Conflict on a Communication Channel

Valerie King Jared Saia Maxwell Young

Department of Computer Science, University of Victoria, Canada, val@cs.uvic.ca
Department of Computer Science, University of New Mexico, USA, saia@cs.unm.edu
David Cheriton School of Computer Science, University of Waterloo, Canada, m22young@cs.uwaterloo.ca

PODC Regular Paper Submission
(with consideration as a Brief Announcement otherwise)

Abstract

Imagine that Alice wants to send a message to Bob, and that Carol wants to prevent this. Assume there is a communication channel between Alice and Bob, but that Carol is capable of blocking this channel. Furthermore, there is a cost of S dollars to send on the channel, L dollars to listen on the channel and J to block the channel. How much will Alice and Bob need to spend in order to guarantee transmission of the message?

This problem abstracts many types of conflict in information networks including: jamming attacks in wireless networks and distributed denial-of-service (DDoS) attacks on the Internet, where the costs to Alice, Bob and Carol represent an expenditure of energy and network resources. The problem allows us to quantitatively analyze the economics of information exchange in an adversarial setting and ask: Is communication cheaper than censorship?

We answer this question in the affirmative by showing that it is significantly more costly for Carol to block communication of the message than for Alice to communicate it to Bob. Specifically, if S, L and J are fixed constants, and Carol spends a total of B dollars to try to block the message, then Alice and Bob must spend only $O(B^{\varphi-1}+1) = O(B^{.62}+1)$ dollars in expectation to transmit the message, where $\varphi = (1+\sqrt{5})/2$ is the golden ratio. Surprisingly, this result holds even if (1) the value of B is *unknown* to both Alice and Bob; (2) Carol knows the algorithms of both Alice and Bob, but not their random bits; and (3) Carol is adaptive: able to launch attacks using total knowledge of past actions of both players.

Finally, we apply our work to two concrete problems: (1) denial-of-service attacks in wireless sensor networks and (2) application-level distributed denial-of-service attacks in a wired client-server scenarios. Our applications show how our results can provide an additional tool in mitigating such attacks.

Keywords: Adverarial fault tolerance, algorithms, jamming attacks, denial-of-service attacks, wireless sensor networks, Byzantine faults.

(Cover Page)

1 Introduction

In November of 2010, several web hosting and banking companies, including Amazon.com, Visa, Mastercard, and PayPal, severed ties with the website Wikileaks [42, 62]. In retaliation, the Anonymous group of Internet activists launched distributed denial-of-service (DDoS) attacks against these companies [2, 43]. Surprisingly, the web pages of both Wikileaks, and all the companies that were attacked by Anonymous emerged relatively unscathed despite the fact that Wikileaks suffered a significant attack on its financial and computational resources, and all parties suffered prolonged and sophisticated DDoS attacks. Some interesting questions arise in light of this incident: Is it fundamentally easier to communicate in large-scale networks than it is to block communication? How does the Internet compare with wireless networks where denial-of-service (DoS) attacks are easily launched via disruption of the communication medium [46, 51]? When altercations arise on modern networks, what are the most effective strategies for both sides?

To understand these questions from an algorithmic perspective, we define the following simple problem, which we call the *3-Player Scenario*: Alice wishes to guarantee transmission of a message m directly to Bob over a single communication channel. However, there exists an adversary Carol who aims to prevent communication by blocking transmissions over the channel. We consider two cases: (Case 1) when Carol may spoof or even control Bob, which allows her to manipulate an unwitting Alice into incurring excessive sending costs; and (Case 2) where Bob is both correct, unspoofable, and his communications cannot be blocked. Here, "cost" corresponds to a network resource, such as energy in wireless sensor networks (WSNs) or bandwidth in wired networks.

In the 3-Player Scenario, we show that communication is fundamentally cheaper than censorship. Specifically, we describe a protocol that guarantees correct transmission of m, and given that Carol incurs a cost of B, has the following properties. In Case 1, the expected cost to both Alice and Bob is $O(B^{\varphi-1} + 1)$ where φ is the golden ratio. In Case 2, the expected cost to both Alice and Bob is $O(B^{0.5} + 1)$. In both cases, Carol's cost asymptotically exceeds the expected cost of either correct player.

In the remainder of this section, we describe our model setup, state our main results and summarize related work. Section 2 includes our full proofs for the 3-Player Scenario. Section 3 addresses jamming adversaries in WSNs and applies our results to the problems of single-hop local broadcast and multi-hop reliable broadcast. Section 4 shows how our results can be employed to mitigate application-level DDoS attacks. We conclude with a discussion of open problems in Section 5

1.1 The 3-Player Scenario: Model Specification and Assumptions

We now describe the critical model parameters of the 3-Player Scenario. We defer an in-depth discussion of these parameters until Sections 3 and 4.

Las Vegas Property: Communication of m from Alice to (a correct) Bob must be guaranteed with probability 1; that is, we require a Las Vegas protocol for solving the 3-Player Scenario. An obvious motivation for this Las Vegas property is a critical application, such as an early warning detection system or the dessimination of a crucial security update, where minimizing the probability of failure is paramount. The Las Vegas property has additional merit in multi-hop WSNs where Monte Carlo algorithms may not be able to achieve a sufficiently low probability of error; due to space constraints, we expand on this in Section B.4.1.

Channel Utilization: Sending or listening on the communication channel by Alice and Bob is measured in discrete units called *slots*. For example, in WSNs, a slot may correspond to an actual time slot in a time division multiple access (TDMA) type access control protocol. The cost for sending or listening is S or L per slot, respectively. When Carol blocks a slot, she disrupts the channel such that no communication is possible; blocking costs J per slot. If a slot contains traffic or is blocked, this is detectable by a player who is *listening* at the *receiving end* of the channel, but not by the originator of the transmission. For example, a transmission (blocked or otherwise) from Bob to Alice is detectable only by Alice; likewise, a transmission (blocked or otherwise) from Alice to Bob is detectable only by Bob. A player cannot discern whether a blocked slot has disrupted a legitimate message; only the disruption is detectable. For example, high energy noise is detectable over the wireless channel in WSNs, but a receiving device cannot tell if this results from a message collision or a device deliberately disrupting the channel. We let B be the total amount Carol will spend over the course of the algorithm; this value is *unknown* to either Alice or Bob. Finally, we say that any player is *active* in a slot if that player is sending, listening or blocking in that slot.

Correct & Faulty Players: If Alice is faulty, there is clearly no hope of communicating m; therefore, Alice is assumed to be correct. Regarding the correctness of Bob, in Case 1, Carol may spoof or control Bob; in Case 2, communications from Bob are always trustworthy. We emphasize that, in Case 1, Alice is uncertain about whether to trust Bob since he may be faulty. This uncertainty corresponds to scenarios where a trusted dealer attempts to dessiminate content to its neighbors, some of whom may be faulty and attempt to consume resources by requesting numerous retransmissions. Case 2 corresponds to situations where communications sent by Bob are never disrupted and can be trusted; here, the blocking of m is the only obstacle.

Types of Adversary: Carol has full knowledge of past actions by Alice and Bob. This allows for *adaptive attacks* whereby Carol may alter her behavior based on observations she has collected over time. Furthermore, under conditions discussed in Section 2.2, Carol can also be *reactive*: in any slot, she may detect a transmission and then disrupt the communication (however, she cannot detect when a player is listening). This is pertinent to WSNs where the effectivess of a reactive adversary has been shown experimentally.

1.2 Solving the 3-Player Scenario: Fair & Favorable Protocols

We analyze the cost of our algorithms as a function of B. In this way, we obtain a notion of cost incurred by a player that is *relative to the cost incurred by Carol*. In devising our algorithms, we seek to achieve two properties with regards to relative cost.

First, our protocol should be *fair*; that is, Alice and Bob should incur the same *worst case asymptotic cost*. When network devices have similar resource constraints, such as in WSNs where devices are typically battery powered, this is critical. Alternatively, in networks where a collection of resource-scarce devices (i.e. client machines represented by Alice) occupy one side of the communication channel and a single well-provisioned device (i.e. a server represented by Bob) occupies the other side, the *aggregate* cost to Alice's side should be roughly equal to that of Bob.

Second, we desire *favorable* protocols; that is, for B sufficiently large, Alice and Bob both incur asymptotically less expected cost than Carol. DoS attacks are effective because a correct device is *always* forced to incur a higher cost relative to an attacker. However, if the correct players incur asymptotically less cost than Carol, then Alice and Bob enjoy the advantage, and Carol is faced with the problem of having her resources consumed disproportionately in her attempt to censor communication.

1.3 Our Main Contributions

Throughout, let $\varphi = (1 + \sqrt{5})/2$ denote the golden ratio. We assume that S, L, and J are fixed constants. Our main analytical contributions are listed below.

Theorem 1. *Assume Carol is an adaptive adversary and that she is active for B slots. There exists a fair and favorable algorithm for the 3-Player Scenario with the following properties:*

- *In Case 1, the expected cost to each correct player is $O(B^{\varphi-1} + 1) = O(B^{0.62} + 1)$. In Case 2, the expected cost to each correct player is $O(B^{0.5} + 1)$.*
- *If Bob is correct, then transmission of m is guaranteed and each correct player terminates within $O(B^{\varphi})$ slots in expectation.*

In networks with sufficient traffic, Theorem 1 still holds when Carol is also reactive (Section 2.2). We also prove that any protocol which achieves $o(B^{0.5})$ expected cost for Bob requires more than $2B$ slots to terminate (Section 2.3); this lower bound has bearing on the worst-case $\omega(B)$ slots required by our protocol.

Our next Theorems 2 & 3 are applications of Case 1 of Theorem 1 to WSNs. We consider a more general setting where Alice wishes to locally (single-hop) broadcast to n neighboring receivers of which any number are spoofed or controlled by Carol. Unfortunately, a naive solution of having each receiver execute a separate instance of our 3-Player Scenario protocol fails to be fair. Thus, we need a different algorithm to achieve the following result.

Theorem 2. *There exists a fair (up to small polylogarithmic factors in n) and favorable algorithm for achieving local broadcast with the following properties:*

- *If Carol's receivers are active for a total of B slots, then the expected cost to Alice is $O(B^{\varphi-1} \ln n + \ln^{\varphi} n)$ and the expected cost to any correct receiver is $O(B^{\varphi-1} + \ln n)$.*

- *Transmission of m is guaranteed and all correct players terminate within $O((B + \ln^{\varphi-1} n)^{\varphi+1})$ slots (not in expectation). For $B \geq \ln^{\varphi-1} n$, this is within an $O(B^{\varphi})$-factor of the optimal latency.*

Reliable broadcast in *multi-hop* WSNs deals with conveying m from one node to all other nodes in the network. We make the standard assumptions that any node p can be heard by the set of neighboring nodes in the topology, $N(p)$ and that, for any p, at most t nodes in $N(p)$ suffer a fault (t-bounded fault model) [12, 13, 36]. We analyze the grid model using the result of Bhandari & Vaidya [13], and general graphs using the Certified Propagation Protocol (CPA) protocol of Pelc & Peleg [49].

Theorem 3. *For each correct node p, assume the t nodes in $N(p)$ are Byzantine and can be used by Carol to disrupt p's communications for $\beta \leq B_0$ time slots. Then, using the local broadcast protocol of Theorem 2, fair and favorable reliable broadcast is possible under the following topologies:*

- *In the grid with the optimal fault tolerance $t < (r/2)(2r + 1)$.*
- *In any graph, assuming that (a) t is appropriately bounded such that CPA achieves reliable broadcast and (b) the topology and location of the dealer is known to all nodes.*

To the best of our knowledge, all previous reliable broadcast protocols require correct nodes to spend more energy in communication attempts than that spent by adversarial nodes. Our results are the first favorable protocols and, importantly, the first to account for the *significant cost of listening* to the wireless channel.

Finally, Theorem 4 is an application of Case 2 of Theorem 1 to a client-server scenario where Carol represents malicious clients engaging in a DDoS attack on a server.

Theorem 4. *Assume Carol commits her DDoS attack using a bandwidth R. Service is guaranteed if the expected aggregate bandwidth (upstream or downstream bits per second) of both the clients and the server is $G = O(R^{0.5})$, and the probability of a serviced request is $G/(G + R)$.*

Therefore, against a server defended by our protocol, Carol must incur additional monetary costs in order to procure the number of machines necessary for sustaining the level of attack she would otherwise achieve.

1.4 Related Work

Jamming Attacks in WSNs: Several works addressing applied security considerations demonstrate that devices in a WSN are vulnerable to adversarial jamming [5, 9, 40, 68] where the adversary deliberately disrupts the communication medium. Defenses include spread spectrum techniques, frequency or channel hopping, and mapping with rerouting (see [30, 47, 66, 67] and references therein).

There are a number of theoretical results on jamming adversaries; however, none explicitly account for listening costs and there is no notion of favorability. Gilbert *et al.* [24] examine the duration for which communication between two players can be disrupted in a model with collision detection in a time-slotted network against an adversary who interferes with an unknown number of transmissions. As we do, the authors assume channel traffic is always detectable at the receiving end (i.e. silent cannot be "forged"). Pelc and Peleg [50] examine an adversary that randomly corrupts messages; we do not require the adversary to behave randomly. Awerbuch *et al.* [7] give a jamming-resistant MAC protocol in a single-hop network with an adaptive, but non-reactive, adversary. Richa *et al* [54] significantly extend this work to multi-hop networks. Dolev *et al.* [18] address a variant of the gossiping problem when multiple channels are jammed. Gilbert *et al.* [23] derive bounds on the time required for information exchange when a reactive adversary jams multiple channels. Meier *et al.* [44] examine the delay introduced by a jamming adversary for the problem of node discovery, again in a multi-channel setting. Dolev *et al.* [19] address secure communication using multiple channels with a non-reactive adversary. Recently, Dolev *et al.* [17] consider wireless synchronization in the presence of a jamming adversary.

Reliable Broadcast: Reliable broadcast has been extensively studied in the grid model [10, 12–14, 35, 36, 60, 61]. Listening costs are accounted for by King *et al.* [35, 61] but jamming adversaries are not considered; however, the authors introduce the *Bad Santa* problem which we use to achieve a lower bound result in Section 2.3. With a reactive jamming adversary, Bhandhari *et al.* [15] give a reliable broadcast protocol when the amount of jamming is bounded and known *a priori*; however, correct nodes must expend considerably more energy than the adversary. Progress towards fewer broadcasts is made by Bertier *et al.* [11]; however, each node spends significant time in the costly listening state. Alistarh *et al.* [3] assume collision detection

3-PLAYER SCENARIO PROTOCOL for round $i \geq 2$

Send Phase: For each of the 2^{ci} slots do

- Alice sends m with probability $2/2^i$.
- Bob listens with probability $2/2^{(c-1)i}$.

If Bob received the message, then Bob terminates.

Ack Phase: For each of the 2^i slots do

- Bob sends a `req` message.
- Alice listens with probability $4/2^i$.

If Alice listened to a slot in the Ack Phase where no `req` message or blocking was detected, she terminates.

Figure 1: Pseudocode for 3-PLAYER SCENARIO PROTOCOL.

and achieve non-cryptographic authenticated reliable broadcast. They apply their result to the grid model with a reactive jamming adversary; however, in their algorithm nodes incur considerable listening costs.

Wired DDoS Attacks: DDoS attacks are common with recorded attacks on high-profile companies such as Yahoo, Amazon, CNN, eBay, and many others [22]. Proposals for dealing with DDoS attacks include over-provisioning [1], throttling techniques [25, 45], currency schemes (see [6, 33, 63] and references therein). In currency schemes, the server provides service only to a client who pays in some form of currency. In [63], bandwidth is used as currency and, if the clients' aggregate bandwidth exceeds that of the attackers, then the clients capture server resources. Our work is complementary in that it delineates bounds on the expected bandwidth required in order to guarantee that the correct clients avoid zero throughput.

2 Our 3-Player Scenario Protocol

Figure 1 gives the pseudocode for our protocol called 3-PLAYER SCENARIO PROTOCOL (3PSP). Each round $i \geq 2$ consists of 2 phases and c is a constant to be determined later. We summarize a round i:

• *Send Phase:* This phase consists of 2^{ci} slots. In each slot: Alice sends m with probability $\frac{2}{2^i}$ for an expected total of $2^{(c-1)i+1}$ slots and Bob listens with probability $\frac{2}{2^{(c-1)i}}$ for an expected total of 2^{i+1} slots.

• *Ack Phase:* This phase consists of 2^i slots. If Bob has not received m, then Bob sends a request for retransmission, `req`, for all 2^i slots. Alice listens in each slot with probability $4/2^i$ (note that $i \geq 2$ is required) for an expected total 4 slots.

Termination Conditions: Termination conditions are important because Carol cannot be allowed to keep the players active in perpetuity while simultaneously forcing them to incur a *higher* cost. Bob terminates the protocol upon receiving m. Since Alice is not spoofed, as discussed in Section 1.1, this termination condition suffices. Alice terminates if she listens to a slot in the Ack Phase which is not blocked and does not contain `req` message; since blocked slots are detectable by Alice (who is on the receiving end of a `req` message) while listening (Section 1.1), this condition suffices. In other words, Alice continues into the next round if and only if (1) Alice listens to zero slots or (2) all slots listened to by Alice in the Ack Phase contain a blocked slot or `req`. We highlight the two situations where this condition is met:

• *Send Failure:* Bob is correct and has not received m.

• *Ack Failure:* Bob is faulty and sends `req`s, *or* Bob is correct and terminated and Carol either spoofs `req`s or blocks slots in order to trick Alice into thinking a valid `req` was indeed sent and/or blocked.

Ack Failures and Cases 1 & 2: Note that an "acknowledgement" occurs via silence in at least one slot in the Ack Phase. We say an *Ack Failure* occurs when Carol blocks for all slots in the Ack Phase.

In Case 1, an Ack Failure corresponds to a critical attack that can be employed in Ack Phase after the delivery of m. Carol can avoid the listening costs in the Send Phase, and then drain Alice's energy by making it appear as if Bob repeatedly did not receive m and is requesting a retransmission in the Ack Phase. This attack affects Alice only. Note that if Bob is actually correct, the attack is only effective once m is received since, if a correct Bob has not received m, a `req` will be issued anyway and the attack accomplishes nothing.

In Case 2, no blocking occurs in the Ack Phase and, therefore, no Ack Failure can occur. In fact, in Case 2, the Ack Phase can be shortened to a single slot where Bob sends his `req` and Alice listens; however, this does not change our cost analysis and our current presentation is more general.

2.1 Analysis of the 3-Party Scenario Protocol

For a given round, we say it is a *send-blocking* round if Carol blocks at least half of the slots in the Send Phase; otherwise, it is a *non-send-blocking* round. Similarly, a *ack-blocking* round is a round where Carol blocks or spoofs `req` messages from Bob in at least half the slots in the Ack Phase; otherwise, it is *non-ack-blocking*. Throughout, assume ceilings on the number of active slots of a player if it is not an integer.

Bounds on c: Clearly, $c > 1$ or Bob's listening probability in the Send Phase is nonsensical. For Case 1, note that if $c \geq 2$, then the expected cost to Alice is at least as much as the expected cost to a potentially faulty/spoofed Bob. If Bob happens to be faulty/spoofed, then the cost to him for an Ack Failure is less than the expected cost to Alice since a faulty/spoofed Bob will simply not listen in the Send Phase; as discussed above, we must avoid this since it admits a draining attack against Alice. Therefore, we have $1 < c < 2$. For Case 2, since Bob is guaranteed to be correct, the acceptable range is $1 < c \leq 2$.

Lemma 1. *Consider a non-send-blocking round of* 3-PLAYER SCENARIO PROTOCOL. *The probability that Bob does not receive the message from Alice is less than e^{-2}.*

Proof. Let $s = 2^{ci}$ be the number of slots in the Send Phase. Let p_A be the probability that Alice sends in a particular slot. Let p_B be the probability that Bob listens in a particular slot. Let $X_j = 1$ if the message is not delivered from Alice to Bob in the j^{th} slot. Then $Pr[$ m is not delivered in the Send Phase$]$=$Pr[X_1X_2 \cdots X_s = 1]$=$Pr[X_s = 1 \mid X_1X_2 \cdots X_{s-1} = 1] \cdot \prod_{i=1}^{s-1} Pr[X_i = 1]$. Let $q_j = 1$ if Carol does not block in slot j; otherwise, let $q_j = 0$. The value of q_j can be selected arbitrarily by Carol. Then $Pr[X_i = 1 \mid X_1X_2 \cdots X_{i-1} = 1] = 1 - p_Ap_Bq_j$ and substituting for each conditional probability, we have $Pr[X_1X_2 \cdots X_s = 1] = (1 - p_Ap_Bq_1) \cdots (1 - p_Ap_Bq_s) = \prod_{j=1}^{s}(1 - p_Ap_Bq_j) \leq e^{-p_Ap_B \sum_{j=1}^{s} q_j} < e^{-2}$ since $p_Ap_B \sum_{j=1}^{s} q_j > (2/2^i)(2/2^{(c-1)i})(s/2) = (2/2^i)(2/2^{(c-1)i})(2^{ci}/2) = 2$ since the round is not send-blocking and so Carol blocks less than $s/2$ slots. □

Note that Lemma 1 handles adaptive (but not reactive) adversaries. A simple but critical feature of tolerating adaptive adversaries is: the probability that a player is active in one slot is independent from the probability that the player is active in another slot. Therefore, knowing that a player was active for k slots in the past conveys no information about future activity. Believing otherwise is the trap of the well-known"Gambler's Fallacy" [59]. For reactive adversaries, we need only modify Lemma 1 as we do later.

Lemma 2. *Assume that Bob is correct and there are no send-blocking rounds and no ack-blocking rounds. Then, the expected cost of each player is $O(S + L) = O(1)$.*

Proof. Using Lemma 1, the expected cost to Alice is at most $\sum_{i=2}^{\infty} e^{-2(i-2)} \cdot (2 \cdot 2^{(c-1)i} \cdot S + 4 \cdot L)$ $\leq \sum_{i=2}^{\infty}(e^{5-i} \cdot S + e^{2-2i} \cdot 4 \cdot L) = (e^5 \cdot S \cdot \sum_{i=2}^{\infty} e^{-i}) + (e^2 \cdot 4 \cdot L \cdot \sum_{i=2}^{\infty} e^{-2i}) = O(S + L) = O(1)$. Similarly, the expected cost to Bob is at most $\sum_{i=2}^{\infty} e^{-2(i-2)} \cdot (2^{i+1} \cdot L + 2^i \cdot S) \leq \sum_{i=2}^{\infty}(e^{5-i} \cdot L + e^{4-i} \cdot S)$ $= O(S + L) = O(1)$ since S and L are constants. □

Now consider when attacks may occur in the Ack Phase:

Lemma 3. *Assume that Bob has received m by round i and that round i is non-ack-blocking. Then the probability that Alice retransmits m in round $i + 1$ is less than e^{-2}.*

Proof. Let $s = 2^i$ be the number of slots in the Ack Phase and let $p = 4/2^i$ be the probability that Alice listens in a slot. For slot j, define X_j such that $X_j = 1$ if Alice does not terminate. Then $Pr[$ Alice retransmits m in round $i + 1] = Pr[X_1X_2 \cdots X_s = 1]$. Let $q_j = 1$ if Carol does not block in slot j; otherwise, let $q_j = 0$. The q_j values are determined arbitrarily by Carol. Since Alice terminates if and only if she listens and does not detect any activity, then $Pr[X_j = 1] = (1 - pq_j)$. Therefore, $Pr[X_1X_2 \cdots X_s = 1] \leq e^{-p \sum_{j=1}^{s} q_j} < e^{-2}$. □

Lemma 4. *Assume there is at least one send-blocking round. Then, the expected cost to Alice is $O(B^{(c-1)/c} + B^{(c-1)})$ and the expected cost to a correct Bob is $O(B^{\frac{1}{c}})$.*

Proof. We consider Case 1 and Case 2 with regards to Bob, discussed in Section 1.1. Let $i \geq 2$ be the last round which is send-blocking. Let $j \geq i$ be the last round which is ack-blocking; if no such ack-blocking round exists, then assume $j = 0$. In Case 1, the total cost to Carol is $B = \Omega(2^{ci} \cdot J + 2^j \cdot J) = \Omega(2^{ci} + 2^j)$ since J is a constant. In Case 2, only send-blocking occurs and so $B = \Omega(2^{ci} \cdot J)$.

Alice: We first calculate the expected cost to Alice prior to successfully transmitting m. In round i, Carol blocks the channel for at least $2^{ci}/2$ slots. Using Lemma 1, the expected cost to Alice prior to m being delivered is $O(2^{(c-1)i} \cdot S + 4 \cdot L) + \sum_{k=1}^{\infty} e^{-2(k-1)} \cdot (2 \cdot 2^{(c-1)(i+k)} \cdot S + 4 \cdot L) = O(2^{(c-1)i} \cdot S + L) = O(2^{(c-1)i})$ by the bounds on c and given that S and L are constants; note, this is the total cost to Alice for Case 2.

Now, using Lemma 3, we calculate the expected cost to Alice after delivery; this addresses ack-blocking rounds possible only in Case 1. By assumption, the last ack-blocking round occurs in round j and therefore Alice's expected cost is $O(2^{(c-1)j} \cdot S + 4 \cdot L) + \sum_{k=1}^{\infty} e^{-2(k-1)} \cdot (2 \cdot 2^{(c-1)(j+k)} \cdot S + 4 \cdot L) = O(2^{(c-1)j} \cdot S + L)$ by the bounds on c. Therefore, the total expected cost to Alice is $O(2^{(c-1)i} \cdot S + 2^{(c-1)j} \cdot S + L) = O(2^{(c-1)i} + 2^{(c-1)j})$. Since $B = \Omega(2^{ci} + 2^j)$, this cost as a function of B is $O(B^{(c-1)/c} + B^{(c-1)})$.

Bob: Finally, assume Bob is correct. Using Lemma 1, Bob's expected cost prior to receiving m is $O(2^{i+1} \cdot L + 2^i \cdot S) + \sum_{k=1}^{\infty} e^{-2(k-1)} \cdot (2 \cdot 2^{i+k} \cdot L + 2^{i+k} \cdot S) = O(2^i \cdot L + 2^i \cdot S) = O(2^i)$ since S and L are constants. Thus, the expected cost for Bob as a function of B is $O(B^{1/c})$. □

We now give the proof for Theorem 1 stated in Section 1.3:

Proof of Theorem 1: In Case 1, Lemma 4 tells us that the expected cost to Alice and Bob in terms of B is $O(B^{(c-1)/c} + B^{(c-1)})$ and $O(B^{1/c})$, respectively. Therefore, the exponents of interest which control the cost to each player are $(c-1)/c$, $c-1$, and $1/c$. The value of c that should be chosen must minimize $\max\{(c-1)/c, c-1, 1/c\}$ since we are interested in fair protocols. Given that $1 < c < 2$, we have $1/c > (c-1)/c$. Therefore, we solve for c in $c - 1 = 1/c$, this gives $c = (1+\sqrt{5})/2$ which is the golden ratio. By Lemma 2 and the above argument, the expected cost to each player is $O(B^{\varphi-1} + 1)$. In Case 2, Lemma 4 tells us that Alice's expected cost in terms of B is $O(B^{(c-1)/c})$ the exponents of interest are simply $(c-1)/c$ and $1/c$; minimizing them yields $c = 2$. Therefore, the cost to each player is $O(B^{1/2} + 1)$.

Finally, define latency to be the number of slots prior that occur to termination by both correct players. Consider how many non-send-blocking or non-ack-blocking rounds *either* player may endure before terminating successfully; let X denote the random variable for this number of rounds. Then, $E[X] \leq 1 \cdot (1-e^{-2}) + 2 \cdot e^{-2}(1-e^{-2}) + 3 \cdot e^{-4}(1-e^{-2}) + ... = \sum_{i=1}^{\infty} i e^{2(1-i)}(1-e^{-2}) = (1-e^{-2})e^2 \sum_{i=1}^{\infty} i(e^{-2})^i$ by Lemmas 1 or 3. Therefore, $E[X] \leq 1/(1-e^{-2}) = O(1)$ which translates into $O(1)$ time slots consumed by non-send or non-ack-blocking rounds. Now consider the send- or ack-blocking rounds; note that Carol is limited to at most $\lg(2B) + O(1)$ such rounds which translates to $O(B^{\varphi})$ time slots. Therefore, regardless of how Carol blocks, the expected number of time slots prior to successful termination is $O(B^{\varphi})$. □

2.2 Tolerating a Reactive Adversary

Consider a reactive adversary Carol who can detect channel activity without cost, and then block; this ability is possible in WSNs (see Section 3.1). In our 3-Player Scenario, Carol can now detect that m is being sent in the Send Phase and block it without fail. To address this powerful adversary, we consider the case where critical data, m, and more often, non-critical data m', is sent over the channel by other participants in addition to Alice and Bob. Carol can detect the traffic; however, she cannot discern whether it is m or m' without listening to a portion of the communication (such as packet header information).

In a slot where channel activity is detected, even if Carol listens for a portion of the message, she incurs a substantial cost. Therefore, the cost to Carol is proportional to the number of messages to which she listens. Importantly, in the presence of m', Carol's ability to detect traffic for free is unhelpful since m' provides "camouflage" for m. Certainly Carol may block *all active slots* to prevent transmission of m; however, this is no different than blocking *all slots* in our original 3-Player Scenario (see Section A for more discussion).

This setting corresponds to situations where communication occurs steadily between many participants or via several distributed applications, and Carol wishes to target only a critical few. If m and m' are sent over the channel in the same slot, the two messages collide and Bob receives neither. Define a slot as *active* if either m or m' is sent in that slot. For this result only, redefine a send-blocking round as one where Carol

listens or blocks for at least a $1/3$-fraction of the *active* slots; otherwise, it is a *non-send-blocking round*. We provide a result analagous to Lemma 1.

Lemma 5. *Let Carol be an adaptive and reactive adversary. Then, in a non-send-blocking round of the* 3-PLAYER SCENARIO PROTOCOL*, the probability that Bob does not receive m from Alice is at most e^{-2}.*

Proof. Let $x = 2^{ci}$ be the number of slots in the Send Phase. Consider the set of slots used by all participants other than Alice. We assume these participants pick their slots at random to send, so that for any slot the probability is 2/3 that the slot is chosen by at least one of them. Since we assume these messages m' are sent independently at random, then Chernoff bounds imply that w.h.p., i.e., $1 - 1/x^{c'}$ for a constants c', ϵ and sufficiently large x, the number of slots y during which m' is sent is greater than $(2x/3)(1-\epsilon)$ where x is the total number of slots in a phase. In the same way, assume the number of slots in which Alice sends is at least $a = (1-\delta)xp_A = (1-\delta)2^{(c-1)i+1}$ with probability $1 - 1/x^{c''}$ for a constant δ, c'' and sufficiently large x. The number of active slots sent by Alice or other participants is clearly at least y.

By definition of a non-send-blocking round, Carol listens to or blocks less than $x/3$ (active) slots. As Carol has no information about the source of a message sent in an active slot until she listens to it, her choice is independent of the source of the message. Given a slot that Alice sends on, there is at least a $1 - (x/3)/y$ chance it will not be listened to or blocked by Carol. The probability that this slot will not be used by another participant is $1/3$ and the probability that Bob will listen to the slot is p_B. Hence the probability of a successful transmission from Alice to Bob on a slot which Alice sends on is at least $(1 - x/(3y))(1/3)p_B = (1 - 1/(2(1-\epsilon)))(1/3)p_B \geq (1/6)p_B$ when $y > (1-\epsilon)(2x/3)$. The probability that all messages that Alice sends fail to be delivered is at most $(1 - p_B/6)^a - 2/x^{c''}$ where the last term is the probability that y or a is small and $c'' > 0$ is a constant. Redefine $p_B = 6/((1-\delta)2^{(c-1)i})$; note that this constant factor increase in the listening probability does not change our asymptotic results and our analysis in Section 2.1 proceeds almost identically. Therefore, we then have $(1 - p_B/6)^a - 2/x^{c''} \leq e^{-2}$. □

The 3-PLAYER SCENARIO PROTOCOL can be modified so that the initial value of i is large enough to render the error arising from the use of Chernoff bounds sufficiently small; we omit these details. Also, the required level of channel traffic detected by Carol is flexible and different values can be accomodated if the players' probabilities for sending and listening are modified appropriately in the 3-PLAYER SCENARIO PROTOCOL; our results hold asymptotically. Finally, we emphasize that Lemma 3 does not require modification. Carol cannot decide to block only when Alice is listening since detecting when a node is listening is impossible. Alternately, Carol cannot silence a `req` through (reactive) blocking since this is still interpreted as a retransmission request. Using Lemma 5, Theorem 1 follows as before.

2.3 On Latency & Lower Bounds

King *et al.* [35] introduced the *Bad Santa* problem which is described as follows. A child is presented with K boxes, one after another. When presented with each box, the child must immediately decide whether or not to open it. If the child does not to open a box, it can never be revisited. Half the boxes have presents in them, but the decision as to which boxes have presents is made by an adversarial Santa who wants the child to open as many empty boxes as possible. The goal is for the child to obtain a present *with probability 1*, while opening the smallest expected number of boxes. In [35, 61], the authors prove a lower bound of $\Omega(K^{0.5})$ on the expected number of opened boxes.

Theorem 5. *Any algorithm that solves the 3-Player Scenario with $o(B^{0.5})$ cost to Bob must have a latency exceeding $2B$.*

Proof. A lower bound for the 3-Player Scenario is complicated by the possibility that the strategies of Alice and Bob may adapt over time; for example, they may change depending on how Carol blocks. To address this, we assume a more powerful Bob. Specifically, assume that communication of m occurs if Bob is able to find an unblocked time slot in which to listen *or* to send. Furthermore, assume Bob can tell when he has found such a slot once he listens or sends in that slot. Therefore, such a Bob is at least as powerful as the Bob in the 3-Player Scenario.

Now, if Carol has a budget of size B, we ask: Does Bob have a strategy with $o(B^{0.5})$ expected active slots such that, with probability 1, he finds at least one unblocked slot within $2B$ slots? Assume that such a strategy exists and consider the Bad Santa problem on $2B$ boxes. Using Bob's strategy, the child is

guaranteed to obtain a present with probability 1 while opening $o(B^{0.5})$ boxes in expectation. However, this contradicts the $\Omega(B^{0.5})$ lower bound result in [35] and the result follows. □

This result illustrates a relationship between the Bad Santa problem and the 3-Player Scenario, and it provides some insight into why our protocol has a worst case latency of $\omega(B)$ slots.

3 Application 1: Jamming Resistance in Wireless Sensor Networks

The shared wireless medium of sensor networks renders them vulnerable to jamming attacks [64]. A jamming attack occurs when an attacker transmits noise at high energy, possibly concurrently with a (legitimate) transmission, such that communication is disrupted within the area of interference. Consequently, this behavior threatens the availability of sensor networks [66].

3.1 Rationale for the 3-Player Scenario Involving WSN Devices

Wireless network cards offer states such as *sleep, receive (or listen)* and *transmit (or send)*. While the sleep state requires negligible power, the cost of the send and listen states are roughly equivalent and dominate the operating cost of a device. For example, the send and listen costs for the popular Telos motes are 38mW and 35mW, respectively (note $S \approx L$) and the sleep state cost is 15μW [52]; therefore, the cost of the send/listen state is more than a factor of 2000 greater and the sleep state cost is negligible. Disruption may not require jamming an entire slot so we set $J < S$ and assume a small m such that J and S are within a constant factor of each other; larger messages can be sent piecewise. In our protocols, we account for both send and receive costs. Throughout, when a node is not active, we assume it is in the energy-efficient sleep state.

Slots: There is a single channel and a time division multiple access (TDMA)-like medium access control (MAC) protocol; that is, a time-slotted network. For example, the well-known LEACH [28] protocol is TDMA-based. For simplicity, a *global* broadcast schedule assumed; however, this is likely avoidable if nodes maintain multiple schedules as with S-MAC [69]. Even then, global scheduling has been demonstrated by experimental work in [39] and secure synchronization has been shown [21].

A blocked slot occurs when Carol jams. Clear channel assessment (CCA), which subsumes carrier sensing, is a common feature on devices for detecting such events [53] and practical under the IEEE 802.11 standard [16]. *Collisions are only detectable by the receiver* [66]. When a collision occurs, a correct node discards any received data. The absence of channel activity cannot be forged; this aligns with the empirical work by Niculescu [48] who shows that channel interference increases linearly with the combined rate of the sources. Finally, we also note that several theoretical models feature collision detection (see [3,7,15,24,54]).

On Reactive Adversaries: CCA is performed via the radio chip using the *received signal strength indicator* (RSSI) [31]. If the RSSI value is below a clear channel threshold, then the channel is assumed to be clear [8]. Such detection consumes on the order of 10^{-6} W which is three orders of magnitude smaller than the send/listen costs; therefore, Carol can detect activity (but not message content) at essentially zero-cost. Listening to even a small portion of a message costs on the order of milliwatts and our argument from Section 2.2 now applies.

Cryptographic Authentication: We assume that messages can be authenticated. Therefore, Carol cannot spoof Alice; however, Bob's `req` can essentially be spoofed by an Ack-Failure (as discussed in Section 2) which, along with jamming, makes the problem non-trivial. Several results show how light-weight cryptographic authentication can be implemented in sensor networks [29, 37, 41, 64, 65]; therefore, it is important to consider its impact as we do here. However, the adversary may capture a limited number of players (such as Bob in the 3-Player Scenario); these players are said to suffer a Byzantine fault and are controlled by the adversary [64, 66]. Given this attack, we emphasize that, while we assume a shared key to achieve authentication, *attempts to share a secret send/listen schedule between Alice and Bob allows Carol to manipulate players in ways that are problematic*; due to lack of space, this is discussed further in Section B.3.

3.2 Local Broadcast & Guaranteed Latency

Our protocol LOCAL BROADCAST handles the general single-hop broadcast situation where Alice sends m to a set of n neighboring receivers within her transmission range. At first glance, this seems achievable by having each receiver execute an instance of 3PSP with Alice. However, the expected active time for Alice

LOCAL BROADCAST(m, Alice, R_{Alice}) for round $i \geq \lg(4 \ln n)$

Probabilistic Send Phase: For each of the $2^{\varphi i}$ slots do

- Alice sends m with probability $\frac{3 \ln n}{2^i}$.
- Each receiver that has not terminated listens with probability $\frac{2}{2^{(\varphi-1)i}}$.

Deterministic Send Phase: For each of the $2^{(\varphi-1)i+1}$ slots do

- Alice sends m.
- Each receiver that has not terminated listens.

Any receiver that receives m terminates the protocol.

Probabilistic Ack Phase: For each of the 2^i slots do

- Each receiver that has not terminated sends a `req` message.
- Alice listens with probability $\frac{4 \ln n}{2^i}$.

Deterministic Ack Phase: For each of the $2^{(\varphi-1)i+1}$ slots do

- Each reciever that has not received m sends a `req` message.
- Alice listens.

If Alice listened in either a Probabilistic Ack Phase or a Deterministic Ack Phase and detected no `req` message or collision then she terminates the algorithm.

Figure 2: Pseudocode for LOCAL BROADCAST.

is an $\Omega(n)$-factor larger than any correct receiver; thus, this is unfair. Furthermore, this protocol has poor latency. Here, we give a fast protocol that is both fair and favorable up to small polylogarithmic factors.

Our pseudocode is given in Figure 2. The probabilities for sending and listening are modified and there are two more phases (the Deterministic Send and Deterministic Ack Phases) where players act deterministically. Note that `req` messages can collide in the Probabilistic Ack Phase and will certainly collide in the Deterministic Ack Phase. This is correct as such a collision is due to either jamming or multiple receivers (correct or faulty) requesting a retransmission; this is fine and Alice will resend. LOCAL BROADCAST takes in as arguments the message m, the sender (Alice) and the set of receivers R_{Alice}. If the adversary jams, then none of the correct receivers receive m in that slot.

An important property of LOCAL BROADCAST is that there is a guaranteed bound on the latency. This is useful for achieving reliable broadcast in multi-hop networks in the next section.

Lemma 6. *Alice and all correct receivers terminate* LOCAL BROADCAST *in* $25 \cdot (B + \ln^{\varphi-1} n)^{\varphi}$ *time slots.*

Proof. The deterministic phases play a key role in establishing the bound on latency. If the adversary is not active for all slots in the deterministic Send Phase, then all correct receivers obtain m. Once all correct receivers terminate, the adversary must be active in all slots of the deterministic Ack Phase in order to prevent Alice from terminating. Therefore, prior to successful termination of all correct players (including Alice), the adversary is active for at least $2^{(\varphi-1)i+1}$ slots per round i in Epochs 2 & 4. For $d = \lg(4 \ln n)$, we seek the number of rounds ρ such that $\sum_{i=d}^{\rho} 2^{(\varphi-1)i+1} \geq B$ which yields that $\rho \geq \varphi \lg(B + 2^{\varphi-1} \ln^{\varphi-1} n)$ rounds suffices to exhaust the adversary (we are not being exact). Each round i has at most $4 \cdot 2^{\varphi \cdot i+1}$ slots so ρ rounds equal at most $25 \cdot (B + \ln^{\varphi-1} n)^{\varphi+1}$ slots. □

Due to space constraints, the full proofs for the following result are included in Section B.1.

Lemma 7. *Assume that Carol's receivers are active for a total of B slots. Then,* LOCAL BROADCAST *has the following properties:*

- *The expected cost to Alice is $O(B^{\varphi-1} \ln n + \ln^{\varphi} n)$. Therefore, for $B = \omega(\ln^{\varphi+1} n)$, Alice spends asymptotically less than Carol.*
- *The expected cost to any correct receiver is $O(B^{\varphi-1} + \ln n)$. Therefore, for $B = \omega(\ln n)$, Bob spends asymptotically less than Carol.*

The value n is the number of devices within the broadcast range of Alice. For a determined adversary, we expect $B > n$; that is, for an adversary intent on preventing communication, the number of time slots jammed will likely exceed the number of neighbors. Therefore, $B \gg \ln^{\varphi+1} n$. In this case (actually for $B \geq \ln^{\varphi-1} n$), the latency is $O(B^{\varphi+1})$ and, noting that Carol can prevent transmission for at least B slots, this is within an $O(B^{\varphi})$-factor of the optimal latency. By this and Lemmas 6 & 7, Theorem 2 now follows.

3.3 Jamming-Resistant Reliable Broadcast: Mitigating the Listening Cost Disadvantage

Reliable broadcast has been extensively studied in the multi-hop grid model [12–14, 36, 61], particularly with a jamming adversary [3, 11, 15]. Reliable broadcast is possible when t Byzantine nodes can each jam at most n_c transmissions [15]. Unfortunately, the protocol of [15], and the improvement by [11], requires that *correct nodes possess much more energy than the Byzantine nodes*. In particular, while the sending costs are improved in [11], both [11, 15] allow the adversary to force a correct node to *listen* for $\Omega(t \cdot n_c)$ slots (listening costs in [3] are similar). In contrast, each Byzantine node is active for n_c. This $\Omega(t)$-factor advantage affords the adversary a DDoS attack since these previous protocols are consistently unfavorable.

Setup: Here, each node $p(x, y)$ is situated at (x, y) in a grid. The dealer d is located at $(0, 0)$ and seeks to propagate m to all correct nodes in the network. When a node p sends a message, all listening nodes in $N(p)$ receive the message (analogous results will hold for the Euclidean metric [13]). There are $t < (r/2)(2r+1)$ Byzantine nodes in any neighborhood. For any correct node p, the adversary can use its t Byzantine nodes in $N(p)$ to jam for up to $B_0 = t \cdot n_c$ slots total. There is a global schedule (obeyed by the correct nodes) that assigns each node a slot for broadcasting; a specification is unimportant here (see [36] for an example).

Unlike the single-hop case, here the amount of jamming in a neighborhood is upper bounded by B_0 and known. This is required in [11, 15] and a similar assumption is made in [7, 54]. B_0 represents the number of times a Byzantine node can deviate from the global schedule within some time frame in a neighborhood before being identified and subjected to defensive techniques (see [66]). Not exceeding B_0 in each time frame allows the adversary to attack throughout the lifetime of the network and we pessimistically assume that B_0 is large so that the adversary may inflict sustained attacks (see Section B.2 for more discussion).

We incorporate LOCAL BROADCAST into the protocol of Bhandari & Vaidya [13] to achieve the first favorable reliable broadcast protocol. The hard latency bound of LOCAL BROADCAST is crucial for establishing when nodes send and listen in order to propagate m. Due to space constraints, our description of the protocol and proofs are given in Section B.4.2. We can show the following:

Lemma 8. *Assume for each node p, $t < (r/2)(2r+1)$ nodes in $N(p)$ are Byzantine and used by Carol to disrupt p's communications for $\beta \leq B_0$ time slots. Let $C = \{$Nodes q at (x, y) s.t. $(-r \leq x \leq r) \wedge (y \geq 0)\}$ be a corridor in the grid. There is a protocol for reliable broadcast in C with the following properties:*

- *If $\beta = O(r^2 \ln^{\varphi+1} r)$, then the expected cost to each each correct node is $O(r^2 \ln^{\varphi+2} r)$.*
- *If $\beta = \omega(r^2 \ln^{\varphi+1} r)$, then the protocol is fair and the expected cost to each correct node is $O(r^{2(2-\varphi)} \beta^{\varphi-1} \ln r + r^2 \ln^{\varphi} r) = o(\beta)$; that is, the expected cost to each correct node is asymptotically less than that incurred by Carol.*

Note that, for ease of exposition, our result applies to a single corridor of the grid; however, this is sufficient to prove reliable broadcast in the *entire network* since the grid can be covered piecewise by such corridors.

3.3.1 Reliable Broadcast for General Topologies

We examine the grid model above because it features in previous literature on jamming-resistant reliable broadcast [3, 11, 15]. However, Pelc & Peleg [49] examine the t-locally bounded fault model for arbitrary graphs.[1] Specifically, they each examine the broadcast protocol of Koo [36], which they call the *Certified Propagation Algorithm* (CPA). For *any* graph G, the authors prove that if t is bounded relative to some parameter corresponding to the topology of G, then CPA achieves reliable broadcast. CPA does not always achieve optimal fault tolerance; for example, it cannot tolerate the optimal number of faults $t = (r/2)(2r + 1) - 1$ in the grid as we do above. However, we address CPA because its generality is powerful.

The details of our protocol are presented in Section B.6. Each node requires knowledge of the full network topology and the location of the dealer. Given that nodes can discover such information – perhaps it is preprogrammed before deployment, or learned robustly after deployment – they then execute LOCAL BROADCAST with their neighbors in a timed fashion that is topology-dependent. The following analysis proves favorability, both for the grid and for general topologies where CPA achieves reliable broadcast:

Theorem 3 – Cost Analysis: In both of our protocols, each correct node p partakes in an execution of LOCAL BROADCAST $O(t)$ times as a sender and receiver; let k denote the total number of such executions.

[1] Ichimura & Shigeno [32] also examine general graphs and their approach can likely be incorporated also; however, in this extended abstract, we focus on the result of Pelc & Peleg [49].

For the i^{th} such execution, let τ_i be the number of slots for which the adversary is active for $i = 1, ..., k$. Denote the adversary's total active time by $\beta = \sum_{i=1}^{k} \tau_i \leq B_0$. Consider two cases:

Case I: Assume the adversary is active for a total of $\beta = \sum_{i=1}^{k} \tau_i = O(t \ln^{\varphi+1} t)$ slots over all k executions of LOCAL BROADCAST involving p. For each execution, p incurs $O(\tau_i^{\varphi-1} \ln t + \ln^{\varphi} t)$ cost in expectation by Theorem 2. Therefore, over $k = O(t)$ executions, p's expected total cost is $O((\sum_i^k \tau_i^{\varphi-1}) \ln t + t \ln^{\varphi} t) = O((\sum_i^k \tau_i) \ln t + t \ln^{\varphi} t) = O(\beta \ln t + t \ln^{\varphi} t)) = O(t \ln^{\varphi+2} t)$.

Case II: Otherwise, $\beta = \sum_{i=1}^{k} \tau_i = \omega(t \ln^{\varphi+1} t)$. By a corollary of Jensen's inequality for concave functions, for a concave function f, $f(\frac{1}{k} \sum_{i=1}^{k} \tau_i) \geq \frac{1}{k} \sum_{i=1}^{k} f(\tau_i)$. Since $f(\tau) = \tau^{\varphi-1}$ is concave, it follows that $\sum_{i=1}^{k} \tau_i^{\varphi-1} \leq k(\frac{1}{k} \sum_{i=1}^{k} \tau_i)^{\varphi-1} = k^{2-\varphi}(\sum_{i=1}^{k} \tau_i)^{\varphi-1}$. Therefore, the total expected cost to p over $k = O(t)$ executions is $O((\sum_{i=1}^{k} \tau_i^{\varphi-1}) \ln t) + O(t \ln^{\varphi} t) = O(t^{2-\varphi}(\sum_{i=1}^{k} \tau_i)^{\varphi-1} \ln t) + O(t \ln^{\varphi} t) = O(t^{(2-\varphi)} \beta^{\varphi-1} \ln t + t \ln^{\varphi} t) = o(\beta)$. Therefore, p's expected cost is less than that of the adversary.

Substituting $t = O(r^2)$ into the above analysis yields the favorability result above in Lemma 8 and, together, gives our result for the grid model in Theorem 3. □

4 Application 2: Application-Level DDoS Attacks

Typically in application-level DDoS attacks, a number of compromised clients, known collectively as a *botnet*, are employed to overwhelm a server with requests. These botnets have become commercialized with operators ("botmasters") renting out time to individuals for the purposes of launching attacks [20, 38].

We assume a model of botnet attacks similar to that described by Walfish *et al.* [63]. In this model, a request is cheap for a client to issue, expensive for the server to service, and all requests incur the same computational cost (heterogeneous requests can likely be handled as in [63]). There is a high-capacity communication channel and the crucial bottleneck is the server's inability to process a heavy request load.

The client rate is g requests per second. The aggregate botnet rate is R requests per second and this is assumed to be both relatively constant and the botnet's maximum possible rate. If the server is overloaded, it randomly drops excess requests. In this case, the good clients only receive a fraction $g/(g + R)$ of the servers resources; it is assumed that $R \gg g$ so that $g/(g + R)$ is very small.

Walfish *et al.* [63] propose a protocol SPEAK-UP for resisting DDoS attacks by having clients increase their sending rate such that their aggregate bandwidth G is on the same order as that of R. Since botnet machines are assumed to have already "maxed-out" their available bandwidth in attacking, SPEAK-UP greatly increases the chance that the server processes a legitimate request since $G/(G+R) \gg g/(g+R)$. A crucial component of SPEAK-UP is a front-end to the server called the "thinner" which controls which requests are seen by the server and asks a client to retry her request if it was previously dropped.

4.1 Our Protocol

We employ Case 2 of our 3-PLAYER SCENARIO PROTOCOL to achieve a SPEAK-UP-*like* algorithm with provable guarantees. Bandwidth (upstream and downstream rates in bits per second) is our measure of cost and, as such, our results should be interpreted as quantifying the expected upstream bandwidth required by the client and the expected downstream bandwidth with which the server should be provisioned. Using bandwidth as a form of currency has been previously employed by the research community [26, 56, 63].

The client plays the role of Alice where the message is a request; the server plays the role of Bob. This application falls into Case 2 of Theorem 1: a DDoS attack targets the server while communications from the server to the clients are not disrupted. The client and server are assumed to be synchronized such that they always agree on the current round and a maximum round number is set *a priori*. Such synchronization is certainly possible over Internet-connected machines and the maximum round value should be set to account for the level of DDoS resistance the participants wish to have; for most attacks, R is in the low hundreds of Mbits/second [55]. We give an overview of our protocol; the pseudocode is provided in Section C.

Send Phase: Each Send Phase occurs over a uniform and fixed duration Δ; for simplicity, we set $\Delta = 1$ second, and the slot length changes in each round appropriately. The client sends in each slot with probability $2/2^i$ with an expected 2^i upstream bits per second. The server listens in each slot with probability $2/2^i$ for an

expected 2^i downstream bits per second. If the received traffic substantially exceeds 2^i, requests are dropped; probabilistic listening and traffic measurement on the server side can be performed by the thinner [63].

Note that in each round, the client increases her sending rate in the Send Phase to "speak up". Any correct client that reaches its bandwidth limit remains at this limit for the duration of the protocol. When the maximum round number is reached, the clients maintain their sending rate until the thinner informs them that the attack has ended. For the purposes of analysis, a blocked slot occurs when Carol overwhelms the server with requests and the client's request is dropped in that slot. Define a send-blocked phase as one where Carol blocks at least $2^{2i}/2$ slots; therefore, Carol uses an upstream bandwidth of *at least* $2^{2i}/2$ bits per second. As in [63], if the thinner drops a request, it immediately asks the client to retry in the next round.

Ack Phase: The server does not increase its sending rate per round (only the client speaks up) since there are no attacks in the Ack Phase for Case 2. This simplifies the Ack Phase as mentioned in Section 2 in our discussion of Ack Failures; the server simply returns the requested data to the client at some reasonable rate.

The constants $S = J$ and L correspond to the rate of 1 bit per second. We assume upstream and downstream bandwidth are capped; this is true of residential Internet packages, as well as hosted services. In the case of residential service, upstream bandwidth is scarcer than downstream bandwidth, while servers are generally well-provisioned for both; this can be reflected in our cost constants. By Case 2 of Theorem 1 we have:

Corollary 1. *If Carol uses bandwidth R to attack, then the client's request is serviced, and the expected bandwidth (upstream and downstream) used by the client and the server is $O(R^{0.5})$.*

Bob can represent multiple good clients. We assume the same synchronization with the server; however, clients joining at different times are informed by the thinner of the current round. In order to be guaranteed *some* of the server's resources, the clients' expected aggregate bandwidth is $G = \Omega(R^{0.5})$. Therefore, our result quantifies the minimum expected aggregate upstream bandwidth for clients and the expected downstream bandwidth for the server required to ensure that total censorship is averted; in contrast, SPEAK-UP cannot make such a guarantee. This is useful for applications where a critical update or warning must be dessiminated, and delivery to even a handful of clients is sufficient since they may then share it with others (via multicast, peer-to-peer distribution, etc.).

As with SPEAK-UP, the probability of legitimate request being serviced is still $G/(G + R)$. In addition to admitting an analysis, our iterative approach of geometrically increasing the aggregrate bandwidth should mitigate attempts by Carol at launching short duration DDoS attacks in order to provoke a steep and disruptive traffic increase from correct clients. Our protocol is fair as described in Section 1.2 – the *aggregate* requirements of the bandwidth constrained clients is asymptotically equal to that of the well-provisioned server. Restating our result above in the context of multiple clients yields Theorem 4.

Finally, in order to achieve the same level of denial-of-service against a server that is defended by our protocol, Carol must procure a much larger botnet in order to obtain the necessary bandwidth; however, this comes at a cost. For example, one study found the cost of a single bot to be between $2 and $25 [20]. Therefore, since Carol's bandwidth requirements increase quadratically, her monetary costs increase significantly with the use of our protocol.

5 Conclusion

We have examined an abstract model of conflict over a communication channel. In the 3-Player Scenario, we remark that there is an $O(1)$ "up-front" cost per execution of our protocol when there are no send- or ack-blocking attacks. Similarly, there are small up-front costs for our other favorable protocols. This is the (tolerable) price for communication in the presence of a powerful adversary, even if that adversary is not necessarily very active. The golden ratio arises naturally from our analysis, and its appearance in this adversarial setting is interesting; an important open question is whether $\Omega(B^{\varphi-1} + 1)$ cost is necessary.

Also of interest is determining whether there are fair and favorable algorithms for other types of problems. An interesting problem to start with would be the problem of conflict over dissemination of an idea in a social network, using the models of Kempe et al. [34].

Acknowledgements: We thank Martin Karsten, Srinivasan Keshav, and James Horey for their valuable comments.

References

[1] Prolexic Technologies, Inc. http://www.prolexic.com.

[2] Esther Addley and Josh Halliday. "wikileaks supporters disrupt visa and mastercard sites in 'operation payback'". http://www.guardian.co.uk/world/2010/dec/08/wikileaks-visa-mastercard-operation-payback, 2010.

[3] Dan Alistarh, Seth Gilbert, Rachid Guerraoui, Zarko Milosevic, and Calvin Newport. Securing Your Every Bit: Reliable Broadcast in Byzantine Wireless Networks. In *Proceedings of the Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 50–59, 2010.

[4] Giuseppe Anastasi, A. Falchi, Andrea Passarella, Marco Conti, and Enrico Gregori. Performance Measurements of Motes Sensor Networks. In *Proceedings of the* 7^{th} *ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, pages 174–181, 2004.

[5] Nils Aschenbruck, Elmar Gerhards-Padilla, and Peter Martini. Simulative Evaluation of Adaptive Jamming Detection in Wireless Multi-hop Networks. In *International Conference on Distributed Computing Systems Workshops*, pages 213–220, 2010.

[6] Tuomas Aura, Pekka Nikander, and Jussipekka Leiwo. DOS-resistant Authentication with Client Puzzles. In *Proceedings of the* 8^{th} *International Workshop on Security Protocols*, pages 170–177, 2000.

[7] Baruch Awerbuch, Andrea Richa, and Christian Scheideler. A Jamming-Resistant MAC Protocol for Single-Hop Wireless Networks. In *Proceedings of the* 27^{th} *ACM Symposium on Principles of distributed computing (PODC)*, pages 45–54, 2008.

[8] J. Bardwell. Converting Signal Strength Percentage to dBm Values, 2002.

[9] Emrah Bayraktaroglu, Christopher King, Xin Liu, Guevara Noubir, Rajmohan Rajaraman, and Bishal Thapa. On the Performance of IEEE 802.11 Under Jamming. In *INFOCOM*, pages 1265–1273, 2008.

[10] Marin Bertier, Anne-Marie Kermarrec, and Guang Tan. Brief announcement: Reliable broadcast tolerating byzantine faults in a message-bounded radio network. In *Proceedings of the* 22^{nd} *International Symposium on Distributed Computing (DISC)*, pages 516–517, 2008.

[11] Marin Bertier, Anne-Marie Kermarrec, and Guang Tan. Message-Efficient Byzantine Fault-Tolerant Broadcast in a Multi-Hop Wireless Sensor Network. In *Proceedings of the International Conference on Distributed Computing Systems (ICDCS)*, pages 408–417, 2010.

[12] Vartika Bhandari and Nitin H. Vaidya. On Reliable Broadcast in a Radio Network. In *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, pages 138–147, 2005.

[13] Vartika Bhandari and Nitin H. Vaidya. On Reliable Broadcast in a Radio Network: A Simplified Characterization. Technical report, CSL, UIUC, May 2005.

[14] Vartika Bhandari and Nitin H. Vaidya. Reliable Broadcast in Wireless Networks with Probabilistic Failures. In *INFOCOM*, pages 715–723, 2007.

[15] Vartika Bhandhari, Jonathan Katz, Chiu-Yuen Koo, and Nitin Vaidya. Reliable Broadcast in Radio Networks: The Bounded Collision Case. In *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, pages 258 – 264, 2006.

[16] Jing Deng, Pramod K. Varshney, and Zygmunt J. Haas. A New Backoff Algorithm for the IEEE 802.11 Distributed Coordination Function. http://surface.syr.edu/eecs/85/.

[17] Shlomi Dolev, Seth Gilbert, Rachid Guerraoui, Fabian Kuhn, and Calvin Newport. The Wireless Synchronization Problem. In *Proceedings of the* 28^{th} *ACM symposium on Principles of distributed computing*, Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC), pages 190–199, 2009.

[18] Shlomi Dolev, Seth Gilbert, Rachid Guerraoui, and Calvin Newport. Gossiping in a Multi-channel Radio Network: An Oblivious Approach to Coping with Malicious Interference. In *Proceedings of the International Symposium on Distributed Computing (DISC)*, pages 208–222, 2007.

[19] Shlomi Dolev, Seth Gilbert, Rachid Guerraoui, and Calvin Newport. Secure communication over radio channels. In *Proceedings of the Symposium on Principles of Distributed Computing (PODC)*, pages 105–114, 2008.

[20] Jason Franklin, Vern Paxson, Adrian Perrig, and Stefan Savage. An Inquiry into the Nature and Causes of the Wealth of Internet Miscreants. In 14^{th} *ACM Conference on Computer and Communications Security*, pages 375–388, 2007.

[21] Saurabh Ganeriwal, Christina Pöpper, Srdjan Čapkun, and Mani B. Srivastava. Secure Time Synchronization in Sensor Networks. *ACM Transactions on Information and System Security*, 11(23), 2008.

[22] Lee Garber. Denial-of-Service Attacks Rip the Internet. *Computer*, 33(4):12–17, 2000.

[23] Seth Gilbert, Rachid Guerraoui, Dariusz Kowalski, and Calvin Newport. Interference-resilient information exchange. In *INFOCOM*, pages 2249–2257, 2009.

[24] Seth Gilbert, Rachid Guerraoui, and Calvin C. Newport. Of Malicious Motes and Suspicious Sensors: On the Efficiency of Malicious Interference in Wireless Networks. In *International Conference On Principles Of Distributed Systems (OPODIS)*, pages 215–229, 2006.

[25] Virgil D. Gligor. Guaranteeing Access in Spite of Distributed Service-Flooding Attacks. In *Security Protocols Workshop*, 2003.

[26] Carl A. Gunter, Sanjeev Khanna, Kaijun Tan, and Santosh Venkatesh. DoS Protection for Reliably Authenticated Broadcast. In *Proceedings of the* 11^{th} *Networks and Distributed System Security Symposium (NDSS)*, 2004.

[27] Ahsan Habib, Mohamed Hefeeda, and Bharat Bhargava. Detecting Service Violations and DoS Attacks. In *Proceedings of Internet Society Symposium on Network and Distributed System Security (NDSS)*, pages 177–189, 2003.

[28] Wendi Rabiner Heinzelman, Anantha Chandrakasan, and Hari Balakrishnan. Energy-Efficient Communication Protocol for Wireless Microsensor Networks. In *HICSS*, pages 3005–3014, 2000.

[29] C. Karlof and N. Sastry and D. Wagner. TinySec: A Link Layer Security Architecture for Wireless Sensor Networks. In *SenSys*, 2004, pp.162-175.

[30] Xin Liu, Guevara Noubir, Ravi Sundaram, and San Tan. SPREAD: Foiling Smart Jammers Using Multi-Layer Agility. In *INFOCOM*, pages 2536–2540, 2007.

[31] Kannan Srinivasan and P. Levis. RSSI is Under Appreciated. In *EmNets*, 2006.

[32] Akira Ichimura and Maiko Shigeno. A New Parameter for a Broadcast Algorithm with Locally Bounded Byzantine Faults. *Information Processing Letters*, 110(12-13):514–517, 2010.

[33] Ari Juels and John Brainard. Client Puzzles: A Cryptographic Countermeasure Against Connection Depletion Attacks. In *Networks and Distributed Security Systems (NDSS)*, pages 151–165, 1999.

[34] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the* 9^{th} *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 137–146. ACM, 2003.

[35] Valerie King, Cynthia Phillips, Jared Saia, and Maxwell Young. Sleeping on the Job: Energy-Efficient and Robust Broadcast for Radio Networks. In *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, pages 243–252, 2008.

[36] Chiu-Yuen Koo. Broadcast in Radio Networks Tolerating Byzantine Adversarial Behavior. In *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, pages 275–282, 2004.

[37] Y. W. Law, J. Doumen, and P. Hartel. Survey and Benchmark of Block Ciphers for Wireless Sensor Networks. *ACM Transactions on Sensor Networks*, 2(1):65–93, 2006.

[38] Michael Lesk. The New Front Line: Estonia under Cyberassault. IEEE *Security and Privacy*, 5(6):76–79, 2007.

[39] Yuan Li, Wei Ye, and John Heidemann. Energy and Latency Control in Low Duty Cycle MAC Protocols. In *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, pages 676–682, 2005.

[40] Guolong Lin and Guevara Noubir. On Link Layer Denial of Service in Data Wireless LANs. *Wireless Communications & Mobile Computing*, 5(3):273–284, 2005.

[41] Donggang Liu and Peng Ning. Multi-Level μTESLA: Broadcast Authentication for Distributed Sensor Networks. *ACM Transactions in Embedded Computing Systems*, 3:800–836, 2004.

[42] Ewen MacAskill. "wikileaks website pulled by amazon after u.s. political pressure'". http://www.guardian.co.uk/media/2010/dec/01/wikileaks-website-cables-servers-amazon, 2010.

[43] Robert Mackey. Latest Updates on Leak of U.S. Cables, Day 9. http://thelede.blogs.nytimes.com/2010/12/06/latest-updates-on-leak-of-u-s-cables-day-9/#operation-payback-plans-attacks-on-paypal, 2010.

[44] Dominic Meier, Yvonne Anne Pignolet, Stefan Schmid, and Roger Wattenhofer. Speed Dating Despite Jammers. In *Proceedings of the International Conference on Distributed Computing in Sensor Systems (DCOSS)*, pages 1–14, 2009.

[45] William G. Morein, Angelos Stavrou, Debra L. Cook, Angelos D. Keromytis, Vishal Misra, and Dan Rubenstein. Using Graphic Turing Tests to Counter Automated DDoS Attacks against Web Servers. In 10^{th} *ACM International Conference on Computer and Communications Security*, pages 8–19, 2003.

[46] Aristides Mpitziopoulos, Damianos Gavalas, Charalampos Konstantopoulos, and Grammati Pantziou. A Survey on Jamming Attacks and Countermeasures in WSNs. *IEEE Communications Surveys & Tutorials*, 11(4):42–56, 2009.

[47] Vishnu Navda, Aniruddha Bohra, Samrat Ganguly, and Dan Rubenstein. Using Channel Hopping to Increase 802.11 Resilience to Jamming Attacks. In *INFOCOM*, pages 2526–2530, 2007.

[48] Dragoş Niculescu. Interference Map for 802.11 Networks. In *Internet Measurement Comference (IMC)*, pages 339–350, 2007.

[49] Andrzej Pelc and David Peleg. Broadcasting with Locally Bounded Byzantine Faults. *Information Processing Letters*, 93(3):109–115, 2005.

[50] Andrzej Pelc and David Peleg. Feasibility and Complexity of Broadcasting with Random Transmission Failures. In *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, pages 334–341, 2005.

[51] Konstantinos Pelechrinis, Marios Iliofotou, and Srikanth V. Krishnamurthy. Denial of Service Attacks in Wireless Networks: The Case of Jammers. *To appear in IEEE Communications Surveys & Tutorials*, 2011.

[52] Joseph Polastre, Robert Szewczyk, and David Culler. Telos: Enabling Ultra-Low Power Wireless Research. In *IPSN*, 2005.

[53] Iyappan Ramachandran and Sumt Roy. Clear Channel Assessment in Energy-Constrained Wideband Wireless Networks. *IEEE Wireless Communications*, 14(3):70–78, 2007.

[54] Andrea Richa, Christian Scheideler, Stefan Schmid, and Jin Zhang. A Jamming-Resistant MAC Protocol for Multi-Hop Wireless Networks. In *Proceedings of the International Symposium on Distributed Computing (DISC)*, pages 179–193, 2010.

[55] Vyas Sekar and Jacobus Van Der Merwe. LADS: Large-scale Automated DDoS Detection System. In *Proceedings of the USENIX ATC*, pages 171–184, 2006.

[56] Micah Sherr, Michael Greenwald, Carl A. Gunter, Sanjeev Khanna, and Santosh S. Venkatesh. Mitigating DoS Attack Through Selective Bin Verification. In *Proceedings of the First international conference on Secure network protocols*, NPSEC'05, pages 7–12, 2005.

[57] Dongjin Son, Bhaskar Krishnamachari, and John Heidemann. Experimental study of the effects of Transmission Power Control and Blacklisting in Wireless Sensor Networks. In *Proceedings of the First IEEE Conference on Sensor and Adhoc Communication and Networks*, pages 289–298, Santa Clara, California, USA, October 2004. IEEE.

[58] Dongjin Son, Bhaskar Krishnamachari, and John Heidemann. Experimental Study of Concurrent Transmission in Wireless Sensor Networks. In *Proceedings of the* 4^{th} *International Conference on Embedded Networked Sensor Systems*, SenSys, pages 237–250, 2006.

[59] James Sundali and Rachel Croson. Biases in Casino Betting: The Hot Hand and the Gamblers Fallacy. *Judgment and Decision Making*, 1(1):1–12, 2006.

[60] Vinod Vaikuntanathan. Brief announcement: Broadcast in Radio Networks in the Presence of Byzantine Adversaries. In *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, 2005.

[61] Valerie King and Cynthia Phillips and Jared Saia and Maxwell Young. Sleeping on the Job: Energy-Efficient and Robust Broadcast for Radio Networks. *Accepted to Algorithmica*, 2010.

[62] Ashlee Vance. WikiLeaks Struggles to Stay Online After Attacks. http://www.nytimes.com/2010/12/04/world/europe/04domain.html?_r=2&hp, 2010.

[63] Michael Walfish, Mythili Vutukuru, Hari Balakrishnan, David Karger, and Scott Shenker. Ddos defense by offense. In *Proceedings of the 2006 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM)*, pages 303–314, 2006.

[64] John Paul Walters, Zhengqiang Liang, Weisong Shi, and Vipin Chaudhary. *Security in Distributed, Grid, Mobile, and Pervasive Computing. Chapter 17: Wireless Sensor Network Security: A Survey*. Auerbach Publications, 2007.

[65] R. Watro, D. Kong, S. Cuti, C. Gariner, C. Lynn, and P. Kruus. TinyPK: Securing Sensor Networks with Public Key Technology . In *SASN*, pages 59–64, 2004.

[66] Anthony D. Wood and John A. Stankovic. Denial of Service in Sensor Networks. *Computer*, 35(10):54–62, 2002.

[67] Wenyuan Xu, Ke Ma, Wade Trappe, and Yanyong Zhang. Jamming Sensor Networks: Attack and Defense Strategies. IEEE *Networks*, 20(3):41–47, 2006.

[68] Wenyuan Xu, Wade Trappe, Yanyong Zhang, and Timothy Wood. The Feasibility of Launching and Detecting Jamming Attacks in Wireless Networks. In *MobiHoc*, pages 46–57, 2005.

[69] Wei Ye, John Heidemann, and Deborah Estrin. An Energy-Efficient MAC Protocol for Wireless Sensor Networks. In *INFOCOM*, pages 1567–1576, 2002.

[70] Marco Zuniga and Bhaskar Krishnamachari. Analyzing the Transitional Region in Low Power Wireless Links. In *Proceedings of the* 1^{st} *IEEE International Conference on Sensor and Ad hoc Communications and Networks (SECON)*, pages 517–526, 2004.

Appendix

A significant amount of exposition and analysis had to be placed here in the appendix to meet the length constraints of our submission. Throughout our paper, there are references to the relevant sections in the event the reader wishes to verify our reasoning and analysis.

A Tolerating a Reactive Adversary: Further Discussion

Lemma 1 works for any adversary who makes her blocking decisions in slot j independently of channel activity. However, it is flawed for a reactive adversary who can detect channel activity for free and then block. The choice to set $q_j = 0$ (Carol blocks) in the proof depends solely on this ability to detect traffic since it is certain to be Alice's transmission that is disrupted. A fix to this problem is render this ability useless to Carol.

As an extreme example, assume that *all* slots in the Send Phase are used either by Alice to send m (as per our protocol) or Dave, whose transmissions of m' do not interest Carol, and the probability that a slot is used by Dave is higher. Then detecting channel activity does not help Carol decide on whether to block; all slots are used. Regardless of how she decides to act, Carol can do no better than picking slots independent of whether she detects channel activity. In other words, channel activity is no longer useful in informing Carol's decisions about whether to block.

But assuming *all* slots are active is problematic: (1) How is this guaranteed or coordinated? (2) Doesn't this much background traffic interfere with Bob's ability to receive from Alice? Instead, assume that other network traffic occurs such that Carol will always detect traffic on at least a constant fraction of slots in the Send Phase. Note that this does not help her block transmissions by Alice since she does not know the total amount of traffic that she will detect. Now, not all slots will necessarily be active. Upon detecting traffic, can Carol listen to a portion of the message to discover if it is m or m' and then decide on whether to block? Yes, but this is roughly as expensive (perhaps more so in WSNs) as simply blocking outright. So again, detecting channel activity does not inform Carol's decisions. This is the idea behind our analysis.

We note that the conclusion of our argument aligns with claims put forth in empirical results on reactive jamming in WSNs; that is, such behavior does not necessarily result in a more energy-efficient attack because the adversary must still be listening to the channel for broadcasts prior to committing itself to their disruption [68].

B Application 1: Wireless Sensor Networks

We present a full discussion along with our proofs that could not be provided in Section 3 which deals with our WSN application.

B.1 Proofs for LOCAL BROADCAST

Below are the full proofs leading to Theorem 2 wth regards to our protocol LOCAL BROADCAST. A round is again defined as send-jamming if at least $1/2$ of the slots in the Probabilistic Send Phase are jammed while a round is ack-jamming if at least $1/2$ the slots in Probabilistic Ack Phase are jammed or spoofed. We omit the constants S, L and J for simplicity.

Lemma 9. *Consider a non-send-jamming round. The probability that at least one correct receiver does not receive the message from Alice is less than* $1/n^2$.

Proof. Let s be the number of slots in the Probabilistic Send Phase of round i. Let $p_A = 3 \ln n / 2^i$ be the probability that Alice transmits in a particular slot. Let $p_b = 2/2^{(\varphi-1)i}$ be the probability that a particular correct receiver b listens in a particular slot. Let $X_j = 1$ if the message is not transmitted from Alice to receiver b in the j^{th} slot. Then $Pr[$ m is not successfully transmitted to the b during the Probabilistic Send Phase$]$=$Pr[X_1 X_2 \cdots X_s = 1]$=$Pr[X_s = 1 \mid X_1 X_2 \cdots X_{s-1} = 1] \cdot Pr[X_1 X_2 \cdots X_{s-1} = 1]$. Let $q_j = 1$ if the adversary does not jam given $X_1 X_2 \cdots X_{i-1}$; otherwise, let $q_j = 0$. The value of q_j can be selected arbitrarily by the adversary. Then $Pr[X_i = 1 \mid X_1 \cdots X_{i-1} = 1] = 1 - p_A p_b q_i = 1 - (6 \ln n / 2^{\varphi i}) q_j$. Then we have $Pr[X_1 X_2 \cdots X_s = 1] = (1 - p_A p_b q_1) \cdots (1 - p_A p_b q_s) \leq \prod_{j=1}^{s} (1 - p_A p_b q_j) \leq e^{-p_A p_b \sum_{j=1}^{s} q_j} <$

$1/n^3$ since $p_A p_b \sum q_j > (6 \ln n / 2^{\varphi i}) \cdot (2^{\varphi i}/2) = 3 \ln n$ given that this is a non-send-jamming round. Taking a union bound, the probability that at least one correct receiver has not received m is less than n^{-2}. □

We note Lemma 9 can be modified to handle a reactive adversary in the same way as done for 3-PLAYER SCENARIO PROTOCOL; we omit the details.

Lemma 10. *Assume that by round i all correct receivers have heard the message m. Assume that round i is non-ack-jamming. Then the probability that Alice retransmits the message in round $i+1$ is less than $1/n^2$.*

Proof. This is computed similarly to the proof of Lemma 9. Let s be the number of slots in the Probabilistic Ack Phase and let $p = 4/2^i$ be the probability that Alice listens in a slot. For slot j, define X_j such that $X_j = 1$ if Alice does not terminate. Then $Pr[$ Alice retransmits m in round $i+1] = Pr[X_1 X_2 \cdots X_s = 1]$. Let $q_j = 1$ if the adversary does not jam given $X_1 X_2 \cdots X_{i-1}$; otherwise, let $q_j = 0$. The q_j values are determined arbitrarily by the adversary who controls the faulty receivers. Since Alice terminates if and only if it listens and does not detect any activity, then $Pr[X_j = 1] = (1 - pq_j)$. Therefore, $Pr[X_1 X_2 \cdots X_s = 1] \leq e^{-p \sum_{j=1}^{s} q_j} < n^{-2}$ since $p \sum_{j=1}^{s} q_j > (4 \ln n / 2^i)(2^i/2) = 2 \ln n$ given that this is a non-ack-jamming round. □

Lemma 11. *Assume all receivers are correct and there are no send-jamming or ack-jamming rounds. Then the expected cost to Alice is $O(\ln^{\varphi} n)$ and the expected cost to any correct receiver is $O(\ln n)$.*

Proof. Let $d = \lg(4 \ln n)$. Using Lemma 9, the expected cost to Alice is at most:

$$\begin{aligned} & \sum_{i=d}^{\infty} n^{-2(i-d)} \cdot (2^{(\varphi-1)i} \cdot 3 \ln n + 2^{(\varphi-1)i+1} + 4 + 2^{(\varphi-1)i+1}) \\ = \; & O(\ln^{\varphi} n) + O\left(\ln n \cdot \sum_{k=1}^{\infty} \left(\frac{2^{(\varphi-1)}}{n^2} \right)^k \right) \\ = \; & O(\ln^{\varphi} n) \text{ by the geometric series.} \end{aligned}$$

Similarly, using Lemma 10, the expected cost to each receiver is at most:

$$\begin{aligned} & \sum_{i=d}^{\infty} n^{-2(i-d)} \cdot (2^{i+1} + 2^{(\varphi-1)i+1} + 2^i + 2^{(\varphi-1)i+1}) \\ = \; & O(\ln n) + O\left(\sum_{k=1}^{\infty} \left(\frac{2}{n^2} \right)^k \right) \\ = \; & O(\ln n) \text{ by the geometric series.} \end{aligned}$$

□

Lemma 12. *Assume there is at least one send-jamming round. The expected cost to Alice is $O(B^{\varphi-1} \ln n + \ln^{\varphi} n)$ and the expected cost to any correct receiver is $O(B^{\varphi-1} + \ln n)$.*

Proof. Let $i \geq \lceil \lg(4 \ln n) \rceil$ be the last round which is send-jamming and let j be the last round which is ack-jamming, $j \geq i$. Then the cost to the adversary is $B = \Omega(2^{\varphi i} + 2^j)$.
Alice: Using Lemma 10, the expected cost to Alice prior to successfully terminating is $O(2^{(\varphi-1)i} \ln n) + \sum_{k=1}^{\infty} n^{-2(k-1)} \cdot O(2^{(\varphi-1)(j+k)} \ln n) = O(2^{(\varphi-1)i} \ln n + 2^{(\varphi-1)j} \ln n)$. Therefore, in terms of B, the cost to Alice is $O(B^{\varphi-1} \ln n)$ and by Lemma 11, Alice's total expected cost is $O(B^{\varphi-1} \ln n + \ln^{\varphi} n)$.
Correct Receivers: In the worst case, all rounds up to i have been send-jamming, in which case the expected cost to each correct receiver up to the end of round $i+1$ is $O(2^i)$. Therefore, in terms of B, and using Lemma 11, the cost to each correct receiver is $O(B^{\varphi-1} + \ln n)$ noting that $1/\varphi = \varphi - 1$. □

B.2 Further Discussion on T_0

The amount of jamming in a neighborhood (but not the total for adversary in the network) is bounded by B_0 and known. This is required in [11, 15] and a similar assumption is made in [7, 54]. Alternatively, nodes may perpetually listen for transmissions; however, the receive state costs are then problematic. In order to let nodes sleep most of the time, our protocol synchronizes sending/receiving. Therefore, a bound seems necessary so that correct nodes know when to wake up as m propagates outward.

B.3 Why a Shared Schedule is Problematic

In our WSN application, we assume that messages from Alice can be authenticated using light-weight cryptographic techniques. Given this, we consider: might Alice and Bob (or even more players) also share a secret schedule? This would reduce the costs in Theorem 1 due to the Send Phase where neither player knows if the other is active with any certainty.

Unfortunately, such a schedule becomes known to the adversary if a player suffers a Byzantine fault and this causes problems in more general scenarios. For instance, consider the simple extension of Alice and two receivers. In our local broadcast problem, which is a key subroutine for our reliable broadcast protocol, Alice broadcasts to its two neighboring receivers concurrently in order to be fair. Therefore, both receivers must know when Alice transmits in the Send Phase. By corrupting one receiver, this schedule becomes known to the adversary who can then block transmissions by Alice perfectly and easily prevent the other receiver from receiving m. Clearly, this attack extends to the case where there are n receivers and Alice wants to achieve a local broadcast.

Other problems arise in a multi-hop scenario. For example, in our reliable broadcast protocol, each node listens to many different senders. A faulty receiver can interfere with many more senders by acting in the same manner as above for each of these senders. Therefore, by purposely avoiding a pre-set shared schedule, our use of randomness allows us to foil such attempts by the adversary.

B.4 Reliable Broadcast

Our expanded discussion and proofs with regards to reliable broadcast are given in this section.

B.4.1 The Las Vegas Guarantee in Multi-Hop WSNs

In addition to the rationale given in Section 1.1, the Las Vegas property is also valuable in multi-hop sensor networks for the following reason. Let n be the number of devices within transmitting distance of a device, and let N be the total number of devices in the network. Monte Carlo protocols that succeed with high probability in n are possible. However, typically, $n \ll N$ and messages will traverse multiple hops; consider $\Omega(N)$ hops. *Even if the failure probability for a single and the failure probability for each hop is $O(n^{-c})$ for some constant $c > 0$, or even $O(2^{-n})$, then communication fails along the chain with at least constant probability*. Alternatively, we might achieve protocols that succeed with high probability in N. However, in large networks, N may not be known *a priori*. Furthermore, achieving a high probability guarantee in N typically involves $\Omega(\log N)$ operations which, for large N, may be too costly. Therefore, by devising Las Vegas protocols, we avoid assumptions that are problematic given that $n \ll N$.

We note that transmission over the wireless medium is subject to error due to radio-irregularity and gray-zone effects. Does this reduce the utility of our Las Vegas guarantee? In many cases, we argue that it does not. Under fair weather conditions, the percentage of successfully received packets is nearly 100% up to a distance threshold exceeding 25 meters in the case of the MICA2DOT mote [4]. Other experimental studies have shown that communication is reliable so long as the signal-to-interference-plus-noise-ratio exceeds a threshold value [57, 58]; therefore, using a transmission power above this threshold yields highly reliable communication. Other experimental studies on the packet reception rate, which closely approximates the probability of successfully receiving a packet between two neighbouring nodes, is perfect up to a fixed distance [70]. Therefore, for an appropriate transmission power in dense sensor networks, communication over the wireless medium should not undermine our Las Vegas guarantee.

B.4.2 Reliable Broadcast in the Grid

We reiterate the grid model: each node $p(x, y)$ is situated at (x, y) in a grid. The dealer d (who is correct) is located at $(0, 0)$ and seeks to propagate m to all correct nodes in the network. When a node p sends

a message, all listening nodes within L_∞ distance r (i.e. the $(2r+1) \times (2r+1)$ square centered about p) receive the message; this *neighborhood* is denoted by $N(p)$. Analogous results hold for the Euclidean metric (see [13]). There are $t < (r/2)(2r+1)$ Byzantine nodes in any neighborhood. For any correct node p, the adversary can use its t Byzantine nodes in $N(p)$ to jam for up to $B_0 = t \cdot n_c$ slots total.

There is a global broadcast schedule (obeyed by the correct nodes) that assigns each node a slot for broadcasting; the ordering is always the same but the actual specification is unimportant (see [36] for an example). A *cycle* is defined as on full pass through a global broadcast schedule – we call these *transmit slots* – plus an additional n slots that we call *response slots*; we expand on this later.

Overview of the Protocol: The pseudocode is in Figure 3. Our protocol starts at slot 0 and synchronizes the timing of nodes for sending and listening. While this synchronization is not mathematically challenging, a full description yields an unreadable protocol. *For ease of exposition, our treatment addresses each node q in $C = \{q(x,y) | -r \le x \le r \ \wedge \ y \ge 0\}$; that is, a corridor of width $2r+1$ moving up from d.* Traversing the x-coordinates is nearly identical and the grid can be covered piecewise by these two types of corridors.

For each node p, define $A_p = \{q(u,v) \mid (a-r) \le u \le (a+z)$ and $(b+1) \le v \le (b+r)\}$, $B_p = \{q(u,v) \mid (a+z+1) \le u \le (a+r)$ and $(b+1) \le v \le (b+r)\}$ and $B'_p = \{q'(u',v') \mid (a+z+1-r) \le u' \le (a)$ and $(b+r+1) \le v' \le (b+2r)\}$ for $0 \le z \le r$. The set B'_p is obtained from B_p by shifting left by r units and up by r units; under this 1-to-1 translation, $q_1 \in B_p$ and $q_2 \in B'_p$ are sister nodes. The reader is referred to [13] or [61] for a more in-depth discussion of these sets.

While a full presentation is somewhat tedious, the main idea is that where a node would have broadcasted a message to a group of nodes in the protocol of Bhandari & Vaidya [13], we now use LOCAL BROADCAST to communicate a message to that group of nodes. In terms of the messages themselves, node q issues a COMMIT(q,m) message if q has committed to m. Node q_2 sends HEARD(q_2,q_1,m) if q_2 has received a message COMMIT(q_1,m). As in [13], p commits to m when it receives $t+1$ COMMIT(q,m) or HEARD(q_2,q_1,m) from node-disjoint paths all lying within a single $(2r+1) \times (2r+1)$ area.

We now discuss how to move from slots in LOCAL BROADCAST to slots in a cycle. In general, the transmit slots in a cycle are used by a node p to transmit a HEARD or COMMIT message via LOCAL BROADCAST to a receiving set of nodes R_p, while the response slots in a cycle are used by nodes in R_p to send back `req` messages to p. Therefore, there are up to n transmit slots needed. For simplicity, we do not go into detail about how these are set up; we simply assume that nodes in R_p know which response slot to use.

In our p in our pseudocode, $R_p = N(p) \cap C$ and p executes LOCAL BROADCAST(m,p,R_p) in the context of the global broadcast schedule. By this, we mean that a slot in the Probabilistic Sending Phase of LOCAL BROADCAST corresponds to p's transmit slot in some cycle, the next slot in that same Probabilistic Sending Phase corresponds to p's transmit slot in the next cycle, and so on. The same thing happens with the Deterministic Sending Phase. In both cases, the response slots are unused. Then in the Probabilistic Ack Phase and Deterministic Ack Phase, the response are used by each R_p set in the same fasion, where p now listens. By using a response slot, the nodes in R_p send back to p simultaneously as in LOCAL BROADCAST. Note that in the Probabilistic and Deterministic Ack phases, the transmit slots are now unused. Therefore, in each cycle, only the transmit slots or response slots, but not both, are used. For LOCAL BROADCAST running in at most D slots, executing LOCAL BROADCAST in the context of the global broadcast schedule requires at most D cycles. Figure 3 gives our pseudocode where $D = 25 \cdot (B_0 + \ln^{\varphi-1} n)^\varphi$ in concordance with Lemma 6.

We include the detailed proof of completeness for Theorem 3 by showing that each node eventually commits to the correct value m sent by the dealer. Our analysis in Section 3.3.1 already provides the cost analysis. The following Lemma 13 proves the correctness of our protocol in the grid; we emphasize that our argument follows that of [13].

Lemma 13. *Assume for each node p, $t < (r/2)(2r+1)$ nodes in $N(p)$ are Byzantine and used by Carol to disrupt p's communications for $\beta \le B_0$ time slots. Let $C = \{$Nodes q at $(x,y) | -r \le x \le r \ \wedge \ y \ge 0\}$ be a corridor of nodes in this network. Then* DOS-RESISTANT RELIABLE BROADCAST *achieves reliable broadcast in C.*

Proof. In [13], it is shown that each node $p(x,y)$ can obtain m by majority filtering on messages from $2t+1$ node-disjoint paths contained within a single $(2r+1) \times (2r+1)$ area since at least $t+1$ will be m. Our

DOS-RESISTANT RELIABLE BROADCAST

1: Starting in cycle 1, and ending no later than cycle D, node d executes LOCAL BROADCAST(m, d, R_d) and each node $i \in R_d$ commits to the first value it receives from d.

As in [13], $p(x, y)$ commits to m when, through Steps 4 & 5, it receives $t + 1$ COMMIT(q, m) or HEARD(q_2, q_1, m) from node-disjoint paths all lying within a single $(2r + 1) \times (2r + 1)$ area; our analysis shows this occurs in cycle $2yD - 1$. The following step is executed by each node p:

2: Starting in cycle $2yD$, and ending no later than cycle $(2y + 1)D - 1$, node $p(x, y)$ performs LOCAL BROADCAST(COMMIT$(p, m), p, R_p)$.

The following steps are executed by each node excluding those nodes in $N(d)$:

3: **for** $i = 0$ to $r - 1$ **do**

4: Starting in cycle $2(y-r+i)D$, and ending no later than cycle $2(y-r+i)D+D-1$, node $p(x, y)$ *listens* for COMMIT messages by executing LOCAL BROADCAST(COMMIT$(q, m), q(x', y'), R_q)$ with each node in row $y' = y - r + i$ in C and where $p \in R_q$.

5: Starting in cycle $2(y - r + i)D + D$, and ending no later than cycle $2(y - r + i)D + 2D - 1$, node $p(x, y)$ *listens* for HEARD messages by executing LOCAL BROADCAST(HEARD$(q_2, q_1, m), q_2, R_{q_2})$ with each node $q_2 \in B'_p$ in row $y + i$ and where $p \in R_{q_2}$.

6: Starting in cycle $2(y - r)D + D$, and ending no later than cycle $2(y - r)D + 2D - 1$, node q_2 *sends* a HEARD message by executing LOCAL BROADCAST(HEARD$(q_2, q_1, m), q_2, R_{q_2})$ where q_1, q_2 are sister nodes.

Figure 3: Pseudocode for DOS-RESISTANT RELIABLE BROADCAST.

correctness proof is similar; however, we argue along a corridor and show that nodes in the y^{th} row can commit to m by slot $2yD - 1$.

Base Case: Each node in $N(d)$ commits to the correct message m immediately upon hearing it directly from the dealer by cycle D. Therefore, clearly, every node $p(x, y) \in N(d)$ commits by cycle $2yD - 1$.

Induction Hypothesis: Let $-r \leq a \leq r$. If each correct node $p'(x', y') \in N(a, b)$ commits to m by cycle $2y'D - 1$, then each correct node $p(x, y) \in N(a, b + 1) - N(a, b)$ commits to m in cycle $2yD - 1$.

Induction Step: We now show $2t + 1$ connectedness within a single neighborhood and we argue simultaneously about the time required for p to hear messages along these disjoint paths. The node $p(x, y)$ lies in $N(a, b + 1) - N(a, b)$ and can be considered to have location $(a - r + z, b + r + 1)$ where $0 \leq z \leq r$ (the case for $r + 1 \leq z \leq 2r$ follows by symmetry). We demonstrate that there exist $r(2r + 1)$ node-disjoint paths $P_1, ..., P_{r(2r+1)}$ all lying within the same neighborhood and that the synchronization prescribed by our protocol is correct:

One-Hop Paths: the set of nodes $A_p = \{q(u, v) \mid (a - r) \leq u \leq (a + z)$ and $(b + 1) \leq v \leq (b + r)\}$ lie in $N(a, b)$ and neighbor p. Therefore, there are $r(r + z + 1)$ paths of the form $q \to p$ where $q \in A_p$.

By their position relative to $p(x, y)$, each correct node $q(u, v) \in A_p$ is such that $v = y - r + c$ for some fixed $c \in \{0, ..., r - 1\}$. Therefore, by the induction hypothesis, q commits to m by cycle $2(y-r+c)D-1$. By the protocol, $q(u, v)$ sends COMMIT messages using LOCAL BROADCAST in cycle $2vD = 2(y-r+c)D$ until cycle $2(v + 1)D - 1 = (2(y - r + c) + 1)D - 1$ at the latest. By the protocol, $p(x, y)$ listens for COMMIT messages from q starting in cycle $2(y - r + c)D$ until v $(2(y - r + c) + 1)D - 1$ at the latest; note that p listens to many executions of LOCAL BROADCAST containing HEARD messages, but we focus on this particular one from q. Therefore, p and q are synchronized in the execution of LOCAL BROADCAST and p will receive q's message by cycle $(2(y - r + c) + 1)D - 1 = (2(b + c + 1) + 1)D - 1$ at the latest. Since this occurs for all nodes in A_p, node p has received all COMMIT messages from A_p by cycle $(2(y - 1) + 1)D - 1 = (2(b + r) + 1)D - 1 \leq (2(b + r + 1) + 1)D - 1 = 2yD - 1$.

Two-Hop Paths: consider the sets $B_p = \{q(u, v) \mid (a + z + 1) \leq u \leq (a + r)$ and $(b + 1) \leq v \leq (b + r)\}$ and $B'_p = \{q'(u', v') \mid (a + z + 1 - r) \leq u' \leq (a)$ and $(b + r + 1) \leq v' \leq (b + 2r)\}$. The nodes in B_p lie in $N(a, b)$ while the nodes in B'_p lie in $N(p)$. Moreover, the set B'_p is obtained by shifting left by r units and up by r units. Recall that there is a one-to-one mapping between the nodes in B_p and the nodes in B'_p;

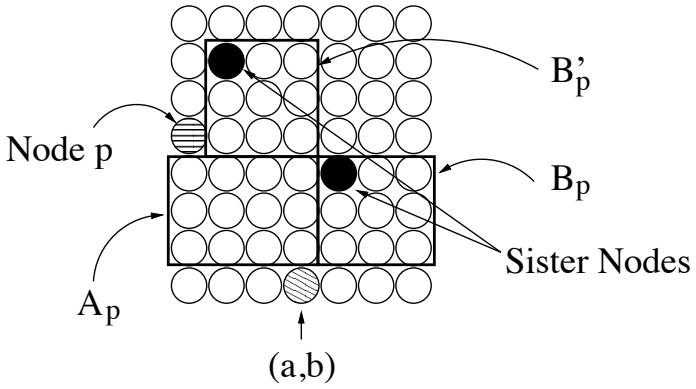

Figure 4: Depiction of the sets A_p, B_p, B'_p and sister nodes for a particular node p in the grid. Here $a, b, z = 0$ and $r = 3$ so the corridor C has width $2r + 1 = 7$.

these are sister nodes. There are $r(r - z)$ paths of the form $q \to q' \to p$. Figure 5 illustrates the sets and sister nodes for a particular p.

Consider a correct node $q(u, v) \in B_p$ and its sister node $q'(u', v') \in B'_p$ where $v' = v + r$ by definition. Again, given the location of $q(u, v)$ relative to $p(x, y)$, we have $v = y-r+c$ for some fixed $c \in \{0, ..., r-1\}$. By the induction hypothesis, q commits to m by cycle $2vD - 1$. Then by DoS-Resistant Reliable Broadcast, q sends a COMMIT message using Local Broadcast in cycle $2vD = 2(y - r + c)D$ until cycle $2(v+1)D-1 = (2(y-r+c)+1)D-1$ at the latest. Again, this is the particular execution of Local Broadcast between q and q'; q performs others. By DoS-Resistant Reliable Broadcast, $q'(u', v')$ receives COMMIT messages from q using Local Broadcast starting in cycle $2(v' - r + c)D = 2vD = 2(y-r+c)D$ and ending no later than cycle $2(v'-r+c+1)D-1 = 2(v+1)D-1 = (2(y-r+c)+1)D-1$. Therefore, q and q' are synchronized in the execution of Local Broadcast and q' will receive q's message by cycle $(2(y - r + c) + 1)D - 1 \leq 2yD - 1$ at the latest.

By the above, each node $q'(u', v') \in B'_p$ can start sending a HEARD message using Local Broadcast in cycle $2(v' - r)D + D$ and ending no later than cycle $2(v' - r)D + 2D - 1$. Starting in cycle $2(y - r + c)D + D$, node $p(x, y)$ uses Local Broadcast to listen for a HEARD message from $q'(u', v')$ where $v' = y + c$. Therefore, p is listening to q' starting in $2(y-r+c)D+D = 2(v'-r)D+D$ and ending no later than $2(v' - r)D + 2D - 1$; p and q' are synchronized. Therefore, p receives all HEARD messages by cycle $2(v' - r)D + 2D - 1$ when $v' = y + r - 1$; that is, by cycle $2(y - 1)D + 2D - 1 = 2yD - 1$.

Therefore, a total of $r(r+z+1)+r(r-z) = r(2r+1)$ node-disjoint paths from $N(a, b)$ to $PN(a, b)$ exist, all lying in in a single neighborhood $N(a, b+r+1)$. For an adversary corrupting $t < (r/2)(2r+1)$ nodes, a correct node can majority filter to obtain m. Furthermore, we have shown that any $p(x, y) \in N(a, b + 1)$ executes Local Broadcast $r(2r + 1) = O(r^2)$ times in order to receives all COMMIT and HEARD messages by cycle $2yD - 1$. Therefore, p can commit to the correct message by cycle $2yD - 1$; this concludes the induction. □

Finally, we reiterate that proving reliable broadcast in a corridor is sufficient as the entire grid can be covered piecewise by such corridors.

B.5 Further Discussion on Our Result in the Grid

A more contemplative point concerns the maximum number of Byzantine faults per broadcast neighborhood $t = (r/2)(2r + 1) - 1$ for which reliable broadcast is shown to be feasible in the presence of a jamming adversary in [15]; recall that previous protocols require that correct nodes possess more energy. However, this is problematic since, if nodes are subverted, it seems more reasonable to assume that Byzantine and correct nodes will each possess roughly equal energy. Assume that each node can be active for n_c slots. Consider the particular situation where the adversary targets p by having each of its nodes in $N(p)$ jam for n_c instances. This exhausts p's energy and *reliable broadcast fails*; this attack is suggested in [15]. Under

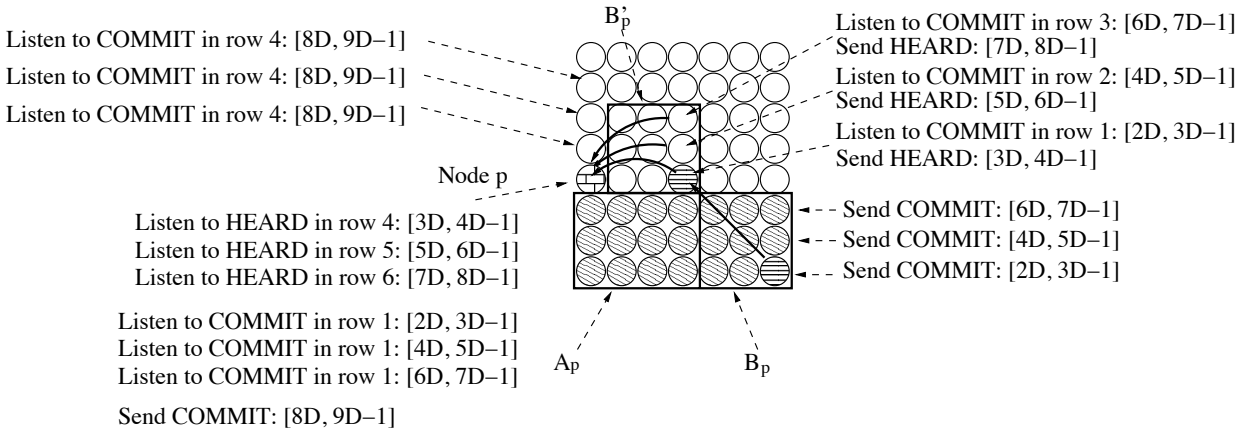

Figure 5: An example of some steps of the protocol for $r = 3$. The node in row 4 highlighted with the horizontal lines in the B'_p listens to a COMMIT message from its sister node in B_p by partaking in LOCAL BROADCAST as a receiver from cycle $2D$ to cycle $3D - 1$. Then, in cycle $3D$ to cycle $4D - 1$, that node uses LOCAL BROADCAST to send a HEARD message to p who is listening in this execution of LOCAL BROADCAST from cycle $3D$ to cycle $4D - 1$. The listening for p for each row is described on the left; note the synchronization. We also illustrate that those nodes above p will be listening for p's COMMIT message using LOCAL BROADCAST at the appropriate time.

our reliable broadcast protocol, such an attack can be mitigated. To see this, we examine the asymptotic upper bound on t for which our protocol still can admit reliable broadcast.

We want to know when n_c is sufficient to support the expected cost incurred by such an attack. We assume p has n_c energy and we know from Lemma 8 that the cost to p of t nodes each jamming for n_c time slots is $O(r^{2(2-\varphi)}(t \cdot n_c)^{\varphi-1} \ln r + r^2 \ln^{\varphi} r)$. Therefore, we ask: for what value of t is p's available energy $n_c = \omega(r^{2(2-\varphi)}(t \cdot n_c)^{\varphi-1} \ln r + r^2 \ln^{\varphi} r)$? Solving yields $t = o((n_c^{2-\varphi} - n_c^{1-\varphi} r^2 \ln^{\varphi} r)/r^{2(2-\varphi)} \ln r)^{\varphi}$. Although, we note that for this t value, p's survival is not guaranteed, this is an improvement over previous results which simply cannot tolerate this attack. Under our protocol, we also note that for larger values of t, the adversary can expect to disable p by using enough of its nodes and it is currently an open question whether there is a reliable broadcast protocol that can tolerate larger t values. In this sense, our protocol and analysis illustrates the importance of accounting for both send and receive state costs when considering bounds on t in the grid when jamming is possible.

Our results for reliable broadcast are also novel in accounting for both sending and listening costs, while most previous results address focus on sending costs only. Under our protocols, nodes spend a substantial amount of time in the energy-efficient sleep state, waking up to send or listen. We note that switching from the sleep state to an active state incurs some cost (less than transmitting or receiving); however, in our protocols, the number of state switches is limited by the number of active slots and, therefore, our asymptotic analysis holds.

B.6 Reliable Broadcast in General Topologies

In this section, we present our results for reliable broadcast on an arbitrary graph $G = (V, E)$. Pelc & Peleg [49] examine a generalization of the t-locally bounded fault model; that is, where each node contains at most t Byzantine nodes within its neighborhood. Specifically, they examine the broadcast protocol of Koo [36], which the authors call the *Certified Propagation Algorithm* (CPA), with the aim of establishing conditions for which it achieves reliable broadcast under arbitrary graphs in contrast to the grid model. Again, CPA addresses the case where all nodes obey a global broadcast schedule (i.e. there is no jamming adversary). Pelc & Peleg [49] define $X(p, d)$ to be the number of nodes in p's neighborhood $N(p)$ that are closer to d than p and then introduce the parameter $X(G) = \min\{X(p, d) \mid p, d \in V,\ (p, s) \notin E\}$. One of their main results is that, for *any* graph G with dealer d such that $t < X(G)/2$, CPA achieves reliable broadcast. For our purposes, define for each node p the set of nodes $X(p)$ to be those $X(p, d)$ nodes closer to the dealer than to p. Clearly, it is possible to identify $X(p)$ in polynomial time and so we observe:

Certified Propagation Algorithm (Koo [36] and Pelc & Peleg [49])

- The dealer d sends the message to all of its neighbors and terminates.
- For a correct node $u \in N(d)$, upon receiving m from d it commits to m, node u announces this committment of its neighbors and terminates.
- If a node is not a neighbor of the source, then upon receiving $t+1$ copies of m from $t+1$ distinct neighbors, it commits to m, and announces this committment to its neighbors and terminates.

Figure 6: Pseudocode for the Certified Propagation Algorithm (CPA).

Observation 1. *If the topology of G and the location of d is known to all nodes, then each node p can calculate $X(p)$.*

B.6.1 A Favorable Protocol in General Topologies

The pseudocode for CPA is given in Figure 6. Note that, unless a node is sending in the slot allotted to it by the global broadcast schedule, or it has terminated, it is perpetually listening. We aim to remove this wasteful listening by synchronizing the sending and listening of nodes.

In the context of CPA, we call a single iteration of the global broadcast schedule a *broadcast round*. Throughout, assume that time is measured from when the dealer first broadcasts m in broadcast round 0. Under CPA, regardless of the worst case delay imposed by the adversary, there is a broadcast round where p must have received at least $t+1$ messages from distinct correct nodes in $X(p)$ allowing p to commit to m; denote this broadcast round by s_p. Note, that in any execution of reliable broadcast, p may actually be able to commit before broadcast round s_p, but s_p is the maximum broadcast round in which p is guaranteed to have all the information it needs to commit to m regardless of how the adversarial nodes behave.

Since a correct node $u \in N(d)$ accepts what it hears from the dealer d immediately, and d's broadcast round is 0, $s_u = 1$. For nodes not in $N(d)$, the situation is slightly more complicated. In the grid, for node $p(x, y)$, we were able to compute s_p explicitly (in terms of cycles) as $2yD - 1$ in the corridor C (see proof of Lemma 13). Here, unlike with the grid, we cannot specify s_p explicitly for any graph G because it is dependent on the topology; however, by the correctness of CPA, every node eventually commits and so s_p must exist for each node p. In fact, our protocol based on CPA is simpler than that in the grid because the protocol of Bhandari & Vaidya [13] uses HEARD messages (which makes the synchronization tedious), while CPA uses only COMMIT messages.

For a *fixed* G whose topology is known to all nodes (including knowledge of where the dealer d is situated), each node p can calculate s_p. This is done by simulating the propagation of m using CPA. In this simulation, each node p has the maximum $t = X(G) - 1$ Byzantine nodes in $X(p)$ and these Byzantine nodes send their faulty messages prior to the $t+1$ correct responses in order delay propagation of m for as long as possible. By assuming that every $X(p)$ has the maximum number of Byzantine nodes, the actual placement of the Byzantine nodes in G does not affect the worst case broadcast time s_p. In tracing this propagation, any node can calculate s_p for any node p. Therefore, we have another observation:

Observation 2. *If the topology of G and the location of d is known to all nodes, then each node can calculate s_p for any node p.*

Now, consider the following minor modifications to CPA: (1) each correct node p only listens to $q \in X(p)$ in broadcast round $s_q + 1$, and (2) each correct node p only sends its commit message in broadcast round s_p+1. In all other slots, a node p is sleeping. This is a minor modification of CPA, call it CPA$_0$. These modifications synchronize the sending/listening and allow nodes to otherwise sleep instead of perpetually listening as in CPA. The pseudocode for CPA$_0$ is given in Figure 7.

Lemma 14. *If CPA achieves reliable broadcast, then CPA$_0$ achieves reliable broadcast.*

Proof. For every node p, assume $X(p)$ has the maximum $t = X(G) - 1$ Byzantine nodes and that these Byzantine nodes all send their messages to p ahead of the correct nodes in $X(p)$ according to the broadcast schedule. Pelc & Peleg [49] showed that CPA is correct in this situation (their result is independent of any particular ordering of sending in the broadcast schedule; that is, CPA remains correct if Byzantine nodes always send first). Therefore, in this case, each correct node p would receive a committment from $q \in X(p)$

CPA$_0$

- In broadcast round 0, the dealer d sends the message to all of its neighbors and terminates.
- For a correct node $u \in N(d)$, u listens in broadcast round 0 and accepts m as correct, announces this committment to its neighbors in broadcast round 1, and terminates.
- If a node p is not a neighbor of the source, then p listens to each neighbor $q \in X(p)$ in broadcast round $s_q + 1$; otherwise, p sleeps. Upon receiving $t+1$ copies of m from $t+1$ distinct neighbors in $X(p)$, it accepts m as correct, announces this committment to its neighbors in broadcast round $s_p + 1$, and terminates.

Figure 7: Pseudocode for CPA$_0$.

FCPA

- The dealer d sends the message to all of its neighbors using LOCAL BROADCAST(m, d, $N(d)$) and terminates after at most D cycles.
- If node $u \in N(d)$, then upon receiving m from d via LOCAL BROADCAST(m, d, $N(d)$), it accepts m as correct, announces this committment to its neighbors in cycle D, and terminates.
- If a node p is not a neighbor of the source, then p listens to each neighbor $q \in X(p)$ via LOCAL BROADCAST(m, q, $N(q)$) starting in cycle $s_q \cdot D + 1$ and ending by $(s_q + 1) \cdot D$; otherwise, p sleeps. Upon receiving $t + 1$ copies of m in this fashion from $t + 1$ distinct neighbors in $X(p)$, it accepts m as correct, announces this committment to its neighbors using LOCAL BROADCAST(m, p, $N(p)$) in broadcast cycle $s_p \cdot D + 1$, and terminates by cycle $(s_p + 1) \cdot D$.

Figure 8: Pseudocode for FCPA.

in broadcast round $s_q + 1$ and node p would announce its committment in round $s_p + 1$. This is exactly what happens in CPA$_0$ with nodes sleeping otherwise. Therefore, if CPA achieves reliable broadcast, then so does CPA$_0$. □

Define a cycle as done before in the grid. In Figure 8, we provide pseudocode for a fair and favorable reliable broadcast algorithm FCPA that tolerates the jamming adversary described in Theorem 3.

Lemma 15. *Assume CPA achieves reliable broadcast on a graph G. Then* FCPA *guarantees reliable broadcast on G.*

Proof. Using FCPA, we claim that every correct node can commit by cycle $s_p \cdot D$. To prove this, assume the opposite: that some node p does not commit to the correct value by cycle $s_p \cdot D$. Then, there is some correct node in $q \in X(p)$ that: (1) could not commit to a message by time slot $s_q \cdot D$ (and could not send p a committment message), or (2) committed to a wrong message (and sent that wrong message to p). Note that the time for any node u to send its commit message to v is at most D cycles by Lemma 6. Therefore, if q cannot commit by $s_q \cdot D$ in FCPA, then q cannot commit by cycle s_q in CPA$_0$; therefore, CPA$_0$ fails to achieve reliable broadcast. Similarly, if q commits to the wrong value in FCPA, then it would also commit to the wrong in CPA$_0$, and so CPA$_0$ fails to achieve reliable broadcast. However, if CPA_0 fails to achieve reliable broadcast, then by the contrapositive of Lemma 14, this contradicts the assumption that CPA achieves reliable broadcast. □

Finally, note that each node will execute LOCAL BROADCAST for $O(t)$ times as a listener, and then execute it once as a sender. Combining Lemma 15 above with the cost analysis in Section 3.3.1 yields the results stated in Theorem 3 for general graphs.

C Application 2: Application-Level DDoS Attacks

We present further discusion of our result in the client-server model. Our pseudocode is given in Figure 9.

Note that the Ack Phase is simplified due to the fact that attacks do not occur in this phase for Case 2 of the 3-PLAYER SCENARIO PROTOCOL. Like [63], our protocol is suitable for applications where there is

DOS-RESISTANT CLIENT-SERVER COMMUNICATION for round $i \geq 2$

Send Phase: For each of the 2^{2i} slots do

- The client sends her request with probability $2/2^i$.
- The server (via the thinner) admits listens with probability $2/2^i$.

Ack Phase:

- The server sends back the requested data.
- The client listens.

If the client receives her data, she terminates; otherwise, the thinner tells her to retry in the next round.

Figure 9: Pseudocode for the application of Case 2 of our 3-Player Scenario to the client-server scenario.

no pre-defined clientele (so the server cannot traffic filter) and the clientele can be non-human (so "proof-of-humanity" tests cannot be relied upon solely). Unlike the wireless domain, we do not address reactive adversaries. Determining when a player is sending over the wire in order to control when its traffic arrives at the targeted player seems beyond the capability of a realistic attacker.

Unlike the WSN domain, neither player sleeps since energy is not a concern. Rather, the client terminates in the sense that her current request is satisfied. The server is always awake to process incoming requests; however, it can be said to terminate in the sense that the client's current request is not re-serviced. The client and server are assumed to be synchronized so that they always agree on the current round and a maximum round number is set *a priori*. This synchronization is certainly possible over Internet-connected machines and the maximum round value should be set to account for the level of DDoS resistance the participants wish to have; again, R upper bounds this rate and it has been observed that most attacks have R in the low hundreds of Mbits/second [55]. Our protocol is used when the server detects a DDoS attack (for details, see [27] and references therein). Finally, we mention that the effects of increasing the client traffic are examined experimentally by Walfish *et al.* [63] and shown to be acceptable.