Secure Capacity of Wireless Broadcast Networks[*] [†]

Guanfeng Liang, Rachit Agarwal, Nitin H. Vaidya
Department of Electrical and Computer Engineering,
University of Illinois at Urbana-Champaign,
IL, USA
{gliang2, agarwa16, nhv}@illinois.edu

ABSTRACT

We give a constructive characterization for the capacity of wireless broadcast networks that are prone to Byzantine attacks. The adversary controls a single node in the network and can modify the packets flowing through the node. The central trade-off in such a scenario is that of security and throughput. We define *secure capacity* as the highest possible transmission rate from the source such that the destination can detect any modification in the information packets. Prior work in this direction mainly concentrate on nodes performing random network coding, where the capacity is shown to be bounded by $C - z_0$, where C is the minimum cut between the source and the destination and z_0 is the maximum number of packets the adversary can modify.

We show that by carefully designing the transmission scheme, rates higher than $C - z_0$ are achievable. In particular, we show that some nodes in the network carefully duplicating the packets results in increased capacity of the network. In order to efficiently search over all the possible transmission strategies, we formulate the problem of characterizing secure capacity as a linear optimization program. We give an explicit routing (and duplication) strategy that achieves the capacity given by the optimal solution of the optimization program, thus establishing the secure capacity of the given wireless broadcast network. We also show that there exist networks in which secure capacity with in-network monitoring can be arbitrarily larger than that without in-network monitoring.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: Data Communications; C.2.6 [**Computer-Communication Networks**]: Wireless Communications

General Terms

Algorithm, Design, Theory

Keywords

Wireless Networks, Byzantine Attack, Capacity, Monitoring

1. INTRODUCTION

Security and reliability are at the forefront of the networking research. It is becoming increasingly crucial to quantify the trade-off between security and traditional quality of service parameters such as throughput and delay. While characterization of this trade-off is relevant in all forms of networking, it is particularly significant in the case of wireless networks, where the medium is shared, and hence is highly susceptible to attacks. It is important for the network to be able to detect any attack in the network, while being able to provide high performance under attack scenarios.

In this paper, we explore the security-throughput trade-off in wireless broadcast networks. An adversary in a wireless network can attack in several ways: it could inject erroneous packets on the outgoing link of the node it attacks on; it could cause interference on the links that are close enough to the node it attacks on; or it can jam the outgoing link it attacks on. In this paper, we study the first kind of attack, *i.e.*, when the adversary in the network injects erroneous packets on its outgoing link. We refer to such an adversary as a Byzantine adversary henceforth, and call such attacks Byzantine attacks.

We define the *secure capacity* of the network as the maximum number of data packets that can be sent from the source to the destination, under the condition that any Byzantine attack can be detected at the destination. In this paper, we consider a *pure broadcast transmission* model in which each node is restricted to broadcasting all the packets it has to all its neighbors. We give a constructive characterization of the secure capacity of networks for the case of a single Byzantine attacker in the network. Under the broadcast model, we show that combining end to end error detection scheme with duplication of packets (whenever possible) increases the capacity of the network.

We approach the problem by formulating a set of constraints that capture the intermediate nodes forwarding or duplicating the information packets. We say that a node is forwarding the incoming packets when it does not mix any of the incoming data packets and simply forwards one or all of the

[*]Submitted for conference publication on September 28, 2009.
[†]This research was supported in part by Army Research Office grant W-911-NF-0710287

data information packets. We say that a node is duplicating a packet if there are multiple copies of the same data packet on its outgoing link. These constraints, combined with a search over the space of solutions that satisfy these constraints, give us an upper bound on the secure capacity of the network. The search space we consider is that of neighbors of the source doing error detection coding and the rest of the nodes in the network either forwarding or duplicating the incoming packets.

We give a routing strategy that allows us to achieve the upper bound given by the search over the space of solutions. We believe that our search space also contains the set of solutions where the intermediate nodes in the network may mix the information packets (known as network coding). In other words, we believe that intermediate nodes in the network doing network coding does not increase the capacity of the network under such Byzantine attacks. Finally, using some examples, we show that the secure capacity of the network can be increased if intermediate nodes in the network may perform monitoring.

1.1 Related Works

The broadcast nature of wireless networks expose them to attacks that are not encountered in wired networks. The shared medium allows an attacker to easily eavesdrop over the information being transmitted, tamper with the messages flowing through the nodes under attack, and, allows the attacker not only to jam the packets flowing through the node it attacks on but also on the links that it could interfere on. Encryption and authentication techniques are expensive approaches to confront these attacks. Such solutions are not only expensive, but may also be impractical for resource constrained networks. This leads to the trade-off between security and resource consumption in the network. This trade-off has been explored in some recent works [17, 1, 16, 11, 12], which characterize the trade-off between security and message complexity for the case of an attacker tampering the packets.

Recently, the security issue in network coding systems has drawn much attention. Due to the *mixing* nature of network coding, such systems are subjects to a severe security threat, known as a *pollution attack*, where attackers inject corrupted packets into the network. Several solutions to address pollution attacks in intra-flow coding systems use special-crafted digital signatures [10, 18, 19, 14] or hash functions [13, 5]. Non-cryptographic solutions have also been proposed [8, 9]. Two practical schemes are proposed in [3] to address pollution attacks against network coding in wireless mesh networks without requiring complex cryptographic functions and incure little overhead. The above results, in the context of network coding, consider only the case of so-called random network coding, where intermediate nodes in the network randomly combine the incoming information packets. With such a transmission scheme, it has been shown that the secure capacity of the network is $C - z_0$, where C is the maximum number of packet that can be transmitted from the source to the destination in the absence of the adversary and z_0 is the maximum number of packets that the adversary can modify. Our approach is significantly different, in that, we show that by carefully designing a routing scheme, including duplicating of data packets when possible, the secure capacity of the network can be increased.

Finally, the impact of in-network monitoring (where intermediate nodes in the network exploit the broadcast medium to perform monitoring using overhearing) on secure capacity of wireless network has been studied empirically in some recent works [15, 4]. However, these works have not characterized the potential benefits of such in-network monitoring. We believe that further work on developing a theoretical framework is required in order to quantify the benefits of in-network monitoring. We take a first step in this direction, demonstrating that there exist networks for which the secure capacity with in-network monitoring is arbitrarily larger than that without in-network monitoring.

1.2 Organization of the Paper

The rest of the paper is organized as follows. In Section 2, we illustrate the ideas in the paper using some simple examples. Section 3 covers the formal definitions of the network and the adversarial model considered in the paper. We formulate the set of constraints for intermediate nodes in the network forwarding or duplicating the data in Section 4. The set of constraints formulated in Section 4 allow us to derive a simple expression for the upper bound on the secure capacity of the networks, discussed in Section 5. We also give a routing strategy in this section that achieves the derived upper bound, thereby establishing the secure capacity of wireless broadcast networks. We show, using some simple examples in Section 6, that the secure capacity of the network with intermediate nodes performing monitoring can be arbitrarily larger than the secure capacity derived in Section 5. We close the paper with future research directions in Section 7.

2. ILLUSTRATIVE EXAMPLES

In this section, we illustrate the ideas explored in this paper using some simple network topologies. To ease the discussion, we informally describe some of the terms used in this section. The *broadcast capacity* of a node is the maximum number of packets that node can transmit to its neighbors per unit time. To avoid the notion of time, we will generally refer to the number of *packets transmitted per unit time by a node (link) as the rate of that node (link)*. Any modification by a node in the packet being routed through that node, as discussed in Section 1, is referred to as Byzantine attack.

Consider the network shown in Fig. 1. First assume that all the relay nodes can only forward one of the data packets received from the source to d. In such a case, to be able to detect any Byzantine attack in the network, we claim that s can only transmit at rate 1. If s transmits 2 packets per unit time, say a and b, without loss of generality, assume that r_1 and r_2 forward packet a and r_3 forwards packet b. In such a case, the attacker can attack at r_3 and modify b without being detected at d. Hence, to be able to detect Byzantine modification, s can only transmit at rate 1.

On the other hand, if the relay nodes were allowed to mix the information, while r_1 and r_3 transmit packets a and b respectively, r_2 could transmit $a \oplus b$. It is easy to see that with such a transmission mechanism, Byzantine modification at any one of the relay nodes r_1, r_2 or r_3 will be successfully detected by d. Hence, a rate of 2 is achievable in the network. It is also easy to show that a rate more than 2 is not achievable

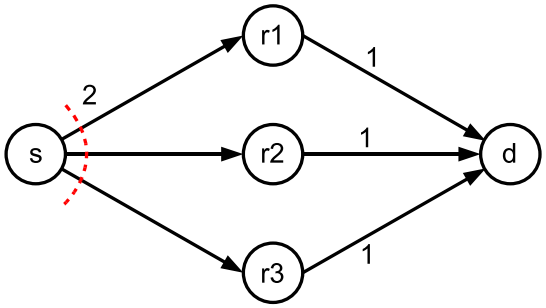

Figure 1: A single source (s), single destination (d) network with three unreliable nodes r_1, r_2 and r_3. s has a broadcast capacity of 2 units and each link $r_i \rightarrow d$ is of capacity 1 unit. All links are reliable. In this network, nodes simply forwarding the received packets does not achieve the secure capacity of the network.

in the network and hence, the secure capacity of the network is 2 packets per unit time.

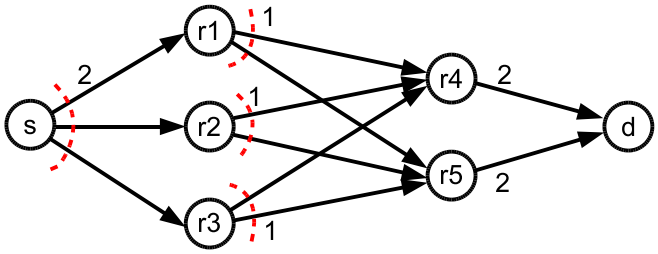

Figure 2: A network in which s has a broadcast capacity of 2 units, each of r_1, r_2, r_3 has a broadcast capacity of 1 unit and r_4, r_5 have a broadcast capacity of 2 units. In this network, packets must be mixed and duplicated within the network.

Now, consider the network shown in Fig. 2. The secure capacity of this network is 2 packets per unit time, which is achieved using the following strategy: s broadcasts a, b to all its neighbors. r_1, r_2, r_3 broadcast a, b and $a \oplus b$ respectively to all their neighbors. r_4 forwards a, b and r_5 forwards $a, a \oplus b$. It is easy to see that with this transmission strategy, attack at any single node in the network can be detected at the destination.

Finally, consider the network shown in Fig. 3. The capacity of this network is 1 packet per unit time, which is achieved by forwarding a single data packet at all the nodes. It is not very difficult to prove that no more than a single packet from the source to the destination can be transmitted per unit time.

We make the following two observations from the above examples:

- Similar to the network of Fig. 1, forwarding at the intermediate nodes in the network of Fig. 2 does not suffice. It is easy to see that if r_3 in Fig. 2 would send either of a or b rather than $a \oplus b$, the adversary can attack at one of the nodes and the destination will not be able to detect the attack. However, in the network of Fig. 3, forwarding does suffice for achieving the capacity.
- We also notice from the networks of Fig. 2 and Fig. 3 that in a capacity achieving scheme, some packets may need to be duplicated for the destination to be able to detect the attack (in this example, packet a for example). It is not a priori clear which packets need to be duplicated and which nodes need to duplicate the packets. Whether the duplication can be done or not is also not very clear. For example, in the network of Fig. 3 sending two packets and duplicating them at any of the intermediate nodes does not help.

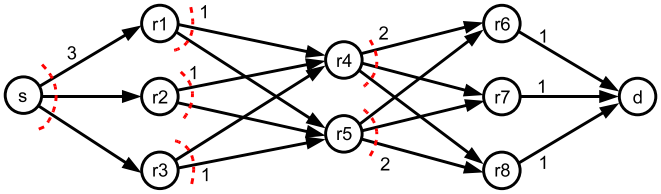

Figure 3: A network in which s has a broadcast capacity of 3 units, each of $r_1, r_2, r_3, r_6, r_7, r_8$ has a broadcast capacity of 1 unit and r_4, r_5 have a broadcast capacity of 2 units each. In this network, no coding at intermediate nodes is required. Some packets need to be duplicated.

In this paper, we take a first step towards formalizing the ideas of above three examples. In particular, we give a constructive characterization of the secure capacity of the networks in which the intermediate nodes can either forward the data packets or duplicate some of the data packets. To achieve this, we first formulate a set on constraints that capture the forwarding and duplication of packets at intermediate nodes and then search over the entire space of solutions satisfying these constraints. Before going to the formulation, let us formally describe our network and adversary models.

3. MODEL AND DEFINITIONS

The transmission model considered in this paper is "broadcast model", where each node is restricted to broadcasting the information packets it has to all its neighbors. To simplify notation, we consider information transmission from a single source to a single destination only.

3.1 Network Model

We model the network as a hypergraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ along with an associated function, $c = \langle c_e \rangle$, $c : \mathcal{E} \rightarrow \mathbb{Z}$, where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ is the set of hyperedges in $\mathcal{G}$. Each packet transmission corresponds to a hyperedge directed from the transmitting node to the set of receiver nodes. c is the function that defines the capacities of edges in $\mathcal{E}$. For any edge $e \in \mathcal{E}$, we denote the capacity of this edge by c_e. The hypergraph model captures both wired and wireless networks. For wired networks, the hyperedge is a simple point-to-point link. For wireless, each such hyperedge connects the transmitter to all nodes that hear the transmission. We denote, by $\mathcal{G}[\mathcal{V} \backslash v]$, the subgraph induced by removing node v from $\mathcal{V}$ and removing all incoming and outgoing hyperedges of v

from $\mathcal{E}$. For any $v \in \mathcal{V}$, denote by $N(v)$ the set of all those nodes that are on the hyperedge corresponding to v's transmission.

3.2 Transmission Model

As mentioned earlier, the transmission model considered in this paper is a "pure broadcast model". In particular, each node $v \in \mathcal{V}$ *broadcasts* the packets it has to all nodes $u \in N(v)$. Note that we do not have any restriction on $|N(v)|$, which means that the model also allows some nodes in the network to have a single neighbor resulting in a unicast from that node to its neighbor. However, if a node has multiple neighbors it is restricted to transmitting the same packets to all its neighbors, where the maximum number of packets that the node can transmit per unit time is equal to the capacity of the hyperedge corresponding to that node's transmission [1].

3.3 Adversary Model

We consider the omniscient adversary model [9], where nothing is hidden from the adversary. In particular, we assume that the adversary in the network is computationally unbounded, has complete knowledge of the network topology (the hypergraph) and the transmission scheme employed in the network (end-to-end error detection scheme and forwarding / mixing of packets at each node in the network). It can also observe all transmissions in the network and in this paper, we assume that it can attack at most a single node in the network, except for the source and destination. At the node it attacks, it can inject erroneous packets on the outgoing link of that node.

3.4 Definitions

A $(v, v', \mathcal{S})$-cut between any $v, v' \in \mathcal{V}$ is a partition of $\mathcal{V}$ into $\mathcal{S}$ and $\mathcal{V} - \mathcal{S}$ such that $v \in \mathcal{S}$ and $v' \in \mathcal{V} - \mathcal{S}$. A hyperedge $e = (u\ N(u))$ is said to be in a cut $(v, v', \mathcal{S})$ if $u \in \mathcal{S}$ and $N(u) \cap (\mathcal{V} - \mathcal{S}) \neq \phi$. The cut edge-set $\mathcal{E}(v, v', \mathcal{S})$ comprises $e \in \mathcal{E}$ such that e is on the cut $(v, v', \mathcal{S})$. The size of a cut $(v, v', \mathcal{S})$, denoted by $|\mathcal{E}(v, v', \mathcal{S})|$, is the sum of the capacities of all the hyperedges that are on that cut. The *min-cut between two nodes* $v, v' \in \mathcal{V}$ is defined as $\min_{\mathcal{S}} |\mathcal{E}(v, v', \mathcal{S})|$ and will be denoted as min-cut$(\mathcal{G}, c, v, v')$. In our formulation, the flow through a node may not be equal to the capacity of the outgoing hyperedge of that node. In such a case, we say that a function $z = \langle z_e \rangle$ defines the number of packets that are routed through node v. Clearly, z_v is bounded by the capacity of its outgoing edge. With such a flow assignment, we denote the cut as $(v, v', z, \mathcal{S})$ and the min-cut as min-cut$(\mathcal{G}, z, v, v')$. Min-cut from a set of nodes $V' \subset \mathcal{V}$ to some node v will be denoted as min-cut$(\mathcal{G}, z, V', v)$.

We also give the following definitions regarding the capacity expressions discussed in the paper:

- The *network capacity*, denoted by C, is the time-average of the maximum number of packets that can be delivered from the source to the destination, assuming no adversarial interference, *i.e.*, the max flow. It can also be expressed as *the min-cut from source to destination*.
- The *broadcast capacity* of a node is the capacity of the hyperedge corresponding to that node's transmission.
- The *secure capacity* of the network is the maximum achievable rate at which information transmission is possible from the source to the destination such that any single node Byzantine attack, which is defined as an adversary injecting errors on its outgoing link, can be detected at the destination node.

[1] We, however, remark that our results can be easily generalized to cases when a node in the network has both broadcast and unicast outgoing links.

4. FORMULATING THE CONSTRAINTS

We start with some definitions that will allow us to succinctly describe our formulation for the secure capacity of the network.

DEFINITION 1 (ORDERING OF NODES). *A partial ordering over the set of nodes is an ordering $<$ between every pair of nodes u, v such that $u < v$ if and only if min-cut$(\mathcal{G}, z, u, v) > 0$.*

In other words, an ordering over the set of nodes defines for every node u, the set of ancestors and descendants of that node in the flow. Recall that our network is a directed acyclic hypergraph. This allows us to define a specific ordering over the set of nodes using *topological sort* [2]. Given such an ordering, we will from here on refer to the set of nodes as $\mathcal{V} = \{1, 2, \ldots, N\}$, where 1 is the source of the network and N is the destination of the network.

DEFINITION 2 (DUPLICATION). *We say that a packet p is duplicated by some node in the network if and only if the destination receives multiple copies of the packet p.*

Before defining the *source of duplication* for a particular packet, we give a network transformation which helps us in formulating the optimization program:

DEFINITION 3 (NETWORK TRANSFORM). *Given a network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, we construct a "transformed network" $\mathcal{G}^T = (\mathcal{V}, \mathcal{E}^T)$ as follows: for each hyperedge $e = (u\ N(u)) \in \mathcal{E}$, we create $|N(u)|$ edges $(u\ v),\ v \in N(u)$ such that there is a single edge between u and every $v \in N(u)$ and the capacity of all these $|N(u)|$ edges is c_e.*

Note that for equivalence of the original network and the transformed network, we need to impose some restrictions on the min-cut between every pair of nodes in the transformed network. Indeed, the only restriction required is the total number of *distinct* packets that can be transferred from u to $N(u)$. We will, in the optimization program, impose the restriction that the total number of distinct packets that can be transferred from u to $N(u)$ is no more than the broadcast capacity of node u. We leave it for the interested readers to see that this restriction results in the equivalence of the original network and the transformed network. We will simply refer to the transformed network as the original network from now on. For any node v, we define $In(v)$ and $Out(v)$ to be the set of incoming and outgoing edges of node v.

DEFINITION 4 (SOURCE OF A DUPLICATE PACKET). *We say that a packet p is duplicated at node with index u if u is the node with least index such that at least two nodes in $N(u)$ forward the packet p.*

Before formulating the optimization problem, we first point out some properties of some optimal duplicate and forward algorithm. Note that it is not necessary that all optimal algorithms should have the following properties. Instead, we argue that there must exist some optimal algorithms that have these properties. For any optimal duplicate-and-forward algorithm that any of the properties not hold, we can derive a optimal algorithm that all these properties hold.

- Property 1: All packets and all duplicated copies of a packet should be routed to the destination. Otherwise nodes can just not forward the copies that will not arrive at N and makes no difference at node N.
- Property 2: At most two copies of each packet are needed to be transmitted.
  Suppose there is an optimal solution in which more than two copies of packet p are transmitted. If between the first node that duplicates p, say u, and the destination N, there is no a single node that forms a cut for the paths of copies of p, then there must be two node-disjoint paths between u and N that carry two copies of p. Since at least two copies of p travel through two node-disjoint path, no one single node can tamper the copies it carries without being detected. So we can keep these two node-disjoint paths and remove all other paths and it is still optimal. If there is some nodes that each is a cut between u and N, then between each consecutive pair of such nodes, including u and N, there is two node-disjoint paths. Following a similar argument as before, we only need to keep two node-disjoint paths between each consecutive pair and remain optimal.

As discussed in Example of Fig. 1, the broadcast nature of the source allows the neighbors of the source to perform coding on the packets received from the source. To make the following discussion more coherent, we will assume that the broadcast capacity of the source is large enough so that the source can transmit these "coded" and "uncoded" packets itself and the neighbors of the source do not need to perform any coding. Under such an assumption, we will now formulate a linear program that characterizes the capacity of a given network for the cases when the intermediate nodes in the network may either forward the packets or "duplicate" the packets.

Finally, we give the following notation used in the rest of the paper:

- $w_{u,e}$: number of packets with u as the duplication source and routed through edge e.
- z_e: number of packets routed through edge e that have not been duplicated at any node $u \leq$ tail(e).
- $w_u^{in}(v) = \sum_{e:e\in In(v)} w_{u,e}$: number of incoming packets at node v that have been duplicated at node u.
- $w_u^{out}(v) = \sum_{e:e\in Out(v)} w_{u,e}$: number of outgoing packets from node v that have been duplicated at node u.
- $z^{in}(v) = \sum_{e:e\in In(v)} z_e$: number of incoming packets at node v that have been not been duplicated at any node $u <$ tail(e).
- $z^{out}(v) = \sum_{e:e\in Out(v)} z_e$: number of outgoing packets from node v that have been not been duplicated at any node $u \leq$ tail(e).

Let $C_{sec}(z, w)$ be the secure capacity of the network, given a flow assignment $\langle z\rangle$ and $\langle w\rangle$ (we give the expression for $C_{sec}(z, w)$ in the following section). Given the above notation and definitions, consider the following optimization program:

$$\max_{z,w} C_{sec}(z, w)$$

subject to

$$w_u^{in}(v) = 0 \qquad \forall u \not< v$$

$$w_u^{out}(u) = 2(z^{in}(u) - z^{out}(u)) \qquad \forall u \neq 1, N$$

$$w_u^{out}(v) = w_u^{in}(v) \qquad \forall v \neq 1, u, N$$

$$z_e + \sum_u w_{u,e} \leq c_e \qquad \forall e \in \mathcal{E}$$

$$z^{out}(u) + \sum_{u\neq v} w_u^{out}(v) + w_u^{out}(u)/2 \leq c_{\text{out(u)}} \qquad \forall u \in \mathcal{V}$$

Let us consider each of the constraint individually:

1. The first constraint captures the acyclic nature of our network model. In particular, only descendants of node u can can forward packets being duplicated at u.
2. The second constraint restricts two possibilities: (1) a packet that has been duplicated by a node before u can not be duplicated again at u, and, (2) no more than two copies of any packet are created at node u (in other words, no more than two neighbors of node u forward the same packet).
3. The third constraint is the flow conservation for the duplicated packets. In particular, we require each node to forward all the duplicated packets that it receives (while also allowing it to be the source of duplication for other packets).
4. The fourth constraint is due to the capacity constraint of the outgoing links for each node. It restricts the *total amount of traffic* flowing on any of the edges to be no more than the broadcast capacity of that node.

5. The last constraint captures the equivalence of the network transform. In particular, it restricts the *total number of distinct packets* flowing through a node to be no more than the broadcast capacity of that node. To count the number of distinct packets flowing through a node, we count all the non-duplicated flows through the node ($z^{out}(u)$), the total number of packets duplicated at other nodes ($\sum_{v\neq u} w_v^{out}(u)$) which u is bound to forward due to the third constraint and the total number of packets duplicated at u, half of which are distinct. Notice that the last constraint is not needed in *wired* networks where all outgoing links from a node are unicast links, and the first four constraints suffice to capture the secure capacity for wired networks.

We remark that these constraints, combined with the following discussion on the number of packets a single adversary in the network may attack and the capacity formulation, capture all the possible routing strategies that include the intermediate nodes in the network forwarding and/or duplicating the packets in the network. It is worth mentioning that not all optimal strategies may satisfy the above constraints; rather we argue that among all the strategies that achieve the optimal throughput, there must be at least one strategy that satisfies the above constraints.

5. CAPACITY CHARACTERIZATION

Given the above constraints, we now discuss the number of packets a single adversary in the network may be able to attack without being detected by duplication . Note that the above formulation allows a node to forward multiple copies of the same packet p (p might be duplicated at some node before this node). Hence, we need to count the number of *distinct* packets that the adversary can attack. To do this, we identify two specific attack possibilities for any node v:

- For any node $u < v$, $w_u^{out}(v) \leq w_u^{out}(u)/2$: there *may* exist a routing scheme such that all $w_u^{in}(v) = w_u^{out}(v)$ packets are unique. In such a case, the adversary can not attack any of these $w_u^{in}(v)$ packets without being detected by the destination.

- For any node $u < v$, $w_u^{out}(v) > w_u^{out}(u)/2$: for any routing scheme, at least $w_u^{out}(v) - w_u^{out}(u)/2$ pairs of packets must be in duplicate. In such a case, the adversary can attack any of these $w_u^{out}(v) - w_u^{out}(u)/2$ packets without being detected by the destination (it can attack both, the original packet and its duplicate and will go undetected at the destination).

- Finally, the adversary at node v can always attack the packets that have not been duplicated by any node before v.

To capture the first two attack scenarios, we define:

$$d_u(v) = \max\{0, w_u^{out}(v) - w_u^{out}(u)/2\} \quad (1)$$

as a lower bound of the number of duplicated packets flowing through v that the adversary at v can attack without being detected by the duplication. The above discussion then leads to the following lower bound on the total number of packets the adversary at any node u may be able to attack:

$$z^{in}(v) + \sum_{u<v} d_u(v) \quad (2)$$

Since the adversary attacks after a routing strategy has been decided, it has the freedom to attack on any node in the network. Hence, a lower bound on the adversary's attack capability is given by:

$$C_{adv}(z, w) = \max_{v\neq 1,N} \left\{ z^{in}(v) + \sum_{u<v} d_u(v) \right\}. \quad (3)$$

Finally, the number of *distinct* packets received by the destination under the flow constraints is clearly:

$$z^{in}(N) + \sum_u w_u^{in}(N)/2 \quad (4)$$

This leads to the following *upper* bound on the capacity of the network:

$$C_{sec}(z, w) \leq z^{in}(N) + \sum_u w_u^{in}(N)/2 - C_{adv}(z, w) \quad (5)$$

5.1 Achieving the upper bound

In this section, we will prove that the upper bound is actually achievable.

To show that the upper bound is achievable, we need to show that for packets duplicated at any node u, there exists a routing strategy such that exactly $d_u(v)$ pairs of identical copies are routed through every v in the network.

For any rational number assignment of $w_{u,e}$ that satisfies the constraints, consider the subgraph between i and N: $\mathcal{G}'(\mathcal{V}, \mathcal{E}, w_u)$, whose link capacity is determined by $w_{u,e}$. Since all $w_{u,e}$ are rational, we can find a number $\triangle$ such that all $w_{u,e}$ and $w_u^{out}(u)/2$ are integer multiples of $\triangle$, and let $\triangle$ be the packet size, or the unit of the weights.

First consider the case when $w_u^{out}(v) \leq w_u^{out}(u)/2$ for all v. The following algorithm finds $w_u^{out}(u)/2$ pair of node-disjoint paths from u to N so that no two copies of any duplicated packet are routed through the same node.

- Step 1: If $\max_{v\in\mathcal{V}} w_u^{out}(v) < w_u^{out}(u)/2$, find a pair of node-disjoint paths from u to N. Go to Step 3.

- Step 2: If $\max_{v\in\mathcal{V}} w_u^{out}(v) = w_u^{out}(u)/2$, find a pair of node-disjoint paths such that all nodes with outgoing weight , $w_u^{out}(v)$, equal to $w_u^{out}(u)/2$ are on either paths. Go to Step 3.

- Step 3: Route the copies of a packet through the node-disjoint paths found in Step 1 or 2. Reduce the weights of the links , $w_{u,e}$, belong to either paths by 1.

- Step 4: Repeat Step 1 to 3 until all weights , $w_{u,e}$, become 0.

LEMMA 1. *Two node-disjoint paths can always be found in Step 1.*

PROOF. Since $\max_{v\in\mathcal{V}} w_u^{out}(v) < w_u^{out}(u)/2$, no two nodes can forward all the $w_u^{out}(u)$ copies of packets. So there are always two node-disjoint paths from u to N. □

LEMMA 2. *Such node-disjoint paths can always be found in Step 2.*

PROOF. To start with, we first split the "source" of the duplicated flow u into two virtual nodes u_1 and u_2 that are not connected with each other and set $w_u(u_1, v) = w_u(u_2, v) = w_u(u, v)/2$ for every node v that is a child of u. The destination N is split into two virtual nodes N_1 and N_2 in the same way. It is obvious that if we can connect (u_1, u_2) to (N_1, N_2) through two node-disjoint paths, so we can for u and N.

Let $M = \{v \in V : w_u^{out}(v) = w_u^{out}(u)/2 \text{ or } w_u^{in}(v) = w_u^{out}(u)/2\}$ be the set of nodes with weight outgoing / incoming flow $w_u^{out}(u)/2$. Obviously, the four virtual nodes we just created belong to M. We will call the nodes in M the *big* nodes for convenience. It is also easy to see that if two big nodes are not connected, then every duplicated copy from u to N must pass through one of these two nodes. We will call two nodes that are not connected to each other in $\mathcal{G}'$ to be *parallel*. So every pair of parallel big nodes form a cut of $\mathcal{G}'$ between u and N.

Let (a, b) and (c, d) be two pairs of parallel big nodes. We will say ordering "up to permutation" – (b,a) < (d,c) and (a,b) < (c,d) if $a = c,\ b < d$ or $a < c,\ b < d$. Also define $(a, b) = (b, a)$. Under such ordering, (u_1, u_2) is the first pair of parallel nodes and (N_1, N_2) is the last pair. We will show that two node-disjoint paths from (u_1, u_2) to (N_1, N_2) that travel through all nodes in M always exist by show such paths exist between every two consecutive pair of such parallel nodes.

Consider any two *consecutive* pairs (s_1, s_2) and (t_1, t_2) such that there does not exist another pair $(s_1, s_2) < (s, t) < (t_1, t_2)$. Let $X \subseteq M$ be the subset of big nodes that are descendants of s_1 or s_2 and are ancestors of t_1 or t_2 (if there is any). Nodes in X cannot be parallel with each other, so there is a paths that goes through all nodes in X and it forms a total ordering along the paths. So we can order them as $x_1, x_2, ..., x_K$, where $K = |X|$. Moreover, nodes in X cannot be parallel with any one of s_1, s_2, t_1, and t_2 by definition. Then there must be two disjoint paths from s_1 and s_2 to x_1. Similarly, there must be two disjoint paths from x_K to t_1 and t_2.

Case 1: $s_1 = t_1$. In this case, $X = \phi$. Since s_2 and t_2 are both parallel with s_1, t_2 must be a descendant of s_2 and s_1 cannot be on any path from s_2 to t_2. So s_1 itself and any path $s_2 - t_2$ consist the desired disjoint paths.

Case 2: $s_1 < t_1,\ s_2 < t_2$ and X is empty. There are two node disjoint paths between (s_1, s_2) and (t_1, t_2), use them.

Case 3: $s_1 < t_1,\ s_2 < t_2$ and the total flow to / from X from / to (s_1, s_2) and (t_1, t_2) is $w_u^{out}(u)/2$, i.e., min-cut$\{\mathcal{G}', w_u, (s_1, s_2), X\}$ = min-cut$\{\mathcal{G}', w_u, X, (t_1, t_2)\} = w_u^{out}(u)/2$. In Appendix A, we show the existence of two node-disjoint paths from (s_1, s_2) to (t_1, t_2), one of which covers all nodes in X.

Case 4: $s_1 < t_1,\ s_2 < t_2$ and the total flow to / from X from / to (s_1, s_2) and (t_1, t_2) is greater than $w_u^{out}(u)/2$, i.e., min-cut$\{\mathcal{G}', w_u, (s_1, s_2), X\}$ = min-cut$\{\mathcal{G}', w_u, X, (t_1, t_2)\} > w_u^{out}(u)/2$.

In this case, X is non-empty, and it must contain at least two nodes, since the cut from X exceeds $w_u^{out}(u)/2$, but each node in X has outgoing flow only $w_u^{out}(u)/2$. Thus $x_1 \neq x_K$. Moreover, there must exist a $x_n, n > 1$ that has a path from s_1 or s_2 that does not cover any other big nodes. Otherwise, the incoming paths to every node in X must be all from x_1. Then the total incoming flow to X is just the incoming flow to x_1, i.e., min-cut$\{\mathcal{G}', w_u, (s_1, s_2), X)\}$ = min-cut$\{\mathcal{G}', w_u, (s_1, s_2), x_1)\} = w_u^{out}(u)/2$, which contradicts the case definition.

Let's call a path between node v and v' a *direct path* if it does not cover any big node except for v and v'. Let $x_n, n > 1$ be the node that has the smallest index among nodes in $X \backslash x_1$ that has a direct path from s_1 or s_2. Then there must exist a direct path to x_n, DP_n that is node disjoint with a path from s_1 or s_2 that goes through $x_1, \ldots, x_{n-1}$. Without loss of generality, suppose that DP_n originates at s_1. Then DP_n must be node disjoint with some path from s_2 to x_{n-1} that goes through nodes $x_1, ..., x_{n-1}$. We will argue the correctness of this observation by contradiction. Consider a path P_{n-1} from s_2 to x_{n-1} that goes through $x_1, \ldots, x_{n-1}$ and intersects with DP_n. If they intersect at a node x_m or between x_{m-1} and $x_m, m \leq n-1$, then node x_m has a direct path from s_1 along the segment of DP_n before the intersection and the segment of P_{n-1} after the intersection, which leads to contradiction, since x_m has a direct path from s_1, but $m < n$. If DP_n and P_{n-1} only intersect in the segment before x_1, let D_1 and D_2 denote two disjoint paths from s_1 and s_2 to x_1. DP_n must intersect with at least one of D_1 and D_2. If the last intersection is on D_1: new DP_n is formed as the union of the segment of D_1 before the last intersection and the segment of DP_n after. Also for P_{n-1} we now choose D_2 as the first segment from s_2 to x_1. We can see the new DP_n is disjoint with the new P_{n-1} and Fig.4 (a) shows an example. If the last intersection is on D_2, we can construct the new DP_n from s_2 to x_n, and new P_{n-1} from s_1 in a similar way.

Now extend DP_n so that it goes through $x_{n+1}, \ldots, x_K$ (if $n = K$, no extension needed). Then every node of X is either on DP_n or P_{n-1}. So there exists at least one pair of two disjoint paths from (s_1, s_2) that terminate at x_K and one of its ancestors in X (in particular, P_{n-1} terminates at x_{n-1}) and cover X. Let's denote (P_1, P_2) as a pair of such disjoint paths from (s_1, s_2) that terminate at x_K and one of its ancestors in X, and cover X. Without loss of generality, assume P_1 terminates at x_K and denote Π as the set of all such (P_1, P_2). Let

$$x_l = \max_{(P_1,P_2)\in\Pi} \{\arg\max_v (v \textit{ covered by } P_2, v \in X)\} \quad (6)$$

be the node in X with the largest index that is covered by some P_2. And let $(P_1^*, P_2^*) \in \Pi$ be a solution in which P_2^* covers x_l. Thus, P_1^* includes $x_{l+1}, ..., x_K$ (and possibly some other nodes in X as well).

Claim 1: There must be two disjoint paths from x_l, say P_l and

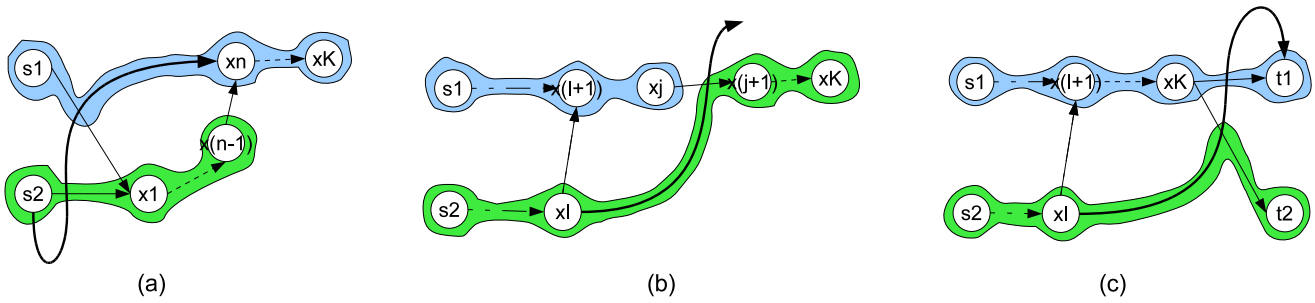

Figure 4: (a) The thick arrow indicates a direct path to x_n that intersects with D_1 and D_2. The lightblue and green area indicate P_1 and P_2 respectively. (b) The thick arrow indicats an outgoing path from x_l that intersects with a path between x_j and x_{j+1}. The green and lightblue area indicate P_1' and P_2' respectively. (c) The thick arrow indicates an direct path from x_l that intersects with two disjoint paths from x_K to t_1 and t_2. The colored areas indicate two disjoint paths that cover all nodes of X.

$P_{l,l+1}$, such that P_l is a direct path to a node in $\{x_{l+2}, \ldots, x_K, t_1, t_2\}$ and $P_{l,l+1}$ is a direct path to x_{l+1}.

PROOF. See Appendix B. □

Claim 2: P_l cannot intersect with P_1^* at descendants of x_{l+1}.

PROOF. If $l+1 = K$, by definition, there is no intersection after $x_{l+1} = x_K$, since P_1^* terminates at x_K. Now let's consider the case when $l+1 < K$. Suppose in contradiction that P_l intersects with P_1^* at some descendants of x_{l+1}. Let's denote their first intersection after x_{l+1} is node v. And assume that, for some $m > l+1$, v is either same as node x_m, or a node between x_{m-1} and x_m (excluding x_{m-1}).

First notice that P_l can only intersect with P_1^* at descendants of x_h, the last bing node on P_1^* before x_{l+1} (x_h may be s_1/s_2 if x_{l+1} has a direct path from one of them). Otherwise, if P_l intersects with P_1^* at x_h or its ancestors, there is a path from the intersection to x_h, hence to x_l, and then through P_l back to the intersection, contradicting with the acyclic assumption of the graph.

Let x_j be the node in X on P_1^* with the smallest index at or after the first intersection with P_l. Notice that j must be at least $l+1$.

If the first intersection of P_l and P_1^* is after x_{l+1}, i.e., $j = m > l+1$, then connect the segments of P_l upto v and the segment of P_1^* after v, and then append it onto P_2^* to create a new path P_1'. And create P_2' by truncating P_1^* at node x_{j-1}. Easy to see $(P_1', P_2') \in \Pi$. Then x_j is covered by P_2' and its index is larger than x_l, which contradicts with the definition of x_l as it is illustrated in Fig.4 (b).

If the first intersection is in the segment between x_h and x_{l+1}, i.e., $j = l+1 < m$, there are two cases:

(1) P_1^* intersects with P_l at some node y before it intersects with $P_{l,l+1}$.

(2) P_1^* intersects with $P_{l,l+1}$ at some node y before it intersects with P_l.

We prove by contradiction that P_l cannot intersect with P_1^* between x_h and x_{l+1} in both cases in Appendix C. □

Claim 2 shows that P_l cannot not intersect with P_1^* after x_{l+1}, hence P_l does not cover any one of $\{x_{l+2}, ..., x_K\}$. And since P_l is a path to $\{x_{l+2}, ..., x_K, t_1, t_2\}$, it must be a direct path from x_l to $\{t_1, t_2\}$ that does not intersect with P_1^* after x_{l+1}.

If P_l does not intersect with P_1^* before x_{l+1} either (P_l cannot intersect AT x_{l+1} by definition of direct path – P_l is a direct path to $\{x_{l+2}...\}$), then similar to the proof of existence of x_n, there must be two disjoint direct paths from x_K to t_1 and t_2, say D_1 and D_2. Without loss of generality, suppose that P_l terminates at node t_1. If P_l does not intersect with D_1 and D_2, then $P_1^* - D_2$ and $P_2^* - P_l$ are two disjoint paths from (s_1, s_2) to (t_1, t_2) covers X. If P_l intersects with at least one of D_1 and D_2. If the first intersection is on D_2: new P_l is formed as the union of the segment of P_l before the first intersection and the segment of D_2 after. The new P_l terminates at t_2 and is disjoint with D_1. Now we have two node disjoint paths from s_1 and s_2 to t_1 and t_2 that cover all nodes of X. An example is shown in Fig.4 (c). If the first intersection is on D_1, we can construct the new P_l from x_l to t_1 and disjoint with D_2 in a similar way.

If P_l intersects with with P_1^* before x_{l+1}, create P_1'' by appending $P_{l,l+1}$ and P_1^*'s segment after x_{l+1} to P_2^*. And create P_2'' by truncating P_1^* upto the node x_h, the last big node on P_1^* before x_{l+1}. Notice that the union of the segment of P_1^* between x_h and its first intersection with P_l, and the segment of P_l after the same intersection creates a direct path from x_h, the last node on P_2'', to $\{t_1, t_2\}$ that does not intersect with P_1''. Then we can treat x_h as x_l and follow the same process in the previous paragraph, we can find two node disjoint paths from (s_1, s_2) to (t_1, t_2) that cover all nodes of X.

To summarize, in any case, we can find two node-disjoint paths connecting every two pairs of consecutive parallel big nodes that cover all big nodes between them. Then we can start with (u_1, u_2) and find two such disjoint paths to one pair of the immediate parallel big nodes, and repeat this until it reaches (N_1, N_2). Then we have two node disjoint paths from u to N that cover all big nodes. □

LEMMA 3. *$w_u^{out}(v) \leq w_u^{out}(v)/2$ for all v after Step 3.*

PROOF. Denote $w_{max} = \max_v\{w_u^{out}(v)\}$.

If $w_{max} < w_u^{out}(u)/2$, then $w_{max} \leq w_u^-(u)/2 - 1$. By Lemma 1, two node-disjoint paths can be found in Step 1. After updating the weights, w_{max} will either decreased by one or remain the unchanged, while $w_u^{out}(u)$ is reduced by 2. So $w_{max}^{new} \leq w_{max} \leq w_u^{out}(u)/2 - 1 = w_u^{out,new}(u)/2$.

If $w_{max} = w_u^{out}(u)/2$, the paths are found in Step 2. According to Lemma 2, all nodes with w_{max} belong to either one of the paths, their weights are all reduced by 1. And since weight of any other node is at most $w_{max} - 1$, so $w_{max}^{new} = w_{max} - 1 = w_u^{out}(u)/2 - 1 = w_u^{out,new}(u)/2$. □

From Lemma 3, we can conclude that the whole process can be repeated until all weights , $w_{u,e}$, become zero. And then we have $w_u^{out}(u)/2$ pairs of disjoint paths from u to N.

Now consider the case when $w_u^{out}(v) > w_u^{out}(u)/2$ for some $v \neq u$. In this case, we first split every such node v into two disjoint nodes v_1 and v_2 such that the weights of links to / from them are in same proportion as the original links to / from v and the total incoming / outgoing flow to / from v_1 is $w_u^{out}(u)/2$. Then we apply the previous algorithm on this extended graph. In each round, v_1 must be on one of the paths while v_2 may or may not be on the other path. If both v_1 and v_2 are one the two paths, it means both copies of a packet is routed through v, otherwise only one copy is routed through v. Since there are $w_u^{out}(v_2) = w_u^{out}(v) - w_u^{out}(u)/2 = d_u(v)$ packets through v_2, exactly $d_u(v)$ pairs of copies are routed through v.

Now we have shown that the solution to the optimization problem indeed achieves the secure capacity of the network. For example, consider the network in Fig. 2. A solution to the optimization problem for this network is $z_{(s,r_1)} = z_{(s,r_2)} = z_{(s,r_3)} = 1$, $w_{r_1,(r_1,r_4)} = w_{r_1,(r_1,r_5)} = 1$, $z_{(r_2,r_4)} = z_{(r_3,r_5)} = 1$, $z_{(r_4,d)} = w_{r_1,(r_4,d)} = z_{(r_5,d)} = w_{r_1,(r_5,d)} = 1$, and all other z's and w's are zero. So the number of distinct packets the destination receives is $z_{(r_4,d)} + z_{(r_5,d)} + (w_{r_1,(r_4,d)} + w_{r_1,(r_5,d)})/2 = 3$, and the adversary's capability is 1 for every intermediate node. So the secure capacity is $3 - 1 = 2$.

6. POWER OF PROMISCUOUS MONITORING

Recall that under a broadcast model, intermediate nodes may need to perform linear operations over the incoming packets to achieve the capacity of the network. In this section, we demonstrate a stronger requirement: even linear operations do not suffice to achieve the capacity [6], [7]. To show this, we use the idea of promiscuous monitoring. Promiscuous monitoring means that if a node A is within the transmission range of a node B, it can overhear the communication to and from B even if those communications do not directly involve A. Promiscuous monitoring allows A to monitor the behavior of B, thereby reducing the possibility of attack at node B (assuming that the channels are symmetric, B could monitor A too). It is easy to see that promiscuous monitoring of B by A can be seen as a non-linear operation.

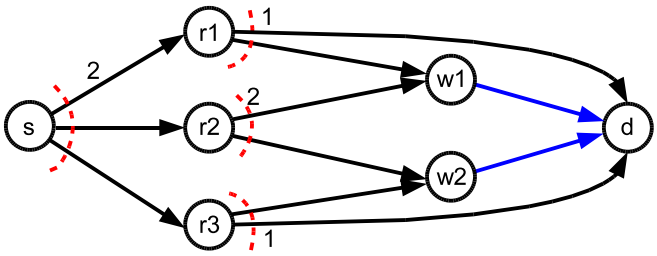

Figure 5: A single source (s), single destination (d) network with three unreliable nodes r_1, r_2 and r_3. s has a broadcast capacity of 2 units, r_1 and r_3 have a broadcast capacity of 1 unit and r_2 has a broadcast capacity of 2 units. The capacity of links from w_1 and w_2 is asymptotically negligible. The secure capacity of the network with promiscuous monitoring is twice the secure capacity of the network without promiscuous monitoring.

Consider the network shown in Fig. 5. The secure capacity of the network without promiscuous monitoring is 1 unit by duplicating the packets at r_1 and r_3. However, the secure capacity of the network is 2 units. To see this, consider the following transmission strategy: r_1 and r_3 broadcast packets a and b respectively. r_2 broadcasts packets a and b. With such a strategy, w_1 can compare the packet from r_1 to one of the packets from r_2 and w_2 can do the same for the packet from r_3. If either of w_1 or w_2 notice that the packet has been modified, they can inform the destination using an asymptotically negligible rate channel.

Next, we show that the ratio of secure capacity with and without promiscuous monitoring can be arbitrarily large [6], [7].

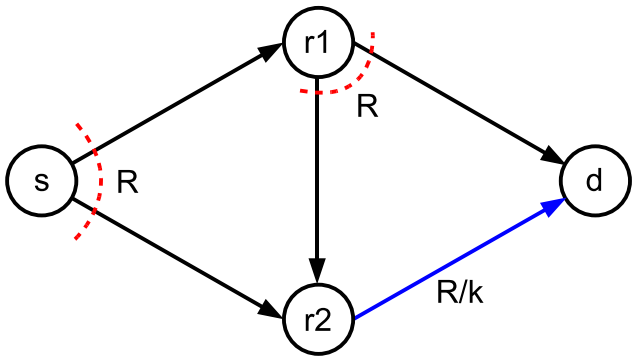

Figure 6: A single source (s), single destination (d) network with two unreliable nodes r_1 and r_2. s has a broadcast capacity of R units, r_1 has a broadcast capacity of R unit and r_2 has a broadcast capacity of R/k packets per unit time to d. All links are reliable. In this network, nodes simply forwarding the received packets or mixing the received packets does not achieve capacity of the network.

Consider the network shown in Fig. 6. The network has edge

capacities as described in the figure (assume that the size of the packets is large enough). It is easy to show that if the relay nodes are restricted to forwarding the received information packets, s can transmit at no more than R/k packets per unit time while detecting any Byzantine modification in the network. It is also easy to show that mixing of packets does not increase the number of packets s can transmit per unit time. However, if a comparison operation (which can be formally described as a non-linear operation over the received packets) is allowed at node r_2, the network can achieve its capacity of $R(1-1/k)$ packet per unit time. In this case, r_2 can compare each packet transmitted by s to each packet forwarded by r_1 and can inform d if any of these packets do not match. On the other hand, if r_1 is under attack s can create a $(R, R(1-1/k), R/k)$ error detecting code so that any modification at r_2 can be detected. Since the adversary can attack at any of the nodes, the secure capacity with promiscuous monitoring is $R(1-1/k)$. The ratio of secure capacity with and without promiscuous monitoring is then given by $R(1-1/k)/R/k = (k-1)$, which can be arbitrarily large depending on k.

7. CONCLUSIONS

In this paper, we have characterized the trade-off between security and throughput in wireless broadcast networks. In particular, we have shown that the problem of characterizing the secure capacity of any wireless broadcast network can be formulated as a linear optimization problem. This, combined with a carefully designed routing strategy that achieves the bound given by the optimal solution for the optimization program, establishes the secure capacity of any given wireless broadcast network. The main observation used in designing a capacity achieving routing strategy is that some nodes in the network may duplicate the packet in order to check the correctness of the received packets at the destination.

Recall the ideas from Example of Fig. 1. In the example, we showed that the neighbors of the source doing coding increases the capacity of the network. This brings up the question whether the capacity can further be increased by nodes other than the neighbors of the source doing coding. We believe that such is not the case. In particular, we believe that forwarding and duplication at all intermediate nodes in the network except for neighbors of the source achieves the secure capacity of the network. Hence, we conjecture the following:

CONJECTURE 1. *The secure capacity of the network is achieved with all nodes in $\mathcal{V}\backslash N(s)$ just forwarding and duplicating the information packets. Network coding at any node other than $N(s)$ does not increase the capacity of the network.*

An interesting observation we make towards the end of the paper is that the capacity of wireless broadcast networks can be increased if the intermediate nodes in the network can perform monitoring. In fact, we show that there exist networks for which the secure capacity of the network with in-network monitoring is arbitrarily larger than that without in-network monitoring. Our current work is in the direction of developing a theoretical framework is required to quantify the benefits of in-network monitoring.

8. REFERENCES

[1] V. Bhandari and N. H. Vaidya. On reliable broadcast in a radio network. In *In ACM Symposium on Principles of Distributed Computing (PODC'05)*, 2005.

[2] T. Cormen, C. Leiserson, and R. Rivest. *Introduction to Algorithms*. MIT Press, 1997.

[3] J. Dong, R. Curtmola, and C. Nita-Rotaru. Practical defenses against pollution attacks in intra-flow network coding for wireless mesh networks. In *WiSec '09: Proceedings of the second ACM conference on Wireless network security*, pages 111–122, New York, NY, USA, 2009. ACM.

[4] T. Ghosh, N. Pissinou, and K. Makki. Towards designing a trusted routing solution in mobile ad hoc networks. *Mob. Netw. Appl.*, 10(6):985–995, 2005.

[5] C. Gkantsidis and P. Rodriguez Rodriguez. Cooperative security for network coding file distribution. *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1–13, April 2006.

[6] L. Guanfeng and V. Nitin. When watchdog meets coding. *Technical Report, CSL, UIUC*, May 2009.

[7] L. Guanfeng, A. Rachit, and V. Nitin. When watchdog meets coding ii. *Technical Report, CSL, UIUC*, September 2009.

[8] T. Ho, B. Leong, R. Koetter, M. Medard, M. Effros, and D. Karger. Byzantine modification detection in multicast networks using randomized network coding, 2004.

[9] S. Jaggi, M. Langberg, S. Katti, T. Ho, D. Katabi, and M. Medard. Resilient network coding in the presence of byzantine adversaries. *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 616–624, May 2007.

[10] D. C. Kamal, D. Charles, K. Jain, and K. Lauter. Signatures for network coding. In *In Proceedings of the fortieth annual Conference on Information Sciences and Systems*, 2006.

[11] C. Y. Koo. Broadcast in radio networks tolerating Byzantine adversarial behavior. In *In ACM Symposium on Principles of Distributed Computing (PODC'04)*, 2004.

[12] C. Y. Koo, V. Bhandari, J. Katz, and N. H. Vaidya. Reliable broadcast in radio networks: The bounded collision case. In *In ACM Symposium on Principles of Distributed Computing (PODC'06)*, 2006.

[13] M. N. Krohn. On-the-fly verification of rateless erasure codes for efficient content distribution. In *In Proceedings of the IEEE Symposium on Security and Privacy*, pages 226–240, 2004.

[14] Q. Li, D.-M. Chiu, and J. Lui. On the practical and security issues of batch content distribution via network coding. *Network Protocols, 2006. ICNP '06. Proceedings of the 2006 14th IEEE International Conference on*, pages 158–167, Nov. 2006.

[15] S. Marti, T. J. Giuli, K. Lai, and M. Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 255–265, New York, NY, USA, 2000. ACM.

[16] A. Pelc and D. Peleg. Broadcasting with locally bounded byzantine faults. In *Information Processing Letters*, 2005.

[17] A. Pelc and D. Peleg. Feasibility and complexity of broadcasting with random transmission failures. In *In ACM Symposium on Principles of Distributed Computing (PODC'05)*, 2005.

[18] Z. Yu, Y. Wei, B. Ramkumar, and Y. Guan. An efficient signature-based scheme for securing network coding against pollution attacks. *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pages 1409–1417, April 2008.

[19] F. Zhao, T. Kalker, M. Medard, and K. J. Han. Signatures for content distribution with network coding. In *In Proc. of International Symposium on Information Theory (ISIT*, 2007.

APPENDIX

A. PROOF OF CASE 3 IN LEMMA 2

$s_1 < t_1$, $s_2 < t_2$ and the total flow to/from X from/to (s_1, s_2) and (t_1, t_2) is $w_u^{out}(u)/2$, i.e., min-cut$\{\mathcal{G}', w_u, (s_1, s_2), X\}$ = min-cut$\{\mathcal{G}', w_u, X, (t_1, t_2)\} = w_u^{out}(u)/2$.

In this case, all the outgoing paths from x_n must pass x_{n+1}, for $n = 1, ..., K-1$. Similarly, all the incoming paths to x_n must pass x_{n-1} for $n = 2, ..., K$. And since only half of the packets go though X, there must be a path P from (s_1, s_2) to (t_1, t_2) that does not intersect with any path P_X that passes through X from x_1 to x_K. P may intersect with some paths from (s_1, s_2) to x_1 and some paths from x_K to (t_1, t_2) though In this case, consider two disjoint paths D_1 and D_2 from s_1 and s_2 to x_1 and two disjoint paths D_3 and D_4 from x_K to t_1 and t_2. Without loss of generality, assume P is from s_1 to t_1. If P does not intersect with any one of D_1, D_2, D_3 and D_4, then P is disjoint with $D_2 - P_X - D_4$ and they form the disjoint paths we are looking for. If P intersects with at least one of D_1 and D_2. Without loss of generality, suppose the last intersection is on D_1, then create the new P as the union of the segment of D_1 before the last intersection and the segment of P after, and it is disjoint with $D_2 - P_X$ since the segment from D_1 is disjoint with D_2 and the segment from P is after the last intersection and cannot intersect with D_2 any more. Similary, if P intersects with at least one of D_3 and D_4, without loss of generality, suppose the first intersection is on D_3. Then form the new P as union of the segment of P before the first intersection and the segment of D_3 after, and it is disjoint with D_4. Now we have two disjoint paths P and $D_2 - P_X - D_4$, and X is covered by $D_2 - P_X - D_4$.

B. PROOF OF CLAIM 1

First notice that x_l must have a direct path that goes into $\{x_{l+2}, \ldots, x_K, t_1, t_2\}$ without covering x_{l+1}. Otherwise all paths from x_l must converge to x_{l+1}. So no matter whether P_1^* intersects with some of these paths or not, x_{l+1} has at least one incoming packet along P_1^* in addition to the $w_u^{out}(u)/2$ from x_l, and then the incoming flow to x_{l+1} is greater than $w_u^{out}(u)/2$, contradicting with the weight assignment of x_{l+1}.

The mincut between x_l and $\{x_{l+1}, \ldots, x_K, t_1, t_2\}$ equals to the outgoing flow of x_l, $w_u^{out}(u)/2$. And since any node between x_l and $\{x_{l+1}, \ldots, x_K, t_1, t_2\}$ has outgoing flow less than $w_u^{out}(u)/2$, there must be two disjoint paths from x_l to $\{x_{l+1}, \ldots, x_K, t_1, t_2\}$, similar to Lemma 1. Note that the two disjoint paths from x_l to $\{x_{l+1}, \ldots, x_K, t_1, t_2\}$ are direct, by definition. Thus, two disjoint direct paths do exist (the paths do not have any intermediate nodes in common,but may have the same destination). To prove claim 1 by contradiction, suppose that there does not exist P_l and $P_{l,l+1}$, such that P_l is a direct path to a node in $\{x_{l+2}, \ldots, x_K, t_1, t_2\}$ and $P_{l,l+1}$ is a direct path to x_{l+1}. Then any two disjoint paths from x_l are either both to x_{l+1} or both to $\{x_{l+2}, \ldots, x_K, t_1, t_2\}$. Consider a pair of such disjoint paths q_1 and q_2 to $\{x_{l+2}, \ldots, x_K, t_1, t_2\}$, then any path $P_{l,l+1}$ from x_l to x_{l+1} must intersect with both, otherwise $P_{l,l+1}$ and the path it does not intersect are the paths we are looking for. Without loss of generality, assume the last intersection is on q_1. Then the union of the segment of q_1 before the last intersection and the segment of $P_{l,l+1}$ after that intersection becomes a path to x_{l+1}. Since the segment from q_1 is disjoint with q_2 by assumption and

the segment from $P_{l,l+1}$ after the last intersection cannot intersect with q_2, then the new path is disjoint with q_2.

It is a similar argument if both disjoint paths are q_1 and q_2 to x_{l+1}. If P_l does not intersect with one of q_1 and q_2, then P_l and the one it does not intersect are the paths we are looking for. If P_l intersects with both, without loss of generality, assume the last intersection is on q_1. Then the union of the segment of q_1 before the last intersection and the segment of P_l after that intersection becomes a path to one of $\{x_{l+2}, ..., x_K, t_1, t_2\}$. Since the segment from q_1 is disjoint with q_2 by assumption and the segment from P_l after the last intersection cannot intersect with q_2, then the new path is disjoint with q_2.

C. PROOF OF LATER PART OF CLAIM 2

(1) P_1^* intersects with P_l at some node y before it intersects with $P_{l,l+1}$.

Then create P_1' by replacing the segment of P_1^* between y and v with the corresponding segment of P_l. Note that by definition of P_1^*, y cannot be same as x_l. We can see that P_1' is disjoint with P_2^* because P_1^* is disjoint with P_2^*, and the segment from P_l is also disjoint with P_2^* since it is an outgoing path from x_l, the last node on P_2^*. Then we extend P_2^* to P_2' by appending $P_{l,l+1}$ and the segment of P_1^* between x_{l+1} and x_{m-1}. $P_{l,l+1}$ is disjoint with P_1' since it is disjoint with P_l and the segment of P_1^* before y. The segment from P_1^* between x_{l+1} and x_{m-1} is disjoint with P_l since it is the segment before v, the first intersection with P_l after x_{l+1}, and hence it is disjoint with P_1'. So P_1' and P_2' are disjoint and they cover X. So $(P_1', P_2') \in \Pi$ and P_2' covers $x_{m-1}, m-1 > l$, and it leads to a contradiction with the definition of x_l.

(2) P_1^* intersects with $P_{l,l+1}$ at some node y before it intersects with P_l. Create P_1' by extending P_2^* by appending the segment of P_l upto v, P_l's first intersection with P_1^* after x_{l+1}, and the segment of P_1^* after v. Easy to see P_1' is disjoint with the segment of P_1^* upto node u. Notice that the segment of $P_{l,l+1}$ between y and x_{l+1} is disjoint with P_1' since it is disjoint with P_l. Also notice that the segment of P_1^* between x_{l+1} and x_{m-1} is also disjoint with P_1' since it is before v, the first intersectiong of P_1^* and P_l after x_{l+1}. Then we can create P_2' by truncating P_1^* at x_{m-1} and replace the segment between y and x_{l+1} with the corresponding segment of $P_{l,l+1}$. So P_1' and P_2' are disjoint and they cover X. So $(P_1', P_2') \in \Pi$ and P_2' covers $x_{m-1}, m-1 > l$, and it leads to a contradiction with the definition of x_l.