# Broadcast CONGEST Algorithms Against Eavesdroppers

**Yael Hitron**
Weizmann Institute of Science, Rehovot, Israel

**Merav Parter**
Weizmann Institute of Science, Rehovot, Israel

**Eylon Yogev**
Bar-Ilan University, Ramat-Gan, Israel

## Abstract

An eavesdropper is a passive adversary that aims at extracting private information on the input and output values of the network's participants, by listening to the traffic exchanged over a subset of edges in the graph. We consider secure congest algorithms for the basic broadcast task, in the presence of eavesdropper (edge) adversaries.

For $D$-diameter $n$-vertex graphs with edge connectivity $\Theta(f)$, we present $f$-secure broadcast algorithms that run in $\widetilde{O}(D + \sqrt{fn})$ rounds. These algorithms transmit some broadcast message $m^*$ to all the vertices in the graph, in a way that is information-theoretically secure against an eavesdropper controlling *any* subset of at most $f$ edges in the graph. While our algorithms are heavily based on network coding (secret sharing), we also show that this is essential. For the basic problem of secure unicast we demonstrate a network coding gap of $\Omega(n)$ rounds.

In the presence of vertex adversaries, known as semi-honest, we introduce the *Forbidden-Set Broadcast* problem: In this problem, the vertices of the graph are partitioned into two sets, trusted and untrusted, denoted as $R, F \subseteq V$, respectively, such that $G[R]$ is connected. It is then desired to exchange a secret message $m^*$ between all the trusted vertices while leaking no information to the untrusted set $F$. Our algorithm works in $\widetilde{O}(D + \sqrt{|R|})$ rounds and its security guarantees hold even when all the untrusted vertices $F$ are controlled by a (centralized) adversary.

**2012 ACM Subject Classification** Networks → Network algorithms; Theory of computation → Distributed algorithms

**Keywords and phrases** congest, edge-connectivity, secret sharing

**Digital Object Identifier** 10.4230/LIPIcs.DISC.2022.27

**Funding** *Merav Parter*: This project is funded by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 949083).

## 1 Introduction

Modern distributed networks are insecure by nature, and consequently, the private information of their users is under a constant threat of leakage to untrusted parties. We consider secure message passing algorithms against eavesdropper adversaries (also known as passive wiretappers) who can eavesdrop on a bounded number of edges in the graph. Our main objective is to provide round-efficient broadcast algorithms that ensure that the eavesdropper obtains no knowledge on the broadcast message, in the information theoretic sense.

The (perfect) secure algorithms in this paper follow the standard congest model [29], where the communication network is abstracted by an $n$-vertex graph $G = (V, E)$ with unique vertex identifiers of $O(\log n)$ bits. The communication proceeds in synchronous rounds, where in each round, a vertex can exchange $O(\log n)$-bit messages with each of its neighbors. The communication occurs in the presence of a computationally unbounded eavesdropper with full information on the graph topology and the protocols executed by


the vertices. It is oblivious, however, to the internal randomness of the vertices. We strive for *information-theoretic* $f$-secure algorithms, which informally guarantee that the messages observed by the eavesdropper controlling at most $f$ edges convey no information on the designated transmitted messages. The design of such *round-efficient* secure algorithms finds a wide range of applications, from digital voting systems to cryptocurrencies and blockchain.

While solutions exist under cryptographic assumptions (e.g., using public-key encryption), our main challenge is in providing the graph-theoretical foundations for distributed algorithms with *information-theoretic* (perfect) security. These solutions are usually easier to compute and have everlasting unconditional security, which is not based on computational assumptions that might be broken in the future. Moreover, information-theoretic security is well fitted to the perspective of the congest model. The latter assumes that the vertices are computationally *unbounded*, and therefore it makes sense to assume that the eavesdropper adversary is computationally *unbounded* as well.

**Perfect-secure algorithms against eavesdroppers.** The notion of perfect security is among the most fundamental and long-studied concepts in cryptography. Within this setting, two main types of passive adversaries have been considered in the literature: semi-honest (vertex adversary) and its edge-analogue, the eavesdropper, which we consider in this paper. Throughout, an algorithm is called $f$-secure if it provides perfect-security against an eavesdropper controlling $f$ edges in the graph.

There has been a long line of work studying secure algorithms in specific graph topologies and under various model assumptions. Most attention has been devoted for the $f$-secure unicast problem where it is required for a source $s$ to send a message $m^*$ to a designated target $t$ over an $(f+1)$ edge-connected graph. One of the earliest works in this area is by Dolev, Dwork, Waarts, and Yung [8]. Their $f$-secure unicast algorithm exchanges the secret message $m^*$ by sending secret-shares of $m^*$ along a collection of $(f+1)$ edge-disjoint $s$-$t$ paths [1]. In a recent work, Hitron and Parter [15] showed that the length of such edge-disjoint paths might be $\min\{n, (D/f)^{\Theta(f)}\}$. This implies that algorithms that are based on exchanging messages along edge-disjoint paths require linear number of rounds in the worst case, already for $f = \Omega(\log n)$.

The dependency in the length of the edge-disjoint paths appears to be unavoidable at first glance. The reason is that resilience against computationally unbounded adversaries calls for establishing graph-theoretical *secure* channels between pairs of vertices. The natural approach for providing such channels is by means of exchanging information along with a collection of sufficiently many edge (or vertex) disjoint paths; The latter guarantees that some of these paths are fault-free, which allows the endpoints to overcome the adversarial effect. Indeed, so-far, all existing secure broadcast algorithms for general graphs follow the above-mentioned scheme (in one way or the other) and consequently required $\min\{\Theta(n), (D/f)^{\Omega(f)}\}$ rounds [23, 27, 24, 15].

Cai and Yeung [4] provided considerably improved solutions for the secure unicast problem that are based on the notion of *secure network coding* for DAG graphs. Informally, this approach provides the throughput advantage of the standard network coding scheme while also providing perfect resilience against eavesdroppers. Feldman, Malkin, Servedio, and Stein [9] generalized and simplified the method of [4] by designing perfect-secure algorithms to

[1] In their model, the source $s$ and the target $t$ are connected by $m$ wires, and the eavesdropper is listening on a bounded number of wires. These wires may correspond to vertex-disjoint paths or edge-disjoint paths.

exchange a message to a restricted subset of vertices for *DAG* networks. Jain [16] presented a secure unicast algorithm, which as shown in this paper can be implemented in $O(D)$ congest rounds for $D$-diameter graphs. A similar algorithm was provided also by Gilboa and Ishai [12]. As secure message transmission algorithms are currently limited to a bounded number of targets and topologies, in this paper we ask the following fundamental question, which for the best of our knowledge, is still vaguely open:

▶ **Question 1.** Is it possible to provide an $f$-secure broadcast algorithm in sub-linear number of congest rounds, even for $f = \Omega(\log n)$?

A naïve solution for this problem can be provided by simply implementing Jain's [16] algorithm for every target $t \in V$, resulting in $\Omega(n)$ rounds. In this paper, we answer the question above in the affirmative by presenting $f$-secure broadcast algorithms with round complexity of $\widetilde{O}(D + \sqrt{fn})$. Our algorithms are based on a particular type of network coding, denoted as secret sharing, which is arguably one of the most foundational concepts in cryptography [31]. We combine these tools with the well-known tree-packing of Nash-Williams [22, 5]. We note that while the area of (perfect) secure message transmission against eavesdroppers is well-established in the cryptography and the networking communities, this setting has been barely addressed before in the context of message-passing models with bandwidth limitations, such as the congest model. The only exception is the work of Parter and Yogev [24] that designed $f$-secure compilers for $f = 1$, that translate any congest (non-secure) $r$-round algorithm into a 1-secure algorithm with $O(rD)$ rounds. Their (implicit) extension to $f$ faults introduces an overhead of $(fD)^{\Theta(f)}$ rounds.

**Network coding gaps in unicast communication.** The task of unicast, in which a source vertex $s$ sends information to a designated target $t$, is one of the most basic and important information dissemination primitives in distributed networks. In typical modern communication networks, it is usually required to run simultaneously many such primitives in parallel. Starting with the influential work of Leighton, Maggs, and Rao [18], the scheduling of multiple unicast tasks has been subject to thorough research over the years, under two main classes of algorithms: (i) *routing-based algorithms* (also known as store-and-forward) where messages are viewed as (atomic) tokens and vertices can only store and forward them, and (ii) *network coding algorithms* where messages are allowed to be mixed together by any form of coding [1]. One of the most intriguing questions in this area is whether coding has a provable advantage over routing-based algorithms. The challenge of providing provable network coding gaps has been addressed extensively over the years [30, 32, 33, 34]. In their influential work, Haeupler, Wajc, and Zuzic [14] demonstrated that the network coding gap for the round-complexity (i.e., makespan) of $k$ unicast tasks is at most polylogarithmic in $k$. In this work, we aim at understanding the network coding gap for the secure unicast problem. In particular, as all known secure unicast algorithms against eavesdroppers are based on some notion of coding schemes, we ask:

▶ **Question 2.** Is it possible to provide store-and-forward algorithms for secure unicast with sublinear complexity?

We provide a negative answer by establishing a network coding gap of $\Omega(n)$ rounds. To the best of our knowledge, all prior provable gaps were limited to the *logarithmic* regime, even in adversarial settings. For example, Censor-Hillel, Haeupler, Hershkowitz, and Zuzic [6] analyzed the throughput of broadcast algorithms in noisy radio networks, and established a network coding gap of $\Theta(\log n)$. In contrast, for the the fault-free setting, Alon et al. [2] showed that this gap is bounded by a constant.

## 1.1 Our Results

We present round-efficient secure algorithms for basic communication primitives. Our main result is an $f$-secure broadcast algorithm for graphs with edge-connectivity of $\Theta(f)$. Throughout, the adversarial edges controlled by the eavesdropper are denoted by $F^* \subseteq E$, and the diameter of the graph is denoted by $D$. Let $B = O(\log n)$ be the edge bandwidth of the congest model. The *congestion* of a distributed algorithm is an upper bound on the total number of messages that the algorithm sends over a given edge [10].

**Warm-Up: Secure unicast.** We start by observing that the existing secure network coding protocols for secure-unicast (e.g., by Jain [16]), can be implemented in near-optimal number of congest rounds. Using random scheduling [19, 10], this provides also efficient solutions to multicast tasks[2].

▶ **Lemma 3** (Optimal Secure-Unicast). *Given a $D$–diameter graph $G$, there is an $O(D)$-round algorithm that allows a private sender $s$ to send a message $m^*$ to a public target $t$, while leaking no information to the eavesdropper regarding the message $m^*$ or the identity of the sender $s$, provided that $s$ and $t$ are connected in $G \setminus F^*$ (i.e., even if $G \setminus F^*$ is not connected).*

A remarkable property of Lemma 3 is that its round complexity is independent of the actual number of faults, and more specifically it does not depend on the diameter of the graph $G \setminus F^*$, as one might expect. Moreover, this unicast algorithm is secure provided that the eavesdropper does not control any $s$-$t$ cut in the graph (i.e., hence potentially protecting against a large number of adversarial edges).

In the non-secure setting, the scheduling of multiple unicast problems has been usually studied in the store-and-forward model [18, 14]. In this model, the messages are viewed as atomic tokens rather than bits of information, and relay vertices can only forward some of their received messages to their neighbors, but are not allowed to mix messages. While the fault-free setting exhibits at most logarithmic gap w.r.t the number of congest rounds, we show a linear gap for solving a single unicast instance, in the presence of eavesdroppers.

▶ **Lemma 4** (Network Coding Gap for Secure Unicasts). *There exists an $n$-vertex 2-diameter graph $G^*$, an $s$-$t$ pair, and a set $F^*$ of adversarial edges, where $s$ and $t$ are connected in $G \setminus F^*$, that satisfies the following: any store-and-forward algorithm that exchanges a message from $s$ to $t$ in a secure manner must run in $\Omega(n)$ congest rounds (even if the vertices know the topology of the graph). In contrast, using network coding, the problem can be solved in $O(1)$ congest rounds.*

**Secure broadcast algorithms.** Our main result in this paper is given by an $f$-secure broadcast algorithm that delivers all the vertices a given (secret) message $m^*$, while leaking no information to the eavesdropper. We start by providing 1-secure broadcast algorithms for 2 edge-connected graphs. By a direct application of *low-congestion cycle covers* introduced by Parter and Yogev [25, 26], we obtain:

▶ **Theorem 5** (1-Secure Broadcast). *For every 2 edge-connected $D$-diameter $n$-vertex graph $G$, there is a 1-secure randomized broadcast algorithm that runs in $D \cdot n^{o(1)}$ rounds, w.h.p.*

[2] In the multicast problem, a source $s$ sends a message $m^*$ to a subset of targets $U \subseteq V$.

Handling $f$ adversarial edges (rather than just one) using cycle-covers inevitable leads to a round complexity of $(D/f)^{\Theta(f)}$ [15]. We devise a new algorithmic technique for broadcast that overcomes this inherent $(D/f)^{\Theta(f)}$ barrier exhibited by all prior algorithms in the adversarial congest model. In particular, the round complexity of the algorithm is sub-linear for every $f \in o(n)$ faults.

▶ **Theorem 6** ($f$-Secure Broadcast). *For every $(2f+3)(1+o(1))$ edge-connected $n$-vertex graph $G$, there exists a randomized $f$-secure broadcast algorithm for sending w.h.p a $b$-bit message $m^*$ that runs in $\widetilde{O}(D + \sqrt{f \cdot b \cdot n} + b)$ rounds. The edge congestion of the algorithm is $\widetilde{O}(\sqrt{f \cdot b \cdot n} + b)$. Moreover, the algorithm can also* hide *the identity of the source vertex holding the message $m^*$.*

It is interesting to note that our algorithm in fact solves the *anonymous broadcast* problem [20], in which it is also desired to hide the identity of the transmitting source. This problem is used as a black-box in many privacy-preserving applications such as anonymous communication and distributed auctions. Prior work has addressed this problem in all-to-all communication models [21], under cryptographic assumptions [7, 3], or alternatively using a large round complexity [3]. To the best of our knowledge, there has been no round-efficient congest algorithm that works for any sufficiently edge-connected graph topology.

For multiple sources $S \subseteq V$ each holding a distinct $b$-bit message, one can generalize the above algorithm to show:

▶ **Corollary 7** ($f$-Secure Multi-Source Broadcast). *Given is a $(2f+3)(1+o(1))$ edge-connected $n$-vertex graph $G = (V, E)$ and a subset of sources $S \subseteq V$ each holding a $b$-bit message. There exists a randomized $f$-secure broadcast algorithm that runs in $\widetilde{O}(D + \sqrt{f \cdot b \cdot |S| \cdot n} + b|S|)$ rounds. The edge congestion of the algorithm is $\widetilde{O}(\sqrt{f \cdot b \cdot |S| \cdot n} + b|S|)$.*

**Handling vertex adversaries.** A semi-honest adversary controlling a vertex $v$ eavesdrops over the set of all edges incident to $v$ [13]. The broadcast problem as is cannot be defined in the setting of semi-honest adversaries (as by the broadcast definition, all vertices, including the adversarial ones, are required to receive the message). We note however that if the vertices know which of their neighbors are eavesdropped, then it is indeed possible to hide the content of the broadcast message from these vertices, as described next.

We introduce the task of *Forbidden-Set Broadcast* which can be thought of as the *vertex*-analog to broadcast with eavesdroppers. In this broadcast problem the graph $G = (V, E)$ consists of two vertex types: *trusted* receivers $R$ and *untrusted* vertices $F$, where $R \cup F = V$. It is then desired for the broadcast message $m^*$ to faithfully arrive at all vertices in $R$, while all untrusted vertices $F \subseteq V$ are required to learn nothing on $m^*$, in the information-theoretic sense. I.e., the vertices in $F$ are controlled by a semi-honest adversary. This formulation might find many applications in real-life distributed networks, e.g., in settings that call for private information exchange over a distributed network that contains also (untrusted) public data-centers. We show:

▶ **Theorem 8** (Forbidden-Set Broadcast). *Given is a $D$-diameter graph $G = (V, E)$, a vertex partition into trusted receivers and untrusted vertices $R \cup F = V$, and a source vertex $s \in R$ holding a broadcast message $m^* \in \{0,1\}^B$. There is an $\widetilde{O}(D + \sqrt{|R|})$-round randomized algorithm that allows all vertices in $R$ to receive $m^*$, w.h.p, while leaking no information to any of the vertices in $R$, provided that the subgraph $G[R]$ is connected.*

It is easy to see that one can solve this task using $\mathsf{Diam}(G[R])$ rounds. It is also clear that the connectivity of $G[R]$ is a necessary condition. Our improved algorithm is based on sending messages through edges that are incident to the untrusted vertices, while guaranteeing that the semi-honest adversary controlling these edges learns nothing. Using edges in $G \setminus G[R]$ is crucial in order to improve upon the trivial bound of $O(\mathsf{Diam}(G[R]))$ rounds.

**A remark on the graph connectivity requirements.** It is well-known that $f$-security requires $(f+1)$ (edge) connectivity, as no security can be provided if the edges controlled by the eavesdropper disconnect the graph. The secure algorithms presented in this paper ask for two types of connectivity requirements. The secure unicast and forbidden-set broadcast procedures of Lemma 3 and Theorem 8 ask for a *minimal* connectivity condition. E.g., as long as $s$ and $t$ are connected in $G \setminus F^*$, the unicast algorithm is secure. In contrast, the broadcast algorithms of Theorems 5 and 6 require that the edge-connectivity of the graph is sufficiently large (above some threshold value). Theorem 5 requires 2 edge-connectivity (which is necessary), while Theorem 6 requires edge-connectivity of $(2f+3)(1+o(1))$. The larger edge connectivity requirement comes from our use of the Nash-William theorem [22], and more specifically from its implementation in the congest model by [5]. Providing efficient broadcast algorithms for $(f+1)$ edge-connected graphs in $o(n)$ rounds is a very interesting open problem.

## 1.2 Preliminaries

### 1.2.1 Graph Notation and Basic Distributed Tools

Given a graph $G = (V, E)$, denote its diameter by $D$. The neighbors of a vertex $v \in V$ are denoted by $N(v)$. Given a tree $T \subseteq G$ and a vertex $v$, let $\mathrm{ch}(v, T)$ denote the children of $v$ in $T$, and $\mathrm{par}(v, T)$ is the parent of $v$. When $T$ is clear from the context, we may omit it. The subtree of $T$ rooted at $v$ is denoted by $T_v$. For a subset of items $X$ and a probability $p \in [0, 1]$, let $X[p]$ denote the random sample obtained by sampling each item of $X$ independently with probability $p$.

**Scheduling of distributed algorithms.** Given a graph $G$, an algorithm $\mathcal{A}$ is said to have congestion at most cong if $\mathcal{A}$ sends at most cong number of messages over each edge in the graph. The dilation of $\mathcal{A}$ is the round complexity of the algorithm. We use the following useful random scheduling procedure due to Leighton, Maggs, and Rao [18], and Ghaffari [10]:

▶ **Theorem 9** ([18, 10]). *Let $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_k$ be a collection of distributed algorithms such that each algorithm takes at most* dilation *rounds, and there are at most* cong *messages sent through each edge in total throughout the execution of all these algorithms. The algorithms $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_k$ can be simulated in parallel using $O(\mathrm{cong} + \mathrm{dilation} \cdot \log^2 n)$ rounds w.h.p.*

**Basic operations on trees.** Given a spanning tree $T$ of diameter $D(T)$, the computation of aggregate functions over a set of input values of $O(\log n)$ bits, can be done in $O(D(T))$ rounds by a simple convergecast algorithm [28].

▷ Claim 10 ([28]). The convergecast of an associative and commutative function $g$ with input values $\{x_u \in \{0,1\}^{O(\log n)}\}_{u \in T_v}$ over a rooted tree $T$ can be performed in $O(D(T))$ rounds and $O(1)$ congestion. Handling $k$ functions $g_1, \ldots, g_k$ can be done in $O(D(T) + k)$ rounds, and congestion $O(k)$.

Our broadcast algorithms are based on decomposing spanning trees into edge-disjoint subtrees (denoted as *fragments*) of bounded size. Since the eavesdropper knows the graph topology, these procedures can be applied in a non-secure manner, as they reveal no information on the secret message(s) $m^*$. In the full version, we show:

▶ **Lemma 11** (Decomposition of Trees into Small Fragments). *Given an $n$-vertex spanning tree $T \subseteq G$ and a parameter $K \in [1, n]$, there exists a randomized $\widetilde{O}(K)$-round algorithm* DecomposeTree *for decomposing the edges of $T$ into edge-disjoint sub-trees $T_1, \ldots, T_q \subseteq T$, such that $|T_i| \in \Theta(K)$ for every tree $T_i$. In the distributed output format, for every sub-tree $T_i$ and a vertex $v \in T_i$, it holds that $v$ knows its incident edges in $T_i$, as well as, a unique identifier of the tree $T_i$.*

### 1.2.2 The Adversarial Setting, Security Definitions and Basic Tools

**The adversarial congest model.** We consider the standard congest model [29], in the presence of an eavesdropper controlling a fixed set of edges $F^*$, denoted as *adversarial*. The remaining edges $E \setminus F^*$ are denoted as *reliable*. The vertices do not know the identity of the edges in $F^*$, but they do know a bound $f$ on their number[3]. The eavesdropper is assumed to be computationally *unbounded*. It is allowed to know the topology of the graph $G$, and the algorithm description run by the vertices. However, it is oblivious to the internal randomness of the vertices. Our congest algorithms provide a perfect notion of *information theoretic security*, formally defined as follows.

**Perfect security against eavesdroppers.** For a given randomized algorithm $\mathcal{A}$ running on an $n$-vertex graph $G = (V, E)$, let $R_e \in \{0,1\}^{**}$ be the random variable specifying all messages sent through $e$ over the execution of $\mathcal{A}$, for every edge $e \in E$. For a subset of edges $F^* = \{e_1, \ldots, e_f\} \subseteq E$, let $M_{F^*} = [R_{e_1}, \ldots, R_{e_f}]$ be the vector of the random variables of the $F^*$ edges. Denote the $n$-length input (output) vector of the algorithm $\mathcal{A}$ by $X$ (resp., $Y$).

Algorithm $\mathcal{A}$ is said to be *secure* against an eavesdropper adversary, if for every graph $G$ and every possible assignment of input values $x_1, x_2$, and output values $y_1, y_2$, it holds that the following two are equivalent distributions:

$$\{M_{F^*} \mid X = x_1, Y = y_1\} \equiv \{M_{F^*} \mid X = x_2, Y = y_2\} \ .$$

In other words, the random variables $M_{F^*}$ and $Y, X$ are fully independent. A distributed algorithm is *$f$-secure* if it is secure against an eavesdropper controlling any fixed set of at most $f$ edges.

**One-time pad encryption.** Our algorithms encrypt messages by XORing them with random keys. This type of encryption is defined as one-time encryption. To prevent the eavesdropper from gaining information on these encrypted messages, it is crucial to use each random key exactly once (hence the phrase one-time). See [17] for further details.

▶ **Definition 12** (One-Time-Pad Encryption). *Let $x \in \{0,1\}^b$ be a $b$-bit message. In the one-time pad encryption $x$ is encrypted using a uniform random key $K \in \{0,1\}^b$, by setting $\widehat{x} = x \oplus K$. To decrypt $\widehat{x}$ using $K$, simply let $x = \widehat{x} \oplus K$.*

[3] Note that in the algorithm of Lemma 3 it is not required to know $f$.

**Secret sharing.** In a secret sharing scheme, the message is split into multiple parts, called *shares* with the following property: knowing all shares uniquely restores the message, and knowing all but one share reveals no information (in the information theoretic sense) on the original message.

▶ **Definition 13** (Secret Sharing [31]). *Given a bound on the message size $B$, a message $x \in \{0,1\}^B$ and a parameter $k$, the secret share* $\mathsf{SecretShare}(x,k,B)$ *is composed of $k$ uniformly randomly chosen strings $x^1,\ldots,x^k \in \{0,1\}^B$ called* shares*, conditioned on $x = \bigoplus_{j=1}^{k} x^j$.*

*When the message $x$ represents an integer number bounded by some integer $q$, the integer-variant, denoted as* $\mathsf{SecretShare}_{\text{int}}(m,k,q)$*, splits $x$ into $k$ randomly chosen integer shares $x^1,\ldots,x^k \in \mathbb{Z}_q$ such that $x = (\sum_{j=1}^{k} x^j) \mod q$.*

▶ **Fact 14.** *The collection of $k$ shares obtain by applying* $\mathsf{SecretShare}(x,k,B)$ *(*$\mathsf{SecretShare}_{\text{int}}(m,k,q)$*) satisfies that the joint distribution of any $k-1$ shares is uniformly distributed over $(\{0,1\}^B)^{k-1}$ (resp., $(\mathbb{Z}_q)^{k-1}$).*

# 2 Secure Unicast

## 2.1 Unicast and Multicast Algorithms

We start by observing that the existing coding-based algorithms for secure unicast [12, 16] can be implemented in $O(D)$ congest rounds. Recall that in the unicast problem, given a (possible hidden) source $s$ holding a message $m^*$, it is required for a public target $t$ to receive $m^*$ while leaking no information to the eavesdropping adversary. We next describe a distributed implementation of Jain [16] whose security guarantees hold provided that $s$ and $t$ are connected in $G \setminus F^*$.

**High level description of Alg. SecureUnicast$(s,t,m^*)$.** The algorithm starts by computing a BFS tree $T$ rooted at the target $t$, and locally setting a value $x_v = 0$ for every vertex $v \neq s$, and $x_s = m^*$. The message $m^*$ is treated as a field element in $\mathbb{F}_q$ for some sufficiently large polynomial prime $q$, hence $\sum_v x_v = m^*$. In the first step of the algorithm, the vertices orient the edges of $G$ based on the given tree $T$. They then secret share their $x_v$ values, and exchange these shares with some of their neighbors (based on the edge-orientation). At this point, each vertex $v$ holds a value $y_v$ which corresponds to the sum of its received share values. The $y_v$ values are then aggregated over the tree $T$, from the leaf vertices to the root $t$, which can then compute $\sum_v x_v = m^*$. The security is based on a combinatorial argument which essentially shows that the collection of all messages observed by the eavesdropper is equivalent to a uniform sample of fair coins, hence learning nothing on $m^*$. The complete proof of Lemma 3 is deferred to the full version.

**Simultaneous unicasts and multicast.** Using the random delay approach, we can also implement several secure unicast procedures in parallel, and obtain:

▶ **Lemma 15** (Simultaneous Unicasts). *A collection of $q$ instances for the secure unicast problem given by $\{s_i,t_i,m_i\}$ where each $m_i \in \{0,1\}^b$, can be executed in parallel using a randomized algorithm* SecureSimUnicast*. The round complexity is $O(D\log^2 n + q\cdot b/\log n)$ rounds, and the edge congestion is $\widetilde{O}(q\cdot b)$. The security holds provided that every $s_i$-$t_i$ pair is connected in $G \setminus F^*$. Moreover, the algorithm leaks no information on the identities of the senders $\{s_i\}$, provided that $G \setminus F^*$ is connected.*

The proof of Lemma 15 is deferred to the full version. This immediately yields a secure multicast procedure, where a single source $s$ sends a message $m^*$ to a subset $U$ of vertices. Since our algorithm is based on applying the unicast algorithm with $s$ as the private source, the identity of $s$ can be fully hidden from the eavesdropper.

▶ **Corollary 16** (Multicast). *Given is a private source $s$, a public subset of vertices $U \subseteq V$ and a $b$-bit message $m^*$ held by $s$. There is a randomized algorithm* SecureMulticast$(s, U, m^*)$ *that lets $s$ securely send $m^*$ to all vertices in $U$ provided that $s$-$u$ are connected in $G \setminus F^*$ for every $u \in U$. The round complexity is $\widetilde{O}(D + b \cdot |U|)$. Moreover, the algorithm leaks no information on the identity of $s$, provided that $G \setminus F^*$ is connected.*

## 2.2 A Network Coding Gap of $\Omega(n)$ Rounds for Secure Unicast

In this section, we show that when restricting to the class of store-and-forward algorithms, the task of secure unicast requires $\Omega(n)$ rounds, providing the proof of Lemma 4. Interestingly, we show that this lower bound holds even if the vertices know the identity of the edges in $F^*$, and the graph topology. In the store-and-forward model, messages are viewed as (atomic) tokens and vertices can only store and forward them. Specifically, in the unicast problem, the source vertex $s$ can send one message to each neighbor in each round, and every other vertex can send in round $r$ only one of the messages it received by round $r-1$ to each neighbor. For every parameter $n \geq 4$, we consider an $n$-vertex graph $G^*$ defined as follows.

**The lower-bound graph $G^*$.** The graph is composed of an $(n-1)$-length $s$-$t$ path $P = \{v_0 = s, v_1 \ldots, v_{n-2} = t\}$, and an additional vertex $u$ connected to all the vertices in $P$ in a star-shape manner (see Figure 1). The eavesdropper listens to all the edges that are adjacent to $u$, denoted as $F^*$ (shown in red in the figure). We note that although $s$ and $t$ are connected in $G^* \setminus F^*$, the diameter of $G^* \setminus F^*$ is $\Omega(n)$, while the diameter of $G^*$ is 2. As we will see, our lower bound argument shows that any store-and-forward secure algorithm must run in $\Omega(n)$ rounds on $G^*$, which stands in stark contrast to the network-coding based solution of Lemma 3 that solves the task in $O(1)$ rounds.

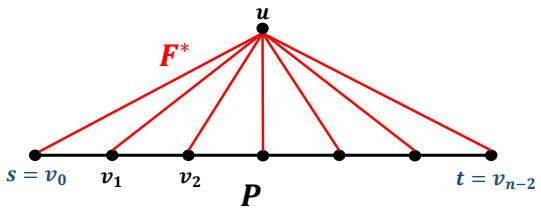


**Figure 1** An illustration of the lower bound graph $G^*$.

On a high level, the key limitation of store-and-forward algorithms that we exploit in our lower bound argument is that all messages sent throughout the algorithm must be originated at the source vertex $s$. Therefore, for every vertex $v_i \in P$, all messages received by $v_i$ in round $j \leq i-1$ must have been sent over "shortcut" edges which are in $F^*$. To be more concrete, we show by induction on $k$, that for every round $k \geq 1$ and $i \geq k+1$, the eavesdropper can calculate the probability distribution of the messages received by $v_i$ by round $k$. The key inductive argument is that for every vertex $v_i$ and a neighbor $v_j \in N(v_i)$, given the probability distribution on the messages that $v_j$ received up to round $k-1$, as the eavesdropper knows the protocol executed by $v_j$, it can deduce the probability distribution over the messages $v_j$ sent in the next round $k$.

It then follows that if the round complexity of the algorithm is at most $n-3$, the eavesdropper can deduce the probability distribution, over all messages received by the target $t$. Consequently, it can calculate the probability that $t$ outputs a given message. By the correctness of the algorithm, w.h.p, $t$ outputs the correct message $m^*$, and therefore the eavesdropper learns this message, w.h.p. We next provide the formal lower-bound proof.

**Proof of Lemma 4.** Let $\mathcal{A}$ be a randomized store-and-forward unicast algorithm, and consider an application of $\mathcal{A}$ on $G^*$ where $s$ wishes to send a message $m^*$ to $t$. Since $\mathcal{A}$ is *randomized*, in each round $k$, every vertex can produce a set of random coins. For a vertex $v \in V$ and round $k$, let $C_{\leq k}(v)$ be the collection of random coins produced by $v$ up to round $k$, and let $\mathcal{C}_{\leq k} = \bigcup_{v \in V} C_{\leq k}(v)$.

We start by noting that for every vertex $v$ and round $r$, the messages sent by $v$ in round $r$ are fully determined by the random coins $C_{\leq r}(v)$ and the information that $v$ received by round $r-1$: the messages, along with the rounds and neighbors on which they were received. For each vertex $v$, we denote the information received by $v$ up to round $k$ by:

$$\mathcal{T}_k(v) = \{(m, r_m, v_m) \mid r_m \leq k,\ v \text{ received the message } m \text{ in round } r_m \text{ from } v_m\}.$$

For every round $k$, let $M_{F^*}(k)$ be the messages sent over the edges in $F^*$ up to round $k$. Towards proving Lemma 4, we show that for every round $k \geq 1$ and $i \geq k+1$, the eavesdropper can deduce the following probability distribution on the value of $\mathcal{T}_k(v_i)$:

$$\mathcal{D}(k, v_i) = \{\Pr_{\mathcal{C}_{\leq k}}[\mathcal{T}_k(v_i) = \tau \mid M_{F^*}(k)]\}_{\tau \in \{0,1\}^*}.$$

In Appendix A, we prove by induction the following claim:

▷ Claim 17. For every round $k \geq 1$ and $i \geq k+1$, the eavesdropper can calculate the probability distribution $\mathcal{D}(k, v_i)$ by the end of round $k$.

We next show that in order to satisfy the security guarantee, the round complexity of Alg. $\mathcal{A}$ must be at least $n-2$. This completes the proof of Lemma 4.

▷ Claim 18. The round complexity of Alg. $\mathcal{A}$ is at least $n-2$.

Proof. Assume by contradiction that the round complexity of $\mathcal{A}$ is bounded by $\mu \leq n-3$. We next show that the eavesdropper can deduce the output message $m^*$, with high probability, in contradiction to the security guarantee.

Let $M_t$ be the message output by the target vertex $t$ by the end of the algorithm. By the correctness of Alg. $\mathcal{A}$, it holds that:

$$\Pr_{\mathcal{C}_{\leq \mu}}[M_t = m^*] \geq 1 - 1/n^c. \tag{1}$$

Since the algorithm maintains perfect security, the random variables $M_{F^*}(\mu)$ and $M_t$ are fully independent (see Section 1.2.2), and therefore:

$$\Pr_{\mathcal{C}_{\leq \mu}}[M_t = m^* \mid M_{F^*}(\mu)] = \Pr_{\mathcal{C}_{\leq \mu}}[M_t = m^*]. \tag{2}$$

Hence, by Equations (1) and (2) it is sufficient to show that the eavesdropper can compute the probability $\Pr_{\mathcal{C}_{\leq \mu}}[M_t = m^* \mid M_{F^*}(\mu)]$. By the law of total probability, it holds that:

$$\Pr_{\mathcal{C}_{\leq\mu}}[M_t = m^* \mid M_{F^*}(\mu)] =$$
$$= \sum_{\tau\in\{0,1\}^*} \Pr_{\mathcal{C}_{\leq\mu}}[M_t = m^* \mid \mathcal{T}_\mu(t) = \tau, M_{F^*}(\mu)] \cdot \Pr_{\mathcal{C}_{\leq\mu}}[\mathcal{T}_\mu(t) = \tau \mid M_{F^*}(\mu)] \,. \tag{3}$$

We next show that the eavesdropper can compute each term in the right-hand side of Equation (3). We will show that for every value $\tau \in \{0,1\}^*$ the eavesdropper can calculate the probabilities:

$$P_1(\tau) = \Pr_{\mathcal{C}_{\leq\mu}}[M_t = m^* \mid \mathcal{T}_\mu(t) = \tau, M_{F^*}(\mu)], \text{ and } P_2(\tau) = \Pr_{\mathcal{C}_{\leq\mu}}[\mathcal{T}_\mu(t) = \tau \mid M_{F^*}(\mu)].$$

Since $\mu \leq n-3$ and $t = v_{n-2}$, by Claim 17 and the definition of $D(\mu, t)$, for every $\tau \in \{0,1\}^*$ the eavesdropper can calculate $P_2(\tau)$. We next consider $P_1(\tau)$. The output message $M_t$ is fully determined by the tuples in $\mathcal{T}_\mu(t)$ and the random coins $C_{\leq\mu}(t)$. Hence, given the tuples in $\mathcal{T}_\mu(t)$ and the random coins $C_{\leq\mu}(t)$, the eavesdropper can determine the output message $M_t$ (with probability 1). It follows that the eavesdropper can compute the probability:

$$\Pr_{\mathcal{C}_{\leq\mu}}[M_t = m^* \mid \mathcal{T}_\mu(t) = \tau, M_{F^*}(\mu)] = P_1(\tau).$$

Altogether, by combining Equations (1)–(3), it follows that the eavesdropper can deduce the message $m^*$ w.h.p., in contradiction to the security guarantee. ◁

# 3 Secure Broadcast Algorithms

In this section, we present broadcast algorithms which remain confidential in the presence of an eavesdropping adversary. We start by describing an $\widetilde{O}(D)$-round algorithm for two edge-connected graphs, in the presence of a single adversarial edge, i.e., $|F^*| = 1$. Extending this algorithm to multiple edges $f$ can be provably shown to require $(D/f)^{\Omega(f)}$ rounds (by [15]). We then present an alternative approach that bypasses the $(D/f)^{\Omega(f)}$-barrier at the cost of requiring a slightly larger edge-connectivity. Throughout this section, the source vertex is denoted by $s$, and the broadcast message by $m^*$.

## 3.1 Handling a Single Adversarial Edge

The broadcast algorithm is based on the notion of low-congestion cycle covers introduced by Parter and Yogev [25]. For a given 2 edge-connected graph $G = (V, E)$, a $(c, d)$-cycle cover is a collection of cycles $\mathcal{C}$ such that (i) each edge $e \in G$ participates in at least one cycle, and at most $c$ cycles in $\mathcal{C}$, and (ii) all cycles are of length at most $d$.

▶ **Lemma 19** ([25, 26]). *There is a deterministic $r$-round algorithm that for every $n$-vertex two edge-connected graph $G = (V, E)$ computes a $(c, d)$-cycle cover $\mathcal{C}$ for $G$, where $c = 2^{O(\sqrt{\log n})}$ and $d, r = D \cdot 2^{O(\sqrt{\log n})}$. In the distributed output format, each vertex knows its incident edges on each cycle $\mathcal{C}$ as well as, a unique identifier for these cycles.*

**The algorithm.** The algorithm has two main phases. The first phase provides a secret key $r_{u,v} \in \{0,1\}^B$ for every neighboring pair $u, v$ in $G$. These keys are *hidden* from the eavesdropper. The second phase implements a standard (fault-free) broadcast algorithm with the only distinction that the messages exchanged (in the fault-free algorithm) are now encrypted with the $\{r_{u,v}\}$ keys.

The algorithm starts by computing a cycle-cover $\mathcal{C}$ for $G$ using Lemma 19. We use these cycles to provide $u, v$ with a key $r_{u,v}$ that is hidden from the eavesdropper. This is done by letting $u$ generate two independent random keys $r'_{u,v}, r''_{u,v} \in \{0,1\}^B$ which are then sent to $v$ using of the cycle $C_e$ as follows: $r'_{u,v}$ is sent directly over the edge $(u, v)$, and $r''_{u,v}$ is sent to $v$ over the $u$-$v$ path $C_e \setminus \{(u, v)\}$. The vertex $v$ then sets $r_{u,v} = r'_{u,v} \oplus r''_{u,v}$.

Finally, the algorithm propagates the message $m^*$ from $s$ downwards a BFS tree rooted at $s$. By round $i \geq 0$ every vertex $u$ at layer $i$ are assumed to know $m^*$. In round $i$ every vertex $u$ at layer $i$ sends the encrypted message $m^* \oplus r_{u,v}$ to each child $v$ in the tree. This completes the description of the algorithm.

**Correctness and security.** Each neighboring pair $(u, v)$ share the key $r_{u,v}$. Therefore, each vertex $v$ can decrypt the received message and obtain $m^*$. Since the eavesdropper controls a fixed edge in the graph, it learns nothing on the collection of $\{r_{u,v} : (u, v) \in T\}$ keys. To see this, consider a single key $r_{u,v}$ exchanged over the cycle $C$. Since the eavesdropper controls at most one edge on that cycle, it knows either $r'_{u,v}$ or $r''_{u,v}$ (but not both). Therefore it learns nothing on $r_{u,v}$. Since all messages sent through the tree $T$ are encrypted using one-time pad encryption (Definition 12) with their corresponding keys, the eavesdropper learns nothing on the message $m^*$.

**Running time.** The time consuming step is the exchange of the keys over all cycles in $\mathcal{C}$. For every edge $(u, v)$, let $\mathcal{A}_{(u,v)}$ be the algorithm that exchanges the $O(\log n)$-length keys $r'_{u,v}, r''_{u,v}$ over the cycle covering $(u, v)$. By the properties of the $(c, d)$ cycle-cover, each cycle $C \in \mathcal{C}$ can be used to cover at most $|C| \leq d$ many edges, and since each edge participates in at most $c$ cycles, overall each edge participates in $O(c \cdot d)$ algorithms. Therefore, the total congestion of the collection of $m$ algorithms is bounded by $O(c \cdot d)$, and the dilation of each algorithm is $d$. Using the random-delay based scheduling of Theorem 9, we can implement the algorithms $\{\mathcal{A}_{(u,v)} : (u, v) \in E\}$ simultaneously within $\widetilde{O}(c \cdot d)$ rounds. Theorem 5 follows by Lemma 19.

## 3.2 Handling Multiple Adversarial Edges

We now turn to consider $f$-secure broadcast algorithms against $f$ adversarial edges $F^*$, and prove Theorem 6. Let $m^*$ be a $b$-bit message held by the source. Our $f$-secure algorithm is based on the distributed computation of a tree collection denoted as *fractional tree packing* [5]. Note that to this date, there is no round-efficient algorithm for computing an *integral* tree-packing.

▶ **Definition 20** (Fractional Tree Packing). *A* fractional tree-packing *of a graph $G$ is a collection of spanning trees $\mathcal{T}$, and a weight function $w : \mathcal{T} \to (0, 1]$, such that for every edge $e \in E$, $\sum_{T_i \in \mathcal{T} : e \in T_i} w(T_i) \leq 1$. The size of the fractional tree packing $\mathcal{T}$ is denoted by $\chi(\mathcal{T}) = \sum_{T_i \in \mathcal{T}} w(T_i)$.*

Our algorithm is based on the distributed computation of fractional tree packing, due to Censor-Hillel, Ghaffari, and Kuhn [5]. Our setting requires a slight adaptation of the construction by [5][4], as summarized in the next lemma. See the full version for the proof.

[4] E.g., for our purposes, it is important that the running time would depend on the number of faulty edges, $f$, rather than on the actual edge connectivity $\lambda$, which might be significantly larger than $f$.

▶ **Lemma 21** (A slight adaptation of Theorem 1.3 [5]). *Given a $D$-diameter $\lambda$ edge-connected $n$-vertex graph, and an integer parameter $\lambda' \leq \frac{\lambda-1}{2}(1-o(1))$, one can compute a fractional tree packing $\mathcal{T}$ and a weight function $w : \mathcal{T} \to (0,1]$, such that:*

1. *for every $T_i \in \mathcal{T}$, $w(T_i) = \frac{n_i}{\lceil \log^8 n \rceil}$ for some positive integer $n_i \geq 1$,*
2. *$\chi(\mathcal{T}) \in [\lambda', \lambda'(1+o(1))]$,*
3. *the round complexity of the algorithm is $\widetilde{O}(D + \sqrt{\lambda' \cdot n})$ and the edge-congestion is $\widetilde{O}(\sqrt{\lambda' \cdot n})$.*

We are now ready to provide the complete description of Alg. SecureEavesBroadcast given a source $s$ holding a broadcast message $m^*$ of $b$-bits (where possibly $b = \Omega(\log n)$).

**Step (0): Fractional tree decomposition.** Apply the fractional tree-packing algorithm of Lemma 21 with $\lambda' = f + 1$. Denote the output tree packing by $\mathcal{T}$ and its size by $\chi(\mathcal{T})$. By the end of this computation, the vertices know the weights $n_1, \ldots, n_{|\mathcal{T}|}$ of the trees in $\mathcal{T}$ (see Lemma 21(1)).

**Step (1): Multicasting secret shares of $m^*$ to sampled landmarks.** The source vertex $s$ secret shares its $b$-bit broadcast message $m^*$ into

$$\widehat{f} = \chi(\mathcal{T}) \cdot \lceil \log^8 n \rceil \quad \text{shares} \tag{4}$$

using Procedure SecretShare$(m^*, \widehat{f}, b)$ (Definition 13), denoted as $M^* = (m_1, \ldots, m_{\widehat{f}})$, where each share $m_i$ has $b$ bits. Note that by Lemma 21(1), $\widehat{f}$ is an integer. Next, the algorithm samples a collection of landmarks $L = V[p]$ for

$$p = \Theta(\log n / \min\{\sqrt{f \cdot b \cdot n}, n\}).$$

This is done by letting each vertex join $L$ independently with probability $p$. The identities of the sampled vertices $L$ are broadcast to all the vertices[5] in $O(D + |L|)$ rounds. Next, the algorithm applies Alg. SecureMulticast$(s, L, M^*)$ of Cor. 16 to securely send the landmarks $L$ the collection of all shares.

**Step (2): Fragmentation of tree-packing and leader selections.** Decompose each tree $T_i \in \mathcal{T}$ into fragments $T_{i,j}$ of size $\Theta(\min\{\sqrt{f \cdot b \cdot n}, n\})$ using Alg. DecomposeTree$(T_i)$ of Lemma 11. In addition, for every fragment $T_{i,j}$, let $\ell_{i,j}$ be some chosen vertex in $L \cap V(T_{i,j})$, denoted as the *leader*, which exists with high probability. This leader can be chosen in $D(T_{i,j})$ rounds by a simple convergecast over each fragment, simultaneously.

**Step (3): Shares propagation in each fragment.** Recall that by Lemma 21(1), $w(T_i) = \frac{n_i}{\lceil \log^8 n \rceil}$ for some positive integer $n_i \geq 1$ for every tree $T_i \in \mathcal{T}$. Our goal is to propagate a distinct collection of $n_i$ shares in $M^*$ over each tree $T_i$. To do that, the landmark vertices partition locally and canonically the shares of $M^*$ into disjoint subsets $M_1^*, \ldots, M_k^*$ such that $|M_i^*| = n_i$ for every $i \in \{1, \ldots, k = |\mathcal{T}|\}$. Note that by Equation (4), $\sum_{T_i \in \mathcal{T}} n_i = \widehat{f}$. The leader $\ell_{i,j}$ of each fragment $T_{i,j}$ sends all the shares in $M_i^*$ over its fragment. Finally, each vertex $v$ recovers the $b$-bit broadcast message, by setting $m^* = \oplus_{i=1}^{k} \oplus_{m \in M_i^*} m$ . This completes the description of the algorithm.

[5] This can be done in a non-secure manner, as the eavesdropper is allowed to know $L$.

**Correctness.** Since the edge-connectivity of $G$ is at least $(2f+3)(1+o(1))$, one can obtain a fractional tree packing $\mathcal{T}$ of size $\chi(\mathcal{T}) \in [f+1, (f+1)(1+o(1))]$ using Lemma 21(2). We claim that each vertex receives w.h.p. all the shares in $M^*$ and therefore recovers $m^*$ successfully. Since all the trees $T_i \in \mathcal{T}$ are spanning, a vertex $v$ belongs to some fragment $T_{i,j_v}$ for every $T_i \in \mathcal{T}$. Since each fragment has $|V(T_{i,j})| \in \Theta(\min\{\sqrt{f \cdot b \cdot n}, n\})$ vertices, and each vertex is sampled into $L$ with probability $p = \Theta(\log n / \min\{\sqrt{f \cdot b \cdot n}, n\})$, by the Chernoff bound, w.h.p., each fragment contains some landmark vertex in $L$. Therefore, each vertex $v$ receives all shares in $M_i^*$ over the fragment $T_{i,j_v}$, for every $T_i \in \mathcal{T}$. The correctness follows as $\bigcup_{i=1}^{k} M_i^* = M^*$, where $k$ denotes the number of trees in $\mathcal{T}$.

**Security.** Recall that the eavesdropper is assumed to know $G$, therefore Step (0) and Step (2) can be implemented in a non-secure manner. Since $G \setminus F^*$ is connected, Step (1) is $f$-secure by Cor. 16. We turn to consider Step (3) in which the shares in $M^*$ are sent over the tree fragments of $\mathcal{T}$. We show that there exists at least one share in $M^*$ that the eavesdropper did not learn, and therefore, by Fact 14, it knows nothing on $m^*$. Since the tree fragments of each $T_i \in \mathcal{T}$ are edge-disjoint, the number of shares in $M^*$ sent over an edge $e$ is bounded by:

$$\sum_{e \in T_i} |M_i^*| = \sum_{e \in T_i} n_i \leq \lceil \log^8 n \rceil \ , \tag{5}$$

where the last inequality follows by the fractional tree packing guarantee that $\sum_{e \in T_i} w(T_i) \leq 1$, and using the fact that $w(T_i) = \frac{n_i}{\lceil \log^8 n \rceil}$ for every $T_i \in \mathcal{T}$. Therefore, the number of shares observed by the eavesdropper can be bounded by:

$$f \cdot \lceil \log^8 n \rceil < \chi(\mathcal{T}) \cdot \lceil \log^8 n \rceil = \widehat{f} = |M^*| \ ,$$

where the first inequality is by Lemma 21(2) (with $\lambda' = f+1$), and the last equality follows by Equation (4).

**Round complexity.** Finally, we turn to bound the round complexity and the edge congestion of the algorithm. In Step (0) the computation of the fractional tree packing can be done in $\widetilde{O}(D + \sqrt{f \cdot n})$ rounds by Lemma 21(3) with $\lambda' = f+1$. Step (1) takes $\widetilde{O}(D + f \cdot b \cdot |L|)$ rounds, by Cor. 16. Since, w.h.p., $|L| = \widetilde{O}(\sqrt{n/(f \cdot b)})$, it takes $\widetilde{O}(D + \sqrt{f \cdot b \cdot n})$ rounds. By Lemma 11 Step (2) takes $\widetilde{O}(\min\{\sqrt{f \cdot b \cdot n}, n\})$ rounds. As for Step (3), since $w(T_i) \leq 1$, it holds that $n_i \leq \lceil \log^8 n \rceil$ for every $T_i \in \mathcal{T}$. Therefore, Step (3) propagates $\widetilde{O}(b)$-bit messages over the edge-disjoint fragments of size $\Theta(\sqrt{f \cdot b \cdot n})$. This can be done in $\widetilde{\Theta}(b + \sqrt{f \cdot b \cdot n})$ rounds via standard pipeline.

It remains to bound the edge congestion. By Lemma 21(3) the edge congestion of Step (0) is $\widetilde{O}(\sqrt{f \cdot n})$. Step (1) sends $\widetilde{O}(\sqrt{f \cdot n})$ messages over each edge, by Cor. 16. The edge congestion of Step (2) is bounded by $\widetilde{O}(\sqrt{f \cdot b \cdot n})$, by Lemma 11. Finally, Step (3) sends $\widetilde{O}(b)$ bits on each edge by Equation (5). Thus, overall the edge congestion is bounded by $\widetilde{O}(b + \sqrt{f \cdot b \cdot n})$.

**Bonus property: Anonymous broadcast.** We note that Alg. SecureEavesBroadcast can also hide from the eavesdropper, not only the identity of the broadcast message $m^*$, but also the identity of the sender $s$. The only involvement of the source $s$ is in Step (1), i.e., in the application of the SecureMulticast algorithm of Cor. 16. Since the latter hides the identity of the source $s$, the final broadcast algorithm leaks no information on $s$, as well.

**Extension to broadcast with multiple sources, proof of Cor. 7.** Consider a collection of sources $S \subseteq V$, where each $s \in S$ holds a $b$-bit message $m_s^*$. For simplicity of explanation, in the following, we do not try to hide the identity[6] of $S$. Hence, within $O(D + |S|)$ rounds, all vertices can define an ordering on these sources, given by $s_1, \ldots, s_k$ (e.g., based on IDs).

The algorithm is based on a reduction to the ($f$-secure) single-source broadcast of Theorem 6 using message size of $b' = b \cdot |S|$. In the first phase, the algorithm picks some arbitrary vertex $s$ (possibly not in $S$) and let it locally define a $b'$-bit message $r^*$ chosen uniformly at random in $\{0,1\}^{b'}$. The source $s$ applies Alg. SecureEavesBroadcast with the message $m^* = r^*$. This can done with $\widetilde{O}(D + \sqrt{f \cdot b \cdot |S| \cdot n})$ rounds. At this point, all the vertices share a random string $r^*$ of $b'$ bits, which can be locally partitioned into $|S|$ random keys $r_1, \ldots, r_{|S|} \in \{0,1\}^b$. Each $r_i$ is used as a random key for encrypting the broadcast message of the $i^{th}$ source $s_i \in S$, as follows.

Each source $s_i \in S$ encrypts its $b$-bit broadcast message $m_s^*$ by letting $r_s^* = m_s^* \oplus r_i$ (one-time padding). Then, we apply the *standard* (i.e., non-secure) broadcast procedure w.r.t each $s_i$ and the message $r_s^*$. Using the random-delay approach of Theorem 9, all these procedures can be done in parallel within $\widetilde{O}(D + b|S|)$ rounds. As each vertex $v$ knows $r^*$, it can recover the $i^{th}$ message by letting $m_s^* = r_s^* \oplus r_i$. The security of the algorithm simply follows by the security guarantees of the single-source broadcast algorithm of Theorem 6.

## 4 Forbidden-Set Broadcast

In this section, we turn to consider secure broadcast algorithms in the presence of a semi-honest adversary that controls a subset of the *vertices*. This adversary can be trusted to run the protocol honestly but aim to extract information on the vertices' input and output. Recall that in the forbidden-set broadcast problem, given is an $n$-graph $G = (V, E)$ with a vertex partition $R \cup F = V$, into trusted *receivers* $R$ and untrusted vertices $F$, controlled by a semi-honest adversary. It is assumed that each vertex knows whether it is in $R$ or $F$ (hence, each vertex in $R$ knows its neighbors in $R$), and that $G[R]$ is connected (which is essential). It is then required for a source vertex $s \in R$ to send a broadcast message $m^*$ (say of $O(\log n)$-bits) to all vertices in $R$, while leaking no information to the eavesdropper controlling the vertices in $F$. That is, the collection of all messages received by the vertices in $F$ are required to convey no information on $m^*$, in the information-theoretic sense.

We start by observing that the security guarantees of the unicast and multicast algorithms of Section 2 also hold in the presence of semi-honest adversaries, provided that $G[R]$ is connected. This allows us to securely exchange messages between $s, t$ pairs in $R$ using $O(D)$ rounds. In the full version, we show:

▷ Claim 22 (Secure Unicast and Multicast against Semi-Honest Adversaries). The security guarantees of Alg. SecureUnicast and Alg. SecureMulticast hold in the presence of semi-honest adversaries, provided that $G[R]$ is connected.

A trivial solution for the problem works by sending the message $m^*$ over the graph $G[R]$. This, however, might possibly lead to a round complexity of $|R|$, which might be linear in $n$. Instead, our $\widetilde{O}(D + \sqrt{|R|})$-round algorithm is based on *using* the *untrusted* vertices for the purpose of faster communication (i.e., as relay vertices), but in a way that guarantees that these vertices still learn nothing on the secret $m^*$. The algorithm has three phases. First, it samples a collection of $\widetilde{O}(\sqrt{|R|})$ vertices from $R$, denoted as *landmarks*. The source $s$

[6] This can be done by small parametrization of Alg. SecureEavesBroadcast.

sends the message $m^*$ to these landmarks using the secure multicast procedure of Claim 22 and Cor. 16. The algorithm then computes a MST $T$ in $G[R]$, and decomposes it into a collection of tree *fragments* $\mathcal{T}$, each of size $\Theta(\sqrt{|R|})$ using Lemma 11. Note that, w.h.p., each fragment contains at least one sampled landmark. Finally, each landmark propagates $m^*$ over the edges of its fragment, this is done in all the fragments of $\mathcal{T}$, in parallel.

**Algorithm 1** ForbiddenSetBroadcast.

**Input**: Graph $G = (V, E)$, $s$ holds a message $m^*$, a vertex partition $V = F \cup R$.
**Output**: All vertices in $R$ learn the message $m^*$, while leaking no information to $F$.

- **Step (1): Multicast $m^*$ to $\widetilde{O}(\sqrt{|R|})$ Landmarks.**
  - Sample a landmark set $L = R[p]$ for $p = \Theta(\log n/\sqrt{|R|})$.
  - Apply Alg. SecureMulticast$(s, L, m^*)$ of Cor. 16.
- **Step (2): Tree Fragmentation**
  - Compute a spanning tree $T$ in $G[R]$ (e.g., using the MST algorithm of [11]).
  - Apply Alg. DecomposeTree$(T, \sqrt{|R|})$ to decompose $T$ into edge-disjoint trees, $\mathcal{T} = \{T_1, \ldots, T_\ell\}$ such that $|T_i| = \Theta(\sqrt{|R|})$ and $\bigcup_i V(T_i) = R$.
- **Step (3): Broadcast over the Fragments.**
  - For each fragment $T_i$, each landmark $\ell \in T_i$ broadcasts $m^*$ over $T_i$.

**Correctness and security.** Since the size of each tree fragment $T_i$ is $\Theta(\sqrt{|R|})$, by the Chernoff bound, we have that w.h.p. $V(T_i) \cap L \neq \emptyset$ for every $T_i \in \mathcal{T}$. As the collection of $T_i$ trees covers all vertices in $R$, we get that, w.h.p., all vertices in $R$ receive the message $m^*$. We now consider security and show that a semi-honest adversary controlling all vertices in $F = V \setminus R$ learns nothing on $m^*$. The security of Step (1) follows by Claim 22. Since the semi-honest adversary knows the graph topology, Step (2) reveals no information on the secret message $m^*$. Additionally, Step (3) sends messages only on edges in $G[R]$, and therefore it is secure as well.

**Running time.** By Chernoff, w.h.p., it holds that $|L| = O(\sqrt{|R|} \cdot \log n)$. Therefore by Cor. 16, Step (1) is implemented in $\widetilde{O}(D + \sqrt{|R|})$ rounds. The computation of $T$ can be done in $O(D + \sqrt{|R|})$ rounds, using a standard MST procedure, e.g., applying the low-congestion shortcut framework of [11]. The fragmentation of $T$ into the trees $\mathcal{T}$ in Step (2) takes $\widetilde{O}(\sqrt{|R|})$ rounds, using Lemma 11. Finally, the communication in each tree $T_i \in \mathcal{T}$ of Step (3) takes $O(|T_i|) = O(\sqrt{|R|})$ rounds. As all trees are edge-disjoint, this can be done in parallel over all vertices using $O(\sqrt{|R|})$ rounds. Overall, the round complexity is $\widetilde{O}(D + |\sqrt{|R|})$. This completes the proof of Theorem 8.

## References


1 Rudolf Ahlswede, Ning Cai, Shuo-Yen Robert Li, and Raymond W. Yeung. Network information flow. *IEEE Trans. Inf. Theory*, 46(4):1204–1216, 2000.

2 Noga Alon, Mohsen Ghaffari, Bernhard Haeupler, and Majid Khabbazian. Broadcast throughput in radio networks: Routing vs. network coding. In Chandra Chekuri, editor, *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 1831–1843. SIAM, 2014.

3 Marshall Ball, Elette Boyle, Ran Cohen, Lisa Kohl, Tal Malkin, Pierre Meyer, and Tal Moran. Topology-hiding communication from minimal assumptions. *IACR Cryptol. ePrint Arch.*, page 388, 2021.

4 Ning Cai and Raymond W Yeung. Secure network coding. In *Proceedings IEEE International Symposium on Information Theory,*, page 323. IEEE, 2002.

5 Keren Censor-Hillel, Mohsen Ghaffari, and Fabian Kuhn. Distributed connectivity decomposition. In Magnús M. Halldórsson and Shlomi Dolev, editors, *ACM Symposium on Principles of Distributed Computing, PODC '14, Paris, France, July 15-18, 2014*, pages 156–165. ACM, 2014.

6 Keren Censor-Hillel, Bernhard Haeupler, D. Ellis Hershkowitz, and Goran Zuzic. Broadcasting in noisy radio networks. In Elad Michael Schiller and Alexander A. Schwarzmann, editors, *Proceedings of the ACM Symposium on Principles of Distributed Computing, PODC 2017, Washington, DC, USA, July 25-27, 2017*, pages 33–42. ACM, 2017. doi:10.1145/3087801.3087808.

7 David L Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–90, 1981.

8 Danny Dolev, Cynthia Dwork, Orli Waarts, and Moti Yung. Perfectly secure message transmission. In *31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume I*, pages 36–45. IEEE Computer Society, 1990.

9 Jon Feldman, Tal Malkin, Cliff Stein, and Rocco A Servedio. On the capacity of secure network coding. In *Proc. 42nd Annual Allerton Conference on Communication, Control, and Computing*, pages 63–68. Cambridge University Press, 2004.

10 Mohsen Ghaffari. Near-optimal scheduling of distributed algorithms. In Chryssis Georgiou and Paul G. Spirakis, editors, *Proceedings of the 2015 ACM Symposium on Principles of Distributed Computing, PODC 2015, Donostia-San Sebastián, Spain, July 21 - 23, 2015*, pages 3–12. ACM, 2015.

11 Mohsen Ghaffari and Bernhard Haeupler. Distributed algorithms for planar networks II: low-congestion shortcuts, mst, and min-cut. In Robert Krauthgamer, editor, *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 202–219. SIAM, 2016.

12 Niv Gilboa and Yuval Ishai. Compressing cryptographic resources. In Michael J. Wiener, editor, *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, volume 1666 of *Lecture Notes in Computer Science*, pages 591–608. Springer, 1999.

13 Oded Goldreich. *The Foundations of Cryptography - Volume 2: Basic Applications*. Cambridge University Press, 2004. doi:10.1017/CBO9780511721656.

14 Bernhard Haeupler, David Wajc, and Goran Zuzic. Network coding gaps for completion times of multiple unicasts. In Sandy Irani, editor, *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 494–505. IEEE, 2020.

15 Yael Hitron and Merav Parter. General CONGEST compilers against adversarial edges. In Seth Gilbert, editor, *35th International Symposium on Distributed Computing, DISC 2021, October 4-8, 2021, Freiburg, Germany (Virtual Conference)*, volume 209 of *LIPIcs*, pages 24:1–24:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

16 Kamal Jain. Security based on network topology against the wiretapping attack. *IEEE Wirel. Commun.*, 11(1):68–71, 2004.

17 Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography, Second Edition*. CRC Press, 2014.

18 Frank Thomson Leighton, Bruce M. Maggs, and Satish Rao. Packet routing and job-shop scheduling in $O$(congestion + dilation) steps. *Comb.*, 14(2):167–186, 1994.

19 Frank Thomson Leighton, Bruce M Maggs, and Satish B Rao. Packet routing and job-shop scheduling ino (congestion+ dilation) steps. *Combinatorica*, 14(2):167–186, 1994.

20 Benoît Libert, Kenneth G. Paterson, and Elizabeth A. Quaglia. Anonymous broadcast encryption: Adaptive security and efficient constructions in the standard model. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *Public Key Cryptography - PKC 2012 - 15th International Conference on Practice and Theory in Public Key Cryptography, Darmstadt, Germany, May 21-23, 2012. Proceedings*, volume 7293 of *Lecture Notes in Computer Science*, pages 206–224. Springer, 2012.

21 Mahnush Movahedi, Jared Saia, and Mahdi Zamani. Secure anonymous broadcast. In Fabian Kuhn, editor, *Distributed Computing - 28th International Symposium, DISC 2014, Austin, TX, USA, October 12-15, 2014. Proceedings*, volume 8784 of *Lecture Notes in Computer Science*, pages 567–568. Springer, 2014.

22 C. St.J. A. Nash-Williams. Edge-Disjoint Spanning Trees of Finite Graphs. *Journal of the London Mathematical Society*, s1-36(1):445–450, 1961.

23 Merav Parter and Eylon Yogev. Distributed algorithms made secure: A graph theoretic approach. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 1693–1710. SIAM, 2019.

24 Merav Parter and Eylon Yogev. Low congestion cycle covers and their applications. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 1673–1692, 2019.

25 Merav Parter and Eylon Yogev. Optimal short cycle decomposition in almost linear time. In *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, pages 89:1–89:14, 2019.

26 Merav Parter and Eylon Yogev. Optimal short cycle decomposition in almost linear time. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, volume 132 of *LIPIcs*, pages 89:1–89:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

27 Merav Parter and Eylon Yogev. Secure distributed computing made (nearly) optimal. In *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing, PODC 2019, Toronto, ON, Canada, July 29 - August 2, 2019*, pages 107–116, 2019.

28 David Peleg. Time-optimal leader election in general networks. *J. Parallel Distributed Comput.*, 8(1):96–99, 1990.

29 David Peleg. *Distributed Computing: A Locality-sensitive Approach*. SIAM, 2000.

30 Christian Scheideler. *Universal routing strategies for interconnection networks*, volume 1390. Springer Science & Business Media, 1998.

31 Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.

32 Aravind Srinivasan and Chung-Piaw Teo. A constant-factor approximation algorithm for packet routing and balancing local vs. global criteria. *SIAM J. Comput.*, 30(6):2051–2068, 2000.

33 Chih-Chun Wang and Minghua Chen. Sending perishable information: Coding improves delay-constrained throughput even for single unicast. *IEEE Transactions on Information Theory*, 63(1):252–279, 2016.

34 Xunrui Yin, Zongpeng Li, Yaduo Liu, and Xin Wang. A reduction approach to the multiple-unicast conjecture in network coding. *IEEE Transactions on Information Theory*, 64(6):4530–4539, 2017.


## A Missing Proofs of Section 2

**Proof of Claim 17.** We show by induction on $k$ that given an assignment to the random coins $\mathcal{C}_{\leq k}$ and the messages $M_{F^*}(k)$, the eavesdropper can deduce $\mathcal{T}_k(v_i)$ for every $i \geq k+1$. As the eavesdropper knows the probability distribution of $\mathcal{C}_{\leq k}$, it follows that the eavesdropper can calculate the distribution $D(k, v_i)$ for every $i \geq k+1$.

For the base case of $k = 1$, in the first round, the only vertex that can send messages is $s = v_0$. Hence, for every $i \geq 2$, the vertex $v_i$ receives no messages in the first round and $\mathcal{T}_1(v_i) = \emptyset$. Since the eavesdropper knows the graph topology, it can deduce that $\mathcal{T}_1(v_i) = \emptyset$ (regardless of the coins $\mathcal{C}_{\leq 1}$ and the messages $M_{F^*}(1)$). Assume that given an assignment to $\mathcal{C}_{\leq k-1}$ and $M_{F^*}(k)$, the eavesdropper can deduce $\mathcal{T}_k(v_i)$, for every $i \geq k$.

We next consider round $k$, and a fixed assignment to $M_{F^*}(k)$ and $\mathcal{C}_{\leq k}$. Let $v_i$ be a vertex such that $i \geq k + 1$. Since $i \geq k$, by the induction assumption the eavesdropper can deduce the tuples in $\mathcal{T}_{k-1}(v_i)$ based on $M_{F^*}(k-1) \subseteq M_{F^*}(k)$ and $\mathcal{C}_{\leq k-1} \subseteq \mathcal{C}_{\leq k}$. We are left to show the eavesdropper can learn the messages received by $v_i$ from its neighbors in round $k$.

Recall that $v_i$ has three neighbors, $v_{i-1}, v_{i+1}$ and $u$. As the eavesdropper controls the edge $(u, v_i) \in F^*$, the message sent from $u$ to $v_i$ in round $k$ is determined by $M_{F^*}(k)$. For $v_j \in \{v_{i-1}, v_{i+1}\}$, we note that the message $v_j$ sent to $v_i$ in round $k$ is fully determined by the information received by $v_j$ up to round $k-1$ (i.e., $\mathcal{T}_{k-1}(v_j)$) and the random coins $\mathcal{C}_{\leq k}(v_j)$. Since $i - 1 \geq k - 1$, by the induction assumption the eavesdropper can deduce $\mathcal{T}_{k-1}(v_j)$ based on $M_{F^*}(k-1)$ and $\mathcal{C}_{\leq k-1}$. It follows that given $\mathcal{C}_{\leq k}$ and $M_{F^*}(k)$, the eavesdropper can compute the messages $v_{i-1}$ and $v_{i+1}$ sent to $v_i$ in round $k$. The claim follows. ◀