Reliable Broadcast in Unknown Fixed-Identity Networks

Lakshminarayanan Subramanian* Randy H. Katz* Volker Roth† Scott Shenker*‡
Ion Stoica*

ABSTRACT

In this paper, we formulate a new theoretical problem, namely the *reliable broadcast problem in unknown fixed-identity networks*. This problem arises in the context of developing decentralized security mechanisms in a specific-class of distributed systems: Consider an undirected graph G connecting n nodes where each node is *aware of only its neighbors* but not of the entire graph. Additionally, each node has a *unique identity* and cannot fake its identity to its neighbors. Assume that k among the n nodes act in an *adversarial* manner and the remaining $n-k$ are *good* nodes. Under what constraints does there exist a distributed algorithm Γ that enables every good node v to reliably broadcast a message $m(v)$ to all other good nodes in G? While good nodes follow the algorithm Γ, an adversary can additionally discard messages, generate spurious messages or collude with other adversaries.

In this paper, we prove two results on this problem. First, we provide a distributed algorithm Γ that can achieve reliable broadcast in an unknown fixed-identity network in the presence of k adversaries if G is $2k+1$ vertex connected. Additionally, a minimum vertex connectivity of $2k+1$ is a necessary condition for achieving reliable broadcast. Next, we study the problem of reliable broadcast in sparse networks ($1-$connected and 2-connected) in the presence of a single adversary *i.e.,* $k = 1$. In sparse networks, we show that a single adversary can partition the good nodes into groups such that nodes within a group can reliably broadcast to each other but nodes across groups cannot. For $1-$connected and $2-$connected graphs, we prove lower bounds on the number of such groups and provide a distributed algorithm to achieve these lower bounds. We also show that in a power-law random graph $G(n, \alpha)$, a single adversary can partition at most $O(n^{1/\alpha} \times (\log n)^{(5-\alpha)/(3-\alpha)})$ good nodes from the remaining set of good nodes.

Addressing this problem has practical implications to two real-world problems of paramount importance: (a) developing decentralized security measures to protect Internet routing against adversaries; (b) achieving decentralized public key distribution in static networks. Prior works on Byzantine agreement [17, 11, 23, 13, 3, 4, 24] are not applicable for this problem since they assume that either G is known, or that every pair of nodes can directly communicate, or that nodes use a key distribution infrastructure to sign messages. A solution to our problem can be extended to solve the byzantine agreement problem in unknown fixed-identity networks.

Categories and Subject Descriptors

C.2 [Computer communication networks]

General Terms

security, reliability, theory

Keywords

reliable broadcast, unknown network, byzantine agreement

1. INTRODUCTION

Reliable communication between nodes in the presence of byzantine adversaries is a fundamental problem in distributed systems [3, 4] that was first considered in the context of the classic Byzantine General's problem [23, 17]. Consider a network $G = (V, E)$ where the edges in E represent reliable channels between nodes in V. By reliable channels, we mean channels over which message transmissions cannot be dropped, tampered, or forged. In the simplest case, when G is a clique, reliable communication between every pair of nodes can be trivially achieved. For a general graph G, Dolev [11] and Dolev *et al.* [12] proved that if there are k faulty nodes, then every pair of nodes can reliably communicate if and only if the underlying graph is $2k + 1$ vertex connected. Biemel and Frankin [3] showed that the connectivity constraint can be relaxed if some pairs of nodes share authentication keys. However, if all nodes can be authenticated using a trusted keyed infrastructure, the problem of reliable communication becomes simple - any two nodes that have a path traversing non-faulty nodes can use *signed* messages to reliably communicate.

Existing works on reliable communication [17, 11, 23, 13, 3, 4, 24] assume that either the graph G is known or that nodes can use a key distribution infrastructure to sign messages.

*465 Soda Hall, EECS Department, University of California at Berkeley, Berkeley, CA-94720. Email:{lakme,randy,istoica}@eecs.berkeley.edu Phone: 510-642-8905

†Fraunhofer Institute, Germany. Email:vroth@igd.fhg.de

‡ICSI, Berkeley. Email:shenker@icsi.berkeley.edu

PODC'05, July 17–20, 2005, Las Vegas, Nevada, USA

There are many practical scenarios in distributed systems where neither of these assumptions hold. Motivated by this observation, this paper addresses the following question: *In the absence of a key distribution infrastructure, under what constraints can one achieve reliable communication when each node is aware of only its neighbors but not the entire graph G?*

1.1 Problem motivation

The Internet and many social networks that we operate in today, fall under the category of *unknown fixed-identity networks* satisfying two properties: (a) each node is assigned a unique identity; (b) the entire graph is not known nor published. For example, the Internet topology comprises of roughly $17,000$ Autonomous systems (AS) where every AS has a unique identity (AS number) assigned by IANA [1] which it cannot fake. When a new AS joins the network, it is only aware of its neighbors but is unaware of the AS topology. In fact, the complete AS graph structure of the Internet is unknown and is an open research problem to characterize the representativeness of the actual Internet topology collected from different measurement studies [8]. Domain Name System [21] and Intra-domain routing are two other real-world examples of unknown fixed-identity networks. Mobile ad-hoc networks [15] and P2P networks [22] are two examples of networks that do *not* belong to this category.

Reliable communication in the presence of adversaries is a fundamental necessity for improving the security of many of these networks. We use Internet routing as a motivating example. Today's Internet routing protocols are built on the basic assumption that all nodes in the network propagate truthful routing information [16, 28, 25]. A single compromised or mis-configured router acting in an adversarial manner by propagating spurious routing information can potentially affect reachability to a large fraction of the Internet [20, 18]. While several security measures have been proposed to improve the resilience of Internet routing [16, 25], none of them have moved towards adoption or deployment since these approaches typically require an extensive cryptographic key distribution infrastructure or a trusted central database [1]. Both of these ingredients are very hard to deploy in practice.

Any solution that secures Internet routing in the presence of adversaries should enable an AS to reliably broadcast its state to all the other nodes in the network [25]. With a key distribution infrastructure, this requirement is trivially met since every node can sign its messages, thereby making verification straightforward. In this paper, we focus on the problem of whether one can achieve this key distribution in a completely decentralized and distributed manner. We formulate this problem next.

1.2 Problem formulation

We use the following two definitions in our problem formulation.

Definition An *unknown fixed-identity network* $U(n, G, N)$ comprises of n nodes connected by an undirected graph G where each node: (a) has a unique identity it cannot fake; (b) knows the identities of its neighbors in G; (c) knows a value $N \geq n$ which represents a bound on the size of the network. For ease of convenience, we also refer to these networks as *unknown networks.*

Definition An *adversarial* node can perform three types of actions to disrupt reliable broadcast: (a) discard messages traversing the node; (b) generate spurious messages; (c) collude with other adversaries by exchanging information using out-of-band communication.

Reliable broadcast problem in unknown fixed-identity networks: *Consider an unknown fixed-identity network $U(n, G, N)$. Assume that k among the n nodes act in an* adversarial *manner and the remaining $n-k$ are* good *nodes that follow a prescribed algorithm. Under what constraints does there exist a distributed algorithm Γ that enables every good node A to reliably broadcast a message $m(A)$ to all other good nodes in G?*

Reliable broadcast is an instantiation of the reliable communication problem where a node intends to communicate the same message reliably to every other node in the network. Once reliable broadcast is achieved, one can perform pair-wise reliable communication by exchanging public keys through reliable broadcast. For this reason, we focus on the problem of reliable broadcast.

1.3 Summary of Results

The primary result we prove in this paper is:

THEOREM 1. *Given a bound k on the number of adversaries, there exists a distributed algorithm Γ that achieves reliable broadcast in an unknown fixed-identity network $U(n, G, N)$ if and only if G is $2k + 1$ vertex connected.*

This result extends the prior result of Dolev [11] for unknown fixed-identity networks. Dolev proved that a minimum $(2k + 1)$ vertex connectivity is essential for achieving reliable broadcast even if the entire graph, G, is known to all the nodes. Our result shows that one can achieve reliable broadcast even in the case where G is unknown to the nodes provided the graph satisfies the $(2k+1)$ connectivity requirement. The time-complexity of the algorithm is dependent on the values of k, N and is discussed in detail in Section 4.

The fixed-identity assumption is critical towards addressing this problem. If this assumption is not met and an adversary uses different identities to different neighbors, then we can show prove the following result:

LEMMA 1. *For any given integer $m > 0$, there exists an m-vertex connected network G on n nodes where each node is initially aware of the identities of only its neighbors, such that, a single adversary using multiple identities is sufficient to disrupt reliable broadcast in G.*

An alternative aspect of the problem arises for sparsely connected networks which do not satisfy the $(2k+1)$ connectivity requirement. In such networks, it is fundamentally impossible to achieve reliable broadcast. The best known result for sparse networks is the non-solvability of the problem [11]. However, we show that it is possible to *limit the damage* that an adversaries may cause in sparse networks.

In this paper, we study the reliable broadcast in sparse networks for the specific case of a single adversary ($k = 1$) and show optimality results for this case. Hence, we only deal with cases where the connectivity of the underlying network ($1-$connected and $2-$ connected) does not permit reliable broadcast even in the presence of a single adversary *i.e.,* $k = 1$. We quantify the damage that a single adversary can cause along two dimensions, as captured by the following definitions:

Definition An adversary is said to create a *broadcast partition of size* m, if it can classify the set of good nodes, V_g, into m groups $(X_1, \ldots X_m)$, such that the following constraints are met:

1. Each X_i is non-empty and $V_g = X_1 \cup \ldots \cup X_m$.
2. Nodes within X_i can reliably broadcast to other nodes in X_i but not to nodes in X_j for $j \neq i$.

Definition Given a broadcast partition $B = (X_1, \ldots X_m)$ of V_g created by an adversary w, the *cumulative damage* caused by w is given by $min_i|V_g - X_i|$. If X_j represents the largest broadcast group in B, then the cumulative damage represents the number of nodes that cannot communicate with X_j.

The cumulative damage of an adversary measures the number of nodes that a single adversary can affect. If we consider Internet routing as an example, this metric represents the lower bound on the number of nodes that an adversary can affect by propagating bogus messages. We prove the following result for reliable broadcast on sparse networks:

THEOREM 2. *Given that an unknown fixed-identity network $U(n, G, N)$ has exactly a single adversary A with degree $d(A)$, the following statements hold:*

1. *If G is $1-$connected, A can create a broadcast partition of size at most $2 \times d(A)$.*
2. *If G is $2-$connected, A can create a broadcast partition of size at most $d(A)$.*

These results are optimal in the sense that these represent not only the lower-bound on the amount of damage that an adversary can cause but also a tight upper-bound. We provide an algorithm that restricts the amount of damage an adversary can cause and achieves the lower bound. While Theorem 2 provides only a bound on the size of a broadcast partition, it does not provide a bound on the cumulative damage caused by a single adversary. One can construct $1-$connected and $2-$connected graphs where the cumulative damage is $O(n)$. However, we show that the cumulative damage of an adversary in a power-law random graph [2, 9] is much smaller than n as summarized below:

THEOREM 3. *Given an unknown fixed-identity network $U(n, G, N)$ where G is a power-law random graph on n vertices with parameter α satisfying $2 < \alpha < 3$, the cumulative damage caused by a single adversary is bounded by $O(n^{1/\alpha} \times (\log n)^{(5-\alpha)/(3-\alpha)})$ with high probability.*

This result has important practical implications for the Internet and many social networks that exhibit structural similarities to power-law random graphs [2]. In such types of networks, the cumulative damage that an adversary may cause is very small compared to the size of the network.

1.4 Related Work

The problem of reliable broadcast (RB) is closely related to the classic problem of Byzantine agreement (BA) first proposed in [23, 17]. More recent works on the problem of byzantine agreement include [11, 13, 12]. We summarize the relationship between the problem of reliable broadcast to the Byzantine agreement problem with the following observation:

Observation: Given n nodes of which k are adversarial, then two results hold: (a) BA $\implies$ RB; (b) If RB is achievable, then one can achieve BA if $n \geq 3k + 1$.

The first result implicitly follows from the fact that $\neg RB \implies \neg BA$. If two good nodes cannot reliably transmit messages between themselves, then they cannot achieve BA. The second result indirectly follows from previous works by Lamport *et al.* [17, 23] and Dolev [11].

In the seminal work by Lamport [17, 23], a protocol for Byzantine agreement was proposed under the assumption that either G is known or that every pair of nodes can directly communicate or that nodes use a key distribution infrastructure to sign messages. These basic assumptions continue to be necessary in later works [11, 12, 13]. The relationship between BA and RB is also implied in Dolev's result [11] (the purifying algorithm presented in Dolev's paper achieves RB when G is known).

An identity-based cryptosystem [26] is an alternative proposal where the identity of a node indicates its public key. Recent work on identity based encryption (IBE) [6, 10] represent the most general implementation of a usable identity-based cryptosystem. While one can envision using IBE to achieve reliable broadcast, IBE requires a trusted master server to be entrusted with the private key of all participants. This assumption is, again, undesirable in large-scale distributed systems.

2. PATH VECTOR SIGNATURES

In this section, we describe the concept of *path vector signatures*, one of the basic building blocks we use to solve the problem of reliable broadcast. A path-vector signature is a signature associated with a message that traverses a particular path within the network. These signatures enable a good node to *differentiate* between genuine messages generated by good nodes from spurious ones generated by adversaries. An adversary (or a set of adversaries) that intends to disrupt a good node v from reliably communicating a message $m(v)$, will attempt to propagate spurious messages $m'(v)$ claiming to be from v. To defend against such adversaries, we associate with each message a specific path-vector signature that is cryptographically computed and updated by every node along the path through which the message is propagated, so that no adversary can tamper with the message. Hence, an adversary intending to propagate a spurious message claiming to be from a good node v is forced to generate a *different signature* in comparison to the same message being generated by the source.

More formally, a *path-vector message* (m, s, p) consists of three parameters: a message m, the identity of the source s, and a path p containing the identities of the nodes the message traverses including the source s. A *path-vector signature*, $sgn(m, s, p)$, is a signature corresponding to a path-vector message (m, s, p) which is initiated by the source s and incrementally updated by every node along the path p. It is important to note that if a node u propagates a path-vector message (m, s, p) to v, then v's identity is already appended to the path p by u signifying that u has propagated the message to v. Hence, a node v that receives a message should have its identity as the last node in the path and cannot remove its identity (in case, v is an adversary). The path-vector signature, $sgn(m, s, p)$, should satisfy three properties:

1. **Verify:** Given (m, s, p) and $sgn(m, s, p)$, any node should be able to verify that the message traversed the nodes in path p provided the message m was initiated at s.

2. **Append an identity:** Let a node with identity x receives a message (m, s, p) along with $sgn(m, s, p)$. If x intends to forward the message to a neighbor with identity y, x should be able to compute the valid signature $sgn(m, s, p')$, for the message (m, s, p'), where p' is the path $(p, \{y\})$.
3. **Inability to modify:** Given a path-vector message, (m, s, p), an adversary *should not* be able to produce a valid signature for any message (m', s, p') where $m' \neq m$ or p' is not a path of the form (p, p_f) where p_f is any other path of identities. In other words, the adversary can append identities to the path but not remove identities.

We now discuss a simple path signature construction that satisfies these requirements. This construction relies on an underlying conventional public-key signature scheme G, where $G(m, P)$ refers to the message m signed using the public key P. There are several known schemes that can be used for this purpose, one example being the El Gamal signature scheme [14]. Consider a node v_1 sending a message m to node v_n over the path $(v_1, \ldots v_n)$. Let each node v_i generate a public key $g(v_i)$. The prescriptions of our protocol are as follows:

1. *Initialization:* Every node v_i generates its public key $g(v_i)$, and for $(i > 1)$, communicates it to its neighbor v_{i-1}.
2. *Message Initiation:* The source v_1 sends the message $m_1 = [\ (m, s, p_1), sgn_1\ ]$ to its neighbor v_2 where $s = (v_1, g(v_1))$, $p_1 = [\ (v_1, g(v_1)), (v_2, g(v_2))\ ]$, and $sgn_1 = \{G((m, s, p_1), g(v_1))\}$.
3. *Incremental update:* Node v_i $(i > 1)$ receives message $m_{i-1} = [\ (m, s, p_{i-1}), sgn_{i-1}\ ]$ from its predecessor v_{i-1}. It then sends message $m_i = [\ (m, s, p_i), sgn_i\ ]$ to its successor v_{i+1} where $p_i = [\ p_{i-1}, (v_{i+1}, g(v_{i+1}))\ ]$ and $sgn_i = \{sgn_{i-1}, G((m, s, p_i), g(v_i))\}$.

Note that each node v_i includes the identity of its successor v_{i+1} in its message m_i. This identity also includes the public key announced by v_{i+1}. Thus, in essence, the message received by node v_i consists of the original message m, the identity of originator s along with its claimed public key $g(v_i)$, and a path signature where the identity and public key of each hop is certified by its predecessor. Any node v_i on the path that receives a message (m, s, p) along with $sgn(m, s, p)$ can verify the correctness of each individual signature according to the signature scheme G. This construction satisfies the following lemma:

LEMMA 2. *Any fake path vector message $M = (m, s, p)$ generated by an adversarial node with a genuine signature $sgn(M)$ can only be one of two categories:*

1. *M was generated by a single adversary v which generated fake public keys $g'(u)$ for all identities u that precede v in p.*
2. *Two colluding adversaries v, w that occur in p can insert a spurious path fragment comprising arbitrary identities (including identities of good nodes) between v and w in the path. For each such good node x whose identity was added by v and w, the adversarial nodes need to generate a fake public key $g'(x)$.*

It is essential for every node along the path to sign every message for this lemma to hold and distinguish any fake message. If not, an adversary can insert arbitrary path-fragments with identities and this can perturb the graph-computation process described in Section 3. This motivates the concept of a **keyed-identity** of a node denoted as $(x, g(x))$, where x is the identity and $g(x)$, the claimed public key of x in a message. Every path-vector message contains a string of keyed-identities. Any message with the keyed-identity $(x, g(x))$ is distinctly different from a message containing the keyed identity $(x, g'(x))$, of which, certainly one of the messages is bogus (since good nodes do not claim conflicting public keys). However, given only these two messages, a receiving node cannot immediately determine as to whether $g(x)$ or $g'(x)$ is the genuine public-key of x. We address this problem of determining the genuine keyed-identity in the next section.

3. RELIABLE BROADCAST ALGORITHM

Based on the concept of path-vector signatures, we describe our reliable broadcast algorithm in this section. The algorithm uses two main ideas:

Keyed-identity graph computation: Every good node x uses the information from path-vector messages to continuously compute a keyed-identity graph G_x, where the nodes in the graph are of the form $(v, g(v))$, comprised of the actual identity v and the claimed public key $g(v)$. To prevent unnecessary path exploration, a path-vector message that contains no additional information(no new edge or vertex) is *not propagated further.*

Determining genuine identities: If the underlying graph G is $2k + 1$ vertex connected with k adversaries, then, between every pair of good nodes, there exists at least $k + 1$ vertex disjoint paths that traverse only good nodes. Hence, in the keyed-identity graph, G_x, if the number of *identity-disjoint* paths to a keyed-identity $(v, g(v))$ is at least $(k+1)$, then x can conclude $g(v)$ to be the genuine public-key corresponding to identity v. Any bogus keyed identity $(v, g'(v))$ generated by adversaries can at most traverse k vertex disjoint paths since the identity of at least one adversary should be present in each path. If a message is not signed by every node along the path, an adversary can generate spurious edges and disrupt the computation of disjoint paths. The fact that adversaries can at most prove k disjoint paths to a fake node is critical for the solvability of this problem.

3.1 Asynchronous Broadcast Algorithm

Based on these two ideas, we describe an asynchronous algorithm to achieve reliable broadcast. Given a path-vector message (m, s, p) and its signature, we define the *keyed identity path* $P_I(m, s, p)$ associated with (m, s, p) to consist of vertices $(v_i, g(v_i))$, where v_i is the identity of a node in p and $g(v_i)$ is the public key of v_i in the signature. We use G_x to denote the keyed-identity graph computed by a node x with a set of neighbors $N(x)$. Every good node x performs the following set of operations.

BROADCAST(Node x, Neighbors $N(x)$)

1. *Asynchronous node wakeup:* A node can either begin broadcast by itself or begin transmissions upon receipt of the first message from a neighbor.
2. *Initiation:* G_x consists of one vertex $(x, g(x))$.
3. For every $u \in N(x)$, x transmits $(m(x), x, [x, u])$ to u along with its signature.
4. *Propagation:* For every path-vector message (m, s, p) with signature S that x receives from $u \in N(x)$, x performs:
   (a) *Immediate-neighbor key check:* Check if public-

key of u in S matches the same public-key used in previous messages. If not, reject (m, s, p). If $v \in N(x) - \{u\}$ appears in p, then the public-key of v should also match the one directly advertised by v.

(b) Verify S.

(c) *Learn one vertex at a time:* Accept the message only if $P_I(m, s, p)$ contains at most one new keyed identity (at the end of the path) not present in G_x. If so, update G_x with $P_I(m, s, p)$.

(d) *Message suppression:* If $P_I(m, s, p)$ adds no new vertices or edges to G_x, ignore the message.

(e) To every $u \in N(x)$, x transmits (m, s, p') where $p' = p \cup \{u\}$ after updating the signature.

5. *Flow computation:* If the number of identity-disjoint paths to $(v, g(v))$ in G_x is at least $k+1$, then x deems v to be a genuine identity and $g(v)$ to be its public key. By identity disjoint paths, we mean that no two paths should contain two different vertices $(v, g(v))$ and $(v, g'(v))$ which share the same identity v.

The immediate-neighbor key check is necessary to ensure that if an adversary $v \in N(x)$, then v uses only a single keyed-identity $(v, g(v))$ in all its messages propagated to x. Any other message that x receives (from other neighbors) which contains the identity v is accepted only if it contains the same public key $g(v)$.

3.2 Proof of Theorem 1: asynchronous version

In this section, we will prove Theorem 1 and show that the asynchronous BROADCAST algorithm will eventually achieve reliable broadcast in an unknown network $U(n, G, N)$ if and only if G is $2k+1$ vertex connected. This proof consists of two parts. First, we establish the requirement that a minimum $2k+1$ vertex connectivity is necessary to achieve reliable broadcast. Next, we show how the BROADCAST algorithm achieves reliable broadcast.

Minimum (2k+1) connectivity requirement: Consider a graph H that is $2k$ vertex connected with k adversaries. Consider any vertex cut C of size $2k$ containing the k adversaries that separates H into two components A and B. The k adversaries can prevent nodes in A from reliably broadcasting to nodes in B by modifying every message $m(u)$ from $u \in A$ to $m'(u)$. Nodes in B cannot determine whether $m(u)$ or $m'(u)$ is genuine. Therefore, $2k$ vertex connectivity is insufficient to achieve reliable broadcast in the presence of k adversaries.

BROADCAST analysis: Let every good node execute the BROADCAST algorithm to broadcast $m(x)$. Consider a particular good node x. For every other good node v with public-key $g(v)$, if $(v, g(v))$ is a vertex in G_x, then x would have learnt the message $m(v)$ in the first path-vector message where $(v, g(v))$ is added to G_x since $(v, g(v))$ was the last node in that message.

Let $G_g = (V_g, E_g)$ represent the sub-graph comprising of all the edges between the set of good nodes. Given that G is $2k+1$ vertex connected with at most k adversaries, G_g is at least $k+1$ vertex connected. If every good node x can learn all the edges in E_g, then it can definitely compute $(k+1)$ identity-disjoint paths to every other good node and hence, can successfully determine every other good node v, its public key $g(v)$ and message $m(v)$. To show that the BROADCAST algorithm achieves reliable broadcast, we need to show that every good node will learn all edges in E_g.

To prove that each good node will eventually learn E_g, let us separate the message exchange process between neighboring nodes into *rounds*. In round i, let each node exchange the new path-vector messages it learnt in round $i-1$ with its neighbors. Within each round i, every node will learn all nodes within a distance i from the node. Hence, within $O(n)$ rounds every good node will learn all edges in E_g and hence discover all the genuine nodes in G. Finally, the following simple result on message complexity holds:

LEMMA 3. *In an unknown network $U(n, G, N)$, let G comprise of e edges. When $k = 0$, each node transmits at most e path-vector messages to each neighbor using the BROADCAST algorithm.*

This result trivially follows from the message suppression step in the BROADCAST algorithm. This step ensures that for each edge in G, a good node propagates only one path-vector message. In the absence of any adversary ($k = 0$), the number of path-vector messages along each link is bounded by e. In the presence of adversaries ($k > 0$) generating fake messages, the number of path-vector messages along each link depends on k. We discuss this complexity in Section 4.

3.3 Multiple Identities: Proof of Lemma 1

Consider the case where an adversarial node uses different identities to different neighbors. A single adversary using multiple identities is sufficient to disrupt reliable broadcast in certain types of networks however large the connectivity may be. For any positive integer value $m > 0$, one can construct a graph H consisting of $n(> m)$ vertices which satisfies the following two constraints:

1. H is $2m-2$ vertex-connected but not $2m-1$ vertex-connected.
2. H contains a vertex-cut C such that $m-1$ vertices $A_1, \ldots A_{m-1} \in C$ have non-overlapping neighbor-sets $N(A_1), \ldots N(A_{m-1})$.

In essence if $A_1, \ldots A_{m-1}$ act as adversaries, then they can disrupt reliable broadcast in H. Given one such graph H, construct a new graph H' where the vertices $A_1, \ldots A_{m-1}$ are collapsed into a single vertex A whose set of neighbors represent the union of the set of neighbors of $A_1, \ldots A_{m-1}$. Clearly H' is m-connected. One can show that a single adversarial node A using different identities is sufficient to disrupt reliable broadcast in H'. To do so, the adversary A uses different identities to its neighbors in $N(A_i)$ for each $0 < i < m$, thereby creating an underlying graph that resembles H.

4. COMPLEXITY ANALYSIS

In this section, we consider a synchronous version of the BROADCAST algorithm and analyze its time complexity. One needs to be careful when analyzing the time-complexity of an algorithm in an unknown fixed identity network. The traditional Byzantine agreement literature uses the concept of rounds [11, 17, 13, 19] to analyze time-complexity. However, given that the entire network is unknown, enforcing the global concept of a round is not feasible. On the other hand, a completely asynchronous mode of communication [5, 19, 7] is also not suitable for our analysis since it is not possible to provide time guarantees for message deliveries. Hence, we revert to the traditional synchronous model and *locally* enforce the concept of time by imposing capacity constraints on links based on the following definition:

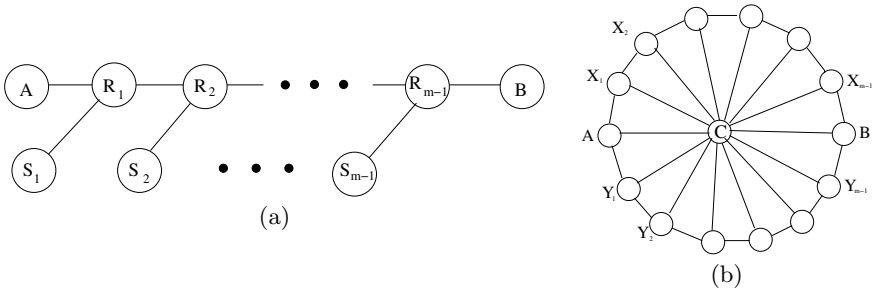

Figure 1: (a) Packet delay along a linear path. (b) Example topology for G with $2m+1$ nodes with C being an adversarial node.

Definition A network G is said to be *capacity-constrained* if every node can only transmit $O(1)$ bits of information to each of its neighbors in a single unit of time.

Capacity constraints enable us to loosely enforce the concept of a round globally while every node operates locally at its own link-capacity rates. By enforcing an $O(1)$ capacity constraint on each link, we ensure that the ratio of the time to deliver a message along a link is $O(1)$. While we analyze the time complexity of our algorithms based on the capacity-constrained assumption, it is conceivable that alternative measures of time may apply.

4.1 Message scheduling algorithm

To produce a synchronous version of the BROADCAST algorithm for a capacity-constrained network, it is essential to determine the mechanism used to schedule messages at every node. In a capacity constrained network, each node receives multiple messages from each neighbor but can propagate only one message on each outgoing link. Hence, each node needs to buffer messages and use a scheduling algorithm to prioritize the messages to be transmitted. The lemma described below shows that using a simple FIFO scheduling algorithm does not suffice:

LEMMA 4. *If every node uses a simple First-in-First-out (FIFO) queue with an infinite buffer to schedule messages on each link, there exists an unknown network, $U(n, G, N)$, where G is $3-$connected such that the minimum time complexity of any algorithm to achieve reliable broadcast in the presence of a single adversary is $2^{(n-3)/2}$.*

Proof: We first show that in the particular topology illustrated in Figure 1, the delay incurred in transmitting a single message from A to B separated by m capacity-constrained hops can be as high as 2^{m-1} if intermediary nodes use FIFO queues. Let this topology be capacity constrained in that every node can transmit only one message along every link in unit time. Let all nodes begin transmission at time $t = 0$ with all queues initially being empty. Now, assume that the nodes S_i connected to R_i continuously transmit one message every unit time. In this case, if all R_i's use FIFO queuing, a single message from A to B will incur a worst-case delay of 2^{m-1}. Since each S_i transmits one packet per unit time to R_i, in the worst-case, A's packet reaches R_2 from R_1 at time $t = 2$, reaches R_3 at $t = 4$ and reaches R_{i+1} at $t = 2^i$. Hence, the bound 2^{m-1}.

Next, using the previous result we construct a topology where reliable broadcast with FIFO queues has a minimum time complexity of $2^{(n-3)/2}$. Consider a modified topology (as shown in Figure 1(b)) with $2m+1$ nodes comprising of a loop of $2m$ nodes and a central node C connected to all the nodes in the loop. Now let node C be the only adversarial node that continuously injects fake path-vector messages on each of its links. Each such fake message is generated from a non-existent vertex and also contains an arbitrary non-existent path. Now, two good nodes A and B can only communicate through the two paths in the loop of length m. By the previous argument, if all nodes use a FIFO queue with an infinite buffer, any message from A to B will incur a minimum delay of 2^{m-1}. However, any algorithm that achieves reliable broadcast should enable at least one message to be communicated from A to B. Hence, the minimum time complexity of any such algorithm using FIFO queues is lower bounded by $2^{m-1} = 2^{(n-3)/2}$. □

The above argument can also be directly extended to the Fair Queuing discipline where a node divides a link's capacity equally amongst all its other neighbors. In the above example, FIFO scheduling and Fair queuing does not work primarily because an adversarial node can flood the network with spurious messages and delay the delivery of packets.

4.1.1 Identity-based rate limiting

The goals of identity-based rate-limiting are two-fold: (a) hold nodes accountable for every message they transmit and limit the capability of adversaries to flood messages; (b) associate a higher priority to the messages from lesser-known nodes which have not received enough opportunities to transmit messages. To achieve these, the key idea is to *rate-limit messages* across identities and keyed-identities. To do so, the algorithm computes two simple metrics:

1. *Identity priority:* Let $I_1(u,t)$ represent the number of path-vector messages transmitted prior to time t with identity u in the path. A message (m,s,p) has an identity priority $p_1(m,s,p)$ which is the maximum value of $I_1(u,t)$ for all identities $u \in p$.
2. *Keyed-identity priority:* Let $I_2((u,g(u)),t)$ represent the number of messages transmitted prior to time t with keyed identity $(u,g(u))$ in the keyed-identity path. The keyed-identity priority $p_2(m,s,p)$ of a message is the maximum value of $I_2((u,g(u)),t)$ over all keyed identities $(u,g(u))$ in the path.

The scheduling strategy is as follows:

1. Schedule message (m,s,p) with the lowest identity priority, $p_1(m,s,p)$.
2. If multiple messages have the same minimum identity priority, schedule the message among them with the lowest keyed-identity priority, $p_2(m,s,p)$.
3. Use FIFO as a final tie-breaking rule.

The rationale behind maintaining two separate priorities is two-fold. First, the *identity-based priority* is essential to prevent an adversary to use multiple keyed identities and propagate spurious messages. For example if we use only keyed-identity priorities in the topology illustrated in Figure 1(b), the lower bound argument used in the case of FIFO holds in this case too. Second, the keyed-identity based count is essential because an adversary can artificially increase the identity-based count of a good node by inserting the good node in a path. However, an adversary cannot artificially increment the keyed-identity based count of a node.

4.2 BROADCAST algorithm complexity

We use $G_g = (V_g, E_g)$ to represent the sub-graph of G comprising of only the good nodes V_g and E_g, the set of edges between them. Assuming G_g is connected, let $diam(G_g)$ denote its diameter. We prove the following theorem on the BROADCAST algorithm.

THEOREM 4. *Given a bound k on the number of adversaries and a bound Δ on $diam(G_g)$ in an unknown network $U(n, G, N)$, the BROADCAST algorithm with identity-rate limiting (IRL) achieves reliable broadcast in:*

1. $O(N^2 n \log n)$ *time for* $k = 0$.
2. $O(N^3 n \log n)$ *time for* $k = 1$.
3. $O((k+1)^{\Delta} N^2 n \log n)$ *time for* $k > 1$.

PROOF. In a capacity constrained network, a node can transmit only $O(1)$ bits per unit time. For simplicity, we bound the total number of bits consumed by every path-vector message by $O(n \log n)$ ($\log n$ bits for each identity and n for the maximum number of signatures in a message). We analyze the complexity separately for three cases:

LEMMA 5. *When $k = 0$, BROADCAST + IRL achieves reliable broadcast in $O(ne \log n)$ time where e is the number of edges in G.*

Proof: If the first node initiates transmission at time $t = 0$, every node will receive a wake up signal after n path-vector message transmissions given that the diameter of G is bounded by n. After wakeup, the message suppression step in the BROADCAST algorithm ensures that every node learns G after e message transmissions along each edge. Since each node will always have a new path-vector message to transmit and that each message has $O(n \log n)$ time complexity for transmission on a link, BROADCAST requires $O(en \log n)$ time to converge. □

LEMMA 6. *When $k = 1$, BROADCAST+IRL achieves reliable broadcast in $O(n^2 e \log n)$ time.*

Proof: We first analyze a simple scenario as illustrated in Figure 2(a) where nodes A and B are $m + 1$ hops away and A propagates one message to B in the presence of an adversary X that continuously injects packets along each hop in the path (No other node propagates any message in this example). If all transmissions begin at $t = 0$ with the identity based counts set to zero, the IRL algorithm will ensure that for every message that A propagates, X can at most insert m messages, one along each of the links XR_i. Hence, A's message is delivered to B within $(m + 1)n \log n$ time assuming a simple upper bound of $n \log n$ bits for each message.

In the general case, from the proof of Lemma 5, we know that every good node needs to receive e different path-vector messages to completely learn the good graph G_g. To analyze the worst-case time bound, we analyze each such message in isolation. Each message (m, s, p) that traverses a path p comprising of $|p|$ good nodes experiences a delay of $(|p|+1)n \log n$ since an adversary X can inject spurious messages (either directly or indirectly) for each node along p. $|p|$ represents the length of path p and is clearly bounded by $n - 1$. Hence, each message in isolation if delivered one at a time is delivered to a good node in $O(n^2 \log n)$ time. The time to deliver e messages is bounded by $O(n^2 e \log n)$ time. □

LEMMA 7. *For a general value k, given a bound $\Delta \geq diam(G_g)$, using BROADCAST+IRL, every good node achieves reliable broadcast after $O((k+1)^{\Delta} en \log n)$ time.*

Proof Sketch: Much like Lemma 6, we consider a simple scenario of two good nodes A and B separated by m hops (as shown in Figure 2(b)), except that the adversaries $X_1, \ldots, X_k$ continuously inject messages at every node R_i along the path. Unlike Lemma 6, we show that the delay incurred by every message from A in this case can be as high as $O((k+1)^m \times n \log n)$. In the general case, given a bound Δ on $diam(G_g)$, the information about every edge in G_g can be transmitted along a path of at most Δ hops. Hence, the maximum time required to discover e edges in G_g is bounded by $O((k+1)^{\Delta} en \log n)$. □

Stoppage constraint: The final element of the proof is the stoppage constraint. Since no node is aware of the values of n and e, each good node cannot determine when the full graph is learned. We use the bound N to represent a bound on n and N^2 as a bound on e, the number of edges. These bounds need to be applied only on the number of messages but not on the size of each message. Hence, we can still retain the $O(n \log n)$ bound on the maximum time complexity of a single message. Replacing these bounds in Lemmas 5, 6 and 7 completes the proof.

4.3 Lower-bound time complexity

We show the following lower-bound on the time complexity of any reliable broadcast algorithm.

LEMMA 8. *Given a bound k on the number of adversaries, there exists a capacity constrained network $U(n, G, N)$ where G is $2k + 1$ vertex connected, such that any algorithm that achieves reliable broadcast in $U(n, G, N)$ has a minimum time complexity of 2^k.*

We refer the reader to our technical report [27] for a proof of this result. Notice the wide gap between the lower-bound result and the time complexity of our algorithm. Addressing this complexity gap is an open research problem; our work primarily illustrates the existence of an algorithm to achieve reliable broadcast but does not target optimality.

4.4 Limitations of capacity-constraints

Analyzing the time complexity of a distributed algorithm with capacity constraints on link-topologies is a very restrictive model. Lemma 4 illustrates the limitation of FIFO in the face of a single adversary, where a single message transmission can be exponentially complex with capacity constraints. In other words, many simple distributed algorithms including the emulation of a single round in a Byzantine agreement

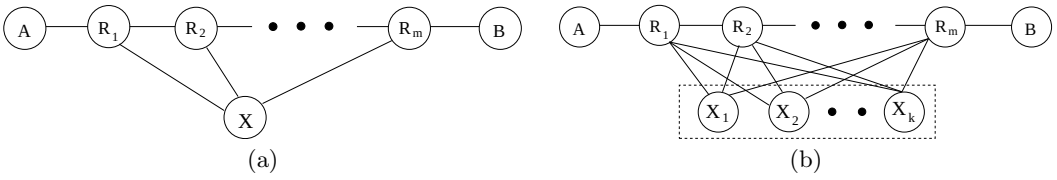

Figure 2: (a) Single adversary case. (b) Example topology for the case when $k > 1$.

algorithm has exponential complexity. Given this, the bound for the case $k > 1$ of the BROADCAST algorithm should be viewed as a worst-case bound (where an adversary generates infinite bogus message) which is not completely reflective of the real-life scenario. In a realistic setting, we would expect the asynchronous version of the BROADCAST algorithm to have much lower run-time complexity and anticipate an adversary to generate only a finite number of bogus announcements. With finite bogus announcements from adversaries, the complexity is polynomial in the number of nodes with capacity constraints.

5. SPARSE NETWORKS

In this section, we describe our results for the problem of reliable broadcast in 1−connected and 2−connected graphs in the presence of a single adversary *i.e.,* $k = 1$.

5.1 Lower bound of Theorem 2

In this section, we show the existence of 1−connected and 2−connected graphs such that a single adversary with a degree d can create broadcast partitions of sizes $2d$ and d respectively in these graphs. This establishes the lower bound on the size of a broadcast partition that a single adversary can create in 1-connected and 2−connected graphs. In this analysis, we assume that the adversary is aware of G while good nodes are aware of only their neighbors. This assumption is valid since the adversary can first learn about G from good nodes and then propagate spurious messages. The following lemmas establish the lower-bound.

LEMMA 9. *There exists a tree G such that an adversarial node A in G with degree d can create a broadcast partition of size $2d$ in $U(n, G, N)$.*

Proof: We first analyze the case when $d = 1$. Let G_0 be a tree rooted at a node r (refer Figure 3(a)). Let v be a child of r such that the sub-tree rooted at v is non-empty and does not contain the adversarial node A. Let $T(v)$ represent the set of nodes in the sub-tree rooted at v (excluding v) and let $T'(v) = V - T(v) - \{v, A\}$. Pick any tree G_0 such that $T(v)$ and $T'(v)$ are non-empty. For every node $u \in T(v)$ that attempts to reliably broadcast a message $m(u)$, let A propagate a spurious message $m'(u)$ as if u propagated the message (as illustrated in Figure 3(a)). Since no node in $T'(v)$ is directly adjacent to any node in $T(v)$, these nodes will receive two messages $m(u)$ and $m'(u)$ and cannot figure out which message is genuine. All genuine messages from $T(v)$ are routed through v and all spurious messages $m'(u)$ are routed through A and nodes in $T'(v)$ cannot determine which node to believe. Hence no node in $T'(v)$ can reliably communicate with any node in $T(v)$. Thereby, A creates a broadcast partition $(T(v) \cup \{v\}, T'(v) \cup \{v\})$ of size 2.
For the general case, we can create a tree G with d replicas of G_0 with the vertex A merged across all these replicas (as illustrated in Figure 3(b)). In this case, by simply dropping all good messages traversing it, A can create d separate components which cannot reliably broadcast to each other. Within each such component, A acts as a leaf node and can create a broadcast partition of size 2. Hence, A can create a broadcast partition of size $2d$. □

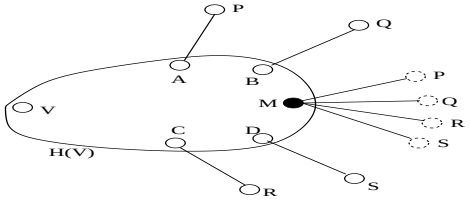

Figure 4: An example topology for the penalty based defense strategy.

LEMMA 10. *There exists a 2-connected graph G such that an adversarial node A in G with degree d can create a broadcast partition of size d in $U(n, G, N)$.*

Proof: Consider the graph G illustrated in Figure 3(c), where $P_1, \ldots P_d$ are paths of good nodes such that the end-points of each path connect to two separate vertices A and v. In this case, A has degree d. For every node $u_i \in P_i$ that intends to reliably broadcast a message $m(u_i)$, A propagates a message $m'(u_i)$ to all its neighbors in P_j where $j \neq i$. Hence, every node $u_j \in P_j$ will receive two messages: $m(u_i)$ (through v) and $m'(u_i)$; hence for every $u_i \in P_i$ $(i \neq j)$, u_j cannot ascertain which of the two messages is genuine. Therefore, no pair of nodes (u_i, u_j) , $u_i \in P_i$ and $u_j \in P_j, j \neq i$, can reliably communicate. A creates a broadcast partition $(P_1 \cup \{v\}, \ldots, P_d \cup \{v\})$ of size d. □

5.2 Defensive strategy

Now, we describe an optimal penalty-based defensive strategy that a good node uses to limit the size of a broadcast partition that an adversary can cause to the lower bound. This mechanism works only for $k = 1$. The defensive strategy uses one key corollary that directly follows from Theorem 1:

Corollary 1: *Let H be a 2−connected subgraph of G in an unknown network $U(n, G, N)$, comprising of only good nodes. In the presence of a single adversary, M, every node in H can reliably broadcast to all nodes in H.*

We explain the intuitive idea behind the penalty-based defense strategy using an example illustrated in Figure 4. Let $H(v)$ represent the set of all keyed-identities that have a 2 identity disjoint paths to v. An adversary M cannot disrupt any of these nodes. In this example, however, M attempts to disrupt reliable communication to multiple nodes P, Q, R, S which connect (directly or indirectly) using good paths to different nodes A, B, C and D in $H(v)$. From the perspective of v, M and A disagree on P, M and B disagree on Q,

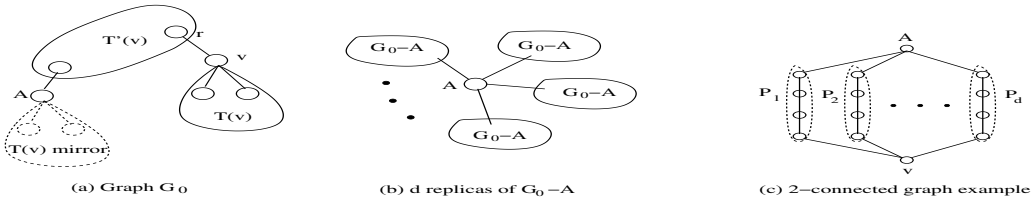

Figure 3: Lower bound analysis for sparse graphs: (a) An example tree G_0 with adversary A being a leaf node. (b) Example tree with adversary A having degree d. (c) Example 2-connected graph.

M and C on R and finally M and D on S. Hence, if v associates a penalty with a node for every identity-disagreement, then M obtains the maximum penalty of 4 while the nodes A, B, C, D each obtain a penalty of 1. Therefore, v notes M to be a suspicious candidate and filters M's messages and chooses the genuine identities propagated by A, B, C, D.

Now, we present the penalty-based algorithm based on the idea explained above. Initially, every node x executes the BROADCAST+IRL algorithm for the case of $k = 1$ and computes the keyed-identity graph G_x. Next, they apply the following penalty algorithm on G_x.

PENALTY (Node x, Graph G_x)

1. We declare a keyed-identity $(y, g(y))$ to be *genuine* if x has two identity disjoint paths to $(y, g(y))$.
2. The *tail-end* of a keyed-identity $(v, g(v))$ is the path from the node $(y, g(y))$ to $(v, g(v))$ in G_x such that $(y, g(y))$ is the last genuine node in any path from x to $(v, g(v))$.
3. An identity u has a *conflict* if there are at least *two* keyed identities with the same identity in G_x.
4. **Penalty assignment:** The *penalty* of an identity u is the number of distinct conflicting identities y for which some keyed-identity $(u, g'(u))$ appears in the tail-end of a keyed-identity $(y, g(y))$.

Node-selection criteria: Based on the penalties assigned to identities, the criteria for selecting nodes is simple. For each keyed identity, $(v, g(v))$ that has a conflict, determine the *maximum penalty* of the identity u that appears in the tail-end of $(v, g(v))$. Choose the keyed-identity with the *minimum value of the maximum penalty*. If no unique minimum exists, then we declare the identity as non-identifiable.

Revisiting the example in Figure 4, v will notice P, Q, R and S to be conflicting identities. The genuine keyed-identity of P propagated by M will have a maximum penalty of 1 while the fake keyed-identity generated by the adversary M will have a penalty of 4. Hence, the penalty based algorithm chooses the genuine keyed-identities for P, Q, R, S.

The following lemma holds regarding the penalty-based algorithm:

LEMMA 11. *In an unknown network $U(n, G, N)$, if all good nodes use the BROADCAST+IRL+PENALTY algorithm for determining genuine nodes, a single adversary with degree $d(A)$ can create a broadcast partition of size at most: (a) $d(A)$ if G is $2-$connected; (b) $2 \times d(A)$ if G is $1-$connected.*

The proof of Lemma 11 is presented in our technical report [27]. This lemma completes the proof of Theorem 2.

5.3 Proof sketch of Theorem 3

Finally, we provide a proof sketch for Theorem 3 described in Section 1.3 where we show that given a power-law random graph (PLRG) $G(n, \alpha)$ on n nodes with parameter α $(2 < \alpha < 3)$, the cumulative damage that a single adversary can cause is bounded by $O(n^{1/\alpha} \times (\log n)^{(5-\alpha)/(3-\alpha)})$ with high probability. We prove two results on power-law random graphs to show this result.

LEMMA 12. *Every PLRG, $G(n, \alpha)$ for large values of n has a 3-connected subgraph H with $O(n/((\log n)^{\frac{\alpha-1}{3-\alpha}}))$ vertices with high probability.*

LEMMA 13. *Given a PLRG $G(n, \alpha)$ and a random vertex v with degree d, the number of vertices that get disconnected from the largest component in $G - \{v\}$ is bounded by $d(\log n)^{(5-\alpha)/(3-\alpha)}$ with high probability.*

We refer the reader to our technical report [27] for detailed proofs of these lemmas. To quantify the cumulative damage an adversary A can cause, let A have a maximum degree in G of $n^{1/\alpha}$. The number of vertices solely reliant on A is bounded by $n^{1/\alpha} \times (\log n)^{(5-\alpha)/(3-\alpha)}$. Given that there exists a sub-graph H which is $3-$connected, A cannot affect any node within this subgraph. Also, all these nodes can reliably broadcast all their messages within H. Additionally, every vertex v has one or more (indirect) neighbors within H. For every identity v that A propagates a spurious message, the penalty value of A increments by 1 and so does the penalty value of the indirect neighbors of v in H. If A targets specific identities such that the indirect neighbors of these identities are distributed among different vertices in H, then A obtains the maximum penalty value and hence is always ignored. To prevent this, A can at most target identities connected to a specific vertex u in H such that the penalties of u and A from the perspective of other nodes is the same. If A targets any additional identity which has a different indirect neighbor in H, then A's penalty overshoots u. Hence, to maximize the cumulative damage, A should target only those identities that solely rely on either A or u or both to connect to H. Using Lemma 12, we can bound this number by $4 \times n^{1/\alpha} \times (\log n)^{(5-\alpha)/(3-\alpha)}$. □

6. CONCLUSIONS AND IMPLICATIONS

In this paper, we study the problem of reliable broadcast in unknown fixed-identity networks. The results presented in this paper on this problem is of practical significance for several widely-used distributed systems including the Internet, Domain Name Service (DNS). From a theoretical standpoint, two immediate implications that follow are: *decentralized key-distribution* and *byzantine agreement*. If reliable broadcast is achievable, then every good node can broadcast its public-key to other good nodes in the system thereby achieving key-distribution in a decentralized manner. This

result has important ramifications for building decentralized security mechanisms for Internet routing as illustrated in [28] and DNS. Apart from key distribution, reliable broadcast is an essential building block for achieving byzantine agreement in unknown networks. The correspondence between the two is described earlier in Section 1.4.

The sparse network results have important ramifications for Internet routing. The Internet topology at the autonomous system is 1−connected and therefore cannot handle even a single adversarial node. The best-known previous result on reliable broadcast for sparse networks is that the problem is not solvable. Here, we show that one can limit the damage of an adversary by using penalty-based filtering. Specifically, for, Internet-like graphs which are modeled based on power-law random graphs, we show that a single adversary can cause very little damage. While the overall Internet topology is sparse, there are sub-graphs within this topology which exhibit high vertex connectivity. Within these sub-graphs, perfect reliable communication is achievable.

Two specific open problems that arise from this work are: (a) Can the connectivity requirements be relaxes for independent adversaries? (b) Can we solve this problem without assuming a known bound on the number of adversaries? Improvements in the time complexity and the communication overhead of our schemes are possible areas of future work.

Acknowledgments

The authors owe a lot of gratitude to Satish Rao, Richard Karp, Jayanthkumar Kannan and Subhash Khot for patiently listening to our arguments and providing insightful technical comments on this work. The authors also thank the anonymous reviewers, Matthew Caesar, Michael Freedman, Brighten Godfrey, Dilip Joseph, Karthik Lakshminarayanan and Sriram Sankaran who have provided invaluable feedback towards improving the presentation.

7. REFERENCES

[1] Internet Assigned Numbers Authority. http://www.iana.org.

[2] Aiello, W., Chung, F. R. K., and Lu, L. A random graph model for massive graphs. In *ACM STOC* (2000), pp. 171–180.

[3] Beimel, A., and Franklin, M. Efficient Reliable Communication over Partially Authenticated Networks. In *Theoretical Computer Science* (1999), vol. 220, pp. 185–210.

[4] Beimel, A., and Malka, L. Efficient Reliable Communication over Partially Authenticated Networks. In *PODC* (2003).

[5] Ben-Or, M. Another Advantage of Free Choice: Completely Asyncrhonous Agreement Protocols. In *PODC* (1983).

[6] Boneh, D., and Franklin, M. Identity-based Encryption from the Weil Pairing. In *CRYPTO* (2001).

[7] Cachin, C., Kursawe, K., and Shoup, V. Random Oracles in Constantinople: Practical Asynchronous Byzantine Agreement using Cryptography. In *PODC* (2000).

[8] Chang, H., R.Govindan, Jamin, S., Shenker, S., and Willinger, W. Towards capturing representative AS-level Internet topologies. In *Journal of Computer Networks* (2004).

[9] Chung, F., and Lu, L. The Average Distance in Random Graphs with given Expected Degrees. In *Internet Mathematics* (2003), vol. 1, pp. 91–114.

[10] Cocks, C. An Indentity-based Encryption Scheme based on Quadratic Residues. In *IMA* (2001).

[11] Dolev, D. The byzantine generals strike again. In *Journal of Algorithms* (1982), pp. 14–30.

[12] Dolev, D., Dwork, C., Waarts, O., and Yung, M. Perfectly secure message transmission. In *Journal of the ACM* (1993), pp. 17–47.

[13] Feldman, P., and Micali, S. Optimal algorithms for byzantine agreement. In *ACM Symposium on the Theory of Computing* (1988), pp. 148–161.

[14] Gamal, T. E. A public-key cryptosystem and a signature scheme based on discrete logarithms. In *IEEE Transactions in Information Theory* (1985), pp. 469–472.

[15] Giordano, S. Mobile ad hoc networks. 325–346.

[16] Kent, S., Lynn, C., and Seo, K. Secure Border Gateway Protocol (Secure-BGP). *IEEE Journal on Selected Areas of Communications 18*, 4 (April 2000), 582–592.

[17] Lamport, L., Shostak, R., and M.Pease. The byzantine generals problem. In *ACM Trans. Program. Lang. Systems* (1982), pp. 382–401.

[18] Levine, M. BGP noise tonight? NANOG mail archives, October 2001. `http://www.merit.edu/mail.archives/nanog/2001-10/msg00221.html`.

[19] Lynch, N. *Distributed Algorithms*. Morgan Kaufmann, San Francisco, 1996.

[20] Misel, A. S. Wow, AS7007! NANOG mail archives, April 1997. `http://www.merit.edu/mail.archives/nanog/1997-04/msg00340.html`.

[21] Mockapetris, P., and Dunlap, K. Development of the domain name system. In *SIGCOMM* (1988), pp. 123–133.

[22] Oram, A. Peer-to-peer: Harnessing the power of disruptive technologies, 2001.

[23] Pease, M., Shostak, R., and Lamport, L. Reaching agreement in the presence of faults. In *Journal of the ACM* (1980), pp. 228–234.

[24] Rabin, M. Randomized Byzantine Generals. In *FOCS'83* (1983), pp. 403–409.

[25] Secure Origin BGP (soBGP). ftp://ftp-eng.cisco.com/sobgp.

[26] Shamir, A. Identity-based Cryptosystems and Signature Schemes. In *CRYPTO* (1984).

[27] Subramanian, L., Katz, R. H., Roth, V., Shenker, S., and Stoica, I. Reliable Broadcast in Unknown Fixed-Identity Networks. In *UC Berkeley Technical Report No.CSD-04-1358*.

[28] Subramanian, L., Roth, V., Stoica, I., Shenker, S., and Katz, R. Listen and Whisper: Security Mechanisms in BGP. In *Proceedings of ACM/USENIX NSDI* (2004).