ABSTRACT

Title of dissertation: BROADCAST AND VERIFIABLE SECRET SHARING: NEW SECURITY MODELS AND ROUND-OPTIMAL CONSTRUCTIONS

Ranjit Kumaresan, Doctor of Philosophy, 2012

Dissertation directed by: Professor Jonathan Katz
Department of Computer Science

Broadcast and verifiable secret sharing (VSS) are central building blocks for secure multi-party computation. These protocols are required to be resilient against a Byzantine adversary who controls at most t out of the n parties running the protocol. In this dissertation, we consider the design of fault-tolerant protocols for broadcast and verifiable secret sharing with stronger security guarantees and improved round complexity.

Broadcast allows a party to send the same message to all parties, and all parties are assured they have received identical messages. Given a public-key infrastructure (PKI) and digital signatures, it is possible to construct broadcast protocols tolerating any number of corrupted parties. We address two important issues related to broadcast: (1) Almost all existing protocols do not distinguish between corrupted parties (who do not follow the protocol) and honest parties whose secret (signing) keys have been compromised (but who continue to behave honestly); (2) all existing protocols for broadcast are insecure against an *adaptive* adversary who can choose which parties to corrupt as the protocol progresses. We propose new security models that capture these issues, and present tight feasibility and impossibility results.

In the problem of verifiable secret sharing, there is a designated player who shares a secret during an initial sharing phase such that the secret is hidden from an adversary that corrupts at most t parties. In a subsequent reconstruction phase of the protocol, a unique secret, well-defined by the view of honest players in the sharing phase, is reconstructed. The round complexity of VSS protocols is a very important metric of their efficiency. We show two improvements regarding the round complexity of information-theoretic VSS. First, we construct an efficient perfectly secure VSS protocol tolerating $t < n/3$ corrupted parties that is simultaneously optimal in both the number of rounds and the number of invocations of broadcast. Second, we construct a statistically secure VSS protocol tolerating $t < n/2$ corrupted parties that has optimal round complexity, and an *efficient* statistical VSS protocol tolerating $t < n/2$ corrupted parties that requires one additional round.

Broadcast and Verifiable Secret Sharing: New Security Models and Round-Optimal Constructions

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2012

Ranjit Kumaresan

Advisor:
Prof. Jonathan Katz
Committee Members:
Prof. William Gasarch
Prof. Samir Khuller
Prof. MohammadTaghi HajiAghayi
Prof. Sennur Ulukus

Department of Computer Science
University of Maryland, College Park, MD 20742

Acknowledgments

I would like to thank my advisor Jonathan Katz for his fantastic guidance and support. Needless to say, I consider myself very fortunate to have him as my advisor. I have learnt a lot from him over the past few years. In particular, his high standards have helped me realize what it takes to be a disciplined researcher. I would also like to thank him for being very patient with me, and for having faith in me even during times when I did not completely deserve it. All this and several other things that I learnt from him have contributed significantly to my personal and intellectual growth. I will be forever indebted to him.

I would like to thank all past and present students and post-doctoral researchers in the Crypto group at UMD. Special thanks to Hong-Sheng Zhou and Seung Geol Choi who were always enthusiastic about teaching me topics in cryptography which I knew very little about. Their constant guidance and feedback has significantly broadened my areas of interest and has made me a much better researcher. Discussions with Vassilis Zikas have greatly helped me in improving the presentation of the results in Chapter 6. I would like to thank him for his advice, and also for being extremely patient with me during these discussions. Special thanks also to Dov Gordon and Arkady Yerukhimovich who have been great friends and officemates for several years. I would also like to thank Chiu-Yuen Koo for being a great friend, a well-wisher, and for giving me an opportunity to collaborate with him during my first year in graduate school.

I am grateful to Vladimir Kolesnikov for taking me on as a summer intern in Bell Labs in 2011. I would like to thank my advisor for introducing me to him. I had a very productive time at Bell Labs, and I thank Vlad for providing me with opportunities, new perspectives, and great advice. I would also like to thank Juan Garay who has been a great friend, a well-wisher, and someone who always had the best advice for every situation. It has been a pleasure to spend time with him both at work as well as outside of it. I am also grateful to C. Pandu Rangan and Aravind Srinivasan. They had tremendous faith in me and always acted keeping my best interests in mind.

The results in this thesis were obtained in collaboration with a number of brilliant researchers. These include Jonathan Katz, Chiu-Yuen Koo, Dov Gordon, Arkady Yerukhimovich, Arpita Patra, C. Pandu Rangan, Juan Garay, and Hong-Sheng Zhou. I have also had the pleasure of collaborating with several fantastic researchers such as Abhi Shelat, Aggelos Kiayias, Seung Geol Choi, Vladimir Kolesnikov, Abdullatif Shikfa, and Yuval Ishai. I thank all of them for providing me the opportunity to collaborate with them.

I would also like to thank William Gasarch, Samir Khuller, MohammadTaghi HajiAghayi, and Sennur Ulukus for agreeing to be on my dissertation examination committee. They have been very kind to me. Special thanks to Fatima and rest of the department staff for making it easy to deal with all the administrative issues.

I have been blessed with great friends who have provided me good company over several years. Special thanks to Sid and Balaji for always being there for me. I owe a lot to Rajsekar and Pranjal for all their inspirational advice which for some reason had a great impact on me. I would like to thank the whole of Tam Gumbal for providing constant entertainment and breaking the monotony when required. Special thanks to Sadia for several enjoyable and colorful moments.

Last but not least, I would like to thank my parents and my brother for their unconditional support. They have been great role models for me. I thank them for loving me all these years.

Contents

Chapter 1

Introduction and Summary of Results

1.1 Motivation

One of the major achievements of modern cryptography is the design of protocols for general secure multi-party computation [46, 89, 5, 18]. Loosely speaking, protocols for secure multi-party computation (MPC) enable participating parties to correctly compute any functionality while learning their prescribed output and nothing more. An important property of such protocols is that they are robust to the malicious behavior of corrupted parties. In this dissertation, we are primarily concerned with two basic primitives which are central to the design of protocols for secure multi-party computation.

The first primitive is fault-tolerant *broadcast* which, informally, allows a party to distribute a value among a set of mutually mistrusting parties that are connected by pairwise point-to-point channels in a network. A formal requirement of broadcast protocols is that all parties need to agree on the distributed value at the end of the protocol; agreement needs to be guaranteed even when the party who is distributing the value is corrupt. MPC protocols are typically designed assuming the existence of broadcast *channels*. However, in real networks, links are typically only point-to-point, and broadcast channels have to be emulated by a secure protocol for broadcast.

The second primitive is *verifiable secret sharing* (VSS) in which a designated party (dealer) shares a secret among other parties such that the secret is hidden from faulty subsets of parties (which hold their shares of the secret) while simultaneously being reconstructible at a later time when all parties disclose their respective shares of the secret. A scheme for secret sharing is said to be *verifiable* if the reconstructed secret is well-defined at the end of sharing even when the dealer is malicious.

In this dissertation, we investigate the problem of broadcast in stronger threat models, and demonstrate tight feasibility results. We also study round complexity of VSS protocols, and present round-optimal constructions in some settings.

Early versions of the work in this disseration appeared in [49, 40, 64, 65, 50, 69].

1.2 Broadcast

Formally, broadcast protocols allow a designated player (the *dealer*) to distribute an input value to a set of parties such that (1) if the dealer is honest, all honest parties output the dealer's value (**validity**), and (2) even if the dealer is dishonest, the outputs of all honest parties agree (**agreement**). Broadcast forms a critical component of multi-party cryptographic protocols. Indeed such protocols conveniently abstract away various details of the underlying communication network and allow protocol designers to simply assume the existence of a broadcast channel. Later, when the protocol is deployed over a real network, participating parties emulate a broadcast channel by running a secure protocol for broadcast. Known composition results imply that this approach is *sound*: namely, given a protocol Π proven secure under the assumption that a broadcast channel exists, and then instantiating the broadcast channel using a secure broadcast protocol BC, the *composed* protocol Π^{BC} is guaranteed to be secure when run over a point-to-point network.

Early work in the design of broadcast protocols was done in the context of Byzantine agreement (BA) [79, 71]. In the problem of Byzantine agreement (also known as the *consensus* problem), each party has an initial input, and at the end of the protocol all parties have to output a common value. Further, when all honest parties have the same input value, the parties are required to output that value. In the case of honest majority among participating parties, a protocol for Byzantine agreement implies a protocol for broadcast using one additional round. Indeed, a protocol in which the dealer sends its message to all parties in the first round, and then the parties run a Byzantine agreement protocol on the values they received gives a protocol for broadcast. In the reverse direction, a protocol for broadcast yields a protocol for Byzantine agreement when there is a honest majority among participating parties. Indeed, a protocol in which each party runs a broadcast protocol on its input value, and then the parties take a majority among the values received gives a protocol for Byzantine agreement. We remark that the problem of Byzantine agreement is not defined if there is no honest majority.

Classical results of Pease, Shostak, and Lamport [79, 71] show that Byzantine agreement (and, equivalently, broadcast) is achievable in a synchronous network of n parties if and only if the number of corrupted parties t satisfies $t < n/3$. Their protocol, however, required the processors to send exponentially long messages and perform exponentially many steps of computation. Soon after, polynomial-time BA protocols for $t < n/3$ were shown [28, 87], and following a long sequence of works, Garay and Moses [41] showed a polynomial-time BA protocol with optimal resilience and optimal number of rounds.

To go beyond the bound $t < n/3$, some form of set-up is required even if randomization is used [61, 51]. The most commonly studied set-up assumption is the existence of a public-key infrastructure (PKI) such that each party P_i has a public signing key pk_i that is known to all other parties (in addition to the cryptographic assumption that secure digital signatures [48] exist). Protocols designed in this setting are termed *authenticated.* Authenticated broadcast is possible for any $t < n$ [79, 71, 29]. We remark that secure digital signatures can be constructed from the minimal cryptographic assumption that one-way functions exist [75, 85], and the resulting broadcast protocol will be secure against a computationally bounded adversary. Alternatively, if information-theoretic "pseudo-signatures" [81] are used, then the resulting broadcast protocol will be secure even against a computationally unbounded adversary.

In this dissertation, we address two important issues related to broadcast: (1) Almost all existing protocols do not distinguish between corrupted parties (who do not follow the protocol), and honest parties whose secret (signing) keys have been compromised (but who continue to behave honestly). (2) All existing protocols for broadcast are insecure against an adaptive adversary who can choose which parties to corrupt as the protocol progresses. We describe our results next.

1.2.1 Broadcast with a Partially Compromised Public-Key Infrastructure

With few exceptions [37, 52], prior work in the PKI model treats each party as either totally honest, or as completely corrupted and under the control of a single adversary; the assumption is that the adversary cannot forge signatures of any honest parties. However, in many situations it makes sense to consider a middle ground: parties who honestly follow the protocol but whose signatures might be forged (e.g., because their signing keys have been compromised). Most existing work treats any such party P as corrupt, and provides no guarantees for P in this case: the output of P may disagree with the output of other honest parties, and validity is not guaranteed when P is the dealer. Clearly, it would be preferable to ensure agreement and validity for honest parties who have simply had the misfortune of having their signatures forged.

Our Contributions. We consider broadcast protocols providing exactly these guarantees. Specifically, say t_a parties in the network are actively corrupted; as usual, such parties may behave arbitrarily and we assume their actions are coordinated by a single adversary $\mathcal{A}$. We also allow for t_c parties who follow the protocol honestly, but whose signatures can be forged by $\mathcal{A}$; this is modeled by simply giving $\mathcal{A}$ their secret keys. We refer to such honest-behaving parties as *compromised*, and require agreement and validity to hold even for compromised parties.

Say t_a, t_c *satisfy the threshold condition* with respect to some total number of parties n if $2t_a + \min(t_a, t_c) < n$. In Chapter 3, we show:

1. For any n and any t_a, t_c satisfying the threshold condition with respect to n, there is an efficient (i.e., polynomial in n) protocol achieving the notion of broadcast outlined above.
2. When the threshold condition is *not* satisfied, broadcast protocols meeting our notion of security are impossible, with the exception of the "classical" case ($t_c = 0$) where standard results like [29] imply feasibility.
3. Except for a few "exceptional" values of n, there is no *fixed* n-party protocol that tolerates all t_a, t_c satisfying the threshold condition with respect to n. (The positive result mentioned above relies on two different protocols, depending on whether $t_a \le t_c$.) For the exceptional values of n, we show protocols that *do* tolerate any t_a, t_c satisfying the threshold condition.

Taken together, our results provide a complete characterization of the problem.

Early versions of this work appeared in [49, 50].

Related and Subsequent Work. Gupta et al. [52] also consider broadcast protocols providing agreement and validity for honest-behaving parties whose secret keys have been compromised. Our results improve upon theirs in several respects. First, we construct *efficient* protocols whenever $2t_a + \min(t_a, t_c) < n$, whereas the protocols presented in the work of Gupta et al. have message complexity exponential in n. Although Gupta et al. [52] also claim impossibility when $2t_a + \min(t_a, t_c) \ge n$, our impossibility result is simpler and stronger in that it holds relative to a weaker adversary.[1] Finally, Gupta et al. treat t_a, t_c as known and do not consider the question of designing a fixed protocol achieving broadcast for any t_a, t_c satisfying the threshold condition (as we do in the third result mentioned above).

Fitzi et al. [37] consider broadcast in a model where the adversary can either corrupt a few parties and forge signatures of *all* parties, or corrupt more parties but forge no signatures. In our notation, their work handles the two extremes $t_a < n/3$, $t_c = n$ and $t_a < n/2$, $t_c = 0$. We stress that, in contrast to [37], our work addresses the intermediate cases, where an adversary might be able to forge signatures of some honest parties but not others.

Subsequent to our work, Hirt and Zikas [58] also considered the problem of Byzantine agreement when honest parties' signatures may be forged. In their model, the adversary is no longer restricted

[1] In [52], the adversary is assumed to have access to the random coins used by the compromised parties when running the protocol, whereas we do not make this assumption.

by an upper bound on the number of parties to corrupt. Instead, they use the notion of *adversary structures* [60, 56] to model a non-threshold adversary, and obtain a generalized version of our results.

1.2.2 Adaptively Secure Broadcast

While designing cryptographic protocols, it is often assumed that the set of parties that are corrupted by the adversary remains unchanged throughout the execution of the protocol. Such adversaries are said to be *static*. (Indeed this model is standard, and we consider static adversaries in Chapter 3.) Adversaries without this restriction, i.e., adversaries that can choose which parties to corrupt as the protocol is being executed, are known as *adaptive* adversaries. It is easy to see that adaptive security (i.e., security against adaptive adversaries) is strictly stronger than static security (i.e., security against static adversaries) for Byzantine adversaries.

Surprisingly, Hirt and Zikas [57] recently observed that all existing protocols for broadcast (which have been known for over 30 years) are *insecure* against an adaptive adversary who can choose which parties to corrupt as the protocol progresses. Furthermore, they proved that there is no broadcast protocol secure against an adaptive adversary that can corrupt $t > n/2$ parties.

Our Contributions. In Chapter 4, we revisit the problem of adaptively secure broadcast. First, we observe that the Hirt-Zikas attack and impossibility result for adaptively secure broadcast (when $t > n/2$) holds in a communication model that is unrealistically pessimistic and allows the adversary to corrupt a party and then retroactively change messages that party had already sent. We revisit the problem of adaptively secure broadcast in the standard communication model (with rushing[2]). We observe that attacks demonstrated by Hirt and Zikas [57] on existing broadcast protocols, succeed even in this model. This immediately raises the question of whether their impossibility result holds in the communication model we consider.

As our main result, we show that the Hirt-Zikas impossibility result does *not* apply in the standard synchronous model, and demonstrate an adaptively secure broadcast protocol tolerating an arbitrary number of corruptions in this setting under the assumption that a PKI and digital signatures exist. We stress that our protocols do not assume erasure. We also prove that our protocol remains secure even when it is concurrently executed with arbitrarily many other protocols.

We also study the impact of adaptive attacks on secure multi-party computation protocols (where broadcast is commonly used as a subcomponent), and establish the variants of broadcast that are needed in this setting. Interestingly, we show that the full functionality of broadcast is not needed in order to obtain secure MPC for $t \geq n/2$; instead, a weaker form of broadcast — which can be realized even in the Hirt-Zikas communication model — suffices.

A preliminary version of this work appeared in [40].

1.3 Verifiable Secret Sharing (VSS)

In secret sharing [7, 86], there is a *dealer* who shares a secret among a group of n parties during a *sharing phase*. The requirements are that, for some parameter $t < n$, any set of t colluding parties gets no information about the dealer's secret at the end of the sharing phase, yet any set of $t + 1$ parties can recover the dealer's secret in a later *reconstruction phase*. Secret sharing assumes the dealer is honest; *verifiable* secret sharing (VSS) [19] requires in addition that, no matter what a

[2] In the rushing model, the adversary is allowed to choose messages of the corrupted parties in some round *after* seeing messages sent by the honest parties in that round.

cheating dealer does (in conjunction with $t-1$ other colluding parties), there is *some* unique secret to which the dealer is "committed" by the end of the sharing phase.

VSS serves as a fundamental building block in the design of protocols for general secure multiparty computation [5, 84, 46]. VSS can also be viewed as a special case of broadcast, with an additional privacy requirement (at the end of the sharing phase). In particular, VSS implies broadcast. VSS was formally introduced by Chor et al. [19]. One of the main applications of VSS described in their paper was to solve the problem of *simultaenous broadcast*, which is a variant of broadcast where parties simultaneously broadcast their input messages while preserving independence of their inputs. (See also [20, 42, 55, 54].) Variants of VSS, such as moderated VSS [62] or graded VSS [31, 39] are important tools in the design of expected constant-round protocols for broadcast. (See [68] for a detailed discussion.) In the other direction, protocols for VSS, as with general protocols for secure computation, are typically designed assuming the existence of broadcast channels. Indeed, constant-round protocols (over point-to-point networks) for VSS do not exist in a model without broadcast channels. We stress that all our VSS protocols are designed assuming the existence of a broadcast channel.

Protocols for VSS are typically classified by the level of security that they guarantee. *Perfect* VSS, where the security guarantees are unconditional (i.e., they hold even against an unbounded adversary) and hold with probability 1, is known to be possible if and only if $t < n/3$ [5, 27]. *Statistical* VSS, a natural relaxation where privacy and/or correctness may fail to hold with negligible probability (still for unbounded adversary), is known to be possible if and only if $t < n/2$ [83, 84, 23]. Lastly, *computational* VSS, where security guarantees hold with respect to a computationally bounded adversary, is achievable if and only if $t < n/2$ [80, 32, 1]. Protocols for computational VSS typically enjoy better efficiency than protocols for statistical VSS.

The round complexity of VSS protocols, typically defined as the number of rounds in the sharing phase, is an important metric of their efficiency, and has been the subject of intense study. An important line of research, initiated by Gennaro et al. [43], has focused on identifying the *exact* round complexity of VSS protocols and their variants. The focus on round complexity is by no means exclusive to the study of VSS protocols. Indeed, much work has been done on establishing bounds on the round complexity of various cryptographic primitives, including Byzantine agreement [33, 71, 79, 29, 8, 31, 62], zero-knowledge [9, 30, 45, 4, 53, 16, 82], and secure computation [26, 59, 89, 44, 3, 66, 67, 63].

In this dissertation, we investigate two important problems related to the round complexity of VSS: (1) Existing protocols for perfect VSS treat the broadcast channel as being available "for free" and do not attempt to minimize its usage. This approach leads to relatively poor round complexity when such protocols are compiled to run over a point-to-point network. (2) There is a large gap between the lower bound and upper bound on the number of rounds required for statistical VSS when $t < n/2$. We describe our results next.

1.3.1 Round Complexity of Perfect VSS in Point-to-Point Networks

As mentioned earlier, perfect VSS is possible iff $t < n/3$. Work of Gennaro et al. [43] and Fitzi et al. [36] shows that, assuming a broadcast channel, three rounds are necessary and sufficient for efficient VSS. Previous research investigating the round complexity of VSS, has focused on optimizing the round complexity *assuming a broadcast channel is available "for free"*. As argued previously [63, 68], however, if the ultimate goal is to optimize the round complexity of protocols for point-to-point networks (where protocols are likely to be run), then it is preferable to minimize the *number of rounds in which broadcast is used* rather than to minimize the *total number of rounds.*

This is due to the high overhead of emulating a broadcast channel over a point-to-point network: deterministic broadcast protocols require $\Omega(t)$ rounds [33]; known randomized protocols [31, 35, 62] require only $O(1)$ rounds in expectation, but the constant is rather high. (The most round-efficient protocol known [62, 68] requires 18 rounds in expectation for $t < n/3$.)

Moreover, when using randomized broadcast protocols, if *more than one* invocation of broadcast is used then special care must be taken to deal with sequential composition of protocols without simultaneous termination (see [72, 62, 63]), leading to a substantial increase in the round complexity. As a consequence, a constant-round protocol that only uses a *single* round of broadcast is likely to yield a more round-efficient protocol in a point-to-point setting than any protocol that uses *two* rounds of broadcast (even if that protocol uses no additional rounds). Prior work [73, 36, 63, 68] shows that optimal round complexity as well as optimal use of the broadcast channel could each be obtained *individually* for VSS, but it was unknown whether they could be obtained *simultaneously*.

Our Contributions. In Chapter 5, show a 3-round VSS protocol which uses one round of broadcast that is, therefore, optimal in both measures. As a consequence, we obtain a VSS protocol with the best known round complexity in point-to-point networks. Our work also leads to an improvement in the round complexity of the most round-efficient broadcast protocols known [62]. Our protocol is efficient, in that the computation and communication are polynomial in n.

Early versions of this work appeared in [64, 65].

1.3.2 Round Complexity of Statistical VSS with Honest Majority

As mentioned earlier, the round complexity of *perfect* VSS was settled by Gennaro et al. [43] and Fitzi et al. [36]. Gennaro et al. also showed that three rounds are necessary for perfect VSS. The 3-round lower bound of Gennaro et al. was generally believed to apply also to the case of *statistical* VSS. It was therefore relatively surprising when Patra et al. [76] showed that statistical VSS could be realized in *two* rounds for $t < n/3$. They also proved that 2-round statistical VSS is impossible for $t \geq n/3$.

The protocol of Patra et al. does not have optimal resilience, and does not apply when $n/3 \leq t < n/2$. Statistical VSS protocols with optimal resilience were first shown by Rabin and Ben-Or [84, 83]. A more efficient protocol for statistical VSS was shown later by Cramer et al. [23]. Their statistical VSS protocol required 11 rounds, and enjoyed the best known round complexity for the same.

Our Contributions. In Chapter 6, we close the gap between the upper bound and the lower bound for the *exact* round complexity of statistical VSS protocols with optimal resilience. The work of Patra et al. [76] showed that 2-round statistical VSS is impossible for $t \geq n/3$. We show that 3-round statistical VSS is possible iff $t < n/2$. We also give an *efficient* 4-round protocol for $t < n/2$. Our protocols require a 2-round reconstruction phase.

A preliminary version of this work appeared in [69].

Related and Subsequent Work. Statistical VSS protocols have also been designed assuming some form of a setup (e.g., a PKI). In such a setting, statistical VSS protocols are often simpler to design, and enjoy better round complexity [35, 63, 68]. For instance, given a setup, the 11-round protocol of [23] can be immediately collapsed to a 6-round protocol (also shown in [23]), and it is known how to collapse this further into a 4-round protocol [68, 63]. We stress that our protocols do not assume any setup.

As argued previously, the number of broadcast rounds in VSS protocols contributes significantly to their round complexity in point-to-point networks. Given a setup, it is known how to design a

5-round statistical VSS protocol tolerating $t < n/2$ corrupted parties that uses broadcast in only a single round [63, 68]. In the plain model (i.e., without any setup), prior work such as our own, has mainly focused only on the round complexity of statistical VSS protocols when a broadcast channel exists. Recent work by Garay et al. [38], subsequent to our own, addresses this issue and shows a 9-round statistical VSS protocol with optimal resilience that uses broadcast only in three rounds. In contrast, our protocols use broadcast in at least four rounds.

Subsequent to our work, Backes et al. [1] investigated the round complexity of computational VSS (with no additional setup) when $t < n/2$. They obtain an optimal 2-round protocol for the same. Furthermore, they show that their construction can be based on the minimal cryptographic assumption that one-way functions exist.

Chapter 2

Preliminaries

In this chapter, we describe the network model, setup assumptions, and security models. The basic network model and the security model vary slightly across the different problems that we discuss. We take sufficient care to emphasize these differences as we introduce the models. Following this, we present definitions of basic primitives. Finally, we include the description of the Dolev-Strong protocol [29] that we employ as a subroutine in some of our constructions.

2.1 Network Model

We consider the standard setting in which a set of n parties, denoted by $\mathcal{P} = \{P_1, \ldots, P_n\}$, communicate in synchronous rounds via *private* and *authenticated* channels in a fully connected, point-to-point network. If a channel connecting two parties is private, then the adversary cannot learn anything about the messages exchanged between these two parties. If a channel connecting two parties is authenticated, then the adversary cannot modify the messages or introduce a new message such that the party who receives the message believes it to originate from the other party. We remark that authenticated and private channels can be realized using cryptographic primitives such as signatures and encryption.[1] However, for the sake of simplicity, we assume that all channels in our network are *unconditionally* private and authenticated, with the understanding that these guarantees (and the guarantees for our protocols) hold only computationally if cryptographic means are employed to realize them. In Chapters 5 and 6, we further assume the existence of an authenticated broadcast channel that is available to all parties. A broadcast channel allows any party to send the same message to all other parties (and all parties to be assured they have received identical messages) in a single round. We stress that we do not assume a *simultaneous* broadcast channel. More generally, we assume that the adversary is *rushing*, i.e., in a given round, the adversary may wait to see messages sent by other parties before computing its own messages to be sent in the same round.

2.2 Setup Assumptions

Our protocols for verifiable secret sharing, presented in Chapters 5 and 6, do not require any setup assumptions (other than a physical broadcast channel). However, our protocols for broadcast, presented in Chapters 3 and 4, are constructed assuming a public-key infrastructure (PKI) and the existence of secure digital signatures [48]. We give more details below.

[1] For adaptive adversaries, private channels may be realized using adaptively secure encryption.

A public-key infrastructure is established as follows: each party P_i runs a key-generation algorithm Gen (specified by the protocol) to obtain public key pk_i along with the corresponding secret key sk_i. Then all parties begin running the protocol holding the same vector of public keys $(pk_1, \ldots, pk_n)$, and with each P_i holding sk_i. We note that malicious parties can generate their keys in an arbitrary fashion, e.g., dependent on the keys generated by parties.

In our protocols, we interpret the public key pk_i as a verification key for a secure digital signature scheme, and the secret key sk_i as the corresponding signing key. Our protocols for broadcast make extensive use of the signature scheme. For the sake of clarity, we often omit additional information that must be signed along with the messages sent across the network. This includes (a) the identity of the recipient, (b) the current round number, (c) an identifier for the message in case multiple messages are sent to the same recipient in the same round, and (d) an identifier for the subprotocol in case multiple subprotocols are being run. Thus, in general when we say a party signs a message, we implicitly assume that the party generates a signature over the message concatenated with the additional information described above. This information is also verified at the time of verification of the signature.

2.3 Security Model

Our constructions are resilient against a centralized *Byzantine* (also called active) adversary, typically denoted $\mathcal{A}$. We say that a protocol tolerates t malicious parties if it is secure against a centralized Byzantine adversary that can corrupt up to t parties, and coordinate the actions of these parties. A corrupted party, under the control of the adversary, might deviate from the prescribed protocol arbitrarily. We say that a party is *honest* if it not corrupted.

It is possible to consider a stronger security model where an adversary decides which parties to corrupt depending on the protocol execution. Such an adversary is called an *adaptive* adversary. In contrast, an adversary that can corrupt only a fixed (but unknown) set of parties in advance of protocol execution is termed *static*. We stress that our adaptively secure protocols do not assume erasure. Another strengthening of the security model allows an adversary to coordinate its actions across multiple protocols that are concurrently executed among the same (or a subset of the) parties. Protocols that remain secure when composed with other arbitrarily many protocols are said to be *universally composable* [11].

We give a brief overview of the exact security guarantees that our results provide. Our constructions in Chapter 3 are demonstrably *insecure* against an adaptive adversary. Indeed, our main contribution there is an impossibility result against static adversaries, which obviously rules out constructions against adaptive adversaries as well. In Chapter 4, we design universally composable broadcast protocols secure against an adaptive adversary. Our protocol for perfect verifiable secret sharing presented in Chapter 5 is proven secure against an adaptive adversary. Using known composition results for perfectly secure protocols [70, 13], we conclude that this protocol also achieves security under universal composition. Our protocols for statistical verifiable secret sharing presented in Chapter 6 are secure in the presence of a static adversary. We conjecture that our constructions are universally composable against adaptive adversaries as well.

2.4 Basic Primitives

In this section we present traditional property-based definitions of broadcast and verifiable secret sharing. We start with the formal definition of broadcast.

Definition 1 (Broadcast) *A protocol for parties $\mathcal{P} = \{P_1, \ldots, P_n\}$, where a distinguished dealer $D \in \mathcal{P}$ holds an initial input m in some message space $\mathcal{M}$, achieves* broadcast *if the following hold (except with negligible probability):*

Agreement *All honest parties output the same value.*

Validity *If the dealer is honest, then all honest parties output m.* $\diamondsuit$

Observe that the above definition refers to an arbitrary input m for the dealer. However, while designing protocols, we assume for simplicity that the message space $\mathcal{M}$ for the dealer's input is $\{0,1\}$. Broadcast for arbitrary message spaces can be obtained from binary broadcast using standard techniques [88].

We now present definitions of VSS and its variants. We require the dealer's secret, typically denoted by s, to lie in some finite field $\mathbb{F}$.

First, we present the definition of perfect *weak* verifiable secret sharing (WSS).

Definition 2 (Perfect weak verifiable secret sharing) *A two-phase protocol for parties $\mathcal{P} = \{P_1, \ldots, P_n\}$, where a distinguished dealer $D \in \mathcal{P}$ holds initial input $s \in \mathbb{F}$, is a* WSS protocol tolerating t malicious parties *if the following conditions hold for any adversary controlling at most t parties:*

Privacy *If the dealer is honest at the end of the first phase (the* sharing phase*), then at the end of this phase the joint view of the malicious parties is independent of the dealer's input s.*

Correctness *Each honest party P_i outputs a value s_i at the end of the second phase (the* reconstruction phase*). If the dealer is honest then $s_i = s$.*

Weak commitment *At the end of the sharing phase the joint view of the honest parties defines a value s' (which can be computed in polynomial time from this view) such that each honest party will output either s' or a default value $\perp$ at the end of the reconstruction phase.* $\diamondsuit$

Next, we present the definition of perfect VSS below.

Definition 3 (Perfect verifiable secret sharing) *A two-phase protocol for parties $\mathcal{P} = \{P_1, \ldots, P_n\}$, where a distinguished dealer $D \in \mathcal{P}$ holds initial input $s \in \mathbb{F}$, is a* VSS protocol tolerating t malicious parties *if it satisfies the privacy and correctness requirements of WSS as well as the following (stronger)* commitment *requirement:*

Commitment *At the end of the sharing phase the joint view of the honest parties defines a value s' (which can be computed in polynomial time from this view) such that all honest parties will output s' at the end of the reconstruction phase.* $\diamondsuit$

Next, we present the definition of perfect VSS with 2-level sharing. Such VSS protocols constitute a useful building block for protocols for general secure multi-party computation (see, e.g., [63, 68]).

Definition 4 (Perfect verifiable secret sharing with 2-level sharing) *A two-phase protocol for parties $\mathcal{P} = \{P_1, \ldots, P_n\}$, where a distinguished dealer $D \in \mathcal{P}$ holds initial input $s \in \mathbb{F}$, is a* VSS protocol with 2-level sharing tolerating t malicious parties *if it satisfies the privacy and correctness requirements of VSS as well as the following requirement:*

Commitment with 2-level sharing *At the end of the sharing phase each honest party P_i outputs s_i and $s_{i,j}$ for $j \in [n]$, satisfying the following requirements:*

1. *There exists a polynomial $p(x)$ of degree at most t such that $s_i = p(i)$ for every honest party P_i, and furthermore all honest parties will output $s' = p(0)$ at the end of the reconstruction phase.*
2. *For each $j \in [n]$, there exists a polynomial $p_j(x)$ of degree at most t such that (1) $p_j(0) = p(j)$ and (2) $s_{i,j} = p_j(i)$ for every honest party P_i.* ♢

This implies the commitment property of VSS, since the value $s' = p(0)$ that will be output in the reconstruction phase is defined by the view of the honest parties at the end of the sharing phase.

In the case of statistical VSS, we allow *error* with probability at most $O(1/|\mathbb{F}|)$, so $\log |\mathbb{F}|$ is the security parameter. Note that the dealer's secret can be padded to lie in a larger field, if desired, to reduce the probability of error.

Our definition of statistical VSS relaxes the *correctness/commitment* requirement, but not the *privacy* requirement. This is the definition that has been considered previously in the literature, and is the definition that our protocols in Chapter 6 achieve.

Definition 5 (Statistical Verifiable Secret Sharing) *Let λ be a statistical security parameter, and let $\epsilon = 2^{-\Theta(\lambda)}$. A two-phase protocol for parties $\mathcal{P} = \{P_1, \ldots, P_n\}$, where a distinguished dealer $D \in \mathcal{P}$ holds initial input $s \in \mathbb{F}$, is a* statistical VSS protocol tolerating t malicious parties *if the following conditions hold for any adversary controlling at most t parties:*

Privacy *If the dealer is honest at the end of the first phase (the* sharing phase*), then at the end of this phase the joint view of the malicious parties is independent of the dealer's input s.*

Correctness *Each honest party P_i outputs a value s_i at the end of the second phase (the* reconstruction phase*). If the dealer is honest, then except with probability at most ϵ, it holds that $s_i = s$.*

Commitment *Except with probability at most ϵ, the joint view of the honest parties at the end of the sharing phase defines a value s' such that $s_i = s'$ for every honest P_i.* ♢

2.5 The Dolev-Strong Protocol

We present a modified version of the Dolev-Strong [29] protocol for authenticated broadcast for $t < n$. (See Figure 2.1.) The protocol assumes the existence of a public-key infrastructure (PKI) and digital signature schemes secure against adaptive chosen-message attacks [48]. The protocol is as secure as the signature scheme employed.

We note that the protocol presented in [29] is recursive, and requires parties to sign over signatures of other parties. We present a simpler version of the Dolev-Strong protocol that requires parties only to produce signatures over messages from the message space [34, 81].

At a high level, the protocol described in Figure 2.1 works as follows. Each party P_i maintains an accepted set of values, denoted by ACC_i. Furthermore, each party P_i also collects signatures on the message 0 in SET_0^i, and on the message 1 in SET_1^i. At the end of each round, each party P_i checks whether the size of the sets $\mathsf{SET}_0^i, \mathsf{SET}_1^i$ exceeds a given threshold in which case they update the set of accepted values ACC_i. Finally, each party P_i computes an output depending on ACC_i and terminates the protocol.

Protocol DS_ϕ

The protocol is carried out among a set $\mathcal{P} = \{P_1, \ldots, P_n\}$ of parties. We let $D \in \mathcal{P}$ denote the sender. We let $m \in \{0,1\}$ denote D's input, and sk_D its secret key.

- **Stage 1:** D sends $(m, \{\mathsf{Sign}_{sk_D}(m)\})$ to every party. It then outputs m and terminates the protocol. Each party initializes $\mathsf{ACC}_i = \emptyset$, $\mathsf{SET}_0^i = \mathsf{SET}_1^i = \emptyset$.
- **Stage 2:** In rounds $r = 1, \ldots, \phi + 1$, execute the following.
  - If a pair (v, SET) is received from some P_j, with $v \in \{0,1\}$, and if SET contains valid signatures on v from at least r distinct parties including the dealer D, then P_i updates $\mathsf{ACC}_i = \mathsf{ACC}_i \cup \{v\}$, and $\mathsf{SET}_v^i = \mathsf{SET}_v^i \cup \mathsf{SET}$.
  - Each party P_i checks whether any value $v \in \{0,1\}$ was *newly* added to the set of accepted values ACC_i during round $r - 1$. In this case, P_i computes $\mathsf{Sign}_{sk_i}(v)$, and sends $(v, \mathsf{SET}_v^i \cup \{\mathsf{Sign}_{sk_i}(v)\})$ to every other party.
- **Stage 3:** If $\mathsf{ACC}_i = \{1\}$, then P_i outputs $m_i = 1$, else it outputs $m_i = 0$.

Figure 2.1: The Dolev-Strong protocol for broadcast.

Theorem 1 *[29] Suppose $\mathcal{A}$ corrupts at most t parties among $\mathcal{P}$. Then, if $n > t$ and $\phi = t$, protocol DS_ϕ achieves broadcast.*

Proof Suppose D is honest with input m then we argue that for every honest P_i, it holds that $\mathsf{ACC}_i = \{m\}$. In fact, at the end of round 1, each P_i will receive $(m, \mathsf{Sign}_{sk_D}(m))$ from D, and thus will update ACC_i to contain m. Now since an honest D's signature cannot be forged, any pair $(1 - m, \mathsf{SET})$ received from any party P_j, will never have SET contain a valid signature from D. Consequently, ACC_i will never be further updated to contain $1 - m$. Thus, every honest P_i will output $m_i = m$, and we conclude that validity holds for an honest D.

Next, we need to show that agreement holds (even when D is dishonest). Assume that P_i and P_j are honest. We will show that $\mathsf{ACC}_i = \mathsf{ACC}_j$ holds at the end of the protocol. Without loss of generality, let some value $v \in \mathsf{ACC}_i$ be first accepted by P_i in some round, say r. Observe that at this point, i.e., at the end of round r, the set SET_v^i does not contain signatures from P_i, but contains r signatures from distinct parties including the dealer D.

If $r \leq \phi$, then in round $r + 1$ party P_i sends $(v, \mathsf{SET}_v^i \cup \{\mathsf{Sign}_{sk_i}(v)\})$ to all parties including P_j. Clearly, $\mathsf{SET}_v^i \cup \{\mathsf{Sign}_{sk_i}(v)\}$ contains at least $r + 1$ signatures from distinct parties including dealer D, and P_j will update ACC_j to contain v.

On the other hand, when $r = \phi + 1 > t$, then at the end of round r, SET_v^i must contain a signature from at least one honest party, say P_k. Since honest P_k's signature cannot be forged, this implies that P_k must have sent $(v, \mathsf{SET}_v^k \cup \{\mathsf{Sign}_{sk_k}(v)\})$ to all parties in some round $r' \leq r$. In particular, this implies that ACC_k must contain v and that SET_v^k must contain at least $r' - 1$ valid signatures from distinct parties including dealer D. Thus, in round r', both P_i and P_j would have received $(v, \mathsf{SET} = \mathsf{SET}_v^k \cup \{\mathsf{Sign}_{sk_k}(v)\})$ from honest P_k. As SET contains r' valid signatures from distinct parties including dealer D, parties P_i and P_j must have added v to ACC_i and ACC_j respectively. (We note that $r' = r$ holds in order to avoid a contradiction.) We conclude that whenever some value v is contained in ACC_i, it is also contained in ACC_j. Thus $\mathsf{ACC}_i = \mathsf{ACC}_j$ holds at the end of the protocol, and agreement follows. ■

Chapter 3

Broadcast with a Partially Compromised Public-Key Infrastructure

In this chapter, we give protocols for authenticated broadcast when the underlying public-key infrastructure (PKI) is compromised. We assume a public-key infrastructure (PKI), established as follows: each party P_i runs a key-generation algorithm Gen (specified by the protocol) to obtain public key pk_i along with the corresponding secret key sk_i. Then all parties begin running the protocol holding the same vector of public keys $(pk_1, \ldots, pk_n)$, and with each P_i holding sk_i.

We model the failure of the PKI by allowing the adversary to learn some of the secret keys corresponding to honest parties. More formally, we divide the set of honest parties into those who have been *compromised* and those who have not been compromised. If honest party P_i is compromised then the adversary $\mathcal{A}$ is given that P_i's secret key sk_i. We stress that compromised parties follow the protocol as instructed: the only difference is that $\mathcal{A}$ is now able to forge signatures on their behalf. On the other hand, we assume $\mathcal{A}$ is unable to forge signatures of any honest parties who have *not* been compromised. We assume authenticated point-to-point channels between *all* honest parties, even those who have been compromised. In other words, although the adversary can forge the signature of an honest party P_i who has been compromised, it cannot falsely inject a point-to-point message on P_i's behalf.

With few exceptions [37, 52], most existing work treats any such compromised party P_i as corrupt, and provides no guarantees for P_i in this case: the output of P_i may disagree with the output of other honest parties, and validity is not guaranteed when P_i is the dealer. As a concrete example, consider the Dolev-Strong protocol presented in Section 2.5. Clearly, if D is compromised, then $\mathcal{A}$ can generate $\mathsf{Sign}_{sk_D}(m')$ on its own for any $m' \neq m$. Now consider an adversarial strategy in which a corrupted party sends $(m', \mathsf{Sign}_{sk_D}(m'))$ to all parties in round 1. It is easy to verify that this simple adversarial strategy is sufficient to violate validity when D is compromised.

We argue that compromised parties are most naturally viewed as honest parties whose secret (signing) keys have been obtained somehow by the adversary. E.g., perhaps an adversary was able to hack into an honest user's system and obtain their secret key, but subsequently the honest party's computer was re-booted and now behaves honestly. Clearly, it would be preferable to ensure agreement and validity for honest parties who have simply had the misfortune of having their signatures forged.

This motivates the following definition of the problem of authenticated broadcast in our setting and our modeling of corruptions:

Definition 6 *A protocol for parties $\mathcal{P} = \{P_1, \ldots, P_n\}$, where a distinguished dealer $D \in \mathcal{P}$ holds an initial input m, is an* authenticated broadcast protocol *if the following hold (except with negligible probability):*

Agreement *All honest parties output the same value.*

Validity *If the dealer is honest, then all honest parties output m.*

We stress that "honest" in the above includes those honest parties who have been compromised. $\diamondsuit$

We model our adversary in the following way. An adversary $\mathcal{A}$ is called a (t_a, t_c)-*adversary* if $\mathcal{A}$ actively corrupts up to t_a parties and additionally compromises up to t_c of the honest parties. In a network of n parties, we call $\mathcal{A}$ a *threshold adversary* if $\mathcal{A}$ chooses t_a, t_c subject to the restriction $2t_a + \min(t_a, t_c) < n$; actively corrupts up to t_a parties; and compromises up to t_c honest parties.

Our results. In the model described above, we obtain tight feasibility results for the problem of authenticated broadcast when the PKI is compromised. Say n, t_a, t_c *satisfy the threshold condition* if $2t_a + \min(t_a, t_c) < n$. We show:

1. For every n, t_a, t_c satisfying the threshold condition, there exists an efficient authenticated broadcast protocol.
2. Authenticated broadcast is impossible whenever t_a, t_c do not satisfy the threshold condition (except when t_c is fixed to 0).
3. Except for a few "exceptional" values of n, there does not exist a single, fixed protocol achieving authenticated broadcast for all t_a, t_c satisfying the threshold condition.
4. We give positive results for the exceptional values of n.

Taken together, our results provide a complete characterization of the problem.

Organization. In Section 3.1 we show a protocol for broadcast when $2t_a + \min(t_a, t_c) < n$ holds and the values of t_a, t_c are known to all parties. We show our impossibility results in Section 3.2. In Section 3.3 we give positive results for the exceptional values of n. Finally, in Section 3.4, we give a protocol that is more efficient than the one in Section 3.3, however the protocol works under the assumption that at least one party in the network is honest and not compromised.

3.1 Broadcast for (t_a, t_c)-Adversaries

In this section, we prove the following result:

Theorem 2 *Fix n, t_a, t_c with $2t_a + \min(t_a, t_c) < n$. Then there exists a protocol achieving broadcast in the presence of a (t_a, t_c)-adversary.*

The case of $t_a \leq t_c$ is easy: $t_a \leq t_c$ implies $3t_a < n$ and the parties can run a standard (unauthenticated) broadcast protocol [79, 71] where the PKI is not used at all. (In this case, it makes no difference whether honest parties are compromised or not.) The challenge is to design a protocol for $t_c < t_a$, and we deal with this case for the remainder of this section.

Let DS_ϕ refer to the Dolev-Strong protocol [29] that achieves broadcast with a PKI, in the usual sense (i.e., when no honest parties' keys can be compromised) when $\phi = t$, for any $t < n$ corrupted parties. (The Dolev-Strong protocol is presented in Section 2.5.) We prove a slightly stronger version of Theorem 1.

Lemma 3 *Suppose $\mathcal{A}$ corrupts at most t_a parties, and further compromises at most t_c parties. Then, if $n > t_a + t_c$ and $\phi = t_a + t_c$, protocol DS_ϕ achieves the following:*

- *If D is honest and non-compromised, then validity holds.*
- *Agreement holds. Further, if P_i and P_j are honest, then $\mathsf{ACC}_i = \mathsf{ACC}_j$ holds at the end of the protocol.*
- *If D is honest, and if $|\mathsf{ACC}_i| = 1$ holds for some honest party P_i, then validity holds.*

Proof Suppose D is honest and non-compromised with input m then we argue that for every honest P_i, it holds that $\mathsf{ACC}_i = \{m\}$. In fact, at the end of round 1, each P_i will receive $(m, \mathsf{Sign}_{sk_D}(m))$ from D, and thus will update ACC_i to contain m. Now since an honest and non-compromised D's signature cannot be forged, any pair $(1 - m, \mathsf{SET})$ received from any party P_j, will never have SET contain a valid signature from D. Consequently, ACC_i will never be further updated to contain $1 - m$. Thus, every honest P_i will output $m_i = m$, and we conclude that validity holds for an honest and non-compromised D.

Next, we need to show that agreement holds (even when D is dishonest). Assume that P_i and P_j are honest. We will show that $\mathsf{ACC}_i = \mathsf{ACC}_j$ holds at the end of the protocol. Without loss of generality, let some value $v \in \mathsf{ACC}_i$ be first accepted by P_i in some round, say r. Observe that at this point, i.e., at the end of round r, the set SET^i_v does not contain signatures from P_i, but contains r signatures from distinct parties including the dealer D.

If $r \leq \phi$, then in round $r + 1$ party P_i sends $(v, \mathsf{SET}^i_v \cup \{\mathsf{Sign}_{sk_i}(v)\})$ to all parties including P_j. Clearly, $\mathsf{SET}^i_v \cup \{\mathsf{Sign}_{sk_i}(v)\}$ contains at least $r + 1$ signatures from distinct parties including dealer D, and P_j will update ACC_j to contain v.

On the other hand, when $r = \phi + 1 > t_a + t_c$, then at the end of round r, SET^i_v must contain a signature from at least one honest and non-compromised party, say P_k. Since honest and non-compromised P_k's signature cannot be forged, this implies that P_k must have sent $(v, \mathsf{SET}^k_v \cup \mathsf{Sign}_{sk_k}(v))$ to all parties in some round $r' \leq r$. In particular, this implies that ACC_k must contain v and that SET^k_v must contain at least $r' - 1$ valid signatures from distinct parties including dealer D. Thus, in round r', both P_i and P_j would have received $(v, \mathsf{SET} = \mathsf{SET}^k_v \cup \{\mathsf{Sign}_{sk_k}(v)\})$ from honest P_k. As SET contains r' valid signatures from distinct parties including dealer D, parties P_i and P_j must have added v to ACC_i and ACC_j respectively. (We note that $r' = r$ holds in order to avoid a contradiction.) We conclude that whenever some value v is contained in ACC_i, then it is also contained in ACC_j. Thus $\mathsf{ACC}_i = \mathsf{ACC}_j$ holds at the end of the protocol, and agreement follows.

Finally, consider the case when D is honest but compromised with input m. In this case, at the end of round 1, each P_i will receive $(m, \mathsf{Sign}_{sk_D}(m)$ from D, and thus will update ACC_i to contain m. Observe that once a value is added to ACC_i it is never deleted. Thus, if $|\mathsf{ACC}_i| = 1$ holds at the end of the protocol then it must hold that $\mathsf{ACC}_i = \{m\}$. Recall that when P_i and P_j are honest, it holds that $\mathsf{ACC}_i = \mathsf{ACC}_j$. Therefore, for every honest P_j, it must also hold that $\mathsf{ACC}_j = \{m\}$, and validity follows. ■

Protocol 1

The protocol is carried out among a set $\mathcal{P} = \{P_1, \ldots, P_n\}$ of parties. For notational convenience, we let D denote the dealer (though in fact any party can act as dealer). We let $m \in \{0, 1\}$ denote D's input. In the following, we use DS to denote an execution of DS_ϕ with $\phi = t_a + t_c$.

- **Stage 1:** D sends m to all other parties. Let m'_i be the value received by P_i from D in this step (if the dealer sends nothing to P_i, then m'_i is taken to be some default value).
- **Stage 2:** In parallel, each party P_i acts as the dealer in an execution of $\mathsf{DS}(m'_i)$ (the original dealer D runs $\mathsf{DS}(m)$). We let $|\mathsf{CLEAN}_0|$ (resp., $|\mathsf{CLEAN}_1|$) denote the number of executions of DS that are both *clean* and result in output 0 (resp., 1). (Note that all honest parties agree on sets $\mathsf{CLEAN}_0, \mathsf{CLEAN}_1$.)
- **Stage 3:** If $|\mathsf{CLEAN}_0| \geq |\mathsf{CLEAN}_1|$ then each P_i sets $m_i = 0$; otherwise, P_i sets $m_i = 1$. Each P_i outputs m_i.

Figure 3.1: A broadcast protocol for $t_c < t_a$ and $2t_a + t_c < n$.

We say that party P_i declares an execution of DS_ϕ *clean* if $|\mathsf{ACC}_i| = 1$, and *dirty* otherwise. From the proof above, we conclude that honest parties agree on whether an execution of DS_ϕ is clean (or dirty) when $n > t_a + t_c$ and $\phi = t_a + t_c$.

Observe that DS_ϕ fails to satisfy Definition 6 only when the dealer is *honest* but *compromised.* Our protocol (cf. Figure 3.1) guarantees validity even in this case (while leaving the other cases unaffected). In our protocol description, and in the rest of the chapter, we will always set $\phi = t_a + t_c$. Simplifying the notation, we use DS to denote an execution of DS_ϕ with $\phi = t_a + t_c$.

Theorem 4 *Let $\mathcal{A}$ be a (t_a, t_c)-adversary with $t_c < t_a$ and $2t_a + t_c < n$. Then Protocol 1 achieves broadcast in the presence of $\mathcal{A}$.*

Proof We prove agreement and validity. Note that $n > t_a + t_c$, so Lemma 3 applies.

Agreement: By Lemma 3, the output of each honest party is the same in every execution of DS in the second stage of the protocol, and all honest parties agree on whether any given execution of DS is clean or dirty. So all honest parties agree on $|\mathsf{CLEAN}_0|$ and $|\mathsf{CLEAN}_1|$, and agreement follows.

Validity: Assume the dealer is honest (whether compromised or not). Letting t_h denote the number of honest, non-compromised parties, we have $t_h + t_a + t_c = n > 2t_a + t_c$ and so $t_h > t_a$. Thus, there are t_h honest and non-compromised parties that act as dealers in the second stage of Protocol 1, and each of these parties runs $\mathsf{DS}(m)$ where m is the initial input of D. By Lemma 3, all honest parties output m in (at least) these t_h executions, and each of these t_h executions is clean. Furthermore, there can be at most t_a clean executions resulting in output $1 - m$, as only adversarial parties will possibly run $\mathsf{DS}(1 - m)$ in the second stage. The majority value output by the honest parties is therefore always equal to the original dealer's input m. ■

3.2 Impossibility Results

In this section we show two different impossibility results. First, we show that there is no protocol achieving broadcast in the presence of a (t_a, t_c)-adversary when $n \leq 2t_a + \min(t_a, t_c)$ and $t_c > 0$, thus proving that Theorem 2 is tight. We then consider the case when t_a, t_c are not fixed, but instead all that is guaranteed is that $2t_a + \min(t_a, t_c) < n$. (In the previous section, unauthenticated broadcast was used to handle the case $t_a \leq t_c$ and Protocol 1 assumed $t_c < t_a$. Here we seek a *single* protocol that handles both cases.) We show that in this setting, broadcast is impossible for almost all n.

3.2.1 The Three-Party Case

We first present a key lemma involving three parties that will be useful for the proofs of both results described above. Our presentation of the three-party impossibility proof closely follows the presentation in [34] which considers a different security model. The explanatory figures in the proof are inspired by suggestions from an anonymous reviewer, and the presentation in [58], a subsequent work that generalizes our results to the non-threshold setting.

Lemma 5 *Broadcast among $n = 3$ parties $\{P_1, P_2, P_3\}$ with P_1 acting as dealer is not achievable in the presence of an adversary $\mathcal{A}$ that can choose to corrupt in one of the following ways:*

- *$\mathcal{A}$ actively corrupts P_1, and compromises the secret key of no other honest party.*
- *$\mathcal{A}$ actively corrupts P_2, and compromises the secret key of P_1.*
- *$\mathcal{A}$ actively corrupts P_3, and compromises the secret key of P_1.*

Proof Suppose, towards a contradiction, that there exists a protocol Π for achieving broadcast in the presence of adversary $\mathcal{A}$ as specified in the statement of the lemma. Let Π_1, Π_2, Π_3 denote the program specified by protocol Π for parties P_1, P_2, P_3 respectively. (See Figure 3.2.)

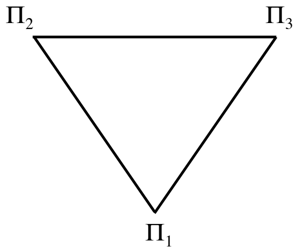

Figure 3.2: The real network among parties P_1, P_2, P_3 where protocol Π is executed.

We analyze the protocol Π in a modified network shown in Figure 3.3. In the modified network, two *independent* identical copies of program Π_1, namely $\Pi_{1,0}$ and $\Pi_{1,1}$ are run in place of Π_1 in the following way. All messages sent by program $\Pi_{1,0}$ to Π_3 are discarded, while all messages it sends to Π_2 are correctly delivered to Π_2. Similarly, all messages sent by program $\Pi_{1,1}$ to Π_2 are discarded, while all messages it sends to Π_3 are correctly delivered to Π_3. Next, all messages sent by program Π_2 to Π_1 in the real network are now routed to program $\Pi_{1,0}$ in the modified network. In particular, $\Pi_{1,1}$ does not receive any messages from Π_2 in the modified network. Similarly, all messages sent by program Π_3 to Π_1 in the real network are now routed to program $\Pi_{1,1}$ in the modified network, and $\Pi_{1,0}$ does not receive any messages from Π_3. In other words, programs $\Pi_{1,0}$ and Π_3 never exchange any messages in the modified network. Similarly, programs $\Pi_{1,1}$ and Π_2 never exchange any messages in the modified network. Finally, programs Π_2 and Π_3 interact with each other in the modified network in the same way as they would in the real network.

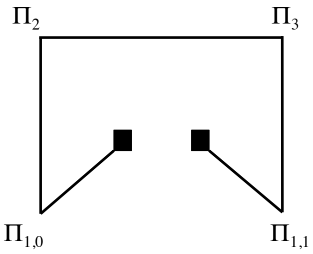

Figure 3.3: The modified network with two copies $\Pi_{1,0}, \Pi_{1,1}$ of P_1's program Π_1. Note that the modified network disables any interaction (represented by the black square at the end of the links) between $\Pi_{1,0}$ and P_3, and also between $\Pi_{1,1}$ and P_2.

We stress that the secret (and public) key used by each of the programs $\Pi_1, \Pi_{1,0}$, and $\Pi_{1,1}$ is identical to the secret (and public) key used by P_1. We are now ready to prove some relationships between executions of Π in the real network that happen in the presence of $\mathcal{A}$, and executions of Π in the modified network where each program behaves honestly.

Claim 6 *There exists an adversary $\mathcal{A}$ that actively corrupts P_1 such that the following holds for every protocol Π and for every pair of public/secret keys held by P_2 and P_3. The joint view of P_2 and P_3 interacting with programs $\Pi_{1,0}$ and $\Pi_{1,1}$ in an execution of Π in the modified network is* perfectly indistinguishable *from the joint view of P_2 and P_3 interacting with $\mathcal{A}$ in an execution of Π in the real network.*

Proof An adversary $\mathcal{A}$ corrupting P_1 obviously has access to its secret keys, and thus access to the secret keys used by programs $\Pi_{1,0}$ and $\Pi_{1,1}$. Now, instead of executing program Π_1 in the real network, $\mathcal{A}$ perfectly simulates programs $\Pi_{1,0}$ and $\Pi_{1,1}$ as shown in Figure 3.4. More concretely, $\mathcal{A}$ internally runs (independent) programs $\Pi_{1,0}$ and $\Pi_{1,1}$ in the following way. $\mathcal{A}$ simulates $\Pi_{1,0}$ such that $\Pi_{1,0}$ exchanges messages only with P_2. Similarly, $\mathcal{A}$ simulates $\Pi_{1,1}$ such that $\Pi_{1,1}$ exchanges messages only with P_3. The claim follows immediately. ■

Claim 7 *There exists an adversary $\mathcal{A}$ that actively corrupts P_2 and compromises the secret key of P_1 such that the following holds for every protocol Π and for every pair of public/secret keys held by P_1 and P_3. The joint view of P_1 and P_3 interacting with programs Π_2 and $\Pi_{1,0}$ in an execution of Π in the modified network is* perfectly indistinguishable *from the joint view of P_1 and P_3 interacting with $\mathcal{A}$ in an execution of Π in the real network.*

Proof An adversary $\mathcal{A}$ that compromises the secret key of P_1 obviously has access to the secret key used by program $\Pi_{1,0}$. Now, instead of allowing program Π_2 to interact with P_1 in the real network, $\mathcal{A}$ perfectly simulates programs $\Pi_{1,0}$ (using P_1's secret key) and Π_2 (using P_2's secret key) as if they are run in the modified network (see Figure 3.5). More concretely, acting as P_2, adversary $\mathcal{A}$ discards any message received from Π_1, and instead internally runs program $\Pi_{1,0}$ and

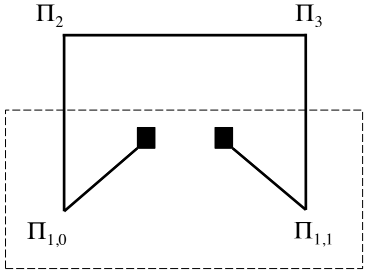

Figure 3.4: The adversary corrupting P_1 simulates $\Pi_{1,0}$ and $\Pi_{1,1}$ interacting with P_2 and P_3 in the real network.

simulates its interaction with Π_2. In $\mathcal{A}$'s simulation, every message sent by Π_2 to Π_1 is now routed to $\Pi_{1,0}$, and every message sent by $\Pi_{1,0}$ to Π_3 is simply discarded by $\mathcal{A}$. Given this, observe that an execution of Π in the real network exactly mimics an execution of Π in the modified network. The claim follows immediately. ■

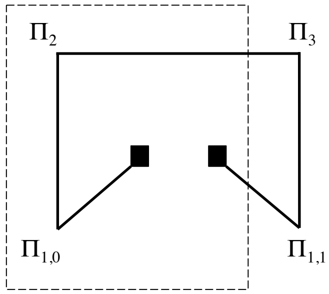

Figure 3.5: The adversary corrupting P_2 and compromising P_1's keys simulates $\Pi_{1,0}$ (using P_1's secret key) and Π_2 (using P_2's secret key) interacting with P_1 and P_3 in the real network.

Claim 8 *There exists an adversary $\mathcal{A}$ that actively corrupts P_3 and compromises the secret key of P_1 such that the following holds for every protocol Π and for every pair of public/secret keys held by P_1 and P_2. The joint view of P_1 and P_2 interacting with programs Π_3 and $\Pi_{1,0}$ in an execution of Π in the modified network is* perfectly indistinguishable *from the joint view of P_1 and P_2 interacting with $\mathcal{A}$ in an execution of Π in the real network.*

Proof An adversary $\mathcal{A}$ that compromises the secret key of P_1 obviously has access to the secret key used by program $\Pi_{1,0}$. Now, instead of allowing program Π_3 to interact with P_1 in the real network, $\mathcal{A}$ perfectly simulates programs $\Pi_{1,1}$ (using P_1's secret key) and Π_3 (using P_3's secret key) as if they are run in the modified network (see Figure 3.6). More concretely, acting as P_3, adversary $\mathcal{A}$ discards any message received from Π_1, and instead internally runs program $\Pi_{1,1}$ and simulates its interaction with Π_3. In $\mathcal{A}$'s simulation, every message sent by Π_3 to Π_1 is now routed to $\Pi_{1,1}$, and every message sent by $\Pi_{1,1}$ to Π_2 is simply discarded by $\mathcal{A}$. Given this, observe that an execution of Π in the real network exactly mimics an execution of Π in the modified network. The claim follows immediately. ■

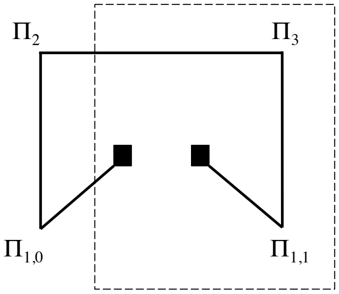

Figure 3.6: The adversary corrupting P_3 and compromising P_1's keys simulates $\Pi_{1,1}$ (using P_1's secret key) and Π_3 (using P_3's secret key) interacting with P_1 and P_2 in the real network.

We will now focus on executions of Π in the modified network. Recall that in the modified network, none of the four programs $\Pi_{1,0}, \Pi_{1,1}, \Pi_2$, and Π_3 are corrupted. Further, observe that the modified network has two dealers each running $\Pi_{1,0}$ and $\Pi_{1,1}$ respectively. Aiming for a contradiction, suppose that one copy of Π_1, say $\Pi_{1,0}$, is initialized with input 0, and the other copy, i.e., $\Pi_{1,1}$, is initialized with input 1. Note that each of the four programs running in the modified network will ultimately terminate outputting some bit (or a default value). Thus for every execution of Π in the modified network, there must be one set of adjacent programs, i.e., one set among $S_3 = \{\Pi_{1,0}, \Pi_2\}, S_1 = \{\Pi_2, \Pi_3\}, S_2 = \{\Pi_3, \Pi_{1,1}\}$ such that validity or agreement does not hold for parties executing programs in that set. This implies that there exists a set $S \in \{S_1, S_2, S_3\}$ such that over all possible executions, either validity or agreement does not hold for parties executing programs contained in S with probability at least $1/3$.

To transfer these observations in the modified network onto the real network, we will pick one of the following strategies at random.

- Let $\mathcal{A}$ be the adversary implied by Claim 8. Observe that $\mathcal{A}$ actively corrupts P_3, and compromises the secret key of P_1. In this case, we let $S' = S_1$.
- Let $\mathcal{A}$ be the adversary implied by Claim 6. Observe that $\mathcal{A}$ actively corrupts P_1, and compromises the secret key of no other honest party. In this case, we let $S' = S_2$.
- Let $\mathcal{A}$ be the adversary implied by Claim 7. Observe that $\mathcal{A}$ actively corrupts P_2, and compromises the secret key of P_1. In this case, we let $S' = S_3$.

In each of the above cases, observe that the joint view of the honest parties interacting with $\mathcal{A}$ in the real network is perfectly indistinguishable from their joint view in the modified network. Next, note that in each scenario, S' denotes exactly the programs executed by the honest parties in the modified network. Thus, when $S = S'$ holds we conclude that adversary $\mathcal{A}$ violates either the validity or agreement conditions for the honest parties in the real network with probability $1/3$. Since S' is chosen at random, we have that for every protocol Π, adversary $\mathcal{A}$ succeeds in violating either the validity or agreement for honest parties with probability at least $1/9$. ■

We remark that Lemma 5 holds only when P_1 acts as the dealer. Indeed, when P_2 or P_3 act as the dealer, it is possible to design a simple protocol for broadcast tolerating $\mathcal{A}$ described in the statement of Lemma 5. For instance, suppose P_2 acts as the dealer. Observe that $\mathcal{A}$ never compromises the secret key of an honest P_2. Consider the following protocol for broadcast. In the first round, P_2 sends a signature on his input message to P_1 and P_3. In the next round, P_1 and P_3 exchange the signed messages received from P_2. If P_2's signature appears on different messages, then P_1 and P_3 output some default value. Else, they output the single value that has P_2's signature on it. This completes the description of the protocol. To see why this protocol works, observe that when P_2 is honest and non-compromised, P_1 and P_3 will obtain signatures on the same message, and the adversary cannot create a valid signature on a different message. Therefore, honest parties will output the same message in this case. On the other hand, if P_2 is malicious, we only need agreement, and it is easy to see that agreement holds at the end of the second round. As was also pointed out in [58], the above gives a non-trivial example of a setting where authenticated broadcast is possible when one party acts as the dealer, and impossible when a different party acts as the dealer.

3.2.2 Impossibility of Broadcast for $2t_a + \min(t_a, t_c) \geq n$

In this section, we extend the three-party impossibility result to the multi-party setting when thresholds t_a, t_c are known. We do this using a standard *player-partitioning* argument.

Theorem 9 *Fix n, t_a, t_c with $n \geq 3, t_c > 0$ and $2t_a + \min(t_a, t_c) \geq n$. There is no protocol achieving broadcast in the presence of a (t_a, t_c)-adversary.*

Proof Suppose, towards a contradiction, that there exists a protocol Π for achieiving broadcast among n parties in $\mathcal{P} = \{P_1, \ldots, P_n\}$ in the presence of a (t_a, t_c)-adversary.

We partition the set of n parties $\mathcal{P} = \{P_1, \ldots, P_n\}$ into $\mathcal{P}_1$, $\mathcal{P}_2$, and $\mathcal{P}_3$ in the following way. $\mathcal{P}_2$ and $\mathcal{P}_3$ each contain at least one party, and at most t_a parties. $\mathcal{P}_1$ contains at most $\min(t_a, t_c)$ parties including the dealer. Indeed, such a partition exists since $t_c > 0$ and $2t_a + \min(t_a, t_c) \geq |\mathcal{P}| = n \geq 3$. Let Π_1, Π_2, Π_3 denote the program specified by protocol Π for parties in the sets $\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3$ respectively. In more detail, each program Π_i (for $i \in \{1, 2, 3\}$) represents the joint input/output behavior of the individual subprograms executed by parties in $\mathcal{P}_i$.

Consider an adversary $\mathcal{A}$ that can choose to corrupt in one of the following ways:

- $\mathcal{A}$ actively corrupts all parties in $\mathcal{P}_1$, and compromises the secret key of no other honest party.
- $\mathcal{A}$ actively corrupts all parties in $\mathcal{P}_2$, and compromises the secret keys of all parties in $\mathcal{P}_1$.
- $\mathcal{A}$ actively corrupts all parties in $\mathcal{P}_3$, and compromises the secret keys of all parties in $\mathcal{P}_1$.

Verify that $\mathcal{A}$ is indeed a (t_a, t_c)-adversary since $|\mathcal{P}_1| \leq \min(t_a, t_c) \leq t_a$, and $|\mathcal{P}_2|, |\mathcal{P}_3| \leq t_a$.

Now the proof proceeds in the same way as the proof of Lemma 5. More concretely, we analyze executions on Π in a modified network among four programs $\Pi_{1,0}, \Pi_{1,1}, \Pi_2$, and Π_3. As before, in the modified network two *independent* identical copies of program Π_1, namely $\Pi_{1,0}$ and $\Pi_{1,1}$ are run in place of Π_1. The modified network is exactly as shown in Figure 3.3 where a bidirectional link between two programs is interpreted as a complete network of bidirectional links between the individual subprograms contained in those two programs. Thus, all messages sent between individual subprograms run as part of $\Pi_{1,0}$ and individual subprograms run as part of Π_2 are discarded. Similarly all messages sent between individual subprograms run as part of $\Pi_{1,1}$ and individual subprograms run as part of Π_3 are discarded. All other messages are exchanged as in the real network.

It is straightforward to verify that each of Claims 6, 7, and 8 still hold when P_1, P_2, P_3 are replaced by $\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3$ in their respective statements. Thus, we can conclude that the joint view of honest parties executing protocol Π in the real network is indistinguishable from the joint view of honest parties executing protocol Π in the modified network. Furthermore, as demonstrated in the proof of Lemma 5, assigning the dealer in $\Pi_{1,0}$ with input 0, and the dealer in $\Pi_{1,1}$ with input 1, we are guaranteed that there exist at least two honest parties in $\mathcal{P}$ for whom either validity or agreement is violated with probability at least 1/3, and that $\mathcal{A}$ succeeds in violating the same with probability at least 1/9. This completes the proof of the theorem. ■

3.2.3 Impossibility of Broadcast with a Threshold Adversary

We now turn to the case of the threshold adversary. Recall that in this setting the exact values of t_a and t_c used by the adversary are not known; we only know that they satisfy $2t_a + \min(t_a, t_c) < n$ (and we do allow $t_c = 0$). In what follows, we show that secure broadcast is impossible if $n \notin \{2, 3, 4, 5, 6, 8, 9, 12\}$. For the "exceptional" values of n, we demonstrate feasibility in Section 3.3.

Theorem 10 *If $n \leq 2\left\lfloor \frac{n-1}{3} \right\rfloor + \left\lfloor \frac{n-1}{2} \right\rfloor$, then there does not exist a secure broadcast protocol for n parties in the presence of a threshold adversary. (Note that $n \leq 2\left\lfloor \frac{n-1}{3} \right\rfloor + \left\lfloor \frac{n-1}{2} \right\rfloor$ for all $n > 1$ except $n \in \{2, 3, 4, 5, 6, 8, 9, 12\}$.)*

Proof Suppose, towards a contradiction, that there exists a protocol Π for achieiving broadcast among n parties in $\mathcal{P} = \{P_1, \ldots, P_n\}$ in the presence of a threshold adversary.

We partition the set of n parties $\mathcal{P} = \{P_1, \ldots, P_n\}$ into $\mathcal{P}_1$, $\mathcal{P}_2$, and $\mathcal{P}_3$ in the following way. $\mathcal{P}_2$ and $\mathcal{P}_3$ each contain at least one party, and at most $\left\lfloor \frac{n-1}{3} \right\rfloor$ parties. $\mathcal{P}_1$ contains at most $\left\lfloor \frac{n-1}{2} \right\rfloor$ parties including the dealer. Indeed, such a partition exists since we are given that $n > 4$ and $2\left\lfloor \frac{n-1}{3} \right\rfloor + \left\lfloor \frac{n-1}{2} \right\rfloor \geq n$ hold. Let Π_1, Π_2, Π_3 denote the program specified by protocol Π for parties in the sets $\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3$ respectively. In more detail, each program Π_i (for $i \in \{1, 2, 3\}$) represents the joint input/output behavior of the individual subprograms executed by parties in $\mathcal{P}_i$.

Consider an adversary $\mathcal{A}$ that can choose to corrupt in one of the following ways:

- $\mathcal{A}$ actively corrupts all parties in $\mathcal{P}_1$, and compromises the secret key of no other honest party.
- $\mathcal{A}$ actively corrupts all parties in $\mathcal{P}_2$, and compromises the secret keys of all parties in $\mathcal{P}_1$.
- $\mathcal{A}$ actively corrupts all parties in $\mathcal{P}_3$, and compromises the secret keys of all parties in $\mathcal{P}_1$.

Verify that $\mathcal{A}$ is indeed a threshold adversary since $2|\mathcal{P}_1| \leq 2\left\lfloor \frac{n-1}{2} \right\rfloor < n$ and both $2|\mathcal{P}_2| + \min(|\mathcal{P}_2|, |\mathcal{P}_1|)$ and $2|\mathcal{P}_3| + \min(|\mathcal{P}_3|, |\mathcal{P}_1|)$ are at most $2\left\lfloor \frac{n-1}{3} \right\rfloor + \min(\left\lfloor \frac{n-1}{3} \right\rfloor, \left\lfloor \frac{n-1}{2} \right\rfloor) = 3\left\lfloor \frac{n-1}{3} \right\rfloor < n$.

Now the proof proceeds in the same way as the proof of Lemma 5. More concretely, we analyze executions on Π in a modified network among four programs $\Pi_{1,0}, \Pi_{1,1}, \Pi_2$, and Π_3. As before, in the modified network two *independent* identical copies of program Π_1, namely $\Pi_{1,0}$ and $\Pi_{1,1}$ are run in place of Π_1. The modified network is exactly as shown in Figure 3.3 where a bidirectional link between two programs is interpreted as a complete network of bidirectional links between the individual subprograms contained in those two programs. Thus, all messages sent between individual subprograms run as part of $\Pi_{1,0}$ and individual subprograms run as part of Π_2 are discarded. Similarly, all messages sent between individual subprograms run as part of $\Pi_{1,1}$ and individual subprograms run as part of Π_3 are discarded. All other messages are exchanged correctly.

It is straightforward to verify that each of Claims 6, 7, and 8 still hold when P_1, P_2, P_3 are replaced by $\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3$ in their respective statements. Thus, we can conclude that the joint view of honest parties executing protocol Π in the real network is indistinguishable from the joint view of honest parties executing protocol Π in the modified network. Furthermore, as demonstrated in

Protocol $\mathsf{ALSP}(\mathcal{P}, D, \tilde{m}, \psi)$

Inputs: The protocol is parameterized by an integer ψ. Let D be the dealer with input $\tilde{m}$ of the form $\tilde{m} = (\text{“}m\text{”}, \mathsf{Sign}_{sk_D}(m))$.

- **Stage 1:** D sends $\tilde{m} = (\text{“}m\text{”}, \mathsf{Sign}_{sk_D}(m))$ to all parties, and outputs m. Let $\tilde{m}'_i$ denote the value received by P_i. If $\tilde{m}'_i$ is invalid (in particular, does not contain a valid signature of D) then it sets $\tilde{m}'_i = 0$. If $\psi = 0$, then each party P_i outputs $\tilde{m}_i = \tilde{m}'_i$, and terminates.

- **Stage 2:** Each $P_i \in \mathcal{P} \setminus \{D\}$ executes $\mathsf{ALSP}(\mathcal{P} \setminus \{D\}, P_i, (\text{“}\tilde{m}'_i\text{”}, \mathsf{Sig}_{sk_i}(\tilde{m}'_i)), \psi - 1)$. For each $P_j \in \mathcal{P} \setminus \{D\}$, let $\tilde{m}_j^{(i)} = (\text{“}m_j^{(i)}\text{”}, \star)$ denote its output of this execution.

- **Stage 3:** Each $P_j \in \mathcal{P} \setminus \{D\}$ computes $\mathsf{PART}^j_{m'} = \{P_i \mid m_j^{(i)}$ contains a valid signature from D on $m'\}$. P_j obtains m_j such that $|\mathsf{PART}^j_{m'}|$ is maximized for $m' = m_j$. Then, it obtains $\tilde{m}_j$ as lexicographically first message in $\{m_j^{(i)} \mid P_i \in \mathsf{PART}^j_{m_j}\}$. (If $\mathsf{PART}^j_{m'} = \emptyset$ for all m', then P_j sets $\tilde{m}_j = 0$.) Finally, P_j outputs $\tilde{m}_j$.

Figure 3.7: Protocol ALSP.

the proof of Lemma 5, assigning the dealer in $\Pi_{1,0}$ with input 0, and the dealer in $\Pi_{1,1}$ with input 1, we are guaranteed that there exist at least two honest parties in $\mathcal{P}$ for whom either validity or agreement is violated with probability at least 1/3, and that $\mathcal{A}$ succeeds in violating the same with probability at least 1/9. This completes the proof of the theorem. ■

3.3 Handling the Exceptional Values of n

We refer to $\{2, 3, 4, 5, 6, 8, 9, 12\}$ as the set of *exceptional values* for n. (These are the only positive, integer values of n for which Theorem 10 does not apply.) We show for any exceptional value of n a broadcast protocol that is secure against any threshold adversary. Designing protocols in this setting is more difficult than in the setting of Section 3.1, since the honest parties are no longer assumed to “know” whether $t_a \leq t_c$.

Our main construction, which we refer to as ALSP, is an authenticated version of the exponential broadcast protocol of Lamport et al. [71] with parameter ψ; see Figure 3.7. Our protocol for broadcast is presented in Figure 3.8. Although the message complexity of this protocol is exponential in the number of parties, the maximum number of parties considered here is 12.

Suppose the dealer's input message is m. In Protocol 2, the dealer signs this message using his secret key sk_D, and then executes protocol ALSP using input $\tilde{m} = (\text{“}m\text{”}, \mathsf{Sig}_{sk_D}(m))$. In general, we use the notation $(\text{“}m\text{”}, \star)$ to denote such messages, with $\star$ acting as a placeholder for an arbitrary (possibly invalid) signature.

We let $t_h = n - t_c - t_a$ denote the number of honest and non-compromised parties. One useful observation about threshold adversaries that we will repeatedly use is that when $t_a > \lfloor \frac{n-1}{3} \rfloor$, it follows that $t_h > t_a$.

The next two lemmas are inspired by [71].

Lemma 11 *If D is honest and $\psi < n - 2t_a$ holds, then $\mathsf{ALSP}(\mathcal{P}, D, \tilde{m}, \psi)$ achieves validity and agreement in the following sense. Each honest P_j outputs a message $\tilde{m}_j$ which contains a valid signature from D, and further, if $\tilde{m} = (\text{“}m\text{”}, \star)$, and $\tilde{m}_j = (\text{“}m_j\text{”}, \star)$, then $m = m_j$ holds for every honest $P_j \in \mathcal{P} \setminus \{D\}$.*

Proof The proof is by induction on ψ. First observe that when D is honest, it ensures (perhaps, by setting an invalid message to 0) that $\tilde{m}$ is always valid and contains its signature. Given this, clearly the lemma is true for $\psi = 0$. We now assume that the lemma is true for $\psi - 1$, and prove it for ψ.

Each honest $P_i \in \mathcal{P} \setminus \{D\}$ executes ALSP with input ("$\tilde{m}'_i$", $\mathsf{Sig}_{sk_i}(\tilde{m}'_i)$) with $\tilde{m}'_i = \tilde{m}$. Since by hypothesis $\psi < n - 2t_a$, we have $\psi - 1 < n - 1 - 2t_a$, so we can apply the induction hypothesis to conclude that every honest P_j obtains $\tilde{m}_j^{(i)} = \tilde{m}'_i$ for honest P_i, and thus each honest $P_i \in \mathsf{PART}^j_m$ since $\tilde{m}'_i = \tilde{m} = (\text{"}m\text{"}, \star)$ contains a valid signature from D. Since there are at most t_a corrupt parties, and $n - 1 > 2t_a + \psi - 1 \geq 2t_a$, a majority of parties in $\mathcal{P} \setminus \{D\}$ are honest. Thus, $\mathsf{PART}^j_{m'}$ is maximized only for $m' = m$, so each honest party P_j outputs $m_j = m$. ■

Lemma 12 *If $n > 3\psi$ and $\psi \geq t_a$, then $\mathsf{ALSP}(\mathcal{P}, D, \tilde{m}, \psi)$ achieves validity and agreement in the following sense. Suppose $\tilde{m} = (\text{"}m\text{"}, \star)$, and for each party P_j, let $\tilde{m}_j = (\text{"}m_j\text{"}, \star)$ denote its output. Then $m_j = m_k$ holds for every honest $P_j, P_k \in \mathcal{P}$. Furthermore, if D is honest, then $m_j = m$ holds for every honest party $P_j \in \mathcal{P}$.*

Proof The proof is by induction on ψ. If there is no corrupted party, then no party sends an invalid signature. Given this, clearly the lemma is true when $\psi = 0$. We now assume that the lemma is true for $\psi - 1$, and prove it for ψ.

We first consider the case when D is honest. Since $n > 3\psi$ and $\psi \geq t_a$, we have $n > 2t_a + \psi$, i.e., $\psi < n - 2t_a$. Thus, we can apply Lemma 11 (which proves something stronger for an honest D) and conclude that this lemma holds. Now we only need to prove the lemma when D is corrupt.

Since there are at most t_a corrupt parties, and D is one of them, so at most $t_a - 1$ of the remaining $n-1$ parties are corrupt. Furthermore, observe that $n - 1 > 3(\psi - 1)$, and $\psi - 1 \geq t_a - 1$ still hold. We therefore apply the induction hypothesis to conclude that the lemma holds for each ALSP execution with $\psi - 1$. That is, for every honest $P_j, P_k \in \mathcal{P} \setminus \{D\}$ participating in $\mathsf{ALSP}(\mathcal{P} \setminus \{D\}, P_i, \tilde{m}'_i, \psi - 1)$, we have $m_j^{(i)} = m_k^{(i)}$. Given this, it follows that $\mathsf{PART}^j_{m'} = \mathsf{PART}^k_{m'}$ holds for every m'. Thus, the lemma holds for ALSP executions with parameter ψ. ■

We now prove several additional lemmas about ALSP.

Lemma 13 *If D is honest and non-compromised, then $\mathsf{ALSP}(\mathcal{P}, D, \tilde{m}, \psi)$ achieves validity and agreement for any ψ.*

Proof Clearly, each honest P_j receives a valid message $\tilde{m}$ in Stage 1. Thus, each honest P_j obtains output $m_j^{(j)} = \tilde{m}$ from its own ALSP execution in Stage 2. Therefore, $\mathsf{PART}^j_{\tilde{m}}$ is non-empty. Furthermore, since D is honest and non-compromised, $\mathsf{PART}^j_{m'}$ is empty for every $m' \neq \tilde{m}$. $\mathsf{PART}^j_{m'}$ is maximized only for $m' = \tilde{m}$, so agreement and validity follow immediately. ■

Lemma 14 *Let t_h be the number of honest and non-compromised parties in $\mathcal{P}$, and let t_a be the number of actively corrupted parties in $\mathcal{P}$. If D is honest, and $t_h > t_a$, then protocol $\mathsf{ALSP}(\mathcal{P}, D, \tilde{m}, \psi)$ achieves validity and agreement for any ψ.*

Proof The proof is by induction on ψ. First observe that when D is honest, it ensures (perhaps, by setting an invalid message to 0) that $\tilde{m}$ is always valid and contains its signature. Given this, clearly the lemma is true for $\psi = 0$. We now assume that the lemma is true for $\psi - 1$, and prove it for ψ.

Protocol 2

The protocol is carried out among a set $\mathcal{P} = \{P_1, \ldots, P_n\}$ of parties. For notational convenience, we let D denote the dealer (though in fact any party can act as dealer). We let $m \in \{0,1\}$ denote D's input, and sk_D its signing key.

- If $n \in \{5,6,8\}$, execute $\mathsf{ALSP}(\mathcal{P}, D, (\text{“}m\text{”}, \mathsf{Sig}_{sk_D}(m)), \left\lfloor \frac{n-1}{3} \right\rfloor + 1)$.
- If $n \in \{9,12\}$, execute $\mathsf{ALSP}(\mathcal{P}, D, (\text{“}m\text{”}, \mathsf{Sig}_{sk_D}(m)), \left\lfloor \frac{n-1}{3} \right\rfloor + 2)$.
- Let $\tilde{m}'_i = (m'_i, \star)$ be P_i's output in the above. P_i outputs $m_i = m'_i$ and terminates.

Figure 3.8: Broadcast against a threshold adversary.

When D is non-compromised, the lemma follows from Lemma 13. Now we only need to prove the lemma when D is compromised.

Observe that when D is compromised, the number of honest and non-compromised parties in $\mathcal{P} \setminus \{D\}$ is still t_h, and the number of actively corrupted parties in $\mathcal{P} \setminus \{D\}$ is still t_a. Since by hypothesis $t_h > t_a$, we can apply the induction hypothesis to conclude that for every honest $P_i \in \mathcal{P} \setminus \{D\}$, protocol $\mathsf{ALSP}(\mathcal{P} \setminus \{D\}, P_i, \tilde{m}'_i, \psi - 1)$ achieves validity and agreement. Since for an honest P_i we have $\tilde{m}'_i = \tilde{m}$, so for every P_j, $\mathsf{PART}^j_{\tilde{m}}$ contains all honest parties. Furthermore, the number of honest parties in $\mathcal{P} \setminus \{D\}$ is at least t_h which is in turn greater than t_a, so $\mathsf{PART}^j_{m'}$ is maximized only for $m' = \tilde{m}$. Given this, validity and agreement follow immediately for ALSP executions with parameter ψ. ■

Observe that when $\psi < n - 2\left\lfloor \frac{n-1}{3} \right\rfloor$ and $t_a \leq \left\lfloor \frac{n-1}{3} \right\rfloor$, then $\psi < n - 2t_a$ holds. Therefore, using Lemma 12 (for the case when $t_a \leq \left\lfloor \frac{n-1}{3} \right\rfloor$) and Lemma 14 (for the case when $t_a > \left\lfloor \frac{n-1}{3} \right\rfloor$), we obtain the following corollary.

Corollary 15 *If D is honest and $\psi < n - 2\left\lfloor \frac{n-1}{3} \right\rfloor$, then protocol $\mathsf{ALSP}(\mathcal{P}, D, \tilde{m}, \psi)$ achieves validity and agreement.*

The following trivial facts and the corollary above imply our next lemma.

Fact 16 *For $n \in \{5,6,8\}$, it holds that $\left\lfloor \frac{n-1}{2} \right\rfloor - 1 \leq \left\lfloor \frac{n-1}{3} \right\rfloor < \frac{n-1}{3}$.*

Fact 17 *For $n \in \{9,12\}$, it holds that $\left\lfloor \frac{n-1}{2} \right\rfloor - 2 \leq \left\lfloor \frac{n-1}{3} \right\rfloor < \frac{n-2}{3}$.*

Lemma 18 *If D is honest, then Protocol 2 achieves broadcast for $n \in \{5,6,8,9,12\}$.*

Lemma 19 *Protocol 2 achieves broadcast for $n \in \{5,6,8\}$.*

Proof By Lemma 18, we see that the lemma is true when D is honest. We now turn to a malicious dealer. Consider the $n-1$ ALSP executions in Stage 2. Since the dealer is malicious, at most $t_a - 1$ out of the $n-1$ parties in $\mathcal{P} \setminus \{D\}$ are corrupt. Recall that t_a is at most $\left\lfloor \frac{n-1}{2} \right\rfloor$ for a threshold adversary, so by Fact 16 we have that for $n \in \{5,6,8\}$, $\psi - 1 = \left\lfloor \frac{n-1}{3} \right\rfloor \geq t_a - 1$, and $n - 1 > 3(\psi - 1)$. Thus, we can apply Lemma 12 to conclude that agreement and validity is achieved in all ALSP executions in Stage 2. Given this, it is easy to see that there is agreement among the outputs of honest parties in Stage 3. Thus, the lemma holds. ■

Lemma 20 *Protocol 2 achieves broadcast for $n \in \{9,12\}$.*

Proof By Lemma 18, we see that the lemma is true when D is honest. We now turn to the case when D is malicious. Consider the $n-1$ ALSP executions in Stage 2. Out of these $n-1$ executions, we focus first on executions with a honest dealer. Using Fact 17, verify that $\psi - 1 = \left\lfloor \frac{n-1}{3} \right\rfloor + 1 < (n-1) - 2\left\lfloor \frac{n-2}{3} \right\rfloor$ holds. Applying Corollary 15, we conclude that agreement and validity are achieved in all Stage 2 ALSP executions with a honest dealer.

We now turn to the Stage 2 ALSP executions with a malicious dealer D'. Since the dealer is malicious, at most $t_a - 2$ out of the $n-2$ parties in $\mathcal{P} \setminus \{D, D'\}$ are corrupt. Using Fact 17, verify that $\psi - 2 = \left\lfloor \frac{n-1}{3} \right\rfloor + 1 \geq t_a - 2$ and $n - 2 > 3(\psi - 2)$. Applying Lemma 12, we conclude that agreement and validity are achieved in all ALSP executions among parties in $\mathcal{P} \setminus \{D, D'\}$. Given this, it is easy to see that there is agreement among the outputs of honest parties in $\mathsf{ALSP}(\mathcal{P} \setminus \{D\}, D', \star, \psi - 1)$.

Thus, we conclude that agreement is achieved in all Stage 2 ALSP executions in $\mathsf{ALSP}(\mathcal{P}, D, \star, \psi)$ irrespective of whether the dealer is honest or malicious. Given this, agreement among the outputs of honest parties in Stage 3 of $\mathsf{ALSP}(\mathcal{P}, D, \star, \psi)$ follows immediately, and the lemma holds. ■

Theorem 21 *For any value $n \in \{2, 3, 4, 5, 6, 8, 9, 12\}$ there exists a protocol for n parties that achieves broadcast in the presence of a threshold adversary.*

Proof The case $n = 2$ is trivial. When $n = 3$, it follows from our constraints that $t_a \leq 1$ and $t_c = 0$, so we can run any authenticated byzantine agreement protocol. When $n = 4$, it follows from our constraints that $t_a \leq 1$, and therefore that $n > 3t_a$, so we can ignore the PKI and run a protocol that is secure without authentication. By Lemma 19 and Lemma 20, we have that Protocol 2 achieves broadcast when $n \in \{5, 6, 8, 9, 12\}$. This concludes the proof of the theorem. ■

3.4 A More Efficient Protocol When $t_h > 0$

As in the previous section, we refer to $\{2, 3, 4, 5, 6, 8, 9, 12\}$ as the set of *exceptional values* for n. We show for any exceptional value of n a broadcast protocol that is secure against any threshold adversary. A full proof of the above was given in Section 3.3, but it was based on the exponential algorithm of Lamport et al. [71] rather than the more efficient protocol of Dolev-Strong [29]. In this section, we deal with the "easier" case where there is guaranteed to be at least one honest, non-compromised party[1] (i.e., $t_a + t_c < n$). This assumption allows us to provide a protocol that is based on the more efficient construction of Dolev-Strong (cf. Section 2.5), similar to the protocol we presented in Section 3.1.

As in Section 3.1, our protocol begins with the dealer sending its input (here referred to as m) to each party; each party then runs $\mathsf{DS}(m)$. However, because we no longer know whether $t_a \leq t_c$, the following problem arises: when the dealer is honest but compromised and $t_a \leq t_c$, we cannot be sure that the value output in the majority of clean runs is m. It is possible that $t_h < t_a$, and (recalling that the adversary can force all executions of DS by compromised parties to be dirty) it is feasible for the adversary to force the majority of clean runs to have output $1 - m$.

To address this, we design our protocol to carefully look at the number of clean runs and use this information in a particular way when determining the output. Specifically, notice that if there are many dirty runs d (specifically, $d > 2n/3$) then the honest parties can conclude that $t_a < n/3$: this follows because $2n/3 < d \leq t_a + t_c$, so if $t_a \geq n/3$ then we would have $n \leq 2t_a + \min(t_a, t_c)$,

[1] The difficulty that arises when $t_a + t_c = n$ is that the compromised parties may not agree on whether an execution of DS is clean or dirty (since Lemma 3 no longer holds). This is not a problem in Section 3.1 because, there, whenever $t_a + t_c < n$, the parties run an unauthenticated broadcast protocol that does not use a PKI.

Protocol 3

The protocol is carried out among a set $\mathcal{P} = \{P_1, \ldots, P_n\}$ of parties. For notational convenience, we let D denote the dealer (though in fact any party can act as dealer). We let $m \in \{0,1\}$ denote D's input. In the following, we use DS to denote an execution of DS_ϕ with $\phi = t_a + t_c$.

- **Stage 1:** D acts as the dealer in an execution of $\mathsf{DS}(m)$. Denote party P_i's output by m_i^{DS}. If the dealer's run was clean, then each party P_i outputs $m_i = m_i^{\mathsf{DS}}$ and terminates. Else, D sends m to all other parties. Let m_i' be the value received by P_i from D.

- **Stage 2:** In parallel, each $P_i \in \mathcal{P} \setminus \{D\}$ acts as the dealer in an execution of $\mathsf{DS}(m_i')$. For $b \in \{0,1\}$, let CLEAN_b denote the set of parties whose execution of DS is clean, and results in an output of b. Initialize $\mathsf{flag} = 0$, and for $b \in \{0,1\}$, set $\mathsf{CLEAN}_b' = \mathsf{CLEAN}_b$.

- **Stage 3:** Let $\mathsf{CLEAN} = \mathsf{CLEAN}_0 \cup \mathsf{CLEAN}_1$, and $\mathsf{DIRTY} = \mathcal{P} \setminus \mathsf{CLEAN}$. Parties execute the following sequentially:

  (a) If $|\mathsf{CLEAN}_0'| > \lfloor \frac{n-1}{3} \rfloor$ or $|\mathsf{CLEAN}_1'| > \lfloor \frac{n-1}{3} \rfloor$, then output 0 if $|\mathsf{CLEAN}_0'| \geq |\mathsf{CLEAN}_1'|$, and output 1 otherwise.

  (b) If $\mathsf{flag} = 1$ or $|\mathsf{CLEAN}| < \lfloor \frac{n-1}{3} \rfloor + 2$, then each party P_i participates in an execution of BGP protocol with input m_i'. Let party P_i's output be m_i^{BGP}. Each party P_i outputs $m_i = m_i^{\mathsf{BGP}}$ and terminates.

  Each party $P_j \in \mathsf{DIRTY}$ sends m_j' to parties in CLEAN. Each party $P_i \in \mathsf{CLEAN}_b$ sets value m_i'' as follows:

  $$m_i'' = \begin{cases} b & \text{if at most } \lfloor \frac{n-1}{3} \rfloor - |\mathsf{CLEAN}_{1-b}| \text{ parties in } \mathsf{DIRTY} \text{ sent } 1-b \text{ to } P_i \\ \mathsf{null} & \text{otherwise} \end{cases}.$$

- **Stage 4:** In parallel, each party $P_i \in \mathsf{CLEAN}$ runs $\mathsf{DS}(m_i'')$. If for some $P_i \in \mathsf{CLEAN}_0$ (resp. CLEAN_1), the run $\mathsf{DS}(m_i'')$ is dirty, we add P_i to DIRTY, remove P_i from CLEAN_0 (resp. CLEAN_1), and go to Stage 3. Else, for $b \in \{0,1\}$, add to CLEAN_b' all parties in CLEAN_{1-b} that gave a clean run on null, set $\mathsf{flag} = 1$, and goto Stage 3.

Figure 3.9: Broadcast against a threshold adversary, assuming $t_a + t_c < n$.

exceeding the assumed threshold. Thus, when many dirty runs are detected the parties can switch to running any unauthenticated broadcast protocol that does not use the PKI at all. (We use the unauthenticated broadcast protocol based on the consensus protocol of Berman et al. [6], denoted by BGP as a subprotocol in this case.) On the other hand, when there are few dirty runs (roughly less than $n/3$), intuitively the parties can safely trust the majority output of the clean runs.

The above leaves a "gap" in which there are too few dirty runs to conclude that $n > 3t_a$, and too many to trust the majority output of the clean runs. This is only problematic if the number of clean runs resulting in output m is close to the number of clean runs resulting in output $1-m$, and this balance will allow us to extract one last piece of information. Such a balance can occur in only two ways. The first is when a Byzantine dealer gives m to some non-faulty parties and $1-m$ to others, splitting the clean runs almost evenly between them. In this case we only need to worry about agreement since the dealer is Byzantine. The more difficult case involves a compromised dealer, since then correctness is also required. Here the key is to note that all clean runs with honest dealers result in output m; thus, to achieve the assumed balance, almost all the Byzantine parties must run cleanly with output $1-m$. If most of the Byzantine parties give clean executions, it follows that the dealers in dirty executions are honest (but compromised), and we can rely on them to correct the majority value back to m. The protocol is described in Figure 3.9.

Theorem 22 *For $n \in \{2,3,4,5,6,8,9,12\}$, Protocol 3 achieves broadcast in the presence of a threshold adversary $\mathcal{A}$ under the additional assumption that $t_a + t_c < n$.*

Proof One can verify that, for n as in the theorem, $n > \lfloor \frac{n-1}{2} \rfloor + 2\lfloor \frac{n-1}{3} \rfloor$. We use this property throughout the proof.

The value $\left\lfloor \frac{n-1}{3} \right\rfloor$ serves as a sort of breakpoint for the parameter t_a: if $t_a \leq \left\lfloor \frac{n-1}{3} \right\rfloor$, then (because $\mathcal{A}$ is a threshold adversary) we have $n > 3\left\lfloor \frac{n-1}{3} \right\rfloor \geq 3t_a$; when $t_a > \left\lfloor \frac{n-1}{3} \right\rfloor$ then it holds that $t_c < t_a$ and, consequently $t_a < t_h$ holds.[2] These are exactly the two cases handled (independently) in Protocol 1. The difficulty here is to identify which of these scenarios is the "right" one. We prove the correctness of our protocol in the following three lemmas.

Lemma 23 *If D is honest and non-compromised then validity and agreement hold.*

Proof When the dealer is honest and non-compromised, then in Stage 1, the dealer's execution of DS is clean and results in output m (cf. Lemma 3). Thus, all honest parties terminate with output m in Stage 1. ■

Lemma 24 *If D is honest but compromised then validity and agreement hold.*

Proof Note that when the protocol terminates at Stage 1, we are guaranteed agreement and validity. For the rest of the proof, we assume that the protocol continues past Stage 1. We consider separately the cases $t_a > \left\lfloor \frac{n-1}{3} \right\rfloor$ and $t_a \leq \left\lfloor \frac{n-1}{3} \right\rfloor$.

Case 1: $t_a > \left\lfloor \frac{n-1}{3} \right\rfloor$. Let t_h denote the number of honest and non-compromised parties. Recall that for a threshold adversary, when $t_a > \left\lfloor \frac{n-1}{3} \right\rfloor$, it follows that $t_c < t_a$ and $t_h > t_a$, i.e., $t_h \geq \left\lfloor \frac{n-1}{3} \right\rfloor + 2$. Clearly, all honest and non-compromised parties would run DS on input m in Stage 2. By Lemma 3, at least t_h runs in Stage 2 are clean and result in output m. Therefore, in Stage 2, either CLEAN_0 or CLEAN_1 is of size at least t_h and hence greater than $\left\lfloor \frac{n-1}{3} \right\rfloor$. The protocol thus terminates in Stage 2 and we have agreement. Observe that all honest parties are contained in either CLEAN_m or in DIRTY (and not in CLEAN_{1-m}). Validity follows from the fact that $t_a < t_h$ and $|\mathsf{CLEAN}_m| > |\mathsf{CLEAN}_{1-m}|$.

Case 2: $t_a \leq \left\lfloor \frac{n-1}{3} \right\rfloor$. When $t_a \leq \left\lfloor \frac{n-1}{3} \right\rfloor$, the honest parties are in two-thirds majority, i.e., $n > 3t_a$. Therefore, whenever the parties terminate with an output obtained by running BGP, it is always guaranteed to be correct, i.e., for every honest P_i, we have $m_i^{\mathsf{BGP}} = m$. The only other termination condition is when $|\mathsf{CLEAN}'_b| > \left\lfloor \frac{n-1}{3} \right\rfloor$ for some $b \in \{0,1\}$. Since for every honest party P_i we have $m'_i = m$, the set CLEAN'_{1-m} will never contain any honest party. Thus $|\mathsf{CLEAN}'_b| > \left\lfloor \frac{n-1}{3} \right\rfloor$ can hold only for $b = m$. Therefore all possible terminations of the protocol result in correct output. ■

Lemma 25 *When D is corrupt, agreement holds.*

Proof Observe that the protocol can terminate in two ways: either (a) by a termination condition that depends on CLEAN'_0 and CLEAN'_1 in Stage 3, or (b) by running an instance of BGP in Stage 3. Since all honest parties agree on the values of CLEAN'_0 and CLEAN'_1, agreement is always guaranteed

[2] Note that $n = t_a + t_c + t_h$. By the threshold condition, $n > 2t_a + min(t_a, t_c)$. Since $t_a > t_c$, we have $n = t_a + t_c + t_h > 2t_a + t_c$, i.e., $t_h > t_a$.

when the protocol terminates because of that condition. Hence, we restrict our attention to the case when parties terminate by running an instance of BGP.

Observe that when $t_a \leq \left\lfloor \frac{n-1}{3} \right\rfloor$, we have $n > 3\left\lfloor \frac{n-1}{3} \right\rfloor > 3t_a$, and agreement is guaranteed whenever the protocol terminates after running BGP. Therefore, we are left with analyzing the case when $t_a > \left\lfloor \frac{n-1}{3} \right\rfloor$. In fact, we will prove that in this case, parties never terminate after running BGP.

Let t_h represent the number of honest and non-compromised parties, recall that in this case $t_h > t_a$, implying $t_h \geq \left\lfloor \frac{n-1}{3} \right\rfloor + 2$. All honest and non-compromised parties give clean runs resulting in $|\mathsf{CLEAN}| \geq t_h \geq \left\lfloor \frac{n-1}{3} \right\rfloor + 2$.

We now argue that there exists $b \in \{0,1\}$ such that $|\mathsf{CLEAN}'_b| > \left\lfloor \frac{n-1}{3} \right\rfloor$. (Indeed if this is the case, then we have proved that all parties agree on the final output.) By way of contradiction assume that $|\mathsf{CLEAN}'_0|, |\mathsf{CLEAN}'_1| < \left\lfloor \frac{n-1}{3} \right\rfloor$. Consequently, $|\mathsf{CLEAN}_0|, |\mathsf{CLEAN}_1| < \left\lfloor \frac{n-1}{3} \right\rfloor$, and thus each of $\mathsf{CLEAN}_0, \mathsf{CLEAN}_1$ contains at least one honest and non-compromised party.

Next, since $|\mathsf{CLEAN}| \geq \left\lfloor \frac{n-1}{3} \right\rfloor + 2$, we have $|\mathsf{CLEAN}'_0 \setminus \mathsf{CLEAN}_0| + |\mathsf{CLEAN}'_1 \setminus \mathsf{CLEAN}_1| \leq \left\lfloor \frac{n-1}{3} \right\rfloor - 2$. Since the maximum value of n considered here is 12, we have $|\mathsf{CLEAN}'_0 \setminus \mathsf{CLEAN}_0| + |\mathsf{CLEAN}'_1 \setminus \mathsf{CLEAN}_1| \leq 1$. Thus for some $b \in \{0,1\}$, we have $|\mathsf{CLEAN}'_{1-b} \setminus \mathsf{CLEAN}_{1-b}| = 0$. Therefore, every honest (and non-compromised) party P_i in CLEAN_b retained his value, i.e., set $m''_i = m'_i = b$. The following claim implies that every honest (and non-compromised) party P_j in CLEAN_b set $m''_j = \mathsf{null}$, and gave a clean run on m''_j.

Claim 26 *If for some $b \in \{0,1\}$, there exists an honest party $P_i \in \mathsf{CLEAN}_b$ that sets $m''_i = b$, then all honest parties $P_j \in \mathsf{CLEAN}_{1-b}$ set $m''_j = \mathsf{null}$.*

Proof Consider an honest party $P_i \in \mathsf{CLEAN}_b$, that sets $m''_i = b$. Let DIRTY contain t'_a corrupt parties and t'_c honest but compromised parties. Let $t'_{c,b}$ be the number of honest but compromised parties who received bit b from D in the first round. Now $t'_{c,1-b} \leq \left\lfloor \frac{n-1}{3} \right\rfloor - |\mathsf{CLEAN}_{1-b}|$. Otherwise, it is easy to see that P_i would have set $m''_i = \mathsf{null}$. We claim that $t'_{c,b} > \left\lfloor \frac{n-1}{3} \right\rfloor - |\mathsf{CLEAN}_b|$. Suppose that is not true, then we have $t'_c = t'_{c,0} + t'_{c,1} \leq 2\left\lfloor \frac{n-1}{3} \right\rfloor - |\mathsf{CLEAN}|$, i.e. $t'_c + |\mathsf{CLEAN}| \leq 2\left\lfloor \frac{n-1}{3} \right\rfloor$. Since $n = t'_a + t'_c + |\mathsf{CLEAN}|$, we have $n \leq t'_a + 2\left\lfloor \frac{n-1}{3} \right\rfloor \leq t_a + 2\left\lfloor \frac{n-1}{3} \right\rfloor$. This is a contradiction since, we are given that $n > 2\left\lfloor \frac{n-1}{3} \right\rfloor + \left\lfloor \frac{n-1}{2} \right\rfloor \geq 2\left\lfloor \frac{n-1}{3} \right\rfloor + t_a$. Hence we have proved that $t'_{c,b} > \left\lfloor \frac{n-1}{3} \right\rfloor - |\mathsf{CLEAN}_b|$ and thus all the honest parties $P_j \in \mathsf{CLEAN}_{1-b}$ will set $m''_j = \mathsf{null}$. ■

Therefore, CLEAN'_b contains all honest and non-compromised parties in CLEAN_b and in CLEAN_{1-b}, and consequently $|\mathsf{CLEAN}'_b| \geq t_h \geq \left\lfloor \frac{n-1}{3} \right\rfloor + 2$ which is a contradiction. ■

This concludes the proof of correctness of Protocol 3 and hence the theorem. ■

Protocol 3 runs at most $O(n^2)$ instances of the DS subprotocol and one instance of the BGP subprotocol. A single DS instance has communication complexity $O(n^3)$ signatures, and a BGP instance has communication complexity $O(n^2)$ bits. Therefore the total communication complexity of Protocol 3 is $O(n^5)$ signatures. In contrast, the protocol ALSP has communication complexity at least $O(n!)$ signatures. We note that even for small values of n as considered here, there is a substantial difference in the efficiency of the two protocols.

Chapter 4

Adaptively Secure Broadcast

Motivated by a recent result by Hirt and Zikas [57] we study adaptive attacks on existing broadcast protocols in this chapter. They studied the problem of designing broadcast protocols with security against *adaptive* adversaries who can choose which parties to corrupt during the course of the protocol (cf. [14]). Hirt and Zikas showed explicit attacks against all existing broadcast protocols when $t \geq n/3$ and, moreover, proved the *impossibility* of realizing adaptively secure broadcast with corruption threshold $t > n/2$. (They gave constructions of adaptively secure protocols for the regime $n/3 \leq t \leq n/2$.) Their work calls into question the feasibility of realizing adaptively secure multi-party computation (MPC) for $t > n/2$ in point-to-point networks.

A closer look at the Hirt-Zikas result shows that they make a very strong assumption regarding the adversary (or, alternately, a very weak assumption regarding the communication network): namely, they assume the adversary has the ability to corrupt parties *in the middle of a round*, in between sending messages to two other parties in the network. Specifically, their impossibility result crucially relies on the fact that the following sequence of events can occur when an honest party P sends its messages in some round:

1. The adversary (who has already corrupted some of the other players) receives the message(s) sent to it by P.

2. Based on this, the adversary then decides whether to corrupt P.

3. If the adversary corrupts P, it can then send messages of its choice (on behalf of P) to the remaining parties in the same round.

While the above is consistent with theoretical models for *asynchronous* cryptographic protocols, as well as some previous treatments of adaptive security in the synchronous setting (e.g., [11]), allowing such adversarial behavior seems unrealistically pessimistic: in the real world, implementing such an attack would require either an exceedingly fast adversary or an extremely slow network. A more realistic model of synchronous communication (see, e.g., [10]) is one in which messages sent by honest parties within any given round are delivered *atomically* to all other parties.[1]

Importantly, however, the attacks that were demonstrated by Hirt and Zikas [57] on existing broadcast protocols remain valid even if we assume atomic message delivery. Consider, for example, the authenticated broadcast protocol of Dolev and Strong [29] where, at a high level, in the first

[1] We still allow *rushing*, meaning that corrupted parties may receive their messages in some round before having to send any of their own. This reflects the fact that corrupted parties can choose to delay their own communication. However, it seems unrealistic to assume that honest parties would delay sending any of their own messages.

round the sender digitally signs and sends his message to all the other parties, while in subsequent rounds parties append their signatures and forward the result. Roughly, if any party ever observes valid signatures of the sender on two *different* messages then that party forwards both signatures to all other parties and disqualifies the sender (and all parties output some default message). The Hirt-Zikas attack against this protocol works as follows: a corrupted party P in the network waits to receive the initial message from the (uncorrupted) sender. If P likes the message sent by the sender then P runs the protocol honestly. If P does not like the message sent by the sender then P adaptively corrupts the sender, uses the sender's signing key to generate a valid signature on another message (in the next round), and thus ensures that the sender will be disqualified and the default message used.

While this outcome might be acceptable with respect to a *property-based* definition (since the sender is corrupted by the end of the protocol in the second case), the outcome is not something that should be possible with respect to a *simulation-based* definition (since corruption of the sender depends on the sender's initial input). Realizing the latter, stronger definition is a natural goal; moreover, a simulation-based definition is especially critical for broadcast which is typically used as a sub-protocol within some larger protocol.

Given that the Hirt-Zikas attack applies even when atomic message delivery is assumed, one might wonder whether their impossibility result holds in that model as well. Alternately, one may be willing to give up on "full" broadcast and hope that some weaker form of broadcast might be sufficient to achieve secure MPC for $t > n/2$. (Indeed, in the presence of a dishonest majority the standard definitions of secure MPC give up on guaranteed output delivery, so in particular secure MPC for $t > n/2$ does not imply broadcast for $t > n/2$.) These are the questions with which we concern ourselves in this chapter.

Our results and techniques. As our main result in this chapter, we show that the Hirt-Zikas impossibility result does *not* apply in the synchronous model with atomic message delivery. That is, we show a construction of an adaptively secure broadcast protocol tolerating an arbitrary number of corruptions in this communication model. We prove security of our protocol within the UC framework [11], under the usual assumptions that a PKI and digital signatures are available. We stress that we require only a standard PKI where each party chooses their public key and all other parties know it; in particular, we do not require the stronger "registered public key" model considered in [2].

The main idea for avoiding the Hirt-Zikas attack is to design a protocol where the adversary does not learn the (honest) sender's message until agreement has already been reached; that way, the adversary must make its decision as to whether or not to corrupt the sender independently of the sender's input. This suggests the following two-stage approach: First, the signer broadcasts a *commitment* to its message; once agreement is reached, the signer then decommits. While this does prevent the above attack, it also introduces a new problem when we try to prove security, since the simulator must commit to the sender's message before knowing what the sender's message is! (Since the sender might still get corrupted in the middle of the protocol, it also does not work for the simulator to obtain the output of the broadcast functionality before starting the simulation.) This could be handled by using a universally composable commitment scheme (e.g., [15]), which satisfies even stronger properties, but we would prefer to avoid the stronger setup assumptions that are required for constructing universally composable commitments [15].

Instead, we show that a very weak form of commitment suffices to make the approach sound. Specifically, we use commitment schemes that (informally) are hiding and binding for an *honest* sender, but where binding can be (easily) violated by a *dishonest* sender. To see why this works, note that the only time binding is needed is when the adversary corrupts the sender *after* the sender has already committed to its message. Since the sender in that case was honest at the time the commitment was generated, the binding property holds and the adversary will not be able to change the committed value. On the other hand, the simulator can behave as a dishonest sender and generate a commitment that it can later open to any desired value, and in particular to the sender's true input in case the sender remains uncorrupted until the end of the protocol. We show that commitment schemes with the desired properties can be constructed from one-way functions (which are, in turn, implied by digital signature schemes); thus, in summary, we obtain an adaptively secure, universally-composable broadcast protocol assuming a PKI and digital signatures.

We also study the impact of adaptive attacks on secure multi-party computation protocols (where broadcast is commonly used as a subcomponent), and establish the variants of broadcast that are needed in this setting. Interestingly, we show that the full functionality of broadcast is not needed in order to obtain secure MPC for $t \geq n/2$; instead, a weaker form of broadcast — which can be realized even in the Hirt-Zikas communication model — suffices.

Organization. In Section 4.1 we present our network model and elaborate on simulation-based definitions of security. Section 4.2 defines various notions of broadcast. In Section 4.3 we introduce a special type of commitment scheme, and we show how to construct such schemes in Section 4.3.1. In Section 4.4 we show how to use such commitments to realize adaptively secure broadcast in the atomic communication model. We discuss the consequences for adaptively secure multi-party computation in Section 4.5.

4.1 Preliminaries

4.1.1 Network Model

We consider a network with synchronous communication, where there is a set of n players (probabilistic polynomial-time Turing machines) $\mathcal{P} = \{P_1, P_2, \cdots, P_n\}$ connected by point-to-point authenticated channels. Each round of the protocol proceeds as follows. The honest parties send their messages for that round, and these messages are received by all parties (both honest and corrupted). The adversary may then choose to corrupt additional players, and then it sends messages on behalf of the parties who were corrupted at the beginning of that round. (This models a *rushing* adversary.) When it is done, the adversary must then "advance the clock" to the next round. We allow the adversary to corrupt any $t < n$ of the parties, and to behave in an arbitrary ("Byzantine") manner. We stress that we do not assume erasures.

We stress that our model is different from that considered by Hirt and Zikas [57], where in each round the honest parties' messages are first delivered *to the corrupted parties only* and then the adversary is allowed to corrupt additional parties and decide what messages to send on behalf of those parties to other honest players. In contrast, we assume that honest parties' messages are delivered "atomically", which is equivalent to assuming that adversarial corruption cannot occur in the time interval between when a message is sent and when it is received. We sometimes refer to our model as "atomic", and to the Hirt-Zikas model as "non-atomic".

4.1.2 Simulation-Based Security

We use a simulation-based definition of security in Chapter 4, which is in line with work in the area of cryptographic-protocol design but which differs from most of the classical work on Byzantine agreement and broadcast. Simulation-based definition are formulated by defining an "ideal" version of some desired functionality that is implemented by a trusted third-party; a protocol is secure if the protocol "emulates" this ideal world no matter what the adversary does. One advantage of a simulation-based approach is that it simultaneously captures *all* the properties that are guaranteed by the ideal world, without having to enumerate some list of desired properties. Simulation-based definitions are also useful for applying *composition theorems* that enable proving security of protocols that use other protocols as sub-routines.

We formulate our simulation-based definitions by presenting appropriate functionalities within the UC framework. We give a brief introduction to this model, and refer readers elsewhere for more details [11]. The basic entities involved are parties $P_1, \dots, P_n$, an adversary $\mathcal{A}$, and an "environment" $\mathcal{Z}$. The environment $\mathcal{Z}$ gives inputs to and receives outputs from all the parties; it also interacts with $\mathcal{A}$ in an arbitrary way throughout its execution. In the ideal world, the parties and $\mathcal{Z}$ all interact via an ideal functionality $\mathcal{F}$: the parties send their inputs to (with corrupted parties sending anything they like) and receive outputs from $\mathcal{F}$, and $\mathcal{A}$ interacts with $\mathcal{F}$ as specified by $\mathcal{F}$ itself. We let $\textsc{ideal}_{\mathcal{F},\mathcal{A},\mathcal{Z}}(n)$ denote the output of $\mathcal{Z}$ in this case. In the real world, the parties run some protocol π with the corrupted parties behaving arbitrarily as directed by $\mathcal{A}$. We let $\textsc{real}_{\pi,\mathcal{A},\mathcal{Z}}(n)$ denote the output of $\mathcal{Z}$ in that case. A protocol π *securely realizes the functionality* $\mathcal{F}$ if for any probabilistic polynomial-time (PPT) real-world adversary $\mathcal{A}$ there exists a PPT ideal-world adversary $\mathcal{S}$ (often called a *simulator*) such that for all PPT environments $\mathcal{Z}$ the following is negligible:

$$\left|\Pr[\textsc{real}_{\pi,\mathcal{A},\mathcal{Z}}(n) = 1] - \Pr[\textsc{ideal}_{\mathcal{F},\mathcal{S},\mathcal{Z}}(n) = 1]\right|.$$

Say we want to design a protocol for some functionality $\mathcal{F}$. It is often helpful to design and reason about this in a *hybrid* world where the parties can run a protocol π while at the same time having access to some ideal functionality $\mathcal{G}$. We let $\textsc{hybrid}^{\mathcal{G}}_{\pi,\mathcal{A},\mathcal{Z}}(n)$ denote the output of $\mathcal{Z}$ in that case, and say that π *securely realizes $\mathcal{F}$ in the $\mathcal{G}$-hybrid model* if for any PPT hybrid-world adversary $\mathcal{A}$ there exists a PPT ideal-world adversary $\mathcal{S}$ such that for all PPT environments $\mathcal{Z}$ we have $\left|\Pr[\textsc{hybrid}^{\mathcal{G}}_{\pi,\mathcal{A},\mathcal{Z}}(n) = 1] - \Pr[\textsc{ideal}_{\mathcal{F},\mathcal{S},\mathcal{Z}}(n) = 1]\right|$. In the UC framework, the following useful composition result holds: if π securely realizes $\mathcal{F}$ in the $\mathcal{G}$-hybrid model, and ρ is any protocol that securely realizes $\mathcal{G}$, then the composed protocol π^ρ securely realizes $\mathcal{F}$ (in the real world).

4.2 Definitions of Broadcast

As mentioned previously, classical results show that broadcast (or even relaxed broadcast) cannot be realized for $t \geq n/3$ corrupted parties in a "plain model", and so some setup must be considered if we wish to go beyond this bound. As stated in the Introduction, we assume a PKI and digital signatures. Within the UC framework, this is modeled by the certificate functionality $\mathcal{F}_{\text{CERT}}$ introduced in [12]. This functionality provides both message-signing capability as well as binding between a signature and a party in the network, and thus simultaneously captures both the presence of a PKI and the ability to issue signatures.

Our definitions of broadcast are induced by ideal functionalities in the UC framework. Namely, we say a protocol π achieves (strong) *broadcast* if it securely realizes the functionality $\mathcal{F}_{\text{BC}}$ shown in Figure 4.1; it achieves *relaxed broadcast* if it securely realizes the functionality $\mathcal{F}_{\text{RBC}}$ given in Figure 4.2. Our definition of broadcast is essentially standard, though one can also consider a definition where the sender's message m is not revealed to $\mathcal{S}$. (I.e., our definition does not guarantee secrecy for m; note that this only makes a difference when $\mathcal{S}$ corrupts no parties.) Our definition of relaxed broadcast is from [57].

It is instructive to examine the two functionalities in light of the Hirt-Zikas attack. Observe that $\mathcal{F}_{\text{BC}}$ does not allow their attack (and so any protocol securely realizing $\mathcal{F}_{\text{BC}}$ must not be

Functionality $\mathcal{F}_{\mathrm{BC}}$

The functionality interacts with an adversary $\mathcal{S}$ and a set $\mathcal{P} = \{P_1, \ldots, P_n\}$ of parties.

- Upon receiving $(\mathsf{Bcast}, \mathsf{sid}, m)$ from P_i, send $(\mathsf{Bcast}, \mathsf{sid}, P_i, m)$ to all parties in $\mathcal{P}$ and to $\mathcal{S}$.

Figure 4.1: The broadcast functionality.

susceptible to the attack) since the adversary cannot change the sender's message m unless the adversary corrupts the sender P_i in advance, before it learns m. On the other hand, $\mathcal{F}_{\mathrm{RBC}}$ allows their attack: this is so because the adversary can first learn m (in step 1) and then decide whether to corrupt the sender P_i based on that information; if the adversary decides to corrupt P_1 then the adversary is allowed change the message that will be received by all the other parties in step 2.

The following result was proved in [57]:

Lemma 27 *The Dolev-Strong protocol [29] securely realizes $\mathcal{F}_{\mathrm{RBC}}$ in the $\mathcal{F}_{\mathrm{CERT}}$-hybrid model against an adaptive adversary corrupting any $t < n$ parties.*

In fact, the above result holds even in the non-atomic communication model.

It is also possible to define a stronger variant of $\mathcal{F}_{\mathrm{RBC}}$, called $\mathcal{F}^{+}_{\mathrm{RBC}}$, that more closely corresponds to what is actually accomplished by the Hirt-Zikas attack. The difference between $\mathcal{F}_{\mathrm{RBC}}$ and $\mathcal{F}^{+}_{\mathrm{RBC}}$ is that the latter only allows the adversary to have $m' = \perp$. That is, the adversary is allowed to adaptively corrupt the sender (based on the sender's original message) and thereby cause agreement on an error, but is unable to cause agreement on some other valid message. $\mathcal{F}^{+}_{\mathrm{RBC}}$ can be realized fairly easily in the $\mathcal{F}_{\mathrm{RBC}}$-hybrid model using the commitment scheme defined in the following section. Alternately, it can be realized directly in the $\mathcal{F}_{\mathrm{CERT}}$-hybrid model using an appropriate variant of the Dolev-Strong protocol.

Functionality $\mathcal{F}_{\mathrm{RBC}}$

The functionality interacts with an adversary $\mathcal{S}$ and a set $\mathcal{P} = \{P_1, \ldots, P_n\}$ of parties.

1. Upon receiving $(\mathsf{Bcast}, \mathsf{sid}, m)$ from P_i, send (Bcast, sid, P_i, m) to $\mathcal{S}$.
2. Upon receiving m' from $\mathcal{S}$, do:
   - If P_i is corrupted, send (Bcast, sid, P_i, m') to all parties in $\mathcal{P}$;
   - If P_i is not corrupted, send (Bcast, sid, P_i, m) to all parties in $\mathcal{P}$.

Figure 4.2: The relaxed broadcast functionality.

4.3 Honest-Binding Commitment Schemes

Commitment schemes are a standard cryptographic tool. Roughly, a commitment scheme allows a sender S to generate a commitment com to a message m in such a way that (1) the sender can later open the commitment to the original value m (*correctness*); (2) the sender cannot generate a commitment that can be opened to two different values (*binding*); and (3) the commitment reveals nothing about the sender's value m until it is opened (*hiding*). For our application, we need a variant of standard commitments that guarantees binding when the sender is honest but ensures that binding can be violated if the sender is dishonest. (In the latter case, we need some additional properties as well; these will become clear in what follows.) Looking ahead, we will use such commitment schemes to construct a broadcast protocol in the following way: the sender will first generate and send a *commitment* to its message, and then send the decommitment information needed to open the commitment. In the simulation for the case when the sender P_i starts out uncorrupted, we will have the simulator $\mathcal{S}$ generate a commitment *dishonestly.* This will give $\mathcal{S}$ the flexibility to break binding and open the commitment to any desired message (if needed), while also being able to ensure binding (when desired) by claiming that it generated the commitment honestly.

We consider only non-interactive commitment schemes. For simplicity, we define our schemes in such a way that the decommitment information consists of the sender's random coins ω that it used when generating the commitment.

Definition 7 *A (non-interactive)* commitment scheme *for message space $\{\mathcal{M}_k\}$ is a pair of* PPT *algorithms S, R such that for all $k \in \mathbb{N}$, all messages $m \in \mathcal{M}_k$, and all random coins ω it holds that $R(m, S(1^k, m; \omega), \omega) = 1$.*

A commitment scheme for message space $\{\mathcal{M}_k\}$ is honest-binding *if it satisfies the following:*

Binding (for an honest sender) *For all* PPT *algorithms $\mathcal{A}$ (that maintain state throughout their execution), the following is negligible in k:*

$$\Pr\left[\begin{array}{r} m \leftarrow \mathcal{A}(1^k); \omega \leftarrow \{0,1\}^*; \mathsf{com} \leftarrow S(1^k, m; \omega); (m', \omega') \leftarrow \mathcal{A}(\mathsf{com}, \omega) : \\ R(m', \mathsf{com}, \omega') = 1 \bigwedge m' \neq m \end{array}\right]$$

Equivocation *There is an algorithm $\tilde{S} = (\tilde{S}_1, \tilde{S}_2)$ such that for all* PPT *$\mathcal{A}$ (that maintain state throughout their execution) the following is negligible:*

$$\left| \Pr\left[\begin{array}{c} m \leftarrow \mathcal{A}(1^k); \\ \omega \leftarrow \{0,1\}^*; \mathsf{com} \leftarrow S(1^k, m; \omega) : \\ \mathcal{A}(1^k, \mathsf{com}, \omega) = 1 \end{array}\right] - \Pr\left[\begin{array}{c} (\mathsf{com}, \mathsf{st}) \leftarrow \tilde{S}_1(1^k); \\ m \leftarrow \mathcal{A}(1^k);\ \omega \leftarrow \tilde{S}_2(\mathsf{st}, m) : \\ \mathcal{A}(1^k, \mathsf{com}, \omega) = 1 \end{array}\right] \right|$$

♢

Equivocation implies the standard hiding property, namely, that for all PPT algorithms $\mathcal{A}$ (that maintain state throughout their execution) the following is negligible:

$$\left| \Pr\left[\ (m_0, m_1) \leftarrow \mathcal{A}(1^k);\ b \leftarrow \{0,1\}; \mathsf{com} \leftarrow S(1^k, m_b) : \mathcal{A}(\mathsf{com}) = b\ \right] - \frac{1}{2} \right|.$$

We also observe that if (com, ω) are generated by $(\tilde{S}_1, \tilde{S}_2)$ for some message m as in the definition above, then binding still holds: namely, no PPT adversary can find (m', ω') with $m' \neq m$ such that $R(m', \mathsf{com}, \omega') = 1$.

4.3.1 Constructing Honest-Binding Commitment

We show two constructions of honest-binding commitment schemes. The proofs that these schemes satisfy Definition 7 are relatively straightforward, and are therefore omitted.

The first construction, based on the commitment scheme of Naor [74], relies on the minimal assumption that one-way functions exist. We describe the scheme for committing single-bit messages, though it could be extended to arbitrary length messages in the obvious way. In the following, G is a length-tripling pseudorandom generator.

```
$S(1^k, m; \omega)$
parse $\omega$ as $\mathsf{crs} \| r$,
  with $|\mathsf{crs}| = 3k$
  and $|r| = k$;
$c := G(r) \oplus (\mathsf{crs} \cdot m)$;
$\mathsf{com} := (\mathsf{crs}, c)$;
return $\mathsf{com}$;
```

```
$R(m, (\mathsf{crs}, c), \omega)$
parse $\omega$ as $\mathsf{crs} \| r$,
  with $|\mathsf{crs}| = 3k$
  and $|r| = k$;
if $c \stackrel{?}{=} G(r) \oplus (\mathsf{crs} \cdot m)$
    return 1;
else return 0;
```

```
$\tilde{S}_1(1^k)$
$r_0, r_1 \leftarrow \{0,1\}^k$;
$\mathsf{crs} := G(r_0) \oplus G(r_1)$;
$c := G(r_0)$;
$\mathsf{com} := (\mathsf{crs}, c)$;
$\mathsf{st} := (r_0, r_1, \mathsf{com})$;
return $(\mathsf{com}, \mathsf{st})$;
```

```
$\tilde{S}_2(\mathsf{st}, m)$
parse $\mathsf{st}$ as $(r_0, r_1, \mathsf{com})$;
parse $\mathsf{com}$ as $(\mathsf{crs}, c)$;
if $m \stackrel{?}{=} 0$
   $\omega := \mathsf{crs} \| r_0$;
else
   $\omega := \mathsf{crs} \| r_1$;
return $\omega$;
```

Next, we show an efficient scheme that allows for direct committments to strings. This construction, based on the Pedersen commitment scheme [80], relies on the discrete-logarithm assumption. In the following, we let $\mathbb{G}$ be a cyclic group of order q, with generator $g \in \mathbb{G}$. (For simplicity, we view $(\mathbb{G}, q, g)$ as public parameters, though they could just as well be generated by the sender.)

```
$S(1^k, m; \omega)$
Parse $\omega$ as $h \| x$,
  with $h \in \mathbb{G}$
  and $x \in \mathbb{Z}_q$;
return $\mathsf{com} := (h, g^m h^x)$;
```

```
$R(m, \mathsf{com}, \omega)$
Parse $\omega$ as $h \| x$,
  with $h \in \mathbb{G}$
  and $x \in \mathbb{Z}_q$;
if $\mathsf{com} \stackrel{?}{=} (h, g^m h^x)$
    return 1;
else return 0;
```

```
$\tilde{S}_1(1^k)$
$r, y \leftarrow \mathbb{Z}_q$;
$\mathsf{com} := (g^r, g^y)$
return $(\mathsf{com}, (r, y))$
```

```
$\tilde{S}_2((r, y), m)$
if $r \stackrel{?}{=} 0$ return $\perp$;
$x := (y - m) \cdot r^{-1} \bmod q$;
return $\omega := g^r \| x$;
```

Protocol π_{BC}

The protocol is carried out among a set $\mathcal{P} = \{P_1, \ldots, P_n\}$ of parties. We let $D \in \mathcal{P}$ denote the sender. We let (S, R) be a non-interactive commitment scheme.

- **Stage 1:** Upon receiving input $(\mathsf{Bcast}, \mathsf{sid}, m)$ from the environment $\mathcal{Z}$, the sender D chooses random $\omega \leftarrow \{0,1\}^*$, computes $\mathsf{com} := S(1^k, m; \omega)$, and sends $(\mathsf{Bcast}, \mathsf{sid}, \mathsf{com})$ to $\mathcal{F}_{\text{RBC}}$. Let com' denote the value received by the honest parties in this stage (note that this value is the same for all honest parties).

- **Stage 2:** Upon receiving $(\mathsf{Bcast}, \mathsf{sid}, D, \mathsf{com})$ from $\mathcal{F}_{\text{RBC}}$, the sender D sends (m, ω) to every other party over point-to-point channels.

- **Stage 3:** The following is done by each party P_i: Let (m'_i, ω'_i) denote the value that P_i received from D in stage 2. (If P_i receives nothing, it takes (m'_i, ω'_i) as some default values.) P_i sends $(\mathsf{Bcast}, \mathsf{sid}, (m'_i, \omega'_i))$ to $\mathcal{F}_{\text{RBC}}$.

- **Stage 4:** Each party P_j receives messages $\{(\mathsf{Bcast}, \mathsf{sid}, P_i, (m''_i, \omega''_i))\}_{i \in [n]}$ from $\mathcal{F}_{\text{RBC}}$, taking (m''_i, ω''_i) as some default values if nothing is received (note that the (m''_i, ω''_i) values are the same for all honest parties). Each party P_j then decides on its output as follows: Let $\mathsf{valid} = \{i \in [n] \mid R(m''_i, \mathsf{com}', \omega''_i) = 1\}$. If valid is empty, then output some default value. Otherwise, let k be the smallest value in valid and output $m_j = m''_k$.

Figure 4.3: A protocol realizing $\mathcal{F}_{\text{BC}}$ in the $\mathcal{F}_{\text{RBC}}$-hybrid model.

4.4 An Adaptively Secure Broadcast Protocol

In this section we show a protocol that securely realizes $\mathcal{F}_{\text{BC}}$ in the $\mathcal{F}_{\text{CERT}}$-hybrid model, in the presence of $t < n$ adaptive corruptions. The challenge of realizing $\mathcal{F}_{\text{BC}}$, and the property that is exactly exploited in the Hirt-Zikas attack on existing protocols, is that when the sender is uncorrupted then the adversary should not learn the sender's message unless all honest parties will (eventually) *agree* on that message (cf. Figure 4.1). In [57], the authors construct a broadcast protocol for $t < n/2$ by having the sender use verifiable secret sharing (VSS) to "commit" to its message before revealing it. (For $t = n/2$ they use a slight variant of this idea.) This approach works even in the non-atomic communication setting; however, it requires at least half of the parties to be honest.

Our approach is to use computationally secure commitment schemes in place of VSS. That is, we first have the sender announce a *commitment* to its message; once agreement on this commitment is reached, the sender then decommits. (We add an additional stage in which the sender's decommitment is "echoed" by all parties; this prevents a dishonest sender from sending valid decommitment information to some honest parties but not others.) In order to simulate this protocol, we have the sender use honest-binding commitments as introduced in the previous section.

The details of our protocol π_{BC} are presented in Figure 4.3. We describe our protocol in the $\mathcal{F}_{\text{RBC}}$-hybrid model. Since $\mathcal{F}_{\text{RBC}}$ can be securely realized in the $\mathcal{F}_{\text{CERT}}$-hybrid model (cf. Lemma 27), this implies that $\mathcal{F}_{\text{BC}}$ can be securely realized in the $\mathcal{F}_{\text{CERT}}$-hybrid model as well.

Theorem 28 *Let (S, R) be an honest-binding commitment scheme. Then protocol π_{BC} securely* *realizes $\mathcal{F}_{\mathrm{BC}}$ in the $\mathcal{F}_{\mathrm{RBC}}$-hybrid model against an adaptive adversary corrupting any $t < n$ of the parties.*

The above theorem holds only in the atomic communication model considered here; protocol π_{BC} does *not* securely realize $\mathcal{F}_{\mathrm{BC}}$ in the non-atomic communication model of [57]. (Indeed, by the impossibility result proven in [57], it cannot.) Atomic communication is used crucially in the second stage of our protocol when the sender transmits decommitment information to all the parties. (Observe this is the only step in our protocol in which parties communication directly, rather than via the ideal functionality $\mathcal{F}_{\mathrm{RBC}}$.) If non-atomic communication were assumed, then the adversary could learn the decommitment information (and thus the sender's message) first, and then decide to corrupt the sender and *not* transmit the decommitment information to any of honest parties.

Proof Let $\mathcal{A}$ be an active, adaptive adversary that interacts with players running the above protocol in the $\mathcal{F}_{\mathrm{RBC}}$-hybrid model. We construct an adversary (simulator) $\mathcal{S}$ running in the ideal world with access to functionality $\mathcal{F}_{\mathrm{BC}}$, such that no PPT environment $\mathcal{Z}$ can distinguish whether it is interacting with $\mathcal{A}$ and parties running π_{BC} in the $\mathcal{F}_{\mathrm{RBC}}$-hybrid model, or whether it is interacting with $\mathcal{S}$ and (dummy) parties communicating directly with $\mathcal{F}_{\mathrm{BC}}$. The simulator $\mathcal{S}$ starts by internally invoking the adversary $\mathcal{A}$, and forwarding all messages between $\mathcal{A}$ and $\mathcal{Z}$ in the usual way. The simulator will simulate both the ideal functionality $\mathcal{F}_{\mathrm{RBC}}$ for $\mathcal{A}$, as well as an execution of protocol π_{BC}.

In our description of $\mathcal{S}$, we distinguish two cases depending on whether or not the sender P_i is corrupted at the outset.

Case 1: We first treat the easier case where D is corrupted at the outset. Here, $\mathcal{A}$ requests to corrupt D (in the hybrid world) and so $\mathcal{S}$ corrupts D (in the ideal world). Any additional corruptions that $\mathcal{A}$ requests throughout its execution can be easily simulated by $\mathcal{S}$, so we do not mention them.

When $\mathcal{Z}$ provides input to D, this input is read by $\mathcal{S}$ who forwards it to $\mathcal{A}$. Then $\mathcal{A}$ begins running the first stage of π_{BC} (on behalf of the corrupted D) by specifying some message $(\mathsf{Bcast}, \mathsf{sid}, \mathsf{com}')$ to send to $\mathcal{F}_{\mathrm{RBC}}$. The simulator $\mathcal{S}$ stores com', and simulates the response of $\mathcal{F}_{\mathrm{RBC}}$ by giving $(\mathsf{Bcast}, \mathsf{sid}, D, \mathsf{com}')$ to $\mathcal{A}$ (and all corrupted parties). Next, $\mathcal{A}$ (now executing the second stage of π_{BC}) decides on messages (m'_i, ω'_i) to send to each honest party P_i on behalf of D. In response, $\mathcal{S}$ simulates the third stage of π_{BC} by giving $(\mathsf{Bcast}, \mathsf{sid}, P_i, (m'_i, \omega'_i))$ to $\mathcal{A}$ for every honest party P_i. For each such P_i, the adversary $\mathcal{A}$ may then choose to (corrupt P_i and) replace (m'_i, ω'_i) by some other message (m''_i, ω''_i). Once $\mathcal{A}$ has sent some (m''_i, ω''_i) to the appropriate instance of $\mathcal{F}_{\mathrm{RBC}}$ for all P_i, the simulator simulates the output of $\mathcal{F}_{\mathrm{RBC}}$ for all corrupted parties in the obvious way. Finally $\mathcal{A}$, executing the third stage of π_{BC} on behalf of the remaining corrupted parties, specifies messages $(\mathsf{Bcast}, \mathsf{sid}, (m''_i, \omega''_i))$ that each such party P_i should send to $\mathcal{F}_{\mathrm{RBC}}$.

$\mathcal{S}$ now has values (m''_i, ω''_i) for every $P_i \in \mathcal{P}$, defined by the output of each appropriate (simulated) instance of $\mathcal{F}_{\mathrm{RBC}}$ in the (simulated) third stage of the protocol. $\mathcal{S}$ defines a set valid and determines k, m''_k as prescribed by the protocol. It then sends $(\mathsf{Bcast}, \mathsf{sid}, m''_k)$ (on behalf of D) to its own ideal functionality $\mathcal{F}_{\mathrm{BC}}$.

It is not hard to see that $\mathcal{S}$ provides a perfect simulation. The view of $\mathcal{A}$ is clearly identical whether it is running in the $\mathcal{F}_{\mathrm{RBC}}$-hybrid model or whether it is being run as a sub-routine by $\mathcal{S}$ in the ideal world with access to $\mathcal{F}_{\mathrm{BC}}$. As for the outputs of the honest parties (i.e., those that are honest by the end of the protocol execution), note that if $\mathcal{A}$ were running in the $\mathcal{F}_{\mathrm{RBC}}$-hybrid model then every honest party P_j would receive com' in the first stage and $\{(m''_i, \omega''_i)\}_{P_i \in \mathcal{P}}$ in the third stage, and would thus decide on output m''_k exactly as $\mathcal{S}$ does. Since $\mathcal{S}$ sends m''_k to $\mathcal{F}_{\mathrm{BC}}$, the output of each honest party in the ideal world is also m''_k. We remark that the fact that the commitment scheme is not binding (for a malicious sender) is irrelevant here.

Case 2: We now turn to the more difficult case where D is not corrupted at the outset. As before, adaptive corruptions of parties other than D can be handled easily, so we do not mention it. Corruption of D will, however, be explicitly mentioned.

$\mathcal{S}$ begins by computing $(\mathsf{com}, \mathsf{st}) \leftarrow \tilde{S}_1(1^k)$. It then simulates the first stage of π_{BC} (on behalf of the honest D) by giving to $\mathcal{A}$ the message $(\mathsf{Bcast}, \mathsf{sid}, D, \mathsf{com})$ on behalf of $\mathcal{F}_{\mathrm{RBC}}$. At this point, $\mathcal{A}$ can choose whether to corrupt D or not, and we further divide our description of $\mathcal{S}$ depending on which is the case.

If $\mathcal{A}$ requests to corrupt D, then $\mathcal{S}$ corrupts D and waits until it receives input $(\mathsf{Bcast}, \mathsf{sid}, m)$ from $\mathcal{Z}$. At that point, $\mathcal{S}$ computes $\omega \leftarrow \tilde{S}_2(\mathsf{st}, m)$ and gives m and ω to $\mathcal{A}$ as the state of D. The remainder of the simulation then proceeds exactly as in the case when P_i was corrupted at the outset. (Note in particular that $\mathcal{A}$ may choose to change com to some other value com'.)

If $\mathcal{A}$ does not corrupt D, then $\mathcal{S}$ waits until it receives a message $(\mathsf{Bcast}, \mathsf{sid}, D, m)$ from its ideal functionality $\mathcal{F}_{\mathrm{BC}}$. (Note that at this point, the output of every honest party in the ideal world is m.) $\mathcal{S}$ then computes $\omega \leftarrow \tilde{S}_2(\mathsf{st}, m)$, and simulates the second phase of the protocol by sending (m, ω) to every corrupted party. The remainder of the protocol is simulated in the obvious way, essentially the same as before (with the only difference being that it provides state m, ω to $\mathcal{A}$ if D is ever corrupted).

In this case, $\mathcal{S}$ provides a computationally indistinguishable simulation for $\mathcal{Z}$. The only difference between the view of $\mathcal{A}$ in the above simulation and the view of $\mathcal{A}$ when it is running in the $\mathcal{F}_{\mathrm{RBC}}$-hybrid model is with regard to (com, ω): in the former case these are produced using $(\tilde{S}_1, \tilde{S}_2)$, whereas in the latter case these are produced using the honest sender algorithm. Definition 7 guarantees that these distributions are computationally indistinguishable. As for the outputs of the honest parties, if D is corrupted during stage 1 then the argument is as given previously. If D is not corrupted during stage 1, then we need to argue that with all but negligible probability every honest party would output m in that case in the $\mathcal{F}_{\mathrm{RBC}}$-hybrid world (since, as noted above, every honest party outputs m in that case in the ideal world). This follows from the honest-binding property of Definition 7. ■

4.5 Adaptively Secure Multi-Party Computation

In the previous section we showed a protocol (call it bc) that securely realizes the broadcast functionality $\mathcal{F}_{\mathrm{BC}}$ in the presence of an adaptive adversary corrupting any number of parties. Given any protocol π (e.g., the one of [17]) for securely computing some function f in the presence of an adaptive adversary corrupting any number of parties in the $\mathcal{F}_{\mathrm{BC}}$-hybrid model (i.e., protocol π assumes an ideal broadcast channel), the composed protocol π^{bc} securely computes f in the presence of an adaptive adversary corrupting any number of parties in the $\mathcal{F}_{\mathrm{CERT}}$-hybrid model, using point-to-point communication only. The above is stated in the UC framework, but an analogous composition theorem could be stated with respect to "stand-alone" notions of security as well [10]. (We refer the reader to [47] for a detailed treatment of security notions for MPC with dishonest majority.)

Even given the above, it is interesting to explore whether adaptively secure MPC can be achieved in the weaker $\mathcal{F}_{\mathrm{RBC}}$-hybrid model, for at least two reasons:

- If we take as our communication model the non-atomic, point-to-point model of Hirt-Zikas, it is impossible to realize $\mathcal{F}_{\mathrm{BC}}$ when $t > n/2$. Thus, if we want to realize adaptively secure MPC for $t > n/2$ in this communication model, some other approach is needed.

- Even in the atomic communication model, one may prefer to base adaptively secure MPC on relaxed broadcast rather than broadcast since protocols for the former may be more efficient than protocols for the latter.

Note that, in the case of dishonest majority, adaptively secure MPC does not imply adaptively secure broadcast because the usual notions of security for MPC do not guarantee output delivery or fairness (see [47] for a more extensive treatment) — these properties are, in general, not achievable [21] — whereas definitions of security for broadcast do require guaranteed output delivery. In particular, the Hirt-Zikas impossibility result for adaptively secure broadcast in the non-atomic communication model says nothing about the feasibility of adaptively secure MPC in that setting.

Although we cannot claim that all adaptively secure MPC protocols using broadcast remain secure when broadcast is replaced with relaxed broadcast, it turns out that specific protocols from the literature do remain secure in that case. Once again, we focus on protocols proven secure in the UC framework, though we expect these results would extend to protocols analyzed in the "stand-alone" setting as well.

Specifically, consider the adaptively secure MPC protocol π of Canetti, Lindell, Ostrovsky, and Sahai [17], which relies on a broadcast channel. We first observe that the protocol remains secure even in the non-atomic communication model. In either communication model, the protocol also remains secure if the broadcast channel is replaced with relaxed broadcast. At a high level, the reason is that the messages that are broadcast are always commitments to some values, except in the last round where the broadcast messages reveal the output. The ability to corrupt a sender based on the message being broadcast is "useless" in the former case; in the latter case such an attack corresponds to preventing output delivery/violating fairness, something which is permitted by the definitions of security when there is a dishonest majority. We remark that the advantage of using relaxed broadcast as opposed to the "echo broadcast" protocol from [47] is that the former ensures agreement on abort.

Even given the above, there are several reasons to securely realize $\mathcal{F}_{\mathrm{BC}}$ rather than be contended with $\mathcal{F}_{\mathrm{RBC}}$. First, one may be interested in broadcast itself, rather than as a sub-protocol for some larger task. Furthermore, there is an advantage to working with $\mathcal{F}_{\mathrm{BC}}$ in that it can be safely used to instantiate the broadcast channel in *arbitrary* protocols, so one can avoid having to examine protocols on a case-by-case basis to determine whether $\mathcal{F}_{\mathrm{RBC}}$ suffices.

Chapter 5

Round Complexity of Perfect VSS in Point-to-Point Networks

In this chapter, we study the round complexity of perfect VSS in point-to-point networks. Perfect VSS is known to be possible if and only if $t < n/3$ [5, 27]. Previous research investigating the round complexity of VSS, surveyed further below, has focused on optimizing the round complexity *assuming a broadcast channel is available "for free"*. (We remark that broadcast is essential for VSS, in a way we make precise below.) As argued previously [63], however, if the ultimate goal is to optimize the round complexity of protocols for point-to-point networks (where protocols are likely to be run), then it is preferable to minimize the *number of rounds in which broadcast is used* rather than to minimize the *total number of rounds.* This is due to the high overhead of emulating a broadcast channel over a point-to-point network: deterministic broadcast protocols require $\Omega(t)$ rounds [33]; known randomized protocols [31, 35, 62] require only $O(1)$ rounds in expectation, but the constant is rather high.

As a concrete example (taken from [63]) to illustrate the point, consider the VSS protocol of Micali and Rabin [73] and the "round-optimal" VSS protocol of Fitzi et al. [36]. The former uses 16 rounds but only a single round of broadcast; the latter uses 3 rounds, two of which require broadcast. Compiling these protocols for a point-to-point network using the most round-efficient techniques known (see [63]), the Micali-Rabin protocol requires 26 rounds in expectation while the protocol of Fitzi et al. requires at least 60 rounds in expectation!

In light of the above, when discussing the round complexity of protocols that assume a broadcast channel we keep track of both the number of rounds as well as the number of rounds in which broadcast is used. (In a given round when broadcast is used, each party may use the broadcast channel but a rushing adversary is still assumed. Existing broadcast protocols can be modified so that the round complexity is unchanged even if many parties broadcast in parallel.) We say a protocol has round complexity (r, r') if it uses r rounds in total, and $r' \leq r$ of these rounds invoke broadcast. Recall that the round complexity of VSS refers to the sharing phase only since most known protocols, as well as the protocols described in this chapter, utilize only a single round of point-to-point communication in the reconstruction phase. (Exceptions include [36, 76, 69].)

Our results and techniques. Gennaro et al. [43] show that three rounds are necessary for perfect VSS, even assuming a broadcast channel. We also observe that it is impossible to construct a *strict* constant-round protocol for VSS without using a broadcast channel at all: VSS implies broadcast using one additional round (the message to be broadcast can be treated as the input for VSS), and results of Fischer and Lynch [33] rule out strict constant-round protocols for broadcast. Prior work [73, 36, 63, 68] shows that optimal round complexity as well as optimal use of the broadcast channel could each be obtained *individually* for VSS, but it was unknown whether they could be obtained *simultaneously*. Here, we resolve this question and show a $(3, 1)$-round VSS protocol that is optimal in both measures. As a consequence, we obtain a VSS protocol with the best known round complexity in point-to-point networks. Our work also leads to an improvement in the round complexity of the most round-efficient broadcast protocols known [62].

A nice feature of our VSS protocol is that it also satisfies a certain "2-level sharing" property that is not achieved by the 3-round protocol from [36]. Roughly speaking, this means that the following conditions hold at the end of the sharing phase when the dealer's (effective) input is s:

1. There exists a polynomial $f(x)$ of degree at most t such that $f(0) = s$ and each honest party P_i holds the value $f(i)$. Said differently, at the end of the sharing phase each honest party P_i holds a value s_i with the property that these $\{s_i\}$ all lie on a degree-t polynomial f (whose constant term is s).

2. For each party P_i, there exists a polynomial $f_i(x)$ of degree at most t such that $f_i(0) = f(i)$ and each honest party P_j holds the value $f_i(j)$.

VSS protocols with these properties (the first one in particular) constitute a useful building block for protocols for general secure multi-party computation (see, e.g., [63, 68]).

Our protocol is efficient, in that the computation and communication are polynomial in n. The communication complexity of our protocol is $O(n^2t)$ field elements, which matches the communication complexity of [36] but is worse than that of [43].

We now summarize the basic techniques used to prove our main result. As in [36], we begin by constructing a protocol for *weak* verifiable secret sharing (WSS) [84]. (In WSS, informally, if the dealer is dishonest then, in the reconstruction phase, each honest party recovers either the dealer's input or a special failure symbol.) Fitzi et al. show a $(3, 2)$-round WSS protocol that essentially consists of the first three rounds of the 4-round VSS protocol from [43]. On a high level, their protocol works as follows: In the first round, the dealer distributes the shares of the secret using a random bivariate polynomial; in parallel, each pair of parties (P_i, P_j) exchanges a random pad $r_{i,j}$. In the second round, P_i and P_j check for an inconsistency between their shares by broadcasting their common shares masked with the random pad. In the third round, if there is a disagreement between P_i and P_j in round 2 (note that all parties agree whether there is disagreement since broadcast is used in round 2), then the dealer, P_i, and P_j all broadcast the share in question. This allows the rest of the parties to determine whether the dealer "agrees" with P_i or with P_j.

A $(5, 1)$-round WSS protocol is implicitly given in [63].[1] There, rather than using the "random pad" technique, a different method is used to detect disagreement between P_i and P_j. While this saves one round of broadcast, it requires additional rounds of interaction.

To construct a $(3, 1)$-round WSS protocol, we modify the $(3, 2)$-round WSS protocol from [36] by using the random pad idea with the following twist: in the second round of the protocol, P_i and P_j check if there is any inconsistency between their shares by exchanging their common shares over a *point-to-point* link; they also send the random pad $r_{i,j}$ to the dealer. In the third round of the protocol, if there is a disagreement between P_i and P_j, then P_i and P_j each broadcast the shares they hold; otherwise, they broadcast the value of their common share masked with the random pad. The dealer will broadcast the corresponding share masked with the random pad (or the share itself if the random pads it received from P_i and P_j are different). Notice that secrecy of the share

[1] That work shows a 6-round VSS protocol that uses broadcast in the final two rounds. The first five rounds of that protocol suffice for WSS.

is preserved if P_i, P_j, and the dealer are all honest. On the other hand, if the dealer is malicious and there is a disagreement between honest parties P_i and P_j, then the dealer can only "agree" with at most one of P_i and P_j in round 3, but not both of them.

The above is the high-level idea of our WSS protocol. Using the same techniques as in [36], we can then immediately obtain a $(3, 1)$-round VSS protocol. However, the VSS protocol constructed in this manner will not have the "2-level sharing" property; as a consequence, the resulting protocol cannot directly be plugged in to existing protocols for general secure multi-party computation.

To convert the VSS protocol into one with 2-level sharing we note that, by the end of the sharing phase, there is a set of honest parties (that we call a "core set") who already *do* have the required 2-level shares; thus, we only need to provide honest parties outside the core set with their required shares. We achieve this, as in [25], by having the dealer use a *symmetric* bivariate polynomial to share its input, and then modifying the protocol so that honest parties who are not in the core set can still generate appropriate shares by interpolating the shares of the parties in the core set. Of course, this process needs to be carefully designed so that no additional information is leaked to the adversary. We defer the details of this to a later section.

Other related work. Gennaro et al. [43] initiated a study of the exact round complexity of VSS. For $t < n/3$, they show an efficient (i.e., polynomial-time) $(4, 3)$-round protocol, and an inefficient $(3, 2)$-round protocol. (Recall that the round complexity of VSS is defined as the number of rounds in the sharing phase; unless otherwise stated, all protocols mentioned use only one round, without broadcast, in the reconstruction phase.) They also show that three rounds are necessary for VSS when $t < n/3$. For $t < n/4$, they show that two rounds are necessary and sufficient for efficient VSS. Settling the question of the absolute round complexity of efficient VSS for $t < n/3$, Fitzi et al. [36] show an efficient $(3, 2)$-round VSS protocol. The reconstruction phase of their protocol, however, requires one round of broadcast.

As discussed extensively already, although the protocol by Fitzi et al. is optimal in terms of the total number of rounds, it is not optimal in terms of its usage of the broadcast channel. VSS protocols for $t < n/3$ using one round of broadcast are known, but these protocols are not optimal in terms of their overall round complexity. Micali and Rabin [73] give a $(16, 1)$-round VSS protocol, and recent work [63, 68] improved this to give a $(7, 1)$-round protocol.

All the works referenced above, as well as the results in this chapter, focus on perfect VSS. A natural relaxation is to consider *statistical* VSS where privacy and/or correctness may fail to hold with negligible probability. Surprisingly, work subsequent to our own [76] shows that the lower bound of Gennaro et al. no longer holds in this setting, and that there exists a protocol for statistical VSS tolerating $t < n/3$ corruptions that uses only two rounds in the sharing phase. Interestingly, the reconstruction phase of their statistical VSS protocol requires two rounds, and so the total round complexity (of the sharing and reconstruction phases combined) matches the total round complexity of our protocol. It remains open whether the total round complexity can be improved for statistical VSS.

5.1 Weak Verifiable Secret Sharing

We show a $(3, 1)$-round WSS protocol tolerating $t < n/3$ malicious parties.

5.1.1 The Protocol

Sharing phase. The sharing phase consists of three rounds, with broadcast used in the last round.

Round 1: The dealer holds s. The following steps are carried out in parallel:

- The dealer chooses a random bivariate polynomial $F(x, y)$ of degree at most t in each variable such that $F(0, 0) = s$. The dealer then sends to each party P_i the polynomials $f_i(x) := F(x, i)$ and $g_i(y) := F(i, y)$.
- Each party P_i picks a random pad $r_{i,j} \in \mathbb{F}$ for all $j \neq i$, and sends $r_{i,j}$ to both P_j and the dealer D.

Round 2: Each player P_i does the following:

- For all $j \neq i$, send $a_{i,j} := f_i(j)$ and $b_{i,j} := g_i(j)$ to P_j.
- Let $r'_{j,i}$ be the random pad that P_i received from P_j in the previous round. For all $j \neq i$, send $r'_{j,i}$ to D.

Round 3: Each player P_i does the following:

- Let $a'_{j,i}, b'_{j,i}$ be the values P_i received from P_j in the previous round.
- For all $j \neq i$, if $b'_{j,i} \neq f_i(j)$ then P_i broadcasts (j: "disagree-f", $f_i(j)$, $r_{i,j}$); otherwise, P_i broadcasts (j: "agree-f", $f_i(j) + r_{i,j}$).
- For all $j \neq i$, if $a'_{j,i} \neq g_i(j)$ then P_i broadcasts (j: "disagree-g", $g_i(j)$, $r'_{j,i}$); otherwise, P_i broadcasts (j: "agree-g", $g_i(j) + r'_{j,i}$).

In parallel with the above, the dealer D does the following for all ordered pairs (i, j):

- Let $r^{(1)}_{i,j}$ be the appropriate random pad sent by P_i to D in round 1, and let $r^{(2)}_{i,j}$ be the appropriate random pad sent by P_j to D in round 2.
- If $r^{(1)}_{i,j} \neq r^{(2)}_{i,j}$, then D broadcasts ((i, j): "not equal", $F(j, i)$). Otherwise, D broadcasts ((i, j): "equal", $F(j, i) + r^{(1)}_{i,j}$).

Local computation. An ordered pair of parties (P_i, P_j) is *conflicting* if, in round 3, party P_i broadcasts (j : "disagree-f", $f_i(j)$, $r_{i,j}$); party P_j broadcasts (i: "disagree-g", $g_j(i)$, $r'_{i,j}$); and $r_{i,j} = r'_{i,j}$. For a pair of conflicting parties (P_i, P_j), we say P_i (resp., P_j) is *unhappy* if one of the following conditions hold:

- The dealer broadcasts ((i, j): "not equal", $d_{i,j}$) and $d_{i,j} \neq f_i(j)$ (resp., $d_{i,j} \neq g_j(i)$).
- The dealer broadcasts ((i, j): "equal", $d_{i,j}$) and $d_{i,j} \neq f_i(j) + r_{i,j}$ (resp., $d_{i,j} \neq g_j(i) + r'_{i,j}$).

A player is *happy* if it is not unhappy. Note that all parties agree on which players are happy and which are not. If there are more than t unhappy parties, the dealer is disqualified and a default value is shared.

Reconstruction phase. The reconstruction phase is similar to the one in [36], except that we do not use broadcast.

1. Every happy party P_i sends the polynomials $f_i(x)$ and $g_i(y)$ to all other parties.
2. Let f^i_j, g^i_j denote the polynomials that P_j sent to P_i in the previous step. P_i then constructs a *consistency graph* G_i whose vertices correspond to the happy parties:
   - Initially, there is an edge between P_j and P_k in G_i if and only if $f^i_j(k) = g^i_k(j)$ and $g^i_j(k) = f^i_k(j)$. (Note that we allow also the case $j = k$ here.)

- If there exists a vertex in G_i whose degree is less than $n-t$ (including self-loops), then that vertex is removed from G_i. This is repeated until no more vertices can be removed.

Let Core_i denote the parties whose corresponding vertices remain in G_i.

3. If $|\mathsf{Core}_i| < n-t$, then P_i outputs $\perp$. Otherwise, P_i reconstructs the polynomial $F'(x,y)$ defined by any $t+1$ parties in Core_i, and outputs $s' := F'(0,0)$.

We remark that, since we do not use broadcast in the reconstruction phase, it is possible that $\mathsf{Core}_i, \mathsf{Core}_j$ are different for different honest parties P_i, P_j.

5.1.2 Proofs

Lemma 29 *If the dealer is not corrupted by the end of the sharing phase, then privacy is preserved.*

Proof Let $\mathcal{C}$ denote the set of parties corrupted by the end of the sharing phase. We show that if the dealer remains uncorrupted, then the information the adversary has about the dealer's input at the end of the sharing phase consists of the polynomials $\{f_i(x), g_i(y)\}_{P_i \in \mathcal{C}}$. Since $F(x,y)$ is a random bivariate polynomial of degree at most t and $|\mathcal{C}| \leq t$, a standard argument implies that the view of the adversary is independent of the dealer's input s.

It is immediate that the adversary learns nothing additional about s in rounds 1 or 2. As for the values broadcast in round 3, consider any ordered pair (P_i, P_j) of parties who remain honest throughout the sharing phase. Since the dealer is honest, we have $f_i(j) = g_j(i) = F(j,i)$ and, since P_i, P_j are honest, we have $r_{i,j} = r'_{i,j}$. Thus, in round 3 parties P_i, P_j, and the dealer all broadcast the same "blinded" value $F(j,i) + r_{i,j}$. Since $r_{i,j}$ is chosen uniformly at random, the parties in $\mathcal{C}$ do not learn anything about the value of $F(j,i)$. ■

Lemma 30 *If the dealer is not corrupted by the end of the sharing phase, then correctness holds.*

Proof If the dealer remains honest then no honest party will be unhappy. It follows that the dealer is not disqualified at the end of sharing phase.

Let P_i be honest. In the reconstruction phase, Core_i contains all the honest parties and so $|\mathsf{Core}_i| \geq n-t$. We claim that for any $P_j \in \mathsf{Core}_i$, it holds that $f_j^i(x) = F(x,j)$ and $g_j^i(y) = F(j,y)$, where F is the dealer's polynomial. When P_j is honest this is immediate. When P_j is malicious, the fact that $P_j \in \mathsf{Core}_i$ means that $f_j^i(k) = g_k^i(j) = F(k,j)$ for at least $n-2t \geq t+1$ honest parties P_k. Since $f_j^i(x)$ has degree at most t, it follows that $f_j^i(x) = F(x,j)$. A similar argument shows that $g_j^i(y) = F(j,y)$. Therefore, the polynomial $F'(x,y)$ reconstructed by P_i is equal to $F(x,y)$, and P_i outputs $s = F(0,0)$. ■

Lemma 31 *Weak commitment holds.*

Proof The case of an honest dealer follows from the proof of correctness, so we consider the case of a malicious dealer. If there are more than t unhappy parties, the dealer is disqualified and weak commitment trivially holds; so, assume there are at most t unhappy parties. Then there are at least $n-2t \geq t+1$ honest parties who are happy. Let $\mathcal{H}$ denote the first $t+1$ such parties. The polynomials f_i sent by the dealer to the parties in $\mathcal{H}$ define a bivariate polynomial $\hat{F}(x,y)$ in the natural way: namely, let $\hat{F}$ be such that $\hat{F}(x,i) = f_i(x)$ for each $P_i \in \mathcal{H}$. Because parties in $\mathcal{H}$ are happy, it holds also that $\hat{F}(i,y) = g_i(y)$ for all $P_i \in \mathcal{H}$. Set $s' := \hat{F}(0,0)$. We show that every honest party outputs either $\perp$ or s' in the reconstruction phase.

Consider an honest party P_i in the reconstruction phase. If $|\mathsf{Core}_i| < n-t$ then P_i outputs $\perp$ and we are done. Say $|\mathsf{Core}_i| \geq n-t$. We claim that for each $P_j \in \mathsf{Core}_i$, it holds that $f_j^i(x) = \hat{F}(x,j)$ and $g_j^i(y) = \hat{F}(j,y)$. When P_j is honest, the fact that P_j is happy (which is true since $P_j \in \mathsf{Core}_i$) means that $f_j^i(k) = f_j(k) = g_k(j) = \hat{F}(k,j)$ for all $t+1$ parties $P_k \in \mathcal{H}$. Since f_j^i is a polynomial of degree at most t, this implies that $f_j^i(x) = \hat{F}(x,j)$. A similar argument shows that $g_j^i(y) = \hat{F}(j,y)$. When $P_j \in \mathsf{Core}_i$ is malicious, we have that $f_j^i(k) = g_k^i(j) = \hat{F}(k,j)$ for at least $n-2t \geq t+1$ honest parties $P_k \in \mathsf{Core}_i$. Again, since $f_j^i(x)$ has degree at most t it follows that $f_j^i(x) = \hat{F}(x,j)$, and a similar argument shows that $g_j^i(y) = \hat{F}(j,y)$. Therefore, the polynomial reconstructed by P_i is equal to $\hat{F}(x,y)$, and P_i outputs $s' = \hat{F}(0,0)$. ∎

As the proof of the above lemma indicates, our WSS protocol also satisfies a weak variant of 2-level sharing that we state for future reference:

Lemma 32 *Say the dealer is not disqualified in an execution of the WSS protocol, and let $\mathcal{H}$ denote the set of all honest parties who are happy. Then there is a bivariate polynomial $\hat{F}$ of degree at most t in each variable such that, at the end of the sharing phase, the polynomials f_i, g_i held by each $P_i \in \mathcal{H}$ satisfy $f_i(x) = \hat{F}(x,i)$ and $g_i(y) = \hat{F}(i,y)$.*

As a consequence, each $P_i \in \mathcal{H}$ can compute s_i and $\{s_{i,j}\}_{j\in\{1,\ldots,n\}}$ such that:

1. *There is a polynomial $p(x)$ of degree at most t with $s_i = p(i)$, and furthermore all honest parties output either $s' = p(0)$ or $\perp$ in the reconstruction phase.*

2. *For each $j \in \{1,\ldots,n\}$, there exists a polynomial $p_j(x)$ of degree at most t such that (1) $p_j(0) = p(j)$ and (2) $s_{i,j} = p_j(i)$.*

Proof When the dealer is honest take $\hat{F}$ to be the dealer's polynomial. When the dealer is dishonest, let $\hat{F}$ be the bivariate polynomial defined in the proof of the preceding lemma. Set $p(x) \stackrel{\text{def}}{=} \hat{F}(0,x)$ and $p_j(x) \stackrel{\text{def}}{=} \hat{F}(x,j)$. In what follows we assume a dishonest dealer, but it is immediate that everything (trivially) holds also if the dealer is honest.

The proof of the preceding lemma shows that, at the end of the sharing phase, each $P_i \in \mathcal{H}$ holds polynomials f_i, g_i with $f_i(x) = \hat{F}(x,i)$ and $g_i(y) = \hat{F}(i,y)$, and such that all honest parties output either $s' = \hat{F}(0,0)$ or $\perp$ in the reconstruction phase. Then each $P_i \in \mathcal{H}$ can compute $s_i := f_i(0) = \hat{F}(0,i) = p(i)$ and $s_{i,j} := g_i(j) = \hat{F}(i,j) = p_j(i)$. Furthermore, $s' = p(0)$. Finally, $p_j(0) = \hat{F}(0,j) = p(j)$ for all $j \in \{1,\ldots,n\}$. Thus, all the stated requirements hold. ∎

5.2 Verifiable Secret Sharing

Before we describe our VSS protocol with 2-level sharing, we review the ideas used in [36] to transform their WSS protocol into a VSS protocol (that does not have 2-level sharing). At a high level, the sharing phase of the VSS protocol is more-or-less the same as the sharing phase of the underlying WSS protocol; the difference is that now, in the reconstruction phase, each party reveals the random pads they used in the sharing phase. A problem that arises is to ensure that a malicious party P_i reveals the "correct" random pads. This is enforced by having each player act as a dealer in its own execution of WSS, and "binding" the random pads of each party to this execution of WSS. In more detail: in parallel with the sharing phase of the larger VSS protocol, each party P_i also acts as a dealer and shares a random secret using the WSS protocol. Let $F_i^{pad}(x,y)$ be the corresponding bivariate polynomial chosen by P_i. Then P_i will use $r_{i,j} := F_i^{pad}(0,j)$ as the appropriate "random pad" in the larger VSS protocol. (The pads $\{r_{i,j}\}$ used by any honest party P_i are thus no longer independent, but secrecy is still preserved since they lie on a random degree-t polynomial.) These random pads are then revealed in the reconstruction phase by using the reconstruction phase of the underlying WSS protocol.

We can use the ideas outlined in the previous paragraph to obtain a $(3, 1)$-round VSS protocol, but the resulting protocol will not have 2-level sharing. Yet all is not lost. As observed already in Lemma 32, by the end of the sharing phase the honest parties who are happy *will* have the required 2-level shares. To achieve our desired result we must therefore only enable any *unhappy* honest party to construct *its* 2-level shares.

At a high level, we do this as follows: Suppose $\hat{F}(x, y)$ is the dealer's bivariate polynomial, defined by the end of the sharing phase of the VSS protocol, and let P_i be an honest party who is unhappy. We need to show how P_i constructs the polynomials $\hat{F}(x, i)$ and $\hat{F}(i, y)$ (which it will use to generate its 2-level shares exactly as in the proof of Lemma 32). Let P_j be a party such that:

- P_j is happy (in the larger VSS protocol);
- P_j was not disqualified as a dealer it its own execution of WSS; and
- P_i is happy in P_j's execution of WSS.

From the proof of Lemma 32, we know there is a bivariate polynomial $\hat{F}_j^{pad}(x, y)$ for which P_i holds the univariate polynomial $\hat{F}_j^{pad}(x, i)$. Furthermore, P_j has effectively broadcasted the polynomial $B_j(x) \overset{\text{def}}{=} \hat{F}(x, j) + \hat{F}_j^{pad}(0, x)$ in round 3, since it has broadcasted $\hat{F}(k, j) + \hat{F}_j^{pad}(0, k)$ for all k. Thus, party P_i can compute

$$\hat{F}(i, j) := B_j(i) - \hat{F}_j^{pad}(0, i) = \hat{F}(i, j)$$

for any party P_j satisfying the above conditions. If there are $t + 1$ parties satisfying the above conditions, then P_i can reconstruct the polynomial $\hat{F}(i, y)$.

Unfortunately, it is not clear how to extend the above approach to enable P_i to also reconstruct the polynomial $\hat{F}(x, i)$ in the case when $\hat{F}$ is an *arbitrary* bivariate polynomial. For this reason, we have the dealer use a *symmetric*[2] bivariate polynomial. Then $\hat{F}(x, i) = \hat{F}(i, x)$ and we are done.

5.2.1 The Protocol

We show a $(3, 1)$-round VSS protocol with 2-level sharing that tolerates $t < n/3$ malicious parties.

Sharing phase. The sharing phase consists of three rounds, with broadcast used in the last round.

Round 1: The dealer holds s. The following steps are carried out in parallel:

- The dealer chooses a random *symmetric* bivariate polynomial $F(x, y)$ of degree t in each variable such that $F(0, 0) = s$. Then D sends to each party P_i the polynomial $f_i(x) := F(x, i)$. Note that $F(x, i) = F(i, x)$ since F is symmetric.
- Each party P_i picks a random value $\hat{s}_i$ and executes the first round of the WSS protocol described in the previous section, acting as a dealer to share the "input" $\hat{s}_i$. We refer to this instance of the WSS protocol as WSS$_i$.
- Let $F_i^{pad}(x, y)$ denote the bivariate polynomial used by P_i in WSS$_i$ (i.e., $F_i^{pad}(0, 0) = \hat{s}_i$). Party P_i sends the polynomial $r_i(y) := F_i^{pad}(0, y)$ to the dealer D.

[2] A polynomial F is symmetric if, for all ℓ, m, the coefficient of the term $x^\ell y^m$ is equal to the coefficient of the term $x^m y^\ell$. If F is symmetric then $F(i, j) = F(j, i)$ for all i, j.

Round 2: Each party P_i does the following:

- Run round 2 of WSS$_j$, for all j.
- For all $j \neq i$, send $a_{i,j} := f_i(j)$ to P_j.
- For all $j \neq i$, let $f_{j,i}^{pad}(x)$ be the polynomial that P_j sent to P_i in round 1 of WSS$_j$. (If P_j is honest then $f_{j,i}^{pad}(x) = F_j^{pad}(x, i)$.) Party P_i sends $r'_{j,i} := f_{j,i}^{pad}(0)$ to D.

Round 3: Each party P_i does the following:

- Run round 3 of WSS$_j$, for all j.
- For all $j \neq i$, let $a'_{j,i}$ be the value P_i received from P_j in the previous round. If $a'_{j,i} \neq f_i(j)$, then P_i broadcasts (j: "disagree", $f_i(j)$, $F_i^{pad}(0, j)$). Otherwise, P_i broadcasts (j: "agree", $f_i(j) + F_i^{pad}(0, j)$).

In parallel with the above, the dealer D does the following for all ordered pairs (i, j):

- Let $r_i^{(1)}$ be the polynomial sent by P_i to D in round 1, and let $r_{i,j}^{(2)}$ be the appropriate random pad sent by P_j to D in round 2.
- If $r_i^{(1)}(j) \neq r_{i,j}^{(2)}$, then D broadcasts ((i, j): "not equal", $F(j, i)$). Otherwise, D broadcasts ((i, j): "equal", $F(j, i) + r_i^{(1)}(j)$).

Local computation. Each party locally carries out the following steps:

1. An ordered pair of parties (P_i, P_j) is *conflicting* if, in round 3, party P_i broadcasts (j: "disagree", $f_i(j)$, $F_i^{pad}(0, j)$); party P_j broadcasts (i: "disagree", $f_j(i)$, $f_{i,j}^{pad}(0)$); and it holds that $F_i^{pad}(0, j) = f_{i,j}^{pad}(0)$. For a pair of conflicting parties (P_i, P_j), we say that P_i (resp., P_j) is *unhappy* if one of the following conditions hold:

   (.1) D broadcasts ((i, j): "not equal", $d_{i,j}$) and $d_{i,j} \neq f_i(j)$ (resp., $d_{i,j} \neq f_j(i)$).

   (.2) D broadcasts ((i, j): "equal", $d_{i,j}$) and $d_{i,j} \neq f_i(j) + F_i^{pad}(0, j)$ (resp., $d_{i,j} \neq f_j(i) + f_{i,j}^{pad}(0)$).

   A party is *happy* if it is not unhappy.

   Let Core denote the set of parties who are happy with respect to the definition above. For every P_i who was not disqualified as the dealer in WSS$_i$, let Core_i denote the set of parties who are happy with respect to WSS$_i$. (If P_i was disqualified in WSS$_i$, then set $\mathsf{Core}_i := \emptyset$.)

   Note that all parties have the same view regarding Core and the $\{\mathsf{Core}_i\}$.

2. For all i, j, remove P_j from Core_i if either of the following hold for the ordered pair (i, j) in round 3:

   - P_i broadcasts (j: "agree", y) and P_j did not broadcast (i: "agree", y).
   - P_i broadcasts (j : "disagree", $\star, w$) and P_j broadcasts either (i : "agree", $\star$) or (i : "disagree", $\star, w'$) with $w' \neq w$. (Here, $\star$ denotes an arbitrary value.)

3. Remove P_i from Core if $|\mathsf{Core} \cap \mathsf{Core}_i| < n - t$. (Thus, in particular, if P_i was disqualified in WSS$_i$ then $P_i \notin \mathsf{Core}$.)

   Note that all parties still have the same view of Core and the $\{\mathsf{Core}_i\}$.

4. If $|\mathsf{Core}| < n-t$, then the dealer is disqualified and a default value (along with default 2-level shares) are shared.

5. Each party P_i computes a polynomial $\hat{f}_i(x)$ of degree at most t:

   (.1) If $P_i \in \mathsf{Core}$, then $\hat{f}_i(x)$ is the polynomial that P_i received from the dealer in round 1.

   (.2) If $P_i \notin \mathsf{Core}$, then P_i computes $\hat{f}_i(x)$ in the following way:

   i. P_i first defines a set Core'_i as follows: A party P_j is in Core'_i if and only if all the following conditions hold:

      - $P_j \in \mathsf{Core}$ and $P_i \in \mathsf{Core}_j$.
      - Define $p_{j,k}$, for $k \in \{1, \ldots, n\}$, as follows: if in round 3 party P_j broadcasted (k : "agree", $y_{j,k}$), then set $p_{j,k} := y_{j,k}$. If P_j broadcasted (k: "disagree", $w_{j,k}, z_{j,k}$), then set $p_{j,k} := w_{j,k} + z_{j,k}$.

        We require that the $\{p_{j,k}\}$ lie on a polynomial $B_j(x)$ of degree at most t; i.e., such that $B_j(k) = p_{j,k}$ for all k. (If not, then P_j is not included in Core'_i.)

      Our proofs will show that $|\mathsf{Core}'_i| \geq t+1$ if the dealer is not disqualified.

   ii. For each $P_j \in \mathsf{Core}'_i$, set $p_j := p_{j,i} - f^{pad}_{j,i}(0)$. Let $\hat{f}_i$ be the polynomial of degree at most t such that $\hat{f}_i(j) = p_j$ for every $P_j \in \mathsf{Core}'_i$. (It will follow from our proof that such an $\hat{f}_i$ exists.)

6. Finally, P_i outputs $s_i := \hat{f}_i(0)$ and $s_{i,j} := \hat{f}_i(j)$ for all $j \in \{1, \ldots, n\}$.

Reconstruction phase. Each party P_i sends s_i to all other parties. Let $s'_{j,i}$ be the value that P_j sends to P_i. Using Reed-Solomon decoding, P_i computes a polynomial $f(x)$ of degree at most t such that $f(j) = s'_{j,i}$ for at least $2t+1$ values of j. The final output of P_i is $f(0)$.

5.2.2 Proofs

We prove that the protocol given in the previous section is a VSS protocol with 2-level sharing that tolerates $t < n/3$ malicious parties.

Lemma 33 *If the dealer is not corrupted by the end of the sharing phase, privacy is preserved.*

Proof Let $\mathcal{C}$ denote the set of parties corrupted by the end of the sharing phase. We show that if the dealer remains uncorrupted, then the view of the adversary can be simulated given the polynomials $\{f_i(x)\}_{P_i \in \mathcal{C}}$. Since $F(x,y)$ is a random symmetric bivariate polynomial of degree at most t and $|\mathcal{C}| \leq t$, a standard argument (see, e.g., [22]) implies that the view of the adversary is independent of the dealer's input s.

It is immediate that the adversary learns nothing additional about s in round 2. As for the values broadcast in round 3, consider an ordered pair (P_i, P_j) of parties who remain honest throughout the sharing phase. Since the dealer is honest, we have $f_i(j) = F(j,i) = F(i,j) = f_j(i)$ and, since P_i, P_j are honest, $r_i(j) = r'_{i,j}$. Thus, in round 3, parties P_i, P_j, and the dealer all broadcast the same "blinded" value $f_i(j) + F^{pad}_i(0,j)$. Since $F^{pad}_i(0,y)$ is a random polynomial of degree at most t this does not leak any information about the $\{f_i(x)\}_{P_i \notin \mathcal{C}}$ that the adversary does not already know. ■

Lemma 34 *If the dealer is not corrupted by the end of the sharing phase, then correctness and commitment with 2-level sharing hold.*

Proof If the dealer is honest, then no honest party is unhappy. Also, all honest parties are in Core_i for any honest player P_i. Since there are at least $n-t$ honest parties, no honest party is removed from Core. It follows that the dealer is not disqualified.

Since all honest parties are in Core, each honest party P_i sets $\hat{f}_i(x) := f_i(x) = F(x,i)$. Defining $p(x) \overset{\text{def}}{=} F(0,x)$ and $p_j(x) \overset{\text{def}}{=} F(j,x)$, it is straightforward to verify that the properties of commitment with 2-level sharing hold:

- Each honest party P_i outputs $s_i := \hat{f}_i(0) = F(0,i) = p(i)$.
- For all j, it holds that $p_j(0) = F(j,0) = F(0,j) = p(j)$.
- For each honest party P_i and all $j \in \{1,\ldots,n\}$, we have
$$s_{i,j} = \hat{f}_i(j) = F(j,i) = p_j(i).$$

In the reconstruction phase, $s'_{j,i} = s_j = p(j)$ for any honest party P_j. Thus, each honest party P_i receives at most t values $s'_{j,i}$ that do not lie on the polynomial $p(x)$. It follows that P_i outputs $s = p(0) = F(0,0)$, the dealer's input. This completes the proof. ■

We now move on to show that commitment with 2-level sharing holds even when the dealer is malicious. The case of a disqualified dealer is obvious, so we focus on the case of a malicious dealer who is not disqualified. We begin by proving three claims:

Claim 35 *If the dealer is not disqualified, then for any honest P_i it holds that $|\mathsf{Core}'_i| \geq t+1$.*

Proof If the dealer was not disqualified, then Core contains at least $n-2t \geq t+1$ honest parties. We show that any honest $P_j \in \mathsf{Core}$ is also in Core'_i, proving the claim.

Since P_i and P_j are both honest, $P_i \in \mathsf{Core}_j$. Set $B(x) \overset{\text{def}}{=} f_j(x) + F_j^{pad}(0,x)$. This is a polynomial of degree at most t, and the $p_{j,k}$ computed by P_i all lie on $B_j(x)$. We conclude that $P_j \in \mathsf{Core}'_i$. ■

Claim 36 *If the dealer is not disqualified in the sharing phase, there is a bivariate symmetric polynomial $\hat{F}(x,y)$ of degree at most t in each variable that is consistent with the polynomials $\hat{f}_i$ computed by every honest party in Core; i.e., for every honest $P_i \in \mathsf{Core}$ it holds that $\hat{f}_i(x) = \hat{F}(x,i)$.*

Proof If the dealer is not disqualified, then there are at least $n-t$ parties in Core and at least $n-2t \geq t+1$ of them are honest. Let $\mathcal{H}$ denote the first $t+1$ such parties. The polynomials f_i sent by the dealer to the parties in $\mathcal{H}$ define a bivariate polynomial $\hat{F}(x,y)$ in the natural way: namely, let $\hat{F}$ be such that $\hat{F}(x,i) = f_i(x)$ for each $P_i \in \mathcal{H}$. We show that $\hat{F}$ satisfies the requirements of the claim.

By definition of $\hat{F}$, we have $\hat{f}_i(x) = f_i(x) = \hat{F}(x,i)$ for any $P_i \in \mathcal{H}$. Next, observe that for every honest $P_i, P_j \in \mathsf{Core}$ it holds that $\hat{f}_i(j) = \hat{f}_j(i)$. Indeed, it must be the case that $f_i(j) = f_j(i)$ (or else one of P_i or P_j would be unhappy), and since $P_i, P_j \in \mathsf{Core}$ we have $\hat{f}_i(x) = f_i(x)$ and $\hat{f}_j(x) = f_j(x)$. Since $\mathcal{H} \subset \mathsf{Core}$, this implies that $\hat{F}$ is symmetric. It also implies that for *every* honest $P_i \in \mathsf{Core}$ (i.e., not just the $P_i \in \mathcal{H}$) we have $\hat{f}_i(x) = \hat{F}(i,x) = \hat{F}(x,i)$, proving the claim. ■

Claim 37 *Assume the dealer is not disqualified in the sharing phase, and let $\hat{F}$ be the polynomial guaranteed to exist by Claim 36. Then for any honest $P_i \notin \mathsf{Core}$, it holds that $\hat{f}_i(x) = \hat{F}(x,i)$.*

Proof Fix an honest $P_i \notin \mathsf{Core}$, and an arbitrary $P_j \in \mathsf{Core}'_i$. (Claim 35 shows that Core'_i is non-empty.) By definition, this means $P_j \in \mathsf{Core}$ and $P_i \in \mathsf{Core}_j$. So P_j was not disqualified as a dealer in WSS$_j$ and, by Lemma 32, there exists a bivariate polynomial $\hat{F}_j^{pad}$ of degree at most t in each variable such that $f_{j,k}^{pad}(x) = \hat{F}_j^{pad}(x,k)$ for all $P_k \in \mathsf{Core}_j$. (Recall that $f_{j,k}^{pad}$ denotes the polynomial that P_j sent to P_k in round 1 of WSS$_j$.)

Let $p_{j,k}$ be the values computed by P_i, and let $B_j(x)$ be a polynomial of degree at most t such that $B_j(k) = p_{j,k}$ for all k. Such a polynomial is guaranteed to exist because otherwise $P_j \notin \mathsf{Core}'_i$.

Since P_j remains in Core, we have $|\mathsf{Core} \cap \mathsf{Core}_j| \geq n - t$. This means that there are at least $n-2t \geq t+1$ honest parties that are in both Core and Core_j. Letting $\hat{F}$ be the symmetric polynomial guaranteed by the previous claim, we now show that for any honest $P_k \in \mathsf{Core} \cap \mathsf{Core}_j$ we have $B_j(k) = \hat{F}(k,j) + \hat{F}_j^{pad}(0,k)$. There are two cases to consider:

– If in round 3 party P_j broadcasted (k : "agree", $y_{j,k}$), then $p_{j,k} := y_{j,k}$. Since $P_k \in \mathsf{Core}_j$, this implies that party P_k must have broadcasted (j : "agree", $y_{k,j}$) with $y_{k,j} = y_{j,k}$ in that round (cf. step 2 of the local computation phase). Since P_k is honest we have

$$\begin{aligned} B_j(k) = p_{j,k} = y_{j,k} = y_{k,j} & \\ &= f_k(j) + f_{j,k}^{pad}(0) \\ &= \hat{F}(j,k) + f_{j,k}^{pad}(0) \quad \text{(using Claim 36 and } P_k \in \mathsf{Core}) \\ &= \hat{F}(j,k) + \hat{F}_j^{pad}(0,k) \quad \text{(since } P_k \in \mathsf{Core}_j) \\ &= \hat{F}(k,j) + \hat{F}_j^{pad}(0,k), \end{aligned}$$

using the fact that $\hat{F}$ is symmetric.

– If in round 3 party P_j broadcasted (k : "disagree", $w_{j,k}, z_{j,k}$) then, because $P_k \in \mathsf{Core}_j$, this implies that party P_k must have broadcasted (j : "disagree", $w_{k,j}, z_{k,j}$) with $z_{k,j} = z_{j,k}$. It must also be the case that $w_{k,j} = w_{j,k}$ or else one of P_j or P_k would be unhappy. It follows that

$$B_j(k) = p_{j,k} = w_{j,k} + z_{j,k} = w_{k,j} + z_{k,j},$$

and then an argument as before shows that $B_j(k) = \hat{F}(k,j) + \hat{F}_j^{pad}(0,k)$.

Summarizing, we have $B_j(k) = \hat{F}(k,j) + \hat{F}_j^{pad}(0,k)$ for at least $t+1$ values of k. Since $B_j(x)$ has degree at most t, this means $B_j(x) = \hat{F}(x,j) + \hat{F}_j^{pad}(0,x)$.

Party P_i next computes

$$\begin{aligned} p_j := p_{j,i} - f_{j,i}^{pad}(0) &= B_j(i) - \hat{F}_j^{pad}(0,i) \\ &= \hat{F}(i,j) + \hat{F}_j^{pad}(0,i) - \hat{F}_j^{pad}(0,i) \;=\; \hat{F}(i,j), \end{aligned}$$

using the fact that $P_i \in \mathsf{Core}_j$ in the first line. Since this is true for arbitrary $P_j \in \mathsf{Core}'_i$, we see that the polynomial $\hat{f}_i$ computed by P_i satisfies $\hat{f}_i(x) = \hat{F}(i,x) = \hat{F}(x,i)$. This completes the proof. ■

Lemma 38 *Even when the dealer is malicious, commitment with 2-level sharing holds.*

Proof By the preceding two claims, at the end of the sharing phase there exists a symmetric bivariate polynomial $\hat{F}(x,y)$ with degree at most t in each variable such that $\hat{f}_i(x) = \hat{F}(x,i)$ for any honest party P_i. Set $p(x) := \hat{F}(x,0)$ and $p_j(x) := \hat{F}(x,j)$. One can then verify that the properties of commitment with 2-level sharing hold:

- Each honest party P_i outputs $s_i \stackrel{\text{def}}{=} \hat{f}_i(0) = \hat{F}(0,i) = \hat{F}(i,0) = p(i)$.
- At the end of the reconstruction phase, each honest party P_i will output $s' = p(0)$.
- For all j, it holds that $p_j(0) = \hat{F}(0,j) = p(j)$.
- For each honest party P_i and all $j \in \{1,\ldots,n\}$, we have

$$s_{i,j} \stackrel{\text{def}}{=} \hat{f}_i(j) = \hat{F}(j,i) = \hat{F}(i,j) = p_j(i).$$

This completes the proof. ■

Chapter 6

Round Complexity of Statistical VSS with Honest Majority

In this chapter we study the round complexity of information-theoretic VSS in the presence of a honest majority. In this setting, we require the security requirements of VSS to hold even when the malicious parties have unbounded computational power. In the previous chapter, we saw *perfect* VSS, where all security requirements hold unconditionally (i.e., even against a computationally unbounded adversary) with zero error probability. However, it is known that perfect VSS is possible if and only if $t < n/3$ [5, 27]. On the other hand, *statistical* VSS (cf. Definition 5), where security requirements may be violated with negligible probability, is possible (assuming the existence of a broadcast channel) up to a threshold $t < n/2$ [83].

The round complexity of perfect VSS has been extensively studied. For the case of optimal threshold (i.e., $t < n/3$), Gennaro et al. [43] showed that 3 rounds[1] are necessary and sufficient for perfect VSS, and gave an efficient 4-round protocol for the task. The 3-round VSS protocol by Gennaro et al. requires communication exponential in the number of players, but Fitzi et al. [36] later demonstrated that an *efficient* 3-round protocol is possible. In Chapter 5, we showed that perfect VSS can be achieved with optimal round complexity and, at the same time, optimal use of the broadcast channel.

For the case of statistical VSS, the best known upper bound on the exact round complexity was obtained by Cramer et al. [23]. Their protocol required eleven rounds. The 3-round lower bound of Gennaro et al. was generally believed to apply also to the case of statistical VSS. Surprisingly, Patra et al. [76] showed that statistical VSS could be realized in *two* rounds for $t < n/3$. (The protocol of Patra et al. does not apply when $n/3 \leq t < n/2$.) On the other hand, the work of Patra et al. proves that 2-round statistical VSS is impossible for $t \geq n/3$, which obviously applies to setting of honest majority as well. Motivated by the discussion above, we aim to develop a better understanding of the round complexity of statistical VSS in the presence of an honest majority.

Our results and techniques. We consider a definition of statistical VSS that relaxes the *correctness/commitment* requirement, but not the *privacy* requirement (cf. Definiton 5). This is the definition that has been considered previously in the literature [83, 23]. In more detail, we relax the security requirements in the following way. For a given a statistical security parameter λ, our VSS protocol always achieves perfect privacy, and may fail to achieve correctness/commitment with probability at most $2^{-\Theta(\lambda)}$. Under this relaxation, we obtain the following results. First, we show a 3-round protocol for statistical VSS, thereby settling the exact round complexity of statistical

[1] Following the accepted convention, the round complexity of VSS refers to that of the sharing phase.

VSS with optimal threshold. Our construction is *inefficient*, in that the computation and communication are exponential in the number of parties. Second, we show an *efficient* 4-round protocol for statistical VSS with optimal threshold, i.e., with computation and communication polynomial in n. Our work also leads to an improvement in the exact round complexity of coin-tossing in the presence of an honest majority.

We now summarize the basic techniques used to prove our results. As in [84, 23], we build statistical VSS protocols using several instances of a primitive known as *information checking.* Loosely speaking, a protocol for information checking allows a *sender* to send a "signature" on a secret s to a *receiver*, such that the receiver can later produce a "proof" that convinces a *verifier* that it indeed received s from the sender. Recently, the work of Patra et al. [78, 77] improved and generalized previous protocols for information checking. More specifically, they construct a 3-round protocol that allows every party to simultaneously act as a verifier while guaranteeing agreement among honest parties' decision to accept or reject the receiver's proof. We employ their information checking protocol as a subroutine in all our constructions. Indeed, much of the efficiency gains we achieve may be attributed to the use of their round-efficient information checking protocol. We observe that prior statistical VSS protocols (e.g., [23]) would also enjoy significant improvements if they are modified to work with the information checking protocol of Patra et al. However, this modification alone does not suffice to yield even a five round protocol for statistical VSS.

We also propose and work with a slightly relaxed definition for information checking, and observe that Patra et al.'s construction satisfies this weaker definition without any further modification. More concretely, our definition of information checking differs from the traditional definition in the following way. Earlier works [23, 78, 77] defined a protocol for information checking as a three-phase protocol that required to satisfy a privacy property and three additional "correctness" properties. In contrast, we define a protocol for information checking (cf. Definition 8) as a two-phase protocol (with the two phases being a sharing phase, and a reconstruction phase) that satisfies three requirements, namely privacy, correctness, and a relaxed commitment property. Our reasons for pursuing this alternate definition of information checking are as follows. First, our definition is sufficient for our purposes, and further considerably simplifies the design of our VSS protocols and proofs. Next, we believe that our definition clearly highlights the similarities between information checking and verifiable secret sharing. In other words, it clearly shows how information checking is weaker than verifiable secret sharing. Furthermore, our definition of information checking allows us to immediately observe how a stronger version of information checking directly yields a statistical VSS protocol. (See Section 6.1.1.)

Our three round statistical VSS protocol may be best viewed as a stronger version of our information checking protocol. We explain this in detail. Patra et al. generalized the traditional information checking protocol to simultaneously accommodate multiple verifiers. We further extend this idea, and generalize their information checking protocol to simultaneously accommodate multiple receivers while preserving both the security guarantees and the round complexity of information checking. We do this in two steps. First, we define a primitive called *weak information checking for multiple receivers*, denoted WICP, which extends information checking to simultaneously accommodate up to t receivers (cf. Definition 9). The guarantees provided by WICP are weak, and in particular, correctness and commitment properties are guaranteed only when *all* receivers are honest. We give a 3-round protocol for WICP. Our next primitive which we call *information checking for multiple receivers*, denoted SICP, builds on, and significantly strengthens the properties provided by WICP. In SICP, both correctness and commitment are guaranteed as long as at least *one* of the receivers is honest (cf. Definition 10). We give a 3-round protocol for SICP. This protocol runs in time exponential in the number of parties, but it achieves strong guarantees that are sufficient to construct a simple protocol for statistical VSS. Indeed, our three round protocol for statistical VSS is obtained by running exponentially many instances of SICP in parallel, one for each possible set of receivers. We remark that in order to guarantee that the security properties fail to hold with probability at most $2^{-\Theta(\lambda)}$, we assume that the dealer's (possibly padded) secret s lies in a finite field $\mathbb{F}$ satisfying $\log|\mathbb{F}| = \kappa > \max(\lambda, 10n)$.[2]

As mentioned before, we also construct an *efficient* 4-round protocol for statistical VSS with optimal threshold. We employ extensively the information checking protocol of Patra et al. [78, 77] in our construction. In the following description, we say that party P_i sends a "signature" on a message s to another party P_j, if P_i acts as a sender with input s and P_j acts as the receiver in an execution of the sharing phase of an information checking protocol. Similarly, we say P_j reveals a "signature" on s, if P_j executes the reconstruction phase of the corresponding information checking protocol. On a high level, we use the following strategy to design our protocol. In the first round, the dealer distributes a signature on the shares of the secret to each party using a random symmetric bivariate polynomial; in parallel, each pair of parties (P_i, P_j) exchanges signatures on a random pad $r_{i,j}$. P_i also sends a signature on random pad $r_{i,j}$ to the dealer. In the second round, P_i and P_j check for an inconsistency between their shares by broadcasting their common shares masked with the random pad. The dealer will broadcast the corresponding share masked with the random pad. In the third round, if there is a disagreement between P_i and P_j in round 2 (note that all parties agree whether there is disagreement since broadcast is used in round 2), then the dealer, P_i, and P_j all broadcast the share in question. This allows the rest of the parties to determine whether the dealer "agrees" with P_i or with P_j. Notice that secrecy of the share is preserved if P_i, P_j, and the dealer are all honest. On the other hand, if the dealer is malicious and there is a disagreement between honest parties P_i and P_j, then the dealer can only "agree" with at most one of P_i and P_j in round 3, but not both of them.

Unfortunately, the above is not sufficient when the signatures are implemented by means of an information checking protocol. This is because information checking provides no security guarantees when both the sender and receiver are dishonest. Furthermore, the information checking protocol that we use allows a dishonest sender to change the message in the very last round of the protocol. To overcome such adversarial strategies, we require parties to begin the reconstruction phase of some instances of information checking protocols in order to expose a cheating dealer by revealing its signatures on inconsistent shares. Such a solution would work, but unfortunately requires 5 rounds since the reconstruction phase of our information checking protocol requires 2 rounds. Our main novelty is to allow the reconstruction phase of some information checking subprotocols to begin while concurrently executing the (last round of the) sharing phase of *all* information checking subprotocols. Of course, this process needs to be carefully designed so that no additional information is leaked to the adversary. We defer the details of this to Section 6.3, where we also prove that the above strategy is sufficient to yield an efficient 4-round statistical VSS protocol.

Other related work. Statistical VSS protocols with optimal resilience were first shown by Rabin and Ben-Or [84, 83], thereby showing information-theoretic MPC for $t < n/2$. A more efficient protocol for statistical VSS, and consequently for secure computation with an honest majority, was shown later by Cramer et al. [23]. Previous research has also investigated the design of statistical VSS protocols when a setup (for e.g., a PKI) is available. In such a setting, statistical VSS protocols are often simpler to design, and enjoy better round complexity [35, 63, 68]. For instance, given a setup, it is possible to collapse the 11-round protocol of [23] into a 4-round protocol [68, 63]. We

[2] Contrast this with the statistical VSS protocol of [23], where the size of the underlying finite field is simply 2^{λ}, and is independent of the number of parties.

stress that our protocols do not assume any setup.

As observed in Chapter 5, the number of broadcast rounds in VSS protocols contributes significantly to their round complexity in point-to-point networks. Recent work by Garay et al. [38] addresses this issue and shows a 9-round statistical VSS protocol with optimal resilience that uses broadcast only in three rounds. In contrast, we note that our protocols use broadcast in at least four rounds. Subsequent to our work, Backes et al. [1] investigated the round complexity of computational VSS (with no additional setup) when $t < n/2$, and show a 2-round protocol that they prove is round-optimal.

Notation. Let λ denote a statistical security parameter. Our VSS protocols will fail with probability at most $\varepsilon = 2^{-\Theta(\lambda)}$. In our protocols, we use a field $\mathbb{F}$ with $|\mathbb{F}| = 2^{\kappa}$ such that $\kappa > \max(\lambda, 10n)$. Without loss of generality, we assume that the dealer's input s lies in $\mathbb{F}$. Finally, we say that an event happens with *negligible probability* if the probability of that event occurring is at most $\varepsilon = 2^{-\Theta(\lambda)}$.

Organization. We define information checking, present the information checking protocol given by Patra et al. [78, 77], and prove its security in Section 6.1.1. Then, in Section 6.1.2, we define WICP and present our WICP protocol and its proof of security. Section 6.1.3 contains the definition of SICP and a protocol that meets this definition. In Section 6.2, we present our inefficient 3-round protocol for statistical VSS. Finally, Section 6.3 contains our efficient 4-round statistical VSS protocol.

6.1 Building Blocks

6.1.1 Information Checking

Our protocols build on *information checking protocol* (ICP), a notion first introduced by Rabin and Ben-Or [84]. The traditional definition of an ICP [84, 23] involves the dealer, a party who acts as the *intermediary*, and a party who acts as the *verifier*. In an initial phase, the dealer gives a secret value $s \in \mathbb{F}$ along with some auxiliary information to the intermediary party and some verification information (that reveals nothing about s) to the verifier. Later, using s and the auxiliary information provided by the dealer, the intermediary party can give s to the verifier along with a "proof" that s is indeed the value that it received from D.

The basic definition of ICP involves only a *single* verifier; Patra et al. [78, 77], extend this definition to allow every party in the network to act as a verifier. Enabling multiple verifiers considerably simplifies the description of our VSS protocols. Formally, an *information checking protocol* (ICP) is defined in the following way.

Definition 8 (Information Checking) *A two-phase protocol for parties $\mathcal{P} = \{P_1, \ldots, P_n\}$, where a dealer D holds initial input s and a receiver P_i, is an* information checking protocol with multiple verifiers *if the following conditions hold for any adversary controlling at most t parties:*

Privacy *If D and P_i are honest at the end of the first phase (the* sharing phase*), then at the end of this phase the joint view of the malicious parties is independent of the dealer's input s.*

Correctness *Each honest party P_k outputs a value s_k at the end of the second phase (the* reconstruction phase*). If the dealer D is honest then except with negligible probability, $s_k = s$ holds.*

Conditional Commitment *If P_i is honest, then at the end of the sharing phase, except with statistically negligible probability, the joint view of the honest parties defines a value s' such that each honest party will output s' at the end of the reconstruction phase.* $\diamondsuit$

We remark that the above definition is a more simplified (and slightly weaker) version of the traditional definition presented in prior work. Earlier work [84, 23, 78, 77] defines a protocol for ICP as a tuple of algorithms Distr, AuthVal, and RevealVal, and furthermore required *perfect* correctness when both D and P_i are honest. However, it is usually the case that Distr and AuthVal are executed successively, and typically in the "sharing phase" of a larger protocol that uses ICP as a subprotocol. Indeed, our definition groups the two algorithms together in the sharing phase of the ICP protocol. Similarly, RevealVal is typically executed in the "reconstruction phase" of a larger protocol that employs ICP as a subprotocol. Our definition simply dubs RevealVal as the reconstruction phase. Finally, note that since we are interested only in developing statistically secure VSS protocols, it is reasonable to consider a definition which requires correctness to hold only with high probability (even when both D and P_i are honest). We also note that information checking protocols are typically employed in constructing protocols for secure computation when $t < n/2$. It is well-known that perfect security is impossible in this setting (or, more generally when $t \geq n/3$ [5]).

We believe that our definition is natural (for our purposes) and furthermore, considerably simplifies the description of our VSS protocols. We will use ICP protocols as building blocks for designing new (and more complicated) primitives, and our simplified definition will make it easier to see the successive strengthenings our new primitives achieve. Indeed, our strategy for obtaining a 3-round VSS protocol is simply to strengthen the conditional commitment property of a 3-round ICP protocol. Suppose we design a protocol π with the exact same guarantees as an ICP protocol except the commitment property holds even for a dishonest P_i. Now consider the following candidate VSS protocol that employs π as a subprotocol. First, the dealer additively shares his secret input into n shares. Then, the dealer executes n instances of π, one with each party P_i using input the ith additive share. It is easy to see that all VSS properties (namely privacy, correctness, and commitment) follow directly from the corresponding properties of π, and furthermore the round complexity of the resulting VSS protocol is the same as the round complexity of π. (Although this may be viewed as a high level overview of our strategy, we stress that we are unable to exactly realize protocol π.)

The Protocol

Here we present a simple ICP protocol tolerating $t < n/2$ malicious parties. Our construction is based on the ICP protocol of Patra et al. [78, 77]. The protocol requires 3 rounds in an initial sharing phase, and an additional 2 rounds in the reconstruction phase.

Sharing phase. The sharing phase consists of three rounds.

Round 1: The dealer holds s. The following steps are carried out in parallel:

- The dealer chooses a random degree-t polynomial $F(x)$ such that $s = F(0)$, and sends $F(x)$ to P_i. In addition, the dealer sends a random degree-t polynomial R to P_i. Let F', R' denote the polynomials received by P_i.
- For each $P_k \in \mathcal{P}$, the dealer chooses *verification points* $x_k \in \mathbb{F} \setminus \{0\}$ at random, and sends $x_k, F(x_k)$, and $R(x_k)$ to P_k. Let x'_k, y'_k, and z'_k denote the values received by P_k.

Round 2: P_i chooses a random *multiplier* $d \in \mathbb{F}\backslash\{0\}$, computes $B = dF'(x)+R'(x)$, and broadcasts $(d, B(x))$.

Round 3: If it does not hold that $B(x) = dF(x) + R(x)$, then the dealer broadcasts $s^{(D)} = s$.

Local Computation. If dealer broadcasted $s^{(D)}$ in round 3 of the sharing phase, then each party sets $\mathsf{conflict}_{D,i}(s) = 1$, else they set $\mathsf{conflict}_{D,i}(s) = 0$.

Reconstruction phase. The reconstruction phase consists of two rounds. For the sake of clarity, we handle the following two cases separately. First, if D broadcasted $s^{(D)}$ in round 3 of the sharing phase, i.e., if $\mathsf{conflict}_{D,i}(s) = 1$ holds, each party $P_k \in \mathcal{P}$ sets $s_k = s^{(D)}$, and terminates the reconstruction protocol.

The remainder of the reconstruction phase deals with the second case where $\mathsf{conflict}_{D,i}(s) = 0$.

Round 1: P_i broadcasts $F'(x)$.

Round 2: Each party $P_k \in \mathcal{P}$ broadcasts "accept" if either of the following conditions hold.

- It holds that $y'_k = F'(x'_k)$.
- It holds that $B(x'_k) \neq dy'_k + z'_k$.

Local Computation. If at least $t + 1$ parties broadcasted "accept", then each party $P_k \in \mathcal{P}$ sets $s_k = F'(0)$. If not, else set $s_k = \perp$.

This completes the description of the protocol.

The following shorthand will be useful in description of more complicated constructions that follow. We say that P_i *conflicts* with P_j if in an execution of $\mathrm{ICP}_{i,j}(s)$, it holds that $\mathsf{conflict}_{i,j}(s) = 1$.

Proofs

We now prove the protocol $\mathrm{ICP}_{D,i}(s)$ given in the previous section is a statistical ICP protocol tolerating $t < n/2$ parties when $\kappa > \max(\lambda, 10n)$. (Recall $|\mathbb{F}| = 2^\kappa$.)

Claim 39 (Privacy) *If D and P_i remain honest throughout the sharing phase, then privacy is preserved.*

Proof Let $\mathcal{C}$ denote the set of parties corrupted by the end of the sharing phase. We show that if the dealer remains uncorrupted, then the view of the adversary can be simulated given the values $\{x_k\}_{P_k \in \mathcal{C}}$, $\{F(x_k)\}_{P_k \in \mathcal{C}}$, and $\{R(x_k)\}_{P_k \in \mathcal{C}}$. Since $F(x)$ and $R(x)$ are random polynomials of degree at most t and $|\mathcal{C}| \leq t$, a standard argument implies that the view of the adversary at the end of round 1 of the sharing phase is independent of the dealer's input s, and the values of polynomials $F(x)$ and $R(x)$ except at points $\{x_k\}_{P_k \in \mathcal{C}}$.

When P_i is honest, clearly an honest D does not broadcast $s^{(D)}$ in round 3 of the sharing phase. Thus, it remains to be shown that the view of the adversary remains independent of the dealer's input s even after P_i broadcasts $(d, B(x))$ in round 2 of the sharing phase. Recall that the view of the adversary is independent of the values of the polynomial $R(x)$ except at points $\{x_k\}_{P_k \in \mathcal{C}}$. Thus, using $R(x)$ to mask the true value of $F(x)$ (as is the case in $B(x) = dF(x) + R(x)$) does not leak any information about the $\{F(x_k)\}_{P_k \notin \mathcal{C}}$ that the adversary does not already know. ■

Claim 40 (Correctness) *If the dealer remains uncorrupted throughout the sharing phase, then correctness holds with all but negligible probability.*

Proof Clearly, when $\mathsf{conflict}_{D,i}(s)$ equals 1, each party outputs $s^{(D)}$ which equals s for an honest D, and correctness follows immediately. For the rest of the proof, we assume $\mathsf{conflict}_{D,i}(s) = 0$. We first consider the case when P_i is honest. In this case, P_i broadcasts $F'(x) = F(x)$. Since the dealer is honest, we have that $y'_k = F'(x_k)$ holds for every honest $P_k \in \mathcal{P}$. Therefore, at least $n - t \geq t + 1$ parties will broadcast "accept" in the reconstruction phase. Correctness follows immediately since $s = F'(0)$ is reconstructed.

Consider now the case when (a dishonest) P_i broadcasts $F'(x) \neq F(x)$. We will prove that for every honest $P_k \in \mathcal{P}$, neither $y'_k = F'(x'_k)$ nor $B(x'_k) \neq dy'_k + z'_k$ holds. First, we will show that $y'_k \neq F'(x'_k)$ holds with high probability when P_k is honest. Using (a) two *distinct* degree-t polynomials are identical at at most t points, and (b) the value of x'_k is completely independent of the adversary's view, we conclude that $F'(x'_k) = F(x'_k)$ holds with probability at most $t/(|\mathbb{F}| - 1)$. Since $\kappa > \max(\lambda, 10n)$ (recall $\kappa = |\mathbb{F}|$), we conclude that $y'_k \neq F'(x'_k)$ holds with all but negligible probability.

Now it remains to be shown that for every honest P_k, it holds that $B(x'_k) = dy'_k + z'_k$. Observe that an honest D checks in round 3 whether $B(x) = dF(x) + R(x)$ holds. In case it does not hold, then D broadcasts $s^{(D)}$ which in turn makes every party update $\mathsf{conflict}_{D,i}(s)$ to hold value 1. This contradicts our assumption that $\mathsf{conflict}_{D,i}(s) = 0$. Therefore, for every honest P_k it must hold that $B(x'_k) = dF(x'_k) + R(x'_k)$. That is, when the dealer is honest, $B(x'_k) = dy'_k + z'_k$ holds for every honest $P_k \in \mathcal{P}$. This completes the proof of the claim. ■

Claim 41 (Conditional Commitment) *If P_i remains uncorrupted throughout the entire protocol, then commitment holds with all but negligible probability.*

Proof It is easy to verify that the claim is true when $\mathsf{conflict}_{D,i}(s) = 1$ holds. For the rest of the proof we assume that $\mathsf{conflict}_{D,i}(s) = 0$ holds. We will prove that for every honest $P_k \in \mathcal{P}$, either $y'_k = F'(x'_k)$ holds, or $B(x'_k) \neq dy'_k + z'_k$ holds. This suffices because since $n > 2t$, at least $n - t \geq t + 1$ honest parties will broadcast "accept" and the value $F'(0)$ held by honest P_i during the sharing phase will be reconstructed.

By way of contradiction, assume that $y'_k \neq F'(x'_k)$ holds. We consider two cases depending on whether $z'_k = R(x'_k)$ holds or not. First, assume that $z'_k = R(x'_k)$ holds. Then, $B(x'_k) = dF'(x'_k) + R'(x'_k) \neq dy'_k + z'_k$ holds for an honest P_i since d is chosen from $\mathbb{F} \setminus \{0\}$ Therefore, in this case, we conclude that with high probability, party P_k will broadcast "accept".

On the other hand, suppose $z'_k \neq R(x'_k)$ holds. Note that each of x'_k, y'_k, and z'_k are received by honest P_k in round 1 of the sharing phase (in particular, before P_i chooses d and broadcasts $(d, B(x))$), and these values are never modified later. Given this, $dy'_k + z'_k$ equals $dF'(x'_k) + R'(x'_k)$ only in the event that $d = (R'(x'_k) - z'_k)/(F'(x'_k) - y'_k)$ holds. However, this event happens only with negligible probability (more concretely, with probability $1/(|\mathbb{F}| - 1)$) when P_i is honest. Therefore, we conclude that with high probability, party P_k will broadcast "accept" in this case as well. This completes the proof of the claim. ■

We will denote an ICP protocol with D as dealer with input s, and receiver P_i, using the notation $\text{ICP}_{D,i}(s)$. More generally, we denote an ICP protocol where P_i acts as dealer with input $r_{i,j}$ and P_j acts as as receiver using the notation $\text{ICP}_{i,j}(r_{i,j})$.

Also, in the protocol constructions that follow, we will execute many instances of ICP subprotocols (sometimes using the same input) in parallel. In each instance, honest parties will obviously use independent randomness. This will allow us to argue the following. Suppose ℓ instances of ICP subprotocols are executed in parallel. Then, using a simple union bound, we see that correctness property holds for each of the ℓ ICP subprotocols with probability at least $1 - (\ell t/(|\mathbb{F}| - 1))$. Similarly, we see that the conditional commitment property holds for each of the ℓ ICP subprotocols with probability at least $1 - (\ell/(|\mathbb{F}| - 1))$.

6.1.2 Weak Information Checking for Multiple Receivers

We will generalize the ICP protocol by allowing multiple receivers. This will allow us to leverage the presence of any honest party who acts as a receiver, in order to obtain stronger commitment guarantees. Second, this will also allow us to base our commitment and correctness properties on a *publicly verifiable* condition. Contrast this with the basic definition of ICP (cf. Definition 8) which guarantees commitment (resp. correctness) properties only when the receiver (resp. dealer) is honest, an inherently unverifiable condition.

As mentioned before, we note that our definition is weak in the sense that the correctness and commitment properties hold only if the view of honest parties satisfies a certain condition (denoted by $\mathsf{clean}_S(s)$). (We will strengthen this further in the next section.) Formally, we define weak information checking for multiple receivers below.

Definition 9 (Weak Information Checking for Multiple Receivers) *A two-phase protocol* $\mathrm{WICP}_S(s)$ *for parties* $\mathcal{P} = \{P_1, \ldots, P_n\}$*, where a dealer D holds initial input s and a set of receivers S with $|S| = t$, is a* weak information checking protocol for multiple receivers *if the following conditions hold for any adversary controlling at most t parties:*

Privacy *If D and all parties in S are honest at the end of the first phase (the* sharing phase*), then at the end of this phase the joint view of the malicious parties is independent of the dealer's input s.*

Conditional Correctness *By the end of the sharing phase, all parties in $\mathcal{P}$ agree on a value* $\mathsf{clean}_S(s) \in \{0, 1\}$*. Further, if* $\mathsf{clean}_S(s) = 1$*, each honest party P_k outputs a value s_k at the end of the second phase (the* reconstruction phase*). If the dealer is honest then except with statistically negligible probability, $s_k = s$ holds.*

Conditional Commitment *By the end of the sharing phase, all parties in $\mathcal{P}$ agree on a value* $\mathsf{clean}_S(s) \in \{0, 1\}$*. Further, if* $\mathsf{clean}_S(s) = 1$*, then at the end of the sharing phase, except with statistically negligible probability, the joint view of the honest parties defines a value s' such that each honest party will output s' at the end of the reconstruction phase.*

Non-triviality *If D and all parties in S are honest at the end of the sharing phase, then at the end of this phase the value of* $\mathsf{clean}_S(s)$ *will always equal* 1. $\diamondsuit$

We briefly mention that our final 3-round statistical VSS protocol is based on *replication-based secret sharing* [43, 24, 60], which loosely speaking allows the same secret share to be held by a set of receivers (rather than just a single receiver). Indeed, it is easy to see that our definition of WICP naturally leads us towards such a design. We mention that schemes based on replication-based secret sharing are typically *not linear* (this is also the case in our VSS construction).

We mention that our scheme (or more generally, typical replication-based secret sharing schemes e.g., [43]) may be viewed as a simple application of Bracha's *player-virtualization* technique [8], in which a set of parties work together to simulate a single *virtual* party. Looking at our definition, we see that S may be viewed as a single virtual party being simulated together by each $P_i \in S$. Given this, the non-triviality requirement (along with conditional correctness and conditional commitment) may be viewed as saying that such a simulation is "useful" when performed faithfully. Viewed via this lens, it is easy to verify that a WICP protocol provides exactly the same guarantees as an ICP protocol between the dealer and a virtual party S as long as the simulation is faithful. This correctness of simulation is exactly what we capture by the condition $\mathsf{clean}_S(s) = 1$.

The Protocol

We show a statistical WICP (weak information checking for multiple receivers) protocol that tolerates $t < n/2$ malicious parties. The protocol requires 3 rounds in the sharing phase, and an additional 2 rounds in the reconstruction phase.

Sharing phase. The sharing phase consists of three rounds.

Round 1: The dealer holds s. The following steps are carried out in parallel:

- The dealer executes the first round of the ICP protocol (described in Section 6.1.1) with each $P_i \in S$ as receiver on input s. We refer to such an instance of the ICP protocol as $\text{ICP}_{D,i}(s)$. Let s'_i denote the value received by P_i. Define $s'_D := s$.[3]
- Each party $P_i \in S \cup \{D\}$ picks a random value $r_{i,j}$ for every $P_j \in S \cup \{D\} \setminus \{P_i\}$ and executes the first round of the ICP protocol with P_j as receiver on input $r_{i,j}$. We refer to this instance of the ICP protocol as $\text{ICP}_{i,j}(r_{i,j})$. Let $r'_{i,j}$ denote the value received by P_j.

Round 2: Each party $P_i \in S \cup \{D\}$ does the following:

- Run round 2 of $\text{ICP}_{i,j}(r_{i,j})$, for all $P_j \in S \cup \{D\} \setminus \{P_i\}$.
- For all $P_j \in S \cup \{D\} \setminus \{P_i\}$, broadcast $a_{i,j} := s'_{i,j} + r_{i,j}$.
- For all $P_j \in S \cup \{D\} \setminus \{P_i\}$, broadcast $b_{i,j} := s'_{i,j} + r'_{j,i}$.

In parallel with the above, the dealer executes the second round of the $\text{ICP}_{D,i}(s)$ with each $P_i \in S$.

Round 3: Each party $P_i \in S \cup \{D\}$ runs round 3 of $\text{ICP}_{i,j}(r_{i,j})$, for all $P_j \in S \cup \{D\} \setminus \{P_i\}$.

In parallel with the above, the dealer D does the following for each $P_i \in S$:

- Run round 3 of $\text{ICP}_{D,i}(s)$. If D conflicts with P_i, or for some $P_j \in S \cup \{D\} \setminus \{P_i\}$, either $a_{i,j} \neq b_{j,i}$, or $a_{i,j} = \perp$, then D broadcasts $s^{(D)} := s$. In this case, each party in $\mathcal{P}$ sets local variable $\mathsf{resolved}_S(s) = 1$.

Local computation. Each party locally carries out the following steps.

1. Each party *disqualifies* the dealer, and outputs some default secret if the dealer does not follow the protocol.

2. For each $P_i \in S$, initialize $\mathsf{flag}_i(s) = 1$. Reset $\mathsf{flag}_i(s) = 0$ if any of the following is true for some $P_j \in S \setminus \{P_i\}$.

[3] We remark that the variable s'_D is used merely to simplify exposition of our claims and proofs.

(.1) P_i conflicts with P_j.

(.2) It holds that $a_{i,j} \neq b_{j,i}$.

(.3) It holds that $a_{j,i} \neq b_{i,j}$.

3. Initialize $\mathsf{resolved}_S(s) = 0$. If the dealer broadcasted $s^{(D)}$ in round 3 of the sharing phase, then each party $P_k \in \mathcal{P}$ resets $\mathsf{resolved}_S(s) = 1$.

4. Initialize $\mathsf{clean}_S(s) = 1$. Reset $\mathsf{clean}_S(s) = 0$ if each of the following conditions are satisfied.

   (.1) It holds that $\mathsf{resolved}_S(s) = 0$.

   (.2) There is pair of parties $P_i, P_j \in S$ such that P_i conflicted with P_j in round 3 of the sharing phase, and P_j also conflicted with P_i in round 3 of the sharing phase.

Reconstruction phase. The reconstruction phase consists of two rounds.

Round 1: Each party executes the following in parallel for every $P_i \in S \cup \{D\}$:

- If $P_i \neq D$, run round 1 of the reconstruction phase of $\mathrm{ICP}_{D,i}(s)$.
- For every $P_j \in S \cup \{D\} \setminus \{P_i\}$, run round 1 of the reconstruction phase of $\mathrm{ICP}_{i,j}(r_{i,j})$.

Round 2: Each party executes the following in parallel for every $P_i \in S \cup \{D\}$:

- If $P_i \neq D$, run round 2 of the reconstruction phase of $\mathrm{ICP}_{D,i}(s)$.
- For every $P_j \in S \cup \{D\} \setminus \{P_i\}$, run round 2 of the reconstruction phase of $\mathrm{ICP}_{i,j}(r_{i,j})$.

In parallel with the above D broadcasts $\tilde{s}_D := s$.[4]

Local Computation. Each party locally carries out the following steps.

1. If D broadcasted $s^{(D)}$ in round 3 of the sharing phase, then each party $P_k \in \mathcal{P}$ outputs $s_k = s^{(D)}$, and terminates the reconstruction protocol.

2. For every $P_i \in S \cup \{D\}$, perform the following:

   (.1) If $P_i \neq D$, carry out the local computation of $\mathrm{ICP}_{D,i}(s)$.
   Let $\tilde{s}_i$ denote the reconstructed value.

   (.2) For every $P_j \in S \cup \{D\} \setminus \{P_i\}$, carry out the local computation of $\mathrm{ICP}_{i,j}(r_{i,j})$.
   Let $\tilde{r}_{i,j}$ denote the reconstructed value.

3. Initialize $\mathsf{REC} = \{D\}$. Add $P_i \in S$ to REC if $\tilde{s}_i \neq \bot$ holds.

4. Delete D from REC if for some $P_i \in \mathsf{REC}$, it holds that $\tilde{s}_D \neq \tilde{s}_i$.

5. Delete P_i from REC if for some $P_j \in S \cup \{D\} \setminus \{P_i\}$ any of the following hold:

   (.1) $\tilde{r}_{i,j} \neq \bot$ and $\tilde{s}_i + \tilde{r}_{i,j} \neq a_{i,j}$.

   (.2) P_j did not conflict with P_i and $b_{i,j} - \tilde{r}_{j,i} \neq \tilde{s}_i$.

[4] We remark that this step is necessary for handling the case when all t parties in S decide to abort the reconstruction phase.

6. If $\mathsf{clean}_S(s) = 0$, then each $P_k \in \mathcal{P}$ outputs $s_k = \bot$, and terminates the reconstruction protocol.[5]

7. Each party picks some $P_i \in \mathsf{REC}$ and outputs $s' = \tilde{s}_i$.

Proofs

We now prove that the protocol given in the previous section is a statistical WICP protocol tolerating $t < n/2$ parties when $\kappa > \max(\lambda, 10n)$. (Recall $|\mathbb{F}| = 2^{\kappa}$.)

Claim 42 (Privacy) *If all parties in $S \cup \{D\}$ remain uncorrupted throughout the sharing phase, privacy is preserved.*

Proof Let $\mathcal{C}$ denote the set of parties corrupted by the end of the sharing phase. We show that if the dealer remains uncorrupted, then the view of the adversary can be simulated using simulators for each ICP subprotocol. Observe that when all parties in $S \cup \{D\}$ are honest, every ICP subprotocol is run with an honest dealer and an honest receiver. We can apply Claim 39 to conclude that each of these ICP subprotocols when considered *individually*, preserve privacy of dealer's input and the random pads that are exchanged between honest players. Furthermore, since honest parties use independent randomness (in particular, independent random masks) in each of these ICP subprotocols (including ones that run using the same input), we note that parallel executions of these ICP subprotocols still preserves privacy of each execution. Therefore, simulators that simultaneously preserve privacy for each ICP subprotocol are implied by Claim 39.

Also, honest parties are guaranteed to follow the protocol, and thus will never conflict (within any ICP subprotocol) with another honest party. Other conditions that force D to broadcast $s^{(D)}$ are easily verified not to hold when all parties $S \cup \{D\}$ is honest.

As for the values broadcast in round 2, consider an ordered pair (P_i, P_j) of parties who remain honest throughout the sharing phase. Since the dealer is honest, we have that $s'_i = s = s'_j$ holds, and since P_i, P_j are honest, $r_{i,j} = r'_{i,j}$ and $r_{j,i} = r'_{j,i}$. Since $r_{i,j}$ and $r_{j,i}$ are completely random field elements, "blinded" values $s'_i + r_{i,j}$ and $s'_j + r_{j,i}$ do not leak any information about the secret s that the adversary does not already know. ■

Note that the value of $\mathsf{clean}_S(s)$ is updated depending on broadcasts made by parties in S. Thus, by the end of the sharing phase, (honest) parties will agree on the final value of $\mathsf{clean}_S(s)$. Further, since an honest party never conflicts with another honest party, the value of $\mathsf{clean}_S(s)$ always equals 1 when all parties in $S \cup \{D\}$ are honest. Thus, our protocol satisfies the *non-triviality* requirement when all parties in $S \cup \{D\}$ are honest.

We will now make an observation about executing multiple instances of ICP subprotocols in parallel with one another. Recall that when ℓ instances of ICP subprotocols are run in parallel, then (a) the correctness property holds for each of the ℓ instances except with probability $O(\ell t/|\mathbb{F}|)$, and (b) the conditional commitment property holds for each of the ℓ instances except with probability $O(\ell/|\mathbb{F}|)$. From the claim above, we see that rest of the WICP execution other than the ICP subprotocols yields no information to the adversary that it already does not know. Finally, in our WICP subprotocol, we see that $\ell = O(n^2)$, and since $|\mathbb{F}| > 2^{\max(\lambda, 10n)}$, we see that except with negligible probability, correctness and conditional commitment holds for every ICP subprotocol simultaneously. We will use this fact repeatedly in our proofs. We are now ready to prove that

[5] We remark that executing this step at the beginning of the reconstruction phase is sufficient. However, since our main goal is to construct a VSS protocol, it will be useful to execute this step after the set REC is defined. This will simplify our proofs, and especially allow us to state and prove Corollary 45 which holds even when $\mathsf{clean}_S(s) = 0$.

our protocol satisfies the conditional commitment property, and derive the conditional correctness property of our protocol as a corollary.

Claim 43 (Conditional Commitment) *Suppose* $\mathsf{clean}_S(s) = 1$. *Then, even when the dealer is malicious, commitment holds with all but negligible probability.*

Proof The claim trivially holds when the dealer is disqualified in the sharing phase, or when the dealer broadcasts $s^{(D)}$ in round 3 of the sharing phase. For the rest of the proof, we assume that none of these events happened. Since $|S \cup \{D\}| > t$, there is at least one honest party $P_i \in S \cup \{D\}$. (For the rest of the proof, we assume that P_i is honest. Note that P_i could be the dealer.)

First, we show that honest $P_i \in S \cup \{D\}$ is contained in REC. We start by considering $P_i \in S$. Note that since D did not conflict with P_i, by the conditional commitment property of $\mathrm{ICP}_{D,i}(s)$ we have that $\tilde{s}_i = s'_i$ is true with all but negligible probability. If $s'_i = \perp$, then $a_{i,j} = \perp$ would hold, and D would be forced to broadcast $s^{(D)}$ in round 3 of the sharing phase. Since we assume that the latter event did not happen, we conclude that $\tilde{s}_i \neq \perp$. In this case, P_i is added to REC. Also, observe REC is initialized with $\{D\}$, therefore an (honest) D is always added to REC. We now argue that with high probability, none of the deletion rules apply to an honest $P_i \in S \cup \{D\}$.

1. As argued earlier, $\tilde{s}_i = s'_i$ must hold except with negligible probability. Thus, $a_{i,j} = \tilde{s}_i + r_{i,j}$ holds for honest P_i. Furthermore, since P_i is honest, $\tilde{r}_{i,j}$ revealed by every P_j must satisfy $\tilde{r}_{i,j} \in \{\perp, r_{i,j}\}$ with all but negligible probability. (This follows from the correctness property of ICP subprotocols. Note that the correctness property is violated only when malicious receiver can guess at least one "verification point" x_k held by an honest party P_k. Since when the dealer is honest, this value x_k is independent of every other ICP execution, and furthermore, is never revealed either in the sharing or reconstruction phase by P_k.) We conclude that $a_{i,j} = \tilde{s}_i + \tilde{r}_{i,j}$ holds with high probability.

2. Suppose some $P_j \in \mathsf{REC}$ did not conflict with P_i. Then, with high probability it holds that $\tilde{r}_{j,i} = r'_{j,i}$. Recalling that when P_i is honest, $\tilde{s}_i = s'_i$ holds with high probability, we see that $b_{i,j} - r'_{j,i} = b_{i,j} - \tilde{r}_{j,i} = s'_i = \tilde{s}_i$ holds with high probability.

(Note that when D is honest, by the correctness property of ICP we have that except with negligible probability no $P_j \in S$ will be able to reveal $\tilde{s}_j \notin \{\perp, \tilde{s}_D\}$. Then, following an argument similar to the one outlined above, we see that none of the deletion rules apply to an honest D.) Thus, every honest $P_i \in S \cup \{D\}$ is contained in REC.

Next, we show that for every $P_j \in \mathsf{REC}$, the value of $\tilde{s}_j$ equals $\tilde{s}_i$. For the rest of the proof we assume that both $a_{i,j} = b_{j,i}$ and $a_{j,i} = b_{i,j}$ hold, since otherwise, D would be forced to broadcast $s^{(D)}$ and consequently, commitment would follow immediately. Since $\mathsf{clean}_S(s) = 1$, then in round 3 of the sharing phase, either P_i did not conflict with P_j, or P_j did not conflict with P_i.

We consider two cases. First, suppose P_i did not conflict with P_j. In this case, since P_j is not deleted from REC, we conclude that $b_{j,i} - \tilde{r}_{i,j} = \tilde{s}_j$ holds. Furthermore since P_i is honest, by the conditional commitment property of ICP, with all but negligible probability, it must hold that $\tilde{r}_{i,j} \in \{\perp, r_{i,j}\}$. Since $b_{i,j} - \tilde{r}_{i,j} = \tilde{s}_j$, it must be the case that $\tilde{r}_{i,j} = r_{i,j}$. Recall that $a_{i,j} = b_{j,i}$ holds by assumption, so we conclude that $a_{i,j} - \tilde{r}_{i,j} = \tilde{s}_j$ must hold. Observe that for $P_i \in \mathsf{REC}$, it holds that $a_{i,j} - \tilde{r}_{i,j} = \tilde{s}_i$ (else P_i is deleted from REC). Thus, we conclude that $\tilde{s}_j = \tilde{s}_i$ holds. Given this, it is immediate that all parties output $s' = \tilde{s}_i$.

Second, suppose P_j did not conflict with P_i. In this case, since P_i is contained in REC, we conclude that $b_{i,j} - \tilde{r}_{j,i} = \tilde{s}_i$. Furthermore since P_i is honest, by the commitment property of ICP, with all but negligible probability, it must hold that $\tilde{r}_{j,i} = r_{j,i}$. Recall that $a_{j,i} = b_{i,j}$ holds by assumption, so we conclude that $a_{j,i} - \tilde{r}_{j,i} = \tilde{s}_i$ must hold. Observe that for $P_j \in \mathsf{REC}$, when $\tilde{r}_{j,i} \neq \perp$ (as is the case here) it holds that $a_{j,i} - \tilde{r}_{j,i} = \tilde{s}_j$ (else P_j is deleted from REC). Thus, we conclude that $\tilde{s}_j = \tilde{s}_i$ holds in this case as well. Given this, it is immediate that all parties output $s' = \tilde{s}_i$.

So far we have that if D neither broadcasted $s^{(D)}$ nor got disqualified during the sharing phase, then all parties output $s' = \tilde{s}_i$ where $\tilde{s}_i$ is held by an honest party P_i. By the conditional commitment property of ICP we have that with all but negligible probability $\tilde{s}_i$ reconstructed by honest P_i will equal s'_i received from D in (round 1 of) the sharing phase. Thus, with high probability, the reconstructed secret s' is fixed at the end of the sharing phase. This concludes the proof of the claim. ■

The following corollary is immediate from the proof of Claim 43.

Corollary 44 *Let $P_i \in S \cup \{D\}$. Suppose P_i is honest such that P_i holds s'_i at the end of round 1 of the sharing phase. Furthermore, suppose $\mathsf{resolved}_S(s) = 0$ and $\mathsf{clean}_S(s) = 1$ hold. Then at the end of the reconstruction phase $s' = s'_i$ holds with all but negligible probability.*

Also, observe that when $\mathsf{flag}_i(s) = 1$ holds, honest P_i does not conflict with any $P_j \in S \cup \{D\} \setminus \{P_i\}$. Following an analysis similar to the proof of Claim 43, we obtain the following corollary that holds even when $\mathsf{clean}_S(s) = 0$.

Corollary 45 *Let $P_i \in S \cup \{D\}$. Suppose P_i is honest such that P_i received s'_i at the end of round 1 of the sharing phase. Furthermore, suppose $\mathsf{flag}_i(s) = 1$ and $\mathsf{resolved}_S(s) = 0$ hold. Then at the end of the reconstruction phase for every $P_j \in \mathsf{REC}$, we have that $\tilde{s}_j = s'_i$ holds with all but negligible probability.*

Claim 46 (Conditional Correctness) *Suppose $\mathsf{clean}_S(s) = 1$. Further suppose that the dealer remains uncorrupted throughout the sharing phase. Then, correctness holds with all but negligible probability.*

Proof Clearly, a dealer that remains uncorrupted throughout the sharing phase follows the protocol, and thus is never disqualified at the end of the sharing phase. It is also easy to see that correctness holds when $\mathsf{resolved}_S(s) = 1$ holds since an honest dealer broadcasts $s^{(D)} = s$ in this case. For the rest of the proof, we assume that $\mathsf{resolved}_S(s) = 0$ holds. Note that when D is honest, by the commitment property of $\text{ICP}_{D,i}(s)$, we have that with high probability $\tilde{s}_j \in \{\perp, s\}$ holds for every $P_j \in S \cup \{D\} \setminus \{P_i\}$. Furthermore, $s'_i = s$ holds for every honest $P_i \in S \cup \{D\}$ at the end of round 1 of the sharing phase. (Recall that since $|S \cup \{D\}| > t$, there is party P_i which remains honest throughout the protocol. In particular, P_i is honest during the entire reconstruction phase.) Given this, the claim follows immediately from Corollary 44. ■

We will denote a WICP protocol with D as dealer with input s, and S as the set of receivers using the notation $\text{WICP}_S(s)$.

Also, in the protocol constructions that follow, we will execute (exponentially) many instances of WICP subprotocols in parallel. In each instance, honest parties will use independent randomness. This will allow us to argue the following. Suppose ℓ instances of WICP subprotocols are executed in parallel. Then, using a simple union bound, we see that the conditional correctness property holds for each of the ℓ WICP subprotocols except with probability $O(\ell n^3/|\mathbb{F}|)$. Similarly, we see that the conditional commitment property holds for each of the ℓ WICP subprotocols except with probability $O(\ell n^2/|\mathbb{F}|)$.

6.1.3 Information Checking for Multiple Receivers

We will now strengthen the weak ICP protocol described in the previous section. In particular, our strengthening will ensure that correctness and commitment properties are satisfied unconditionally. Formally, we define information checking for multiple receivers below.

Definition 10 (Information Checking for Multiple Receivers) *A two-phase protocol* $\mathrm{SICP}_S(s)$ *for parties* $\mathcal{P} = \{P_1, \ldots, P_n\}$*, where a dealer* D *holds initial input* s *and a set of receivers* S *with* $|S| = t$*, is an* information checking protocol for multiple receivers *if the following conditions hold for any adversary controlling at most* t *parties:*

Privacy *If* D *and all parties in* S *are honest at the end of the first phase (the* sharing phase*), then at the end of this phase the joint view of the malicious parties is independent of the dealer's input* s*.*

Correctness *Each honest party* P_k *outputs a value* s_k *at the end of the second phase (the* reconstruction phase*). If the dealer is honest then except with statistically negligible probability* $s_k = s$ *holds.*

Commitment *At the end of the sharing phase, except with statistically negligible probability, the joint view of the honest parties defines a value* s' *such that each honest party will output* s' *at the end of the reconstruction phase.* ♢

Recall that we designed the WICP protocol by virtualizing the receiver in an ICP protocol, and that this provided slightly weaker guarantees since the simulation of the virtual party may not be performed honestly. In order to obtain stronger guarantees in our SICP protocol, we virtualize the verifiers of the ICP protocol as well. Such a virtualization ensures that the conditions in the definition of information checking (cf. Definition 8) are always satisfied. In more detail, since $|S \cup \{D\}| > t$, at least one party in $S \cup \{D\}$ is honest. This turns out to be sufficient to strengthen the *conditional* commitment and correctness properties of WICP (and ICP) to their respective unconditional variants as seen in the definition above.

At a high level, our protocol introduces an additional level of information checking to the WICP protocol. In more detail, our protocol runs the 3-round WICP protocol described in Section 6.1.2 using the dealer's input secret. In parallel with the above, parties also execute a variant of the ICP protocol that uses a polynomial of high degree and whose verification points are distributed (using a WICP protocol) among subsets of parties (i.e., virtual parties).

The easy case is when the WICP execution involving the dealer's secret input itself is clean. The commitment property follows directly from the guarantees of a clean WICP execution. To the deal with the case when the above WICP execution is not clean, our protocol requires the parties to broadcast their high degree polynomial when the parties, loosely speaking, sense that there is a possibility that the WICP execution may not be clean. Polynomials revealed in this fashion will then be checked (in the reconstruction phase) against all verification points whose corresponding WICP executions were clean. Thus, in this case, commitment follows because (a) revealed polynomials are obviously fixed at the end of the sharing phase, and (b) the verification points that are used for checking the polynomial (i.e., those involving clean WICP executions) is also fixed at the end of the sharing phase. We remark that the above is merely an informal overview of our protocol and omits several minor but important details.

The Protocol

We show a statistical SICP (information checking for multiple receivers) protocol that tolerates $t < n/2$ malicious parties. Our protocol requires 3 rounds in the sharing phase, and an additional 2 rounds in the reconstruction phase.

Sharing phase. The sharing phase consists of three rounds.

Round 1: The dealer holds s. Let $S_1, \ldots, S_\eta$ be distinct subsets of $\mathcal{P} \setminus \{D\}$ of size-t. The following steps are carried out in parallel:

- The dealer chooses a random degree-τ polynomial $F(x)$ such that $F(0) = s$, and sends $F(x)$ to each $P_i \in S$. In addition, the dealer sends a random degree-τ polynomial $R_i(x)$ to each $P_i \in S$. Let $F'_i(x), R'_i(x)$ denote the polynomials received by P_i.
- The dealer executes the first round of the WICP protocol with S as receivers on input s. We refer to this instance of the WICP protocol as $\text{WICP}_S(s)$. We let s'_i denote the value $P_i \in S$ receives from the dealer in the first round of $\text{WICP}_S(s)$.
- For each $k \in [\eta]$, the dealer chooses $x_k \in \mathbb{F} \setminus \{0\}$ at random and executes the following.
  - The first round of the WICP protocol with S_k as receivers on input x_k. We refer to this instance of the WICP protocol as $\text{WICP}_{S_k}(x_k)$.
  - The first round of the WICP protocol with S_k as receivers on input $F(x_k)$. We refer to this instance of the WICP protocol as $\text{WICP}_{S_k}(F(x_k))$.
  - For each $P_i \in S$, the first round of the WICP protocol with S_k as receivers on input $R_i(x_k)$. We refer to this instance of the WICP protocol as $\text{WICP}_{S_k}(R_i(x_k))$.

Round 2: Each party $P_i \in S$ chooses a random $d_i \in \mathbb{F} \setminus \{0\}$, computes $B_i(x) = d_i F'_i(x) + R'_i(x)$ and broadcasts $(d_i, B_i(x))$. In addition, each $P_i \in S$ checks whether $s'_i = F'_i(0)$ holds. If not, P_i broadcasts "complaint".

In parallel with the above, the dealer executes the following.

- Run round 2 of $\text{WICP}_S(s)$.
- For each $k \in [\eta]$, run round 2 of $\text{WICP}_{S_k}(x_k)$.
- For each $k \in [\eta]$, run round 2 of $\text{WICP}_{S_k}(F(x_k))$.
- For each $k \in [\eta]$ and for each $P_i \in S$, run round 2 of $\text{WICP}_{S_k}(R_i(x_k))$.

Round 3: If for some $P_i \in S$, it does not hold that $d_i F(x) + R_i(x) = B_i(x)$, or P_i broadcasted "complaint" in round 2 of the sharing phase, then the dealer broadcasts $s^{(D)} = s$. Else, the dealer executes the following in parallel.

- Run round 3 of $\text{WICP}_S(s)$.
- For each $k \in [\eta]$, run round 3 of $\text{WICP}_{S_k}(x_k)$.
- For each $k \in [\eta]$, run round 3 of $\text{WICP}_{S_k}(F(x_k))$.
- For each $k \in [\eta]$ and for each $P_i \in S$, run round 3 of $\text{WICP}_{S_k}(R_i(x_k))$.

(Recall that each $P_i \in S$ maintains a local variables $\mathsf{flag}_i(s)$ in protocol $\text{WICP}_S(s)$.)

If for some $P_i \in S$, party P_i set $\mathsf{flag}_i(s) = 0$, then it broadcasts $F'_i(x)$.

Local computation. Each party locally carries out the following steps.

1. Each party *disqualifies* the dealer, and outputs some default secret if the dealer does not follow the protocol.

2. Recall that each party $P_k \in \mathcal{P}$ maintains a local variable $\mathsf{resolved}_S(s)$ in protocol $\mathrm{WICP}_S(s)$. Suppose the dealer broadcasted $s^{(D)}$ in round 3 of the sharing phase, then each party P_k sets $\mathsf{resolved}_S(s) = 1$, and terminates the local computation.

3. Each party *disqualifies* the dealer, and outputs some default secret if the dealer gets disqualified within any WICP subprotocol.

4. The local computation of each WICP subprotocol is carried out. In particular, the values of $\mathsf{clean}_S(s), \mathsf{clean}_{S_k}(x_k), \mathsf{clean}_{S_k}(F(x_k)), \mathsf{clean}_{S_k}(R_i(x_k))$ are determined for each $P_i \in S$ and for each $k \in [\eta]$.

5. Initialize $\mathcal{U} = \{P_i \mid P_i \in S \text{ and } \mathsf{flag}_i(s) = 0\}$.

6. For each $k \in [\eta]$, parties set $\mathsf{recflag}_k = 1$ if each of the following are simultaneously satisfied:

   (.1) It holds that $\mathsf{clean}_{S_k}(x_k) = 1$ and $\mathsf{resolved}_{S_k}(x_k) = 0$.

   (.2) It holds that $\mathsf{clean}_{S_k}(F(x_k)) = 1$ and $\mathsf{resolved}_{S_k}(F(x_k)) = 0$.

   (.3) For every $P_i \in S$, both $\mathsf{clean}_{S_k}(R_i(x_k)) = 1$ and $\mathsf{resolved}_{S_k}(R_i(x_k)) = 0$ hold.

   Otherwise, parties set $\mathsf{recflag}_k = 0$.

7. If there does not exist some $k \in [\eta]$ such that $\mathsf{recflag}_k = 1$, then each party *disqualifies* the dealer, and outputs some default secret.

Reconstruction phase. The reconstruction phase consists of two rounds. For the sake of clarity, we handle the following three cases separately. First, if D broadcasted $s^{(D)}$ in round 3 of the sharing phase of $\mathrm{SICP}_S(s)$ or $\mathrm{WICP}_S(s)$, i.e., if $\mathsf{resolved}_S(s) = 1$ holds, then each party $P_k \in \mathcal{P}$ outputs $s_k = s^{(D)}$, and terminates the reconstruction protocol.

Second, if $\mathsf{clean}_S(s) = 1$, then parties run the 2-round reconstruction phase of $\mathrm{WICP}_S(s)$. Each party $P_k \in \mathcal{P}$ outputs s_k as the output of $\mathrm{WICP}_S(s)$, and terminates the reconstruction protocol.

The remainder of the reconstruction phase deals with the third case where $\mathsf{clean}_S(s) = 0$.

Round 1: The following steps are carried out in parallel.

- Run round 1 of the reconstruction phase of $\mathrm{WICP}_S(s)$.
- For each $k \in [\eta]$, if $\mathsf{recflag}_k = 1$, then parties in S_k execute the following.
  - Run round 1 of the reconstruction phase of $\mathrm{WICP}_{S_k}(x_k)$.
  - Run round 1 of the reconstruction phase of $\mathrm{WICP}_{S_k}(F(x_k))$.
  - For each $P_i \in S$, run round 1 of the reconstruction phase of $\mathrm{WICP}_{S_k}(R_i(x_k))$.

Round 2: The following steps are carried out in parallel.

- Run round 2 of the reconstruction phase of $\mathrm{WICP}_S(s)$.
- For each $k \in [\eta]$, if $\mathsf{recflag}_k = 1$, then parties in S_k execute the following.
  - Run round 2 of the reconstruction phase of $\mathrm{WICP}_{S_k}(x_k)$.
  - Run round 2 of the reconstruction phase of $\mathrm{WICP}_{S_k}(F(x_k))$.
  - For each $P_i \in S$, run round 2 of the reconstruction phase of $\mathrm{WICP}_{S_k}(R_i(x_k))$.

Local Computation. Each party locally carries out the following steps:

1. For each $k \in [\eta]$ with $\mathsf{recflag}_k = 1$, perform the local computation steps of $\mathrm{WICP}_{S_k}(x_k)$, $\mathrm{WICP}_{S_k}(F(x_k))$, and of $\mathrm{WICP}_{S_k}(R_i(x_k))$ for each $P_i \in S$. Let x'_k, y'_k, and $z'_{i,k}$ denote the corresponding reconstructed values.

2. For every $P_i \in \mathcal{U}$, check whether the following is true. For each $k \in [\eta]$ with $\mathsf{recflag}_k = 1$, either

   (.1) it holds that $y'_k = F'_i(x'_k)$, or

   (.2) it holds that $B_i(x'_k) \neq d_i y'_k + z'_{i,k}$.

   If for some $k \in [\eta]$, neither condition holds, then remove P_i from $\mathcal{U}$.

3. Suppose $|\mathcal{U}| \geq 2$. If for some $P_i, P_j \in \mathcal{U}$ it holds that $F'_i(0) \neq F'_j(0)$, then each party $P_k \in \mathcal{P}$ outputs $s_k = \perp$ and terminates the reconstruction protocol. If after this step, $|\mathcal{U}| > 0$ holds, then each party $P_k \in \mathcal{P}$ picks any $P_i \in \mathcal{U}$ and outputs $s_k = F'_i(0)$.

4. Perform the local computation step of $\mathrm{WICP}_S(s)$. Each party $P_k \in \mathcal{P}$ outputs s_k as the output of $\mathrm{WICP}_S(s)$, and terminates the reconstruction protocol.

Proofs

We now prove that the protocol given in the previous section is a statistical SICP protocol tolerating $t < n/2$ parties when $\kappa > \max(\lambda, 10n)$. (Recall $|\mathbb{F}| = 2^\kappa$.)

Claim 47 (Privacy) *If all parties in $S \cup \{D\}$ remain uncorrupted throughout the sharing phase, privacy is preserved.*

Proof Let $\mathcal{C}$ denote the set of parties corrupted by the end of the sharing phase. We show that if the dealer remains uncorrupted, then the view of the adversary can be simulated given the values $\{x_k\}_{\mathcal{C} \cap S_k \neq \emptyset}$, $\{F(x_k)\}_{\mathcal{C} \cap S_k \neq \emptyset}$, and $\{R_i(x_k)\}_{P_i \in S, \mathcal{C} \cap S_k \neq \emptyset}$, along with simulators for each WICP subprotocol involving receivers that are all honest. Suppose all parties in S_k are honest. We can apply Claim 42 to conclude that each WICP subprotocol involving parties in S_k when considered *individually*, preserves privacy of dealer's input of that execution. Furthermore, observe that honest parties use independent randomness within each WICP subprotocol, and therefore, parallel executions of these WICP subprotocols still preserves privacy of each execution. Therefore, simulators that simultaneously preserve privacy for all WICP subprotocols that involve a honest set of receivers are implied by Claim 42.

Also, honest parties are guaranteed to follow the protocol, and thus will never conflict (within any ICP subprotocol) with another honest party, nor will they broadcast "complaint" and force the dealer to reveal the secret. Thus, for an honest P_i, the value $\mathsf{flag}_i(s)$ always remains 1. Therefore, no honest party P_i in S will broadcast $F(x)$.

Observe that $F(x)$ and $R_i(x)$ are random polynomials of degree at most τ, and that $|\{S_k \mid \mathcal{C} \cap S_k \neq \emptyset\}| < \eta = \tau$ holds when $n > 2t$. Using a standard argument it follows that the view of the adversary at the end of round 1 of the sharing phase is independent of the dealer's input s, and the values of polynomials $F(x)$ and $\{R_i(x)\}_{P_i \in S}$ except at points $\{x_k\}_{\mathcal{C} \cap S_k \neq \emptyset}$.

When all parties in $S \cup \{D\}$ are honest, clearly an honest D does not broadcast $s^{(D)}$ in round 3 of the sharing phase. Thus, it remains to be shown that the view of the adversary remains independent of the dealer's input s even after each $P_i \in S$ broadcasts $(d_i, B_i(x))$ in round 2 of the sharing phase.

Recall that the view of the adversary is independent of the values of the polynomial $R_i(x)$ except at points $\{x_k\}_{\mathcal{C}\cap S_k\neq\emptyset}$. Thus, using random and independent polynomials $R_i(x)$ to mask the true value of $F(x)$ (as is the case in $B_i(x) = d_iF(x) + R_i(x)$ for every honest $P_i \in S$) does not leak any information about the $\{F(x_k)\}_{\mathcal{C}\cap S_k\neq\emptyset}$ that the adversary does not already know. ■

We will now make an observation about executing multiple instances of WICP subprotocols in parallel with one another. Recall that when ℓ instances of WICP subprotocols are run in parallel, then (a) the conditional correctness property holds for each of the ℓ instances except with probability $O(\ell n^3/|\mathbb{F}|)$, and (b) the conditional commitment property holds for each of the ℓ instances except with probability $O(\ell n^2/|\mathbb{F}|)$. From the claim above, we see that rest of the SICP execution other than the WICP subprotocols yields no information to the adversary that it already does not know. Finally, in our WICP subprotocol, we see that $\ell = O(\eta) = O(2^n)$, and since $|\mathbb{F}| > 2^{\max(\lambda,10n)}$, we see that except with negligible probability, correctness and conditional commitment holds for every WICP subprotocol simultaneously. We will use this fact repeatedly in our proofs. We are now ready to prove that our protocol satisfies the commitment and correctness properties.

First, note that the value of $\mathsf{recflag}_k$ is updated depending on broadcasts made by parties in S. Thus, by the end of the sharing phase, (honest) parties will agree on the final value of $\mathsf{recflag}_k$.

Claim 48 *Suppose P_i is honest and it holds that $\mathsf{resolved}_S(s) = 0$. Then, if $\mathsf{flag}_i(s) = 0$ holds and if the dealer is not disqualified at the end of the sharing phase, then with all but negligible probability P_i is contained in $\mathcal{U}$ at the end of the reconstruction phase.*

Proof Clearly, P_i with $\mathsf{flag}_i(s) = 0$ is included in $\mathcal{U}$ when $\mathcal{U}$ is initialized. We only need to show that honest P_i is never deleted from $\mathcal{U}$.

Fix some $k \in [\eta]$ such that $\mathsf{recflag}_k = 1$. (Such a k exists, since otherwise the dealer is disqualified at the end of the sharing phase.) We will now prove that either $y'_k = F'_i(x'_k)$, or $B_i(x'_k) \neq d_iy'_k + z'_{i,k}$ holds. Since this is trivially true when $y'_k = F'_i(x'_k)$, for the rest of the proof we only consider the case when $y'_k \neq F'_i(x'_k)$ holds. Furthermore, when $z'_{i,k} = R'_i(x'_k)$ holds, we have that $B_i(x'_k) = d_iF'_i(x'_k) + R'_i(x'_k) \neq d_iy'_k + z'_{i,k}$ holds for an honest P_i except when $d_i = 0$. However, $d_i = 0$ holds for an honest P_i only with negligible probability. Thus, it suffices to consider the case when $y'_k \neq F'_i(x'_k)$ and $z'_{i,k} \neq R'_i(x'_k)$ hold simultaneously.

Since $\mathsf{recflag}_k = 1$, we have that each of the following are simultaneously satisfied:

1. It holds that $\mathsf{clean}_{S_k}(x_k) = 1$ and $\mathsf{resolved}_{S_k}(x_k) = 0$.
2. It holds that $\mathsf{clean}_{S_k}(F(x_k)) = 1$ and $\mathsf{resolved}_{S_k}(F(x_k)) = 0$.
3. It holds that $\mathsf{clean}_{S_k}(R_i(x_k)) = 1$ and $\mathsf{resolved}_{S_k}(R_i(x_k)) = 0$.

By Corollary 44, we conclude that for every honest party $P_j \in S_k \cup \{D\}$, party P_j must have received x'_k, y'_k, $z'_{i,k}$ from D in round 1 of the sharing phase of $\text{WICP}_{S_k}(x_k)$, $\text{WICP}_{S_k}(F(x_k))$, $\text{WICP}_{S_k}(R_i(x_k))$ respectively. Thus, we conclude that the dealer must have chosen the values x'_k, y'_k, and $z'_{i,k}$ before round 2, i.e., before it learned the value of d_i broadcasted by P_i. Recalling that $y'_k \neq F'_i(x'_k)$ and $z'_{i,k} \neq R'_i(x'_k)$ hold by assumption, we see that $d_iy'_k + z'_{i,k} = d_iF'_i(x'_k) + R'_i(x'_k)$ is true only when $d_i = (R'_i(x'_k) - z'_{i,k})/(F'_i(x'_k) - y'_k)$. Since honest P_i chooses $d_i \in \mathbb{F}$ at random, the probability that the above equality is satisfied is negligible. We conclude that with all but negligible probability $B_i(x'_k) \neq d_iF'_i(x'_k) + R'_i(x'_k)$ holds. This completes the proof of the claim. ■

Claim 49 *Suppose $\mathsf{resolved}_S(s) = 0$ holds. If the dealer remains uncorrupted at the end of the sharing phase, then at the end of the reconstruction phase $s = F'_i(0)$ holds for every $P_i \in \mathcal{U}$.*

Proof We begin by observing that when $n > 2t$ there exists $k \in [\eta]$ such that all parties in $S_k \cup \{D\}$ are honest. Furthermore, since (honest) parties in $S_k \cup \{D\}$ never conflict or contradict with one another, the dealer never broadcasts in round 3 of the WICP protocols with receiver set S_k. Combining this with the non-triviality condition, we see that each of the following are simultaneously satisfied:

1. It holds that $\mathsf{clean}_{S_k}(x_k) = 1$ and $\mathsf{resolved}_{S_k}(x_k) = 0$.
2. It holds that $\mathsf{clean}_{S_k}(F(x_k)) = 1$ and $\mathsf{resolved}_{S_k}(F(x_k)) = 0$.
3. It holds that $\mathsf{clean}_{S_k}(R_i(x_k)) = 1$ and $\mathsf{resolved}_{S_k}(R_i(x_k)) = 0$.

We conclude that $\mathsf{recflag}_k$ is set to 1 at the end of the sharing phase. Furthermore, by Corollary 44, we have that with all but negligible probability $x'_k = x_k$ is reconstructed. Using a similar argument, we conclude that reconstructed values y'_k, $z'_{i,k}$ both $y'_k = F(x_k)$ and $z'_{i,k} = R_i(x_k)$ are satisfied with high probability.

We now prove that when dishonest P_i broadcasts $F'_i(x) \neq F'_i(x)$, then with all but negligible probability, P_i will be removed from $\mathcal{U}$. More concretely, we will prove that $y'_k \neq F'_i(x'_k)$ and $B_i(x'_k) = d_i y'_k + z'_{i,k}$ both hold. These are precisely the conditions checked in the reconstruction phase, and if both hold, it is easy to verify that P_i will be deleted from $\mathcal{U}$. The second condition $B_i(x'_k) = d_i y'_k + z'_{i,k}$ is easily verified to be true for an honest D when $\mathsf{resolved}_S(s) = 0$. In the rest of the proof, we will show that $y'_k \neq F'_i(x'_k)$.

First, observe that by the privacy property of $\text{WICP}_{S_k}(x_k)$, $\text{WICP}_{S_k}(F(x_k))$, $\text{WICP}_{S_k}(R_i(x_k))$, we can conclude that dishonest P_i does not know the values of x'_k, y'_k, $z'_{i,k}$ from these executions. (Note that any information about y'_k and $z'_{i,k}$ leaks information about x'_k.)

We use the fact that two *distinct* degree-τ polynomials can have identical values at at most τ points. This implies that for a randomly chosen $x'_k \in \mathbb{F}$, the equation $F'_i(x'_k) = F(x'_k)$ is satisfied with probability at most $\tau/|\mathbb{F}|$. When $\tau = \eta < 2^n$ and $\kappa > \max(\lambda, 10n)$ (recall $|\mathbb{F}| = 2^\kappa$), and since honest D chooses x_k at random, we have that $y'_k \neq F'_i(x'_k)$ holds except with negligible probability. This concludes the proof of the claim. ■

Claim 50 (Commitment) *Even when the dealer is malicious, commitment holds with all but negligible probability.*

Proof The claim trivially holds when the dealer is disqualified in the sharing phase, or when $\mathsf{resolved}_S(s) = 1$ holds. For the rest of the proof, we assume that $\mathsf{resolved}_S(s) = 0$ holds.

Suppose at the end of the sharing phase, $\mathcal{U}$ is non-empty, say $\mathcal{U}$ contains P_j. Note that P_j may or may not be removed from $\mathcal{U}$ during the reconstruction phase. However, we will show that, except with negligible probability, this decision is determined by the joint view of the honest parties at the end of the sharing phase. As a first step, observe that for every $k \in [\eta]$ the value of $\mathsf{recflag}_k$ is determined at the end of the sharing phase. Now if for some $k \in [\eta]$, it holds that $\mathsf{recflag}_k = 1$, then by Corollary 44, we see that except with negligible probability, the reconstructed values x'_k, y'_k, and $\{z'_{i,k}\}_{P_i \in S}$ are completely determined by the joint view of the honest parties at the end of the sharing phase. Combining this with the fact that $F'_j(x)$ is broadcasted by P_j during the sharing phase (and since $F'_j(x)$ remains unchanged through the end of the protocol), we conclude that the decision to delete P_j from $\mathcal{U}$ is, with high probability, determined by the joint view of the honest parties at the end of the sharing phase. If $|\mathcal{U}| > 0$ holds at the end of the reconstruction phase, we conclude that commitment holds as well.

For the rest of the proof we assume that at the end of the reconstruction phase it holds that the set $\mathcal{U}$ is empty. Observe that in this case (i.e., $|\mathcal{U}| = 0$), the reconstructed value s' is simply the output of $\text{WICP}_S(s)$. Note that since $|S \cup \{D\}| > t$, there is at least one honest party $P_i \in S \cup \{D\}$. (For the rest of the proof, we assume that P_i is honest. Note that P_i could be the dealer.) We consider two cases depending on the value of $\mathsf{flag}_i(s)$. If $\mathsf{flag}_i(s) = 0$, then by Claim 48 we see that with high probability P_i is contained in $\mathcal{U}$ at the end of the reconstruction phase. This contradicts the assumption that $|\mathcal{U}| = 0$. Now, on the other hand if $\mathsf{flag}_i(s) = 1$, then by Corollary 45, we see that, except with negligible probability, the set $\{\tilde{s}_j\}_{P_j \in \mathsf{REC}}$ contains exactly one value, namely s'_i. Since s'_i is defined by the view of honest P_i at the end of round 1 of the sharing phase, we conclude that commitment holds in this case as well. This completes the proof of the claim. ∎

Claim 51 (Correctness) *Suppose that the dealer remains uncorrupted throughout the sharing phase. Then, correctness holds with all but negligible probability.*

Proof Clearly, a dealer that remains uncorrupted throughout the sharing phase certainly follows the protocol. Observe that when $n > 2t$ there exists $k \in [\eta]$ such that all parties in $S_k \cup \{D\}$ are honest. Furthermore, since (honest) parties in $S_k \cup \{D\}$ never conflict or contradict with one another, the dealer never broadcasts in round 3 of the WICP protocols with receiver set S_k. Combining this with the non-triviality condition, we see that $\mathsf{recflag}_k$ will be set to 1 at the end of the sharing phase. (See also proof of Claim 49.) Thus, we conclude that the dealer that remains uncorrupted throughout the sharing phase does not get disqualified.

Correctness holds when $\mathsf{resolved}_S(s) = 1$ since an honest dealer broadcasts $s^{(D)} = s$ in this case. For the rest of the proof, we assume that $\mathsf{resolved}_S(s) = 0$ holds. Given this, observe that the protocol can terminate either because $|\mathcal{U}| > 0$, or by running the local computation step of $\text{WICP}_S(s)$. In the former case, applying Claim 49, we conclude that s is reconstructed with high probability. In the latter case, applying Claim 46, we conclude that, with high probability, s is reconstructed here as well. This completes the proof of correctness. ∎

We will denote an SICP protocol with D as dealer with input s, and S as the set of receivers using the notation $\text{SICP}_S(s)$.

In our VSS protocol, we will execute (exponentially) many instances of SICP subprotocols in parallel. In each instance, honest parties will use independent randomness. This will allow us to argue the following. Suppose ℓ instances of SICP subprotocols are executed in parallel. Then, using a simple union bound, we see that correctness property holds for each of the ℓ SICP subprotocols except with probability $O(\ell n^3 2^n/|\mathbb{F}|)$. Similarly, we see that the commitment property holds for each of the ℓ SICP subprotocols except with probability $O(\ell n^2 2^n/|\mathbb{F}|)$.

6.2 An Inefficient 3-Round Statistical VSS Protocol

Recall that SICP provided the unconditional version of the guarantees provided by an ICP protocol between the dealer and a virtual party S. Here, we show that this is sufficient to obtain a statistical VSS protocol. At a high level, our protocol runs parallel instances of SICP protocols that are executed between the dealer (using input an additive share of s) and each of $\eta = O(2^n)$ virtual parties. The strong properties guaranteed by SICP ensures that every share (and, thus the secret) is recovered in the reconstruction phase.

6.2.1 The Protocol

We now formally describe our statistical VSS protocol that tolerates $t < n/2$ malicious parties. Our protocol requires 3 rounds in the sharing phase, and an additional 2 rounds in the reconstruction phase.

Sharing phase. The sharing phase consists of three rounds.

Round 1: The dealer holds s. Let $S_1, \ldots, S_\eta$ be distinct subsets of $\mathcal{P} \setminus \{D\}$ of size-t. The following steps are carried out in parallel:

- The dealer chooses a random η-sharing $s_1, \ldots, s_\eta$ such that $s = s_1 + \ldots + s_\eta$.
- For each $k \in [\eta]$, the dealer executes round 1 of the SICP protocol with S_k as receivers on input s_k. We refer to this instance of the SICP protocol as $\mathrm{SICP}_{S_k}(s_k)$.

Round 2: For each $k \in [\eta]$, the dealer executes round 2 of $\mathrm{SICP}_{S_k}(s_k)$.

Round 3: For each $k \in [\eta]$, the dealer executes round 3 of $\mathrm{SICP}_{S_k}(s_k)$.

Local computation. Each party locally carries out the local computation following the sharing phase of $\mathrm{SICP}_{S_k}(s_k)$ for each $k \in [\eta]$.

Reconstruction phase. The reconstruction phase consists of two rounds. If the dealer is not already disqualified, parties run the 2-round reconstruction phase of $\mathrm{SICP}_{S_k}(s_k)$ for each $k \in [\eta]$. Else, each party outputs $s' = \perp$, and terminates the reconstruction protocol.

Local Computation. For each $k \in [\eta]$, let s'_k denote the value reconstructed at the end of $\mathrm{SICP}_{S_k}(s_k)$. If for some $k \in [\eta]$ it holds that $s'_k = \perp$, then each party outputs $s' = \perp$ and terminates the protocol. Else, each party outputs $s' = s'_1 + \ldots + s'_\eta$ and terminates the protocol.

6.2.2 Proofs

We now prove that the protocol given in the previous section is a statistical VSS protocol tolerating $t < n/2$ parties when $\kappa > \max(\lambda, 10n)$. (Recall $|\mathbb{F}| = 2^\kappa$.)

Lemma 52 (Privacy) *If the dealer remains uncorrupted throughout the sharing phase, privacy is preserved.*

Proof Let $\mathcal{C}$ denote the set of parties corrupted by the end of the sharing phase. Given $n > 2t$ and $|\mathcal{C}| \leq t$, there must exist $k \in [\eta]$ such that $\mathcal{C} \cap S_k = \emptyset$ holds, i.e., all parties in $S_k \cup \{D\}$ remain honest at the end of the sharing phase. (Recall $|S_k| = t$.) We can apply Claim 47 to conclude that $\mathrm{SICP}_{S_k}(s_k)$ when considered *individually* preserves privacy of share s_k (which is the dealer's input in that execution. Furthermore, since instances of SICP subprotocols (in particular, those involving some subset of parties in S_k) employ independent randomness (in particular, independent of the share s_k), privacy is preserved even when exponentially many such subprotocols are run in parallel (even among overlapping sets of receivers). Therefore, at the end of the sharing phase, the view of the adversary is independent of the value of s_k and consequently, independent of the dealer's secret s. This completes the proof of privacy. ■

We will now make an observation about executing multiple instances of SICP subprotocols in parallel with one another. Recall that when ℓ instances of SICP subprotocols are run in parallel, then (a) the correctness property holds for each of the ℓ instances except with probability

$O(\ell n^3 2^n/|\mathbb{F}|)$, and (b) the commitment property holds for each of the ℓ instances except with probability $O(\ell n^2 2^n/|\mathbb{F}|)$. From the claim above, we see that rest of the VSS execution other than the SICP subprotocols yields no information to the adversary that it already does not know. Finally, in our SICP subprotocol, we see that $\ell = O(\eta) = O(2^n)$, and since $|\mathbb{F}| > 2^{\max(\lambda, 10n)}$, we see that except with negligible probability, correctness and conditional commitment holds for every SICP subprotocol simultaneously. We will use this fact repeatedly in our proofs. We are now ready to prove that our protocol satisfies the commitment and correctness properties.

Lemma 53 (Commitment) *Even when the dealer is malicious, commitment holds with all but negligible probability.*

Proof The claim trivially holds when the dealer is disqualified in the sharing phase. Suppose the dealer is not disqualified, we will apply Claim 50 which states that except with negligible probability, the values s'_k reconstructed were all defined by the joint view of the honest parties (in $\mathcal{P}$) at the end of the sharing phase of $\mathrm{SICP}_{S_k}(s_k)$. Since instances of SICP subprotocols employ independent randomness, commitment holds with high probability even when exponentially many such subprotocols are run in parallel. This is because we This follows from a simple union bound: suppose each instance of SICP fails to satisfy the commitment property with probability $O(n^3\eta/|\mathbb{F}|)$ (as proved in Claim 50), then when η independent instances of SICP are run, the commitment property fails to hold in any one of them (and consequently in the VSS protocol) with probability $O(n^3\eta^2/|\mathbb{F}|)$. Recalling that $\eta < 2^n$ and $|\mathbb{F}| = 2^\kappa > 2^{\max(\lambda, 10n)}$, we see that the lemma is immediate. ■

Lemma 54 (Correctness) *Suppose that the dealer remains uncorrupted throughout the sharing phase. Then, correctness holds with all but negligible probability.*

Proof Clearly, a dealer that remains uncorrupted throughout the sharing phase certainly follows the protocol, and never disqualified. We will apply Claim 51 which states that except with neglligible probability, the value s_k is reconstructed at the end of protocol $\mathrm{SICP}_{S_k}(s_k)$. Since instances of SICP subprotocols employ independent randomness, correctness holds with high probability even when exponentially many such subprotocols are run in parallel. This follows from a simple union bound: suppose each instance of SICP fails to satisfy the commitment property with probability $O(n^2\eta/|\mathbb{F}|)$ (as proved in Claim 50), then when η independent instances of SICP are run, the correctness property fails to hold in any one of them (and, consequently in the VSS protocol) with probability $O(n^2\eta^2/|\mathbb{F}|)$. Recalling that $\eta < 2^n$ and $|\mathbb{F}| = 2^\kappa > 2^{\max(\lambda, 10n)}$, we see that the lemma is immediate. ■

By Lemmas 52, 53, and 54, we conclude that the following theorem holds.

Theorem 55 *There exists 3-round statistical VSS protocol tolerating $t < n/2$ parties.*

Using a standard transformation, any VSS protocol can be used to construct a protocol for coin-tossing with similar security guarantees and resilience. In more detail, consider a protocol in which each P_i chooses random $r_i \in \mathbb{F}$, and executes VSS in parallel with every other party. At the end of the reconstruction phase, each party obtains $\{r'_j\}$ as the output of the VSS protocols, and sets the random coin $r' = r'_1 + \ldots + r'_n$. The resulting protocol is a secure protocol for coin-tossing since (a) by the privacy property of VSS, each malicious party's choice of the random field element is independent of the choices of every honest party, and (b) by the commitment property of VSS, each malicious party is committed to some field element which cannot be changed later upon knowing the choices of the honest parties. Furthermore, note that the transformation outlined above is round preserving. We omit the proofs and merely state the following.

Corollary 56 *There exists a 5-round protocol for statistically secure coin-tossing tolerating $t < n/2$ parties.*

It might be instructive to compare the above corollary with the results of Katz and Ostrovsky [66], who prove a similar upper bound on the number of rounds required to perform *two-party* coin tossing. In more detail, they show a 5-round protocol for coin-tossing (and for general secure computation) in the two-party setting. Furthermore, they show that no 4-round protocol for coin-tossing exists in the two-party setting. We remark that the question of whether a 4-round protocol for coin-tossing exists in the honest majority setting (i.e., $n > 2t$) is open. In particular, we observe that the 2-round impossibility for statistical VSS shown by Patra et al. [76] might not rule out 4-round protocols for coin-tossing. Finally, we note that our protocol has complexity exponential in the number of parties, and leave the construction of an efficient round-optimal coin-tossing protocol as an open question.

6.3 An Efficient 4-Round Statistical VSS Protocol

We now show an *efficient* 4-round statistical VSS protocol tolerating $t < n/2$ corrupted parties. In our protocol, the dealer shares a symmetric bivariate polynomial among all parties. We then employ some standard round reduction techniques from literature. (See e.g., [43].) Our main novelty is to allow the reconstruction phase of some ICP subprotocols to begin while concurrently executing the (last round of the) sharing phase of *all* ICP subprotocols. (We use a sequence of flag variables to keep track of the state of concurrent executions.) This is a necessary step in our protocol to ensure that a malicious dealer is committed to some secret at the end of the 4-round sharing phase.

6.3.1 The Protocol

We now formally describe an efficient statistical VSS protocol that tolerates $t < n/2$ malicious parties. Our protocol requires 4 rounds in the sharing phase, and additional 2 rounds in the reconstruction phase.

Sharing phase. The sharing phase consists of four rounds.

Round 1: The dealer holds s. The following steps are carried out in parallel:

- The dealer chooses a random *symmetric* bivariate polynomial $F(x, y)$ of degree t in each variable such that $F(0, 0) = s$. Note that $F(x, i) = F(i, x)$ since F is symmetric. Let $f_i(x) := F(i, x)$.
- The dealer executes the first round of the ICP protocol (described in Section 6.1.1) with $P_i \in \mathcal{P} \setminus \{D\}$ as receiver on input $f_i(j)$ for each $j \in [n]$. We refer to such an instance of the ICP protocol as $\mathrm{ICP}_{D,i}(f_i(j))$. Let $s'_{i,j}$ denote the value received by P_i.
- Each party $P_i \in \mathcal{P}$ picks a random value $r_{i,j}$ for every $P_j \in \mathcal{P} \setminus \{P_i\}$ and executes the first round of the ICP protocol with P_j as receiver on input $r_{i,j}$. We refer to this instance of the ICP protocol as $\mathrm{ICP}_{i,j}(r_{i,j})$. Let $r'_{i,j}$ denote the value received by P_j.

- For every $j \neq i$, party P_i executes the first round of the ICP protocol with D as receiver on input $r_{i,j}$ (used in the previous step). We refer to this instance of the ICP protocol as $\mathrm{ICP}_{i,D}(r_{i,j})$. Let $r_{i,j}^{(D)}$ denote the value received by D.
- For all ordered pairs (i,j), each party $P_k \in \mathcal{P}$ initializes $\mathsf{flag}_{D,i}(f_i(j)) = 0$, $\mathsf{flag}_{i,j}(r_{i,j}) = 0$, and $\mathsf{flag}_{i,D}(r_{i,j}) = 0$.

Round 2: Each party P_i does the following:

- If the values $\{(j, s'_{i,j})\}_j$ do not lie on a degree-t polynomial, then broadcast "invalid share".
- Run round 2 of $\mathrm{ICP}_{i,j}(r_{i,j})$, for all $j \neq i$.
- Run round 2 of $\mathrm{ICP}_{i,D}(r_{i,j})$, for all $j \neq i$.
- For all $j \neq i$, broadcast $a_{i,j} := s'_{i,j} + r_{i,j}$.
- For all $j \neq i$, broadcast $b_{i,j} := s'_{i,j} + r'_{j,i}$.

In parallel with the above, the dealer D does the following for all ordered pairs (i,j):

- Run round 2 of $\mathrm{ICP}_{D,i}(f_i(j))$.
- Broadcast $a_{i,j}^{(D)} = f_i(j) + r_{i,j}^{(D)}$.
- Broadcast $b_{i,j}^{(D)} = f_i(j) + r_{j,i}^{(D)}$.

Round 3: Each party P_i does the following for all $j \neq i$:

- Run round 3 of $\mathrm{ICP}_{i,j}(r_{i,j})$. If P_i conflicts with P_j, or $a_{i,j} \neq b_{j,i}$, or $a_{j,i} \neq b_{i,j}$, or $a_{i,j} \neq a_{i,j}^{(D)}$, or $b_{i,j} \neq b_{i,j}^{(D)}$, then P_i does the following steps.
  - Run round 1 of the reconstruction phase of $\mathrm{ICP}_{D,i}(f_i(j))$.
  - Run round 1 of the reconstruction phase of $\mathrm{ICP}_{j,i}(r_{j,i})$.
- Run round 3 of $\mathrm{ICP}_{i,D}(r_{i,j})$. If P_i conflicts with D, then P_i does the following:
  - For all $k \in [n]$, run round 1 of the reconstruction phase of $\mathrm{ICP}_{D,i}(f_i(k))$.

In parallel with the above, the dealer D does the following for all ordered pairs (i,j):

- Run round 3 of $\mathrm{ICP}_{D,i}(f_i(j))$. If D conflicts with P_i, or $a_{i,j} \neq a_{i,j}^{(D)}$, or $a_{i,j} = \perp$, or P_i broadcasted "invalid share" in round 2 of the sharing phase, then D does the following:
  - Broadcast $f_i^{(D)}(x) := f_i(x)$.
  - For all $k \in [n]$, run round 1 of the reconstruction phase of $\mathrm{ICP}_{i,D}(r_{i,k})$.
  - For all $k \in [n]$, run round 1 of the reconstruction phase of $\mathrm{ICP}_{k,D}(r_{k,i})$.

Round 4: Define $\mathcal{U} = \{P_i \mid D \text{ broadcasted } f_i^{(D)}(x) \text{ in round 3 of the sharing phase}\}$. If $|\mathcal{U}| > t$, then each party *disqualifies* the dealer, and outputs some default secret.

Each party P_k updates local variables in the following way. For all order pairs (i,j), if round 1 of the reconstruction phase of $\mathrm{ICP}_{D,i}(f_i(j))$, $\mathrm{ICP}_{i,j}(r_{i,j})$, or $\mathrm{ICP}_{i,D}(r_{i,j})$ was executed in the previous round, then the corresponding variables $\mathsf{flag}_{D,i}(f_i(j))$, $\mathsf{flag}_{i,j}(r_{i,j})$, $\mathrm{ICP}_{i,D}(r_{i,j})$ are set to 1.

Each party P_k does the following for all ordered pairs (i,j):

- Run round 2 of the reconstruction phase of $\mathrm{ICP}_{D,i}(f_i(j))$ if $\mathsf{flag}_{D,i}(f_i(j)) = 1$.

- Run round 2 of the reconstruction phase of $\mathrm{ICP}_{i,j}(r_{i,j})$ if $\mathsf{flag}_{i,j}(r_{i,j}) = 1$.
- Run round 2 of the reconstruction phase of $\mathrm{ICP}_{i,D}(r_{i,j})$ if $\mathsf{flag}_{i,D}(r_{i,j}) = 1$.

Local computation. Each party locally carries out the following steps.

1. Each party *disqualifies* the dealer, and outputs some default secret if the dealer does not follow the protocol.
2. For all ordered pairs (i, j) do the following.
   (.1) If D broadcasted $f_i^{(D)}(x)$ in round 3 of the sharing phase, then set $\tilde{s}_{i,j} := f_i^{(D)}(j)$, and $\mathsf{flag}_{D,i}(f_i(j)) = 1$. Else if $\mathsf{flag}_{D,i}(f_i(j)) = 1$, then perform the local computation steps following the reconstruction phase of $\mathrm{ICP}_{D,i}(f_i(j))$, and let $\tilde{s}_{i,j}$ denote the reconstructed value.
   (.2) If $\mathsf{flag}_{i,j}(r_{i,j}) = 1$, then perform the local computation steps following the reconstruction phase of $\mathrm{ICP}_{i,j}(r_{i,j})$, and let $\tilde{r}_{i,j}$ denote the reconstructed value.
   (.3) If $\mathsf{flag}_{i,D}(r_{i,j}) = 1$, then perform the local computation steps following the reconstruction phase of $\mathrm{ICP}_{i,D}(r_{i,j})$, and let $\tilde{r}_{i,j}^{(D)}$ denote the reconstructed value.
3. Each party disqualifies the dealer, and outputs some default secret if for some pair (i, j) any of the following conditions are violated:
   (.1) The points in the set $\{(k, \tilde{s}_{i,k}) \mid (\mathsf{flag}_{D,i}(f_i(k)) = 1) \wedge (\tilde{s}_{i,k} \neq \perp)\}$ must lie on a polynomial of degree at most t.
   (.2) There is a unique value $c \in \mathbb{F}$ satisfying each of the following:
      i. Suppose D broadcasted $f_i^{(D)}(x)$, then $f_i^{(D)}(j) = c$ holds.
      ii. Suppose D broadcasted $f_j^{(D)}(x)$, then $f_j^{(D)}(i) = c$ holds.
      iii. Suppose $\mathsf{flag}_{D,i}(f_i(j)) = 1$, then $\tilde{s}_{i,j} = c$ holds.
      iv. Suppose $\mathsf{flag}_{D,j}(f_j(i)) = 1$, then $\tilde{s}_{j,i} = c$ holds.
   (.3) Suppose $\mathsf{flag}_{i,D}(r_{i,j}) = 1$ holds. Then $\tilde{r}_{i,j}^{(D)} \neq \perp$ also holds.
   (.4) Suppose D broadcasted $f_j^{(D)}(x)$ and suppose $\mathsf{flag}_{i,D}(r_{i,j}) = 1$ holds. Further, if P_i did not conflict with D, then $a_{i,j}^{(D)} = b_{j,i}^{(D)} = f_j^{(D)}(i) + \tilde{r}_{i,j}^{(D)}$ must hold.

Reconstruction phase. The reconstruction phase consists of two rounds.

Round 1: The following steps are carried out in parallel for all ordered pairs (i, j).

- Run round 1 of the reconstruction phase of $\mathrm{ICP}_{D,i}(f_i(j))$ if $\mathsf{flag}_{D,i}(f_i(j)) = 0$.
- Run round 1 of the reconstruction phase of $\mathrm{ICP}_{i,j}(r_{i,j})$ if $\mathsf{flag}_{i,j}(r_{i,j}) = 0$.
- Run round 1 of the reconstruction phase of $\mathrm{ICP}_{i,D}(r_{i,j})$ if $\mathsf{flag}_{i,D}(r_{i,j}) = 0$.

Round 2: The following steps are carried out in parallel for all ordered pairs (i, j).

- Run round 2 of the reconstruction phase of $\mathrm{ICP}_{D,i}(f_i(j))$ if $\mathsf{flag}_{D,i}(f_i(j)) = 0$.
- Run round 2 of the reconstruction phase of $\mathrm{ICP}_{i,j}(r_{i,j})$ if $\mathsf{flag}_{i,j}(r_{i,j}) = 0$.

– Run round 2 of the reconstruction phase of $\mathrm{ICP}_{i,D}(r_{i,j})$ if $\mathsf{flag}_{i,D}(r_{i,j}) = 0$.

Local Computation. Each party locally carries out the following steps.

1. For all ordered pairs (i, j) do the following.

   (.1) If $\mathsf{flag}_{D,i}(f_i(j)) = 0$, then perform the local computation steps following the reconstruction phase of $\mathrm{ICP}_{D,i}(f_i(j))$, and let $\tilde{s}_{i,j}$ denote the reconstructed value.

   (.2) If $\mathsf{flag}_{i,j}(r_{i,j}) = 0$, then perform the local computation steps following the reconstruction phase of $\mathrm{ICP}_{i,j}(r_{i,j})$, and let $\tilde{r}_{i,j}$ denote the reconstructed value.

   (.3) If $\mathsf{flag}_{i,D}(r_{i,j}) = 0$, then perform the local computation steps following the reconstruction phase of $\mathrm{ICP}_{i,D}(r_{i,j})$, and let $\tilde{r}_{i,j}^{(D)}$ denote the reconstructed value.

2. Initialize $\mathsf{REC} = \emptyset$. Add P_i to REC if either of the following holds.

   (.1) P_i is contained in $\mathcal{U}$.

   (.2) For all $j \in [n]$, $\tilde{s}_{i,j} \neq \perp$ holds, and the points in the set $\{(j, \tilde{s}_{i,j})\}_j$ lie on a degree-t polynomial.

3. Delete $P_i \notin \mathcal{U}$ from REC if for some P_j any of the following hold:

   (.1) $P_j \in \mathcal{U}$ and $\tilde{s}_{i,j} \neq f_j^{(D)}(i)$.

   (.2) $\tilde{r}_{i,j} \neq \perp$ and $\tilde{s}_{i,j} + \tilde{r}_{i,j} \neq a_{i,j}$.

   (.3) P_j did not conflict with P_i and $b_{i,j} - \tilde{r}_{j,i} \neq \tilde{s}_{i,j}$.

4. For $P_i \in \mathsf{REC}$, define $f_i'(x)$ to be the degree-t polynomial which passes through all of the points in $\{(j, \tilde{s}_{i,j})\}_j$. Reconstruct a symmetric bivariate polynomial $F'(x, y)$ of degree-t from $\{(i, f_i'(x))\}_{P_i \in \mathsf{REC}}$. Each party outputs $s' = F'(0, 0)$.

6.3.2 Proofs

We prove that the protocol given in the previous section is a statistical VSS protocol that tolerates $t < n/2$ malicious parties when $\kappa > \max(\lambda, 10n)$. (Recall $|\mathbb{F}| = 2^\kappa$.)

Lemma 57 (Privacy) *If the dealer is not corrupted by the end of the sharing phase, privacy is preserved.*

Proof Let $\mathcal{C}$ denote the set of parties corrupted by the end of the sharing phase. We show that if the dealer remains uncorrupted, then the view of the adversary can be simulated given the polynomials $\{f_i(x)\}_{P_i \in \mathcal{C}}$. Since $F(x, y)$ is a random symmetric bivariate polynomial of degree at most t and $|\mathcal{C}| \leq t$, a standard argument implies that the view of the adversary is independent of the dealer's input s.

Recall that when the dealer and receiver are honest, by Claim 39, ICP subprotocols when considered *individually*, guarantee privacy of dealer's input in an information-theoretic sense. Furthermore, observe that honest parties use independent randomness within each ICP subprotocol, and therefore, parallel executions of these ICP subprotocols (i.e., those involving an honest dealer and receiver) still preserves privacy of each execution. Also, honest parties are guaranteed to follow the protocol, and thus will never conflict (within an ICP subprotocol) with another honest party, nor will they broadcast "invalid share" and force the dealer to reveal a secret share held by an honest party. Other conditions that force participating parties to begin reconstruction phase of ICP subprotocols in the sharing phase, are never satisfied for honest parties. In particular, this implies that no honest party is contained in $\mathcal{U}$.

As for the values broadcast in round 2, consider an ordered pair (P_i, P_j) of parties who remain honest throughout the sharing phase. Since the dealer is honest, we have $s'_{i,j} = F(i,j) = F(j,i) = s'_{j,i}$ and, since P_i, P_j are honest, $r_{i,j} = r'_{i,j} = r^{(D)}_{i,j}$ and $r_{j,i} = r'_{j,i} = r^{(D)}_{j,i}$. Thus, in round 2, parties P_i, P_j, and D all broadcast the same "blinded" values $F(i,j) + r_{i,j}$ and $F(j,i) + r_{j,i}$. Since $r_{i,j}$ and $r_{j,i}$ are completely random field elements, this does not leak any information about the $\{f_i(x)\}_{P_i \notin \mathcal{C}}$ that the adversary does not already know. ■

We will now make an observation about executing multiple instances of ICP subprotocols in parallel with one another. Recall that when ℓ instances of ICP subprotocols are run in parallel, then (a) the correctness property holds for each of the ℓ instances except with probability $O(\ell t/|\mathbb{F}|)$, and (b) the conditional commitment property holds for each of the ℓ instances except with probability $O(\ell/|\mathbb{F}|)$. We also observe that concurrently executing round 3 of the sharing phase along with round 1 of the reconstruction phase does not affect the correctness or conditional commitment properties of ICP subprotocols. In more detail, the correctness property is preserved since "verification points" (held by honest verifiers) are never revealed, so a dishonest receiver cannot broadcast a different polynomial which will get accepted by the honest parties. The conditional commitment property is slightly more complicated. First, note that if the dealer did not conflict with the receiver, then conditional commitment is preserved since the "multiplier" is chosen by the honest receiver after the dealer sends out verification points to the honest receivers. Of course, a malicious dealer can always conflict, and change the secret share that it originally sent to an honest receiver. (Nevertheless, this conflict happens in the sharing phase, and trivially implies that commitment will hold at the end of the sharing phase.) In our proofs we will explicitly apply the conditional commitment property only in ICP subprotocols where the dealer does not conflict with the receiver.

From the claim above, we see that rest of the VSS protocol other than the ICP subprotocols yields no information to the adversary that it already does not know. Finally, in our VSS protocol, we see that $\ell = O(n^2)$, and since $|\mathbb{F}| > 2^{\max(\lambda, 10n)}$, we see that except with negligible probability, correctness and conditional commitment holds for every ICP subprotocol simultaneously. We will use this fact repeatedly in our proofs. We are now ready to prove that our protocol satisfies the commitment and correctness properties. We do this by a series of claims.

Claim 58 *If the dealer is not disqualified in the sharing phase, then for every honest P_i and for every $P_j \in \mathcal{U}$, except with negligible probability $\tilde{s}_{i,j} = f^{(D)}_j(i)$ holds.*

Proof Suppose $P_i \in \mathcal{U}$. Then in round 4 of the sharing phase, $\tilde{s}_{i,j} := f^{(D)}_i(j)$. Furthermore, if $f^{(D)}_j(i) \neq f^{(D)}_i(j)$, then D is disqualified during local computation of the sharing phase. Since we are given that D is not disqualified, we have $\tilde{s}_{i,j} = f^{(D)}_j(i)$.

Suppose $P_i \notin \mathcal{U}$. We can conclude that D did not conflict with P_i in the sharing phase, and except with negligible probability $s'_{i,j} = \tilde{s}_{i,j}$ must hold for an honest P_i. On the other hand, since $P_j \in \mathcal{U}$, D must have broadcasted $f^{(D)}_j(x)$ and also must have revealed $r^{(D)}_{i,j}$ in the sharing phase. There are two cases to consider. First, suppose P_i conflicted with D (in round 3 of the sharing phase). In this case, P_i reveals $s'_{i,j}$ during the sharing phase. During local computation of the sharing phase, parties check if the reconstructed $\tilde{s}_{i,j}$ satisfies $\tilde{s}_{i,j} = f_j^{(D)}(i)$, and disqualify D if it doesn't. Since D was not disqualified, it must be the case that $\tilde{s}_{i,j}= f_j^{(D)}(i)$.

Second, suppose P_i did not conflict with D. Then, since P_i is honest, $\tilde{r}_{i,j}^{(D)} = r_{i,j}$ holds with all but negligible probability. Furthermore, if $a_{i,j}^{(D)} \neq a_{i,j}$, then D is required to broadcast $f_i^{(D)}(x)$, and P_i would be contained in $\mathcal{U}$. But since $P_i \notin \mathcal{U}$, we conclude that $a_{i,j}^{(D)} = a_{i,j}$. Thus, except with negligible probability, both $a_{i,j}^{(D)} = a_{i,j}$ and $\tilde{r}_{i,j}^{(D)} = r_{i,j}$ hold. Recall that when $P_i \notin \mathcal{U}$, $s'_{i,j}= \tilde{s}_{i,j}$ must hold, and therefore, $a_{i,j} = \tilde{s}_{i,j} + r_{i,j}$ must hold as well. We conclude that $a_{i,j}^{(D)} - \tilde{r}_{i,j}^{(D)} = \tilde{s}_{i,j}$. Now, if $a_{i,j}^{(D)} - \tilde{r}_{i,j}^{(D)} \neq f_j^{(D)}(i)$, then clearly D is disqualified during local computation of the sharing phase. Therefore, it must be the case that $\tilde{s}_{i,j} = f_j^{(D)}(i)$. ■

Claim 59 *If the dealer is not disqualified in the sharing phase, then for every honest P_i, it holds that $P_i \in \mathsf{REC}$ with all but negligible probability.*

Proof If $P_i \in \mathcal{U}$, then by construction $P_i \in \mathsf{REC}$ holds. Suppose $P_i \notin \mathcal{U}$. We can conclude that D did not conflict with P_i in the sharing phase, and except with negligible probability $s'_{i,j}= \tilde{s}_{i,j}$ must hold for an honest P_i. Furthermore, the values $\{(j, s'_{i,j})\}_j$ must lie on a degree-t polynomial. Otherwise, P_i would have reconstructed $\tilde{s}_{i,j}$ in the sharing phase (starting from round 2 of the sharing phase), and consequently, D would have been disqualified in the local computation of the sharing phase. Since $s'_{i,j}= \tilde{s}_{i,j}$ holds except with negligible probability, we conclude that values $\{(j, \tilde{s}_{i,j})\}_j$ must lie on a degree-t polynomial with all but negligible probability. Thus, P_i must have been added to REC. Given this, we will now show that, except with negligible probability, none of the deletion rules apply to an honest P_i.

1. For each $P_j \in \mathcal{U}$, except with negligible probability, we have $\tilde{s}_{i,j} = f_j^{(D)}(i)$ by Claim 58.
2. When $P_i \notin \mathcal{U}$, $s'_{i,j} = \tilde{s}_{i,j}$ must hold except with negligible probability. Thus, $a_{i,j} = \tilde{s}_{i,j} + r_{i,j}$. Since P_i is honest, $\tilde{r}_{i,j}$ revealed by every P_j must satisfy $\tilde{r}_{i,j} \in \{\perp, r_{i,j}\}$ with all but negligible probability. We conclude that $a_{i,j} = \tilde{s}_{i,j} + \tilde{r}_{i,j}$ holds with high probability.
3. Suppose P_j did not conflict with P_i. Then, with high probability, $\tilde{r}_{j,i}= r'_{j,i}$ holds. Recalling that when $P_i \notin \mathcal{U}$, $\tilde{s}_{i,j}= s'_{i,j}$ holds with high probability, we see that $b_{i,j} = s'_{i,j} + r'_{j,i} = \tilde{s}_{i,j} + \tilde{r}_{j,i}$ holds with high probability.

Thus, every honest P_i is contained in REC. ■

Claim 60 *If the dealer is not disqualified in the sharing phase, then for every honest P_j, and for every $P_i \in \mathsf{REC}$, it holds that $\tilde{s}_{i,j} = \tilde{s}_{j,i}$ with all but negligible probability.*

Proof The case when $P_i \in \mathcal{U}$ is handled by Claim 58. Also, the case when $P_j \in \mathcal{U}$ follows from the deletion rule in local computation of the reconstruction phase. For the rest of the proof, assume that $P_i, P_j \notin \mathcal{U}$. Recall that when $P_i, P_j \notin \mathcal{U}$, it is expected that the reconstructed values $\tilde{s}_{i,j}$, $\tilde{s}_{j,i}$ (resp.), are equal to the values $s'_{i,j}$, $s'_{j,i}$ (resp.) received from D in the first round. Furthermore, the reconstructed values $\tilde{s}_{i,j}$, $\tilde{s}_{j,i}$ are expected to be consistent with broadcasted values, $a_{i,j}$, $b_{j,i}$, $a_{j,i}$, and $b_{i,j}$. (Indeed for honest $P_j \in \mathcal{U}$, except with negligible probability, $\tilde{s}_{j,i} = s'_{j,i}$ holds, and consequently $a_{j,i} = \tilde{s}_{j,i} + r_{j,i}$ and $b_{j,i} = \tilde{s}_{j,i} + r'_{i,j}$ hold as well.) This will motivate the rest of the proof.

Next, suppose $a_{i,j} \neq b_{j,i}$, or $a_{j,i} \neq b_{i,j}$, or both P_i and P_j conflicted with each other. Observe that in all these cases, P_i and P_j are required to reveal $s'_{i,j}$ and $s'_{j,i}$ respectively. In the event that the reconstructed values $\tilde{s}_{i,j}$ and $\tilde{s}_{j,i}$ are not equal, D is disqualified in the local computation of the sharing phase. Hence, we conclude that if any of the above cases hold, then the lemma is true. For the rest of the proof, assume that $a_{i,j} = b_{j,i}$, $a_{j,i} = b_{i,j}$, and at least one of P_i, P_j did not conflict with the other.

We consider two cases. First, suppose P_j did not conflict with P_i. In this case, except with negligible probability, P_i can reveal $\tilde{r}_{j,i}$ either as $\perp$ or the correct value $r_{j,i}$. Recall that since $P_i \in \mathsf{REC}$, $b_{i,j} - \tilde{r}_{j,i} = \tilde{s}_{i,j}$ must hold, otherwise P_i is deleted from REC. Also, recall that for an honest $P_j \in \mathcal{U}$, we have $a_{j,i} = \tilde{s}_{j,i} + r_{j,i}$. Thus, we conclude that $\tilde{r}_{j,i} = r_{j,i}$, and since $a_{j,i} = b_{i,j}$, we have $\tilde{s}_{j,i} = \tilde{s}_{i,j}$.

Second, suppose P_i did not conflict with P_j. In this case, except with negligible probability, P_j successfully reveals $\tilde{r}_{i,j} = r'_{i,j}$. Since $b_{j,i} = \tilde{s}_{j,i} + r'_{i,j}$, and by assumption $a_{i,j} = b_{j,i}$, we conclude that $a_{i,j} = \tilde{s}_{j,i} + \tilde{r}_{i,j}$. This implies that $\tilde{s}_{j,i} = \tilde{s}_{i,j}$, as otherwise $a_{i,j} \neq \tilde{s}_{i,j} + \tilde{r}_{i,j}$, and P_i will be deleted from REC.

Thus, we have shown that in all cases, $\tilde{s}_{j,i} = \tilde{s}_{i,j}$ holds. ■

Claim 61 *If the dealer is not disqualified in the sharing phase, then for every $P_i, P_j \in \mathsf{REC}$, it holds that $\tilde{s}_{i,j} = \tilde{s}_{j,i}$ with all but negligible probability.*

Proof Observe that when $t < n/2$, there are at least $n - t \geq t + 1$ honest parties. By Claim 59, REC contains at least $t+1$ honest parties. Denote the first $t+1$ such honest parties in REC by $\mathcal{H}$. The polynomials $\{f'_k(x)\}_{P_k \in \mathcal{H}}$ define a bivariate polynomial $\hat{F}(x, y)$ in the natural way: namely, let $\hat{F}$ be such that $\hat{F}(x, k) = f'_k(x)$ for each $P_k \in \mathcal{H}$. By Claim 59 and Claim 60, we can conclude that $\hat{F}$ is a bivariate symmetric polynomial of degree at most t in each variable.

Since $\mathcal{H} \subseteq \mathsf{REC}$, using Claim 60, we see that $f'_i(k) = f'_k(i)$ holds for every $P_k \in \mathcal{H}$. Substituting, we have $f'_i(k) = \hat{F}(i, k)$ for all $P_k \in \mathcal{H}$. Since $|\mathcal{H}| > t$ and $f'_i(x)$ is a polynomial of degree at most t, it must hold that $\hat{F}(i, x) = f'_i(x)$. Similarly, $\hat{F}(j, x) = f'_j(x)$ for some $P_j \in \mathsf{REC}$. Finally, since $\hat{F}$ is symmetric, we have that $\tilde{s}_{i,j} = f'_i(j) = \hat{F}(i, j) = \hat{F}(j, i) = f'_j(i) = \tilde{s}_{j,i}$. ■

Lemma 62 (Commitment) *Even when the dealer is malicious, commitment holds with all but negligible probability.*

Proof The lemma trivially holds when the dealer is disqualified in the sharing phase. For the rest of the proof, we assume that the dealer is not disqualified in the sharing phase. From the proof of Claim 61, with all but negligible probabiilty there exists a bivariate symmetric polynomial $\hat{F}$ of degree at most t in each variable such that $f'_i(x) = \hat{F}(i, x)$ for every $P_i \in \mathsf{REC}$. Since $|\mathsf{REC}| > t$ (follows from Claim 59), $F' = \hat{F}$ will be reconstructed at the end of the reconstruction phase. The key observation we make is that $\hat{F}$ is completely defined by the joint view of the honest parties at the end of the sharing phase. Indeed, the values $\{\tilde{s}_{i,j}\}_{P_i, P_j \in \mathcal{H}}$ (which are all known by the end of the sharing phase) completely define the bivariate symmetric polynomial $\hat{F}$ and consequently the reconstructed secret $s' = \hat{F}(0, 0)$. This concludes the proof of the lemma. ■

Lemma 63 (Correctness) *If the dealer is not corrupted by the end of the sharing phase, then correctness holds with all but negligible probability.*

Proof When the dealer is honest, it executes ICP subprotocols using the correct input. In particular, each P_i will obtain $s'_{i,j} = F(i,j)$. Furthermore, when P_i is honest, it is easily verified that $\tilde{s}_{i,j} = F(i,j)$. Thus $\hat{F}$, which is completely defined by shares held by honest parties, will be equal to F. If the dealer is not disqualified during the sharing phase, $s = F(0,0)$ will be reconstructed at the end of the protocol, and the lemma holds.

It remains to show that an honest dealer will never be disqualified during the sharing phase. We first argue that $|\mathcal{U}| \leq t$ when D is honest. Recall that no conflicts occur in an ICP protocol as long as dealer and receiver are honest. Also, when P_i is honest, it is easy to see that $a_{i,j} = a^{(D)}_{i,j}$ always holds. Thus, parties contained in $\mathcal{U}$ have to be corrupted, and thus $|\mathcal{U}| \leq t$.

Next, we argue that when D is honest during the sharing phase, none of conditions checked in local computation of the sharing phase are violated. The correctness property of the ICP subprotocols implies that except with negligible probability, every P_i will reconstruct a value $\tilde{s}_{i,j} \in \{\perp, F(i,j)\}$. Since D is honest, these values will lie on a polynomial of degree at most t. Also, if D broadcasted $f^{(D)}_i(x)$, then $f^{(D)}_i(x) = F(i,x)$ will hold. Similarly, broadcasted $f^{(D)}_j(x)$ will always equal $F(j,x)$, and since honest D ensures that F is a symmetric polynomial, $a^{(D)}_{i,j} = b^{(D)}_{j,i} = f^{(D)}_j(i) + r^{(D)}_{i,j}$ also holds. Finally, the commitment property of the ICP subprotocols implies that for an honest D, we have $\mathsf{flag}_{i,D}(r_{i,j}) \neq \perp$, and also the reconstructed value $\tilde{r}^{(D)}_{i,j}$ equals $r^{(D)}_{i,j}$ as long as P_i did not conflict with D in the sharing phase.

From the above it is clear that D is never disqualified in the sharing phase. This concludes the proof of the lemma. ■

Chapter 7

Conclusions

In this dissertation, we have explored stronger security models for broadcast, and shown improvements regarding the round complexity of information-theoretic VSS.

We have addressed two important issues related to broadcast. The first issue is that almost all existing protocols do not distinguish between corrupted parties (who do not follow the protocol) and honest parties whose secret keys have been compromised (but who continue to behave honestly). The second issue is that all existing protocols for broadcast are insecure against an adaptive adversary who can choose which parties to corrupt as the protocol progresses. We have proposed new security models that capture these issues, and presented tight feasibility and impossibility results.

We have shown two improvements regarding the round complexity of information-theoretic VSS. First, we have argued that if the ultimate goal is to minimize round complexity of VSS for point-to-point networks, then it is important to focus on minimizing the number of rounds in which broadcast is used in addition to minimizing the total number of rounds. Towards this end, we have constructed an efficient perfectly secure VSS protocol tolerating $t < n/3$ corrupted parties that is simultaneously optimal in both the number of rounds and the number of invocations of broadcast. Second, we have constructed a statistically secure VSS protocol tolerating $t < n/2$ corrupted parties that has optimal round complexity, and an efficient statistical VSS protocol tolerating $t < n/2$ corrupted parties that requires one additional round.

Some problems are left open by this dissertation, and we discuss these below:

Efficient Round Optimal Statistical VSS with Honest Majority. We have shown an efficient statistical VSS protocol that requires four rounds in the sharing phase when $t < n/2$. It will be interesting to see if it is possible to construct an efficient statistical VSS protocol which is also round optimal.

Total Round Complexity of Statistical VSS with Honest Majority. We have shown a statistical VSS protocol which is round-optimal in the sharing phase but requires two rounds in the reconstruction phase. Thus, the total round complexity of our construction is five. The best known lower bound on the total round complexity of statistical VSS with honest majority is four. It will be interesting to close the gap between the lower and upper bounds for this problem.

Exact Round Complexity of Information-Theoretic Secure Computation. In this dissertation we have studied the exact round complexity of perfect VSS when $t < n/3$ and statistical VSS when $t < n/2$. Our study of VSS has been motivated by the fact that VSS serves as an important building block in protocols for information-theoretic secure computation. Thus, it is important to extend our study to focus on the exact round complexity of information-theoretic secure computation. Partial progress towards this has been made in [59, 43], but a complete answer is yet to be obtained.

Bibliography

[1] Michael Backes, Aniket Kate, and Arpita Patra. Computational verifiable secret sharing revisited. In *Advances in Cryptology – Asiacrypt 2011*, volume 7073 of *Lecture Notes in Computer Science*, pages 590–609. Springer, 2011.

[2] Boaz Barak, Ran Canetti, Jesper Buus Nielsen, and Rafael Pass. Universally composable protocols with relaxed set-up assumptions. In *Proc. 45th Annual Symposium on Foundations of Computer Science*, pages 186–195. IEEE, 2004.

[3] Donald Beaver, Silvio Micali, and Phillip Rogaway. The round complexity of secure protocols (extended abstract). In *Proc. 22nd Annual ACM Symposium on Theory of Computing*. ACM, 1990.

[4] Mihir Bellare, Markus Jakobsson, and Moti Yung. Round-optimal zero-knowledge arguments based on any one-way function. In *Advances in Cryptology – Eurocrypt '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 280–305. Springer, 1997.

[5] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for noncryptographic fault-tolerant distributed computations. In *Proc. 20th Annual ACM Symposium on Theory of Computing*, pages 1–10. ACM, 1988.

[6] Piotr Berman, Juan A. Garay, and Kenneth J. Perry. Towards optimal distributed consensus (extended abstract). In *Proc. 30th Annual Symposium on Foundations of Computer Science*, pages 410–415. IEEE, 1989.

[7] G. R. Blakley. Safeguarding cryptographic keys. *Proceedings of AFIPS 1979 National Computer Conference*, 48:313–317, 1979.

[8] Gabriel Bracha. An o(log n) expected rounds randomized byzantine generals protocol. *Journal of the ACM*, 34(4):910–920, 1987.

[9] Gilles Brassard, Claude Crépeau, and Moti Yung. Everything in NP can be argued in perfect zero-knowledge in a bounded number of rounds (extended abstract). In *Advances in Cryptology – Eurocrypt '89*, volume 434 of *Lecture Notes in Computer Science*, pages 192–195. Springer, 1990.

[10] Ran Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13(1):143–202, 2000.

[11] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *Proc. 42nd Annual Symposium on Foundations of Computer Science*, pages 136–145. IEEE, 2001.

[12] Ran Canetti. Universally composable signature, certification, and authentication. In *CSFW*, pages 219–. IEEE, 2004.

[13] Ran Canetti, Ivan Damgård, Stefan Dziembowski, Yuval Ishai, and Tal Malkin. Adaptive versus non-adaptive security of multi-party protocols. *Journal of Cryptology*, 17(3):153–207, 2004.

[14] Ran Canetti, Uriel Feige, Oded Goldreich, and Moni Naor. Adaptively secure multi-party computation. In *Proc. 28th Annual ACM Symposium on Theory of Computing*, pages 639–648. ACM, 1996.

[15] Ran Canetti and Marc Fischlin. Universally composable commitments. In Joe Kilian, editor, *Advances in Cryptology – Crypto 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 19–40. Springer, 2001.

[16] Ran Canetti, Joe Kilian, Erez Petrank, and Alon Rosen. Black-box concurrent zero-knowledge requires omega (log n) rounds. In *Proc. 33rd Annual ACM Symposium on Theory of Computing*, pages 570–579. ACM, 2001.

[17] Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally composable two-party and multi-party secure computation. In *Proc. 34th Annual ACM Symposium on Theory of Computing*, pages 494–503. ACM, 2002.

[18] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols. In *Proc. 20th Annual ACM Symposium on Theory of Computing*, pages 11–19. ACM, 1988.

[19] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch. Verifiable secret sharing and achieving simultaneity in the presence of faults. In *Proc. 26th Annual Symposium on Foundations of Computer Science*, pages 383–395. IEEE, 1985.

[20] Benny Chor and Michael O. Rabin. Achieving independence in logarithmic number of rounds. In *Proc. 6th Annual ACM Symposium on Principles of Distributed Computing*, pages 260–268. ACM, 1987.

[21] Richard Cleve. Limits on the security of coin flips when half the processors are faulty (extended abstract). In *Proc. 18th Annual ACM Symposium on Theory of Computing*, pages 364–369. ACM, 1986.

[22] R. Cramer and I. Damgard. Multiparty computation, an introduction, 2004. Lecture notes available at http://www.daimi.au.dk/˜ivan/mpc_2004.pdf.

[23] Ronald Cramer, Ivan Damgård, Stefan Dziembowski, Martin Hirt, and Tal Rabin. Efficient multiparty computations secure against an adaptive adversary. In *Advances in Cryptology – Eurocrypt '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 311–326. Springer, 1999.

[24] Ronald Cramer, Ivan Damgård, and Yuval Ishai. Share conversion, pseudorandom secret-sharing and applications to secure computation. In *TCC 2005: 2nd Theory of Cryptography Conference*, volume 3378 of *Lecture Notes in Computer Science*, pages 342–362. Springer, 2005.

[25] Ronald Cramer, Ivan Damgård, and Ueli M. Maurer. General secure multi-party computation from any linear secret-sharing scheme. In *Advances in Cryptology – Eurocrypt 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 316–334. Springer, 2000.

[26] Ivan Damgård and Yuval Ishai. Constant-round multiparty computation using a black-box pseudorandom generator. In *Advances in Cryptology – Crypto 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 378–394. Springer, 2005.

[27] Danny Dolev, Cynthia Dwork, Orli Waarts, and Moti Yung. Perfectly secure message transmission. *Journal of the ACM*, 40(1):17–47, 1993.

[28] Danny Dolev, Michael Fisher, Robert Fowler, Nancy Lynch, and Raymond Strong. An efficient algorithm for byzantine agreement without authentication. *Information and Control*, 52(3):257–274, 1982.

[29] Danny Dolev and H. Raymond Strong. Authenticated algorithms for Byzantine agreement. *SIAM Journal on Computing*, 12(4):656–666, 1983.

[30] Uriel Feige and Adi Shamir. Zero knowledge proofs of knowledge in two rounds. In *Advances in Cryptology – Crypto '89*, volume 435 of *Lecture Notes in Computer Science*, pages 526–544. Springer, 1990.

[31] P. Feldman and S. Micali. An optimal probabilistic protocol for synchronous Byzantine agreement. *SIAM Journal on Computing*, 26(4):873–933, 1997.

[32] Paul Feldman. A practical scheme for non-interactive verifiable secret sharing. In *Proc. 28th Annual Symposium on Foundations of Computer Science*, pages 427–437. IEEE 1987.

[33] M. J. Fischer and N. A. Lynch. A lower bound for the time to assure interactive consistency. *Information Processing Letters*, 14(4):183–186, 1982.

[34] M. Fitzi. *Generalized Communication and Security Models in Byzantine Agreement.* PhD thesis, ETH Zurich, 2002.

[35] Matthias Fitzi and Juan A. Garay. Efficient player-optimal protocols for strong and differential consensus. In *Proc. 22nd Annual ACM Symposium on Principles of Distributed Computing*, pages 211–220. ACM, 2003.

[36] Matthias Fitzi, Juan A. Garay, Shyamnath Gollakota, C. Pandu Rangan, and K. Srinathan. Round-optimal and efficient verifiable secret sharing. In *TCC 2006: 3rd Theory of Cryptography Conference*, volume 3876 of *Lecture Notes in Computer Science*, pages 329–342. Springer, 2006.

[37] Matthias Fitzi, Thomas Holenstein, and Jürg Wullschleger. Multi-party computation with hybrid security. In *Advances in Cryptology – Eurocrypt 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 419–438. Springer, 2004.

[38] Juan A. Garay, Clint Givens, and Rafail Ostrovsky. Broadcast-efficient secure multi-party computation. Cryptology ePrint Archive, Report 2012/130, 2012. `http://eprint.iacr.org/`.

[39] Juan A. Garay, Jonathan Katz, Chiu-Yuen Koo, and Rafail Ostrovsky. Round complexity of authenticated broadcast with a dishonest majority. In *Proc. 48th Annual Symposium on Foundations of Computer Science*, pages 658–668. IEEE, 2007.

[40] Juan A. Garay, Jonathan Katz, Ranjit Kumaresan, and Hong-Sheng Zhou. Adaptively secure broadcast, revisited. In *Proc. 30th Annual ACM Symposium on Principles of Distributed Computing*, pages 179–186. ACM, 2011.

[41] Juan A. Garay and Yoram Moses. Fully polynomial byzantine agreement in t+1 rounds. In *Proc. 25th Annual ACM Symposium on Theory of Computing*, pages 31–41. ACM, 1993.

[42] Rosario Gennaro. Achieving independence efficiently and securely. In *Proc. 14th Annual ACM Symposium on Principles of Distributed Computing*, pages 130–136. ACM, 1995.

[43] Rosario Gennaro, Yuval Ishai, Eyal Kushilevitz, and Tal Rabin. The round complexity of verifiable secret sharing and secure multicast. In *Proc. 33rd Annual ACM Symposium on Theory of Computing*, pages 580–589. ACM, 2001.

[44] Rosario Gennaro, Yuval Ishai, Eyal Kushilevitz, and Tal Rabin. On 2-round secure multiparty computation. In *Advances in Cryptology – Crypto 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 178–193. Springer, 2002.

[45] Oded Goldreich and Ariel Kahan. How to construct constant-round zero-knowledge proof systems for NP. *Journal of Cryptology*, 9(3):167–190, 1996.

[46] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game, or a completeness theorem for protocols with honest majority. In *Proc. 19th Annual ACM Symposium on Theory of Computing*, pages 218–229. ACM, 1987.

[47] Shafi Goldwasser and Yehuda Lindell. Secure multi-party computation without agreement. *Journal of Cryptology*, 18(3):247–287, 2005.

[48] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, 1988.

[49] S. Dov Gordon, Jonathan Katz, Ranjit Kumaresan, and Arkady Yerukhimovich. Authenticated broadcast with a partially compromised public-key infrastructure. In *Stabilization, Safety, and Security of Distributed Systems - 12th International Symposium, SSS 2010*, pages 144–158. Springer, 2010.

[50] S. Dov Gordon, Jonathan Katz, Ranjit Kumaresan, and Arkady Yerukhimovich. Authenticated broadcast with a partially compromised public-key infrastructure. *Information & Computation*, To appear, 2012.

[51] Ronald Graham and Andrew Yao. On the improbability of reaching byzantine agreements. In *Proc. 21st Annual ACM Symposium on Theory of Computing*, pages 467–478. ACM, 1989.

[52] Anuj Gupta, Prasant Gopal, Piyush Bansal, and Kannan Srinathan. Authenticated byzantine generals in dual failure model. In *ICDCN*, volume 5935 of *Lecture Notes in Computer Science*, pages 79–91. Springer, 2010.

[53] Satoshi Hada and Toshiaki Tanaka. On the existence of 3-round zero-knowledge protocols. In *Advances in Cryptology – Crypto '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 408–423. Springer, 1998.

[54] Alejandro Hevia. Universally composable simultaneous broadcast. In *SCN 2006: 5th International Conference on Security in Communication Networks*, volume 4116 of *Lecture Notes in Computer Science*, pages 18–33. Springer, 2006.

[55] Alejandro Hevia and Daniele Micciancio. Simultaneous broadcast revisited. In *Proc. 24th Annual ACM Symposium on Principles of Distributed Computing*, pages 324–333. ACM, 2005.

[56] Martin Hirt and Ueli M. Maurer. Complete characterization of adversaries tolerable in secure multi-party computation (extended abstract). In *Proc. 16th Annual ACM Symposium on Principles of Distributed Computing*, pages 25–34. ACM, 1997.

[57] Martin Hirt and Vassilis Zikas. Adaptively secure broadcast. In *Advances in Cryptology – Eurocrypt 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 466–485. Springer, 2010.

[58] Martin Hirt and Vassilis Zikas. Player-centric byzantine agreement. In *ICALP 2011: 38th International Colloquium on Automata, Languages and Programming, Part I*, volume 6755 of *Lecture Notes in Computer Science*, pages 281–292. Springer, 2011.

[59] Yuval Ishai and Eyal Kushilevitz. Randomizing polynomials: A new representation with applications to round-efficient secure computation. In *Proc. 41st Annual Symposium on Foundations of Computer Science*, pages 294–304. IEEE, 2000.

[60] M. Ito, A. Saito, and Takao Nishizeki. Secret sharing schemes realizing general access structure. In *Proc. IEEE Global Telecommunication Conf. (Globecom'87)*, pages 99–102, 1987.

[61] A. Karlin and Andrew Yao. Probabilistic lower bounds for byzantine agreement. *Unpublished manuscript*, 1984.

[62] Jonathan Katz and Chiu-Yuen Koo. On expected constant-round protocols for byzantine agreement. In *Advances in Cryptology – Crypto 2006*, volume 4117 of *Lecture Notes in Computer Science*, pages 445–462. Springer, 2006.

[63] Jonathan Katz and Chiu-Yuen Koo. Round-efficient secure computation in point-to-point networks. In *Advances in Cryptology – Eurocrypt 2007*, volume 4515 of *Lecture Notes in Computer Science*, pages 311–328. Springer, 2007.

[64] Jonathan Katz, Chiu-Yuen Koo, and Ranjit Kumaresan. Improving the round complexity of VSS in point-to-point networks. In *ICALP 2008: 35th International Colloquium on Automata, Languages and Programming, Part II*, volume 5126 of *Lecture Notes in Computer Science*, pages 499–510. Springer, 2008.

[65] Jonathan Katz, Chiu-Yuen Koo, and Ranjit Kumaresan. Improving the round complexity of VSS in point-to-point networks. *Information & Computation*, 207(8):889–899, 2009.

[66] Jonathan Katz and Rafail Ostrovsky. Round-optimal secure two-party computation. In *Advances in Cryptology – Crypto 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 335–354. Springer, 2004.

[67] Jonathan Katz, Rafail Ostrovsky, and Adam Smith. Round efficiency of multi-party computation with a dishonest majority. In *Advances in Cryptology – Eurocrypt 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 578–595. Springer, 2003.

[68] C.Y. Koo. *Studies on Fault-Tolerant Broadcast and Secure Computation.* PhD thesis, University of Maryland, 2007.

[69] Ranjit Kumaresan, Arpita Patra, and C. Pandu Rangan. The round complexity of verifiable secret sharing: The statistical case. In *Advances in Cryptology – Asiacrypt 2010*, volume 6477 of *Lecture Notes in Computer Science*, pages 431–447. Springer, 2010.

[70] Eyal Kushilevitz, Yehuda Lindell, and Tal Rabin. Information-theoretically secure protocols and security under composition. In *Proc. 38th Annual ACM Symposium on Theory of Computing*, pages 109–118. ACM, 2006.

[71] Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4:382–401, 1982.

[72] Yehuda Lindell, Anna Lysyanskaya, and Tal Rabin. Sequential composition of protocols without simultaneous termination. In *Proc. 21st Annual ACM Symposium on Principles of Distributed Computing*, pages 203–212. ACM, 2002.

[73] Silvio Micali and Tal Rabin. Collective coin tossing without assumptions nor broadcasting. In *Advances in Cryptology – Crypto '90*, volume 537 of *Lecture Notes in Computer Science*, pages 253–266. Springer, 1991.

[74] Moni Naor. Bit commitment using pseudorandomness. *Journal of Cryptology*, 4(2):151–158, 1991.

[75] Moni Naor and Moti Yung. Universal one-way hash functions and their cryptographic applications. In *Proc. 21st Annual ACM Symposium on Theory of Computing*, pages 33–43. ACM, 1989.

[76] Arpita Patra, Ashish Choudhary, Tal Rabin, and C. Pandu Rangan. The round complexity of verifiable secret sharing revisited. In *Advances in Cryptology – Crypto 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 487–504. Springer, 2009.

[77] Arpita Patra, Ashish Choudhary, and C. Pandu Rangan. Round efficient unconditionally secure multiparty computation protocol. In *Progress in Cryptology - Indocrypt 2008: 9th International Conference in Cryptology in India*, volume 5365 of *Lecture Notes in Computer Science*, pages 185–199. Springer, 2008.

[78] Arpita Patra, Ashish Choudhary, and C. Pandu Rangan. Simple and efficient asynchronous byzantine agreement with optimal resilience. In *Proc. 28th Annual ACM Symposium on Principles of Distributed Computing*, pages 92–101. ACM, 2009.

[79] Marshall C. Pease, Robert E. Shostak, and Leslie Lamport. Reaching agreement in the presence of faults. *Journal of the ACM*, 27:228–234, 1980.

[80] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *Advances in Cryptology – Crypto '91*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140. Springer, 1992.

[81] Birgit Pfitzmann and Michael Waidner. Unconditional byzantine agreement for any number of faulty processors. In *STACS '92: Proceedings of the 9th Annual Symposium on Theoretical Aspects of Computer Science*, volume 577 of *Lecture Notes in Computer Science*, pages 339–350. Springer, 1992.

[82] Manoj Prabhakaran, Alon Rosen, and Amit Sahai. Concurrent zero knowledge with logarithmic round-complexity. In *Proc. 43rd Annual Symposium on Foundations of Computer Science*, pages 366–375. IEEE, 2002.

[83] Tal Rabin. Robust sharing of secrets when the dealer is honest or cheating. *Journal of the ACM*, 41(6):1089–1109, 1994.

[84] Tal Rabin and Michael Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority. In *Proc. 21st Annual ACM Symposium on Theory of Computing*, pages 73–85. ACM, 1989.

[85] John Rompel. One-way functions are necessary and sufficient for secure signatures. In *Proc. 22nd Annual ACM Symposium on Theory of Computing*, pages 387–394. ACM, 1990.

[86] Adi Shamir. How to share a secret. *Communications of the Association for Computing Machinery*, 22(11):612–613, 1979.

[87] Sam Toueg, Kenneth Perry, and T. K. Srikanth. Fast distributed agreement. *SIAM Journal on Computing*, 16(3):445–457, 1987.

[88] Russell Turpin and Brian A. Coan. Extending Binary Byzantine Agreement to Multivalued Byzantine Agreement. *Information Processing Letters*, 18(2): 73–76, 1984.

[89] Andrew Yao. How to generate and exchange secrets (extended abstract). In *Proc. 27th Annual Symposium on Foundations of Computer Science*, pages 162–167. IEEE, 1986.