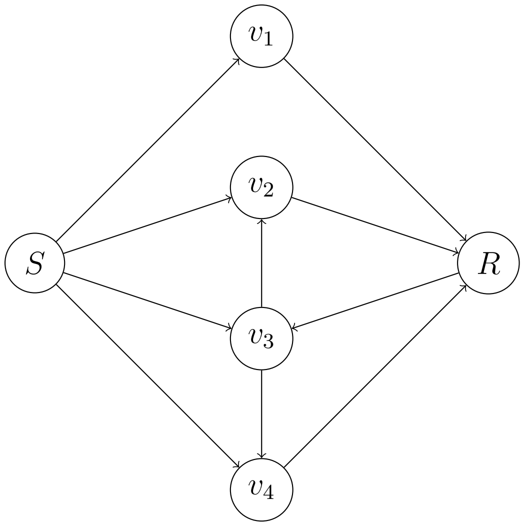

Figure 4.4: Graph G tolerating two passive faults and one fail-stop fault.

4.6 PSMT in special networks

In previous section, in arbitrary directed graph model, we presented a protocol that achieves *reliability* and *security* if the given network admits `PSMT` tolerating mixed adversary. However, the presented protocol requires exponential communication. In this section, we attempt to design an efficient `PSMT` protocol. Towards the same, we study a sufficiency condition for `PSMT` possibility in arbitrary directed graph setting. Subsequently, we present a communication efficient protocol for the class of networks that admit this particular sufficiency condition. Moreover, we show that every network that admits `PSMT` if we abstract it as in routing model also admits this particular sufficiency condition. Furthermore, we show that there exist networks which meet this particular sufficiency condition but fail to meet the sufficiency condition of routing model. These two together proves that this class of networks is more general than the class of networks modelled as a collection of disjoint wires.

Consider the Graph G_3 given in Figure 4.5. If we abstract the graph G_3 as a collection of wires then there exist only two wires in `top band` namely $\langle S, v_3, v_2, R\rangle$ and $\langle S, v_1, v_5, R\rangle$ and no wire exists in `bottom band`. There are wires from R to S but none of them is disjoint from the two wires in `top band`. Therefore according to **Theorem 4.4**, no protocol exists tolerating one passive fault and one fail-stop fault as we need at least three disjoint wires. However, we show that `PSMT` protocols exist in G_3 and one such protocol $\mathbf{\Pi_{G_3}}$ is presented below.

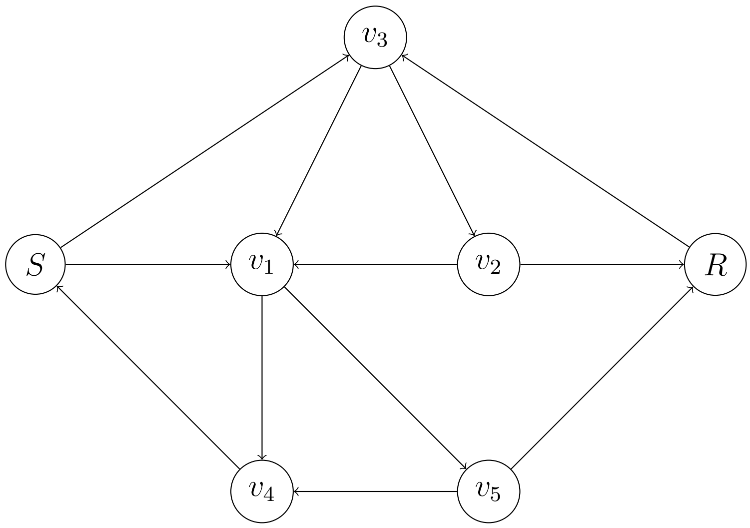

Figure 4.5: Graph G_3 tolerating up to one `passive fault` and up to one `fail-stop fault`.

Protocol $\mathbf{\Pi_{G_3}}$:

1. S chooses a random degree-1 polynomial $p(x)$ and sets $p(0) = m$ where m is the message that S wishes to send to R secretly.

2. Let $\mathcal{P}_{v_3} = \{\langle v_3, v_2, R\rangle, \langle v_3, v_1, v_5, R\rangle\}$ be a set of two disjoint paths from v_3 to R.

3. Let $\mathcal{P}_{v_1} = \{\langle v_1, v_5, R\rangle, \langle v_1, v_4, S, v_3, v_2, R\rangle\}$ be a set of two disjoint paths from v_1 to R.

4. R, v_2, v_5 chooses random numbers r, r_2, r_5 respectively.

5. R sends r to v_3, v_2 sends r_2 to both v_1 and R.

6. v_5 sends r_5 to both v_4 and R; v_4 forwards r_5 to S.

7. S sends $p(1)$, $p(2) - r_5$ to v_1 and sends $p(3)$, $p(2) - r_5$ to v_3.

8. v_3 sends $p(3) - r$ and $p(2) - r_5$ to R along both of the paths in $\mathcal{P}_{v_3}$.

9. v_1 sends $p(3) - r$ and $p(1) - r_2$ to R along both of the paths in $\mathcal{P}_{v_1}$.

The protocol $\mathbf{\Pi_{G_3}}$ is *reliable* and *secure*. We have three disjoint weak paths from S to R namely $p_1 : \langle S, v_3, R\rangle$, $p_2 : \langle S, v_1, v_2, R\rangle$ and $p_3 : \langle S, v_4, v_5, R\rangle$. Notice that, even if one of the nodes, say v_i, for some $i \in [1, 5]$, is corrupted in fail-stop fashion, the receiver receives at least two points. For example, if v_3 is failed, then both S and R receive r_5 from v_5. Also, R receives $p(2) - r_5$ sent by S along the path $\langle S, v_1, v_5, R\rangle$. Similarly, both v_1 and R receive r_2 from v_2 and R also receives $p(1) - r_2$ along the path $\langle v_1, v_5, R\rangle$. Therefore, R gets two points. The security argument follows same as in previous sections.

Definition 4.1. *Let $V_1, V_2, \ldots, V_k$ and W be any subsets of V. We say that $V_1, V_2, \ldots, V_k$ are pair-wise disjoint* `modulo` *W if $V_i \cap V_j \subseteq W$ for every $i, j (\neq i) \in [1, k]$. By extending this definition to weak paths in G, we say that any k weak paths $p_1, p_2, \ldots, p_k$ are pair-wise vertex-disjoint* `modulo` *weak path q if $V(p_1), V(p_2), \ldots, V(p_k)$ are pair-wise disjoint* `modulo` *$V(q)$. In other words, no two different weak paths can intersect out side the nodes of weak path q.*

Note: Every node v_i in a path $p : \langle S, v_1, v_2, \ldots, v_k, R\rangle$ trivially have $t_f + 1$ disjoint paths to R modulo p because we can count $t_f + 1$ times the same sub path $\langle v_i, v_{i+1}, \ldots, v_k, R\rangle$ of p from v_i to R. All these $t_f + 1$ sub paths are disjoint modulo path p.

Theorem 4.8. `PSMT` *from S to R is possible in G* ***`if`*** *there exist at least $t_p + t_f + 1$ disjoint weak paths from S to R in G, namely p_i for each $i \in [1, t_p + t_f + 1]$, such that from every node u in the weak path p_i to R there exist at least $t_f + 1$ disjoint paths* ***`modulo`*** *p_i.*

Proof. The following **protocol $\mathbf{\Pi_{Eff}}$** always guarantees `PSMT` from S to R as long as the graph meets the above said connectivity.

Protocol $\mathbf{\Pi_{Eff}}$:

1. S chooses a random t_p-degree polynomial $p(x)$ and replaces constant term $p(0)$ with m, where m is the message S wants to send to R secretly.

2. S will send $p(i)$ by simulating the corresponding path $p_i^{'}$ of weak path p_i, for

each $i \in [1, t_p + t_f + 1]$ using the **protocol Π_{Sim}** given below.

3. Once R receives $t_p + 1$ points on t_p-degree polynomial $p(x)$, reconstructs $p(x)$ and takes the constant term as the message.

Protocol Π_{Sim} :

Let $p_i : \langle S(= u_0), u_1, \ldots, u_f, u_{f+1}(= R)\rangle$ be a weak path in G and $p(i)$ be the point S wishes to send to R along the corresponding path $p_i^{'}$.

1. `if` weak path p_i is a path in G then S simply sends $p(i)$ to R along path p_i.

2. `otherwise` let $\{u_{l_1}, u_{l_2}, \ldots, u_{l_k}\}$ be the set of all nodes in the weak path p_i such that $(u_{l_j}, u_{l_j+1}) \notin E$ for $j \in [1, k]$ and without loss of generality assume that $l_m < l_n$ for $m < n$.

   (a) As $(u_{l_j}, u_{l_j+1}) \notin E$, we have (i) $(u_{l_j+1}, u_{l_j}) \in E$ and (ii) a subpath $p_{l_j+1} : \langle u_{l_j+1}, u_{l_j+2}, \ldots, u_{l_{(j+1)}}\rangle$ from u_{l_j+1} to $u_{l_{(j+1)}}$ containing only nodes of the weak path p_i.

   (b) u_{l_j+1} chooses a random number r_{l_j+1} and sends it to u_{l_j} along the edge (u_{l_j+1}, u_{l_j}) and sends it to $u_{l_{(j+1)}}$ along the path p_{l_j+1}.

   (c) $u_{l_j}(j \neq 1)$ calculates $r_{l_{(j-1)}+1} - r_{l_j+1}$ and sends it to R along any $t_f + 1$ disjoint paths `modulo` p_i.

3. S sends $p(i)$ to u_{l_1} along the path $\langle S(= u_0), u_1, \ldots, u_{l_1}\rangle$ and u_{l_1} sends $p(i) - r_{i_1+1}$ to R along (some) $t_f + 1$ disjoint paths `modulo` p_i.

4. `for` each $j = k-1, k-2, \ldots, 1$; the receiver R recursively computes $r_{l_j+1} = (r_{l_j+1} - r_{l_{(j+1)}+1}) + r_{l_{(j+1)}+1}$.

5. Once R gets r_{l_1+1} for $j = 1$, R finally computes $p(i) = (p(i) - r_{l_1+1}) + r_{l_1+1}$.

The correctness of the **protocol Π_{Eff}** is formally proved in **Theorem 4.10**. □

Before going to the **Theorem 4.10**, we prove a lemma which we use to prove the correctness of the **Protocol Π_{Sim}**. The correctness of the **Protocol Π_{Sim}** trivially guarantees the correctness of the **protocol Π_{Eff}** as **protocol Π_{Eff}** is nothing but executing the **Protocol Π_{Sim}** $t_p + t_f + 1$ times.

Lemma 4.3. *In a graph G, let u, v, w be any three nodes of a* `fail-stop fault-free` *weak path p such that there exist at least $t_f + 1$ disjoint paths* `modulo` *p from u to v, namely q_i for $i \in [1, t_f + 1]$ then the following two properties hold:*

1. *If* `Perfect Reliability` *is possible from w to u then* `Perfect Reliability` *is possible from w to v.*

2. *If u, v, w are honest and* `PSMT` *is possible from w to both u and v then* `PSMT` *is possible from u to v.*

Proof. PART(1): Since we have $t_f + 1$ disjoint paths `modulo` p from u to v, out of which at most t_f disjoint paths can be corrupted by the adversary in fail-stop fashion, clearly `Perfect Reliability` is possible from u to v as p is `fail-stop fault-free`. Therefore `Perfect Reliability` from w to u and u to v gives `Perfect Reliability` from w to v.
PART(2): Let m be the message that u wants to communicate to v secretly. As `PSMT` is possible from w to both u and v, w chooses a random number r and sends the same to both u and v secretly. Now u masks the message m with r as $m - r$ and sends to v using each path q_i. At most t_f paths can be corrupted by the adversary in fail-stop fashion and as p is `fail-stop fault-free`, there exists at least one reliable path q_j from u to v, for some $j \in [1, t_f + 1]$. Once v receives $m - r$ by path q_j, finally v unmasks the message m by adding r to $m - r$. This protocol is perfectly secure even if path q_j contains passive faulty nodes, since in a field $\langle \mathbb{F}, +, * \rangle$; for given $x, z \in \mathbb{F}$, $\exists$ unique $y \in \mathbb{F}$ such that $x - y = z$. □

Corollary 4.9. *The protocol $\mathbf{\Pi_{Sim}}$ for simulating the corresponding path p_i' of a weak path $p_i : \langle S(= u_0), u_1, \ldots, u_f, u_{f+1}(= R) \rangle$ holds the following two properties:*

1. *If weak path p_i is* `fail-stop fault-free` *then* `Perfect Reliability` *is possible from each node u_j of p_i to R, where $j \in [0, f]$.*

2. *If weak path p_i is* `fault-free` *then* `PSMT` *is possible from each node u_j of p_i to R, where $j \in [0, f]$.*

Proof. PART(1): Trivially follows from Lemma 4.3 if we replace $w = u = u_j$ and $v = R$.
PART(2): Since u_{l_k} is the last node in p_i such that edge $(u_{l_k}, u_{l_k+1}) \notin E$, we have (i) Secure edge $(u_{l_k+1}, u_{l_k}) \in E$ which implies `PSMT` is possible from u_{l_k+1} to u_{l_k} and (ii) `fault-free` path $\langle u_{l_k+1}, u_{l_k+2}, \ldots, u_f, u_{f+1}(= R) \rangle$ from u_{l_k+1} to R. Therefore `PSMT` is

possible from each u_j to R for $j \in [l_k + 1, f]$. In particular, PSMT is possible from u_{l_k+1} to R. We conclude that from Lemma 4.3, PSMT is possible from u_{l_k} to R. Now we show that PSMT is possible from each of the remaining nodes of p_i to R.

1. `for` $g = k-1, k-2, \ldots, 1$ we have:

   (a) `fault-free` path $p_{l_g+1} : \langle u_{l_g+1}, u_{l_g+2}, \ldots, u_{l_{(g+1)}} \rangle$ from u_{l_g+1} to $u_{l_{(g+1)}}$ containing only nodes of p_i. This implies PSMT is possible from u_{l_g+1} to $u_{l_{(g+1)}}$.

   (b) PSMT is possible from $u_{l_{(g+1)}}$ to R.

   (c) Above two steps together gives, PSMT is possible from u_j to R for every $j \in [l_g + 1, l_{(g+1)}]$.

   (d) Secure edge $(u_{l_g+1}, u_{l_g}) \in E$ which implies PSMT is possible from u_{l_g+1} to u_{l_g}.

   (e) The Lemma 4.3 and above two steps together proves PSMT is possible from u_{l_g} to R.

2. In particular, when $g = 1$ we get, PSMT is possible from u_{l_1} to R in G.

3. We have a `fault-free` path $\langle S(u_0), u_1, \ldots, u_{l_1} \rangle$ from S to u_{l_1} containing only nodes of p_i.

4. From the above two steps we get, PSMT is possible from u_j to R for every $j \in [0, l_1]$. In particular, PSMT is possible from S to R.

□

Theorem 4.10. *The protocol* $\mathbf{\Pi_{Eff}}$ *is a* PSMT *protocol.*

Proof. `Reliability:` We have $t_p + t_f + 1$ vertex-disjoint weak paths from S to R out of which at least $t_p + 1$ weak paths are `fail-stop fault-free`, with out loss of generality assume that these $t_p + 1$ weak paths are namely p_i for each $i \in [1, t_p + 1]$. Corollary 4.9 implies, each $u_{l_j} (j \neq 1)$ in p_i `reliably` sends $r_{l_{j-1}+1} - r_{l_j+1}$ to R and u_{l_1} `reliably` sends $p(i) - r_{l_1+1}$ to R. Therefore R `reliably` gets $p(i)$ by adding all these values to r_{l_k+1}.
`Secrecy:` The protocol $\mathbf{\Pi_{Eff}}$ is secure since there exists at least one `fault-free` weak path p_j out of these t_p+1 paths and Corollary 4.9 implies $p(j)$ is secure. On a t_p-degree polynomial, t_p or fewer points reveal nothing about constant term [64], which is the message m. □

Recall the following two results. We have shown that:

1. There are networks (Graph G_3 given in Figure 4.5 is one such example) that meet the above said **sufficiency** condition and hence admit PSMT, whereas PSMT is not possible in the same network (tolerating the corresponding/same adversary) if we abstract it as in **routing** model.

2. Every network that admits PSMT if we abstract it as in **wires** model also admits PSMT if we abstract it as in **routing** model but not the converse.

These two together imply that, there are networks (Graph G_3 given in Figure 4.5 is one such example) that meet the above said sufficiency condition and hence admit PSMT, whereas PSMT is not possible in the same network (tolerating the corresponding/same adversary) if we abstract it as in **wires** model.

Now we show that if PSMT is possible in a given network G which is abstracted as a collection of wires as in routing mode (or as in wires model) then G satisfies the **sufficiency** condition given in **Theorem 4.8** as well.

We know from **Theorem 4.4** that, there exist (1) at least $t_f + 1$ disjoint wires from S to R and (2) a total of at least $t_p + t_f + 1$ disjoint wires together from S to R and R to S. By Definition 4.1 (see the **Note** on Definition 4.1), it is clear that the wires from S to R not only just vertex-disjoint but also every node has $t_f + 1$ paths to R modulo the corresponding wire/path that the node belongs to. Let a node v belongs to a wire W from R to S. Since each wire from R to S is a path from R to S, v has a path to S, say p_v, which is a sub path of the wire W. And, the sub-path p_v from v to S followed by each wire from S to R gives at least $t_f + 1$ vertex-disjoint paths from v to R modulo W. Therefore, there exist at least $t_p + t_f + 1$ weak paths from S to R which meets the sufficiency condition of **Theorem 4.8**. The same argument holds for wires model condition as well.

4.7 Counter-intuitive examples

In this section, we study PSMT possibility in couple of arbitrary digraphs where the protocols are slightly counter-intuitive. We show that in arbitrary digraph setting, there exist networks that admit PSMT but fail to meet the PSMT sufficiency condition given in **Theorem 4.8**. In other words, there are directed graphs in which there exist only $t_p + t_f + 1$ weak paths and a node from one of these weak paths say p, fails to have $t_f + 1$ paths modulo p to R but PSMT protocol exists.

Consider the graph G_4 given in Figure 4.6 that has a maximum of three disjoint weak paths. Consider a set of three weak paths $p_1 : \langle S, v_1, R\rangle$, $p_2 : \langle S, v_4, R\rangle$ and

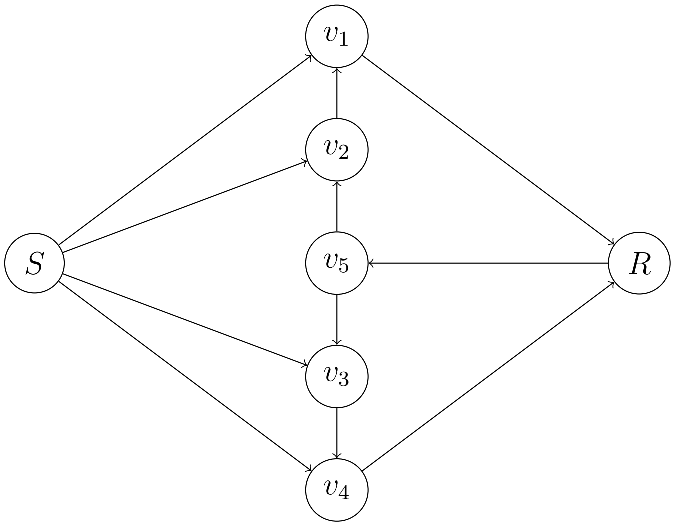

Figure 4.6: Graph G_4 with three weak paths tolerating up to one `passive fault` and one `fail-stop fault`.

$p_3 : \langle S, v_2, v_5, R\rangle$. A node v_2 in the weak path p_3 has only one vertex-disjoint path to R modulo p_3 which is $\langle v_2, v_1, R\rangle$. We always find one such node in any combination of three weak paths as avoiding both v_2 and v_3 it is impossible to have three disjoint weak paths and clearly there exists only one path to R from each v_2 and v_3. Therefore, it fails to satisfy the sufficiency condition of the **Theorem 4.8** tolerating one fail-stop fault and one passive fault but the following protocol $\mathbf{\Pi_{G_4}}$ achieves `PSMT` tolerating the same.

The protocol $\mathbf{\Pi_{G_4}}$:

1. The sender S chooses degree-1 polynomial $p(x)$ such that $p(0)$ is the message.
2. The sender S sends $p(i)$ to the node v_i for each $i \in [1, 4]$.
3. The receiver R sends two random numbers r_2 and r_3 to the node v_5.
4. The node v_5 sends r_2 to the node v_2 and r_3 to the node v_3.
5. The node v_2 sends $p(2) - r_2$ to the node v_1.
6. The node v_3 sends $p(3) - r_3$ to the node v_4.
7. The node v_1 sends $p(1)$ and $p(2) - r_2$ to the receiver R.
8. The node v_4 sends both $p(4)$ and $p(3) - r_3$ to the receiver R.

The protocol $\mathbf{\Pi_{G_4}}$ is reliable and secure. Case(1): If neither of v_1 and v_4 is fail-stop faulty then clearly R gets both $p(1)$ and $p(4)$ which are enough to reconstruct degree-1 polynomial. By passively corrupting any one node, as explained in previous section(s) the adversary gets a maximum of one point which guarantees security.
Case(2): If one of these v_1, v_2 is non fail-stop faulty node then the receiver at least gets either $p(1)$ and $p(2) - r_2$ or $p(3)$ and $p(4) - r_4$. In any case as the receiver R knows both r_2 and r_3, gets at least two points. Rest follows as in case(1).

We give yet another example where there exist more than $t_p + t_f + 1$ disjoint weak paths but no combination of t_p+t_f+1 paths satisfy the condition of the **Theorem 4.8** but `PSMT` protocols exist. For example, Graph G_5 (given in Figure 4.7) has a maximum of four disjoint weak paths, namely, $\langle S, v_1, R\rangle$, $\langle S, v_2, R\rangle$, $\langle S, v_3, R\rangle$ and $\langle S, v_4, R\rangle$. Avoiding both v_1 and v_2 (resp. v_3 and v_4) we can not get three vertex-disjoint weak paths and both v_1 and v_2 (resp. v_3 and v_4) has only one path to R. Therefore, G_5 fails to satisfy the condition of the **Theorem 4.8** tolerating one fail-stop fault and one passive fault. However, we can present a protocol similar to $\mathbf{\Pi_{G_4}}$ in G_5 as well tolerating one fail-stop fault and one passive fault.

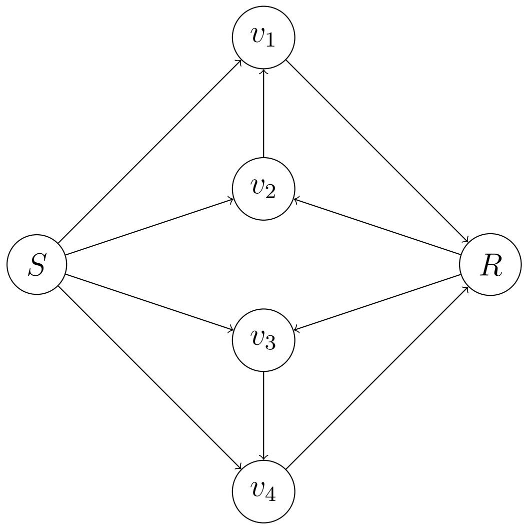

Figure 4.7: Graph G_5 with four weak paths tolerating up to one `passive fault` and one `fail-stop fault`.

4.8 PSMT between every pair

In this section we study a necessary and sufficient condition for `PSMT` possibility between every pair of nodes in arbitrary directed graph setting. Also, we design an efficient protocol that achieves PSMT if the given graph/network admits PSMT tolerating the mixed adversary – up to t_f fail-stop faults in addition to t_p passive faults. Furthermore, to test the characterization, all we need is, efficient algorithms to test whether the given graph is k connected or not for a given k. In the literature there are many algorithms ([67,68]) that check the same efficiently. Now, we move to the characterization, which is presented below.

Theorem 4.11. PSMT *is possible between every pair of nodes in a directed graph G* if and only if *it satisfies the following two conditions:*

1. PSMT *must be possible between every pair of nodes in the corresponding undirected graph G_u.*
2. *Directed graph G is $t_f + 1$ connected.*

Proof. **Necessity**: As G is a sub graph of G_u, if PSMT is not possible in G_u from u to v then clearly PSMT is not possible from u to v in G.
Suppose G is not $t_f + 1$ connected then there exists a pair of nodes u, v and a set F of t_f nodes such that nodes in F can disconnect u from v. Therefore, communication itself is not possible if the adversary corrupts each node of F in fail-stop fashion.

Sufficiency: Suppose PSMT is possible in G_u between every pair of nodes and G is $t_f + 1$ connected. We show that PSMT is possible between every pair in G. Let S and R be any two arbitrary nodes. Since PSMT is possible in G_u from S to R, there exist at least $t_p + t_f + 1$ vertex-disjoint paths from S to R in G_u [60]. Therefore, there exist at least $t_p + t_f + 1$ vertex-disjoint weak paths from S to R in G, since each path in G_u is a weak path in G. Also, we have G is $t_f + 1$ connected, this implies, from every node to every other node there exist at least $t_f + 1$ disjoint paths. In particular, every node in these weak paths has at least $t_f + 1$ disjoint paths to R in G. We already shown that, this connectivity is sufficient to exist a PSMT protocol from S to R in G and the proof is given in **Theorem 4.8**. This completes the proof. □

The communication complexity of the protocol given in **Theorem 4.8** is $\mathcal{O}(|V|^2)$. In the worst case we have to run $(|V|)(|V| - 1)$ independent protocols for all pair PSMT. Therefore the communication complexity is bounded by $\mathcal{O}(|V|^4)$.

4.9 Two Main Challenges in designing efficient protocols in arbitrary directed graph setting

In this section, we discuss two main challenges in designing efficient protocols and solve one of the two challenges completely. It is clear that there may exist exponential number of different (need not be disjoint) weak paths from S to R. In particular, there may exist exponential number of weak paths to R from each node in a given network G. We know that there exists a set of $t_p + 1$ fail-stop fault-free weak paths from S to R such that every node in these weak paths has a path to R avoiding the actual set F of t_f fail-stop faults if PSMT is possible in G (see **Theorem 4.7**).

- **Challenge #1**: How to find whether such $t_p + 1$ fail-stop fault-free weak paths exist or not ? If such paths exist then how to find them efficiently?
- **Challenge #2**: Each node v in a given weak path may have exponential number of different paths to R and only one of them can be fail-stop fault-free. How to use/choose that particular fail-stop fault-free path (from v to R), to get a communication efficient protocol once we find the $t_p + 1$ fail-stop fault-free weak paths?

We completely address the Challenge #2. Suppose if PSMT is possible in G and we get the $t_p + 1$ fail-stop fault-free weak paths, then we present a communication efficient protocol. Before presenting an efficient protocol, we design a protocol $\mathbf{\Pi_{Flood}}$ whose communication complexity is $\mathcal{O}(n^2)$. We use this protocol $\mathbf{\Pi_{Flood}}$ as sub-routine to the main protocol $\mathbf{\Pi_{Main}}$ which is communication efficient.

The Protocol $\mathbf{\Pi_{Flood}}$:

1. Let v be an arbitrary node in G and v wishes to communicate a field element x to R using the protocol $\mathbf{\Pi_{Flood}}$.
2. Let $Nhbd(v) = \{u \mid (v, u) \in E\}$ denotes the set of out neighbours of v.
3. Node v sends x to every out neighbour $u \in Nhbd(v)$.
4. Every $u \in Nhbd(v)$ forwards x to each $w \in Nhbd(u)$.
5. Each arbitrary node A, once it receives x, continues to forward x to every out neighbour B in $Nhbd(A)$ except that the node A earlier did not forward

x to B.

6. As network is synchronous, R waits till n time units once v sends x to R, where n is the number of nodes in digraph G.

The communication complexity of the protocol $\mathbf{\Pi_{Flood}}$ for sending a field element x to R is $\mathcal{O}(n^2)$. This is because, even if a node A receives x from multiple neighbours, it sends only once to its neighbours. Therefore in the worst case, every edge in the network carries x. We know that in a digraph G there are maximum of $\mathcal{O}(n^2)$ edges.

The **main property** of the protocol $\mathbf{\Pi_{Flood}}$ is, if a node v floods a field element x using the protocol $\mathbf{\Pi_{Flood}}$ then every node receives x if it has a fail-stop fault-free path from the node v in digraph G with in n times units since the network is synchronous.

Now we are ready to present the main protocol $\mathbf{\Pi_{Main}}$ and this protocol is similar to **Protocol** $\mathbf{\Pi_{Sim}}$ given in section 4.6. Let the $t_p + 1$ fail-stop fault-free weak paths are namely p_i for each $i \in [1, t_p + 1]$. Let $p_i : \langle S(= u_0), u_1, \ldots, u_f, u_{f+1}(= R)\rangle$ and $p(i)$ be the point S wishes to send to R along the corresponding path $p_i^{'}$.

The Protocol $\mathbf{\Pi_{Main}}$:

1. `if` the weak path p_i is a path in G then S simply sends $p(i)$ to R along the path p_i.

2. `otherwise` let $\{u_{l_1}, u_{l_2}, \ldots, u_{l_k}\}$ be the set of all nodes in the weak path p_i such that $(u_{l_j}, u_{l_j+1}) \notin E$ for $j \in [1, k]$ and without loss of generality assume that $l_m < l_n$ for $m < n$.

   (a) As $(u_{l_j}, u_{l_j+1}) \notin E$, we have (i) $(u_{l_j+1}, u_{l_j}) \in E$ and (ii) a subpath $p_{l_j+1} : \langle u_{l_j+1}, u_{l_j+2}, \ldots, u_{l_{(j+1)}}\rangle$ from u_{l_j+1} to $u_{l_{(j+1)}}$ containing only nodes of weak path p_i.

   (b) u_{l_j+1} chooses a random number r_{l_j+1} and sends it to u_{l_j} along the edge (u_{l_j+1}, u_{l_j}) and sends it to $u_{l_{(j+1)}}$ along the path p_{l_j+1}.

   (c) $u_{l_j}(j \neq 1)$ computes $r_{l_{(j-1)}+1} - r_{l_j+1}$ and sends it to R by using the protocol $\mathbf{\Pi_{Flood}}$.

3. S sends $p(i)$ to u_{l_1} along the path $\langle S(= u_0), u_1, \ldots, u_{l_1}\rangle$ and u_{l_1} sends $p(i) - r_{i_1+1}$ to R by using the protocol $\mathbf{\Pi_{Flood}}$.

4. `for` each $j = k-1, k-2, \ldots, 1$: the receiver R computes the value $r_{l_j+1} = (r_{l_j+1} - r_{l_{(j+1)}+1}) + r_{l_{(j+1)}+1}$.

5. Once R gets r_{l_1+1} for $j = 1$, R finally computes $p(i) = (p(i) - r_{l_1+1}) + r_{l_1+1}$.

Theorem 4.12. *The Protocol* $\mathbf{\Pi_{Main}}$ *is reliable and secure.*

We have t_p+1 fail-stop fault-free weak paths and each node in these weak paths has a fail-stop fault-free path to R. The main property of the protocol $\mathbf{\Pi_{Flood}}$ guaranties that if some node in these weak paths floods a field element then the receiver is guaranteed to receive the same. Therefore, reliability is trivial and the proofs are similar to that of the Corollary 4.9 and the **Theorem 4.10**. Also, the proofs of the Corollary 4.9 and **Theorem 4.10** implies the secrecy of the protocol $\mathbf{\Pi_{Main}}$. This is mainly because, at least one of the $t_p + 1$ fail-stop free weak paths is a fault-free say p_i, therefore the point $p(i)$ is secure from the adversary.

4.10 Summary

In this chapter, for each network model, we studied the trade-off between network connectivity requirement(s) and the existence of `PSMT` protocols tolerating mixed adversary (up to t_f nodes in addition to t_p nodes). In wires model setting and routing model setting, proposed protocols are communication efficient. Whereas, in arbitrary directed graph setting, proposed protocol is not communication efficient. It is interesting and worthy studying whether efficient protocols exist or not if a given arbitrary directed graph admits `PSMT`. Also, it is interesting to design efficient algorithms (if exist) to check whether the given graph has the required connectivity or not for `PRMT/PSMT` possibility.

Chapter 5

SMT Tolerating Omission Faults

5.1 Introduction

In this chapter, we study omission faults in detail. We assume that the network is synchronous and the adversary is threshold static, that can corrupt up to any t_o nodes in omission fashion. We consider undirected graph setting as well as directed graph setting. In directed graph setting, we study omission faults in wires model and routing model.

5.2 PSMT in undirected graph model

By definition, in omission corruption, the adversary can eavesdrop as well as crash the corrupted node at its will. Therefore, it is clear that, to tolerate up to t_o omission faults, $t_o + 1$ vertex-disjoint paths from S to R are necessary for reliability. Unlike in fail-stop corruption, the sender can not send the message in *plain* form, as the adversary can also eavesdrop the corrupted node. However, in undirected graph setting, we know that the reliability from S to R implies the reliability from R to S and vice-versa. Therefore, if there exist $t_o + 1$ vertex disjoint paths between S and R then both the sender and receiver can exchange any information *reliably*. And, in particular, if the sender (resp. receiver) knows some faulty information about any of the nodes (or paths) then it can inform the same to the receiver (resp. sender) reliably. The existing results used this idea to design `PSMT` protocols. More precisely, the sender and receiver run a `PSMT` protocol tolerating t_o passive faults, assuming that no fail-stop faults can occur in the network. Therefore, the receiver gets the message *securely* as long as no fail-stop faults occur during the course of the protocol. If protocol fails, then both the sender and receiver learn the information about the corrupted nodes (or paths); subsequently, both the sender and receiver eliminate the corresponding faulty nodes and/or paths. Upon eliminating known faulty nodes and/or paths, the sender repeats the above process, until the receiver gets the message securely. The existing result in literature is the following.

Theorem 5.1 ([60]). *In an undirected graph G,* PSMT *between S and R tolerating up to t_o omission faults is possible if and only if there exist at least $t_o + 1$ vertex-disjoint paths between S and R in G.*

Proof. **Necessity**: We have already seen that t_o+1 vertex-disjoint paths are necessary to tolerate t_o passive faults. Therefore, $t_o + 1$ vertex-disjoint paths are necessary to tolerate t_o omission faults as well, since the adversary can also eavesdrop the corrupted (in omission fashion) node.

Sufficiency: Assume that there exist at least $t_o + 1$ vertex-disjoint paths between S and R, namely p_i, for $i \in [1,\ t_o + 1]$. The sender starts with a random degree-t_o polynomial (assuming that no crash failures occur in the network) $p(x)$ such that its constant term $p(0)$ is the message m. The protocol works as follows. The sender S sends the point $p(i)$ along the path p_i to the receiver R, for each $i \in [1,\ t_o + 1]$. *If* R receives $t_o + 1$ points, then R reconstructs the polynomial and hence gets the message. *Else*, due to crash failures the receiver does not receive some of the points, say k_1 points. Then, the receiver *reliably* informs the identities of each of the k_1 crashed paths to the sender. Upon receiving information about the k_1 corrupted paths, S removes (from the network) each intermediate node of these k_1 paths (and hence removes k_1 faults from the network) and repeats the above process with a random polynomial of degree $t_o - k_1$, as the maximum number of faults can occur in the rest of the network is $t_o - k_1$. In the worst case, we have to repeat the above process t_o times, since in each run only one crash failure may occur. Therefore, R eventually gets the message *securely*. The reliability and security of the protocol directly follow from Shamir's secret sharing scheme. □

Intuitively we know that achieving PSMT tolerating t omission faults is harder than achieving PSMT tolerating t fail-stop faults. However, from the Theorem 4.1 and 5.1, we know that PSMT between S and R tolerating up to t faults is possible if and only if there exist at least $t+1$ vertex-disjoint paths between S and R, where all the t faults are of the same type, either omission or fail-stop. In other words, the connectivity requirement is same for tolerating t omission faults as well as t fail-stop faults. Then, the natural question arises is what parameter(s) differentiates these two problems. And, the answer is the following. We show that, even if *feedback* is not at all possible from R to S then also t_f+1 connectivity is sufficient for PSMT tolerating t_f fail-stop faults, whereas this is not the case with omission faults. In detail, the following result shows that one-phase PSMT protocol is not possible to design with t_o+1 connectivity tolerating t_o omission faults. In multi-phase protocols, the sender and receiver can interact with each other if required. Whereas in single phase protocols the sender sends enough information in one go to the receiver to reconstruct the message reliably as well as securely. Recall that, a phase is a send from the sender to the receiver or vice-versa.

Theorem 5.2 ([69]). *In an undirected graph G, single phase* PSMT *between S and R tolerating up to t_o omission faults is possible if and only if there exist at least $2t_o+1$ vertex-disjoint paths between S and R in G.*

Proof. **Necessity**: We notice that no set of t_o paths from S to R can contain full information about the message being transmitted as we execute the protocol in one-phase. Otherwise, by corrupting those particular t_o paths, the adversary can get the message without any error. Now, if there exist maximum of $2t_o$ vertex-disjoint paths between S and R then the adversary can crash any one set of t_o paths and hence the receiver receives messages from at most t_o paths which are uncorrupted. However, as the uncorrupted paths (maximum t_o paths) do not have full information about the message, the receiver can not reconstruct the message.

Sufficiency: If there exist $2t_o+1$ disjoint paths from S to R then S chooses a random degree-t_o polynomial $p(x)$ such that the constant term $p(0)$ is the message m. And, S sends $p(i)$ along i^{th} path to R. As the maximum number of faults can occur is t_o, the adversary can crash at most t_o paths. Hence, R gets at least t_o+1 points on degree-t_o polynomial, which are enough to reconstruct the polynomial. The security is guaranteed, because the adversary can learn at most t_o points on degree-t_o polynomial which reveal nothing about the message [64]. □

The above result clearly shows the power of interaction. The interaction between the sender and receiver greatly reduces the required connectivity, from $2t_o+1$ to t_o+1, for PSMT tolerating t_o omission faults.

5.3 PSMT in wires model

We have already seen that $t_o + 1$ disjoint wires from S to R are necessary for reliable communication from S to R tolerating t_o omission faults. And, if there is no path from R to S then from Theorem 5.2 we know that, $2t_o + 1$ disjoint wires from S to R are necessary and sufficient for `PSMT`. Moreover, it is easy to see that, along with $t_o + 1$ disjoint paths from S to R, if there exist $t_o + 1$ disjoint paths from R to S then `PSMT` is possible from S to R. This follows from the fact that, $t_o + 1$ disjoint wires (from R to S) guarantee reliable communication from R to S; hence, if any wire (from S to R) is failed to deliver information to R then R informs the same to S. Consequently, S removes each such faulty wire from the network. Protocols proposed for undirected graph model can be appropriately modified to work for this case.

The general case is when there are fewer than $2t_o + 1$ disjoint wires from S to R and fewer than $t_o + 1$ disjoint wires from R to S. The characterization is given in the following theorem for such networks.

Theorem 5.3. *Let $G(V, E)$ be a directed graph, $S, R \in V$, $1 \leq k \leq t_o$, and there are k vertex-disjoint paths from R to S. Then a necessary and sufficient condition for* `PSMT` *from S to R tolerating t_o omission faults is that there are $Maximum\{t_o+1, 2(t_o-k)+1\}$ vertex-disjoint paths (these paths are vertex-disjoint with those k paths from R to S) from S to R.*

Proof. **Necessity**: Let $Max = Maximum\{t_o + 1,\ 2(t_o - k) + 1\}$. If $Max = t_o + 1$ then the necessity is trivial to understand. Let us consider the other case, $Max = 2(t_o - k) + 1$. If there exist maximum of $Max - 1$ disjoint paths from S to R then the adversary has the following winning strategy. The adversary corrupts each of the k disjoint paths from R to S, and corrupts a set of $t_o - k$ disjoint paths (of its choice) from S to R. Since each of the paths from R to S is corrupted, the adversary makes sure that no communication (feedback) is possible from R to S. Therefore, one-phase `PSMT` should be possible from S to R tolerating $t_o - k$ faults. However, from Theorem 5.2 we know that $2(t_o - k) + 1$ vertex disjoint paths from S to R are necessary for one phase `PSMT` tolerating $t_o - k$ omission faults. Therefore, if there are maximum of $2(t_o - k)$ disjoint paths from S to R then `PSMT` is impossible from S to R.

Sufficiency: Let us assume that there are Max disjoint paths from S to R, namely p_i for each $i \in [Max]$, and k disjoint paths (which are disjoint with those Max paths) from R to S, namely, p_{Max+i} for each $i \in [k]$. The Sender S chooses a random degree-t_o

polynomial $p(x)$ such that its constant term $p(0)$ is the message m. And, S sends $p(i)$ along the path p_i to R, for $i \in [Max]$. The protocol works as follows.

1. If the receiver receives $t_o + 1$ points, then R reconstructs the polynomial $p(x)$ and gets the message. Also, R sends *Success* message to S along each of the k paths.

2. If the receiver did not receive $t_o + 1$ points, then R informs S along each of the k paths, the identities of each of the paths which are failed to deliver message/point.

3. If the sender receives neither *Success* message nor faulty information about the paths then S informs R that all the k paths are corrupted and initiates a one-phase protocol tolerating $t_o - k$ faults.

4. If the sender receives *Success* message from R then S acknowledges back R that it received *Success* message and consequently the protocol terminates.

5. If the sender receives information about faulty paths (say f paths failed to deliver) from R then S acknowledges back R that it received information about faulty paths and consequently both S and R discards those f paths from the network. And, S restarts the protocol with degree-($t_o - f$) polynomial.

6. The above protocol terminates when S receives either *Success* message or finds each of the t_o faulty paths.

The above protocol is secure since the sender restarts the protocol with $t_o - f$ degree polynomial only when S finds the faultiness of the corresponding f paths. And, we already know that t or fewer points on a random degree-t polynomial reveal nothing about the constant term, which is the message M [64]. □

5.4 PSMT in routing model

In this section, we study omission faults in routing model. We characterize networks that admits `PSMT` tolerating t_o omission faults. Also, we present an efficient protocol that achieves `PSMT` if the given network admits `PSMT`.

We know that an overlap is not allowed between channels in wires model. Whereas in routing model such an overlap is allowed. The main challenge in designing protocols tolerating omission faults in routing model (when such an overlap exist) is the following.

In case if there is no overlap between channels (as in wires model) then each time upon finding a faulty path, one can simply remove each intermediate node of the corresponding faulty path from the network to remove one faulty node (as paths from top band and bottom band are disjoint). However, this is not the case in routing model; as an overlap is allowed between channels, corrupted node may present in more than one path. Therefore, we do not afford to remove multiple paths from the network to remove one faulty path (or one faulty node). More precisely, consider the graph G given in 5.1, it has only one feedback channel, namely, $\langle R, v_1, v_0, S\rangle$. This feedback channel exhausts all the nodes from the network. Upon S learning that the feedback channel is corrupted, it just learns that either v_0 or v_1 is corrupted. However, S knows that information a priori.

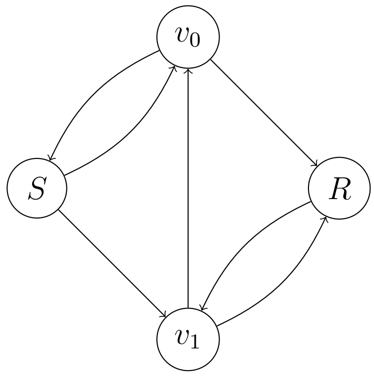

Figure 5.1: Graph G with only one feedback channel.

We have already seen that, if there is no feedback channel then $2t_o + 1$ vertex-disjoint paths are necessary for PSMT from S to R tolerating t_o omission faults. Let us consider the graph G given in Figure 5.1. Clearly, G do not meet the conditions of the **Theorem 5.3** tolerating one omission fault. However, we show that PSMT protocols exist in G tolerating one omission fault, and one such PSMT protocol Π_G is presented below.

The Protocol Π_G:

1. S sets $m = m_0 + m_1$ and it sends m_i to the node v_i, for $i \in [1, 2]$.
2. The node v_i forwards m_i to the receiver R.
3. If R receives both m_0 and m_1 then it gets the message by adding them.

4. Also, R constructs a binary vector (b_0, b_1), where R sets $b_i = 1$ if it receives m_i else it sets $b_i = 0$.

5. R sends the vector (b_0, b_1) to v_1 and v_1 forwards the same to v_0.

6. If v_0 does not receive (b_0, b_1) from v_1 then it informs S that the node v_1 is crashed.

7. If v_0 receives (b_0, b_1) then it forwards the same to S.

8. If v_0 receives (b_0, b_1) but it fails to forward the same to S then S learns that the node v_0 is crashed.

9. If S receives the vector (b_0, b_1) then it learns that R successfully received the message. Else it exactly finds the corrupted node.

10. Upon S learning the information about the corrupted node v_i, it sends the message in plain form (without encoding) to the receiver along the path $\langle S, v_{1-i}, R\rangle$.

The Protocol Π_G is reliable and secure since either R gets both m_1 and m_2, or S finds the faulty node. Once S finds the faulty node, it sends the message in plain form through the non-faulty path. Therefore, R gets the message securely.

Before proceeding to the main theorem we note few remarks, which are useful for designing an efficient PSMT protocol.

- **Remark #1**: In a network $G(V, E)$, assume that the adversary can corrupt up to any t_o nodes in omission fashion. If both the sender and receiver know (and agree on) a set O of k faulty nodes out of t_o faulty nodes then PSMT tolerating t_o omission is possible in G if and only if PSMT tolerating $t_o - k$ omission faults is possible in $G[V \setminus O]$. This directly follows from the fact that, once both S and R agree on a set O of k faulty nodes, they simply remove these k faulty nodes from the network G; and work in the rest of the graph $G[V \setminus O]$ as the maximum number of faults occur in $G[V \setminus O]$ is $t_o - k$.

- **Remark #2**: In a network G, in addition to the existence of $t_o + 1$ disjoint forward channels, if there exists at least one fail-stop fault-free feedback channel, then the following protocol guarantees PSMT from S to R. First, S starts with a random degree-t_o polynomial such that $p(0)$ is the message and sends t_o+1 points to R along $t_o + 1$ disjoint forward channels (each forward channel carries exactly one point). If R gets $t_o + 1$ points then it gets the message by reconstructing the polynomial. Further, R acknowledges S the receipt of the message; consequently protocol ends. In case, if any forward channel is failed to deliver information to R then R informs the same to S along the fail-stop fault-free feedback channel, that guarantees reliable communication from R to S. Consequently, S discards all such faulty forward channels (say k of them) from the network and repeats the protocol (with degree-$(t_o - k)$ polynomial) in the rest of the network (which has $t_o - k + 1$ disjoint forward channels) as the maximum number of faults can occur is $t_o - k$. In next phase, if any faults occur further, then R again informs the same to S using the same fail-stop fault-free feedback channel. Consequently, S discards them from the network and repeats the above process. In this way, S and R reliably communicate with each other till S learns all t_o faulty forward channels or R gets the message, whichever is earlier.

- **Remark #3**: Let $p : \langle v_0, v_1, v_2, \ldots, v_n, v_{n+1} \rangle$ be a path in a synchronous network $G(V, E)$. Assume that v_0 sends a message to v_{n+1} along p. If v_{n+1} does not receive the message due to any crash failure(s) then v_{n+1} finds the identity of at least one fail-stop faulty node present in p from the following protocol. First, v_0 sends the message to the node v_1, then the node v_i forwards the message to the node v_{i+1} if it receives from the node v_{i-1}, for each $i \in [1, n]$. If the node v_{i-1} does not forward the message to the node v_i then v_i informs v_{i+1} that v_{i-1} is crashed. Subsequently, for each $j \in [i+1, n]$, v_j informs v_{j+1} that the node v_{i-1} is crashed. Further, for some $j \in [i + 1, n]$, if a node v_j is crashed and failed to forward the faulty information to the node v_{j+1} then the node v_k informs v_{k+1} that the node v_j is crashed, for each $k \in [j + 1, n]$. In this process, v_{n+1} learns that the node v_{i_l} is crashed, where $\{v_{i_1}, v_{i_2}, v_{i_3}, \ldots, v_{i_l}\}$ be the set of all crash failures occurred in p such that $i_u < i_v$ for $u < v$. That is, v_{i_l} is the nearest node to v_{n+1} which is crashed.

- **Remark #4**: In a synchronous network assume that there exist $k \geq 1$ disjoint feedback channels from R to S. Also, assume that R sends the same message along each of these k feedback channels to S. As the network is synchronous, if S does not receive the message from R along any of these feedback channels then S learns that all feedback channels are crashed. Further, S also finds the identities of at least k faulty (crashed) nodes (follows from **Remark #3**).

Now we are ready to present the main theorem, which is as follows.

Theorem 5.4. *In a directed graph G,* PSMT *is possible from S to R, tolerating up to t_o omission faults if and only if G satisfies the following two conditions:*

1. *There exist at least $t_o + 1$ disjoint forward channels.*
2. *If there exist maximum of k disjoint feedback channels from R to S then there should exist $2(t_o - k) + 1$ disjoint forward channels in $G[V \setminus O]$, where O is any vertex-cut from R to S (of size k).*

Proof. **Necessity**: If there exist maximum of t_o disjoint forward channels then there exist a vertex-cut of size t_o from S to R. Therefore, by corrupting each of the nodes from the vertex-cut the adversary corrupts each of the forward channels. Thus, no communication is possible from S to R.
The necessity of the second condition is as follows. As there exist maximum of k disjoint feedback channels, the size of any vertex-cut from R to S is k. Suppose, assume that for some vertex-cut O (from R to S) there exist maximum of $2(t_o - k)$ disjoint forward channels in $G[V \setminus O]$. Then, by corrupting each of the nodes from the vertex-cut O, the adversary makes sure that no communication is possible from R to S. Therefore, even if both S and R agree on the identities of each of the corrupted nodes from O and subsequently remove them from the network, PSMT should be possible tolerating $t_o - k$ faults in the rest of the graph $G[V \setminus O]$ (from **Remark #1**). However, $2(t_o - k) + 1$ disjoint forward channels are necessary for PSMT in $G[V \setminus O]$ tolerating $(t_o - k)$ faults, as there is no feedback channel exists in $G[V \setminus O]$.

Sufficiency: Assume that network G satisfies both conditions of the theorem. If at least one of the feedback channels is fault-free, then PSMT is possible as there exist $t_o + 1$ forward channels (see **Remark #2**). In case if each feedback channel (k of them) is crashed then S finds a set of k faulty nodes in the network and shares the same information with R. Subsequently, both S and R remove these k faulty nodes from the network. In the rest of the network, as the maximum number of faults can occur is $t_o - k$ and there exist $2(t_o - k) + 1$ forward channels, S runs one-phase protocol tolerating $t_o - k$ faults. More formally, the protocol $\mathbf{\Pi_{RO}}$ which achieves PSMT works as follows. Let $t_o + 1$ disjoint forward channels be, namely, p_i for each $i \in [1, t_o + 1]$.

The Protocol $\mathbf{\Pi_{RO}}$:

1. S computes a random degree-t_o polynomial $p(x)$ such that its constant term $p(0)$ is the message m.

2. As we have t_o+1 disjoint forward channels, S sends t_o+1 points to R, exactly one point along each forward channel.

   - *If* R receives t_o+1 points then it gets the message m by reconstructing the polynomial $p(x)$. Further, R acknowledges S by sending $Success$ to S along each of the k feedback channels (as explained in **Remark #4**).
   - *If* R does not receive $f(\leq t_o)$ points, then R finds a set F of f corrupted nodes. Further, R sends the identities of f corrupted nodes to S along each of the k feedback channels (as explained in **Remark #4**).

3. *If* S receives neither $Success$ message nor the identities of corrupted nodes, then S learns that all feedback channels are crashed, since the network is synchronous. Further, S finds a set O of k corrupted nodes (see **Remark #4**) and execute the following steps.

   - S informs R the identities of each of the k corrupted nodes (from the set O) along each of the t_o+1 disjoint feedback channel, which guarantees reliability from S to R. Subsequently, both S and R remove these k nodes from the network and work on the rest of the graph $G[V \setminus O]$.
   - As $t_o - k$ is the maximum number of faults can occur in $G[V \setminus O]$ and there exist $2(t_o - k) + 1$ disjoint forward channels in $G[V \setminus O]$, S runs one-phase `PSMT` protocol given in **Theorem 5.2**.

4. *If* S receives (from R) the identities of each of the f corrupted nodes (of F) then it acknowledges the same back to R.

   - Subsequently, both S and R remove these f nodes (of F) from the network and work in the rest of the graph $G[V \setminus F]$.
   - In $G[V \setminus F]$, S repeats the above process from step-1 replacing degree-t_o polynomial with degree-$(t_o - f)$ polynomial.[a]

[a] As f faulty nodes (one from each of the f disjoint forward channels) were removed from G, in $G[V \setminus F]$ there exist at least $t_o - f + 1$ disjoint forward channels. Further, the maximum number of faults can occur in $G[V \setminus F]$ is $t_o - f$.

It is easy to see that the protocol $\mathbf{\Pi_{RO}}$ is reliable and secure. The protocol is reliable because, at some stage, one-phase protocol should be possible (if every feedback channel is corrupted) or R should have received the message (in the worst case after removing all t_o faulty nodes). The protocol is secure because, at any stage of the protocol the adversary gets a maximum of l points (depending on the knowledge of S and R about faulty nodes) on degree-l polynomial, which reveal nothing about the message. □

5.5 Summary

In this chapter we characterized networks for PSMT possibility tolerating t_o omission faults in wires model setting as well as routing model setting. In both settings, we designed communication efficient protocols whenever the given network admits corresponding PSMT. However, the characterization of PSMT possibility in arbitrary directed graph setting is left as an interesting open problem to work.

Chapter 6

SMT Tolerating Byzantine Faults

6.1 Introduction

In this chapter, we study Byzantine faults in undirected graph setting as well as directed graph setting. We consider threshold static adversary, that can corrupt up to t_b nodes in Byzantine fashion. We know that in any network, if some of the nodes are corrupted in fail-stop or omission fashion then the information sent by one node to another node may or may not reach reliably as the adversary can crash the corrupted nodes. However, if the intended recipient receives some information then that information must be authentic/correct as the adversary can not modify/change the information passing through faulty nodes. Whereas, the same is not true in the case of Byzantine faults; the recipient node may receive faulty information as well. In particular, the reviver R may receive faulty information from multiple nodes due to Byzantine faults, thus may not be able to reconstruct the correct message m sent by the sender S. Therefore, one should design reliable protocols before aiming for protocols which are both reliable and secure. In line with this, we first outline few existing results (with informal proofs) related to reliable message transmission. Then, we move to `SMT` results. At the end, we present our result, which answers the following question. Under what conditions is `USMT` (im)possible tolerating t_b Byzantine faults in routing model?

6.2 SMT in undirected graph model

In earlier chapters, in undirected graph model setting, we have seen elegant characterizations tolerating various types of faults. In Byzantine case too, we have such characterizations for different variants of `SMT` problem tolerating t_b Byzantine faults.

In undirected graph setting, Dolev et al. [1] introduced the SMT problem and studied the trade-off between network connectivity and the existence of PRMT and PSMT protocols tolerating t_b Byzantine faults. Later, Franklin et al. [12, 70] studied about URMT and USMT protocols. Below, we outline few existing results.

Theorem 6.1 (Reliable Message Transmission [12])**.** *In an undirected graph G,* URMT *from S to R tolerating t_b Byzantine faults is possible if and only if there exist at least $2t_b + 1$ vertex-disjoint paths between S and R.*

Proof. **Necessity**: An informal proof is presented here. If there exist maximum of $2t_b$ channels then the adversary corrupts half of the channels by corrupting t_b nodes, one from each of the corrupted t_b channels. The corrupted half paths behave as if the message sent by S is m' and the other half (non-faulty) paths behave as if the message sent by S is m. In other words, we can create two executions E and E', E for transmitting message m and E' for m', such that, for any given k, there is an adversary strategy for which, the receiver can not distinguish these two executions at the end of k rounds/phases. More precisely, the adversary has the following winning strategy. On faulty paths, while information is going from S to R, the adversary simulates the behaviour of E' and while information is coming from R to S the adversary stops the communication. As the adversary follows this strategy throughout the protocol execution, the receiver can not distinguish the original message from the faulty message.

Sufficiency: If there exist $2t_b + 1$ disjoint channels then S sends the message m to R along each of the $2t_b + 1$ disjoint channels. As t_b is the maximum number of Byzantine faults can occur, at least $t_b + 1$ (out of $2t_b$) copies of m will reach to R and at most t_b (out of $2t_b$) copies of m may be modified/corrupted during transit. Therefore, the receiver gets the message m by taking majority among the received messages. □

We just learned that URMT protocols exist only if there are $2t_b + 1$ disjoint paths between S and R. Therefore, as a corollary we get, $2t_b + 1$ disjoint paths between S and R are necessary for perfect reliability. Dolev et al. proved the same in [1].

Now we are ready to answer the following question. In an undirected graph, under what condition(s) PSMT protocols exist tolerating t_b Byzantine faults? And, the same is presented below.

Theorem 6.2 ([1])**.** *In an undirected graph G,* PSMT *from S to R tolerating t_b Byzantine faults is possible if and only if there exist $2t_b + 1$ vertex-disjoint paths between S and R.*

Proof. **Necessity**: Clearly $2t_b + 1$ disjoint paths between S and R are necessary for PSMT as it is necessary for PRMT itself.

Sufficiency: Suppose there exist $2t_b + 1$ disjoint paths between S and R, namely p_i for each $i \in [1, 2t_b + 1]$, we know that PRMT is possible between S and R. The designed PSMT protocols run in a sequence of phases and PSMT is achieved in two steps.

1. Step 1: The sender S starts with a random number r and tries to deliver r securely to the receiver R. In this process one of the following two can happen. As per the protocol code, the information received by R is either *consistent* or *inconsistent* with the information sent by S.

   - If the received information is consistent then R gets r securely. And, R reliably informs the sender that the received information is consistent (by sending the *Success* message to S along each of the $2t_b + 1$ disjoint paths). Following the *Success* message from R, S sends $m + r$ reliably to R as PRMT is possible from S to R. Upon receiving $m + r$ from S, R gets the message m by subtracting r from $m + r$.
   - If the information received by R is inconsistent, then, R reliably informs back S the whole information that it had received from each of the $2t_b + 1$ paths. We notice that in this process, the secrecy of r may be lost since R informs S the whole information that it had received. However, as r is a random number which is independent of the message m, it reveals nothing about m.

2. Step 2: S upon receiving the inconsistent information from R, it checks which path(s) delivered inconsistent information and subsequently eliminates corresponding faulty path(s). After eliminating the faulty path(s), S again initiates the above process to deliver another random number securely. Either R receives the random number securely or S finds at least one faulty path. The algorithm converges after certain number of phases as the maximum number of faults can occur is t_b, which is finite.

How to check if the information received by S is (in)consistent?
For an example, we can use the following method to check if the information received by S is (in)consistent. The sender S starts with a random degree-t_b polynomial $p(x)$ such that the constant term $p(0)$ is r and it sends $p(i)$ to R along the path p_i, for each $i \in [1, 2t_b + 1]$. We say that the information received by R is consistent if it satisfies

following two conditions: (1) R receives $2t_b + 1$ points, and (2) upon reconstructing a degree-t_b polynomial $q(x)$ with any $t_b + 1$ points (out of $2t_b + 1$ received points), rest of the t_b points must lie on the same polynomial $q(x)$. Else we say that the information received by R is inconsistent. This protocol is perfectly secure because $q(x) = p(x)$ only if $q(i) = p(i)$ for each $i \in [1, 2t_b + 1]$. For more details see [1]. □

We have seen that URMT protocols exist between S and R only if there exist $2t_b + 1$ vertex-disjoint paths between S and R. Also, we have seen that if there exist $2t_b + 1$ vertex-disjoint paths between S and R then PSMT protocols exist between S and R. Sandwiching these two we get that, USMT is possible between S and R if and only if there exist $2t_b + 1$ vertex-disjoint paths between S and R.

6.3 USMT in wires model

In this section, we study existing results related to USMT protocols in wires model setting. Wang et al. [14, 71] initiated the study of (different variants of) SMT protocols in wires model setting as there are networks in which each channel between S and R need not be two-way connected. For some of the variants of SMT problem, Wang et al. studied about the required number of disjoint feedback channels for the existence of corresponding SMT protocols given that a certain number of disjoint forward channels (which are disjoint with feedback channels) exist. In particular, one of the results is the following. Under what conditions USMT is (im)possible if there exists no feedback channel.

Theorem 6.3 ([14]). *Let $G(V, E)$ be a directed graph $S, R \in V$ and $0 < \delta < 1/2$. If there is no directed path from R to S, then the necessary and sufficient condition for $(0, \delta)$-secure message transmission from S to R against a t_b-Byzantine adversary is that there are $2t_b + 1$ directed node disjoint paths from S to R.*

We know that, if there exist maximum of t_b vertex-disjoint forward channels then no communication is possible from S to R as the adversary cuts all the channels by corrupting one node from each of these t_b forward channels. Also, from Theorem 6.3 we know that $2t_b + 1$ forward channels are sufficient for USMT. However, consider the graph G of Figure 6.1, which fails to have three forward paths, hence Theorem 6.3 can not answer whether USMT is possible or not in G tolerating one Byzantine fault (as there exist a feedback channel). Wang et al. characterized this kind of networks in the wires model by studying the required number of feedback channels, when there exist at least $t_b + 1$ and at most $2t_b$ forward channels. The main result is the following.

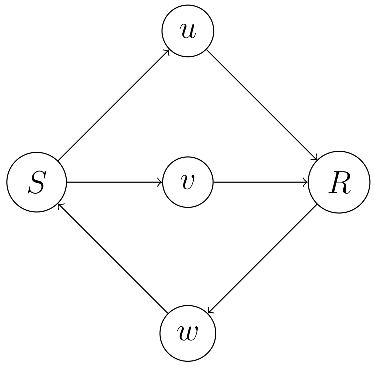

Figure 6.1: Graph G having two forward channels and one feedback channel.

Theorem 6.4 ([14]). *Let $G(V,E)$ be a directed graph, $S,R \in V$, $t_b \geq u \geq 1$, $\delta < \frac{1}{2}(1-\frac{1}{|\mathbb{F}|})$, and there are $2t_b+1-u$ directed node disjoint paths $p_1,\ldots,p_{2t_b+1-u}$ from S to R. Then a necessary and sufficient condition for $(0,\delta)$-secure message transmission protocol from S to R against a t_b-active adversary is that there are u directed node disjoint paths $q_1,\ldots,q_u(q_1,\ldots,q_u$ are node disjoint from $p_1,\ldots,p_{2t_b+1-u})$ from R to S.*

6.4 USMT in routing model

In this section, we study `USMT` in routing model setting. Analogous to results in earlier chapters, we show that there are networks in which `USMT` protocols exist if we abstract the network as in routing model whereas `USMT` is impossible if we abstract the network as in wires model. For example, consider the graph G given in Figure 6.2. G has `only` two disjoint `forward` channels, namely $p_1 : \langle S,u,R\rangle$ and $p_2 : \langle S,v,w,R\rangle$. Notice that, these two paths exhaust all the nodes in G. Hence, there exists *no* feedback channel which is disjoint with both p_1 and p_2. Therefore, G fails to meet the conditions of the Theorem 6.4. However, notice that there are two feedback channels, namely, $p_3 : \langle R,v,S\rangle$ and $p_4 : \langle R,w,S\rangle$ and both p_3 and p_4 are disjoint with p_1. This implies, actually, there are *three* disjoint channels in total, namely, p_1, p_3 and p_4. We show that these two disjoint forward channels p_1, p_2 and three disjoint channels p_1, p_3 and p_4, together, are sufficient for the existence of a $(0,\delta)$-`SMT` protocol in G, tolerating one Byzantine fault. Before proceeding to the main theorem, we note couple of remarks from the existing result.

Remark 1: The sufficiency condition of the Theorem 6.4, states that, if there exist a total of $2t_b+1$ disjoint channels between S and R such that at least t_b+1 of them are from S to R then $(0,\delta)$-`SMT` is possible from S to R.

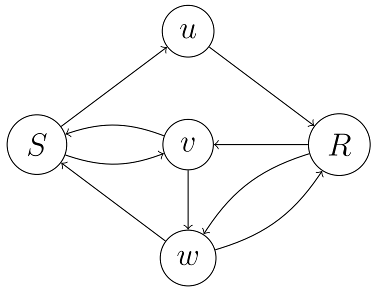

Figure 6.2: Graph G with three disjoint channels.

Remark 2: Suppose there exist a total of $2t_b + 1$ disjoint channels between S and R, then at least $t_b + 1$ of them must be *either* from S to R *or* from R to S. This implies (from Remark 1) that, $(0, \delta)$-SMT must be possible, *either* from S to R *or* from R to S.

Theorem 6.5. *In a directed graph G, $(0, \delta)$-SMT is possible from S to R, tolerating up to t_b Byzantine faults if and only if G satisfies the following two conditions:*

1. *There exists at least $t_b + 1$ disjoint forward channels - for reliability.*
2. *There exist at least $2t_b + 1$ disjoint channels - for security.*

Proof. **Necessity:** The necessity of the first condition is trivial to understand. The necessity of the second condition follows from the fact that, if each channel is two-way connected then also $2t_b + 1$ channels are necessary for the existence of a $(0, \delta)$-SMT protocol (see **Theorem 5.1** of [12]).
Sufficiency: Let us assume that G satisfies both the conditions. The **Remark 2** and second condition together imply that $(0, \delta)$-SMT must be possible, *either* from S to R *or* from R to S. In the former case nothing left to prove. As in the latter case, suppose $(0, \delta)$-SMT is possible from R to S then R securely sends three keys, say K_1, K_2, K_3, to S using a $(0, \delta)$-SMT protocol (e.g., protocol given by Wang et al. [14] or Patra et al. [41]). Upon receiving K_1, K_2, K_3, S simply sends the pair $\langle m + K_1, (m + K_1)K_2 + K_3 \rangle$ to R using each of the $t_b + 1$ disjoint forward channels. The first condition of the theorem assures the existence of such $t_b + 1$ channels. R upon receiving a pair $\langle X, Y \rangle$ from S, checks whether $XK_2 + K_3$ is same as Y or not. If it is same then R subtracts K_1 from X and gets the message m else R discards that pair. As there are $t_b + 1$ disjoint forward channels, and at most t_b Byzantine faults can occur, at least one pair satisfies the relation. For the correctness of the authentication scheme, see [11, 41, 49]. □

Recall that, the graph G (given in Figure 6.2) has two disjoint forward channels

p_1, p_2 and three disjoint channels p_1, p_3, p_4 - thus meets the conditions of the above theorem tolerating one Byzantine fault.

6.5 Summary

In this chapter we studied about Byzantine faults. First, we presented the existing `USMT` result in wires model setting and moved to the characterization of `USMT` in routing model setting. We explicitly showed that there are networks in which `USMT` is possible if we abstract the network as in routing model where as `USMT` (tolerating the same/corresponding adversary) is impossible if we abstract the network as in wires model setting. However, many questions remain unanswered. For example, under what condition(s) `PSMT` is possible tolerating t_b Byzantine faults in arbitrary directed graph setting is an interesting open problem to investigate.

Chapter 7

Summary and Future work

7.1 Summary

In this thesis, we studied about SMT protocols under four different types of network models, namely, undirected graph model, wires model, routing model and arbitrary directed graph model. We introduced routing model, which is a variant of wires model. In routing model, the network is abstracted as a collection of paths from S to R as well as paths from R to S. The main difference comparing with wires model is that paths from S to R need not be disjoint with paths from R to S and vice-versa. That is, a path from S to R can share a vertex with a path from R to S and vice-versa – overlap is allowed. By introducing routing model, we are able to answer the following fundamental question; for a given variant of SMT problem, under what condition(s) corresponding SMT protocols exist if each intermediate node is a mere router (nodes can not do computations but can route the messages to their neighbours). Moreover, we showed that:

1. There are networks of type $\mathcal{N}$ such that, for some variants of SMT problem, if we abstract $\mathcal{N}$ as in routing model then corresponding SMT protocols exist whereas, if we abstract $\mathcal{N}$ as in wires model then no SMT protocol exists in $\mathcal{N}$.

2. There is no network such that, if we abstract it as in wires model setting then a given variant of SMT is possible whereas if we abstract it as in routing model setting then the corresponding SMT is not possible.

The above two points together imply that if we abstract the network as in wires model then there is a loss of generality even if each intermediary node is a mere router. Also, in each of the network model (except for the case of undirected graph model, as protocols already exist in the literature), we presented various (communication efficient) protocols tolerating different types of faults, namely, passive faults, mixed faults (passive + fail-stop), omission faults and Byzantine faults. More elaborately:

1. In Chapter 3, we studied about passive faults. In passive setting, for each of the network model (except for the case of undirected graph model), we characterized networks by giving necessary and sufficient condition(s) for the existence of a PSMT protocol tolerating t_p passive faults. In each case, we designed a communication efficient PSMT protocol. Also, we completely characterized the optimality of protocols for PSMT in arbitrary networks under the influence of the passive static adversary. We subsequently extended the same to tolerate mobile faults. We proved that empowering the adversary from static to mobile, alters neither the connectivity requirements nor the efficiency parameters. We also extended our ideas to incorporate multiple receivers (multicast), and arrive at optimal protocols in the more general setting.

2. In Chapter 4, we completely characterized the PSMT possibility in wires model setting as well as routing model setting, tolerating up to t_f fail-stop faults in addition to t_p passive faults. And, proposed protocols are communication efficient as well. Also, in arbitrary directed graphs, we characterized the PSMT possibility tolerating up to t_f fail-stop faults and further up to t_p passive faults. However, the protocol presented is not communication efficient. Towards an efficient protocol, we presented a sufficiency condition for PSMT possibility and proved that each graph that satisfy the wires model characterization also satisfies this particular sufficiency condition but not the other way around. Moreover, we explicitly showed that there are graphs which fail to satisfy this particular sufficiency condition but have PSMT protocols. That is, this condition is just sufficient but not necessary. To close this gap, it is worth studying to prove that: (1) efficient protocols always exist or (2) there exist digraphs in which no efficient protocols exist. Our conjecture is that there exist digraphs in which no efficient protocols exist. Finally, we characterized the set of digraphs in which PSMT is possible between each pair of nodes.

3. In Chapter 5, we studied about omission faults. In wires model setting as well as routing model setting, we characterized networks by giving necessary and sufficient condition(s) for the existence of PSMT protocols tolerating t_o omission faults. Furthermore, in both settings, proposed protocols are communication efficient as well.

4. In Chapter 6, under routing model setting, we characterized networks by giving necessary and sufficient condition(s) for the existence of USMT protocols tolerating t_b Byzantine faults. Moreover, we designed a communication efficient protocol, whenever the given network meets the sufficiency condition(s) of the corresponding theorem.

7.2 Future Work

We have seen various characterizations in undirected graph model as wells as directed graph model, tolerating different kinds of faults. However, many questions remain unanswered and are worth studying. For example, for the following problems, the characterization itself is unknown.

1. In arbitrary directed graph setting, what is the necessary and sufficient condition for the existence of a PSMT protocol tolerating t_o omission faults?

2. In arbitrary directed graph setting, what is the necessary and sufficient condition for the existence of a PRMT/PSMT protocol tolerating t_b Byzantine faults?

3. In routing model setting, what is the necessary and sufficient condition for the existence of a PSMT protocol tolerating t_b Byzantine faults?

4. In routing model setting, under what conditions USMT is (im)possible tolerating mixed adversary – up to t_b Byzantine faults, t_o omission faults, t_f fail-stop faults and t_p passive faults.

In arbitrary directed graph model, is it always possible to a design communication efficient protocol in every graph which admits corresponding SMT? For example:

1. Is it always possible to design a communication efficient PSMT protocol in every graph which admits PSMT tolerating the threshold mixed adversary (passive+fail-stop)?

2. We have characterization for URMT/USMT in arbitrary directed graph setting, but the communication complexity of each of the protocol presented in the literature is exponential. The natural question is, does there exist any communication efficient URMT/USMT protocol (if it admits corresponding SMT) in a given network? If yes, how to design one such?

It is also interesting and worth studying about designing efficient algorithms (if at all possible) to check whether the given graph have the required connectivity or not for the existence of corresponding `SMT` protocols. In particular:

1. Does there exist any efficient algorithm that checks whether the graph has k disjoint wires (together from S to R and R to S) or not, for any given k, where a wire is a path from S to R or R to S? If there exists one such algorithm then in routing model setting, one can always efficiently test whether the given variant of `SMT` is possible or not.

2. Does there exist any efficient algorithm which checks whether the graph meets the necessary and sufficient conditions for the existence of a `PSMT` protocol tolerating up to t_f fail-stop faults and further up to t_p passive faults in arbitrary digraphs? In other words, is there any efficient algorithm which checks whether the given network G meets the following condition or not. For any given two natural numbers f and p, on removal of any set F of f nodes (except S and R) from the network, does there exist p vertex-disjoint weak paths from S to R in $G[V \setminus F]$ such that every node in $G[V \setminus F]$ has a path to R in $G[V \setminus F]$.

Publications

The following list of articles are contributed to this thesis.

1. Round-Optimal Perfectly Secret Message Transmission with Linear Communication Complexity.
   Ravi Kishore, Ashutosh Kumar, Chiranjeevi Vanarasa and Kannan Srinathan.
   (Information Theoretic Security - 8th International Conference, ICITS 2015, Lugano, Switzerland, May 2-5, 2015. Proceedings, volume 9063 of Lecture Notes in Computer Science, pages 33-50, Springer, 2015).
   DOI: `https://doi.org/10.1007/978-3-319-17470-9_3`.

2. On Perfectly Secret Message Transmission in Digraphs Tolerating Dual Failures.
   Ravi Kishore, Chiranjeevi Vanarasa, Tushant Jha and Kannan Srinathan.
   (Proceedings of the 17th International Conference on Distributed Computing and Networking (ICDCN), Singapore, January 4-7, 2016, pages 29:1–29:10).
   DOI: `https://doi.org/10.1145/2833312.2833327`.

3. On the Price of Proactivizing Round-Optimal Perfectly Secret Message Transmission.
   Ravi Kishore, Ashutosh Kumar, Chiranjeevi Vanarasa and Kannan Srinathan.
   (To appear in IEEE Transactions on Information Theory Journal).
   DOI: `https://doi.org/10.1109/TIT.2017.2776099`.

4. On Minimal Connectivity Requirement for Secure Message Transmission in Directed Networks.
   Ravi Kishore, Chiranjeevi Vanarasa and Kannan Srinathan.
   (To appear in Journal of Information Processing Letters).
   DOI: `https://doi.org/10.1016/j.ipl.2017.11.001`.

Bibliography

[1] D. Dolev, C. Dwork, O. Waarts, M. Yung, Perfectly secure message transmission, J. ACM 40 (1) (1993) pp. 17–47.

[2] J. Katz, Y. Lindell, Introduction to Modern Cryptography, Chapman and Hall/CRC Press, 2007.

[3] C. E. Shannon, Communication theory of secrecy systems, Bell System Technical Journal 28 (4) (1949) pp. 656–715.

[4] D. Dolev, The byzantine generals strike again, J. Algorithms 3 (1) (1982) pp. 14–30.

[5] M. J. Fischer, N. A. Lynch, M. Paterson, Impossibility of distributed consensus with one faulty process, J. ACM 32 (2) (1985) pp. 374–382.

[6] M. C. Pease, R. E. Shostak, L. Lamport, Reaching agreement in the presence of faults, J. ACM 27 (2) (1980) pp. 228–234.

[7] D. Beaver, Efficient multiparty protocols using circuit randomization, in: Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, Proceedings, 1991, pp. 420–432.

[8] M. Ben-Or, S. Goldwasser, A. Wigderson, Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract), in: Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, Chicago, Illinois, USA, 1988, pp. 1–10.

[9] O. Goldreich, S. Micali, A. Wigderson, How to play any mental game or A completeness theorem for protocols with honest majority, in: Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA, 1987, pp. 218–229.

[10] A. C. Yao, Protocols for secure computations (extended abstract), in: 23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November, 1982, pp. 160–164.

[11] T. Rabin, M. Ben-Or, Verifiable secret sharing and multiparty protocols with honest majority (extended abstract), in: Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, Seattle, Washigton, USA, 1989, pp. 73–85.

[12] M. K. Franklin, R. N. Wright, Secure communication in minimal connectivity models, J. Cryptology 13 (1) (2000) pp. 9–30.

[13] A. Narayanan, K. Srinathan, C. P. Rangan, Perfectly reliable message transmission, Inf. Process. Lett. 100 (1) (2006) pp. 23–28.

[14] Y. Wang, Y. Desmedt, Perfectly secure message transmission revisited, IEEE Transactions on Information Theory 54 (6) (2008) pp. 2582–2595.

[15] A. Badanidiyuru, A. Patra, A. Choudhury, K. Srinathan, C. P. Rangan, On the trade-off between network connectivity, round complexity, and communication complexity of reliable message transmission, J. ACM 59 (5) (2012) pp. 22.

[16] A. Mehta, S. Agrawal, K. Srinathan, Interplay between (im)perfectness, synchrony and connectivity: The case of reliable message transmission, Theor. Comput. Sci. 496 (2013) pp. 2–16.

[17] M. K. Franklin, M. Yung, Secure hypergraphs: privacy from partial broadcast (extended abstract), in: Proceedings of the Twenty-Seventh Annual ACM Symposium on Theory of Computing, 29 May - 1 June, Las Vegas, Nevada, USA, 1995, pp. 36–44.

[18] M. K. Franklin, M. Yung, Secure hypergraphs: Privacy from partial broadcast, SIAM J. Discrete Math. 18 (3) (2004) pp. 437–450.

[19] K. Srinathan, A. Patra, A. Choudhary, C. P. Rangan, Unconditionally reliable message transmission in directed hypergraphs, in: Cryptology and Network Security, 7th International Conference, CANS 2008, Hong-Kong, China, December 2-4. Proceedings, 2008, pp. 285–303.

[20] M. Hirt, U. M. Maurer, Complete characterization of adversaries tolerable in secure multi-party computation (extended abstract), in: Proceedings of the Sixteenth Annual ACM Symposium on Principles of Distributed Computing, Santa Barbara, California, USA, August 21-24, 1997, pp. 25–34.

[21] A. Patra, B. Shankar, A. Choudhary, K. Srinathan, C. P. Rangan, Perfectly secure message transmission in directed networks tolerating threshold and non threshold adversary, in: Cryptology and Network Security, 6th International Conference, CANS 2007, Singapore, December 8-10, Proceedings, 2007, pp. 80–101.

[22] R. Ostrovsky, M. Yung, How to withstand mobile virus attacks (extended abstract), in: Proceedings of the Tenth Annual ACM Symposium on Principles of Distributed Computing, Montreal, Quebec, Canada, August 19-21, 1991, pp. 51–59.

[23] M. Backes, C. Cachin, R. Strobl, Proactive secure message transmission in asynchronous networks, in: Proceedings of the Twenty-Second ACM Symposium on Principles of Distributed Computing, PODC 2003, Boston, Massachusetts, USA, July 13-16, 2003, pp. 223–232.

[24] K. Srinathan, P. Raghavendra, C. P. Rangan, On proactive perfectly secure message transmission, in: Information Security and Privacy, 12th Australasian Conference, ACISP 2007, Townsville, Australia, July 2-4, Proceedings, 2007, pp. 461–473.

[25] A. Patra, A. Choudhary, M. Vaidyanathan, C. P. Rangan, Efficient perfectly reliable and secure message transmission tolerating mobile adversary, in: Information Security and Privacy, 13th Australasian Conference, ACISP 2008, Wollongong, Australia, July 7-9, Proceedings, 2008, pp. 170–186.

[26] A. Patra, A. Choudhury, C. P. Rangan, Brief announcement: perfectly secure message transmission tolerating mobile mixed adversary with reduced phase complexity, in: Proceedings of the 29th Annual ACM Symposium on Principles of Distributed Computing, PODC 2010, Zurich, Switzerland, July 25-28, 2010, pp. 245–246.

[27] M. Fitzi, M. Hirt, U. M. Maurer, Trading correctness for privacy in unconditional multi-party computation (extended abstract), in: Advances in Cryptology - CRYPTO '98, 18th Annual International Cryptology Conference, Santa Barbara, California, USA, August 23-27, Proceedings, 1998, pp. 121–136.

[28] K. Srinathan, M. V. N. A. Kumar, C. P. Rangan, Asynchronous secure communication tolerating mixed adversaries, in: Advances in Cryptology - ASIACRYPT 2002, 8th International Conference on the Theory and Application of Cryptology and Information Security, Queenstown, New Zealand, December 1-5, Proceedings, 2002, pp. 224–242.

[29] A. Patra, A. Choudhary, K. Srinathan, C. P. Rangan, Perfectly reliable and secure communication in directed networks tolerating mixed adversary, in: Distributed Computing, 21st International Symposium, DISC 2007, Lemesos, Cyprus, September 24-26, Proceedings, 2007, pp. 496–498.

[30] A. Choudhary, A. Patra, B. V. Ashwinkumar, K. Srinathan, C. P. Rangan, Perfectly reliable and secure communication tolerating static and mobile mixed adversary, in: Information Theoretic Security, Third International Conference, ICITS 2008, Calgary, Canada, August 10-13, Proceedings, 2008, pp. 137–155.

[31] K. Srinathan, A. Patra, A. Choudhary, C. P. Rangan, Unconditionally secure message transmission in arbitrary directed synchronous networks tolerating generalized mixed adversary, in: Proceedings of the 2009 ACM Symposium on Information, Computer and Communications Security, ASIACCS 2009, Sydney, Australia, March 10-12, 2009, pp. 171–182.

[32] H. M. Sayeed, H. Abu-Amara, Perfectly secure message transmission in asynchronous networks, in: Proceedings of the Seventh IEEE Symposium on Parallel and Distributed Processing, SPDP 1995, San Antonio, Texas , USA, October 25-28, 1995, pp. 100–105.

[33] M. V. N. A. Kumar, K. Srinathan, C. P. Rangan, Asynchronous perfectly secure computation tolerating generalized adversaries, in: Information Security and Privacy, 7th Australian Conference, ACISP 2002, Melbourne, Australia, July 3-5, Proceedings, 2002, pp. 497–512.

[34] A. Choudhary, A. Patra, B. V. Ashwinkumar, K. Srinathan, C. P. Rangan, On minimal connectivity requirement for secure message transmission in asynchronous networks, in: Distributed Computing and Networking, 10th International Conference, ICDCN 2009, Hyderabad, India, January 3-6. Proceedings, 2009, pp. 148–162.

[35] A. Mehta, S. Agrawal, K. Srinathan, Brief announcement: Synchronous las vegas URMT iff asynchronous monte carlo URMT, in: Distributed Computing, 24th International Symposium, DISC 2010, Cambridge, MA, USA, September 13-15. Proceedings, 2010, pp. 201–203.

[36] A. Choudhury, A. Patra, B. V. Ashwinkumar, K. Srinathan, C. P. Rangan, Secure message transmission in asynchronous networks, J. Parallel Distrib. Comput. 71 (8) (2011) pp. 1067–1074.

[37] S. Agrawal, A. Mehta, K. Srinathan, Secure message transmission in asynchronous directed graphs, in: Progress in Cryptology - INDOCRYPT 2011 - 12th International Conference on Cryptology in India, Chennai, India, December 11-14. Proceedings, 2011, pp. 359–378.

[38] K. Menger, Zur allgemeinen kurventheorie, Fundamenta Mathematicae 10 (1927) pp. 96–115.

[39] A. Patra, A. Choudhary, C. P. Rangan, Brief announcement: perfectly secure message transmission in directed networks re-visited, in: Proceedings of the 28th Annual ACM Symposium on Principles of Distributed Computing, PODC 2009, Calgary, Alberta, Canada, August 10-12, 2009, pp. 278–279.

[40] A. Patra, A. Choudhary, C. P. Rangan, On communication complexity of secure message transmission in directed networks, in: Distributed Computing and Networking, 11th International Conference, ICDCN 2010, Kolkata, India, January 3-6. Proceedings, 2010, pp. 42–53.

[41] A. Patra, A. Choudhary, C. P. Rangan, Unconditionally reliable and secure message transmission in directed networks revisited, in: Security and Cryptography for Networks, 6th International Conference, SCN 2008, Amalfi, Italy, September 10-12. Proceedings, 2008, pp. 309–326.

[42] R. Kishore, C. Vanarasa, T. Jha, K. Srinathan, On perfectly secret message transmission in digraphs tolerating dual failures, in: Proceedings of the 17th International Conference on Distributed Computing and Networking, Singapore, January 4-7, 2016, pp. 29:1–29:10.

[43] M. V. N. A. Kumar, P. R. Goundan, K. Srinathan, C. P. Rangan, On perfectly secure communication over arbitrary networks, in: Proceedings of the Twenty-First Annual ACM Symposium on Principles of Distributed Computing, PODC 2002, Monterey, California, USA, July 21-24, 2002, pp. 193–202.

[44] R. Kishore, A. Kumar, C. Vanarasa, K. Srinathan, Round-optimal perfectly secret message transmission with linear communication complexity, in: Information Theoretic Security - 8th International Conference, ICITS 2015, Lugano, Switzerland, May 2-5. Proceedings, 2015, pp. 33–50.

[45] K. Kurosawa, K. Suzuki, Truly efficient 2-round perfectly secure message transmission scheme, IEEE Trans. Information Theory 55 (11) (2009) pp. 5223–5232.

[46] A. Patra, A. Choudhary, K. Srinathan, C. P. Rangan, Constant phase bit optimal protocols for perfectly reliable and secure message transmission, in: Progress in Cryptology - INDOCRYPT 2006, 7th International Conference on Cryptology in India, Kolkata, India, December 11-13, Proceedings, 2006, pp. 221–235.

[47] K. Srinathan, C. P. Rangan, Possibility and complexity of probabilistic reliable communication in directed networks, in: Proceedings of the Twenty-Fifth Annual ACM Symposium on Principles of Distributed Computing, PODC 2006, Denver, CO, USA, July 23-26, 2006, pp. 265–274.

[48] A. Patra, A. Choudhary, C. P. Rangan, Constant phase efficient protocols for secure message transmission in directed networks, in: Proceedings of the Twenty-Sixth Annual ACM Symposium on Principles of Distributed Computing, PODC 2007, Portland, Oregon, USA, August 12-15, 2007, pp. 322–323.

[49] M. Nayak, S. Agrawal, K. Srinathan, Minimal connectivity for unconditionally secure message transmission in synchronous directed networks, in: Information Theoretic Security - 5th International Conference, ICITS 2011, Amsterdam, The Netherlands, May 21-24. Proceedings, 2011, pp. 32–51.

[50] J. Renault, L. Renou, T. Tomala, Secure message transmission on directed networks, Games and Economic Behavior 85 (2014) pp. 1–18.

[51] K. Srinathan, N. R. Prasad, C. P. Rangan, On the optimal communication complexity of multiphase protocols for perfect communication, in: 2007 IEEE Symposium on Security and Privacy (S&P 2007), 20-23 May, Oakland, California, USA, 2007, pp. 311–320.

[52] M. Fitzi, M. K. Franklin, J. A. Garay, H. V. Simhadri, Towards optimal and efficient perfectly secure message transmission, in: Theory of Cryptography, 4th Theory of Cryptography Conference, TCC 2007, Amsterdam, The Netherlands, February 21-24, Proceedings, 2007, pp. 311–322.

[53] K. Srinathan, A. Choudhary, A. Patra, C. P. Rangan, Efficient single phase unconditionally secure message transmission with optimum communication complexity, in: Proceedings of the Twenty-Seventh Annual ACM Symposium on Principles of Distributed Computing, PODC 2008, Toronto, Canada, August 18-21, 2008, pp. 457–457.

[54] W. Diffie, M. E. Hellman, New directions in cryptography, IEEE Transactions on Information Theory 22 (6) (1976) pp. 644–654.

[55] H. M. Sayeed, H. Abu-Amara, Efficient perfectly secure message transmission in synchronous networks, Inf. Comput. 126 (1) (1996) pp. 53–61.

[56] P. K. Vasishta, P. Gopal, A. Gupta, P. Bansal, K. Srinathan, Brief announcement: topology knowledge affects probabilistic reliable communication, in: Proceedings of the 28th Annual ACM Symposium on Principles of Distributed Computing, PODC 2009, Calgary, Alberta, Canada, August 10-12, 2009, pp. 314–315.

[57] K. Srinathan, A. Narayanan, C. P. Rangan, Optimal perfectly secure message transmission, in: Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, Proceedings, 2004, pp. 545–561.

[58] B. Shankar, P. Gopal, K. Srinathan, C. P. Rangan, Unconditionally reliable message transmission in directed networks, in: Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2008, San Francisco, California, USA, January 20-22, 2008, pp. 1048–1055.

[59] A. Patra, A. Choudhury, C. P. Rangan, K. Srinathan, Unconditionally reliable and secure message transmission in undirected synchronous networks: possibility, feasibility and optimality, IJACT 2 (2) (2010) pp. 159–197.

[60] K. Srinathan, A. Patra, A. Choudhary, C. P. Rangan, Probabilistic perfectly reliable and secure message transmission - possibility, feasibility and optimality, in: Progress in Cryptology - INDOCRYPT 2007, 8th International Conference on Cryptology in India, Chennai, India, December 9-13, Proceedings, 2007, pp. 101–122.

[61] A. Patra, A. Choudhary, C. P. Rangan, K. Srinathan, P. Raghavendra, Perfectly reliable and secure message transmission tolerating mobile adversary, IJACT 1 (3) (2009) pp. 200–224.

[62] B. V. Ashwinkumar, A. Patra, A. Choudhary, K. Srinathan, C. P. Rangan, On tradeoff between network connectivity, phase complexity and communication complexity of reliable communication tolerating mixed adversary, in: Proceedings of the Twenty-Seventh Annual ACM Symposium on Principles of Distributed Computing, PODC 2008, Toronto, Canada, August 18-21, 2008, pp. 115–124.

[63] N. A. Lynch, Distributed Algorithms, Morgan Kaufmann, 1996.

[64] A. Shamir, How to share a secret, Commun. ACM 22 (11) (1979) pp. 612–613.

[65] H. Shi, S. Jiang, R. Safavi-Naini, M. A. Tuhin, On optimal secure message transmission by public discussion, IEEE Transactions on Information Theory 57 (1) (2011) pp. 572–585.

[66] A. V. Goldberg, R. E. Tarjan, Efficient maximum flow algorithms, Commun. ACM 57 (8) (2014) pp. 82–89.

[67] A. V. Goldberg, R. E. Tarjan, A new approach to the maximum flow problem, Journal of the ACM 35 (1988) pp. 921–940.

[68] R. E. Tarjan, Testing graph connectivity, in: Proceedings of the Sixth Annual ACM Symposium on Theory of Computing, STOC '74, ACM, New York, NY, USA, 1974, pp. 185–193.

[69] K. Srinathan, Secure distributed communication, Ph.D. thesis, IIT Madras (2006).

[70] M. K. Franklin, R. N. Wright, Secure communications in minimal connectivity models, in: Advances in Cryptology - EUROCRYPT '98, International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, May 31 - June 4, Proceeding, 1998, pp. 346–360.

[71] Y. Desmedt, Y. Wang, Perfectly secure message transmission revisited, in: Advances in Cryptology - EUROCRYPT 2002, International Conference on the Theory and Applications of Cryptographic Techniques, Amsterdam, The Netherlands, April 28 - May 2, Proceedings, 2002, pp. 502–517.