Time-Delayed Broadcasting for Defeating Inside Jammers

Sisi Liu, *Student Member, IEEE,* Loukas Lazos, *Member, IEEE,* and Marwan Krunz, *Fellow, IEEE*
Department of Electrical and Computer Engineering, University of Arizona, Tucson, AZ, 85721
E-mail:{sisimm, llazos, krunz}@ece.arizona.edu

Abstract—We address the problem of jamming-resistant broadcast communications under an *internal threat model*. We propose a time-delayed broadcast scheme (TDBS), which implements the broadcast operation as a series of unicast transmissions distributed in frequency and time. TDBS does not rely on commonly shared secrets, or the existence of jamming-immune control channels for coordinating broadcasts. Instead, each node follows a unique pseudo-noise (PN) frequency hopping sequence. Contrary to conventional PN sequences designed for multi-access systems, the PN sequences in TDBS exhibit correlation to enable broadcast. Moreover, they are designed to limit the information leakage due to the exposure of a subset of sequences by compromised nodes. We map the problem of constructing such PN sequences to the 1-factorization problem for complete graphs. We further accommodate dynamic broadcast groups by mapping the problem of updating the assigned PN sequences to the problem of constructing rainbow paths in proper edge-colored graphs.

Index Terms—Jamming, broadcast communications, denial-of-service, wireless networks, graph factorization, security.

1 Introduction

Wireless communications are vulnerable to intentional interference attacks, typically referred to as jamming. Conventional anti-jamming techniques rely on spread spectrum (SS) communications, such as direct sequence spread spectrum (DSSS) and frequency hopping spread spectrum (FHSS) [30]. DSSS provides bit-level protection by spreading bits according to a secret pseudo-random noise (PN) code, known only to the communicating parties. In FHSS, the sender and the receiver hop synchronously using a secret random frequency hopping (FH) sequence. For jamming-resistant broadcast communications, a common secret to be shared between the sender and all (potentially non-trustworthy) receivers. The disclosure of this common secret due to the compromise of any receiver nullifies the SS gains [19], [26].

Several researchers have studied the broadcasting problem in the presence of inside jammers [6], [10], [16], [17], [19], [26], [27], [31], [32]. Methods in [6], [16], [17], [19], [27] eliminate the dependency of SS on shared secrets. Baird et al. proposed the encoding of "indelible marks" at specific locations within each broadcasted message [6]. Assuming that the jammer cannot flip a bit '1' to a bit '0', it was shown that a jammer cannot erase packets from the wireless channel. Pöpper et al. [26] proposed a method called Uncoordinated DSSS (UDSSS), in which broadcast transmissions are spread according to a PN code, randomly selected from a public codebook. Receivers decode transmitted messages by exhaustively applying every PN code in the public codebook. An advanced adversary with ample computation power can jam a UDSSS system if it can recover the selected PN code by exhausting the public codebook before the end of an ongoing transmission. Liu et. al. proposed RD-DSSS, which is resilient to reactive jammers by delaying the disclosure of the secret PN code. The computational efficiency of RD-DSSS was further improved in [17] by encoding the seed of the PN code used to spread a message, at the end of that message. This delayed seed disclosure prevented a jammer from acquiring the PN code, before the message was fully received. In this work, we develop anti-jamming methods that adopt the FHSS design. FHSS is known to exhibit a graceful degradation in performance with the increase of interference [25], [30].

Our Contributions: We propose the *Time-Delayed Broadcast Scheme* (TDBS).as an emergency mechanism for temporarily restoring broadcast communications until inside jammers are physically removed from the network. TDBS differs from classical FHSS designs in that two communicating nodes do not follow the same FH sequence, but are assigned unique ones. Unlike the typical broadcast in which all receivers tune to the same channel, TDBS propagates broadcast messages as a series of unicast transmissions, spread both in frequency and time. To ensure resilience to inside jammers, the locations of these unicast transmissions, defined by a frequency band/slot pair, are only partially known to any subset of receivers. Assuming that the jammer can only interfere with a limited number of frequency bands, a subset of the unicast transmissions are interference-free, thus propagating broadcast messages.

The problem of FH sequence design is mapped to a 1-factorization problem in complete graphs. While a broad class of scheduling algorithms are known to employ 1-factors (perfect matchings) (e.g., [9], [14], [28], [29], [33], [38]), they are, in general, concerned with unicast communications in a benign setting and require the existence of a coordination channel [9], [14]. TDBS is specifically designed to facilitate broadcasting in the presence of jammers and in the absence of a coordination channel. We further consider the problem of updating the FH sequences of existing nodes when the broadcast groups are dynamic. This problem is mapped to the construction of a rainbow path of fixed size in proper edge-colored complete graphs. Note that TDBS is not meant as a permanent replacement of the conventional broadcast mechanism. Broadcasting on a common frequency band achieves the optimal communication efficiency in

A preliminary version of this paper was presented at the ACM WiSec 2011 Conference (full paper)*, June 2011 [18].*

the absence of any jammer. TDBS is designed as an emergency mechanism for temporarily restoring communications until the jammer is physically removed.

Paper Organization: The remainder of the paper is organized as follows. In Section 2, we state the system and adversarial model assumptions. In Section 3 we present an overview of TDBS. Section 4 describes TDBS for single-hop networks. The mechanisms for updating the FH sequences in dynamic broadcast groups are presented in Section 5. In Section 6, we extend the TDBS operation to multi-hop networks. The security and performance of TDBS are evaluated in Section 7. In Section 8, we present related work, and in Section 9, we conclude the paper.

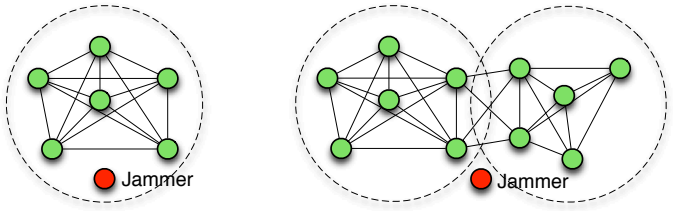
2 System and Adversarial Models

Network Model: We consider two network topology models. In the topology of Fig. 1(a), nodes form a single-hop broadcast group. This topology is typical in wireless LANs, personal area networks, and in military scenarios, in which mobile coalitions move in a team-coordinated fashion. In Fig. 1(b), we consider a static multi-hop network connected in ad hoc mode. To make TDBS scalable with the network size, we assume that the network is partitioned to clusters, which form cliques [34]. Broadcast transmissions in a multi-hop scenario may be limited to one cluster, or propagate to other clusters.

System Model: Nodes communicate over a set $\mathcal{C} = \{f_1, \ldots, f_K\}$ of K non-overlapping frequency bands. Each node is equipped with a single half-duplex transceiver. All nodes are synchronized to a time-slotted system. Nodes equipped with a GPS radio independently synchronize with the clock of GPS satellites. If the adversary jams GPS signals, nodes can maintain synchronization using the jamming-resistant synchronization protocols developed in [12]. These protocols fit well with the TDBS network model because they provide secure synchronization for pairwise, group and multi-hop communications in the presence of inside jammers.

Besides clock synchronization, nodes achieve frame synchronization using standard PHY-layer frame detection methods for FHSS systems [23]. Specifically, a receiver uses preamble correlation techniques to detect the start of a frame. To facilitate the frame synchronization process, long messages are fragmented to smaller frames that fit within a single slot. The frame length is a function of PHY-layer parameters such as the slot duration, the bandwidth allocated to each frequency band, and the modulation order.

Nodes are initialized by a trusted central authority (CA) before deployment. For every node v_i, the CA generates a public/private key pair $< pk_i, sk_i >$. Node v_i is preloaded with the CA's public key pk_{CA} and its own secret key sk_i. To communicate message m to v_i, the CA encrypts $m||sn_i$ with pk_i and signs $(id_i||m||sn_i)$ with its private key sk_{CA}. Here, id_i is v_i's unique, sn_i is a random sequence number that is incremented by one with every message sent to v_i, and $||$ denotes message concatenation. Upon reception of an encrypted message from the CA, node v_i decrypts it using sk_i and verifies the CA's signature using pk_{CA}. It then verifies the freshness of m by checking that sn_i is larger than the previously stored value. Node v_i updates sn_i to the new value. Without access to the preloaded cryptographic keys, the adversary cannot impersonate the CA. Similar security properties could be achieved by adopting a symmetric key cryptosystem. We emphasize that the cryptographic keys are solely used to update the FH sequences following the deployment of new nodes. These keys are not used for any other communications. Hence, it is not necessary to refresh them frequently.



(a) (b)

Fig. 1: (a) A single-hop broadcast group, (b) a clustered multi-hop broadcast group.

Adversary Model: The goal of the adversary is to prevent the sender(s) from communicating with all, or a subset of the intended receivers. For this purpose, the adversary deploys a set of jamming devices capable of collectively jamming any J frequency bands. The jamming devices can switch between frequency bands on a per-slot basis. Note that with dedicated hardware, the jammer may be able to hop at a much higher rate than that of regular nodes. However, the jammer's hopping rate is limited by the time that he has to remain on a particular band (dwell time) to corrupt a sufficient number of bits from the targeted packet(s). Taking into account the interleaving and error correction functions, this time can represent a significant portion of the slot duration [22].

The jammer could target the frame synchronization process by jamming the preamble of a transmitted frame. In most PHY-layer standards, frame detection is based on the signal cross-correlation between the received signal and the known preamble and does not require preamble decoding. The signal cross-correlation has been shown to have high immunity to interference [13]. For all practical purposes, frame transmissions over any of the jammed frequency bands are assumed to be irrecoverably corrupted.

The adversary is capable of physically compromising network devices and recovering stored information including cryptographic keys, PN codes, certificates, etc. Moreover, the adversary is aware of the methods used to protect broadcast transmissions (in our case the specifics of the TDBS).

3 Overview of TDBS

To achieve jamming-resistant communications in the presence of insiders, TDBS implements broadcast as a series of unicasts distributed in frequency and time. The locations of these unicasts, defined by a frequency band/slot pair (f, s), are only partially known to each node. Therefore, a compromised node reveals only the set of locations assigned to it, while the locations of other communications are kept secret.

For this purpose, nodes are divided into pairs scheduled to communicate over randomly selected frequency bands. The pairs and assigned frequency bands change on a per-slot basis, thus realizing a FH system. TDBS differs from traditional FH designs in that: (a) nodes do not follow a common FH sequence, but

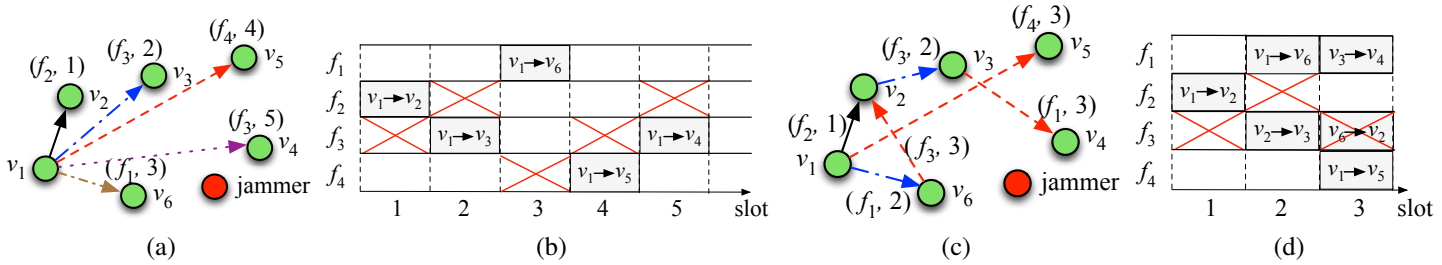


Fig. 2: (a) Operation in SU mode, (b) the timeline of the unicast transmissions of v_1 in SU mode, (c) operation in AB mode, (d) the timeline of the unicast transmissions in AB mode.

hop according to unique hopping patterns and, (b) these patterns are coordinated to reduce the broadcast delay. Moreover, TDBS differs from rendezvous systems for coordinating multi-channel access (e.g., [5], [7], [38]), in that it focuses on the broadcast operation as opposed to rendezvous for unicast transmissions.

TDBS can operate in two modes: the Sequential Unicast mode (SU) and the Assisted Broadcast mode (AB). In the SU mode, the sender sequentially relays information to intended receivers. This more inefficient mode is appropriate when receivers do not have relaying capabilities, or are not trusted to relay broadcast messages. In the AB mode, any node that receives a broadcast message can act as a relay for that message.

Fig. 2 shows an example of the two modes. In Fig. 2(a), node v_1 operates in the SU mode. It sequentially unicasts a broadcast message to nodes $v_2 - v_6$. Fig. 2(b), depicts the timeline of transmissions for Fig. 2(a). The broadcast is completed after five slots. The "x" marks denote the frequency band jammed by the adversary at each time slot. Fig. 2(c), shows the operation in the AB mode. Node v_1 initiates a broadcast in slot 1 by transmitting a message m to v_2. In slot 2, v_1 and v_2 relay m to v_6 and v_3, respectively, using frequency bands f_1 and f_3 in parallel. In slot 3, the broadcast is completed with the relay of m from v_1, v_3, and v_6 to v_5, v_4, and v_2, respectively. The timeline of the transmissions taking place in the AB mode is shown in Fig. 2(d). Observe that in this scenario, the broadcast is completed despite the jamming of the transmission between v_6 and v_2 in slot 3.

The main challenge in TDBS is to design the FH sequences of individual nodes such that the following requirements are met: (a) hopping sequences are pseudo-random, (b) compromise of a subset of nodes (insiders) limits the information leakage relevant to the FH sequences of uncompromised nodes, and (c) every node has the same opportunity to perform a broadcast (fairness). We develop algorithms that satisfy the above requirements. We first illustrate our algorithms for single-hop topologies and then extend our results to multi-hop topologies.

4 TDBS for Single-hop Topologies

The problem of distributing unicast transmissions in frequency and time can be viewed as a link scheduling problem under the node-exclusive interference model. A large body of literature models this type of scheduling after various instances of the matching problem in general graphs [9], [14], [28], [29], [33]. However, prior methods are not immediately applicable to our setup. In link scheduling problems, the goal is to maximize the aggregate network throughput, realized as the sum of individual traffic flows. We are concerned with the dissemination of one message to all members of the broadcast group over unpredictable frequency band/slot locations. This property is not necessarily satisfied by maximum throughput centralized designs [33]. Moreover, decentralized solutions require the exchange of coordination messages over a commonly-agreed channel [9], [14]. Clearly, such a channel can be targeted by an inside jammer.

To ensure the broadcast property, we map the problem of constructing FH sequences to the problem of constructing 1-factorizations in complete graphs. 1-factorizations realize a series of perfect matchings (1-factors), which span the edges of a complete graph [36]. We first present relevant preliminaries from graph theory. Interested readers are referred to [36] for an in-depth treatise of the 1-factorization problem.

4.1 Definitions and Useful Theorems

Definition 1: **Complete graph:** A graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, with vertex set $\mathcal{V}$ and edge set $\mathcal{E}$ is complete if each vertex pair is connected by an edge. We denote such a graph by K_{2n}, where $|\mathcal{V}| = 2n$.

Definition 2: **1-factor:** A 1-factor or a perfect matching F of a graph $\mathcal{G}$ is a subset of $\mathcal{E}$ that partitions $\mathcal{V}$, i.e., F is a set of pairwise disjoint edges of $\mathcal{G}$ that spans $\mathcal{V}$.

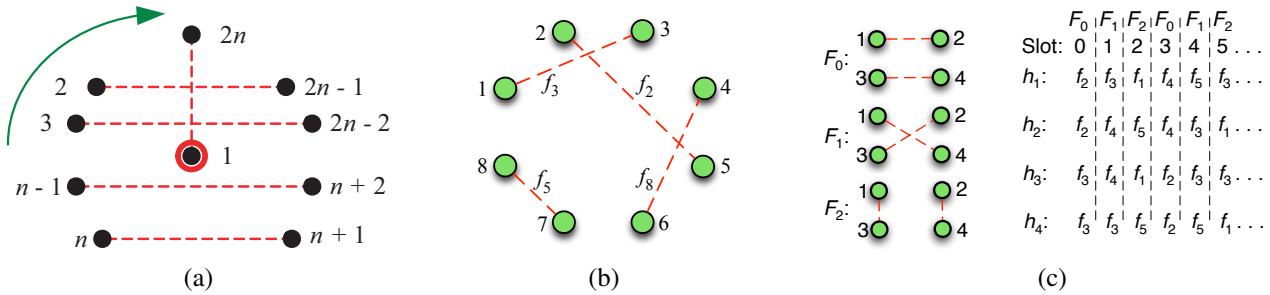
Definition 3: **1-factorization:** A 1-factorization $\mathcal{F}_{2n} = \{F_0, F_1, \ldots, F_{2n-2}\}$ of a graph $\mathcal{G}$ is a partition of its edge set $\mathcal{E}$ to $(2n-1)$ 1-factors.

Theorem 1: **1-factorization of K_{2n}:** A complete graph K_{2n} is 1-factorable [36].

Construction of $\mathcal{F}_{2n}$: 1-factorizations of K_{2n} can be systematically constructed using well-known algorithms (e.g., [11], [21], [35], [36]). A simple method for constructing $\mathcal{F}_{2n}$ is to select a "starter" 1-factor and apply a shift-and-rotate operation [36]. This method is illustrated in Fig. 3(a). $\mathcal{F}_{2n}$ is initialized by 1-factor F_0. Node 1 remains fixed. To obtain F_i, nodes in the perimeter are rotated clockwise by i steps.

4.2 Mapping to the 1-factorization Problem

We map the problem of constructing FH sequences for TDBS to the problem of generating 1-factorizations in complete graphs. In our mapping, the vertex set $\mathcal{V}$ of K_{2n} represents the broadcast group of size $2n$ and an edge $(u, v) \in \mathcal{E}$ represents a unicast between u and v. A 1-factor corresponds to partitioning the broadcast group to n communicating pairs. These pairs are scheduled to rendezvous in parallel over non-overlapping frequency bands. A 1-factorization $\mathcal{F}_{2n}$ partitions $\mathcal{E}$ to $(2n-1)$ disjoint 1-factors, in which each edge appears exactly once. In a schedule constructed



(a) (b) (c)

Fig. 3: (a) Algorithm for constructing a 1-factorization $\mathcal{F} = \{F_0, \ldots, F_{2n-2}\}$. To obtain a 1-factor F_i, every node is rotated by i positions clockwise. Node 1 remains fixed, (b) mapping of a 1-factor to parallel unicast transmissions. Paired nodes communicate concurrently on separate frequency bands, (c) construction of FH sequences for the SU mode for a group of four nodes.

Algorithm 1 TDBS-SU: Sequential Unicast Mode

```
Generate F_{2n} of K_{2n}
repeat
for i = 0 to (2n − 2) do
  for j = 1 to ⌈n/K⌉ do
    π = rand(perm(C))
    for w = 1 to min{n, K} do
      h_{F((j−1)K+w,1)} = h_{F((j−1)K+w,2)} = π(w)
    end for
  end for
end for
end repeat
```

according to $\mathcal{F}_{2n}$, every node rendezvous with all remaining $(2n-1)$ nodes. An example of this mapping is shown in Fig. 3(b). A group of eight nodes is partitioned into four pairs, which are scheduled to communicate over four frequency bands. Fig. 3(c) shows a feasible set of FH sequences h_j for nodes, $j = 1, \ldots, 4$, based on the 1-factorization of K_4. We now present algorithms for constructing FH sequences.

4.3 TDBS-SU: Sequential Unicast Mode

In the SU mode, a sender sequentially unicasts the broadcast message to $(2n-1)$ receivers. Let P_K be the possible permutation set of $\mathcal{C}$. The FH sequences are constructed as follows:

Step 1: Construct a 1-factorization $\mathcal{F}_{2n}$ of K_{2n}, where $\mathcal{F}_{2n} = \{F_0, F_1, \ldots, F_{2n-2}\}$.
Step 2: For all $F_i \in \mathcal{F}_{2n}$, repeat Steps 3–5.
Step 3: Randomly select permutation $\pi \in P_K$ with replacement.
Step 4: Assign frequency bands in π to the first $\min\{n, K\}$ unassigned pairs in F_i.
Step 5: Repeat Steps 3 and 4 until all pairs in F_i are assigned a frequency band.
Step 6: Repeat Steps 1-5.

The pseudocode for TDBS-SU is shown in Algorithm 1. We emphasize that although the rendezvous schedule repeats every $(2n-2)$ slots, the FH sequences assigned to each node have a much longer period. This is because the permutation π that indicates the channel assignment per slot, is randomly selected with every iteration. As a result, pairs rendezvous every $(2n-2)$ slots on random permutations of $\mathcal{C}$.

In Fig. 3(c), we apply Algorithm 1 to a four-node group, when $\mathcal{C} = \{f_1, \ldots, f_5\}$, $(K = 5)$. Because $K \geq n$, the n pairs corresponding to a 1-factor communicate in parallel in one slot. In slot 0, pairs communicate according to F_0. The random permutation for F_0 is $\pi = \{f_2, f_3, f_5, f_1, f_4\}$. Pair $(1,2)$ is assigned $\pi(1) = f_2$ and pair $(3,4)$ is assigned $\pi(2) = f_3$. The process is repeated for 1-factors F_1 and F_2. When node pairs $(1,2)$ and $(3,4)$ rendezvous again in slot 3, a new permutation $\pi = \{f_4, f_2, f_1, f_3, f_5\}$ is randomly selected from P_5 to perform the channel assignment. Note that it is not necessary for $K \geq n$. When $K < n$, parallel transmissions corresponding to one 1-factor are distributed over multiple slots. We now show that Algorithm 1 constructs uniformly distributed FH sequences.

Proposition 1: The FH sequences constructed by Algorithm 1 are uniformly distributed.

Proof: The proof is provided in Appendix 1. □

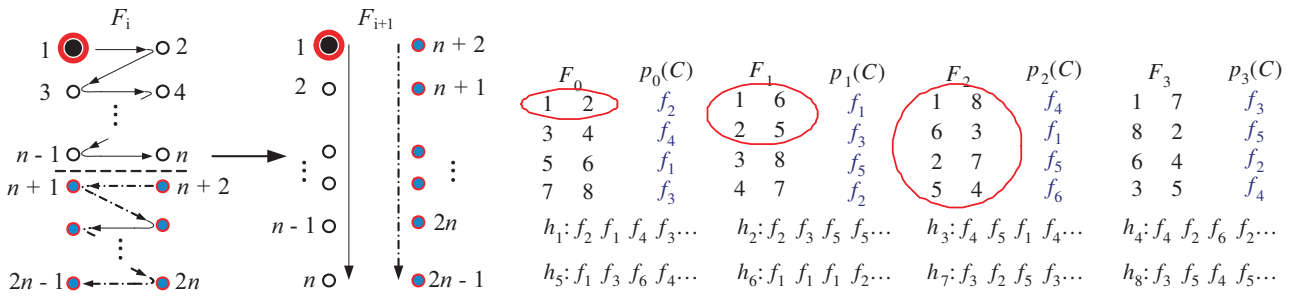
4.4 TDBS-AB: Assisted Broadcast Mode

In the AB mode, any node that has already received a broadcast message operates as a broadcast relay. To construct FH hopping sequences for the AB mode, the 1-factors F_i are selected and arranged such that the *number of nodes that can relay a broadcast transmission at each 1-factor is maximized*. This property reduces the broadcast delay, while increasing the resilience to jamming. We first define the *relay set*.

Definition 4: **Relay set** R_j^i : The relay set R_j^i of a node v_j that originated a message m is defined as the set of nodes that can relay m in 1-factor F_i.

The goal of our FH sequence construction algorithm is to maximize the size of the relay set R_j^i, for every node v_j and in every 1-factor F_i. Note that in the AB mode, it is not necessary that the series of 1-factors form a 1-factorization, because nodes can receive a broadcast transmission indirectly via relay nodes. The FH hopping sequences are constructed as follows.

Step 1: Obtain an arbitrary 1-factor F_0 of K_{2n}. Set $i = 0$.
Step 2: Randomly select a permutation $\pi \in P_K$.
Step 3: Assign frequency bands in π to the first $\min\{n, K\}$ unassigned pairs in F_i.
Step 4: Repeat Steps 2 and 3 until all pairs in F_i are assigned a frequency band.
Step 5: Construct 1-factor F_{i+1} according to the *splitting algorithm*. Set $i = i + 1$.
Step 6: Repeat Steps 2 – 5.



(a) (b)

Fig. 4: (a) Splitting algorithm used to obtain the 1-factor F_{i+1} from the 1-factor F_i. The first n nodes of F_i are obtained in a "zigzag" fashion and are placed on the first column of F_{i+1}. The last n nodes of F_i are obtained in an "inverse zigzag" fashion and are placed in the second column of F_{i+1}, (b) the first four 1-factors for a group of eight nodes and the corresponding hopping sequences.

Algorithm 2 TDBS-AB: Assisted Broadcast Mode

Generate random F_0 of K_{2n}
initialize $i = 0$
repeat
for $j = 1$ to $\lceil \frac{n}{K} \rceil$ **do**
  $\pi = rand(\text{perm}(\mathcal{C}))$
  for $w = 1$ to $\min\{n, K\}$ **do**
    $h_{F_i((j-1)K+w,1)} = h_{F_i((j-1)K+w,2)} = \pi(w)$
  end for
end for
$F_{i+1} = \texttt{split}(F_i)$
$i++$
end repeat

Algorithm 3 Splitting Algorithm `split`

$F_{i+1}(1,1) = F_i(1,1)$
if n even **then**
  $F_{i+1}(1,2) = F_i(\frac{n}{2}+1, 2)$
else
  $F_{i+1}(1,2) = F_i(\lceil \frac{n}{2} \rceil, 2)$
end if
for $j = 2$ to n **do**
  $F_{i+1}(j,1) = F_i(\lceil \frac{j}{2} \rceil, 2)$, if j even
  $F_{i+1}(j,1) = F_i(\lceil \frac{j}{2} \rceil, 1)$, if j odd
  if n even **then**
    $F_{i+1}(j,2) = F_i(\lceil \frac{n+j}{2} \rceil, 1)$, if j even
    $F_{i+1}(j,2) = F_i(\lceil (\frac{n+j}{2} \rceil, 2)$, if j odd
  else
    $F_{i+1}(j,2) = F_i(\lceil \frac{n+j}{2} \rceil, 2)$, if j even
    $F_{i+1}(j,2) = F_i(\lceil (\frac{n+j}{2} \rceil, 1)$, if j odd
  end if
end for

The pseudocode for the TDBS-AB is shown in Algorithm 2. The pseudocode for the splitting algorithm is shown in Algorithm 3 and is illustrated in Fig. 4(a). Nodes that rendezvous according to 1-factor F_i, are split (placed in adjacent rows) in F_{i+1}. The propagation of this property in subsequent 1-factors minimizes the broadcast delay by maximizing the size of the relay set R_j^i for any v_j and for every i (see Section 7 for a formal proof).

Fig. 4(a) shows a conceptual diagram of the splitting algorithm. The first n nodes encountered using a "zigzag" operation on F_i, are placed on the left column of F_{i+1} in a sequential order. The remaining n nodes are placed on the right column of F_{i+1} following a similar "zigzag" operation, but starting from $n+2$. Fig. 4(b) shows the application of the splitting algorithm on the first four 1-factors of a broadcast group of size 8. It also shows the resulting FH sequences. Node 1 transmits a message m at F_0. The circles mark the nodes that have received m at the end of each 1-factor. At the end of F_0, m is received by 1 and 2, at the end of F_1, m is received by 1, 2, 3, and 4, etc. In fact, one can verify that a broadcast initiated by any node in F_0 is completed by $F_{\log_2(8)-1} = F_2$. In Section 7, we prove that this property holds for any broadcast initiated at any time slot.

5 Accommodating Dynamic Groups

In this section, we develop methods for updating the FH schedule when the broadcast group is dynamic. Specifically, we design a node addition mechanism that minimizes the changes in the FH schedule of existing nodes. Our mechanism extends the FH schedule based on $\mathcal{F}_{2n}$ to a FH schedule based on $\mathcal{F}_{2n+2}$ by constructing rainbow paths in complete graphs. We present a protocol for communicating the modified FH schedule to existing nodes using the newly deployed node, without direct connection to the CA. For node deletion, we show how the remaining nodes can modify their original FH schedule to an optimal schedule for $2n-2$ nodes. This modification is performed individually, without any information exchange. Finally, we show that the addition and deletion mechanisms preserve the TDBS security properties.

5.1 Node Addition

The simplest method for accommodating dynamic broadcast groups is to re-compute all FH sequences at the CA and directly assign them to existing nodes. However, in several deployment scenarios, on "always online" CA may not be available. To address these scenarios, we design a node addition mechanism that satisfies the following goals: (a) existing nodes are updated without a direct connection to the CA, (b) the number of FH schedule changes is minimized, and after the FH schedule update the performance and security properties of TDBS are preserved. Node addition is divided to the extension of $\mathcal{F}_{2n}$ to $\mathcal{F}_{2n+2}$ and the channel assignment for the new node pairs in $\mathcal{F}_{2n+2}$.

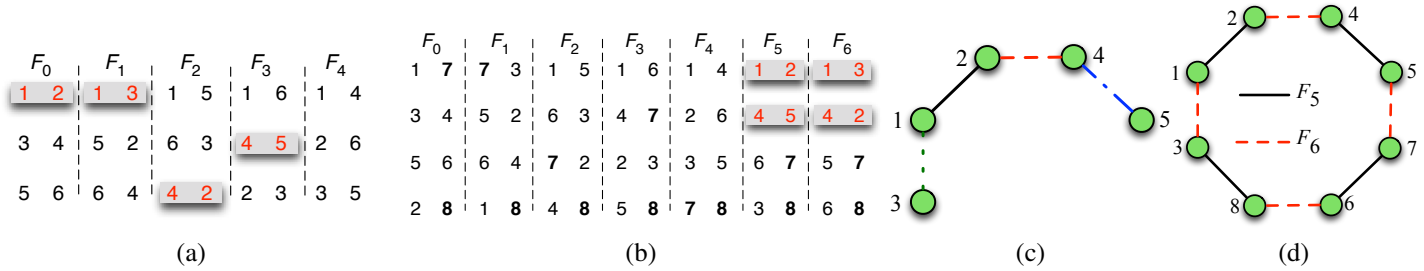


Fig. 5: (a) The five 1-factors of $\mathcal{F}_6$. Shaded pairs are split pairs, (b) extension of $\mathcal{F}_6$ to $\mathcal{F}_8$. Shaded pairs are split pairs. Non-shaded pairs of the last two factors are deferred pairs, (c) an R-path of length four in K_6, (d) the augmented R-path.

5.1.1 Construction of $\mathcal{F}_{2n+2}$

Consider a broadcast group of $2n$ nodes (extending an odd size group is trivially treated by adding the new node in the position of the dummy one). Addition of one node requires the migration from a rendezvous schedule constructed under $\mathcal{F}_{2n}$ to a schedule constructed under $\mathcal{F}_{2n+2}$. Before we describe our node addition mechanism, we state the following useful definitions.

Definition 5: **Split pair:** A node pair scheduled to rendezvous at $F_i \in \mathcal{F}_{2n}$ is said to be a *split pair* if it does not rendezvous at the same 1-factor $F_i \in \mathcal{F}_{2n+2}$.

Definition 6: **Deferred pair:** A node pair consisting of one existing and one newly added node is said to be a *deferred pair* if it is not scheduled during the first $(2n-1)$ 1-factors of $\mathcal{F}_{2n+2}$.

As an example, consider the extension of the $\mathcal{F}_4$ shown in Fig. 5(a) to the $\mathcal{F}_6$ shown in Fig. 5(b). Pairs $(1,2)$, $(1,3)$, $(4,2)$, and $(4,5)$ of $\mathcal{F}_6$ are split pairs because they do not rendezvous on 1-factors F_0, F_1, F_2, and F_3 of $\mathcal{F}_8$, respectively. Pairs $(6,7)$, $(5,7)$, $(3,8)$, and $(6,8)$ of $\mathcal{F}_8$ are deferred pairs because they consist of one existing node and one newly added node and do not rendezvous during the first five 1-factors of $\mathcal{F}_8$. The number of split and deferred pairs are related to the number of FH schedule changes that must occur for the existing nodes as follows.

Proposition 2: The minimum number of FH schedule changes for existing broadcast group members when migrating from $\mathcal{F}_{2n}$ to $\mathcal{F}_{2n+2}$ due to member addition is $(8n-4)$. This minimum is achieved when $(2n-2)$ pairs are split and four pairs are deferred.

Proof: The proof is provided in Appendix 2. □

Based on Proposition 2, minimizing the FH sequence schedule changes for the existing nodes reduces to the problem of constructing $\mathcal{F}_{2n+2}$ from $\mathcal{F}_{2n}$, such that $(2n-2)$ pairs are split. We map the latter problem to the problem of constructing rainbow paths in complete graphs. First, we introduce relevant definitions.

Definition 7: **Proper edge coloring:** A proper edge coloring of $\mathcal{G}(\mathcal{V},\mathcal{E})$ is an assignment of colors to the edges in $\mathcal{E}$, such that all edges of the same color form a matching [24].

Definition 8: **Rainbow path (or R-path):** For a proper edge-colored graph $\mathcal{G}(\mathcal{V},\mathcal{E})$, a path P_R is called a rainbow path if all edges in P_R have distinct colors.

In our mapping, the nodes incident to an edge in P_R represent split pairs. According to Proposition 2, the minimum number of such pairs is $(2n-2)$. Therefore, by finding an R-path of length $(2n-2)$ on K_{2n}, we identify the pairs of $\mathcal{F}_{2n}$ that must be split. As an example, Fig. 5(c) shows an R-path of length $(2n-2)=4$ on K_6. The corresponding split pairs for extending $\mathcal{F}_6$ to $\mathcal{F}_8$ are shown in Fig. 5(a) (shaded boxes).

An R-path of length $(2n-2)$ is not guaranteed to exist for arbitrary proper edge-colored complete graphs [15]. Andersen has conjectured that such R-paths do exist, but a formal proof remains elusive [3]. To construct R-paths of length $(2n-2)$, we employ the heuristic algorithm proposed in [15]. While this heuristic is not guaranteed to always find an appropriate R-path for all graph sizes, our simulations show that a solution is always found for complete graphs of up to 80 vertices, which is sufficient for most target applications. After P_R is found, it is extended by adding the edges formed by deferred pairs. These edges are added according to the following proposition.

Proposition 3: Let $P_R = \{v_1, \ldots, v_{2n-1}\}$ denote an R-path of length $(2n-2)$ on K_{2n}. Let also $v_n \in K_{2n}$ denote the only node that is not part of P_R, and v_{2n+1}, v_{2n+2} denote the newly added nodes when extending the schedule from $\mathcal{F}_{2n}$ to $\mathcal{F}_{2n+2}$. The deferred pairs are formed by (v_n, v_{2n+1}), (v_n, v_{2n+2}), (v_1, v_{2n+1}), and (v_{2n-1}, v_{2n+2}).

Proof: The proof is provided in Appendix 3. □

The cycle formed by the augmented path consists of split and deferred pairs that must be scheduled during 1-factors F_{2n-1} and F_{2n} of $\mathcal{F}_{2n+2}$. To construct a conflict-free schedule for F_{2n-1} and F_{2n}, we traverse the cycle starting from any vertex. Each pair of nodes corresponding to the vertices incident to a traversed edge is assigned to either F_{2n-1} or F_{2n} in an alternating fashion. As an example, Fig. 5(d) shows one cycle for extending $\mathcal{F}_6$ to $\mathcal{F}_8$. We traverse the cycle starting from node 1. Pairs $(1,2)$, $(4,5)$, $(7,6)$, and $(3,8)$ are assigned to F_5 (solid lines), while pairs $(2,4)$, $(5,7)$, $(6,8)$, and $(3,1)$ are assigned to F_6 (dotted lines). The resulting rendezvous schedule shown in Fig. 5(b) is conflict-free. Note that knowledge of the rendezvous schedule by all broadcast group members does not compromise the secrecy of the FH sequences, since the channel assignment for the rendezvous pairs remains secret.

5.1.2 Channel Assignment in $\mathcal{F}_{2n+2}$

For the channel assignment, the CA computes the updated FH sequences of existing and new broadcast group members. To accommodate schedule changes while preserving the FH sequence secrecy, we introduce a training phase in which a newly added node communicates FH sequence updates to the existing nodes.

FH sequence update phase: In this phase, the CA updates the FH sequences of existing nodes to accommodate a conflict-free rendezvous schedule according to $\mathcal{F}_{2n+2}$. This update is performed as follows:

Step 1: The CA computes $\mathcal{F}_{2n+2}$ as described in Section 5.1.1.
Step 2: For $F_i \in \mathcal{F}_{2n+2}, i \leq 2n-2$, a node pair that consists of existing nodes maintains the channel assignment of $F_i \in F_{2n}$. A node pair that consists of one new and one existing node is randomly assigned a channel from $\mathcal{C}$ that does not conflict with the existing assignments for F_i.
Step 3: For each of last two 1-factors of $\mathcal{F}_{2n+2}$, the CA randomly selects a permutation $\pi \in P_K$. Rendezvous pairs corresponding to 1-factors F_{2n-1} and F_{2n}, are assigned frequency bands according to the two random permutation selections for each 1-factor. Steps 2 and 3 are repeated a number of times equal to the length (in 1-factorizations) of the FH sequences.
Step 4: Let h'_j denote the FH sequence changes for node v_j. The CA encrypts $< h'_j || sn_j >$ with pk_j and signs it with sk_{CA} (sn_j denotes a sequence number). The encrypted updates of all existing nodes are preloaded to the newly added node.
Step 5: The CA preloads the new node with a *training* FH sequence th. The training sequence indicates the rendezvous channels of split and deferred pairs according to $\mathcal{F}_{2n}$.

For example, consider the extension of the broadcast group from six nodes to eight nodes due to the deployment of node 7. The CA is aware of the FH sequences of nodes 1-6 based on $\mathcal{F}_6$, which is shown in Fig. 5(a). In Step 1, the CA computes the $\mathcal{F}_8$ shown in Fig. 5(b). In Step 2, the channel assignments for pairs that are not split in the first $(2n-1)$ 1-factors remain intact (non-shaded pairs in Fig. 5(a)). Pairs that consist of one new node (7 or 8) and one existing node are randomly assigned a channel such that the schedule for F_0 to F_4 remains conflict-free. For instance, let the channel assignment for pairs (3,4) and (5,6) for slot 0 be f_1 and f_2, respectively. In $\mathcal{F}_8$ and for the same slot, pairs (3,4) and (5,6) still rendezvous on f_1 and f_2, respectively. New rendezvous pairs (1,7) and (2,8) are randomly assigned channels f_5 and f_3, such that the schedule for $F_0 \in \mathcal{F}_8$ remains conflict-free.

In Step 3, the CA generates two random permutations for defining the channel assignment for 1-factors F_5 and F_6. Note that although these 1-factors contain the $(2n-2)$ split pairs of $\mathcal{F}_6$, the channel assignment of those pairs is not retained in the transition from $\mathcal{F}_6$ to $\mathcal{F}_8$. This is because split pairs were scheduled to rendezvous in different 1-factors of $\mathcal{F}_6$. Therefore, it cannot be guaranteed that merging them to the same 1-factor will yield a conflict-free schedule. Steps 2 and 3 are repeated a number of times equal to the length of the FH sequences in 1-factorizations. With every iteration, the permutations selections in Step 3 are randomized.

In Step 4, the CA loads all the FH sequence changes to the new node, which acts as a proxy for transferring channel assignments to existing nodes. FH sequence changes are encrypted using the public key of the destination node and are signed by the CA. This is necessary for preventing the new node from knowing the FH sequence changes for existing nodes and, therefore, protecting against node compromise. Moreover, the attached CA signature prevents the new node (or any other entity) from fabricated fake FH sequence changes. Finally, the sequence number sn_j prevents the replay of old FH sequence changes. In Step 5, the CA preloads the new node with th. For instance, in the extension from $\mathcal{F}_6$ to $\mathcal{F}_8$ depicted in Fig. 5, node 7 is preloaded with $\{f_0^{(1,2)}, f_1^{(1,3)}, f_2^{(4,2)}, f_3^{(4,5)}, f_4^{(2,6)}, f_5^{(1,2)}, \ldots\}$, where $f_i^{(u,v)}$ denotes the rendezvous channel of pair (u,v) on slot i.

Training phase: During the training phase, the newly added node communicates the FH sequence changes to the existing nodes. The training phase is as follows:

Step 1: The new node hops according to th. At every slot i, the new node communicates with one existing node on channel $th(i)$. The updated node acknowledges the receipt of the schedule change.
Step 2: Updated nodes hop according to the new FH sequences and remain in receiving mode.
Step 3: The new node continues to hop according to th until all existing nodes are updated.
Step 4: Upon the completion of all updates, the new node switches to its own FH sequence, which conforms to the new schedule (according to $\mathcal{F}_{2n+2}$).
Step 5: The new node notifies all existing nodes (who remain in receiving mode) of the training phase termination.

Following with the example of Fig. 5, node 7 hops according to the training sequence $\{f_0^{(1,2)}, f_1^{(1,3)}, f_2^{(4,2)}, f_3^{(4,5)}, f_4^{(2,6)}, f_5^{(1,2)}, \ldots\}$. Node 7 sequentially conveys FH sequence changes to nodes 1, 3, 4, 5, 6, and 2. Once all FH sequence updates are completed, node 7 switches to its own FH schedule and notifies all existing nodes of the training phase termination.

5.2 Node Deletion

When nodes leave the broadcast group, a process reverse to node addition is followed. Consider the reduction of the broadcast group from $(2n+2)$ to $2n$ members (deletion of a single node from an even size broadcast group does not cause any schedule change. A dummy node replaces the departing node in the rendezvous schedule). The FH schedule constructed under $\mathcal{F}_{2n+2}$ has to be migrated to a schedule constructed under $\mathcal{F}_{2n}$. We first provide relevant definitions.

Definition 9: **Peerless node:** A node is said to be peerless, if it rendezvous with a departing node at a given 1-factor.
Definition 10: **Merged pair:** A pair is said to be merged, if it consists of two peerless nodes of the same 1-factor.

The steps for constructing $\mathcal{F}_{2n}$, based on $\mathcal{F}_{2n+2}$ are as follows:

Step 1: For every 1-factor of $\mathcal{F}_{2n+2}$, peerless nodes are merged to form merged pairs.
Step 2: The two 1-factors of $\mathcal{F}_{2n+2}$ that consist solely of merged pairs are deleted from $\mathcal{F}_{2n+2}$ to form $\mathcal{F}_{2n}$.
Step 3: A merged pair (u,v), consisting of two peerless nodes of $F_i \in F_{2n+2}$, selects the channel assigned to either u or v in $F_i \in F_{2n+2}$, to rendezvous at the corresponding 1-factor of $\mathcal{F}_{2n}$.

For example, consider the rendezvous schedule for a broadcast group of size eight shown in Fig. 6(a). Assume that nodes 3 and 4 leave the broadcast group. In Step 1, peerless nodes form

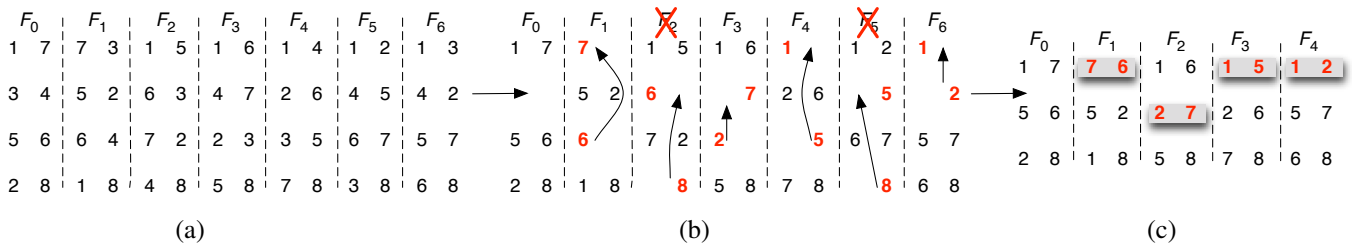


Fig. 6: (a) 1-factorization $\mathcal{F}_8$, (b) peerless nodes are joined to form merged pairs. 1-factors F_2 and F_5 which consist only of merged pairs are deleted to generate $\mathcal{F}_6$, (c) highlighted pairs indicate the channel assignment changes that must occur due to member deletion. Merged pairs select one of the channel assigned to them during the respective 1-factors of $\mathcal{F}_8$ before the member deletion.

the merged pairs indicated in Fig. 6(b). In Step 2, 1-factors $F_2 \in \mathcal{F}_8$ and $F_5 \in \mathcal{F}_8$, which consist solely of merged pairs that already rendezvous in other 1-factors, are deleted. The resulting 1-factorization $\mathcal{F}_6$ is shown in Fig. 6(c). Since, every node is aware of the same 1-factorization $\mathcal{F}_{2n+2}$, it is straightforward to show that each node can independently compute $\mathcal{F}_6$ via Steps 1 and 2. In Step 3, merged pairs use their rendezvous channels according to $\mathcal{F}_8$ to agree on the rendezvous channels in the new 1-factorization. For example, consider the merged pair $(7, 6)$ that is created at 1-factor $F_1 \in \mathcal{F}_8$. After migrating from $\mathcal{F}_8$ to $\mathcal{F}_6$, nodes 6 and 7 select one channel from $\{f_1^{(7,3)}, f_1^{(6,4)}\}$ as their rendezvous channel for $F_1 \in \mathcal{F}_6$. The channel selection can be automated by a deterministic rule (e.g., select the channel assigned to the peerless node with the lowest id). This process is repeated for all slots that correspond to F_1.

Note that this assignment always leads to a conflict-free schedule for $\mathcal{F}_{2n}$ because the schedule based on $\mathcal{F}_{2n+2}$ is conflict-free. Hence, the channels assigned to peerless nodes within the same 1-factor are not assigned to any other node pair. The channel assignment for merged peerless nodes is only known to the those nodes. Hence, the secrecy of the assignment is preserved (we assume that the departing nodes are no longer part of the system and therefore, cannot be compromised).

6 TDBS in Multi-hop Networks

In this section, we extend the operation of TDBS to multi-hop networks. In this scenario, the FH sequence design can be viewed as a global scheduling problem. While several distributed methods have been proposed for distributed scheduling (e.g., [9], [14]), we note that these methods require coordination via a common channel. However, such a channel can be blocked by an inside jammer. We, therefore, develop a scalable solution that does not rely on the existence of a common channel. We partition the network into clusters where each cluster forms a clique [34]. We then divide the broadcast operation into two phases; an intra-cluster phase and an inter-cluster phase. During the intra-cluster phase, communication is limited within each cluster. In the inter-cluster phase, messages are exchanged between border nodes of adjacent clusters. The two phases are interleaved in time.

6.1 Intra-cluster Phase

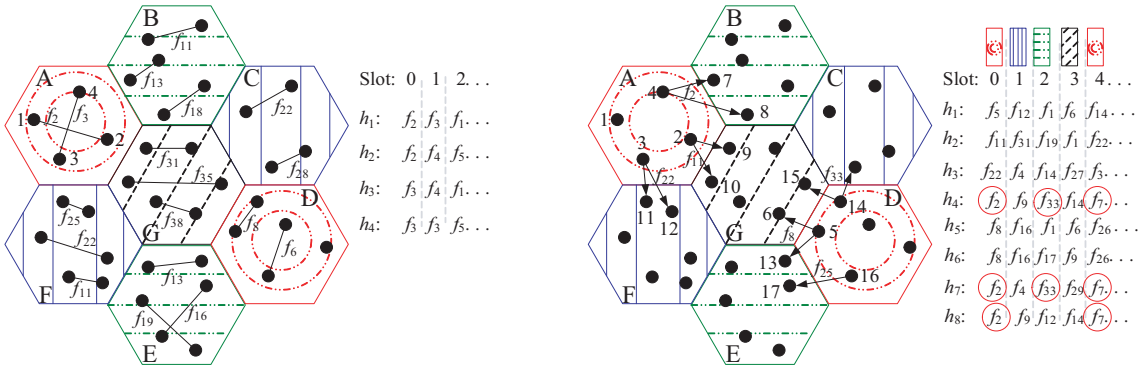
In the intra-cluster phase, a broadcast message propagates to all cluster nodes. Because the nodes of a cluster form a clique, the SU or the AB mode of TDBS can be employed for broadcast. To avoid interference between adjacent clusters, the set of available frequency bands $\mathcal{C}$ is partitioned to four subsets, which are assigned to clusters according to the four-color theorem [4]. One such assignment is shown in Fig. 7(a). The shading pattern denotes a separate set of frequency bands assigned to each cluster. In this example, ten channels are assigned per cluster. The steps for deriving FH sequences for this phase are as follows.

Step 1: Color each cluster using the four-color theorem.
Step 2: Assign a subset of channels to each cluster according to its color.
Step 3: For each cluster, construct FH sequences using either the SU mode or the AB mode.

6.2 Inter-cluster Phase

In the inter-cluster phase, border nodes relay broadcast messages beyond the origin cluster. To do so while avoiding schedule conflicts, we exploit the cluster coloring produced by the four-color theorem. During this phase, every time slot is marked with one of the four colors, indicating the clusters that are allowed to transmit on that slot. As an example, in Fig. 7(b), clusters A and D are scheduled to transmit during slots $i : i \mod 4 = 0$, clusters C and F are scheduled to transmit on slots $i : i \mod 4 = 1$, clusters B and E are scheduled to transmit on slots $i : i \mod 4 = 2$, and cluster G is scheduled to transmit on slots $i : i \mod 4 = 3$, (slot numbers indicate the assignment before the interleaving with the intra-cluster phase). After the slot coloring, the FH sequences are generated as follows.

Step 1: For each cluster x, find the nodes in x bordering adjacent clusters. Place these nodes to a set $\mathcal{A}$.
Step 2: For each $v_i \in \mathcal{A}$, find the neighbors of v_i in adjacent clusters and assign them to v_i. If a neighbor is common to more than one nodes in x, assign it to the node with the fewer neighbors. Break ties arbitrarily. Merge nodes assigned to the same v_i to a single vertex and place vertices to set $\mathcal{B}$. Create a bipartite graph $\mathcal{G}(\mathcal{A} \cup \mathcal{B}, \mathcal{E})$, where an edge (v_i, v_j) exists if nodes corresponding to $v_j \in \mathcal{B}$ are assigned to $v_i \in \mathcal{A}$. By construction, graph $\mathcal{G}$ forms a 1-factor F_x.
Step 3: For each slot colored with x's color, obtain a random permutation $\pi \in P_K$.
Step 4: Assign frequency bands in π to the first $\min\{n, K\}$ unassigned pairs of F_x.
Step 5: Repeat Steps 3 and 4 until all pairs in F_x are assigned a frequency band.
Step 6: Repeat Steps 1-5, until all clusters are processed.



(a) (b)

Fig. 7: (a) The intra-cluster phase, (b) the inter-cluster phase.

The inter-cluster phase is illustrated in Fig. 7(b). Clusters A and D are scheduled to broadcast during slots $0, 4, 8, \ldots$ of the inter-cluster phase. Nodes 2, 3, and 4 belong to set $\mathcal{A}$ of cluster A, as they are border nodes. Nodes 9 and 10 of cluster G are assigned to node 2 and are merged to a vertex in $\mathcal{B}$, nodes 11 and 12 of cluster F are assigned to node 3 and are merged to a second vertex in $\mathcal{B}$, and nodes 7 and 8 of cluster B are assigned to node 4 and are merged to a third vertex in $\mathcal{B}$. In slot 0, the communicating pairs are $(2-9, 10)$ $(3-11, 12)$ and $(4-7, 8)$, and are assigned frequency bands f_{11}, f_{22}, and f_2, respectively. Similarly, for cluster D and slot 0, the communicating pairs are $(5-6, 13)$ $(14, 15)$ and $(16, 17)$, and are assigned frequency bands f_8, f_{33}, and f_{25}, respectively. Note that during the inter-cluster phase, all channels in $\mathcal{C}$ are available for assignment to the communications of adjacent pairs of nodes. The intra-cluster and inter-cluster slots are interleaved in the FH design, to allow for both single hop and multi-hop broadcast transmissions.

7 Performance and Security Evaluation

In this section, we evaluate the performance of TDBS by analyzing the broadcast delay, defined as follows.

Definition 11: **Broadcast Delay** D**:** Number of slots required until all broadcast group members have received a copy of the broadcasted message.

The broadcast delay is the inverse measure of the throughput achieved by the TDBS broadcasting operation. In the absence of jammers, the broadcast delay of TDBS is given by the following two propositions.

Proposition 4: The broadcast delay of the TDBS-SU is $D = \lceil \frac{n}{K} \rceil (2n-1)$ slots.

Proof: The proof is provided in Appendix 4. □

Proposition 5: The TDBS-AB mode minimizes the broadcast delay when broadcast is realized as a series of concurrent unicast transmissions. This minimum delay is equal to $D = \lceil \frac{n}{K} \rceil \lceil \log_2(2n) \rceil$ slots.

Proof: The proof is provided in Appendix 5. □

Although TDBS is designed for enabling broadcast in the presence of internal jammers, we can evaluate the per-node throughput in the absence of jamming. By design, a node has the opportunity to access one slot every $\lceil \frac{n}{K} \rceil$ slots. Hence, the per node throughput is equal to $\frac{1}{\lceil \frac{n}{K} \rceil}$, if a node is backlogged with traffic to all other nodes. Alternatively, the per-node throughput can be defined as the inverse of the delay to rendezvous with *a specific* node. For the latter definition, the per-node throughput is equal to the inverse of the delay stated in Proposition 4. This is because two nodes rendezvous every $\lceil \frac{n}{K} \rceil (2n-1)$ slots.

7.1 Security Analysis

We first analyze the resilience of TDBS to external and internal jammers for single-hop networks.

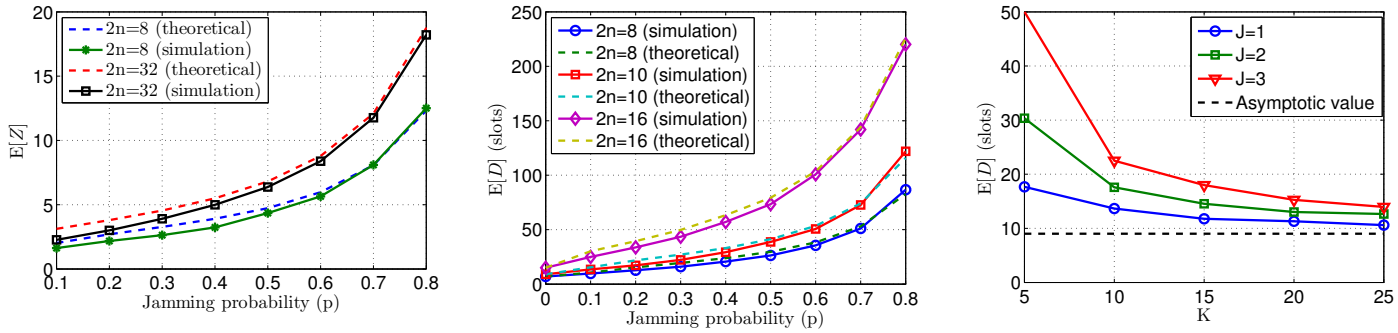
7.1.1 Resilience to External Jammers

Under an external threat model, the FH sequences assigned to various nodes are assumed secret. This is because the FH sequences are based on random permutations of the channel set C, which independently change on a per-slot basis. Hence, within a FH sequence period, knowledge of previous channel hops does not reveal any information about future ones. A jammer acting as an eavesdropper by randomly hopping on various channels would require many FH sequence periods to reconstruct the FH sequence of a node. To do so, the node must be a transmitter at every slot within a FH sequence period (or several periods) and the jammer must overhear every transmission made by the target node.

For the external jammer scenario, we assume that the adversary deploys multiple jamming devices that can jam up to J frequency bands per time slot, with $J < K$. For convenience, we assume $K \geq n$ so that all node pairs corresponding to a 1-factor can communicate in parallel in one time-slot. Our results can be extended to the $K < n$ case in a straightforward manner. Suppose that a jammer attempts to jam the broadcast of a single node v_j. To compute D, we evaluate the average number of 1-factorizations needed to complete the broadcast in the presence of the external jammer, and for each TDBS mode.

Proposition 6: In the presence of an external jammer, the expected number $\mathrm{E}[Z]$ of 1-factorizations needed to complete a broadcast operation in the SU mode is:

$$\mathrm{E}[Z] = (1-p)^{2n-1} + \sum_{i=2}^{\infty} i(1-p^{i-1})^{2n-1} \times \sum_{k=1}^{2n-1} \binom{2n-1}{k} \left(\frac{p^{i-1}(1-p)}{1-p^{i-1}} \right)^k, \quad (1)$$



(a) (b) (c)

Fig. 8: (a) E$[Z]$ as function of the jamming probability p when $K = 20$, (b) E$[D]$ as a function of jamming probability p when $K = 20$, (c) E$[D]$ as a function of K when $2n = 10$.

where $p = \frac{J}{K}$ denotes the jamming probability.

Proof: The proof is provided in Appendix 6. □

We further compared the theoretical value of E$[Z]$ with the simulated one as a function of $p = \frac{J}{K}$. We performed our simulation experiments using MATLAB 2012 [20]. We generated sequences of length $1,000$ according to Algorithm 1. We randomly selected J channels to be jammed per time-slot, out of a total of $K = 20$ channels. To isolate the impact of jamming, we ignored any losses due to the imperfections of the wireless channel. Two nodes scheduled to rendezvous over a jamming-free channel were assumed to successfully complete one packet transmission. If the jammer was present on the rendezvous channel, the packet was assumed to be lost. All results were averaged over 100 runs. In Fig. 8(a), we observe that the simulation values of E$[Z]$ agree with the theoretical ones. As expected, the number of 1-factorization needed to complete a broadcast increases with p.

Proposition 6 allows us to compute the expected broadcast delay E$[D]$ for an external jamming scenario. This delay is approximated by the expected number of 1-factorizations needed for the completion of a broadcast, times the number of slots needed for the completion of one 1-factorization. Specifically, the first $(E[Z]-1)$ 1-factorizations require $(2n-1)$ slots, while the last 1-factorization requires, on average, $\frac{2n-1}{2}$ slots (the last successful transmission takes place on any of the 1-factors of the last 1-factorization with equal probability). Therefore, $\mathrm{E}[D] = (2n-1)\left(\mathrm{E}[Z] + \frac{1}{2}\right)$.

Fig. 8(b) shows the theoretical and simulated values of E$[D]$ as a function of the jamming probability p, when $K = 20$. We observe that even when the adversary jams 80% of the available channels (i.e., 16 out of 20), nodes can still complete their broadcast transmissions at the expense of some delay. We note that broadcast communication is maintained as long as at least one channel remains jam-free. In Fig. 8(c), we show E$[D]$ as a function of K for various J. E$[D]$ decreases with K, and approaches the asymptotic value obtained in the absence of a jammer, as it is expressed in Proposition 4.

For the AB mode, E$[D]$ does not have a simple closed-form expression but involves complex summation formulas. However, we can derive useful lower and upper bounds when $J = 1$.

Proposition 7: Let the per-slot jamming probability be equal to $p = \frac{1}{K}$, and let $K \geq n$. After the first successful relay of a broadcast message m, the broadcast delay D_2 until m is received by $(2n-2)$ nodes (all nodes but one) is bounded by,

$$\lceil \log_2(2n) \rceil - 1 \leq D_2 \leq \lceil \log_2(2n) \rceil. \quad (2)$$

Proof: The proof is provided in Appendix 7. □

Proposition 7 allows us to estimate the expected broadcast delay for the AB mode. Let D_1 denote the expected delay until the first success, D_2 the delay until $(2n-2)$ nodes receive message m and D_3 the delay until the last node receives m. The expected broadcast delay is bounded by

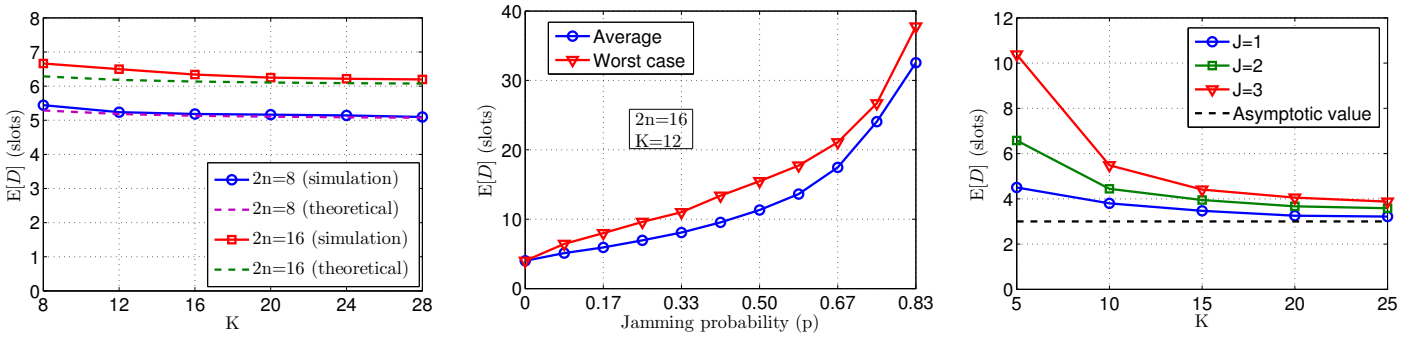
$$\begin{aligned} \mathrm{E}[D] &= \mathrm{E}[D_1 + D_2 + D_3] \\ &\leq \frac{K}{K-1} + \lceil \log_2(2n) \rceil + \frac{K}{K-1}. \end{aligned} \quad (3)$$

In (3), we have used the fact that it takes, on average, $\frac{K}{K-1}$ slots for the first successful relay when $p = \frac{1}{K}$. After the first success, $\lceil \log_2(2n) \rceil$ slots are needed in the worst case until $(2n-2)$ nodes receive m. The last node receives m after $\frac{K}{K-1}$ slots, on average.

We also studied the performance of the AB mode via simulations. We generated FH sequences of length $1,000$ hops according to Algorithm 2. We randomly selected J channels to be jammed per time slot. All results were averaged over 100 runs. Fig. 9(a) shows E$[D]$ as a function of K for $J = 1$. We observe that the theoretical bound derived using Proposition 7 agrees with the simulation. In Fig. 9(b), we show the simulated average and worst-case broadcast delay as a function of p. We observe that even when $p = 0.83$ (i.e., 10 out of 12 channels are jammed), the average and worst-case delays differ by less than six slots. This is due to the "relay explosion" effect of the splitting algorithm. The AB mode is significantly more resilient to jamming than the SU mode, due to the larger number of broadcast relays. Even when 83% of the available channels are jammed, the AB mode requires only 38 slots to complete a broadcast, compared to 228 slots needed in the SU mode. In Fig. 9(c), we show E$[D]$ as a function K for different J. We observe that with the increase of K, E$[D]$ asymptotically approaches the performance of the AB mode in the absence of jammers, given by Proposition 5.

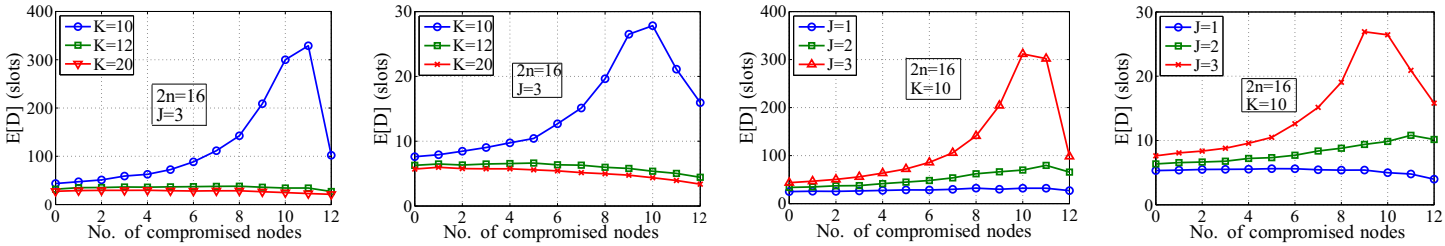
7.1.2 Resilience to Internal Jammers

In this set of experiments, we assumed that the adversary has compromised r nodes and recovered their FH sequences. We are



(a) (b) (c)

Fig. 9: (a) E$[D]$ as a function of K when $J = 1$, for the AB mode. The theoretical value is computed based on (3), (b) the average and worst case E$[D]$ as a function of p, for the AB mode, (c) E$[D]$ as a function of K and for various J. The asymptotic value is equal to $\lceil \log_2(2n) \rceil$.



(a) SU mode (b) AB mode (c) SU mode (d) AB mode

Fig. 10: (a), (b) E$[D]$ as a function of the number of compromised nodes for various values of K, when $J = 3$, (c), (d) E$[D]$ as a function of the number of compromised nodes for various values of J, when $K = 10$.

interested in determining the broadcast delay until the remaining $(2n - r - 1)$ *legitimate* nodes receive a broadcast message m. Knowledge of the r FH sequences reduces the adversary's uncertainty with respect to the rendezvous channels of legitimate pairs. This is because the channels assigned to each 1-factor form a permutation of the set of available channels $\mathcal{C}$. The exact value of E$[D]$ depends on the 1-factorization that is used to construct the FH sequences and the specific arrangement of the compromised nodes on that 1-factorization. The jamming probability p varies on a slot-by-slot basis and is given in the following proposition.

Proposition 8: Under the compromise of r nodes, the jamming probability p is bounded by

$$\min\{1, \frac{J}{K - \lceil \frac{r}{2} \rceil}\} \leq p \leq \min\{1, \frac{J}{K - r}\}. \quad (4)$$

Proof: The proof is provided in Appendix 8. □

We further used simulations to investigate the impact of node compromise on the broadcast delay. We generated FH sequences of length $1,000$ hops according to Algorithms 1 and 2 (depending on the TDBS mode). We randomly selected r of these sequences to be exposed to the adversary. At each time slot, the adversary randomly jammed J bands, excluding the exposed ones. A broadcast was deemed successful, when all legitimate nodes obtained a message copy. All results were averaged over 100 runs. Figs. 10(a) and 10(b) show E$[D]$ as a function of r when $J = 3$ and $K = 10, 12$, and 20. We observe that legitimate nodes complete their broadcast transmissions even when more than 50% of the broadcast group has been compromised. The AB mode exhibits significantly lower delay compared with the SU mode, due to the use of multiple relays. Note that when K is small and several nodes are compromised, the jammers have a high chance of jamming legitimate pairs. This fact can be seen from the sharp increase of E$[D]$ when $K = 10$.

In Figs 10(c) and 10(d), we show E$[D]$ as a function of r and for various J when $K = 10$, for the SU and the AB communication modes. Even with the increase of J, legitimate nodes are able to complete their broadcast transmissions in both modes, with the AB mode being the most efficient. Note that in all graphs of Fig. 10, E$[D]$ decreases when a large number of nodes is compromised, since fewer legitimate nodes need to receive the broadcast message for completing a broadcast transmission.

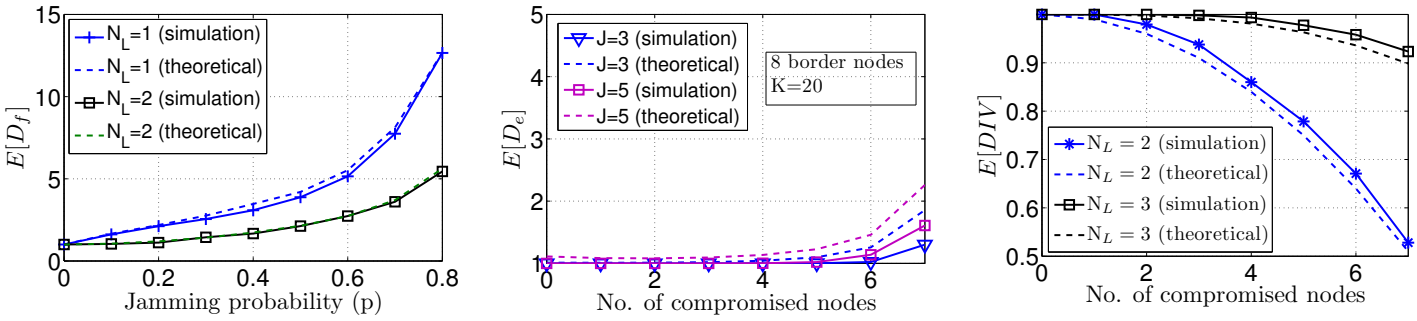
7.2 Evaluation of Multi-hop Scenarios

In this section, we evaluate TDBS for multi-hop networks. We focus on the jamming-resistance of the inter-cluster phase, since for the intra-cluster phase, the security analysis for single-hop networks holds. For the inter-cluster phase, we define the following performance metrics.

Definition 12: **Flooding Delay D_f:** Number of slots needed until all clusters adjacent to a cluster x, have received a broadcast that originated from a node in x.

Definition 13: **Escape Delay D_e:** Number of slots needed until a broadcast message that originated at a cluster x is received by any node in any adjacent cluster.

Definition 14: **Escape diversity DIV:** Fraction of adjacent clusters that receive a broadcast directly from a cluster x, when a subset of the border nodes in x are compromised.



(a) (b) (c)

Fig. 11: (a) $\mathrm{E}[D_f]$ as a function of the jamming probability p when $K = 20$, (b) $\mathrm{E}[D_e]$ as a function of the number of compromised nodes r for various J, (c) $\mathrm{E}[DIV]$ as a function of r for various N_L when $K = 20$.

We first analytically evaluate $\mathrm{E}[D_f]$ in the presence of external jammers. Assume a cluster with N_C adjacent clusters. Let N_L denote the average number of "bridge links" between two adjacent clusters.

Proposition 9: In the presence of an external jammer, $\mathrm{E}[D_f]$ is equal to

$$\mathrm{E}[D_f] = (1-\tilde{p})^{N_C} + \sum_{i=2}^{\infty} i(1-\tilde{p}^{i-1})^{N_C} \times \sum_{k=1}^{N_C} \binom{N_C}{k} \left(\frac{\tilde{p}^{i-1}(1-\tilde{p})}{1-\tilde{p}^{i-1}}\right)^k, \quad (5)$$

where $\tilde{p} = \left(\frac{J}{K}\right)^{N_L}$ denotes the probability that all N_L links to an adjacent cluster are jammed at a particular slot.

Proof: The proof of Proposition 9 follows the same steps as the proof of Proposition 6, by substituting $p = \frac{J}{K}$ with $\tilde{p} = \left(\frac{J}{K}\right)^{N_L}$. We refer to the proof provided in Appendix 6. □

We also verified Proposition 9 via simulations. In our setup, we generated a multi-hop topology consisting of 50 nodes, organized in clusters. We then generated FH schedules for all nodes in the network for the inter-cluster phase, according to the algorithm described in Section 6.2. At each time slot, the jammer was assumed to block J random frequency bands across the entire network. Results were averaged over all clusters in the network. Fig. 11(a) shows $\mathrm{E}[D_f]$ as a function of the jamming probability p. We observe that, even when 80% of the available frequency bands are jammed, only 13 inter-cluster slots are needed until all neighboring clusters directly receive a broadcast. We also evaluated the expected escape delay $\mathrm{E}[D_e]$ under the compromise of r border nodes.

Proposition 10: Under the compromise of r border nodes of a cluster x, $\mathrm{E}[D_e]$ is given by,

$$\mathrm{E}[D_e] = \frac{1}{1-\left(P_c^{N_L} + \sum_{i=1}^{N_L}\binom{N_L}{i}\left(\frac{J(1-P_c)}{K-r}\right)^i\right)^{N_C}}, \quad (6)$$

where $P_c = \frac{r}{N_C \times N_L}$ denotes the compromise probability.

Proof: The proof is provided in Appendix 9. □

The expected escape diversity $\mathrm{E}[DIV]$ is evaluated in the following proposition.

Proposition 11: Under the compromise of r nodes, $\mathrm{E}[DIV]$ is given by

$$\mathrm{E}[DIV] = 1 - P_c^{N_L}. \quad (7)$$

Proof: The proof is provided in Appendix 10. □

Figs 11(b) and 11(c) show $\mathrm{E}[D_e]$ and $\mathrm{E}[DIV]$ as a function of the number of compromised border nodes. In our simulation, compromised border nodes do not relay messages and their FH sequences are assumed to be known to the jammer. From Fig. 11(b), we observe that a small number of slots is sufficient for the first copy of a broadcast message to reach one adjacent cluster. From Fig. 11(c), we observe that more than 90% of neighboring clusters are guaranteed to receive the broadcast message when $N_L = 3$, while this value being reduced to 50% when $N_L = 2$. We emphasize that even if certain adjacent clusters do not receive the broadcast message directly from the originating cluster, such a relay can occur indirectly via other clusters.

7.3 Evaluation of Dynamic Groups

Our evaluations so far considered network snapshots of static broadcast groups. This is because the broadcast delay is orders of magnitude smaller than the time scale of group dynamics. A group membership change affects the delay characteristics of a limited number of broadcasts due to the execution of the training phase. In this section, we evaluate the delay overhead until the FH schedule is updated when nodes are added or deleted.

Node addition: A node addition involves the expansion of $\mathcal{F}_{2n}$ to $\mathcal{F}_{2n+2}$, the computation of the modified FH schedule by the CA, and the training of existing nodes by the newly deployed ones. The construction of $\mathcal{F}_{2n+2}$ is individually performed by existing nodes, without requiring any message exchange. The computation of the modified FH schedule is done by the CA before the deployment of the new node, without any input from existing nodes. Communications take place during the training phase only, when the newly added node communicates the FH schedule changes to existing nodes. This phase is equivalent to the SU operation mode.

To complete the training phase, the newly added node must unicast one FH sequence update message to each deployed node. The FH sequence th followed by the newly added node during the

training phase allows this node to communicate with one existing node per slot. This is identical to the SU mode, where the source node unicasts the same broadcast message to one other node per slot. Under an external jamming model, the delay incurred during the training phase is given by Proposition 6, where $2n-1$ is replaced with $2n$ (the newly added node has to communicate with $2n$ other nodes instead of $2n-1$). For internal jammers, the broadcast delay is obtained by the simulations presented in Section 7.1.2 for the SU mode. This delay represents the training phase delay.

Node deletion: When nodes are deleted from the broadcast group, the remaining nodes individually modify their FH sequences without any information exchange. The only information required is the identities of the deleted nodes, which can be communicated using a single broadcast message originating from the group leader.

8 Related Work

The jamming problem has been extensively studied under an external threat model (for example, see [30] and the references therein). Jamming is typically mitigated by spreading the transmitted signal to a larger bandwidth using a secret PN code. Without knowledge of the PN code, the jammer has to transmit with several orders of magnitude more power than the transmitter (typically 20-30 dB) to interfere with ongoing transmissions. However, in broadcast communications, compromise of common PN codes suppresses the advantages of SS.

Several researchers have considered the jamming problem under an inside threat model. Chan et al. showed that a jammer that targets the broadcast control channel in GSM networks can reduce the required power for performing a DoS attack by several orders of magnitude [8]. To alleviate jamming, the authors proposed the BBK scheme, in which base stations (BSs) broadcast over several control channels in parallel. Each node is preloaded with the "locations" of a subset of the available channels. These locations are determined according to a cryptographic combinatorial design. If the number of compromised nodes is less than a threshold T, all legitimate nodes receive the broadcast transmission in one slot. Otherwise, all broadcast communications are blocked. Compared with the BBK scheme, in TDBS, nodes transmit on one channel per slot. Moreover, TDBS implements an any-to-any communication model as opposed to an one-to-any communication model. In [31], [32], the BBK scheme was extended to a probabilistic combinatorial design that provides a graceful degradation in performance as a function of the number of inside jammers. Similar to the BBK scheme, the authors allow the BS to simultaneously broadcast over multiple channels.

Alternative methods eliminate the dependency on shared secrets [16], [19], [26]. Pöpper et al. proposed a DSSS-based method called Uncoordinated DSSS (UDSSS) [26]. In UDSSS, broadcast transmissions are spread according to a PN code, randomly selected from a public set of codes. Receivers must exhaustively apply all codes in the codebook to recover the broadcast message. Liu et. al. showed that UDSSS is vulnerable to a reactive jammer with sufficient computational power to recover the PN code before the end of an ongoing transmission [19]. They proposed RD-DSSS, a randomized differential DSSS scheme, that expands the public code set and discloses the selected code after the message transmission has ended, thus providing resilience to reactive jammers. The computational efficiency of RD-DSSS was further improved in [17] by encoding the seed of the PN code used to spread a message, at the end of that message. This delayed seed disclosure prevented a jammer from acquiring the PN code before the message was fully received.

DSSS-based schemes are not directly comparable with TDBS because they employ different physical-layer mechanisms for rejecting narrowband interference. DSSS communications are resilient to low/medium interference levels. A powerful jammer that is unaware of the PN spreading sequence can eventually overwhelm a DSSS receiver if its power is raised to sufficient levels. This is because any increase in interference power increases the despread noise floor in a linear fashion [2], [30]. In a FHSS system, a narrowband interference signal has no effect on communications taking place in frequencies outside the targeted band. Hence, a power increase in the narrowband interference will not impact the system's interference rejection capability. However, FHSS systems require band orthogonality with guard bands exceeding the 200KHz. Moreover, FHSS transceivers must be capable of accessing a large spectrum, thus complicating the transceiver design.

Several methods attempt to identify the compromised nodes that leaked information to the jammer. Lazos et al. proposed the assignment of unique FH hopping sequences to each receiver, overlapping in a fixed subset of hops [16]. A compromised node is identified when the jammer follows the node's unique PN code (or a unique subset of it). Tague et al. proposed the GUIDE scheme for identifying compromised nodes based on the set of channels that are jammed. They formulated the identification problem as a maximum likelihood estimation problem [32].

Chiang and Hu, developed a code-tree based approach for identifying compromised PN codes [10]. In this design, each node is assigned a subset of orthogonal pseudo-noise (PN) codes (either DS or FH codes). A broadcasting node communicates with the broadcast group using several PN codes. The goal of the transmitter is to identify a set of codes that covers all legitimate receivers and excludes the compromised ones. Once the system converges to a disjoint set cover of the legitimate receivers, the broadcast delay becomes equal to one slot. A broadcaster must repeat its transmission using a minimum number of codes related to the number of jammers.

TDBS is also related to a class of rendezvous protocols that were recently proposed for regulating dynamic spectrum access [1], [7], [37], [38]. Zhang et al. proposed the decomposition of complete graphs to perfect rainbow matchings to generate the rendezvous schedule of secondary users in cognitive radio networks (CRNs) [38]. This is equivalent to creating a 1-factorizations of complete graphs. The perfect matchings proposed in [38] are a function of the number of available channels, but independent of the number of nodes. When the group size is larger than the number of channels, the same FH sequence is assigned to more than one nodes. Moreover, the FH sequence period is equal to $(2K-1)$, where K is the number of available channels. In TDBS, the perfect matchings are a function of the broadcast group size rather than the number of channels. Hence, nodes are guaranteed

to hop using *unique*, collision-free FH sequences, with much longer periods.

Rahman et al. proposed three adaptive FH algorithms for multicast rendezvous in DSA networks [1]. Their algorithms guarantee the rendezvous of all nodes on the same slot without requiring synchronous FH. The FH sequence design is based on two special quorum systems that satisfy the rotation k-closure property. They also account for the PU dynamics by designing optimal channel ordering mechanisms.

9 Conclusion

We proposed TDBS, a scheme for jamming-resistant broadcast communications in the presence of inside jammers. In TDBS, broadcast is realized as a series of unicast transmissions distributed in frequency and time. Because the adversary is limited in the number of channels he can jam, several unicast transmissions remain interference-free. We mapped the problem of constructing FH sequences for the TDBS to the problem of 1-factorization of complete graphs. We further developed mechanisms for updating the FH sequences assigned to nodes, when the broadcast group is dynamic. We mapped the problem of minimizing the number of FH sequence changes required for node addition, to the problem of finding rainbow paths in proper edge-colored complete graphs. We analytically evaluated the security properties of TDBS under both an external and an internal threat model and showed that TDBS maintains broadcast communications even when multiple nodes are compromised.

References

[1] M. Abdel Rahman, H. Rahbari, and M. Krunz. Adaptive frequency hopping algorithms for multicast rendezvous in DSA networks. In *Proc. of the IEEE DYSPAN Symposium*, pages 517–528, 2012.

[2] D. Adamy. *EW 101: A first course in electronic warfare*. Artech House Publishers, 2001.

[3] L. D. Andersen. Hamilton circuits with many colours in properly edge-coloured complete graphs. *Mathematica Scandinavica*, 64:5–14, 1989.

[4] K. Appel and W. Haken. Every planar map is four colorable: part I. *Illinois journal of Mathematics*, 21(3):491–567, 1977.

[5] P. Bahl, R. Chandra, and J. Dunagan. SSCH: slotted seeded channel hopping for capacity improvement in IEEE 802.11 ad-hoc wireless networks. In *Proc. of the MOBICOM Conference*, pages 216–230, 2004.

[6] L. C. Baird, W. L. Bahn, M. D. Collins, M. C. Carlisle, and S. C. Butler. Keyless jam resistance. In *Proc. of the IEEE Workshop on Information Assurance United States Military Academy*, 2007.

[7] K. Bian, J. Park, and R. Chen. A quorum-based framework for establishing control channels in dynamic spectrum access networks. In *Proc. of the MOBICOM Conference*, pages 25–36, 2009.

[8] A. Chan, X. Liu, G. Noubir, and B. Thapa. Broadcast control channel jamming: Resilience and identification of traitors. In *Proc. of the ISIT Conference*, pages 2496–2500, 2007.

[9] P. Chaporkar, K. Kar, X. Luo, and S. Sarkar. Throughput and fairness guarantees through maximal scheduling in wireless networks. *IEEE Transactions on Information Theory*, 54(2):572–594, 2008.

[10] J. T. Chiang and Y.-C. Hu. Dynamic jamming mitigation for wireless broadcast networks. In *Proc. of the INFOCOM Conference*, pages 1211–1219, 2008.

[11] J. H. Dinitz and D. R. Stinson. A hill-climbing algorithm for the construction of one-factorizations and room squares. *SIAM J. Algebraic Discrete Methods*, 8(3):430–438, 1987.

[12] S. Ganeriwal, C. Pöpper, S. Čapkun, and M. B. Srivastava. Secure time synchronization in sensor networks. *ACM Transactions on Information and System Security (TISSEC)*, 11(4):23, 2008.

[13] S. Gollakota and D. Katabi. Zigzag decoding: Combating hidden terminals in wireless networks. In *Proc. of the ACM SIGCOMM Conference*, pages 159–170, 2008.

[14] A. Gupta, X. Lin, and R. Srikant. Low-complexity distributed scheduling algorithms for wireless networks. *IEEE/ACM Transactions on Networking (TON)*, 17(6):1846–1859, 2009.

[15] A. Gyárfás and M. Mhalla. Rainbow and orthogonal paths in factorizations of k_n. *Journal of Combinatorial Designs*, 18(3):167–176, 2010.

[16] L. Lazos, S. Liu, and M. Krunz. Mitigating control-channel jamming attacks in multi-channel ad hoc networks. In *In Proc. of the 2nd ACM WiSec Conference*, pages 169–180, 2009.

[17] A. Liu, P. Ning, H. Dai, Y. Liu, and C. Wang. Defending DSSS-based broadcast communication against insider jammers via delayed seed-disclosure. In *Proc. of the Annual Computer Security Applications Conference (ACSAC'10)*, pages 367–376, 2010.

[18] S. Liu, L. Lazos, and M. Krunz. Thwarting inside jamming attacks on wireless broadcast communications. In *Proc. of the 4th ACM WiSec Conference*, pages 29–40, 2011.

[19] Y. Liu, P. Ning, H. Dai, and A. Liu. Randomized differential DSSS: Jamming-resistant wireless broadcast communication. In *Proc. of the INFOCOM Confernece*, pages 1–9, 2010.

[20] Mathworks. Matlab r2012. http://www.mathworks.com/.

[21] E. Mendelsohn and A. Rosa. One-factorizations of the complete graph-a survey. *Journal of Graph Theory*, 9(1):43–65, 1985.

[22] G. Noubir and G. Lin. Low-power DoS attacks in data wireless LANs and countermeasures. *Mobile Computing and Communications Review*, 7(3):29–30, 2003.

[23] B. O'hara and A. Petrick. *IEEE 802.11 handbook: a designer's companion*. IEEE Standards Association, 2005.

[24] M. D. Plummer. Graph factors and factorization: 1985–2003: A survey. *Discrete Mathematics*, 307(7):791–821, 2007.

[25] R. Poisel. *Modern communications jamming principles and techniques*. Artech House on Demand, 2004.

[26] C. Popper, M. Strasser, and S. Capkun. Jamming-resistant broadcast communication without shared keys. In *Proc. of the USENIX Security Symposium*, pages 231–248, 2009.

[27] C. Popper, M. Strasser, and S. Capkun. Anti-jamming broadcast communication using uncoordinated spread spectrum techniques. *IEEE Journal on Selected Areas in Communications*, 28(5):703–715, 2010.

[28] S. Sarkar and L. Tassiulas. End-to-end bandwidth guarantees through fair local spectrum share in wireless ad-hoc networks. *IEEE Transactions on Automatic Control*, 50(9):1246–1259, 2005.

[29] G. Sharma, C. Joo, and N. Shroff. Distributed scheduling schemes for throughput guarantees in wireless networks. In *Proc. of the 44th Annual Allerton Conference on Communications, Control, and Computing*, 2006.

[30] M. K. Simon, J. K. Omura, R. A. Scholtz, and B. K. Levitt. *Spread Spectrum Communications Handbook*. McGraw-Hill, 2001.

[31] P. Tague, M. Li, and R. Poovendran. Probabilistic mitigation of control channel jamming via random key distribution. In *Proc. of the IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pages 1–5, 2007.

[32] P. Tague, M. Li, and R. Poovendran. Mitigation of control channel jamming under node capture attacks. *IEEE Transactions on Mobile Computing*, 8(9):1221–1234, 2009.

[33] L. Tassiulas and A. Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control*, 37(12):1936–1948, 2002.

[34] P. Tosic and G. Agha. Maximal clique-based distributed group formation for autonomous agent coalitions. In *Proc. of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, 2004.

[35] W. Wallis. One-factorizations of complete graphs. *Contemporary Design Theory: A Collection of Surveys*, pages 692–731, 1992.

[36] W. Wallis. *One-factorizations*. Kluwer Academic Publishers, 1997.

[37] Y. Zhang, Q. Li, G. Yu, and B. Wang. ETCH: Efficient channel hopping for communication rendezvous in dynamic spectrum access networks. In *Proc. of the INFOCOM Conference*, pages 2471–2479, 2011.

[38] Y. Zhang, G. Yu, Q. Li, H. Wang, X. Zhu, and B. Wang. Channel-hopping-based communication rendezvous in cognitive radio networks. *IEEE/ACM Transactions on Networking*, pages 1–14, 2013.