

# Authenticated Broadcast with a Partially Compromised Public-Key Infrastructure

S. DOV GORDON\*      JONATHAN KATZ\*<sup>†</sup>      RANJIT KUMARESAN\*  
ARKADY YERUKHIMOVICH\*

## Abstract

Given a public-key infrastructure (PKI) and digital signatures, it is possible to construct broadcast protocols tolerating any number of corrupted parties. Almost all existing protocols, however, 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). We explore conditions under which it is possible to construct broadcast protocols that still provide the usual guarantees (i.e., validity/agreement) to the latter.

Consider a network of  $n$  parties, where an adversary has compromised the secret keys of up to  $t_c$  honest parties and, in addition, fully controls the behavior of up to  $t_a$  other parties. We show that for any fixed  $t_c, t_a$  there exists an efficient protocol for broadcast if and only if  $2t_a + \min(t_a, t_c) < n$ . We also show that if  $t_c, t_a$  are not fixed, but are only guaranteed to satisfy the bound above, then broadcast is impossible to achieve except for a few specific values of  $n$ ; for these “exceptional” values of  $n$ , we demonstrate a broadcast protocol. Taken together, our results give a complete characterization of this problem.

---

\*Dept. of Computer Science, University of Maryland. Email: {gordon, jkatz, ranjit, arkady}@cs.umd.edu

<sup>†</sup>Work done in part while visiting IBM. Supported by NSF CNS-0447075, NSF CNS-0627306, the U.S. DoD/ARO MURI program, and the US Army Research Laboratory and the UK Ministry of Defence under agreement number W911NF-06-3-0001.

# 1 Introduction

Protocols for *broadcast* 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 protocols are fundamental for distributed computing and secure computation: they are crucial for simulating the functionality of a broadcast channel over a point-to-point network, and thus form a critical sub-component of various higher-level protocols.

Classical results of Pease, Shostak, and Lamport [15, 11] show that broadcast (and, equivalently, Byzantine agreement) is achievable in a synchronous network of  $n$  parties if and only if the number of corrupted parties  $t$  satisfies  $t < n/3$ . To go beyond this bound, some form of set-up is required. 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 exist). In this model, broadcast is possible for any  $t < n$  [15, 11, 3].

With few exceptions [5, 7] (see below), 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 (see below) 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  $P_i$  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. Clearly, it would be preferable to ensure agreement and validity for honest parties who have simply had the misfortune of having their signatures forged.

Here, 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  $2t_a + \min(t_a, t_c) < n$ . 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.
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  $t_a \leq t_c$ .) 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 thus provide a complete characterization of the problem.

**Motivating the problem.** 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. Exactly this scenario is addressed by *proactive* cryptosystems [14], in a somewhat different context.

We remark, however, that our model is meaningful even if such full-scale compromise of honest users’ secret keys is deemed unlikely. Specifically, our work provides important guarantees whenever there is a possibility that an honest user’s signature might be *forged* (whether or not the adversary learns the user’s actual secret key). Signature forgery can potentially occur due to cryptanalysis, poor implementation of cryptographic protocols [12, 13], or side-channel attacks [9, 10, 1, 16, 2]. In all these cases, it is likely that an adversary might be able to forge signatures of a small number of honest parties without being able to forge signatures of everyone.

**Prior work.** Gupta et al. [7] 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 protocols whenever  $2t_a + \min(t_a, t_c) < n$ , whereas they only show feasibility only under the weaker condition  $2(t_a + t_c) - 1 < n$ . More importantly, *all* the protocols presented in the work of Gupta et al. are *inefficient* and have message complexity exponential in  $n$ , while our protocols have polynomial complexity in all respects. Although Gupta et al. [7] also claim impossibility when  $2t_a + \min(t_a, t_c) \geq n$ , our impossibility result is stronger in that it holds relative to a weaker adversary.<sup>1</sup> Finally, Gupta et al. always 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. [5] consider broadcast in a model where the adversary can either corrupt a few players and forge signatures of *all* parties, or corrupt more players but forge no signatures. In our notation, their work handles the extremes  $t_a < n/3, t_c = n$  and  $t_a < n/2, t_c = 0$ . Our work addresses all the intermediate cases, where an adversary might be able to forge signature of some honest parties but not others.

**Organization.** Section 2 introduces our model and provides a formal definition of broadcast in our setting. In Section 3 we show that for every  $n, t_a, t_c$  satisfying the threshold condition, there exists an efficient broadcast protocol. We show our impossibility results in Section 4: namely, broadcast is impossible whenever  $t_a, t_c$  do not satisfy the threshold condition, and (other than for the exceptional values of  $n$ ) there does not exist a single, fixed protocol achieving broadcast for all  $t_a, t_c$  satisfying the threshold condition. In Section 5 and Appendix B we give positive results for the exceptional values of  $n$ . Although dealing with these “outliers” may seem like a minor point, in fact all the exceptional values of  $n$  are small and so are more likely to arise in practice. Furthermore, dealing with these exceptional values is, in some sense, the most technically challenging part of our work.

## 2 Model and Definitions

We consider the standard setting in which  $n$  players communicate in synchronous rounds via authenticated channels in a fully connected, point-to-point network. (See below for further discussion regarding the assumption of authenticated channels.) We assume a public-key infrastructure (PKI), established as follows: each party  $P_i$  runs a key-generation algorithm *Gen* (specified by the protocol)

---

<sup>1</sup>In [7], 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.

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  $(pk_1, \dots, pk_n)$ , and with each  $P_i$  holding  $sk_i$ .

A party that is *actively corrupted* (or “Byzantine”) may behave arbitrarily. All other parties are called *honest*, though we further divide the set of honest parties into those who have been *compromised* and those who have not been compromised, as discussed below. We view the set of actively corrupted players as being under the control of a single adversary  $\mathcal{A}$  coordinating their actions. We always assume such parties are *rushing*, and may wait to see the messages sent by honest parties in a given round before deciding on their own messages to send in that round. Actively corrupted parties may choose their public keys arbitrarily and even dependent on the public keys of honest parties. We continue to assume, however, that all honest parties hold the same vector of public keys.

Some honest players may be *compromised*; if  $P_i$  is compromised then the adversary  $\mathcal{A}$  is given that  $P_i$ ’s secret key  $sk_i$ . We stress that compromised players 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 players 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 corrupted, it cannot falsely inject a point-to-point message on  $P_i$ ’s behalf. In practice, authenticated channels would be guaranteed using pairwise symmetric keys (that are less easily compromised or cryptanalyzed than signing keys), or could also be ensured via physical means in small-scale networks. Note that without the assumption of authenticated channels, no meaningful results are possible.

**Definition 1** *A protocol for parties  $\mathcal{P} = \{P_1, \dots, P_n\}$ , where a distinguished dealer  $D \in \mathcal{P}$  holds an initial input  $M$ , achieves **broadcast** if the following hold:*

**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.*

Although the above definition refers to an arbitrary input  $M$  for the dealer, we will assume for simplicity that the dealer’s input is just a single bit. Broadcast for arbitrary length messages can be obtained from binary broadcast using standard techniques.

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  $n$  players, 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.

### 3 Broadcast for $(t_a, t_c)$ -Adversaries

In this section, we prove the following result:

**Theorem 1** *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: here the stated condition implies  $3t_a < n$  and the parties can thus run a standard (unauthenticated) broadcast protocol [15, 11] where the PKI is not used at all. (In this case, it makes no difference whether honest players 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 refer to the Dolev-Strong protocol [3] that achieves broadcast with a PKI, in the usual sense (i.e., when no honest parties' keys can be compromised), for any  $t < n$  corrupted parties; the Dolev-Strong protocol is reviewed in Appendix A for completeness. We say that  $P_i$  calls an execution of the DS protocol *dirty* if  $P_i$  receives valid signatures by the dealer on two different messages, or never receives any valid signed messages from the dealer; i.e., if  $P_i$  detects that the dealer is either corrupted or compromised.  $P_i$  declares the execution *clean* otherwise. The following is easy to prove (the proof is omitted due to lack of space):

**Lemma 1** *Consider an execution of protocol DS in the presence of  $t_a$  adversarial parties and  $t_c$  compromised honest parties, where  $t_a + t_c < n$ . Then:*

1. *All honest parties agree on whether an execution of DS is clean or dirty.*
2. *Agreement holds. (I.e., the outputs of all honest players are identical.)*
3. *If the dealer is honest and has not been compromised, then validity holds (i.e., all honest parties agree on the dealer's input) and the execution is clean. If the dealer is honest and the execution is clean, then validity also holds.*

Thus, DS fails to satisfy Definition 1 only when the dealer is *honest* but *compromised*. Our protocol (cf. Figure 1) guarantees validity even in this case (while leaving the other cases unaffected).

**Protocol 1**

**Inputs:** Let  $D$  be the dealer, with input bit  $b$ .

**Computation:**

1.  $D$  sends  $b$  to all other players. Let  $b_i$  be the value received by  $P_i$  from  $D$  in this step (if the dealer sends nothing to  $P_i$ , then  $b_i$  is taken to be some default value).
2. In parallel, each party  $P_i$  acts as the dealer in an execution of DS( $b_i$ ) (the original dealer  $D$  runs DS( $b$ )). We let  $|\text{CLEAN}_0|$  (resp.,  $|\text{CLEAN}_1|$ ) denote the number of executions of DS that are both *clean* and result in output 0 (resp., 1).

**Output:** If  $|\text{CLEAN}_0| \geq |\text{CLEAN}_1|$  then all parties output 0; otherwise, all parties output 1.

Figure 1: A broadcast protocol for  $t_c < t_a$  and  $2t_a + t_c < n$ .

**Theorem 2** *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 1 applies.

**Agreement:** By Lemma 1, the output of each honest player is the same in every execution of DS in step 2, and all honest parties agree on whether any given execution of DS is clean or dirty. So all honest players agree on  $|\text{CLEAN}_0|$  and  $|\text{CLEAN}_1|$ , and agreement follows.

**Validity:** Assume the dealer is honest (whether compromised or not). Letting  $t_h$  denote the number of honest and non-compromised players, we see that  $t_h + t_a + t_c = n > 2t_a + t_c$  and so  $t_h > t_a$ . Thus, there are  $t_h$  honest and non-compromised dealers in step 2 of Protocol 1, and (since  $D$  is honest) each of these runs  $\text{DS}(b)$  where  $b$  is the initial input of  $D$ . By Lemma 1, all honest players output  $b$  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  $1 - b$ , as only adversarial players will possibly run  $\text{DS}(1 - b)$  in step 2. The majority value output by the honest players is therefore always equal to the original dealer's input  $b$ . ■

## 4 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  $n \leq 2t_a + \min(t_a, t_c)$ , thus proving that Theorem 1 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  $n$ .

### 4.1 The Three-Player Case

We first present a key lemma that will be useful for the proofs of both results described above. For this, define a general adversary  $\mathcal{A}$  as follows:

**Definition 2** Let  $\mathcal{S}$  be a set of pairs  $\{(S_a^1, S_c^1), (S_a^2, S_c^2), \dots\}$  where  $S_a^i, S_c^i \subset \{P_1, \dots, P_n\}$ . An  $\mathcal{S}$ -adversary can choose any  $i$ , and actively corrupt any subset of the players in  $S_a^i$  and additionally compromise the secret keys of any subset of the players in  $S_c^i$ .

We restrict our attention to the case of three parties (named  $A$ ,  $B$ , and  $C$ ) and  $\mathcal{S}$  defined as follows:

$$\mathcal{S} = \left\{ \begin{array}{l} (\{A\}, \emptyset) \\ (\{B\}, \{A\}) \\ (\{C\}, \{A\}) \end{array} \right\}. \quad (1)$$

**Lemma 2** *In the presence of an  $\mathcal{S}$ -adversary, for  $\mathcal{S}$  defined as above, there does not exist a protocol achieving broadcast for dealer  $A$ .*

**Proof** Suppose, towards a contradiction, that there exists a protocol  $\Pi$  for computing broadcast in the presence of an  $\mathcal{S}$ -adversary when  $A$  is the dealer. Let  $\Pi_A, \Pi_B, \Pi_C$  denote the code specified by  $\Pi$  for players  $A, B$ , and  $C$ , respectively.

Consider an experiment in which four machines are arranged in a rectangle (see Figure 2). The top left and top right nodes will run  $\Pi_B$  and  $\Pi_C$ , respectively. The bottom left node will run  $\Pi_A$  using input 1, and the bottom right node will run  $\Pi_A$  using input 0. Public and secret keys for  $A, B$ , and  $C$  are generated honestly, and both executions of  $\Pi_A$  use the same keys.

**Claim 1** *In the experiment of Figure 2,  $\Pi_B$  outputs 1.*

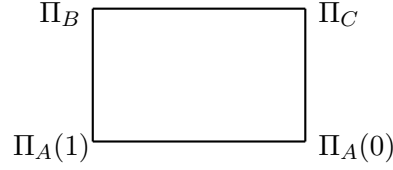


Figure 2: A mental experiment involving a four-node network.

**Proof** Consider an execution in the real network of three players, in the case where  $A$  holds input 1 and the adversary corrupts  $C$  and compromises the secret key of  $A$ . The adversary then simulates the right edge of the rectangle from Figure 2 while interacting with the (real) honest players  $A$  and  $B$  (running the code for  $\Pi_A(1)$  and  $\Pi_B$ , respectively). That is, every time the corrupted player  $C$  receives a message from  $B$  the adversary forwards this message to its internal copy of  $\Pi_C$ , and every time  $C$  receives a message from  $A$  the adversary forwards this message to its internal copy of  $\Pi_A(0)$ . Similarly, any message sent by  $\Pi_C$  to  $\Pi_B$  is forwarded to the real player  $B$ , and any message sent by  $\Pi_A(0)$  to  $\Pi_A(1)$  is forwarded to the real player  $A$ . (Messages between  $\Pi_C$  and  $\Pi_A(0)$  are forwarded internally.) This defines a legal  $\mathcal{S}$ -adversary. If  $\Pi$  is a secure protocol, validity must hold and so  $B$  in the real network (and hence  $\Pi_B$  in the mental experiment) must output 1. ■

**Claim 2** *In the experiment of Figure 2,  $\Pi_C$  outputs 0.*

The proof is the same as above.

**Claim 3** *In the experiment of Figure 2,  $\Pi_B$  and  $\Pi_C$  output the same value.*

**Proof** Consider an execution in the real network of three players when the adversary corrupts  $A$  (and does not further compromise anyone). The adversary then simulates the bottom edge of the rectangle when interacting with the real players  $B$  and  $C$ , in the obvious way. Since this defines a legal  $\mathcal{S}$ -adversary, security of  $\Pi$  implies that agreement must hold between  $B$  and  $C$  in the real network and so the outputs of  $\Pi_B$  and  $\Pi_C$  must agree in the mental experiment. ■

The three claims are contradictory, and so we conclude that no secure protocol  $\Pi$  exists. ■

We remark that the entire proof works, unchanged, even if we relax our definition of broadcast and allow agreement/validity to fail with negligible probability.

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

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

**Proof** We prove the theorem by demonstrating that a broadcast protocol  $\Pi$  secure in the presence of a  $(t_a, t_c)$ -adversary with  $2t_a + \min(t_a, t_c) \geq n$ , yields a protocol  $\Pi'$  for 3-player broadcast in the presence of an  $\mathcal{S}$ -adversary for  $\mathcal{S}$  as defined in the previous section. Using Lemma 2, this shows that such a protocol  $\Pi$  cannot exist. In fact, we show this even assuming the dealer is fixed in advance.

Assume that such a protocol  $\Pi$  exists. We construct a protocol  $\Pi'$  for 3-player broadcast by having each player simulate a subset of the players in the  $n$ -player protocol  $\Pi$ . The simulation proceeds in the obvious way, by having each of the 3 players run the code of the parties they simulate in  $\Pi$ . They forward any messages sent by the simulated parties to the player simulating the destination party, who uses these as incoming messages for his simulated players. To provide a PKI for the simulated protocol we view the keys of each of the 3 players as consisting of multiple keys. Player  $A$ 's public key is  $PK_A = (pk_1, \dots, pk_a)$  and his secret key is  $SK_A = (sk_1, \dots, sk_a)$  for some number of simulated players  $a$ . The players in the 3-player protocol determine their outputs from the outputs of the players they simulate. If all players simulated by  $A$  output the same value  $b$  in the simulated protocol, then  $A$  outputs  $b$ . Otherwise,  $A$  outputs a special value  $\perp$ . Note that an adversarial player can only simulate adversarial players and an honest but compromised player can only simulate compromised players since the adversary learns all the secret keys of player  $A$ 's simulated players when  $A$ 's key is compromised.

We let  $A$  simulate a set of at most  $\min(t_a, t_c)$  players, including the dealer, and let  $B$  and  $C$  each simulate at most  $t_a$  players. Since  $2t_a + \min(t_a, t_c) \geq n$ , it is possible to do this in such a way that each of the  $n$  original players is simulated by one of  $A, B$ , or  $C$ . We now consider each of the three allowed types of corruption for the adversary  $\mathcal{A}$  as per Definition 2, and demonstrate that the corresponding corruption in the  $n$ -player protocol is also “legal”: that is, we demonstrate that the allowed actions for  $\mathcal{A}$  translate into adversarial actions for which the non-faulty players in  $\Pi$  terminate correctly, achieving broadcast in the simulated  $n$ -player protocol. This implies a secure 3-player broadcast protocol in the presence of  $\mathcal{A}$ .

Recall that, by assumption,  $\Pi$  is secure against a  $(t_a, t_c)$ -adversary; as long as no more than  $t_a$  players are corrupt, and no more than  $t_c$  are compromised,  $\Pi$  satisfies the requirements of authenticated broadcast. If  $\mathcal{A}'$  chooses the pair  $(\{A\}, \emptyset)$ , all players simulated by  $A$  in  $\Pi'$  are corrupt and the players simulated by  $B$  and  $C$  are honest and non-compromised. Since,  $\min(t_a, t_c) \leq t_a$ , this is an allowed corruption for a  $(t_a, t_c)$ -adversary, and  $\Pi$  executes correctly implying that  $\Pi'$  terminates with the correct output. Next, if  $\mathcal{A}'$  chooses  $(\{B\}, \{A\})$  this will result in a  $(t_a, \min(t_a, t_c))$  corruption. Since  $\min(t_a, t_c) \leq t_c$ , this corruption type is also permitted in  $\Pi$ , and  $\Pi'$  executes correctly. Finally, the corruption type  $(\{C\}, \{A\})$  is handled identically to that of  $(\{B\}, \{A\})$ . Since we proved in Lemma 2 that no such protocol  $\Pi'$  exists, this proves the theorem. ■

### 4.3 Impossibility of Broadcast in the Presence of 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  $2t_a + \min(t_a, t_c) < n$ . 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 B.

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

**Proof** Assume there exists a protocol  $\Pi$  for  $n$  satisfying the stated inequality. We show that this implies a protocol  $\Pi'$  for broadcast with 3 players in the presence of the adversary  $\mathcal{A}$  from Definition 2. By Lemma 2, we conclude that  $\Pi$  cannot exist. In fact, we show this even assuming the dealer is fixed in advance.



We construct  $\Pi'$  using a player simulation argument as in the previous section. Let  $A$  simulate a set of at most  $\lfloor \frac{n-1}{2} \rfloor$  players, and including the dealer.  $B$  and  $C$  each simulate at most  $\lfloor \frac{n-1}{3} \rfloor$  players and at least one player. By the stated inequality, it is possible to do this in such a way that  $A$ ,  $B$ , and  $C$  simulate all  $n$  players. We now show that the three allowed types of corruption for  $\mathcal{A}$  (in the 3-party network) are also allowed corruption patterns for the  $n$ -player threshold adversary  $\mathcal{A}'$ .

If  $\mathcal{A}$  corrupts  $A$ , this corresponds to corruption of  $\lfloor \frac{n-1}{2} \rfloor$  players in  $\Pi$  (and no compromised players). Since  $2\lfloor \frac{n-1}{2} \rfloor < n$ , this is a legal corruption pattern for a threshold adversary and  $\Pi$  should remain secure. If  $\mathcal{A}$  corrupts  $B$  and compromises  $A$ , this corresponds to  $t_a = \lfloor \frac{n-1}{3} \rfloor$  players and  $t_c = \lfloor \frac{n-1}{2} \rfloor$  players in  $\Pi$ . Since  $2\lfloor \frac{n-1}{3} \rfloor + \min\{\lfloor \frac{n-1}{3} \rfloor, \lfloor \frac{n-1}{2} \rfloor\} = 3\lfloor \frac{n-1}{3} \rfloor < n$ , this is again a legal corruption pattern for a threshold adversary and  $\Pi$  should remain secure. The case when  $C$  is corrupted and  $A$  is compromised is exactly analogous. ■

## 5 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 4 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, since the honest parties are no longer assumed to “know” whether  $t_a \leq t_c$ .

Our protocol, which we refer to as **authLSP**, is an authenticated version of the exponential protocol of Lamport et al. [11]; see Figure 3. Although the message complexity of this protocol is exponential in the number of players, the maximum number of players considered here is 12 so this is not a serious problem (at least in theory). However, we also provide a more efficient protocol based on the protocol of Dolev-Strong [3] in Appendix B, under the assumption that there is at least one honest and uncompromised player.

We say a message  $M$  is *valid* if it has the form  $(v, s_{P_1}, \dots, s_{P_i})$ , where all  $P_j$ 's are distinct, the string  $s_{P_j}$  is a valid signature on  $(v, s_{P_1}, \dots, s_{P_{j-1}})$  relative to the verification key of  $P_j$ , and one of the  $s_{P_j}$  is the signature of the dealer. (We note that **authLSP** is defined recursively, and the criteria for deciding if a message is valid is defined with respect to the dealer of the *local* execution.) We also assume implicitly that each message has a tag identifying which execution it belongs to. These tags (together with uncompromised signatures) will prevent malicious players from substituting the messages of one execution for those of another execution. We refer to  $v$  as the *content* of such a message. When we say that an execution of **authLSP** satisfies agreement or validity (cf. Definition 1), we mean that the output is a valid message whose *content* satisfies these properties. We note that in the protocol **authLSP**, it is possible for honest players to have invalid input. In this case, we change the definition of validity slightly to require that all honest players (including the dealer) output messages with content 0. Finally, we let  $t_h = n - t_c - t_a$  denote the number of honest and uncompromised 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 follow readily from the results of [11]; we do not prove them here.

**Lemma 3** *If  $n > 3m$  and  $m \geq t_a$ , **authLSP**( $m$ ) achieves validity and agreement.*

**Lemma 4** *If the dealer  $D$  is honest, and  $n > 2t_a + m$ , **authLSP**( $m$ ) achieves validity and agreement.*

We now prove several additional lemmas about Protocol **authLSP**.

**Protocol authLSP( $m$ )**

**Inputs:** The protocol is parameterized by an integer  $m$ . Let  $D$  be the dealer with input  $M$  of the form  $M = (v, s_{P_1}, \dots, s_{P_i})$  with  $0 \leq i \leq n$  ( $M$  is not necessarily valid).

**Case 1:**  $m = 0$

1. If the content of  $M$  is not in  $\{0, 1\}$ ,  $D$  sets  $M = 0$ .<sup>a</sup>  $D$  sends  $M_d = (M, \text{Sign}_{s_{k_D}}(M))$  to all other players and outputs  $M_d$ .
2. Letting  $M_i$  denote the message received by  $P_i$ ,  $P_i$  outputs  $M_i$ .

**Case 2:**  $m > 0$

1. If the content of  $M$  is not in  $\{0, 1\}$ ,  $D$  sets  $M = 0$ .  $D$  sends  $M_d = (M, \text{Sign}_{s_{k_D}}(M))$  to all other players and outputs  $M_d$ .
2. Let  $\mathcal{P}' = \mathcal{P} \setminus \{D\}$ . For  $P_i \in \mathcal{P}'$ , let  $M_i$  denote the message received by  $P_i$  from  $D$ .  $P_i$  plays the dealer in  $\text{authLSP}(m - 1)$  for the rest of the players in  $\mathcal{P}'$ , using message  $M_i$  as its input.
3. Each  $P_i$  locally does the following: for each  $P_j \in \mathcal{P}'$ , let  $M_j$  be the output of  $P_i$  when  $P_j$  played the dealer in  $\text{authLSP}(m - 1)$  in step 2. For each  $M_j$ ,  $P_i$  sets value  $b_j$  as follows:

$$b_j = \begin{cases} \text{the content of } M_j & \text{if } M_j \text{ is valid} \\ \perp & \text{otherwise} \end{cases}$$

(We stress that the above also includes the output of  $P_i$  when he was dealer in step 2.)  $P_i$  computes  $b^* = \text{majority}(b_j)$ . If there is no value in strict majority,  $P_i$  outputs 0.

4.  $P_i$  outputs the first valid message  $M_j$  (lexicographically) with content  $b^*$ .

---

<sup>a</sup>As mentioned in the text, we assume the dealer also includes the appropriate tag identifying which execution  $M$  belongs to. We do not mention this again going forward.

Figure 3: Protocol authLSP.

**Lemma 5** *If the dealer is honest and uncompromised, Protocol authLSP( $m$ ) guarantees validity and agreement for all values of  $m$ .*

**Proof** Let  $D$  be the dealer with input that has content  $b_d$ . (Recall that if  $b_d \notin \{0, 1\}$ , then  $D$  switches his input for valid input with content  $b_d = 0$ .) It follows from the protocol description that  $D$  outputs a valid message with content  $b_d$ . Furthermore, when an honest player is dealer in the recursive call in step 2, it has input and output with content  $b_d$ . Therefore, when honest  $P_i$  computes  $\text{majority}(b_j)$  in step 3 of  $\text{authLSP}$ , it sets the value  $b_i = b_d$ . On the other hand, since  $D$  is honest and uncompromised, the adversary cannot produce a valid message with content  $1 - b_d$  (recall that for a message to be valid, it must contain the signature of the dealer). It follows then that  $b_j \neq 1 - b_d$  for all values used to compute majority in step 3. Validity and agreement follow. ■

**Lemma 6** *If the dealer is honest and compromised, and  $t_h > t_a$ , then Protocol authLSP( $m$ ) achieves validity and agreement for all values of  $m$ .*

**Proof** It is easy to see that the lemma holds for  $m = 0$ . Let us assume the lemma holds for  $\text{authLSP}(m - 1)$ , and consider  $\text{authLSP}(m)$ . If an honest and uncompromised player is the dealer in step 2 of  $\text{authLSP}(m)$  (i.e. in the recursive call to  $\text{authLSP}(m - 1)$ ), then by Lemma 5 this run

achieves validity and agreement. If an honest but compromised player is the dealer in step 2, then it still holds in the recursive execution that  $t_h > t_a$ , since the dealer is not counted in  $t_h$ , and all other players participate in the execution of  $\text{authLSP}(m-1)$ ; by the induction hypothesis this execution achieves validity and agreement on output  $b_d$  as well. It follows that in step 3 of  $\text{authLSP}(m)$ , for each honest player  $P_i$ , at least  $n - t_a - 1$  of the  $b_j$  values equal  $b_d$  and at most  $t_a$  of the  $b_j$  values equal  $(1 - b_d)$ . Since  $n - t_a - 1 \geq t_h > t_a$ ,  $b_d$  is the majority value for each honest player, and the lemma follows. ■

We now prove the following theorem:

**Theorem 4** *For any value  $n \in \{2, 3, 4, 5, 6, 8, 9, 12\}$  there exists a protocol for  $n$  players 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. The remainder of the proof deals with  $n \in \{5, 6, 8, 9, 12\}$ .

**Lemma 7** *For  $n \in \{5, 6, 8\}$ ,  $\text{authLSP}(\lfloor \frac{n-1}{3} \rfloor + 1)$  achieves broadcast in the presence of a threshold adversary.*

**Proof** We prove the lemma by considering all possible types of dealers. We let  $b_d$  denote the input bit of the dealer  $D$ .

**D is honest and not compromised:** This case follows from lemma 5

**D is honest and compromised:** We break this case up into the following two scenarios:

$t_a \leq \lfloor \frac{n-1}{3} \rfloor$ : For  $n \in \{5, 6, 8\}$ , we have  $n > 2\lfloor \frac{n-1}{3} \rfloor + \lfloor \frac{n-1}{3} \rfloor + 1 \geq 2t_a + m$ , where the first inequality holds because of our assumption on  $n$ , and the second because of our assumption on  $t_a$ . Applying lemma 4 we get validity and agreement as claimed.

$t_a > \lfloor \frac{n-1}{3} \rfloor$ : Since we assume a threshold adversary, in this case we have  $t_h > t_a$  (cf. section 2). Applying lemma 6, agreement and validity follow.

**D is malicious:** Since we assume a threshold adversary, we have that  $n > 2t_a$ . We note that the malicious dealer is excluded from each of the executions of  $\text{authLSP}(\lfloor \frac{n-1}{3} \rfloor)$  in step 2, and therefore, of the  $n - 1$  players that participate in those executions, only  $t_a - 1$  are malicious. The reader can verify that for  $n \in \{5, 6, 8\}$ ,  $n - 1 > 3\lfloor \frac{n-1}{3} \rfloor$ , and that  $\lfloor \frac{n-1}{3} \rfloor \geq t_a - 1$  (recalling that  $t_a < \frac{n}{2}$ ). Applying lemma 3, we have agreement in each execution of  $\text{authLSP}(\lfloor \frac{n-1}{3} \rfloor)$  in step 2. Agreement in  $\text{authLSP}(\lfloor \frac{n-1}{3} \rfloor + 1)$  follows when the players compute their output in steps 3 and 4. ■

**Lemma 8** *For  $n \in \{9, 12\}$ , Protocol  $\text{authLSP}(\lfloor \frac{n-1}{3} \rfloor + 2)$  achieves broadcast in the presence of a threshold adversary.*

**Proof** We prove the lemma by considering all possible types of dealers.

**D is honest and uncompromised:** This case follows from lemma 5.

**D is honest and compromised:** We break this case up into the following two scenarios:

$t_a \leq \lfloor \frac{n-1}{3} \rfloor$ : For  $n \in \{9, 12\}$ , we have  $n > 2\lfloor \frac{n-1}{3} \rfloor + \lfloor \frac{n-1}{3} \rfloor + 2 \geq 2t_a + m$ , where the first inequality holds by our assumption on  $n$ , and the second inequality holds by our assumption on  $t_a$ . Applying lemma 4 we get validity and agreement as claimed.

$t_a > \lfloor \frac{n-1}{3} \rfloor$ : Because we assume a threshold adversary, we have  $t_h > t_a$  (cf. section 2). Applying lemma 6, agreement and validity follow.

**D is malicious:** We consider the recursive execution of  $\text{authLSP}(\lfloor \frac{n-1}{3} \rfloor + 1)$  in step 2, and prove agreement for each of the  $n-1$  dealers. When the dealer in step 2 is honest and uncompromised, by lemma 5 we have agreement in his execution. If the dealer is honest and compromised we consider two further possibilities. If  $t_a \leq \lfloor \frac{n-1}{3} \rfloor$ , then among the  $n-1$  players participating in this recursive execution, of which at most  $t_a - 1$  are malicious, we have

$$n-1 > 3\lfloor \frac{n-1}{3} \rfloor - 1 = 2\left(\lfloor \frac{n-1}{3} \rfloor - 1\right) + \left(\lfloor \frac{n-1}{3} \rfloor + 1\right) \geq 2(t_a - 1) + \left(\lfloor \frac{n-1}{3} \rfloor + 1\right)$$

By lemma 4, agreement follows. On the other hand, if the dealer is honest and compromised and  $t_a > \lfloor \frac{n-1}{3} \rfloor$ , then  $t_h > t_a$ , and by lemma 6 agreement follows. Finally, if the dealer in step 2 is malicious, we consider what happens in the *next* recursive step, namely when the players execute  $\text{authLSP}(\lfloor \frac{n-1}{3} \rfloor)$ . Now two malicious dealers have been excluded: both the dealer in  $\text{authLSP}(\lfloor \frac{n-1}{3} \rfloor + 2)$  and the dealer in  $\text{authLSP}(\lfloor \frac{n-1}{3} \rfloor + 1)$ . Noting that the maximum number of malicious players is 4 when  $n = 9$  and 5 when  $n = 12$  (because we have a threshold adversary), it follows that among the remaining  $n-2$  players,  $n-2 > 3(\lfloor \frac{n-1}{3} \rfloor)$ , and  $\lfloor \frac{n-1}{3} \rfloor \geq t_a - 2$ . Applying lemma 3, we have agreement for all dealer types in  $\text{authLSP}(\lfloor \frac{n-1}{3} \rfloor)$ , and agreement follows for all malicious dealers in the executions of  $\text{authLSP}(\lfloor \frac{n-1}{3} \rfloor + 1)$ . Since we have proven agreement for all dealer types in  $\text{authLSP}(\lfloor \frac{n-1}{3} \rfloor + 1)$ , we have agreement in the execution of  $\text{authLSP}(\lfloor \frac{n-1}{3} \rfloor + 2)$  as well. ■

This concludes the proof of Theorem 4. ■

**Note:** The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the US Army Research Laboratory, the US Government, the UK Ministry of Defense, or the UK Government. The US and UK Governments are authorized to reproduce and distribute reprints for Government purposes, notwithstanding any copyright notation herein.

## References

- [1] D. Agrawal, B. Archambeault, J. R. Rao, and P. Rohatgi. The EM side-channel(s). In *CHES '02: Revised Papers from the 4th International Workshop on Cryptographic Hardware and Embedded Systems*, pages 29–45, London, UK, 2003. Springer-Verlag.
- [2] D. Brumley and D. Boneh. Remote timing attacks are practical. *Computer Networks*, 48(5):701–716, 2005.
- [3] D. Dolev and H. R. Strong. Authenticated algorithms for Byzantine agreement. *SIAM J. Comput.*, 12(4):656–666, 1983.
- [4] A. Dornan. New viruses search for strong encryption keys. *PlanetIT Systems Management News*, March, 1999.

- [5] M. Fitzi, T. Holenstein, and J. Wullschlegler. Multi-party computation with hybrid security. In *EUROCRYPT*, pages 419–438, 2004.
- [6] O. Goldreich. *Foundations of Cryptography: Volume 2, Basic Applications*. Cambridge University Press, New York, NY, USA, 2004.
- [7] A. Gupta, P. Gopal, P. Bansal, and K. Srinathan. Authenticated Byzantine generals strike again. Cryptology ePrint Archive, Report 2008/287, 2008. <http://eprint.iacr.org/2008/297>.
- [8] J. Katz and Y. Lindell. *Introduction to Modern Cryptography (Chapman & Hall/Crc Cryptography and Network Security Series)*. Chapman & Hall/CRC, 2007.
- [9] P. C. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In *Advances in Cryptology — Crypto '96*, pages 104–113. Springer-Verlag, 1996.
- [10] P. C. Kocher, J. Jaffe, and B. Jun. Differential power analysis. In *Advances in Cryptology — Crypto '99*, pages 388–397. Springer-Verlag, 1999.
- [11] L. Lamport, R. E. Shostak, and M. C. Pease. The byzantine generals problem. *ACM Trans. Program. Lang. Syst.*, 4(3):382–401, 1982.
- [12] Ms00-008: Incorrect registry setting may allow cryptography key compromise. Microsoft Help and Support. <http://support.microsoft.com/kb/259496>.
- [13] P. Nguyen. Can we trust cryptographic software? Cryptographic flaws in GNU privacy guard v1.2.3. In *Eurocrypt 2004*.
- [14] R. Ostrovsky and M. Yung. How to withstand mobile virus attacks. In *ACM Symposium on Principles of Distributed Computing (PODC)*, pages 51–59, 1991.
- [15] M. C. Pease, R. E. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *J. ACM*, 27(2):228–234, 1980.
- [16] J. Quisquater and F. Koene. Side channel attacks: State of the art. Ecrypt Network of Excellence, 2002. [http://www.crypto.ruhr-uni-bochum.de/en\\_sclounge.html](http://www.crypto.ruhr-uni-bochum.de/en_sclounge.html).
- [17] J.-J. Quisquater and D. Samyde. Electromagnetic analysis (EMA): Measures and countermeasures for smart cards. In *E-SMART '01: Proceedings of the International Conference on Research in Smart Cards*, pages 200–210, London, UK, 2001. Springer-Verlag.

## A The Dolev-Strong Protocol

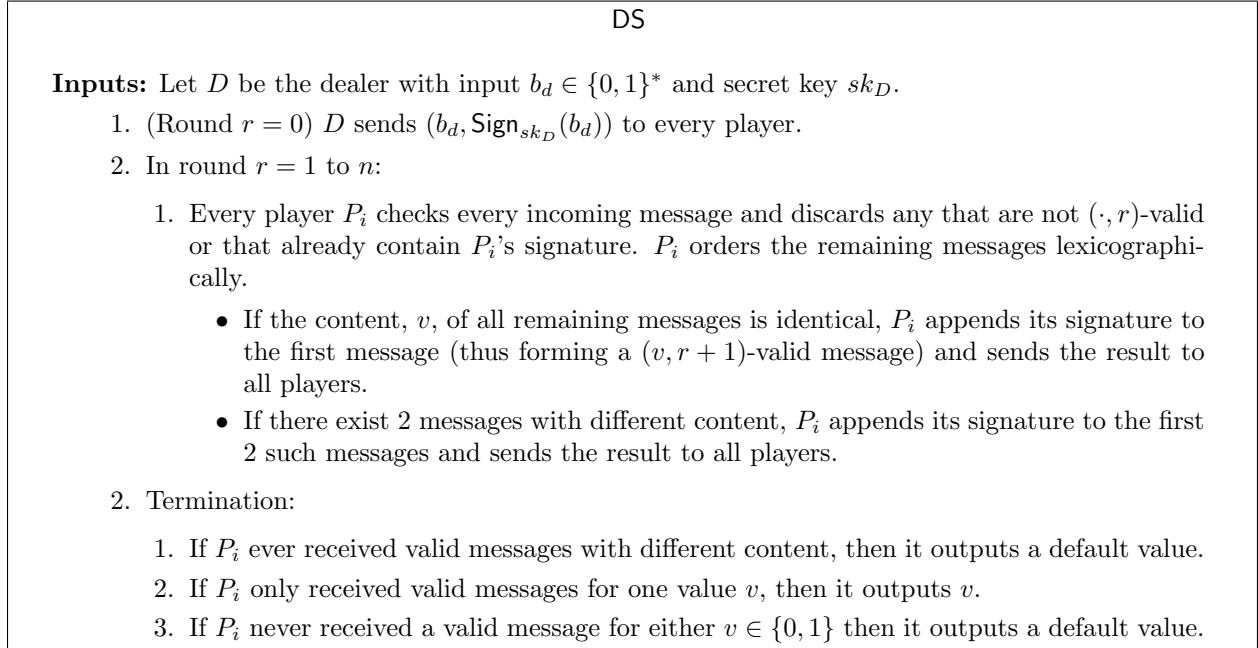


Figure 4: The Dolev-Strong protocol for broadcast.

For completeness, we present a modified version of the Dolev-Strong [3] protocol for authenticated broadcast. (See Figure 4.) A message  $M$  is called  $(v, i)$ -valid if it was received in round  $i$  and has the form  $(v, s_{P_1}, \dots, s_{P_i})$ , where  $P_1 = D$ , all  $P_j$ 's are distinct, and for every  $j = 1, \dots, i$  the string  $s_{P_j}$  is a valid signature on  $(v, s_{P_1}, \dots, s_{P_{j-1}})$  relative to the verification key of  $P_j$ . We refer to  $v$  as the *content* of a valid message. If the dealer is honest and uncompromised, there will only be  $(v, \star)$ -valid messages for a single value  $v$ , in which case the players consider the execution *clean*. Otherwise, the execution is called *dirty*. We note that the protocol differs from the original Dolev-Strong [3] protocol only in round complexity. We require  $n + 1$  rounds to ensure that all players agree whether the run was dirty; here we require that there is at least one honest and uncompromised player.

## B A More Efficient Protocol When $t_h > 0$ .

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 5, but it was based on the exponential algorithm of Lamport et al. [11] rather than the more efficient protocol of Dolev-Strong [3]. In this section, we deal with the “easier” case where there is guaranteed to be at least one honest, non-compromised party<sup>2</sup> (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. Appendix A), similar to the protocol we presented in Section 3.

<sup>2</sup>The difficulty that arises when  $t_a + t_c = n$  is that the compromised players may not agree on whether an execution of DS is clean or dirty (since Lemma 1 no longer holds). This is not a problem in Section 3 because, there, whenever  $t_a + t_c < n$ , the players run an unauthenticated broadcast protocol that does not use a PKI.

As in Section 3, our protocol begins with the dealer sending its input (here referred to as  $b_d$ ) to each player; each player then runs  $\text{DS}(b_d)$ . 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  $b_d$ . It is possible that  $t_h < t_a$ , and (recalling that the adversary can force all executions of  $\text{DS}$  by compromised players to be dirty) it is feasible for the adversary to force the majority of clean runs to have output  $1 - b_d$ .

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  $2t_a + \min\{t_a, t_c\} \geq n$ , exceeding the assumed threshold. Thus, when many dirty runs are detected the parties can switch to running a protocol that does not use the PKI at all. 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  $b_d$  is close to the number of clean runs resulting in  $1 - b_d$ , 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  $b_d$  to some non-faulty players and  $1 - b_d$  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  $b_d$ ; thus, to achieve the assumed balance, almost all the Byzantine players must run cleanly with output  $1 - b_d$ . If most of the Byzantine players 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  $b_d$ . The protocol is described fully in Figure 5.

**Theorem 5** *For  $n \in \{2, 3, 4, 5, 6, 8, 9, 12\}$ , Protocol 2 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  $\lfloor \frac{n-1}{3} \rfloor$  serves as a sort of breakpoint for the parameter  $t_a$ : if  $t_a \leq \lfloor \frac{n-1}{3} \rfloor$ , then (because  $\mathcal{A}$  is a threshold adversary) we have  $n > 3\lfloor \frac{n-1}{3} \rfloor \geq 3t_a$ ; when  $t_a > \lfloor \frac{n-1}{3} \rfloor$  then it holds that  $t_c < t_a$ . 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 9** *If  $D$  is honest and uncompromised then validity and agreement hold.*

**Proof** When the dealer is honest and uncompromised, then in step 2 the dealer’s execution of  $\text{DS}$  is clean and results in output  $b_d$  (cf. Lemma 1). Thus, all honest players terminate with output  $b_d$  in step 3. ■

**Lemma 10** *If  $D$  is honest but compromised then validity and agreement hold.*

## Protocol 2

**Inputs:** Let  $D$  be the dealer, with input  $b$ .

**Computation:**

1.  $D$  sends  $b_d$  to all other players.
2. Let  $b_i$  be the value received by  $P_i$  from  $D$  in step 1. (A missing value is taken as some default value.) In parallel, each  $P_i \in \mathcal{P}$  acts as the dealer in an execution of  $\text{DS}(b_i)$  (the original dealer  $D$  runs  $\text{DS}(b_d)$ ).
3. For  $b \in \{0, 1\}$ , let  $\text{CLEAN}_b$  denote the set of players whose execution of  $\text{DS}$  resulted in an output of  $b$ . Let  $\text{CLEAN} = \text{CLEAN}_0 \cup \text{CLEAN}_1$ , and let  $\text{DIRTY} = \mathcal{P} \setminus \text{CLEAN}$ . If there exists a  $b$  for which  $D \in \text{CLEAN}_b$ , then all players output  $b$  and terminate.
4. If  $|\text{CLEAN}| > 2\lfloor \frac{n-1}{3} \rfloor$  then output 0 if  $|\text{CLEAN}_0| \geq |\text{CLEAN}_1|$ , and output 1 otherwise.
5. If  $|\text{CLEAN}| < \lfloor \frac{n-1}{3} \rfloor + 2$ , then run any (unauthenticated) broadcast protocol for  $n > 3t_a$  that does not use a PKI, and terminate with the indicated output.
6. If there exists a  $b$  for which  $|\text{CLEAN}_b| > \lfloor \frac{n-1}{3} \rfloor$ , then output  $b$  and terminate.
7. Otherwise, we have  $\lfloor \frac{n-1}{3} \rfloor + 2 \leq |\text{CLEAN}| \leq 2\lfloor \frac{n-1}{3} \rfloor$ . Each player  $P_j \in \text{DIRTY}$  sends  $b_j$  to players in  $\text{CLEAN}$ . Each player  $P_i \in \text{CLEAN}_b$  sets value  $b'_i$  as follows:

$$b'_i = \begin{cases} b & \text{if at most } \lfloor \frac{n-1}{3} \rfloor - |\text{CLEAN}_{1-b}| \text{ players in DIRTY sent } 1-b \text{ to } P_i \\ \star & \text{otherwise} \end{cases}$$

8. Each player  $P_i \in \text{CLEAN}$  runs  $\text{DS}(b'_i)$ . For each  $P_i$  whose run is dirty, we add  $P_i$  to  $\text{DIRTY}$  and remove  $P_i$  from  $\text{CLEAN}$ . If the sets  $\text{CLEAN}$  and  $\text{DIRTY}$  were changed in this step, go back to step 4.
9. Let  $c_b$  denote the number of players in  $\text{CLEAN}_b$  that gave a clean run on  $b$  in step 8 and  $c_b^\star$  denote the number of players in  $\text{CLEAN}_b$  that gave a clean run on  $\star$  in step 8.
  1. If  $|\text{CLEAN}_0| + c_1^\star > \lfloor \frac{n-1}{3} \rfloor$  and  $|\text{CLEAN}_1| + c_0^\star > \lfloor \frac{n-1}{3} \rfloor$ , output a default value and terminate.
  2. Else if there exists a  $b$  for which  $|\text{CLEAN}_b| + c_{1-b}^\star > \lfloor \frac{n-1}{3} \rfloor$ , output  $b$  and terminate.
10. Run any (unauthenticated) protocol for  $n > 3t_a$  that ignores the PKI and terminate with the indicated output.

Figure 5: Broadcast against a threshold adversary, assuming  $t_a + t_c < n$ .

**Proof** We consider separately the cases  $t_a > \lfloor \frac{n-1}{3} \rfloor$  and  $t_a \leq \lfloor \frac{n-1}{3} \rfloor$ .

**Case 1:**  $t_a > \lfloor \frac{n-1}{3} \rfloor$ . By definition of a threshold adversary it follows that  $t_c < t_a$ . Letting  $t_h$  denote the number of honest and non-compromised players, we have  $t_a + t_c + t_h = n > 2t_a + t_c$  and so  $t_h > t_a > \lfloor \frac{n-1}{3} \rfloor$  or  $t_h \geq \lfloor \frac{n-1}{3} \rfloor + 2$ . Agreement and validity are immediate when the protocol terminates in step 3. If the protocol terminates at step 4, the majority answer is correct since  $t_h > t_a$  (by Lemma 1, only corrupt players will give clean runs with output  $1 - b_d$ , and all non-compromised honest players give clean runs with output  $b_d$ ). Termination in step 5 cannot happen, since  $t_h \geq \lfloor \frac{n-1}{3} \rfloor + 2$  and all non-compromised honest players give clean runs (i.e.,  $|\text{CLEAN}| \geq \lfloor \frac{n-1}{3} \rfloor + 2$ ). Finally, since the dealer is honest, all the non-compromised honest players will be in  $\text{CLEAN}_{b_d}$  and so  $|\text{CLEAN}_{b_d}| > \lfloor \frac{n-1}{3} \rfloor$ . It follows that termination is guaranteed by the end of step 6, and the majority value in that step is correct since  $t_h > t_a$ .



**Case 2:**  $t_a \leq \lfloor \frac{n-1}{3} \rfloor$ . Validity and agreement hold when the protocol terminates in step 4 since at most  $t_a$  corrupted players will give clean runs with output  $1 - b_d$ , and  $t_a \leq \lfloor \frac{n-1}{3} \rfloor$ . The same holds for termination in step 5, since  $n > 3 \lfloor \frac{n-1}{3} \rfloor \geq 3t_a$ . Termination after step 6 with output  $b$  implies that at least one honest player  $P_i$  gave a clean run on  $b$  (since  $t_a \leq \lfloor \frac{n-1}{3} \rfloor$ ). Since the dealer is honest, it follows that  $b = b_d$ . It remains to prove that termination in steps 9 or 10 always yields the correct output.

We next show that for  $b \neq b_d$ , the value of  $|\text{CLEAN}_b| + c_{1-b}^*$  is never greater than  $\lfloor \frac{n-1}{3} \rfloor$ .

**Claim 4** *When the dealer is honest, no honest players in  $\text{CLEAN}_{b_d}$  run  $\text{DS}(\star)$  in step 8.*

**Proof** When the dealer is honest and begins with input  $b_d$ , all the players in  $\text{CLEAN}_{1-b_d}$  are corrupt. Since  $t_a \leq \lfloor \frac{n-1}{3} \rfloor$ , the number of corrupt players in DIRTY is at most  $\lfloor \frac{n-1}{3} \rfloor - |\text{CLEAN}_{1-b_d}|$ . In step 7, therefore, the honest players in  $\text{CLEAN}_{b_d}$  never get more than  $\lfloor \frac{n-1}{3} \rfloor - |\text{CLEAN}_{1-b_d}|$  votes for  $1 - b_d$ . ■

Since honest players do not contribute to  $c_{b_d}^*$ , all the players contributing to  $c_{b_d}^*$  must be corrupt. Hence  $|\text{CLEAN}_{1-b_d}| + c_{b_d}^* \leq t_a \leq \lfloor \frac{n-1}{3} \rfloor$ . Therefore, termination in step 9 yields the correct answer. Finally if we end up running step 10 (i.e., the protocol for  $n > 3t_a$ ), we still obtain the correct output  $b_d$  since  $n > 3 \lfloor \frac{n-1}{3} \rfloor \geq 3t_a$ .

This completes the proof of Lemma 10. ■

**Lemma 11** *When  $D$  is corrupt, agreement holds.*

**Proof** We consider separately the cases  $t_a > \lfloor \frac{n-1}{3} \rfloor$  and  $t_a \leq \lfloor \frac{n-1}{3} \rfloor$ .

**Case 1:**  $t_a > \lfloor \frac{n-1}{3} \rfloor$ . Again letting  $t_h$  represent the number of honest and non-compromised players, recall that in this case  $t_h > t_a$ , implying  $t_h \geq \lfloor \frac{n-1}{3} \rfloor + 2$ . Agreement on termination after step 3 is obvious. Agreement in steps 4 or 6 follows from Lemma 1, since all honest players agree on whether a given run of DS was clean or dirty. We do not terminate in step 5 when  $t_a > \lfloor \frac{n-1}{3} \rfloor$ , since all honest players give clean runs resulting in  $|\text{CLEAN}| \geq t_h \geq \lfloor \frac{n-1}{3} \rfloor + 2$ .

Once the protocol proceeds past step 4, termination is possible either in step 9 or in step 10. Recall that the properties of DS ensure that all honest players agree on the sets  $\text{CLEAN}_0, \text{CLEAN}_1$ , and on the values  $c_0, c_1, c_0^*, c_1^*$ . Therefore agreement follows if we terminate at the end of step 9 rather than in step 10; we now prove that this is the case. By way of contradiction, suppose we do not terminate in step 9. We have  $|\text{CLEAN}_0| + c_1^* \leq \lfloor \frac{n-1}{3} \rfloor$  and  $|\text{CLEAN}_1| + c_0^* \leq \lfloor \frac{n-1}{3} \rfloor$ . Therefore,  $|\text{CLEAN}| + c_0^* + c_1^* \leq 2 \lfloor \frac{n-1}{3} \rfloor$ . We now examine what happens for all possible values of  $c_0^*$  and  $c_1^*$ . There are three sub-cases to consider.

**Sub-case 1:**  $c_0^* = c_1^* = 0$ . We prove that this case never happens. We know that  $t_h \geq \lfloor \frac{n-1}{3} \rfloor + 2$  and that  $|\text{CLEAN}_0|, |\text{CLEAN}_1| \leq \lfloor \frac{n-1}{3} \rfloor$  (because we did not terminate in step 6). Therefore, both  $\text{CLEAN}_0$  and  $\text{CLEAN}_1$  contain at least one non-compromised honest player. Since  $c_0^* = 0$ , some honest player in  $\text{CLEAN}_0$  did not output  $\star$  in step 8. By claim 5 (below), we have that all honest players in  $\text{CLEAN}_1$  output  $\star$  in step 8. This implies that  $c_1^* \neq 0$ , yielding a contradiction.

**Claim 5** *If there exists a non-compromised honest player  $P_i \in \text{CLEAN}_b$  that sets  $b'_i = b$ , then all the honest players  $P_j \in \text{CLEAN}_{1-b}$  set  $b'_j = \star$ .*

**Proof** Consider an honest player  $P_i \in \text{CLEAN}_b$ , that sets  $b'_i = b$ . Let DIRTY contain  $\alpha$  corrupt players and  $\beta$  honest but compromised players. Let  $\beta_b$  be the number of honest but compromised players who received bit  $b$  from  $D$  in the first round. Now  $\beta_{1-b} \leq \lfloor \frac{n-1}{3} \rfloor - |\text{CLEAN}_{1-b}|$ . Otherwise, it is easy to see that  $P_i$  would have set  $b'_i = \star$ . We claim that  $\beta_b > \lfloor \frac{n-1}{3} \rfloor - |\text{CLEAN}_b|$ . Suppose that is not true, then we have  $\beta = \beta_0 + \beta_1 \leq 2\lfloor \frac{n-1}{3} \rfloor - |\text{CLEAN}|$ , i.e.  $\beta + |\text{CLEAN}| \leq 2\lfloor \frac{n-1}{3} \rfloor$ . Since  $n = \alpha + \beta + |\text{CLEAN}|$ , we have  $n \leq \alpha + 2\lfloor \frac{n-1}{3} \rfloor \leq t_a + 2\lfloor \frac{n-1}{3} \rfloor$ . This is a contradiction since, we are given that  $n > 2\lfloor \frac{n-1}{3} \rfloor + \lfloor \frac{n-1}{2} \rfloor \geq 2\lfloor \frac{n-1}{3} \rfloor + t_a$ . Hence we have proved that  $\beta_b > \lfloor \frac{n-1}{3} \rfloor - |\text{CLEAN}_b|$  and thus all the honest players  $P_j \in \text{CLEAN}_{1-b}$  will set  $b'_j = \star$ . ■

**Sub-case 2:**  $c_0^*, c_1^* > 0$ . We prove that this case never happens for the values of  $n$  dealt with by our protocol (i.e., whenever  $n > \lfloor \frac{n-1}{2} \rfloor + 2\lfloor \frac{n-1}{3} \rfloor$ ). Recall that we assumed  $|\text{CLEAN}| + c_0^* + c_1^* \leq 2\lfloor \frac{n-1}{3} \rfloor$ . In this case we have  $|\text{CLEAN}| \leq 2\lfloor \frac{n-1}{3} \rfloor - 2$ . Note that  $|\text{CLEAN}| \geq \lfloor \frac{n-1}{3} \rfloor + 2$  or we would have terminated in step 5. This is only true for  $\lfloor \frac{n-1}{3} \rfloor \geq 4$  and no  $n \in \{2, 3, 4, 5, 6, 8, 9, 12\}$  satisfies  $\lfloor \frac{n-1}{3} \rfloor \geq 4$ .

**Sub-case 3:**  $\exists b$  such that  $c_{1-b}^* > 0, c_b^* = 0$ . Recall that we assumed  $|\text{CLEAN}| + c_0^* + c_1^* \leq 2\lfloor \frac{n-1}{3} \rfloor$ . Therefore,  $|\text{CLEAN}| \leq 2\lfloor \frac{n-1}{3} \rfloor - 1$ . Since  $|\text{CLEAN}| \geq \lfloor \frac{n-1}{3} \rfloor + 2$ , we have that  $\lfloor \frac{n-1}{3} \rfloor \geq 3$ . We already argued that  $\lfloor \frac{n-1}{3} \rfloor < 4$  for the values of  $n$  for which we claim correctness. We are left with  $\lfloor \frac{n-1}{3} \rfloor = 3$ , which implies that  $n = 12$  and so  $|\text{CLEAN}| = 5$ . Since  $t_a > \lfloor \frac{n-1}{3} \rfloor = 3$  and  $12 > 2t_a + \min(t_a, t_c) \geq 2t_a$ , the allowed values of  $t_a$  are 4 and 5. However when  $t_a = 5 > \lfloor \frac{n-1}{3} \rfloor$ , it follows that  $t_h \geq t_a + 1 = 6$ . This is impossible when  $|\text{CLEAN}| = 5$ , since all non-compromised honest players give clean runs. Hence we are left with  $t_a = 4$ . In this case,  $t_h \geq 5$ , and therefore, all players in  $\text{CLEAN}_0, \text{CLEAN}_1$  are honest and uncompromised (since  $|\text{CLEAN}| = 5$ ). Note that  $|\text{CLEAN}_0|, |\text{CLEAN}_1| \leq \lfloor \frac{n-1}{3} \rfloor = 3$ , otherwise, the protocol would have terminated at the end of step 6. Since  $|\text{CLEAN}| = 5$  we can conclude that both  $\text{CLEAN}_0$  and  $\text{CLEAN}_1$  are non-empty. Since  $\text{CLEAN}_0$  and  $\text{CLEAN}_1$  are non-empty and contain only non-compromised honest players, we claim that  $c_{1-b}^* = |\text{CLEAN}_{1-b}|$ . This is because  $c_b^* = 0$ , implying that a non-compromised honest player  $P_i \in \text{CLEAN}_b$  set  $b'_i = b$ , and by Claim 5, all honest players  $P_j \in \text{CLEAN}_{1-b}$  set  $b'_j = \star$ . Hence,  $|\text{CLEAN}_b| + c_{1-b}^* = |\text{CLEAN}_b| + |\text{CLEAN}_{1-b}| = |\text{CLEAN}| = 5 > \lfloor \frac{n-1}{3} \rfloor$ , and the protocol terminates in step 9.2.

This completes the analysis of Case 1.

**Case 2:**  $t_a \leq \lfloor \frac{n-1}{3} \rfloor$ : If  $t_a \leq \lfloor \frac{n-1}{3} \rfloor$ , then  $n > 3\lfloor \frac{n-1}{3} \rfloor > 3t_a$ . Agreement is guaranteed when the protocol terminates in steps 4 and 6 due to the properties of DS, and in step 5 since  $n > 3t_a$ . Now suppose the protocol terminates at step 9. Note that we have agreement on the sets  $\text{CLEAN}_0, \text{CLEAN}_1$  and on the values  $c_0, c_0^*, c_1, c_1^*$  due to DS. Since all decisions for termination in this step are made using the values  $|\text{CLEAN}_0|, |\text{CLEAN}_1|, c_0, c_0^*, c_1, c_1^*$ , we have agreement. Finally, if we terminate in step 10, we are guaranteed agreement since  $n > 3\lfloor \frac{n-1}{3} \rfloor > 3t_a$ . ■ ■