Byzantine Agreement, Made Trivial

Silvio Micali
CSAIL, MIT
Cambridge, MA 02139, USA
silvio@csail.mit.edu

April 14, 2017

Abstract

We present a very simple, cryptographic, Byzantine-agreement protocol that, with $n \geq 3t + 1 \geq 3$ players, at most t of which are malicious, halts in expected 9 rounds.

1 Introduction

Byzantine agreement is hugely important, and has attracted an enormous amount of attention over the course of multiple decades. Yet, somehow, the simple, digital-signature-based, solution contributed in these few pages appears to have eluded detection. (A possible explanation may lie in the fact that, although digital signatures have been used in Byzantine agreement for a long time, they could have, for the most part, been replaced by an abstract form of authentication by *fiat*. By contrast, the protocol presented here exploits their very *numerical values*.)

Our protocol is not only simple and theoretically attractive, but also extremely practical, despite the fact that the envisaged adversarial model is very strong.

Historical Note The notion of Byzantine agreement was introduced by Pease Shostak and Lamport [1] for the case of *binary* values. However, it was quickly extended to allow for arbitrary initial values. (See the surveys of Fischer [2] and Chor and Dwork [3].) Initially, BA protocols were deterministic. Probabilistic BA protocols were pioneered by Ben-Or in asynchronous settings [6].

2 Preliminaries

Cryptography We shall rely on two well known cryptographic tools, digital signatures and a hash function H. Both are idealized, for simplicity sake. Digital signatures are not existentially forgeable under adapting chosen message attacks and enjoy the following *uniqueness property*: for each public key and each message m, there is only one string that is accepted as the signature of m relative to that public key. The hash function H is modeled as a random oracle.

NOTE. We wish to stress that the result holds also via verifiable random functions, which can be constructed under concrete complexity assumptions [12]. (However, this introduces conceptual complexities that are quite orthogonal to the real contributions of this paper.)

Model

- *Communication.* The players communicate, in rounds, over a synchronous and complete point-to-point network. Messages are sent at the start of a round and received by the end of the round.
- *PKI.* Each player i has a public key, and a sufficiently long (e.g., 256-bit long) string R is selected randomly and independently of the players' public keys. The players, their public keys, and R are common knowledge.
- *The Adversary.* A player is *honest* and follows all his prescribed protocol instructions, until he becomes *malicious*, that is *corrupted* by the Adversary, a polynomial-time algorithm (unable to forge digital signatures), who initially knows all players, all public keys and R. The Adversary perfectly coordinates all malicious players. He takes all actions on their behalf, including receiving and sending all their messages, and can choose to have them deviate from their prescribed instructions in arbitrary ways. At the start of every round, the Adversary first chooses which additional players to corrupt, then learns all the messages sent by the currently honest players, and finally chooses the messages sent by the currently malicious players. The

Adversary cannot, however, tamper or interfere in any way with the messages sent by (currently) honest players. (Thus, the Adversary does not have the power of, after seeing the message sent by an honest player i in a given round, corrupt i, and modify any of i's messages in the round.)

The Adversary is a *t-adversary* if he cannot corrupt more than t players.

In this paper we show how to defeat adversaries who can corrupt less than a third of the players, and thus, for simplicity, we solely define Byzantine agreement protocols for such adversaries.

Byzantine agreement Protocols Let n and t be positive integers, such that $n \geq 3t + 1 \geq 3$. A protocol $\mathcal{P}$ is an *arbitrary-value* (respectively, *binary*) *(n, t)-Byzantine agreement (BA) protocol* with soundness $\sigma \in (0, 1]$, if, for every set of values V (respectively, for $V = \{0, 1\}$), and any t-adversary A, in an execution e of $\mathcal{P}$ with A with n players, in which every player i starts with an *initial value* $v_i \in V$, every honest player j halts with probability 1, outputting a value $out_j \in V$, so as to satisfy, with probability σ, the following two properties:

1. *Agreement:* $out_i = out_j$ for all honest players i and j.
2. *Consistency:* if, for some v, $v_i = v$ for all honest players i, then $out_j = v$ for all honest players j.

($\mathcal{P}$ *halts* when each honest player halts, and its *output* is v if, for some honest player i, $out_i = v$.)

Remark We stress that, in the antecedent of the Consistency property, a player i is honest if he is honest throughout the execution e. Thus our notion of consistency is more robust than the traditional one, which requires that the output of every honest player j is v only when v is the initial input of all *initially* honest (or even all) players i. Our BA protocols have soundness 1.

Notation

- Throughout this paper, the set of all players is N, the cardinality of N is n, and the upperbound to the number of malicious players is t, where $n \geq 3t + 1 \geq t$.
- We identify each player i with his public key. We denote i's digital signature of a string x by $sig_i(x)$. To ensure signers and messages are retrievable from digital signatures, we define

$$SIG_i(x) \triangleq (i, x, sig_i(x)).$$

- For all rounds r and players i and j, in a BA protocol i receives a single message from j, denoted by $m_i^r(j)$, either an element of V or the symbol $\perp \notin V$.[1]
- For each possible value v we denote by $\#_i^r(v)$ (or just $\#_i(v)$ when r is clear) the number of players from which i has received the value v.

[1] That is, we assume that a player sends exactly one message to another player in a round. Indeed, if i sent two different messages to j in a round r, then —say— $m_i^r(j)$ is assumed to be the lexicographically first element of V that j has received from i.

3 The Binary BA Protocol $BBA^\star$

Historical Note and Quick Announcement The binary BA protocol of this paper, $BBA^\star$, is a novel adaptations to our public-key setting of the probabilistic binary BA protocol of Feldman and Micali [9]. Their protocol was the first to work in an expected constant number of steps. It worked by having the players themselves implement a *common coin*, a notion proposed by Rabin, who implemented it via an external trusted party [7].

The protocol of [9] was not cryptographic and required a 2/3 honest majority. To the best of our knowledge, BA protocols with digital signatures were pioneered by Dolev and Strong [4]. To the best of our knowledge, Dolev and Strong [4] were the first to prove the existence of (n, t) BA protocol,, with $n \geq 2t + 1$, assuming digital signatures. A probabilistic, cryptographic version of the Feldman-Micali protocol, halting in expected constant number of rounds and tolerating any $t < n/2$ malicious players was discovered by Katz and Koo [10]. Their protocol is beautiful, but quite complex and not too practical.

A practical modification of $BBA^\star$ tolerating a t-adversary for any $t < n/2$, has been recently discovered by Vinod Vaikuntanathan and the author. Stay tuned!

All the above protocols have soundness 1.

3.1 The Intuition behind $BBA^\star$

Before our new binary (n, t)-BA protocol $BBA^\star$, we wish to build some intuition.

Consider executing, for $\gamma = 1, 2, \ldots$, the following 3-step protocol $\mathcal{P}(\gamma)$. Throughout these executions, each player i updates a retained binary variable b_i, representing his current opinion about what his final binary value out_i should be. (Initially, b_i is the initial binary value of player i.)

THE IDEALIZED PROTOCOL $\mathcal{P}(\gamma)$

1. *Every i sends b_i to all players, including himself.*
2. A new randomly and independently selected bit $c^{(\gamma)}$ magically appears in the sky, visible to all.
3. *Every player i updates b_i as follows.*

   3.1 If $\#_i^1(0) \geq 2t + 1$, then i (re)sets b_i to 0.

   3.2 Else, if $\#_i^1(1) \geq 2t + 1$, then i (re)sets b_i to 1.

   3.3 Else i (re)sets b_i to $c^{(\gamma)}$.

We refer to each such bit $c^{(\gamma)}$ as a *magic coin*.

QUICK ANALYSIS. Since we assume that at least $2t + 1$ players are honest, it is clear that

(A) *If, at the start of each protocol $\mathcal{P}(\gamma)$, the honest players are in agreement on a bit b (i.e., if $b_i = b$ for all honest player i), then they remain in agreement on b by its end.*

It is not not hard to see (and will be shown in our proof of Theorem 3.1 anyway) that

(B) *If the honest players are not in agreement (on any bit) at the start of $\mathcal{P}(\gamma)$, then, with probability 1/2, they will be in agreement (on some bit) by its end.*

Finally, it is also clear that

(C) *If all honest verifiers started with the same initial binary value b (i.e, if the honest players were originally in agreement on a bit b), then they will continue to agree on b.*

Properties A, B, and C guarantee that each $\mathcal{P}(\gamma)$ is a binary BA protocol with soundness 1/2.

Replacing Magic Coins with Frequently Magic Coins

Notice that, to bring the players into agreement with probability significant probability, it is not necessary that a magic coin is generated in the second step of $\mathcal{P}(\gamma)$. Trivially, if the bit $c^{(\gamma)}$ were always visible to all players, but only with probability 2/3 random and independent, then the players not already in agreement would be brought into agreement with probability 1/3.

Slightly less trivially, $\mathcal{P}(\gamma)$ would bring the players into agreement with probability 1/3 with an even less magic coin. Namely, it suffices that, in the second step of $\mathcal{P}(\gamma)$, each player i sees *his own* bit $c_i^{(\gamma)}$, provided that, with probability $> 2/3$,

E: *for all players i and j, $c_i^{(\gamma)} = c_j^{(\gamma)}$, where $c_i^{(\gamma)}$ is a random bit.*

With complementary probability, the bits $c_i^{(\gamma)}$ can be adversarially chosen.

We call such a bit-vector $\{c_i^{(\gamma)} : i \in N\}$ a *frequently magic coin.*

(Note that, when event E occurs, the players happen to see a "common and random coin", but they may not be aware that this is the case. By contrast, with a magic coin, it is common knowledge that the bit $c^{(\gamma)}$ is "random and common".)

The reason why a frequently magic coin suffices to bring the players into agreement with probability 1/3 is that, once the coin happens to be random and common, which happens with probability 2/3, it brings the palyers into agreement with probability 1/2. Also notice that, once the players are in agreement, no matter what the future frequently magic coins might be, the honest players remain in agreement, due to rules 3.1 and 3.2.

Replacing Frequently Magic Coins with Concrete Coins

Frequently magic coins are less magic, but magic nonetheless. Let us thus show that the players themselves, without any help from the "sky", can, by themselves (with digital signatures, random oracle H, and random string R), implement a frequently magic coin $c^{(\gamma)}$ in a single step.

Single-Step Protocol *ConcreteCoin(γ)*

Each player i

- *Sends the value $v_i \triangleq SIG_i(R, \gamma)$.*
- *Computes the first player, m, such that $H(v_m) \leq H(v_j)$ for all players j.*
- *Sets $c_i^{(\gamma)}$ to be the least significant bit of $H(v_m)$.*

Note that all the 256-bit strings $H(v_i)$ are random (because H is a random oracle) and independent (because, due to the retrievability property of digital signatures, $SIG_i(R, \gamma) \neq SIG_{i'}(R, \gamma')$ whenever $(i, \gamma) \neq (i', \gamma')$).

Also note that the value v_m is not totally random. For instance, since it is the minimum of n, random, 256-bit values, the first bit of v_m is very likely to be 0 when n is sufficiently large.

However, for each γ, the least significant bit of $H(v_m)$, $\texttt{lsb}(H(v_m))$, is, *ignoring negligible biases*, a random and independent bit, no matter whether n is large or small.

We refer to the so generated bit $\texttt{lsb}(H(v_m))$ as a *concrete coin*.

One Last Problem

Since each $\mathcal{P}(\gamma)$, executed with concrete coins, reaches Byzantine agreement with probability 1/3, one may consider executing $\mathcal{P}(1)$, $\mathcal{P}(2)$,..., until agreement is reached. The problem with this idea is that one would have to keep executing the protocols $\mathcal{P}(\gamma)$ for ever. In fact, when Byzantine agreement is reached, the honest players are not *aware* that this is the case, and thus cannot terminate.

Two Possible Solutions

A simple solution consists of executing $\mathcal{P}$ k times, where k is fixed and sufficiently high. For instance, $k = 300$. By doing so, however, one may waste rounds unnecessarily, since most of the time agreement will be reached early on.

A second solution, followed in protocol $BBA^\star$, consists of executing, for $\gamma = 1, 2, \ldots$ a protocol $\mathcal{P}'(\gamma)$ consisting of three *correlated* executions of $\mathcal{P}(\gamma)$.

More precisely, each $\mathcal{P}'(\gamma)$ consists of

1. A first execution of $\mathcal{P}(\gamma)$ with $c^{(\gamma)} = 0$ (I.e., with "the concrete coin is forced to be 0".)
2. A second execution of $\mathcal{P}(\gamma)$ with $c^{(\gamma)} = 1$. (I.e., with "the concrete coin is forced to be 1".)
3. A third execution of $\mathcal{P}(\gamma)$ with $c^{(\gamma)}$ being a "genuinely flipped" concrete coin.

As already mentioned, the third execution of $\mathcal{P}(\gamma)$ —and thus an execution of $\mathcal{P}'(\gamma)$— enables reaching agreement with probability 1/3. The first two executions of $\mathcal{P}(\gamma)$, as we shall see, ensure that, once agreement has already been reached on some bit b, an honest player i can learn that this is the case, and terminate the iteration with the binary output b.

3.1.1 Description and Analysis of $BBA^\star$

Protocol $BBA^\star$ is a 3-step loop, where the players repeatedly exchange Boolean values, and different players may exit this loop at different times. A player i exits this loop by *propagating* (i.e., by sending to all players, including himself), at some step, either a special value $0*$ or a special value $1*$, thereby instructing all players to "pretend" they respectively receive 0 and 1 from i in all future steps. (Alternatively said: assume that the last message received by a player j from another player i was a bit b. Then, in any step in which he does not receive any message from i, j acts as if i sent him the bit b.)

The protocol uses a counter γ, representing how many times its 3-step loop has been executed. At the start of $BBA^\star$, $\gamma = 0$. (One may think of γ as a global counter, but it is actually increased by each individual player every time that the loop is executed.)

PROTOCOL $BBA^\star$

(COMMUNICATION) STEP 1. [Coin-Fixed-To-0 Step] *Each player i propagates b_i.*

1.1 If $\#_i^1(0) \geq 2t+1$, then i sets $b_i = 0$, sends $0*$, outputs $out_i = 0$, and HALTS.

1.2 If $\#_i^1(1) \geq 2t+1$, then, then i sets $b_i = 1$.

1.3 Else, i sets $b_i = 0$.

(COMMUNICATION) STEP 2. [Coin-Fixed-To-1 Step] *Each player i propagates b_i.*

2.1 If $\#_i^2(1) \geq 2t+1$, then i sets $b_i = 1$, sends $1*$, outputs $out_i = 1$, and HALTS.

2.2 If $\#_i^2(0) \geq 2t+1$, then i set $b_i = 0$.

2.3 Else, i sets $b_i = 1$.

(COMMUNICATION) STEP 3. [Coin-Genuinely-Flipped Step] *Each player i propagates b_i and $SIG_i(R,\gamma)$.*

3.1 If $\#_i^3(0) \geq 2t+1$, then i sets $b_i = 0$.

3.2 Else, if $\#_i^3(1) \geq 2t+1$, then i sets $b_i = 1$.

3.3 Else, letting $S_i = \{j \in N : m_i^3(j) = SIG_j(R,\gamma)\}$,
i sets $b_i = c_i^{(\gamma)} \triangleq \mathtt{lsb}(\min_{j \in S_i} H(SIG_i(R,\gamma)))$; increases γ_i by 1; and returns to Step 1.

Theorem 3.1. *$BBA^\star$ is a binary (n,t)-BA protocol with soundness 1.*

Proof Sketch. We start by proving the following claims.

Claim $\mathcal{A}$: If, at the start of an execution of Step 3, no player has yet halted and agreement has not yet been reached, then, with probability at least 1/3, the players will be in agreement at the end of the step.

Proof of Claim $\mathcal{A}$. Let γ be the current value of the counter. By the uniqueness property of the underlying digital signature scheme, the n values $H(SIG_1(R,\gamma)), \ldots H(SIG_n(R,\gamma))$ are well defined, 256-bit long, random strings. Ignoring the remote possibility of ties, let $H(SIG_m(R,\gamma))$ be the minimum of these value. Then, with probability at least 2/3, m is a honest player. In this case, at the start of the Step 3 in question, player m sends $SIG_m(R,\gamma)$, and his signature is received by all players. Thus, when m is honest, whether or not the malicious players send their signatures, for all honest players i

$$H(SIG_m(R,\gamma)) = \min_{j \in S_i} H(SIG_i(R,\gamma)).$$

Define the bit $c^{(\gamma)}$ as follows:

$$c^{(\gamma)} \triangleq \mathtt{lsb}(H(SIG_m(R,\gamma))).$$

Then, ignoring negligible biases, when m is honest, $c^{(\gamma)}$ is a random bit and $c^{(\gamma)} = c_i^{(\gamma)}$ for all honest players i. Continuing to assume that m is honest, there are five exhaustive cases to consider: namely,

(i) All honest players i update their binary variables b_i according to 3.1.

In this case, Agreement holds on 0.

(ii) All honest players i update their binary variables b_i according to 3.2.

In this case, Agreement holds on 1.

(iii) All honest i update their binary variables b_i according to 3.3.

In this case, at the end of Step 3, Agreement holds on c.

(iv) Some honest i update their binary variables b_i according to 3.1, and all others according to 3.3.

Let $\mathcal{H}_0$ (respectively, $\mathcal{H}(1)$) be the set of honest players updating their bits according to 3.1 (respectively, 3.2). If $c^{(\gamma)} = 0$, then all players i in $\mathcal{H}(0)$ reset their binary variables b_i to 0, while all those in $\mathcal{H}(1)$ reset theirs to $c^{(\gamma)}$, which is 0. Since $c^{(\gamma)} = 0$ with probability 1/2, then in case (iii) Agreement on 0 is reached with probability 1/2.

(v) Some honest i update their binary variables b_i according to 3.2, and all others according to 3.3.

In this case, a symmetric argument shows that Agreement is reached on 1 with probability 1/2.

In sum, when m is honest, at the end of an execution of Step 3, no matter what malicious players might do, agreement is reached with probability at least 1/2. Thus, since the probability that m is honest is $> 2/3$, the probability that agreement is reached at the end of Step 3 is

$$> \tfrac{2}{3} \cdot \tfrac{1}{2} = 1/3. \quad \square$$

Claim $\mathcal{B}$: If, at some step, agreement holds on some bit b, then it continues to hold on the same bit b.

Proof of Claim $\mathcal{B}$. Assume that the players are in agreement on a bit b at the start of some step. Then in that step all honest players send b, so that, for every honest i, $\#_i(b) \geq 2t + 1$ and thus i sets $b_i = b$ at the end of the step. Thus, agreement holds ob b at the start of the next step. And so on. $\square$

Claim $\mathcal{C}$: If, at some step, a honest player i halts, then agreement will hold at the end of the step.

Proof of Claim $\mathcal{C}$. Assume first that a honest player i halts in Coin-Fixed-To-0 Step. Then, in that step, $\#_i(0) \geq 2t + 1$ and i sets $b_i = 0$. Since $\#_i(0) \geq 2t + 1$, at least $t + 1$ honest players send 0 at the start of the step. Thus, $\#_j(0) \geq t + 1$ for all other honest player j.

For each such player j two mutually exclusive cases may occur : (1) $\#_j(0) \geq 2t + 1$ and (2) $t + 1 \leq \#_j(0) < 2t + 1$. Accordingly, j sets $b_j = 0$ in substep 1.1 in the first case, and in substep 1.3 in the second. Thus Agreement holds on 0 at the end the Coin-Fixed-To-0 Step.

A symmetric argument shows that Agreement holds on 1 at the end of a Coin-Forced-To-1 Step in which a honest player halts. $\square$

Let us now prove our theorem. That is, let us prove that, in every execution of $BBA^\star$, in which there are $n \geq 3t + 1$ players and t malicious players, the following 3 properties hold.

0. All honest players halt with probability 1.

Before proving Property 0, let us note that in $BBA^\star$ agreement is reached with probability 1. (This trivially follows from the fact that, as already proven, if the players were not in agreement at the start of the main loop of $BBA^\star$, then they will be in agreement by the end of the loop with probability at least 1/3.)

To prove Property 0, assume first that agreement is reached on 0. Consider the first Coin-Fixed-To-0 Step at the start of which the players are in agreement on 0. Such a step must exists, because of property $\mathcal{B}$. Then, at the start of the step, all honest players send 0, including those who have already halted, because when they did halt they sendd $0*$, thereby virtually propagating 0 at every subsequent step. Accordingly, for all honest players j who have not yet halted, in substep 1.1 $\#_j(0) \geq 2t+1$ and j halts.

A symmetric argument shows that all honest players halt if agreement is reached on 1. □

1. $out_i = out_j$ for all honest players i and j.

   Property 1 holds because, by Property 0, all honest players halt, and whenever they do, by Claim $\mathcal{C}$, they are in agreement. □

2. If the initial value of every honest player is a bit b, then $out_i = b$ for all honest i.

   To prove Property 2, assume first that the initial bit of all honest players is 0. Then, in Substep 1.1 of the very first Coin-Fixed-To-0 step, $\#_i(0) \geq 2t+1$ for each honest player i; accordingly, i outputs $out_i = 0$ and halts.

   A symmetric and equally elementary reasoning shows that, if the initial bit of all honest players were 1, then they halt with output 1 in the very first Coin-Fixed-To-1 Step. □

This completes the proof of Theorem 3.1. ■

3.2 From Binary to Arbitray-Value BA

In [5], Turpin and Coan show a general reduction converting any *binary* Byzantine agreement protocol ba, tolerating at least $n/3$ malicious players, into an *arbitrary-value* BA protocol TC_{ba}, tolerating at most $n/3$ malicious players.

Their reduction calls ba as a subroutine in a black-box manner and introduces only two additional rounds. Once a player i computes his output, out_i, in ba, he computes his output, out_i, in TC_{ba} without any additional communication. Thus their reduction can also turn, with only two initial rounds, the binary BA protocol $BBA^\star$ into a corresponding arbitrary-value BA protocol, $BA^\star$.

References

[1] M. Pease, R. Shostak, and L. Lamport. *Reaching agreement in the presence of faults.* J. Assoc. Comput. Mach., 27 (1980), pp. 228-234.

[2] M. Fischer. *The consensus problem in unreliable distributed systems (a brief survey).* Proc. International Conference on Foundations of Computation, 1983.

[3] B. Chor and C. Dwork. *Randomization in Byzantine agreement, in Randomness and Computation.* S. Micali, ed., JAI Press, Greenwich, CT, 1989, pp. 433-498.

[4] D. Dolev and H.R. Strong. *Authenticated algorithms for Byzantine agreement.* SIAM Journal on Computing 12 (4), 656-666.

[5] R. Turpin and B. Coan. *Extending binary Byzantine agreement to multivalued Byzantine agreement.* Inform. Process. Lett., 18 (1984), pp. 73-76.

[6] M. Ben-Or. *Another advantage of free choice: Completely asynchronous agreement protocols.* Proc. 2nd Annual Symposium on Principles of Distributed Computing, ACM, New York, 1983, pp. 27-30.

[7] M. Rabin. *Randomized Byzantine generals.* Proc. 24th Annual IEEE Symposium on Foundations of Computer Science, IEEE Computer Society Press, Los Alamitos, CA, 1983. pp. 403-409.

[8] M. Castro and B. Liskov. *Practical Byzantine Fault Tolerance*, Proceedings of the Third Symposium on Operating Systems Design and Implementation. New Orleans, Louisiana, USA, 1999, pp. 173–186.

[9] P. Feldman and S. Micali. *18. An Optimal Probabilistic Algorithm for Synchronous Byzantine Agreement.* (Preliminary version in STOC 88.) SIAM J. on Computing, 1997.

[10] J. Katz and C-Y Koo *On Expected Constant-Round Protocols for Byzantine Agreement* `https://www.cs.umd.edu/ jkatz/papers/BA.pdf`

[11] N. Lynch. *Distributed Algorithms.* Morgan Kaufmann Publishers, 1996.

[12] S. Micali, M. Rabin, and S. Vadhan. *Verifiable Random Functions.* 40th Foundations of Computer Science (FOCS), New York, Oct 1999.