Deterministic Causal Order under Byzantine Sybil Tolerance: Techniques and Limitations

Ajay D. Kshemkalyani[1⋆][0000−0003−2451−7306] and Anshuman Misra[2][0000−0003−3987−7945]

[1] University of Illinois Chicago, Chicago, USA
ajay@uic.edu
[2] Purdue University Fort Wayne, Fort Wayne, USA
misra47@pfw.edu

Abstract. A spectrum of solvability and unsolvability results for causal ordering of messages in the presence of Byzantine processes in asynchronous systems for unicast, multicast, and broadcast modes of communication have been shown. The possibility results implicitly assumed that the number of Byzantine processes f was less than $n/3$, where n is the total number of processes in the system. In this paper, we extend these results for the same system assumptions and parameters – mode of communication (unicast/broadcast/multicast), strong safety, weak safety, and liveness, and use of cryptography, to systems with $f < n$. Thus, we show corresponding possibility and impossibility results for the highest degree of Byzantine fault-tolerance. We also give the best-known bounds on f for solvability of causal ordering using deterministic algorithms in synchronous systems under the same combinations of system assumptions and parameters as for asynchronous systems.

Keywords: Byzantine fault-tolerance · Causal Order · Broadcast · Causality · Asynchronous · Message Passing

1 Introduction

Causality is defined by the *happened before* [17] relation on the set of events, and by extension, on the set of messages. Causal ordering of messages is specified by the safety and liveness properties. The *strong safety* property requires that if message m_1 causally precedes m_2 and both are sent to p_i, then m_2 cannot be delivered before m_1 at p_i [2]. The *liveness* property requires that a message from a correct process to another correct process is eventually delivered. Causally related updates to data occur in a valid manner enforcing semantic correctness if causal ordering is enforced [16]. Applications of causal ordering include distributed data stores, fair resource allocation, and collaborative applications such as multiplayer online gaming, social networks, event notification systems, group editing of documents, and distributed virtual environments.

⋆ Corresponding author

A spectrum of solvability and unsolvability results for causal ordering in the presence of Byzantine processes for unicast, multicast, and broadcast modes of communication were shown in [22, 23, 25, 26]. The possibility results implicitly assumed that the number of Byzantine processes f was less than $n/3$, where n is the total number of processes in the system. This dependency arose because of the reliance on Bracha's Byzantine Reliable Broadcast (BRB) primitive [3, 4] in the possibility results. A weakening of the safety condition, termed *weak safety*, was also given in [22,23,25,26]. Weak safety requires that if m_1 causally precedes m_2 and there is a causal path from the send event of m_1 to the send event of m_2 passing through only correct processes in the execution, then m_2 should not be delivered before m_1 at all common destinations of m_1 and m_2. The possibility and impossibility results considered strong safety, weak safety, and liveness, as well as the optional use of cryptography. The results from [23,25,26] for systems satisfying $f < n/3$ are given in Table 1.

In this paper, we consider the highest degree of Byzantine behavior possible in a distributed system by allowing Byzantine processes to control up to all but one process in the system and assess the solvability of the spectrum of causal ordering problems. Typically, distributed algorithms solving problems such as consensus and agreement are subject to a bound, commonly $f < n/3$. Here, we extend the model by allowing Byzantine processes to not only control all but one process but also spawn and create new Byzantine processes, leading to a permissionless setting. Assuming a static bound on the number of Byzantine processes implicitly restricts the system to a permissioned model. Our system model, however, allows for maximal Byzantine behavior, also referred to as Byzantine Sybil Tolerance [9], and provides a critical analysis of causal ordering in permissionless systems, often overlooked in prior work. This relaxation of the traditional $f < n/3$ bound to $f < n$ encapsulates the notion of permissionless Byzantine Sybil tolerance, where the system must remain secure and functional despite an unbounded number of adversarial identities. In such systems, both f and as a result n can change over time; our results are valid as long as the corresponding relationship(s) between f and n hold, e.g., $f < n$. By permitting spawning of new Byzantine processes, this model captures a more generalized adversarial setting. Byzantine Sybil Tolerance has been mooted in [13] and is important in real-world use cases such as the Matrix system [12].

Contributions: Corresponding to the solvability and unsolvability results for various combinations of system assumptions shown in Table 1 for $f < n/3$ in asynchronous systems, we show results assuming only that $f < n$, thus considering systems with the highest degree of Byzantine fault-tolerance. Our results are summarized in Table 2. This study is significant because it is important to understand the limitations on what is solvable under the highest degree of Byzantine fault-resilience and Byzantine Sybil tolerance.

Of particular note on the negative side is the following result.

1. For broadcasts without cryptography, liveness cannot be guaranteed when $f < n$, whereas it could be guaranteed when $f < n/3$.

Of particular note on the positive side are the following results.

Mode of communication	SS + L	SS + L with cryptography	WS + L	WS + L with cryptography
Unicasts	No $\overline{SS}, \overline{L}$	No $\overline{SS}, L$	No $WS, \overline{L}$	Yes WS, L
Broadcasts	No $\overline{SS}, L$	No $\overline{SS}, L$	Yes WS, L	Yes WS, L
Multicasts	No $\overline{SS}, \overline{L}$	No $\overline{SS}, L$	No $WS, \overline{L}$	Yes WS, L

Table 1: Solvability of causal ordering using deterministic algorithms in asynchronous systems under different communication modes [23,25,26]. SS = strong safety, WS = weak safety, L = liveness, $\overline{SS}, \overline{WS}, \overline{L}$ represent that strong safety, weak safety, liveness, respectively, cannot be guaranteed. Results are assuming $3f < n$ as all "Yes" results require $3f < n$.

Mode of communic.	SS + L	SS + L with cryptography	WS + L	WS + L with cryptography
Unicasts	No, Th. 1 $\overline{SS}, \overline{L}$	No, Theorem 8 $\overline{SS}, L(f < n)$	No, Theorem 4 $WS(f < n), \overline{L}$	Yes, Corollary 1 $WS(f < n), L(f < n)$
Broadcasts	No, Th. 2 $\overline{SS}, \overline{L}$	No, Theorem 9 $\overline{SS}, L(f < n)$	No, Theorem 5 $WS(f < n), \overline{L}$	Yes, Corollary 2 $WS(f < n), L(f < n)$
Multicasts	No, Th. 3 $\overline{SS}, \overline{L}$	No, Theorem 7 $\overline{SS}, L(f < n)$	No, Theorem 6 $WS(f < n), \overline{L}$	Yes, Theorem 10 $WS(f < n), L(f < n)$

Table 2: Solvability of causal ordering using deterministic algorithms in asynchronous systems under different communication modes. SS = strong safety, WS = weak safety, L = liveness, $\overline{SS}, \overline{WS}, \overline{L}$ represent that strong safety, weak safety, liveness, respectively, cannot be guaranteed. Results are assuming $f < n$. For each entry in the table, the relation between f and n is explicitly indicated next to the respective property that is satisfiable.

1. Liveness is possible to be guaranteed with the use of cryptography when $f < n$, not just when $f < n/3$.
2. Weak safety can be guaranteed without the use of cryptography when $f < n$, not just when $f < n/3$.
3. Both weak safety and liveness can be guaranteed with the use of cryptography when $f < n$, not just when $f < n/3$.

The above results were for asynchronous systems. We also give the best-known bounds on f for solvability of causal ordering using deterministic algorithms in synchronous systems under the same combinations of system assumptions and parameters as for asynchronous systems. The results are summarized in Table 3.

In particular, on the negative side:

1. Strong safety cannot be guaranteed without the use of cryptography.

On the positive side, we have the following.

1. Strong safety and liveness using cryptography can be guaranteed for $f < n/2$.
2. Weak safety and liveness even without cryptography can be guaranteed for $f < n$.

Outline. Section 2 reviews related work. Section 3 gives the system model. Section 4 gives the main results about the solvability of Byzantine causal unicast, multicast, and broadcast in a deterministic manner in an asynchronous system. Section 5 gives the corresponding results for synchronous systems. Section 6 concludes.

2 Related Work

Algorithms for causal ordering of unicast and multicast messages in an asynchronous setting under a fault-free model have been proposed, e.g., in [6, 14, 15, 21, 30]. An algorithm for causal multicasts in a fault-free setting for mobile systems is given in [7]. There has been some work on causal broadcasts under various failure models. Causal ordering of broadcast messages under crash failures in asynchronous systems was introduced in [2]. This algorithm required each message to carry the entire set of messages in its causal past as control information. An algorithm for causally ordering broadcast messages – providing only a variant of weak safety and liveness – in an asynchronous system with Byzantine failures is proposed in [1]. The feasibility of solving Byzantine causal order for unicasts, multicasts, and broadcasts was analyzed in [23]. Previously, a *probabilistic* algorithm based on atomic (total order) broadcast and cryptography for secure causal atomic broadcast (liveness and strong safety) in an asynchronous system was proposed [5]. This logic used acknowledgements and effectively processed the atomic broadcasts serially. For the client-server configuration, two protocols for crash failures and a third for Byzantine failure of clients based on cryptography were proposed for secure causal atomic broadcast [10]. The third made assumptions on latency of messages, and hence works only in a synchronous system.

A spectrum of solvability and unsolvability results for causal ordering in the presence of Byzantine processes for unicast, multicast, and broadcast modes of communication were shown in [22, 23, 25, 26]. The results were shown for the regular or *strong safety* property, as well as for a weakening of the safety property, termed *weak safety*, defined alongside the above results. The possibility results implicitly assumed that the number of Byzantine processes f was less than $n/3$. This dependency arose because of the reliance on Bracha's Byzantine Reliable Broadcast (BRB) primitive [3] in the possibility results. The above results were for asynchronous systems; an analogous analysis and results for synchronous systems were presented in [28].

3 System Model

This paper deals with a distributed system having Byzantine processes which are processes that can misbehave [18, 29]. A correct process behaves exactly as specified by the algorithm whereas a Byzantine process may exhibit arbitrary behaviour by deviating arbitrarily from its protocol during the execution. This subsumes crash failures. A Byzantine process cannot impersonate another process. However, a Byzantine process may spawn other Byzantine processes and an unbounded number of Byzantine processes may join the system. This models a permissionless setting.

The distributed system is modeled as an undirected graph $\mathcal{G} = (P, C)$. Here P is the set of processes communicating asynchronously in the distributed system. The set P can change dynamically. Let n denote $|P|$; thus n can change over time. C is the set of FIFO (logical) communication links over which processes communicate by message passing. The communication links are reliable implying messages cannot get lost or be duplicated, and communication is authenticated.

When new processes are spawned and due to churn, there is a dynamically varying number of processes. Some of the causal ordering protocols do broadcast or multicast and we assume there is a mechanism in place by which processes learn of such view changes.

While stating and proving our solvability results, the system is first assumed to be asynchronous, i.e., there is no fixed upper bound δ on the message latency, nor any fixed upper bound ψ on the relative speeds of processors [11]. We then deal with solvability results for a synchronous system, which is defined as one in which both δ and ψ exist and are known [11].

Likewise our results also deal with the two cases: disallowing or allowing the use of cryptography. For results in asynchronous systems allowing the use of cryptography, we assume a public key infrastructure (PKI) with public and private keys, and group keys. We also assume an on-the-fly key creation and distribution mechanism to deal with a varying number of processes. For results in synchronous systems allowing the use of cryptography, we additionally assume a threshold encryption system.

Let e_i^x, where $x \geq 1$, denote the x-th event executed by process p_i. An event may be an internal event, a message send event, or a message receive event. Let the state of p_i after e_i^x be denoted s_i^x, where $x \geq 1$, and let s_i^0 be the initial state. The *execution* at p_i is the sequence of alternating events and resulting states, as $\langle s_i^0, e_i^1, s_i^1, e_i^2, s_i^2 \ldots \rangle$. The *execution history* at p_i is the finite execution at p_i up to the current or most recent or specified local state. The *happens before* [17] relation, denoted $\rightarrow$, is an irreflexive, asymmetric, and transitive partial order defined over events in a distributed execution that is used to define causality.

Definition 1. *The happens before relation on events consists of the following rules:*

1. ***Program Order**: For the sequence of events $\langle e_i^1, e_i^2, \ldots \rangle$ executed by process p_i, $\forall\ j, k$ such that $j < k$ we have $e_i^j \rightarrow e_i^k$.*

2. ***Message Order***: *If event e_i^x is a message send event executed at process p_i and e_j^y is the corresponding message receive event at process p_j, then $e_i^x \rightarrow e_j^y$.*
3. ***Transitive Order***: *If $e \rightarrow e' \wedge e' \rightarrow e''$ then $e \rightarrow e''$.*

Next, we define the partial order happens before relation $\rightarrow$ on the set of all application-level messages R [25, 26].

Definition 2. *The happens before relation $\rightarrow$ on messages in R consists of the following rules [25, 26]:*

1. *If p_i sent or delivered message m before sending message m', then $m \rightarrow m'$.*
2. *If $m \rightarrow m'$ and $m' \rightarrow m''$, then $m \rightarrow m''$.*

Definition 3. *The causal past of message m is denoted as $CP(m)$ and defined as the set of messages in R that causally precede message m under $\rightarrow$.*

To accommodate the possibility of Byzantine behaviour, we use a partial order on messages called *Byzantine happens before*, denoted as $\xrightarrow{B}$, defined on S, the set of all application-level messages that are both sent by and delivered at correct processes in P [25, 26].

Definition 4. *The Byzantine happens before relation $\xrightarrow{B}$ on messages in S consists of the following rules [25, 26]:*

1. *If p_i is a correct process and p_i sent or delivered message m (to/from another correct process) before sending message m' to a correct process, then $m \xrightarrow{B} m'$.*
2. *If $m \xrightarrow{B} m'$ and $m' \xrightarrow{B} m''$, then $m \xrightarrow{B} m''$.*

The Byzantine causal past of a message is defined as follows:

Definition 5. *The Byzantine causal past of message m, denoted as $BCP(m)$, is defined as the set of messages in S that causally precede message m under $\xrightarrow{B}$.*

We consider three possible modes of communication: multicast, unicast, and broadcast. In a multicast/unicast/broadcast, a message m is sent at a send event using send(m, G), send$(m, \{p_i\})$, send(m, P), respectively, and is delivered at a receive event via deliver(m).

Definition 6. *Byzantine Reliable Multicast (BRM) of message m to group G satisfies the following properties:*

1. *(Validity:) If a correct process p_i delivers message m from a correct process sender(m) sent to group G, then sender(m) must have executed send(m, G) and $p_i \in G$.*
2. *(Self-delivery:) If a correct process executes send(m, G), then it eventually delivers m. (Note, a group always contains the sender.)*

3. *(Agreement:) If a correct process delivers a message m from a possibly faulty process, then all correct processes in G will eventually deliver m.*
4. *(Integrity:) For any message m, a correct process p_i delivers m at most once.*
5. *(No Information Leakage:) No process outside the group G sees the content of m.*

It can be seen that Byzantine Reliable Unicast (BRU) and Byzantine Reliable Broadcast (BRB) are special cases of BRM. As a unicast has a single destination, the Agreement property of BRM goes away in BRU. As the destination set of a broadcast is the set of all processes, the No Information Leakage property of BRM goes away in BRB.

The correctness of Byzantine causal order multicast/unicast/broadcast is specified on $(R, \rightarrow)$ and $(S, \xrightarrow{B})$. The definitions of BCM/ BCU/ BCB need to be satisfied in addition to safety and liveness.

Definition 7. *(Strong safety and liveness:) A causal ordering algorithm for unicast/multicast/broadcast messages must ensure the following:*

1. ***Strong Safety:*** *$\forall m' \in CP(m)$ such that m' and m are sent to the same (correct) process, no correct process delivers m before m'.*
2. ***Liveness:*** *Each message sent by a correct process to another correct process will be eventually delivered.*

Definition 8. *(Weak safety and liveness:) A causal ordering algorithm for unicast/multicast/broadcast messages must ensure the following [23, 25, 26]:*

1. ***Weak Safety:*** *$\forall m' \in BCP(m)$ such that m' and m are sent to the same (correct) process, no correct process delivers m before m'.*
2. ***Liveness:*** *Each message sent by a correct process to another correct process will be eventually delivered.*

The goal is to satisfy both properties in the above definitions. A trivial but unacceptable solution to satisfy strong safety, but no liveness, is to never deliver a message. Likewise, a trivial but unacceptable solution to satisfy liveness, but no strong safety, is to simply deliver any message that is received. Ruling out such trivial but unacceptable solutions, in the sequel when we say that neither safety nor liveness can be guaranteed, or say that one of the two properties but not the other can be guaranteed, we mean using a non-trivial algorithm that attempts to satisfy both properties.

In our solvability results, we focus on only the strong safety or weak safety, and liveness properties. The Validity, Self-delivery, Agreement, Integrity, No Information Leakage properties follow in a straightforward manner.

4 Solvability Results

4.1 Strong Safety and Liveness without Cryptography

Theorem 1. *It is impossible to solve causal ordering (Definition 7) of unicast messages in an asynchronous message passing system with f Byzantine processes where $f < n$, as neither strong safety nor liveness is guaranteed.*

Proof. It was proved in [23, 25, 26] that strong safety cannot be satisfied in a system with even one Byzantine process. Hence it cannot be satisfied in a system with f Byzantine processes where $f < n$. It was also proved in [23, 25, 26] that liveness cannot be satisfied in a system with even one Byzantine process. Hence it cannot be satisfied in a system with f Byzantine processes where $f < n$. □

Theorem 2. *It is impossible to solve causal ordering (Definition 7) of broadcast messages in an asynchronous message passing system with f Byzantine processes where $f < n$, as neither strong safety nor liveness is guaranteed.*

Proof. It was proved in [23, 25, 26] that strong safety cannot be satisfied in a system with even one Byzantine process. Hence it cannot be satisfied in a system with f Byzantine processes where $f < n$.

Let a process p_a send a (supposed) broadcast message m that is received by p_b. p_b has two options. (1) p_b does a relay broadcast of m to all other processes as p_a may not have done a true broadcast but sent m selectively to some processes. This tries to ensure that each correct process p_c receives m directly and/or indirectly. But if p_b is Byzantine, it can relay broadcast fake messages by some supposed sender p_z, and this will cause p_c to receive and deliver fake messages. Hence this option cannot be used. (2) When p_b has to send its own broadcast m', it sends control information of all messages like m received since its own previous broadcast. When p_c receives m', it knows not to deliver m' until the causally preceding m is received and delivered by it. As the system is asynchronous, p_c cannot verify with p_a in bounded time whether p_a actually sent m via a broadcast and until then p_c will not be able to deliver m'. If p_a is Byzantine and it never included p_c as a destination of m, i.e., did not send m via a broadcast but sent it to p_b via a unicast, the message m' from a correct p_b to a correct p_c is never delivered, resulting in a liveness violation. □

In a multicast, a message is sent to a subset of processes and different send events can send to different multicast groups. We have the following result.

Theorem 3. *It is impossible to solve causal ordering (Definition 7) of multicast messages in an asynchronous message passing system with f Byzantine processes where $f < n$, as neither liveness nor strong safety is guaranteed.*

Proof. As neither strong safety nor liveness can be guaranteed for unicasts and broadcasts (Theorems 1, 2, respectively) — and unicasts and broadcasts are special cases of multicast — it follows that these guarantees cannot be provided for multicasts either. □

4.2 Weak Safety and Liveness without Cryptography

We now show a similar result to Theorem 1 with strong safety (Definition 7) defined in terms of the $\rightarrow$ relation replaced by weak safety (Definition 8) defined in terms of the $\xrightarrow{B}$ relation in the correctness criteria for causal ordering.

Theorem 4. *It is impossible to solve causal ordering (Definition 8) of unicast messages in an asynchronous message passing system with f Byzantine processes where $f < n$, as liveness cannot be guaranteed even though weak safety can be guaranteed.*

Proof. It was proved in [23,25,26] that liveness cannot be satisfied for the unicast mode of communication in a system with even one Byzantine process. Hence it cannot be satisfied in a system with f Byzantine processes where $f < n$.

Weak safety implies that in the space-time diagram of the execution, there is a path from a sender process p_a to a process p_c that goes through only correct processes. Along such a path, all processes including p_a and p_c are correct. It is straightforward to devise an algorithm in which causal dependencies on messages m sent by p_a (to their respective destinations) can be faithfully propagated along such paths as control information received on messages such as m' received by p_c. m belongs to $BCP(m')$, and if p_c is a destination of m as per the control information, p_c will wait for m to arrive and be delivered before delivering m'. m will certainly arrive as p_a is a correct process that must have sent m to p_c. Thus weak safety is guaranteed. □

Theorem 5. *It is impossible to solve causal ordering (Definition 8) of broadcast messages in an asynchronous message passing system with f Byzantine processes where $f < n$, as liveness cannot be guaranteed even though weak safety can be guaranteed.*

Proof. To prevent a liveness attack, it should not be possible for a Byzantine process p_b to insert a fake dependency on a fake message m sent (broadcast) by p_a. If p_b attempts to do so, it should be detectable by a correct process p_c so that p_c will not wait indefinitely to deliver a message m' carrying information about such a fake dependency. By sending broadcasts using Byzantine Reliable Broadcast (BRB) [3] which guarantees that exactly the same message is delivered to all or no correct process, p_c can wait for the arrival of m (if genuinely sent, it is guaranteed to eventually arrrive at p_c if it arrived at p_b) before delivering m'; if m does not arrive, the dependency information is fake and the sender of m' that included information of the dependency on m is Byzantine. So there is no liveness violation. This guarantee of no liveness attack depends on the ability to do broadcasts satisfying the Agreement property of BRB – and a necessary condition for BRB is that $f < n/3$ [3]. It is well-known that the Agreement property cannot be satisfied when $f \geq n/3$ unless cryptography is used [19,20]. As we are not allowing the use of cryptography in this theorem, when $f < n$, this necessary condition is not met and there is no way that a correct process p_c can verify that a reported causal dependency on a prior message is genuine. The prior message m may have been sent via a unicast by a Byzantine process p_a to correct process p_b. (Refer to the argument in the proof of Theorem 2.) Thus liveness attacks cannot be prevented when $f < n$.

The proof of weak safety being guaranteed in Theorem 4 for the unicast case applies almost identically for the guarantee of weak safety in this broadcast case. □

Theorem 6. *It is impossible to solve causal ordering (Definition 8) of multicast messages in an asynchronous message passing system with f Byzantine processes where $f < n$, as liveness cannot be guaranteed even though weak safety is guaranteed.*

Proof. Unicast and broadcast modes of communication are special cases of the multicast mode of communication. As it is not possible to guarantee liveness for unicasts (Theorem 4) and broadcasts (Theorem 5), it is not possible to guarantee liveness for multicasts.

The proof of weak safety being guaranteed in Theorem 4 for the unicast case applies almost identically for the guarantee of weak safety in the multicast case. □

4.3 Strong Safety and Liveness Using Cryptography

Here we use the simple idea of relay broadcasts alongside cryptography. In a *relay broadcast*, when a process receives a message m for the first time, it broadcasts m to other processes.

Theorem 7. *It is impossible to solve causal ordering (Definition 7) of multicast messages in an asynchronous message passing system with f Byzantine processes where $f < n$, even with the use of cryptography as strong safety cannot be guaranteed even though liveness can be guaranteed.*

Proof. The logic that causal ordering (Definition 7) cannot be solved for the multicast mode of communication is as follows.

It was proved in [23,25,26] that strong safety cannot be satisfied in a system with even one Byzantine process. Hence it cannot be satisfied in a system with f Byzantine processes where $f < n$.

To prevent fake causal dependencies on send event e_a^x of message m sent by process p_a, p_a has to sign m with its private key K_a^- so that any other process that gets a copy of m can verify (using decryption by p_a's public key K_a^+) that m was truly sent by p_a. However to maintain confidentiality within the group, the message needs to be encrypted. When p_a has to multicast a message m to group G at event e_a^x, it creates the ciphertext C_m by encrypting m with the group key K_G and sends $K_a^-(C_m, e_a^x, G)$ signed by the private key K_a^- to all the processes in the system. When a correct process p_c receives $K_a^-(C_m, e_a^x, G)$ (whether from a or via a relay broadcast by some other process) for the first time, p_c does a relay broadcast of $K_a^-(C_m, e_a^x, G)$ to all the processes in the system. It then applies $K_a^+(X)$ to verify that m is a legitimate message sent/originated by p_a. If m passes this test and $p_c \in G$, p_c decrypts C_m using K_G and a local application receive event e_c^y of m can occur if causal dependencies are satisfied.

In the next message $K_c^-(C_{m'}, e_c^w, G')$ sent (multicast) by p_c, control information of the form $\langle K_c^-(K_a^-(C_m, e_a^x, G), e_c^y)\rangle$ (and for similar other messages application-received since the last multicast by p_c) is also included.

When $K_c^-(C_{m'}, e_c^w, G')$ (along with its control information of the form $\langle K_c^-(K_a^-(C_m, e_a^x, G), e_c^y)\rangle$) is received by p_d, if $p_d \in G'$, p_d first delivers messages

(dependencies) of the form $K_a^-(C_m, e_a^x, G)$ in the control information if $p_d \in G$, before $C_{m'}$, i.e., m' is locally delivered. As all the dependencies can be verified as being genuine using the public key via $K_a^+(K_a^-(C_m, e_a^x, G))$, those messages C_m, i.e., m, must have been sent by p_a (directly or through relay broadcasts), and liveness is guaranteed.

Note that strong safety is not guaranteed, neither is a common order of delivery at the correct processes. However, a message from a correct process to another correct process will never be blocked waiting for fake causal dependencies to be satisfied, thus ensuring liveness. □

Theorem 8. *It is impossible to solve causal ordering (Definition 7) of unicast messages in an asynchronous message passing system with f Byzantine processes where $f < n$, even with the use of cryptography as strong safety cannot be guaranteed even though liveness can be guaranteed.*

Proof. The proof of Theorem 7 applies almost identically to unicasts with the observation that each multicast group is of size two – the sender and the receiver. □

Theorem 9. *It is impossible to solve causal ordering (Definition 7) of broadcast messages in an asynchronous message passing system with f Byzantine processes where $f < n$, even with the use of cryptography as strong safety cannot be guaranteed even though liveness can be guaranteed.*

Proof. The proof of Theorem 7 which is for multicast-based communication applies to the broadcast mode of communication which is a special case of group size equal to n. □

In the above three theorems, liveness can be guaranteed with $f < n$ because we simulate Byzantine Reliable Broadcast using cryptography and relay broadcasts, and overcome the limitation of $f < n/3$ without cryptography. BRB prevents equivocation, and we are able to meet all the specifications of BRB, and based on BRB, of BRM and BRU. For each point-to-point message sent in a multicast, there are up to n^2 point-to-point messages (or n broadcasts) sent.

4.4 Weak Safety and Liveness Using Cryptography

Theorem 10. *It is possible to solve causal ordering (Definition 8) of multicast messages in an asynchronous message passing system with f Byzantine processes where $f < n$, with the use of cryptography as weak safety and liveness can be guaranteed.*

Proof. Liveness can be guaranteed as shown in the proof of Theorem 7.

Weak safety can be guaranteed as shown in the proof of Theorem 6 (even without cryptography) – so the guarantee of weak safety holds with cryptography. Also, the algorithm described in the proof of Theorem 7 assuming cryptography guarantees weak safety because under the $\xrightarrow{B}$ relation, not just multicasts but also relay broadcasts by Byzantine processes are not considered. Relay

Mode of communic.	SS + L	SS + L with cryptography	WS + L	WS + L with cryptography
Unicasts	No, [28] $\overline{SS}, L(f < n)$	Yes, [24] and [28] $SS, L, f < n/2$	Yes, [22,25] and [28] $WS, L, f < n$	Yes, Section 5.2 $WS, L, f < n$
Broadcasts	No, [28] $\overline{SS}, L(f < n)$	Yes, [28] $SS, L, f < n/2$	Yes, [28] $WS, L, f < n$	Yes, Section 5.2 $WS, L, f < n$
Multicasts	No, [28] $\overline{SS}, L(f < n)$	Yes, [28] $SS, L, f < n/2$	Yes, [28] $WS, L, f < n$	Yes, Section 5.2 $WS, L, f < n$

Table 3: Best-known bounds on f for solvability of causal ordering using deterministic algorithms in synchronous systems under different communication modes. SS = strong safety, WS = weak safety, L = liveness, $\overline{SS}, \overline{WS}, \overline{L}$ represent that strong safety, weak safety, liveness, respectively, cannot be guaranteed. For each entry in the table, the relation between f and n is explicitly indicated for properties that are satisfiable.

broadcasts by a Byzantine process can cause messages between the same sender-receiver correct process pair to be delivered out-of-order by selectively doing relay broadcasts or violate safety by selectively suppressing dependencies on messages sent by correct processes in the control information in a relay broadcast. However, a message chain containing a relay broadcast by a Byzantine process that does application-message forwarding is not valid under the $\xrightarrow{B}$ relation used in the definition of weak safety. □

As unicasts and broadcasts are special cases of multicast, we have the following two results.

Corollary 1. *It is possible to solve causal ordering (Definition 8) of unicast messages in an asynchronous message passing system with f Byzantine processes where $f < n$, with the use of cryptography as weak safety and liveness can be guaranteed.*

Corollary 2. *It is possible to solve causal ordering (Definition 8) of broadcast messages in an asynchronous message passing system with f Byzantine processes where $f < n$, with the use of cryptography as weak safety and liveness can be guaranteed.*

5 Results for Synchronous Systems

Table 3 gives the best-known bounds on f for solvability of causal ordering using deterministic algorithms in synchronous systems under different communication modes for the same system assumptions and parameter values (strong safety/weak safety, liveness, with or without cryptography) as considered for asynchronous systems.

Rounds can be simulated in a synchronous system, where a process first sends messages within a round and then receives messages sent in that round. Thus all messages sent in a round are received by the end of that round. Rounds greatly simplify the design of synchronous distributed algorithms.

5.1 Strong Safety and Liveness without cryptography

As shown in [28], strong safety cannot be guaranteed with even a single Byzantine process when no cryptography is used.

Following the round-based cryptography-free algorithm in [28], liveness can be guaranteed with $f < n$ because all messages m' supposedly sent to a correct process in the causal past of message m would have been sent in earlier rounds than the round in which m is sent, and thus would have been delivered in an earlier round. Thus there is no need for m to wait on arrival at its destination for the arrival/delivery of m'. This result holds for multicasts, and by extension, to the special cases of unicast and broadcast.

5.2 Weak Safety and Liveness without Cryptography

The two round-free algorithms – Sender-Inhibition and Channel Sync – given for unicasts in [22, 25] guarantee weak safety and liveness, and work for $f < n$. The round-based cryptography-free algorithm for multicasts (and by extension, for the special cases of unicasts and broadcasts) given in [28], guarantees weak safety and liveness as shown in [28], and as per the logic for liveness outlined in Section 5.1.

5.3 Strong Safety and Liveness with Cryptography

The algorithm for unicasts given in [24] uses threshold encryption to guarantee strong safety and liveness. Threshold encryption requires $f < n/2$.

The cryptography-based algorithm in [28] for strong safety and liveness for multicasts (and for its special cases of unicasts and broadcasts) uses Byzantine Reliable Broadcast (BRB) in conjunction with threshold encryption. BRB requires that $f < n/3$ whereas threshold encryption requires that $f < n/2$. Thus the effective bound is $f < n/3$. In this algorithm, we propose replacing the BRB algorithm of Bracha by a module that implements Dolev-Strong authenticated Byzantine Agreement in synchronous systems [8], as done in [27]. Instead of doing a BRB broadcast of m, Dolev-Strong authenticated agreement is reached on m in exactly $f + 1$ rounds, tolerating up to $n - 1$ Byzantine processes, i.e., $f < n$. If the agreement value is a non-null message, then the m is processed further and the corresponding decryption shares are sent as per the rest of the algorithm in [28]; otherwise if the agreement value is the null message then m is ignored. Dolev-Strong algorithm authenticated Byzantine agreement requires $f < n$ whereas threshold encryption requires $f < n/2$. Thus the effective bound is $f < n/2$.

5.4 Weak Safety and Liveness with Cryptography

Weak safety and liveness can be guaranteed (for unicasts, multicasts, and broadcasts) in synchronous systems without the use of cryptography, for $f < n$ (see Section 5.2). So the result extends to systems with the use of cryptography.

6 Conclusions

Corresponding to the solvability and unsolvability results for various combinations of system assumptions shown in Table 1 for $f < n/3$ in asynchronous systems in [26], we showed results assuming only that $f < n$, thus considering systems with the highest degree of Byzantine fault-tolerance. The results for asynchronous systems are summarized in Table 2. We also gave the best-known bounds on f for solvability of causal ordering using deterministic algorithms in synchronous systems under the same combinations of system assumptions and parameters as for asynchronous systems. The results for synchronous systems are summarized in Table 3.

This study is significant because it is important to understand the limitations on what is solvable under the highest degree of Byzantine fault-resilience. In particular, when $f < n$ the results indicate what is solvable under Byzantine Sybil tolerance in peer-to-peer systems.

Some of our solvability results pertain to weak safety when $f < n$. We note that in practice, it is not possible to determine whether $m \xrightarrow{B} m'$ holds because this requires identifying the correct processes and the Byzantine processes. Therefore it is not possible to determine whether weak safety actually holds. The solvability results simply say that if weak safety holds then causal order can be satisfied.

Acknowledgements. We thank Hannes Hartenstein and Florian Jacob for suggesting the problem of studying Byzantine Sybil fault-tolerance for causal ordering. The paper has benefited from several discussions with Hannes and Florian.

References

1. Auvolat, A., Frey, D., Raynal, M., Taïani, F.: Byzantine-tolerant causal broadcast. Theoretical Computer Science **885**, 55–68 (2021)
2. Birman, K.P., Joseph, T.A.: Reliable communication in the presence of failures. ACM Transactions on Computer Systems **5**(1), 47–76 (1987)
3. Bracha, G.: Asynchronous byzantine agreement protocols. Information and Computation **75**(2), 130–143 (1987)
4. Bracha, G., Toueg, S.: Asynchronous consensus and broadcast protocols. J. ACM **32**(4), 824–840 (Oct 1985). https://doi.org/10.1145/4221.214134
5. Cachin, C., Kursawe, K., Petzold, F., Shoup, V.: Secure and efficient asynchronous broadcast protocols. IACR Cryptol. ePrint Arch. p. 6 (2001), http://eprint.iacr.org/2001/006

6. Chandra, P., Gambhire, P., Kshemkalyani, A.D.: Performance of the optimal causal multicast algorithm: A statistical analysis. IEEE Trans. Parallel Distributed Syst. **15**(1), 40–52 (2004), https://doi.org/10.1109/TPDS.2004.1264784
7. Chandra, P., Kshemkalyani, A.D.: Causal multicast in mobile networks. In: 12th International Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS). pp. 213–220 (2004), https://doi.org/10.1109/MASCOT.2004.1348235
8. Dolev, D., Strong, H.R.: Authenticated algorithms for byzantine agreement. SIAM J. Comput. **12**(4), 656–666 (1983), https://doi.org/10.1137/0212045
9. Douceur, J.R.: The sybil attack. In: Revised Papers from the First International Workshop on Peer-to-Peer Systems. p. 251–260. IPTPS '01, Springer-Verlag, Berlin, Heidelberg (2002)
10. Duan, S., Reiter, M.K., Zhang, H.: Secure causal atomic broadcast, revisited. In: 2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN). pp. 61–72. IEEE (2017)
11. Dwork, C., Lynch, N.A., Stockmeyer, L.J.: Consensus in the presence of partial synchrony. J. ACM **35**(2), 288–323 (1988), http://doi.acm.org/10.1145/42282.42283
12. Jacob, F., Beer, C., Henze, N., Hartenstein, H.: Analysis of the matrix event graph replicated data type. IEEE Access **9**, 28317–28333 (2021), https://doi.org/10.1109/ACCESS.2021.3058576
13. Kleppmann, M., Howard, H.: Byzantine eventual consistency and the fundamental limits of peer-to-peer databases. arXiv preprint arXiv:2012.00472 (2020)
14. Kshemkalyani, A.D., Singhal, M.: An optimal algorithm for generalized causal message ordering (abstract). In: Proceedings of the Fifteenth Annual ACM Symposium on Principles of Distributed Computing (PODC). p. 87. ACM (1996), https://doi.org/10.1145/248052.248064
15. Kshemkalyani, A.D., Singhal, M.: Necessary and sufficient conditions on information for causal message ordering and their optimal implementation. Distributed Comput. **11**(2), 91–111 (1998), https://doi.org/10.1007/s004460050044
16. Kshemkalyani, A.D., Singhal, M.: Distributed Computing: Principles, Algorithms, and Systems. Cambridge University Press (2011), https://doi.org/10.1017/CBO9780511805318
17. Lamport, L.: Time, clocks, and the ordering of events in a distributed system. Commun. ACM 21, 7 pp. 558–565 (1978)
18. Lamport, L., Shostak, R.E., Pease, M.C.: The byzantine generals problem. ACM Trans. Program. Lang. Syst. **4**(3), 382–401 (1982), http://doi.acm.org/10.1145/357172.357176
19. Malkhi, D., Merritt, M., Rodeh, O.: Secure reliable multicast protocols in a WAN. In: Proceedings of the 17th International Conference on Distributed Computing Systems. pp. 87–94 (1997)
20. Malkhi, D., Reiter, M.K.: A high-throughput secure reliable multicast protocol. J. Comput. Secur. **5**(2), 113–128 (1997)
21. Mattern, F., Fünfrocken, S.: A non-blocking lightweight implementation of causal order message delivery. In: Birman, K.P., Mattern, F., Schiper, A. (eds.) Theory and Practice in Distributed Systems, International Workshop. Lecture Notes in Computer Science, vol. 938, pp. 197–213. Springer (1994), https://doi.org/10.1007/3-540-60042-6_14
22. Misra, A., Kshemkalyani, A.D.: Causal ordering in the presence of byzantine processes. In: 28th IEEE International Conference on Parallel and Distributed Sys-

tems. pp. 130–138. IEEE (2022), https://doi.org/10.1109/ICPADS56603.2022.00025
23. Misra, A., Kshemkalyani, A.D.: Solvability of byzantine fault-tolerant causal ordering problems. In: Koulali, M., Mezini, M. (eds.) 10th International Conference on Networked Systems. Lecture Notes in Computer Science, vol. 13464, pp. 87–103. Springer (2022), https://doi.org/10.1007/978-3-031-17436-0_7
24. Misra, A., Kshemkalyani, A.D.: Byzantine fault-tolerant causal order satisfying strong safety. In: Proceedings of the 25th International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS). pp. 111–125 (2023), https://doi.org/10.1007/978-3-031-44274-2_10
25. Misra, A., Kshemkalyani, A.D.: Byzantine fault-tolerant causal ordering. In: 24th International Conference on Distributed Computing and Networking. pp. 100–109. ACM (2023), https://doi.org/10.1145/3571306.3571395
26. Misra, A., Kshemkalyani, A.D.: Byzantine-tolerant causal ordering for unicasts, multicasts, and broadcasts. IEEE Trans. Parallel Distributed Syst. **35**(5), 814–828 (2024), https://doi.org/10.1109/TPDS.2024.3368280
27. Misra, A., Kshemkalyani, A.D.: Detecting causality in the presence of byzantine processes: The case of synchronous systems. Inf. Comput. **301**, 105212 (2024), https://doi.org/10.1016/j.ic.2024.105212
28. Misra, A., Kshemkalyani, A.D.: Solvabiity of byzantine fault-tolerant causal ordering: Synchronous systems case. In: Proceedings of the 39th ACM Symposium on Applied Computing (SAC). pp. 251–256 (2024), https://doi.org/10.1145/3605098.3636063
29. Pease, M.C., Shostak, R.E., Lamport, L.: Reaching agreement in the presence of faults. J. ACM **27**(2), 228–234 (1980), http://doi.acm.org/10.1145/322186.322188
30. Raynal, M., Schiper, A., Toueg, S.: The causal ordering abstraction and a simple way to implement it. Information processing letters **39**(6), 343–350 (1991)