Asynchronous Secure Communication Tolerating Mixed Adversaries

K. Srinathan[1]*, M. V. N. Ashwin Kumar[2], and C. Pandu Rangan[1] **

[1] Department of Computer Science and Engineering,
Indian Institute of Technology, Madras,
Chennai-600036, INDIA.
ksrinath@cs.iitm.ernet.in,rangan@iitm.ernet.in
[2] Department of Computer Science,
Cornell University, Ithaca, New York.
mvnak@cs.cornell.edu

Abstract. We study the problem of secure communication tolerating generalized mixed adversaries across an underlying completely asynchronous incomplete network. We explore the interplay between the minimal network connectivity required and the degree of security attainable, and completely characterize the network requirements for attaining perfect and unconditional (with negligible error) security. We also consider networks with additional broadcast capabilities and prove that unconditionally secure communication can be achieved with much lesser connectivity if the network assures the broadcast primitive.

1 Introduction

Consider n players who are connected by an underlying communication network $\mathcal{N}$. Our concern is to make sure that every player can *talk* to every other player. Two players can *talk* to each other if they are connected by an edge. Hence, we can trivially guarantee that every player can *talk* to every other player if the underlying network $\mathcal{N}$ is complete. But do we require all the ${}^nC_2$ direct connections (or edges)? Can we not ensure that all the players can communicate with one another with a lesser number of edges? Evidently, the smallest connected network (viz. a tree) would suffice to allow every pair of players to be able to *talk* (though indirectly). However, such minimal connectivity is not enough if one player wants to *secretly talk* to another player, i.e., the sender **S** has to transmit a message to the receiver **R** such that all the other players should get no information about the message transmitted, even if some non-trivial subset of players (excluding **S** and **R**) collude and behave maliciously. The interplay between information-theoretically secure communication and minimal network connectivity has been studied extensively. Dolev et. al. in [5] proved that a *synchronous* network has to be at least $(\max(t_a, t_p) + t_a + 1)$-connected for secure

* Financial support from Infosys Technologies Limited, India, is acknowledged.
** Partially supported by DRDO collaborative project on Communication and Networking Technologies.

message transmission to be guaranteed between every two players, where up to t_p players collude and only eavesdrop on the messages routed through them (passive faults), while up to t_a players collude and maliciously try to disrupt the protocol apart from eavesdropping (active or Byzantine faults). These result were recently generalized in [12] to the non-threshold case modeling only Byzantine faults: perfectly secure transmission is possible if and only if the union of the players in no two potentially faulty subsets of players forms a vertex cut-set of the graph (wherein the potentially faulty subsets of players are enumerated as an *adversary structure*[9]). However, these results are restricted to the case where the underlying network is synchronous. Asynchronous perfectly secure communication for the case of threshold adversaries was studied in [13]. In essence, a $(\max(t_a, t_p) + 2t_a + 1)$-connected network is necessary and sufficient. Unconditionally secure communication with negligible error in reliability (i.e., **R** might receive a wrong message) was studied in detail in [8]. However, these results are restricted to the threshold case assuming that the underlying network is synchronous. We initiate a study of asynchronous secure communication tolerating faulty players, some passive and some others active, characterized as generalized mixed adversary structures (as in [7]) and investigate the minimal connectivity requirements for perfect security and unconditional security.

A very important primitive used by all protocols for secure communication is that of *reliable, yet insecure* communication (e.g. [5] calls it public transmission). By this we mean, any message m sent by player **S** to player **R** is *correctly* received by **R**; however, the other players may have considerable (or even full) information about m. We can achieve this *reliable* communication trivially if we are assured that the network $\mathcal{N}$ has *broadcast* capability.[3] However, if $\mathcal{N}$ does not have broadcast capabilities, one should be able to simulate reliable transmission using a protocol. We study the minimum network connectivity which guarantees the possibility of such reliable transmission in networks without broadcast capabilities. We show that the existence of a broadcast channel does *not* reduce the connectivity requirement for perfectly secure asynchronous communication.

In line with [7], the generalized mixed adversary is characterized by a generalized adversary structure (see Definition 1), i.e. a set of pairs (D, E), where D and E are disjoint subsets of the set of players, wherein the adversary may select one arbitrary pair from the structure and corrupt the players in D actively (i.e. take full control) and in addition passively corrupt (i.e. read and process information of) the players in E. Among our results, we show that in the perfect setting, secure message transmission between any pair of (honest) nodes in a completely asynchronous network is possible if and only if neither the removal of the players in the union of any *three* sets of potential active collusions, nor the removal of the players in the union of any *two* sets of potential active collusions with any *one* corresponding set of potential passive collusion, leaves the network *disconnected*. Evidently, the above condition generalizes the threshold adversary requirement of $(\max(t_a, t_p) + 2t_a + 1)$-connected network. Interestingly, we prove

[3] By definition, if a message m is sent using a *broadcast* channel then *all* the players correctly receive (the *same*) m.

that the same condition given above is necessary and sufficient even in the case of unconditional security, though the resultant protocol is less complex. Thus, the minimal connectivity requirement is unaffected by he weakening of security. However, in the presence of a broadcast channel, the perfect setting *still* requires the same amount of connectivity whereas in the unconditional setting, it is (necessary and) sufficient if the players in the union of any *two* sets of potential active collusions with any *one* corresponding set of potential passive do not form a *vertex cut-set* of the network (i.e., only the second half of the above condition is sufficient). In all the above cases, the designed protocols have both their computation and communication complexities polynomial in the size of the maximal basis of the adversary structure.

Motivated by the facts that network synchrony is hard to achieve and that a threshold adversarial model is insufficient to model all types of mutual (dis)trust, we generalize the results of [5, 13, 12] to the generalized mixed adversary model and/or to asynchronous networks (see Theorem 7), in the perfect setting. Furthermore, in the unconditional setting, we initiate the study of asynchronous secure communication (see Theorem 5) and the study of unconditional with broadcast capability model (see Theorem 3).

Asynchronous secure communication is an important primitive for secure multiparty computation over asynchronous incomplete networks. Thus, our results can be used to transform the asynchronous secure computation protocols that run over a complete network (e.g. [2, 3]) into ones that can be executed over incomplete networks.

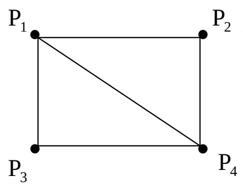

Fig. 1. Network.

The usefulness of some of our results is illustrated, for instance, through the following implication: Consider a asynchronous chorded ring network of four players as shown in Fig. 1. The most powerful adversary that previous known protocols (e.g. [13]) for perfectly secure communication among the players over the asynchronous network in Fig. 1 could tolerate is one that *passively* corrupts *one* arbitrary player (since the chordal ring network is 2-connected, we require that $2 > \max(t_a, t_p) + 2t_a$, giving $t_a = 0$ and $t_p = 1$). Using our results, one can perfectly tolerate an adversary that passively corrupts player P_1 or player P_4 or (even) *actively* corrupts player P_2 or player P_3.

2 Preliminaries

We consider a network $\mathcal{N}(\mathcal{P}, \mathcal{E})$, where $\mathcal{P} = \{P_1, P_2, \ldots, P_n\}$ denote the set of players (nodes) in the network that are connected by the edges as defined by $\mathcal{E} \subseteq \mathcal{P} \times \mathcal{P}$. Formally, all the n players (nodes) in the network $\mathcal{N}$ can be modeled as probabilistic interactive Turing Machines. We assume that randomization is achieved through random bits. We assume that the underlying network $\mathcal{N}$ is *asynchronous*, i.e., a message sent on a channel/path can be arbitrarily delayed (similar to the communication model in [6]). However, if two nodes P_i and P_j

are directly connected by a link (edge), then each message that player P_i sends to P_j through the link is eventually received (albeit probably in any order).

The set of faults in the players is usually captured using the notion of an external centralized *adversary*. A computationally unbounded *adversary* $\mathcal{B}$ is a probabilistic strategy that controls/corrupts a subset of players (and/or the edges connecting the players) and endeavors to violate the security of the system. We assume, without loss of generality, that the adversary can control/corrupt only the players and *not* the edges connecting them.[4]

Our notional adversary is a *passive* adversary if all dishonest players exhibit only *passive* adversarial behavior, that is, all the corrupted players can collusively gather all the information they get and run any arbitrary computation on this information. The adversary is a *Byzantine* adversary if all dishonest players show *active* adversarial behaviour, that is, in addition to *eavesdropping*, they can maliciously alter their behavior in an arbitrary and coordinated fashion.[5] If some players exhibit only *passive* adversarial behavior while others exhibit *active* adversarial behavior, the adversary is called a *mixed* adversary. Depending upon the amount of knowledge one has about the adversarial behaviour of the players, adversaries can be modeled as either *threshold* adversaries or *generalized* (or *non-threshold*) adversaries. When modeled as a *threshold* adversary, a maximum of t out of the n players are assumed to exhibit adversarial behaviour. Hirt and Maurer [9] transferred and adjusted the notion of *access structures* (introduced in [10] for secret sharing) to the field of general secure multiparty computation, which was subsequently adapted to the secure communication setting in [12]: the behaviour of the faulty players is characterized by an *adversary structure*, which is a monotone set of subsets of players, wherein the players in any *one* of these subsets is corruptible by the adversary.

In our study we consider *mixed* adversaries modeled using generalized adversary structures (like in [7]). In this model, some subset of players D show *active* adversarial behaviour and *at the same time*, some other subset of players E show only *passive* adversarial behaviour. Hence, the adversary is characterized by a monotone[6] set of classes $C = (D, E)$, where $D, E \subset \mathcal{P}$ and $D \cap E = \emptyset$. The players in *one* specific class is corruptible by the adversary. – players in D are actively corrupted while those in E are passively corrupted.

Definition 1 ([7]). *A* generalized mixed adversary structure $\mathcal{A}$ *is a monotone set of classes* $C = (D, E)$*, where* $D, E \subset \mathcal{P}$ *and* $D \cap E = \emptyset$*. The maximal basis of* $\mathcal{A}$ *is defined as the collection of classes* $\{(D, E) | (D, E) \in \mathcal{A}, \nexists (X, Y) \in \mathcal{A}, ((X \supset D) \cap (Y \supset E))\}$*. We abuse the notation* $\mathcal{A}$ *to denote the maximal basis.*

[4] This is because, any adversary corrupting both the players and edges of a network $\mathcal{N}$ can be simulated by an adversary corrupting the players alone on a new network $\mathcal{N}'$ got by replacing each insecure edge $e = (P_i, P_j)$ by a player P_{ij} and two edges $e_1 = (P_i, P_{ij})$ and $e_2 = (P_{ij}, P_j)$.

[5] Note that this subsumes fail-stop faults wherein the dishonest players alter their behaviour in a pre-specified manner, viz., do not respond at all.

[6] *Monotone* means that if a class $C = (D, E)$ belongs to the structure, then all classes $C' = (D', E')$ such that $D' \subseteq D$ and $E' \subseteq E$ are also elements of the structure.

Remark: Similar to the classical threshold model, the threshold mixed adversarial model can be defined as one in which up to t_a players maliciously attempt to disrupt the protocol, while up to t_p other players only eavesdrop.

We introduce the notion of a *path adversary* (like in [12]). Any message transmitted from a player P_i to a player P_j should traverse a path in $\mathcal{N}$ connecting the players. Hence, it is more appropriate to consider paths as corruptible entities rather than considering the adversarial behaviour of the individual players. This *path adversary* we characterize with the help of a generalized mixed structure. Let the set of all paths between players P_i and P_j in $\mathcal{N}$ be denoted by $\mathcal{X}_{path}(P_i, P_j)$.

Definition 2. *Given the generalized mixed adversary structure $\mathcal{A}$, we denote the* path adversary structure *over a subset of paths $\Phi(P_i, P_j) \subseteq \mathcal{X}_{path}(P_i, P_j)$ as $\mathcal{A}^{[\Phi]}_{path}(P_i, P_j)$. $\mathcal{A}^{[\Phi]}_{path}(P_i, P_j)$ is a monotone set of subset pairs of $\Phi(P_i, P_j)$. For every class $C = (D, E) \in \mathcal{A}$ there is a* corresponding *class $\Lambda = (\Lambda_D, \Lambda_E) \in \mathcal{A}^{[\Phi]}_{path}(P_i, P_j)$ such that Λ_D (or Λ_E) is the set of all paths in $\Phi(P_i, P_j)$ between P_i and P_j passing through* any *of the players in D (or E, respectively). More precisely, $\mathcal{A}^{[\Phi]}_{path}(P_i, P_j) \subset 2^{\Phi(P_i, P_j)}$, such that*

$$\mathcal{A}^{[\Phi]}_{path}(P_i, P_j) = \{(\Lambda_D, \Lambda_E) | _{\forall (D,E) \in \mathcal{A}, (\Lambda_D = \Phi(P_i,P_j) \setminus (N_{[\mathcal{P} \setminus D]})) \cap (\Lambda_E = \Phi(P_i,P_j) \setminus (N_{[\mathcal{P} \setminus E]}))}\}$$

where $N_{[V]}$ denotes the set of all paths in the sub-network induced by $\mathcal{N}$ on the vertices in V.

Definition 3. *Given the generalized mixed adversary structure $\mathcal{A}$, the network is said to be $\mathcal{A}^{(k,\ell)}$-*connected *if for any* $\max(k, \ell)$ *classes $C_{i_1}, C_{i_2}, \ldots, C_{i_{\max(k,\ell)}}$ from $\mathcal{A}$, the deletion of the nodes in $\bigcup_{j=1}^{k} D_{i_j} \cup \bigcup_{j=1}^{\ell} E_{i_j}$ from the network does not disconnect the network. With respect to two players (nodes) P_i and P_j, the network is said to be $\mathcal{A}^{(k,\ell)}(P_i, P_j)$-*subconnected *if for any* $\max(k, \ell)$ *classes $C_{i_1}, C_{i_2}, \ldots, C_{i_{\max(k,\ell)}}$ from $\mathcal{A}$, the deletion of the nodes in $\bigcup_{j=1}^{k} D_{i_j} \cup \bigcup_{j=1}^{\ell} E_{i_j}$ from the network does not render P_i unreachable from P_j.*

Remark: It is evident that that for the threshold mixed adversarial model, the $\mathcal{A}^{(k,\ell)}$-connectivity condition translates to $\kappa > kt_a + \ell t_p$, where κ denotes the size of the smallest vertex cut.

Since the underlying network is asynchronous, the adversary has the power to *schedule* the messages. A message routed on a path having an actively corrupted player (which we shall call henceforth as an actively corrupted path) can schedule the message in such a way that the receiver will be made to wait for it for arbitrary long periods of time. Actually, these actively corrupted paths may just withhold the messages routed through them and thus receiver $\mathbf{R}$ may not listen from the sender $\mathbf{S}$ on paths in Λ_D, $(\Lambda_D, \Lambda_E) \in \mathcal{A}^{\mathcal{X}}_{path}(\mathbf{S}, \mathbf{R})$. However, the receiver can not distinguish between *honest* paths which are *slow* (thanks to the malicious scheduling) and *malicious* paths which *withhold* information. Hence, in the worst case, the set of paths on which $\mathbf{R}$ can expect to receive information might contain all the *malicious* and the *eavesdropping* paths!

3 Secure Message Transmission

We consider the problem of transmitting a message m from a player P_i to a player P_j *securely*. We consider perfect and unconditional (with a small error probability) security in the information theoretic sense. In this section we define perfect security and unconditional security. Let the message to be transmitted securely be drawn from a (prespecified) fixed finite field $\mathcal{F}$ and let Γ denote the underlying probability distribution on this field. Define the View of a player P_j in $\mathcal{N}$, at any point of the execution of a protocol Π for secure message transmission, to be the information the player can get from its *local input* to the protocol (if any), all the messages that P_j had earlier *sent* or *received*, the protocol *code* executed by P_j and its *random coins*. The View of the adversary (i.e. the View of the players exhibiting adversarial behaviour) at any point of the execution of Π is defined as all the information that the adversary can get from the Views of all the players corrupted by the adversary (i.e. all the information that these players can commonly compute from their Views).

For every message $m \in \mathcal{F}$, any adversary $\mathcal{B}$ characterized by $\mathcal{A}$, and any protocol Π for secure message transmission, let $\widehat{\Gamma}(\mathcal{B}, m, \Pi)$ denote the probability distribution on the View of the adversary $\mathcal{B}$ at the end of the execution of Π when the message sent is m.

Definition 4 (Secure Message Transmission). *A protocol Π is said to facilitate perfectly secure message transmission between two players P_i and P_j if for any message m, drawn from Γ on $\mathcal{F}$, and for every adversary $\mathcal{B}$, characterized as an (generalized mixed) adversary structure $\mathcal{A}$, the following conditions are satisfied:*

1. Secrecy*: $\widehat{\Gamma}(\mathcal{B}, m', \Pi) \equiv \widehat{\Gamma}(\mathcal{B}, m, \Pi) \quad \forall m' \in \mathcal{F}$. That is, the above two distributions are identical irrespective of the message transmitted.*
2. Resiliency*: The protocol certainly terminates with the receiver P_j receiving the message m correctly.*

The protocol Π is said to be unconditionally secure (with negligible error) if a negligible error probability δ can be tolerated with respect to the Resiliency *condition, i.e., the protocol terminates with an overwhelming probability $1-\delta$ and the receiver P_j receives m with a negligibly small error probability δ. The probability is over the choice of m and the coin flips of each of the players and the adversary (this is same as the $(0, \delta)$-security as defined in [8]).*

4 Issues

Before we start designing protocols for asynchronous secure communication between the sender $\mathbf{S}$ and the receiver $\mathbf{R}$ over the network $\mathcal{N}$, tolerating generalized mixed adversaries, there are a few critical issues which have to be dealt with:

1. *What are the paths that, out of the potentially* exponential *number of paths between $\mathbf{S}$ and $\mathbf{R}$, should be used for transmission?* Note that irrespective

of the total number of paths from **S** to **R**, only a polynomial (on the input size, i.e., $n + |\mathcal{A}|$) sized subset of the paths should be used if the resultant protocol is to be feasible. Furthermore, one should be able to compute the above subset of paths in polynomial time!

2. *What to send along the above chosen paths?* In the sequel, it is shown that the answer depends on the setting.
3. *How to route a message along a path?* Since, the adversary can actively corrupt some players, these intermediate players can misroute a message.
4. *How does the receiver* **R** *distinguish between two different paths having the same final link?* This may be required since the receiver invariably has to "reconstruct" the sender's message from the data that he receives via many different paths, and the data may be an ordered set.

4.1 Solving Issue #1: Critical Paths

Consider a network in which at most t nodes are faulty. In such a case, irrespective of which nodes are corrupted it is evident that in the worst case, not more than t disjoint paths can be corrupted (one node per path). Hence if the network is κ-connected, it is sufficient to abstract the network as k disjoint paths between **S** and **R** of which any t paths could be faulty; or even better as k wires of which any t could be corrupted. This is exactly what [5, 13] have done! In the above case, we call the k (disjoint) paths that are chosen as the *critical paths*. Note that the number of critical paths is usually much lower than the total number of paths between **S** and **R**. Unfortunately, when the players' adversarial behaviour is modeled as a generalized mixed adversary, however, the non-disjointness of the communication paths is *indispensable*.

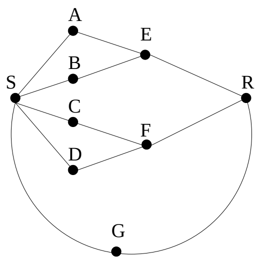

Fig. 2. Network $\mathcal{N}_1$

For example, consider the network $\mathcal{N}_1$ in the Fig. 2. Let the adversary be characterized by the following mixed adversary structure $\mathcal{A} = \{(A, \emptyset), (B, \emptyset), (F, \emptyset), (G, \emptyset)\}$. There are in total five paths from **S** to **R**. It can be easily seen (using the results of [5, 13]) that any protocol for asynchronous secure message transmission between **S** and **R** should necessarily use four of the paths between **S** and **R**, leaving out one of the two paths passing through **F** (since the node **F** is potentially corruptible). Note that in any case, the chosen four paths are all not disjoint! Furthermore, the path that is left out is not a critical path.

We now need to develop a deterministic methodology for computing the critical paths; moreover, we require that the algorithm runs in time polynomial in the input size. Assume that the sender **S** and receiver **R** are $\mathcal{A}^{(k_1,0)}$-subconnected as well as $\mathcal{A}^{(k_2,1)}$-subconnected.[7] We solve the above issue by providing an algorithm (see Fig. 3) with the following properties: (1) The algorithm takes as

[7] It will be clear from the sequel that in the various settings considered in this paper, we will be dealing with only $\mathcal{A}^{(3,0)}$-subconnectivity and $\mathcal{A}^{2,1)}$-subconnectivity. More precisely, in the perfect and unconditional settings, $k_1 = 3$ and $k_2 = 2$; and in the unconditional with broadcast setting, $k_1 = 0$ and $k_2 = 2$.

input $\mathcal{N}(\mathcal{P},\mathcal{E})$, $\mathcal{A}$, and the sender $\mathbf{S}$ and the receiver $\mathbf{R}$. (2) The algorithm outputs a set of paths between $\mathbf{S}$ and $\mathbf{R}$ in $\mathcal{N}$, denoted by $Pot\mathcal{X}_a(\mathbf{S},\mathbf{R})$. (3) The algorithm runs in time polynomial in $|\mathcal{P}|$ and $|\mathcal{A}|$, viz. $O(|\mathcal{P}|\cdot|\mathcal{A}|^6)$. (4) The number of paths in $Pot\mathcal{X}_a(\mathbf{S},\mathbf{R})$ is polynomial in $|\mathcal{A}|$, viz. $O(|\mathcal{A}|^3)$. (5) A solution using $Pot\mathcal{X}_a(\mathbf{S},\mathbf{R})$ exists if and only if a solution that uses the full set of paths $\mathcal{X}_{path}(\mathbf{S},\mathbf{R})$ exists, i.e. it is ensured that $\mathbf{S}$ and $\mathbf{R}$ are $\mathcal{A}^{(k_1,0)}$-subconnected as well as $\mathcal{A}^{(k_2,1)}$-subconnected even if the set of paths is restricted to $Pot\mathcal{X}_a(\mathbf{S},\mathbf{R})$.

Computing $Pot\mathcal{X}_a(\mathbf{S},\mathbf{R})$

Inputs: $\mathcal{N}(\mathcal{P},\mathcal{E})$, $\mathcal{A}$, sender $\mathbf{S}$ and receiver $\mathbf{R}$.

```
Let $Pot\mathcal{X}_a(\mathbf{S},\mathbf{R}) = \emptyset$ and let $\mathcal{A}^{(1)}_{access} = \{A_i = (\mathcal{P} \setminus D) | \forall (D,E) \in \mathcal{A}\}$.
For $i_1 = 1$ to $|\mathcal{A}^{(1)}_{access}|$
  For $i_2 = i_1 + 1$ to $|\mathcal{A}^{(1)}_{access}|$
  ...
    For $i_{k_1} = i_{k_1 - 1} + 1$ to $|\mathcal{A}^{(1)}_{access}|$
       IF $(Pot\mathcal{X}_a(\mathbf{S},\mathbf{R}) \cap \mathcal{N}_{[A_{i_1} \cap A_{i_2} \cap \ldots \cap A_{i_{k_1}}]}) == \emptyset$
       THEN Select at random some path $p$ in $\mathcal{N}_{[A_{i_1} \cap A_{i_2} \cap \ldots \cap A_{i_{k_1}}]}$
              and set $Pot\mathcal{X}_a(\mathbf{S},\mathbf{R}) \leftarrow Pot\mathcal{X}_a(\mathbf{S},\mathbf{R}) \cup \{p\}$.
    NEXT $i_{k_1}$
  ...
  NEXT $i_2$
NEXT $i_1$
Let $\mathcal{A}^{(2)}_{access} = \{A_i = (\mathcal{P} \setminus (D \cup E)) | \forall (D,E) \in \mathcal{A}\}$.
For $i_1 = 1$ to $|\mathcal{A}^{(2)}_{access}|$
  ...
  For $i_{k_2} = i_{k_2 - 1} + 1$ to $|\mathcal{A}^{(1)}_{access}|$
    IF $(Pot\mathcal{X}_a(\mathbf{S},\mathbf{R}) \cap \mathcal{N}_{[A_{i_1} \cap \ldots \cap A_{i_{k_2}}]}) == \emptyset$
    THEN Select at random some path $p$ in $\mathcal{N}_{[A_{i_1} \cap \ldots \cap A_{i_{k_2}}]}$
           and set $Pot\mathcal{X}_a(\mathbf{S},\mathbf{R}) \leftarrow Pot\mathcal{X}_a(\mathbf{S},\mathbf{R}) \cup \{p\}$.
  NEXT $i_{k_2}$
  ...
NEXT $i_1$
```

Comment: *The above construction ensures that* $\mathbf{S}$ *and* $\mathbf{R}$ *are* $\mathcal{A}^{(k_1,0)}$*-subconnected as well as* $\mathcal{A}^{(k_2,1)}$*-subconnected even if the set of paths is restricted to* $Pot\mathcal{X}_a(\mathbf{S},\mathbf{R})$.

Fig. 3. Identifying the critical paths $Pot\mathcal{X}_a(\mathbf{S},\mathbf{R})$.

Theorem 1. *The algorithm in Fig. 3 satisfies all the above stated properties. We assume the worst case of* $k_1 = 3$ *and* $k_2 = 2$.

PROOF OF PROPERTY 3: The only computational intensive step is the IF step, which takes $O(|\mathcal{P}|\cdot|Pot\mathcal{X}_a(\mathbf{S},\mathbf{R})|)$ time. Since, $|Pot\mathcal{X}_a(\mathbf{S},\mathbf{R})| = O|\mathcal{A}|^3$ (see Proof of Property 4), the overall computational complexity is $O(|\mathcal{P}|\cdot|\mathcal{A}|^6)$.
PROOF OF PROPERTY 4: The property is clear from the fact that in each of the $\left\{\binom{|\mathcal{A}|}{3} + \binom{|\mathcal{A}|}{2}\right\}$ iterations, the size of $Pot\mathcal{X}_a(\mathbf{S},\mathbf{R})$ increases by at most *one*.
PROOF OF PROPERTY 5: Follows from the construction. □

4.2 Solving Issues #2 & #4: Anonymous Secret Sharing

We require that the adversary should get no information about the message, whilst **R** should be able to reconstruct the message (or should at least be able to detect a fault, if not correct it!). This is reminiscent of secret sharing. Moreover, **R** may not be able to distinguish between shares routed on *different* paths arriving through the *same* final link. Therefore the requirement is that of *anonymous secret sharing* [4]. In our case, anonymization is easily achieved by creating *self-identifying* message packets by appending the intended path number to the message, i.e., if the message packet ϖ_i is to be routed through path p_i, the *self-identifying* packet would be (ϖ_i, i). The above abstraction helps us work with secret sharing alone since it can easily be "compiled" into anonymous secret sharing. In the unconditional case, as will be illustrated in the sequel, it is sufficient if the secret is split into shares such that their sum gives back the secret. However, in the perfect setting, we use the linear perfect secret sharing schemes based on monotone span programs [11].

4.3 Solving Issue #3: Routing Algorithm

As a recap, a routing primitive is essential since it is not guaranteed that a message intended to be routed along a path will reach the receiver on that particular path. We prove that such a strong primitive is not required and the weaker primitive described in Observation 11 is sufficient to design secure message transmission protocols. We next provide the routing primitive Δ_{async} (see Fig. 4) for asynchronous networks.

Observation 11 *It is sufficient to have a routing algorithm Δ that guarantees that on every honest path p_i (identified by i),* **R** *receives exactly one correct self-identifying packet, namely (ϖ_i, i).*

We now prove that the algorithm Δ_{async} satisfies the weak routing primitive described in Observation 11.

Theorem 2. *The Routing Algorithm Δ_{async} for asynchronous networks, given in Fig. 4, satisfies the specification as in Observation 11.*

Proof: Let p_i be an *honest* path through which **S** intended to send message ϖ_i. On the contrary, assume that **R** received on path p_i either:

1. *One incorrect packet*: This leads to a contradiction because if the packet was correct up to q hops, our algorithm ensures that it is correct even after $q+1$ hops. Proof follows through induction.
2. *No packet*: As all the nodes on the path are honest it is clear that at least one packet (viz. the packet routed through that path (ϖ_i, i)) will *eventually* reach **R**. (Note that, however, **R** may not wait for this message.)
3. *More than one packet*: Since more than one packet was received on the honest path p_i (with the same path identifier), there exists at least one packet whose path identifier was corrupted to i. Let N be the corrupted node where the

Routing Algorithm $\boldsymbol{\Delta}_{async}$

Inputs: The sender $\mathbf{S}$, receiver $\mathbf{R}$, message ϖ_i to be routed through path p_i, and the subset of paths $\Phi(\mathbf{S},\mathbf{R}) \subset \mathcal{X}_{path}(\mathbf{S},\mathbf{R})$ under consideration.
Code for the Sender S:
Send packet (ϖ_i, i) to node N, IF path $p_i = (\mathbf{S}, N, \ldots, \mathbf{R})$ and $p_i \in \Phi(\mathbf{S},\mathbf{R})$.
Code for an Internal Node N:
IF the packet (ϖ_i, i), where path $p_i = (\mathbf{S}, \ldots, N_1, N_2, N_3, \ldots, \mathbf{R})$ and $p_i \in \Phi(\mathbf{S},\mathbf{R})$, is received from N_1 and $N_2 == N$, THEN send it to N_3. ELSE *throw away* the packet (ϖ_i, i).
Code for the Receiver R:
For each $p_i \in \Phi(\mathbf{S},\mathbf{R})$, initialize $\tau_i = \vdash$.
Let S_{recd} (initialized to $\emptyset$) keep track of the paths through which $\mathbf{R}$ receives.

```
IF $\mathbf{R}$ receives the packet $(\varpi_i, i)$ from $N_\ell$, where path $p_i = (\mathbf{S}, \ldots, N_\ell, \mathbf{R})$
  THEN
  $S_{recd} \longleftarrow S_{recd} \cup \{p_i\}$.
    IF $\tau_i == \vdash$
      THEN set $\tau_i = (\varpi_i, i)$.
      ELSE set $\tau_i = \perp$.
  ELSE set $\tau_i = \perp$. // i.e., when the message is received but not along $p_i$.
```

Wait until $S_{recd} \supseteq (\Phi(\mathbf{S},\mathbf{R}) \setminus \Lambda_D)$ for some $(\Lambda_D, \Lambda_E) \in \mathcal{A}^{[\Phi]}_{path}(\mathbf{S},\mathbf{R})$.

Comment: If τ_i still remains as $\vdash$, it means *no messages were received on* p_i. If τ_i is $\perp$, it means that *the message(s) received over* p_i *are probably invalid and better not used.*

Fig. 4. The Routing Algorithm for Asynchronous Networks.

path identifier was corrupted to i and sent to a honest node N_h in p_i. Δ_{async} ensures that this packet is *thrown away* by N_h. Notice that the same holds even when $N_h = \mathbf{R}$. □

5 Unconditionally Secure Communication with Broadcast

5.1 Impossibility

Theorem 3. *Unconditionally secure message transmission tolerating $\mathcal{A}$ across an asynchronous network $\mathcal{N}$ with broadcast capability is possible* only if *the sender* $\mathbf{S}$ *and the receiver* $\mathbf{R}$ *are $\mathcal{A}^{(2,1)}(\mathbf{S},\mathbf{R})$-subconnected.*

Proof: Assume that, on the contrary, secure transmission with negligible error is possible even when $\mathbf{S}$ and $\mathbf{R}$ are *not* $\mathcal{A}^{(2,1)}(\mathbf{S},\mathbf{R})$-subconnected. In this case, the adversary can exploit the asynchrony of the network and delay the messages routed through the paths in Λ_D for some class $\Lambda = (\Lambda_D, \Lambda_E) \in \mathcal{A}^{\mathcal{X}}_{path}(\mathbf{S},\mathbf{R})$. Since $\mathbf{S}$ and $\mathbf{R}$ are not $\mathcal{A}^{(2,1)}(\mathbf{S},\mathbf{R})$-subconnected, there exists a class Λ' such that $\Lambda' = (\Lambda'_D, \Lambda'_E) \in \mathcal{A}^{\mathcal{X}}_{path}(\mathbf{S},\mathbf{R})$ such that $\Lambda_D \cup \Lambda'_D \cup \Lambda'_E = \mathcal{X}_{path}(\mathbf{S},\mathbf{R})$. Thus by choosing to corrupt this class Λ' the adversary has all the knowledge that $\mathbf{R}$ takes into consideration thus violating the secrecy requirement. □

5.2 Possibility

We propose a protocol sketch for unconditionally secure communication on the lines of [8] and show that the $\mathcal{A}^{(2,1)}(\mathbf{S},\mathbf{R})$-connectivity condition is sufficient.

Asynchronous Unconditionally Secure Transmission

1. $\mathbf{S}$ sends different $\rho_j^{\mathbf{S}}, \sigma_j^{\mathbf{S}} \in \mathcal{F}$ on each path p_j in $\Phi(\mathbf{S}, \mathbf{R})$.
2. For each, $\rho_j^{\mathbf{R}}, \sigma_j^{\mathbf{R}}$ that $\mathbf{R}$ receives on the correct path p_j, $\mathbf{R}$ *reliably sends* (using the broadcast channel) a random $\nu_j^{\mathbf{R}} \in \mathcal{F}$ and $s_j^{\mathbf{R}} = (\nu_j^{\mathbf{R}} \cdot \rho_j^{\mathbf{R}} + \sigma_j^{\mathbf{R}})$ to $\mathbf{S}$.
3. $\mathbf{S}$ constructs $G = \{j | s_j^{\mathbf{R}} = (\nu_j^{\mathbf{R}} \cdot \rho_j^{\mathbf{S}} + \sigma_j^{\mathbf{S}})\}$. $\mathbf{S}$ *reliably sends* (using the broadcast channel) G and $Z = m^{\mathbf{S}} + \Sigma_G\ \rho_j^{\mathbf{S}}$.
4. $\mathbf{R}$ computes $m^{\mathbf{R}} = Z - \Sigma_G\ \rho_j^{\mathbf{R}}$.

Fig. 5. Unconditionally Secure Transmission with Broadcast.

Theorem 4. *The protocol given in Fig. 5 is indeed a protocol guaranteeing unconditionally secure communication* if *the sender* $\mathbf{S}$ *and the receiver* $\mathbf{R}$ *are* $\mathcal{A}^{(2,1)}$*-subconnected.*

Proof of Secrecy: Since the network is asynchronous, the receiver can expect to receive messages only on paths in $(\mathcal{X}_{path}(\mathbf{S}, \mathbf{R}) \setminus \Lambda_D)$ (for some $\Lambda = (\Lambda_D, \Lambda_E) \in \mathcal{A}_{path}^{\mathcal{X}}(\mathbf{S}, \mathbf{R})$. However, since $\mathbf{S}$ and $\mathbf{R}$ are $\mathcal{A}^{(2,1)}$-subconnected, even if the adversary corrupts some other class $\Lambda' = (\Lambda'_D, \Lambda'_E)$, there exists one path p_h on which the messages will reach $\mathbf{R}$ and has no corrupted (active or passive) player. Hence, $\mathbf{R}$ receives the values $\rho_j^{\mathbf{S}}$ and $\sigma_j^{\mathbf{S}}$ correctly. The adversary can not find out the value $\rho_j^{\mathbf{S}}$ even using $s_j^{\mathbf{R}}$ and $\nu_j^{\mathbf{R}}$. Hence the adversary can only *guess* the value of $m^{\mathbf{S}}$ even after knowing Z.
Proof of Resiliency: $m^{\mathbf{S}} \neq m^{\mathbf{R}}$ if and only if $\rho_j^{\mathbf{S}} \neq \rho_j^{\mathbf{R}}$ for some $j \in G$. This occurs with a probability $\frac{1}{|\mathcal{F}|}$. Hence, $Pr(m^{\mathbf{S}} \neq m^{\mathbf{R}}) \leq \frac{|G|}{|\mathcal{F}|}$, which can be made sufficiently small since one could choose the working field such that $|\mathcal{F}| > \frac{|G|}{\delta}$ and still have the compute, round and communication complexity of the resultant protocol polynomial in the size of the network, the size of the maximal basis of the adversary structure and $\log \frac{1}{\delta}$ (if $\delta > 0$). □

6 Unconditionally Secure Communication without Broadcast

6.1 Impossibility

Theorem 5. *Unconditionally secure message transmission tolerating* $\mathcal{A}$ *across an asynchronous network* $\mathcal{N}$ *without broadcast capability is possible* only if *the sender* $\mathbf{S}$ *and the receiver* $\mathbf{R}$ *are* $\mathcal{A}^{(2,1)}(\mathbf{S}, \mathbf{R})$*-subconnected as well as* $\mathcal{A}^{(3,0)}(\mathbf{S}, \mathbf{R})$*-subconnected.*

Proof: As a direct consequence of Theorem 3, $\mathbf{S}$ and $\mathbf{R}$ should be at least $\mathcal{A}^{(2,1)}(\mathbf{S}, \mathbf{R})$-subconnected for any secure communication protocol to satisfy the *secrecy* condition. For the sake of contradiction, assume that there exists a scheme ξ for unconditionally secure communication even when $\mathbf{S}$ and $\mathbf{R}$ are *not* $\mathcal{A}^{(3,0)}(\mathbf{S}, \mathbf{R})$-subconnected. Using ξ, we construct a protocol ξ' for unconditionally secure communication between a sender $\mathbf{S}'$ and a receiver $\mathbf{R}'$ over a network

$\mathcal{N}'$, wherein $\mathbf{S}'$ and $\mathbf{R}'$ are connected by exactly *three* disjoint paths (p_1, p_2 and p_3), at least one of which is corrupted by a Byzantine adversary. However, in this case, secure communication of any sort is impossible. The adversary can delay the message through one of the paths so that $\mathbf{R}'$ does not take it into consideration. Then the adversary can corrupt one of the other two messages. Thus $\mathbf{R}'$ will have to reconstruct the secret using one of the two messages and in a sense has exactly the same amount of information as the adversary. Since the adversary should not get more information about the secret (than what he can get by random guessing), unconditionally secure communication is not possible.

Construction of ξ' from ξ is simple. Since, $\mathbf{S}$ and $\mathbf{R}$ are not $\mathcal{A}^{(3,0)}(\mathbf{S},\mathbf{R})$-subconnected, there exist three classes $\Lambda_1 = (\Lambda_{D_1}, \Lambda_{E_1}), \Lambda_2 = (\Lambda_{D_2}, \Lambda_{E_2}), \Lambda_3 = (\Lambda_{D_3}, \Lambda_{E_3})$ in the path adversary structure such that $\Lambda_{D_1} \cup \Lambda_{D_2} \cup \Lambda_{D_3} = \mathcal{X}_{path}(\mathbf{S},\mathbf{R})$. Construct scheme ξ' wherein, every message sent through a path in Λ_{D_1} in ξ is sent through the path p_1 in ξ'; every message sent through a path in $\Lambda_{D_2} \setminus \Lambda_{D_1}$ in ξ is sent through the path p_2 in ξ'; and every message sent through a path in $\Lambda_{D_3} \setminus (\Lambda_{D_1} \cup \Lambda_{D_2})$ in ξ is sent through the path p_3 in ξ'. Hence, if ξ is a protocol for unconditionally secure communication, so is ξ'. □

6.2 Possibility

We remark that the protocol for unconditionally secure communication in Fig. 5 will work even in this case, if we are able to simulate *reliable but insecure* transmission. We provide below a protocol for the same (which we call PUBLIC transmission) whenever the sender and receiver are at least $\mathcal{A}^{(3,0)}(\mathbf{S},\mathbf{R})$-subconnected.

Public Transmission & Reception

Transmission: Sender $\mathbf{S}$ sends message m on all paths in $\Phi(\mathbf{S},\mathbf{R})$.
Reception: The receiver $\mathbf{R}$ receives on paths in $S_{recd} = (\Phi(\mathbf{S},\mathbf{R}) \setminus \Lambda_D)$ for some $(\Lambda_D, \Lambda_E) \in \mathcal{A}_{path}^{[\Phi]}(\mathbf{S},\mathbf{R})$.
Let the receiver $\mathbf{R}$ receive m'_i on path $p_i \in S_{recd}$.
Set $m' = m'_i$ such that $\exists(\Lambda'_D, \Lambda'_E) \in \mathcal{A}_{path}^{[\Phi]}(\mathbf{S},\mathbf{R}), \{j | m'_j = m'_i\} \supseteq (S_{recd} \setminus \Lambda'_D)$.
m' is the PUBLICLY received message.
Comment: *The* PUBLICLY RECEIVED *message m' is that message which is received through the paths which form a set in the path access structure.*

Fig. 6. Public Transmission and Reception.

Theorem 6. Reliable*, yet insecure transmission is possible* if *and* only if $\mathbf{S}$ *and* $\mathbf{R}$ *are* $\mathcal{A}^{(3,0)}(\mathbf{S},\mathbf{R})$*-subconnected.*

NECESSARY: Assume the contrary. Then there exists classes Λ_1, Λ_2 and Λ_3 such that $\Lambda_1 \cup \Lambda_2 \cup \Lambda_3 = \mathcal{X}_{path}(\mathbf{S},\mathbf{R})$. The adversary slows down the messages in Λ_1 and corrupts those in either Λ_2 or Λ_3. Note that $\mathbf{R}$ has no idea whether Λ_2 is corrupted or Λ_3 is corrupted and can not decide whether messages through Λ_2

are correct or those through Λ_3 are correct. Thus reliable transmission is not possible.
SUFFICIENT: See protocol in Fig. 6. Suppose $\mathbf{S}$ transmits a message m to $\mathbf{R}$. Let $\mathbf{R}$ receive m'. Assume that $m' \neq m$. This would require the adversary to corrupt the paths in $\Lambda_B = (S_{recd} \setminus \Lambda'_D) = (\Phi(\mathbf{S},\mathbf{R}) \setminus (\Lambda_D \cup \Lambda'_D))$. Hence, $\exists(\Lambda''_D, \Lambda''_E) \in \mathcal{A}^{[\Phi]}_{path}(\mathbf{S},\mathbf{R})$ such that $\Lambda''_D \supseteq \Lambda_B$. This would imply that, $\exists(\Lambda_D,\Lambda_E),(\Lambda'_D,\Lambda'_E)$, $(\Lambda''_D,\Lambda''_E) \in \mathcal{A}^{[\Phi]}_{path}(\mathbf{S},\mathbf{R})$ such that $\Lambda_D \cup \Lambda'_D \cup \Lambda''_D = \Phi(\mathbf{S},\mathbf{R})$. This leads to a contradiction since $\mathbf{S}$ and $\mathbf{R}$ are $\mathcal{A}^{(3,0)}(\mathbf{S},\mathbf{R})$-subconnected. □

7 Perfectly Secure Communication

7.1 Impossibility

Theorem 7. *Perfectly secure message transmission tolerating $\mathcal{A}$ across an asynchronous network $\mathcal{N}$ is possible* only if *the sender* $\mathbf{S}$ *and the receiver* $\mathbf{R}$ *are* $\mathcal{A}^{(2,1)}(\mathbf{S},\mathbf{R})$*-subconnected as well as* $\mathcal{A}^{(3,0)}(\mathbf{S},\mathbf{R})$*-subconnected. This is irrespective of whether $\mathcal{N}$ has broadcast capabilities or not.*

PROOF: Assume for the sake of contradiction that there exists a scheme ξ for secure message transmission of $m \in \mathcal{F}$ from $\mathbf{S}$ to $\mathbf{R}$ tolerating $\mathcal{A}$ across $\mathcal{N}$ *not* satisfying either $\mathcal{A}^{(3,0)}(\mathbf{S},\mathbf{R})$-subconnectivity or the $\mathcal{A}^{(2,1)}(\mathbf{S},\mathbf{R})$-subconnectivity condition. Also assume that each execution of ξ proceeds in phases, and that in the odd phase the sender $\mathbf{S}$ sends messages to the receiver $\mathbf{R}$ while in an even phase transmits to $\mathbf{S}$.
Case 1– Violation of $\mathcal{A}^{(3,0)}(\mathbf{S},\mathbf{R})$-subconnectivity:
In this case, there exist three classes $(\Lambda_{D_1},\Lambda_{E_1}),(\Lambda_{D_2},\Lambda_{E_2}),(\Lambda_{D_3},\Lambda_{E_3}) \in \mathcal{A}^{\mathcal{X}}_{path}(\mathbf{S},\mathbf{R})$ such that $\Lambda_{D_1} \cup \Lambda_{D_2} \cup \Lambda_{D_3} = \mathcal{X}_{path}(\mathbf{S},\mathbf{R})$. Let $m \neq m' \in \mathcal{F}$. We construct two executions, Ψ and Ψ' of ξ that, for every k, are indistinguishable to $\mathbf{R}$ after k phases of communication. However, we construct the two executions in such a way that in Ψ the message being transmitted is m, while in Ψ' the message being transmitted is m' thus proving that these executions cannot terminate, violating the resiliency condition.

Assume that in phase $2i+1$ of the execution of Ψ (or Ψ'), $\mathbf{S}$ sends α_i(respectively $\alpha_i{}'$) through the paths in Λ_{D_1}, β_i(respectively $\beta_i{}'$) through the paths in $(\Lambda_{D_2} \setminus \Lambda_{D_1})$ and γ_i(respectively $\gamma_i{}'$) through the paths in $(\Lambda_{D_3} \setminus (\Lambda_{D_1} \cup \Lambda_{D_2}))$. The adversary corrupts Λ_{D_3}(respectively Λ_{D_2}) in the execution of Ψ(respectively Ψ') and delays the messages sent through paths in Λ_{D_1} so that $\mathbf{R}$ will not consider these messages. In each phase of the execution of Ψ(respectively Ψ'), the adversary corrupts the message γ_i(respectively $\beta_i{}'$) to $\gamma_i{}'$(respectively β_i). At the end of the i^{th} phase, $\mathbf{R}$ receives $\beta_i, \gamma_i{}'$ on paths in $(\Lambda_{D_2} \cup \Lambda_{D_3}) \setminus \Lambda_{D_1}$ in both the executions. Clearly, the receiver cannot distinguish between the two executions, violating the *resiliency* requirement. Note that the existence of a broadcast channel does not help.
Case 2– Violation of $\mathcal{A}^{(2,1)}(\mathbf{S},\mathbf{R})$-subconnectivity:
This means that there exist two classes $(\Lambda_{D_1},\Lambda_{E_1}),(\Lambda_{D_2},\Lambda_{E_2}) \in \mathcal{A}^{\mathcal{X}}_{path}(\mathbf{S},\mathbf{R})$

such that $\Lambda_{D_1} \cup \Lambda_{E_1} \cup \Lambda_{D_2} = \mathcal{X}_{path}(\mathbf{S}, \mathbf{R})$. Hence, by choosing the class $(\Lambda_{D_1}, \Lambda_{E_1})$ from $\mathcal{A}$ and delaying the messages routed through paths in Λ_{D_2} (so that $\mathbf{R}$ doesn't consider these messages), the adversary gains as much knowledge of the message as does $\mathbf{R}$ violating the *secrecy* requirement.

The fact that the presence of a broadcast primitive leaves the condition unaffected follows from Theorem 6 and the protocol (see Fig. 6) for simulating broadcast. □

7.2 Possibility

We show that the protocols in line with [5, 12], when modified to the asynchronous and generalized mixed adversary setting, works correctly over any network that is $\mathcal{A}^{(2,1)}(\mathbf{S}, \mathbf{R})$-subconnected as well as $\mathcal{A}^{(3,0)}(\mathbf{S}, \mathbf{R})$-subconnected. We begin by describing a method for sharing and reconstructing a secret, which we will denote as algorithm Υ (see Fig. 7). It is known that a polynomial sized MSP[8] [11] $\mathcal{M} = (\mathcal{F}, M_{d \times e}, \psi)$ can be constructed (preferably of size as small as possible), corresponding to the adversary structure $\mathcal{A}^{[\mathcal{X}]}_{path}(\mathbf{S}, \mathbf{R})$ with the range of ψ being $\Phi(\mathbf{S}, \mathbf{R})$ and the target vector $\vec{T} = [1, 0, \ldots, 0]$. With these inputs, we describe the *sharing* and *reconstruction* algorithm Υ in Fig. 7.

Our transmission protocol (see Fig. 8) runs in iterations. In each iteration the sender $\mathbf{S}$ attempts the transmission of a random pad ρ by sending share ϖ_i (constructed using Υ) along each path p_i. Since, is not assured of hearing on all the paths, $\mathbf{R}$ waits for messages on a subset of paths S_{recd} and attempts to reconstruct the random pad ρ (using the reconstruction algorithm of Υ) from the shares received. If $\mathbf{R}$ is not able to conclusively reconstruct a unique pad, $\mathbf{R}$ PUBLICLY SENDS all the received messages to $\mathbf{S}$. From these shares, $\mathbf{S}$ constructs the set of faulty paths F. First $\mathbf{S}$ constructs the set S_{recd}, the paths on which $\mathbf{R}$ actually received messages. Among these paths, $\mathbf{S}$ marks a path to be faulty (and adds to the set F) if $\mathbf{R}$ had received a wrong message. Note that at least one path is recognized as faulty, since otherwise the transmission of ρ would have been successful, terminating the pad-agreement phase of the protocol. Now $\mathbf{S}$ and $\mathbf{R}$ prune the adversary structure (as in the synchronous case) and restart the protocol for a different pad ρ'. When the transmission of a pad ρ is successful, $\mathbf{R}$ PUBLICLY SENDS **OK** upon which $\mathbf{S}$ sends $Z = m \oplus \rho$ PUBLICLY. The receiver can get back m by $m = Z \ominus \rho$.

[8] Every linear secret sharing scheme can be represented as a Monotone Span Program defined as the triple $(\mathcal{F}, M, \Im)$ where $\mathcal{F}$ represents a finite field, M is a $d \times e$ matrix with entries in $\mathcal{F}$, and $\Im : \{1 \ldots d\} \to \{P_1 \ldots P_n\}$ is a function. Each row of the matrix M is labeled by players in the sense that $\Im$ assigns the label $\Im(k)$ to the k-th row of M, $1 \leq k \leq d$. For $A \subset \{P_1 \ldots P_n\}$, M_A denotes the matrix that consists of all rows in M labeled by players in A. Let $\vec{T} \in \mathcal{F}^e$ be the *target vector*. A MSP is said to accept (or reject) a structure $\mathcal{Z}$ if $\forall Z \in \mathcal{Z}$, there exists (does not exist, respectively) a linear combination of the rows of M_Z which equals $\vec{T}$. An MSP is said to correspond to an adversary structure $\mathcal{A}_{adv}$ if it rejects *exactly* $\mathcal{A}_{adv}$. By the *size* of an MSP, we mean the number of rows in M.

Sharing & Reconstruction

Sharing: We arrive at the entity to be sent across each of the paths in $\Phi(\mathbf{S}, \mathbf{R})$ in the following three steps.

1. *Computing the Secret Shares*: Let ρ be the random pad to be agreed upon and $\vec{s} = [\rho, \varphi_1, \varphi_2, \ldots, \varphi_{e-1}]$ where all φ_i's are random elements in $\mathcal{F}$. The shares are constructed as $\vec{\chi} = M \times \vec{s}^{\,T}$ and the share χ_j corresponds to path $\psi(j)$ (label of the j^{th} row in M).
2. *"Anonymization" and the packet construction*: We convert each of the shares to be sent into what we call *self-identifying* packets. Consider each share χ_j as a binary string of length lg $|\mathcal{F}|$. The packet ϖ_i corresponding to path p_i is the base 3 number obtained as follows:
   - Append the digit 2 to the binary strings representing each χ_j, to get a ternary string denoted by χ'_j.
   - Concatenate all the χ'_j's assigned to p_i to get $(\varpi_i)_{base\ 3} = \chi'_{j_1} \circ \chi'_{j_2} \circ \ldots \circ \chi'_{j_k} = \chi_{j_1} \circ 2 \circ \chi_{j_2} \circ 2 \circ \ldots \circ 2 \circ \chi_{j_k} \circ 2$ where $\forall j_\ell$, the label $\psi(j_\ell) = p_i$.

Reconstruction: The receiver undoes the transformations performed by $\mathbf{S}$ to obtain the original shares $\vec{\chi}$. If all the shares are **correct**, then the receiver $\mathbf{R}$ can indeed reconstruct the secret as follows:

1. For each set $A \in \mathcal{A}^{[\Phi]}_{acc}$, find $\vec{\lambda_A}$ such that $\vec{\lambda_A} \times M_A = \vec{T}$. (By the MSP definition, such a $(1 \times e)$ vector $\vec{\lambda_A}$ should exist.)
2. $\rho = \vec{T} \times \vec{s}^{\,T} = \vec{\lambda_A} \times M_A \times \vec{s}^{\,T} = \vec{\lambda_A} \times \vec{\chi_A}$.

Fig. 7. Algorithm for Sharing and Reconstruction.

Theorem 8. *The transmission protocol given in Fig. 8 has the following properties: (1) The protocol provably terminates and runs in time polynomial in $(n + |\mathcal{A}_{adv}|)$. (2) The protocol satisfies the security requirements. (3) The overall message complexity of the protocol is polynomial in $(n + |\mathcal{A}_{adv}|)$.* The proof follows from the Lemma 1, Theorem 1, Lemma 2 and Lemma 3.

Lemma 1 (Termination). *The transmission protocol will terminate in at most $|\mathcal{A}|$ iterations.*

PROOF: We show that if an iteration did not successfully transmit the random pad ρ, at least one faulty path will be detected. In each iteration, every path can be classified as an *OK* path ($\mathbf{R}$ receives one message), or a *talkative* path ($\mathbf{R}$ receives more than one message), or a *silent* path. The transmission of ρ will be successful if all the messages received on the *OK* paths were correct and there are no *talkative paths.* (We know that *silent* paths form a disruptive set in one of the classes in the path adversary structure.) If there is even one *talkative* path p_i, it would be marked with $\tau_i = \perp$ and hence be recognized by $\mathbf{S}$ as faulty. If any one of the messages on the *OK* paths are wrong, this path will be recognized as faulty since $\mathbf{S}$ reliably receives all the messages received by $\mathbf{R}$ (due to PUBLIC TRANSMISSION). Therefore, in each unsuccessful iteration, at least one faulty path is detected and thereby eliminated. Because of PRUNING STEP((b)&(c)) (see Fig. 8) the faulty path cannot occur in all the sets in $\mathcal{A}^{[\Phi]}_{path}(\mathbf{S}, \mathbf{R})$. This would result in the elimination of at least one set from the path-adversary in each iteration, because of PRUNING STEP(a). Hence, the algorithm will terminate in at most $|\mathcal{A}^{[\Phi]}_{path}(\mathbf{S}, \mathbf{R})| = |\mathcal{A}|$ iterations. $\square$

```
The Transmission Protocol
Inputs: $m$, $\Phi(\mathbf{S},\mathbf{R})$, $\mathcal{A}^{[\Phi]}_{path}(\mathbf{S},\mathbf{R})$, Algorithm $\Upsilon$ (see Fig. 7) and $\Delta_{async}$.
Stage 1:
Code for S:
Choose a random pad $\rho \in \mathcal{F}$. Divide $\rho$ into shares $\vec{\sigma}$ using the Algorithm$\Upsilon$. SEND (using $\Delta$) each
share $\sigma_i$ on its corresponding path $p_i \in \Phi(\mathbf{S},\mathbf{R})$.
Stage 2:
Code for R:
Let $T := (\tau_1, \tau_2, \ldots, \tau_k)$ where $\tau_i$ is received on $p_i$.
IF     $\forall \Lambda \in \mathcal{A}^{[\phi]}_{acc}(\mathbf{S},\mathbf{R})$, the same value $\ell$ is reconstructed using $\Upsilon$ from shares in $\{\tau_i | p_i \in \Lambda\}$
THEN Assign $\rho := \ell$ and PUBLICLY SEND OK on all paths $p_i \in \Phi(\mathbf{S},\mathbf{R})$.
ELSE  PUBLICLY SEND $T$ on all paths $p_i \in \Phi(\mathbf{S},\mathbf{R})$.
Stage 3:
Code for S:
IF     OK received PUBLICLY in Step 2
THEN PUBLICLY SEND $Z = \rho \oplus m$ to R.
ELSE  DO

  1. Receive $T' = \{\tau_1, \ldots, \tau_{|\Phi(\mathbf{S},\mathbf{R})|}\}$ PUBLICLY and construct the set $S_{recd} = \{p_i \mid \tau'_i \neq \vdash\}$
  2. Define the set of faulty paths $F = \{p_j \mid \sigma_j \neq \tau'_j, \tau'_j \in T', p_j \in S_{recd}\}$.
  3. PUBLICLY SEND $F$ to R.
  4. PRUNING STEP:
     (a) $\mathcal{A}^{[\Phi]}_{path}(\mathbf{S},\mathbf{R}) \longleftarrow \{(\Lambda_D, \Lambda_E) \mid (\Lambda_D, \Lambda_E) \in \mathcal{A}^{[\Phi]}_{path}(\mathbf{S},\mathbf{R}),\ F \subseteq \Lambda_D\}$.
         Comment: Remove those sets from the path adversary structure which we know are
         certainly not corrupted by the adversary.
     (b) $G = \{p \mid p \in \Phi(\mathbf{S},\mathbf{R}), \forall (\Lambda_D, \Lambda_E) \in \mathcal{A}^{[\Phi]}_{path}(\mathbf{S},\mathbf{R}),\ p \in \Lambda_D\}$.
         Comment: Set G as the set of paths which we know are definitely corrupted.
     (c) Set $\Phi(\mathbf{S},\mathbf{R}) = (\Phi(\mathbf{S},\mathbf{R}) \setminus G)$ and update $\mathcal{A}^{[\Phi]}_{path}(\mathbf{S},\mathbf{R})$.
  5. REPEAT the protocol with the modified inputs and a new pad $\rho' \in \mathcal{F}$.

Code for R:
IF     OK PUBLICLY SENT to S in Stage 2
THEN RECEIVE $Z$ PUBLICLY and compute $m = Z \ominus \rho$.
ELSE   PUBLICLY RECEIVE $F$. Perform the PRUNING STEP and locally modify $\mathcal{A}^{[\Phi]}_{path}(\mathbf{S},\mathbf{R})$ and
$\Phi(\mathbf{S},\mathbf{R})$. REPEAT the protocol with the modified inputs.
```

Fig. 8. Perfectly Secure Transmission Protocol over Asynchronous Networks.

Lemma 2 (Security). *The protocol satisfies the resiliency and secrecy conditions for perfectly secure message transmission.*

PROOF OF RESILIENCE: The proof of resilience is similar to the one for the synchronous case. All that we need to prove is that whenever $\mathbf{R}$ is able to successfully reconstruct a value of ρ', then $\rho' = \rho$, i.e., $\mathbf{R}$ always reconstructs the correct value. We know that the path adversary structure satisfies $\mathcal{Q}^{(3,0)} \cap \mathcal{Q}^{(2,1)}$.[9] Exploiting the asynchrony of the network, the adversary can schedule the messages on the honest paths in some Λ_D, where $(\Lambda_D, \Lambda_E) \in \mathcal{A}^{[\Phi]}_{path}(\mathbf{S},\mathbf{R})$ and corrupt messages in some other paths in Λ'_D, where $(\Lambda'_D, \Lambda'_E) \in \mathcal{A}^{[\Phi]}_{path}(\mathbf{S},\mathbf{R})$. By the definition of the corresponding access structure, there exists an access set $\Lambda_A = \mathcal{P} \setminus (\Lambda_D \cup \Lambda'_D)$. Clearly in our case, the messages $\mathbf{R}$ receives through these paths are correct, i.e., they are the actual shares that $\mathbf{S}$ sent. Hence, secret reconstructed using the MSP and this access set will be ρ, the pad that $\mathbf{S}$ intended to send. $\mathbf{R}$ reconstructs the secret correctly if and only if the secrets

[9] The notation $\mathcal{Q}^{(k,\ell)}$ has the same meaning as defined in [7].

reconstructed using all the sets in the access structure are the same. Hence, if **R** successfully reconstructs the secret, surely the reconstructed pad will be ρ.

Proof of Secrecy:Let the secret message to be transmitted be $m \in \mathcal{F}$. We first observe that in each of attempted transmissions of a random pad ρ, the adversary cannot "access" the secret (by definition of a MSP). Moreover, each of the ρ used in an iteration is independent of all the previous pads and the message m. Let $r \in \mathcal{F}$ be a random field element. We claim that for any View $\mathcal{V}$ of $\mathcal{A}$, $\mathcal{V}$ occurs with the same probability in a transmission of m as in a transmission of $m' = m \oplus r$. Consider the case when the transmission of the pad ρ is successful. For the transmission of ρ, $\vec{s} = [\rho, \varphi_1, \ldots, \varphi_{e-1}]$ while during the transmission of ρ', $\vec{s'} = [\rho', \varphi'_1, \ldots, \varphi'_{e-1}]$. **S** sends $Z = m \oplus \rho = m \oplus (\vec{\lambda} \times M \times \vec{s}^{\,T}) = (m \oplus r) \oplus (r \oplus (\vec{\lambda} \times M \times \vec{s}^{\,T})) = (m') \oplus (\vec{\lambda} \times M \times \vec{s'}^{\,T}) = m' \oplus \rho'$. □

Lemma 3 (Communication Complexity). *Each iteration of the protocol communicates polynomial in* $(n + |\mathcal{A}|)$ *bits.*

Proof: From Theorem 1, it is clear that total number of paths used in the transmission protocol is polynomial in the size of $\mathcal{A}$. In **Stage 1** of every iteration, we have, **S** sends $O(|\Phi(\mathbf{S}, \mathbf{R})|)$ field elements to **R**. In **Stage 2**, **R** replies with $O((|\Phi(\mathbf{S}, \mathbf{R})|)^2)$ field elements. Since size of $\Phi(\mathbf{S}, \mathbf{R})$ is polynomial in the size of $\mathcal{A}$, the communication complexity of the protocol is polynomial in the size of the input. □

8 Conclusion

Network synchrony is a very difficult primitive to achieve in real-life, and more so in the presence of Byzantine faults in the system. This work initiates and completely characterizes the minimum connectivity requirements for secure communication over completely asynchronous networks (see Table 1). Furthermore, the choice of generalized mixed adversaries has meant that the necessary and sufficient conditions for secure communication over incomplete asynchronous networks for a variety of adversarial settings is studied in an unified manner and expressed in one-shot. The study of information-theoretically secure communication is far from closed. We have not considered the third kind of fundamental faulty behaviour, viz. fail-stop faults. Another open thread yet to be explored is to suitably adapt the protocols to (more practical) settings with lower amounts of synchrony though not completely asynchronous. Such networks are called *partially synchronous* networks. Yet another interesting setting is one in which the players possess only a partial knowledge of the topology of the network. More realistic adversary models are worth exploring and will have repercussions not only to the secure communication problem but also in the field of secure multiparty computation. Extant adversary models characterize a deviant player as either an *honest* player or a *dishonest* player. However, in real-life players being "fairly honest" and "slightly dishonest" makes sense. Viewing the honesty of the

	Perfect security	Unconditional security	Unconditional with Broadcast
Threshold Adversary	$\max(t_a, t_p) + 2t_a + 1$ [13]	$\max(t_a, t_p) + 2t_a + 1$	$2t_a + t_p + 1$
Generalized Adversary	$(\mathcal{A}^{(3,0)}$ & $\mathcal{A}^{(2,1)})$	$(\mathcal{A}^{(3,0)}$ & $\mathcal{A}^{(2,1)})$	$(\mathcal{A}^{(2,1)})$

Table 1. Necessary and sufficient connectivity requirements for the possibility of secure communication between any two honest players over arbitrary asynchronous networks.

players with this fuzzy outlook is worth exploring. Furthermore, among the efficiency considerations, it would be worth investigating the direct-sum question with respect to the communication as well as randomness complexities. Moreover, our protocols (for the perfect security case) are based on perfect linear secret schemes. This work does not investigate the deployment of non-linear secret sharing schemes that may prove to be more efficient (see [1]).

References

1. A. Beimel and Y. Ishai. On the power of nonlinear secret sharing. In *16th Annual IEEE Structure in Complexity Theory*, pages 188–202, 2001.
2. M. Ben-Or, R. Canetti, and O. Goldreich. Asynchronous secure computations. In *25th ACM STOC*, pages 52–61, 1993.
3. M. Ben-Or, B. Kelmer, and T. Rabin. Asynchronous secure computation with optimal resilience. In *13th ACM PODC*, pages 183–192, 1994.
4. C. Blundo and D. R. Stinson. Anonymous secret sharing schemes. *Discrete Applied Mathematics*, 77:13–28, 1997.
5. D. Dolev, C. Dwork, O. Waarts, and M. Yung. Perfectly secure message transmission. *JACM*, volume 40, number 1, pages 17–47, 1993.
6. M. Fischer, N. Lynch, and M. Paterson. Impossibility of distributed consensus with one faulty process. *JACM*, volume 32, number 2, pages 374–382, 1985.
7. M. Fitzi, M. Hirt, and U. Maurer. General adversaries in unconditional multiparty computation. In *ASIACRYPT'99*, volume 1716 of LNCS. Springer-Verlag, 1999.
8. M. Franklin and R. N. Wright. Secure communication in minimal connectivity models. *Journal of Cryptology*, 13(1):9–30, 2000.
9. M. Hirt and U. Maurer. Player simulation and general adversary structures in perfect multiparty computation. *Journal of Cryptology*, 13(1):31–60, April 2000. Preliminary version appeared in *16th ACM PODC*, pages 25–34, August 1997.
10. M. Ito, A. Saito, and T. Nishizeki. Secret sharing scheme realizing general access structure. In *IEEE Globecom 87*, pages 99–102. IEEE, 1987.
11. M. Karchmer and A. Wigderson. On span programs. In *8th IEEE Structure in Complexity Theory*, pages 102–111, 1993.
12. M.V.N. Ashwin Kumar, P.R. Goundan, K. Srinathan, and C.P. Rangan. On perfectly secure communication over arbitrary networks. In *21st ACM PODC*, 2002.
13. H. Sayeed and H. Abu-Amara. Perfectly secure message transmission in asynchronous networks. In *7th IEEE Symposium on Parallel and Distributed Processing*, 1995.