Cross-Layer Jamming Detection and Mitigation in Wireless Broadcast Networks

Jerry T. Chiang, *Student Member, IEEE*, and Yih-Chun Hu, *Member, IEEE*

***Abstract*—Wireless communication systems are often susceptible to the jamming attack where adversaries attempt to overpower transmitted signals by injecting a high level of noise. Jamming is difficult to mitigate in broadcast networks because transmitting and receiving are inherently *symmetric* operations: A user that possesses the key to decode a transmission can also use that key to jam the transmission. We describe a *code tree* system that provides input to the physical layer and helps the physical layer circumvent jammers. In our system, the transmitter has more information than any proper subset of receivers. Each receiver cooperates with the transmitter to detect any jamming that affects that receiver. In the resulting system, each benign user is guaranteed to eliminate the impact of the attacker after some finite number of losses with arbitrarily high probability. We show that any system that relies on only using spreading code, and no other physical factors, to mitigate jamming must use at least $j+1$ codes, where j is the number of jammers. We then propose an optimized scheme that is power-efficient: Each transmission is sent on at most $2j+1$ codes simultaneously. Finally, we demonstrate that our scheme approaches the best possible performance by performing an extensive analysis of the system using both event-driven *ns-2* and chip-accurate MATLAB simulations.**

***Index Terms*—Broadcast networks, jamming mitigation, spread spectrum.**

I. Introduction

WIRELESS communication systems are often susceptible to the jamming attack in which adversaries attempt to overpower transmitted signals by injecting a high level of noise, thereby lowering the signal-to-noise ratio (SNR). Lowering the SNR, in turn, can significantly reduce the achievable rate of a communication system.

An effective countermeasure to the jamming attack is increasing the bandwidth of the spectrum of the communication system and using *spread spectrum* as part of the modulation technique [3]. In spread-spectrum systems, a transmitter takes advantage of the increased bandwidth to redundantly encode information using a *spreading code*. To receive a message, a spread-spectrum receiver decodes the incoming signal by correlating the signal with the spreading code. Spread-spectrum codes are thus inherently *symmetric*; that is, the transmitter and the receiver use the same information for encoding and for decoding. Without knowing the spreading code used by a pair of a transmitter and receiver, unintended signals such as jamming or self-interference will likely appear noise-like upon decoding, and most of the unintended signal power can then be rejected by filtering. However, if a jammer discovers the spreading code in use (for example, by compromising the receiver), all benefit of using spread spectrum against jamming is lost.

While using spread spectrum as part of modulation can be highly effective against jamming in point-to-point wireless communication systems in which a single transmitter transmits to a single receiver, it is difficult to prevent jamming in a broadcast system that transmits information to multiple users at once. This is because if a group of receivers shares a single code, then a jammer can deny service to the entire group by compromising any one of the group members. However, a transmitter is interested in conserving the number of spreading codes used in a broadcast system since the total transmission power is divided between the set of spreading codes simultaneously used.

In this paper, we present a scheme that allows a receiver to detect jamming by observing that a secondary message is received without the primary message. We then present a keying scheme that allows the transmitter to cooperate with the receiver to isolate the set of jammers from the set of benign users. Finally, we develop a technique called *tree remerging* to optimize our keying scheme so that a transmitter can group benign receivers together and let that group share one spreading code, thereby providing satisfactory quality of service to the receivers without requiring higher total transmission power.

Modern mobile networks commonly use broadcast messages to discover routing information. Cellular phone standards, such as IS-95, also use broadcast messages for synchronization and paging. It is therefore very important to defend broadcast from jamming. Previously proposed approaches to mitigate jamming, such as spread spectrum, reside entirely at the physical layer. However, substantial previous work shows that upper layer feedback can improve lower layer performance in areas such as transmit power control [4]. Our keying scheme uses a binary tree structure implemented above the physical layer that takes advantage of the unique properties of spreading codes in order to mitigate jamming in any broadcast systems based on existing spread-spectrum techniques.

Current commercial spread-spectrum systems, such as IS-95, are not suitable for use in an adversarial environment due to the use of fixed and published codes. We will assume the use of unpredictable and time-varying spreading codes, such as a code generated using Advanced Encryption Standard [5], to eliminate

Manuscript received April 30, 2009; revised February 27, 2010 and June 29, 2010; accepted July 10, 2010; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor D. Agrawal. Date of publication September 16, 2010; date of current version February 18, 2011. This work was supported in part by the USARO under Contract No. W-911-NF-0710287. This paper was presented in part at the 13th Annual ACM International Conference on Mobile Computing and Networking (MobiCom), Montréal, QC, Canada, September 12, 2007, and at the 27th IEEE International Conference on Computer Communications (INFOCOM), Phoenix, AZ, April 17, 2008.

The authors are with the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Urbana, IL 61801 USA (e-mail: chiang2@illinois.edu; yihchun@illinois.edu).

Digital Object Identifier 10.1109/TNET.2010.2068576

the security flaws inherent in using fixed spreading codes over an extended period of time. Our proposed protocol has broad applicability to a wide variety of existing wireless access technologies such as IEEE 802.11 [6], IS-95 [7], and CDMA2000 [8], which are already CDMA systems.

To adopt our protocol, a CDMA system must be able to assign each user a different set of spreading codes that should change over time. Most current CDMA systems already require a client registration phase, where each client is given an identification number. For example, a client needs to provide some identification in order to obtain 3G service on a CDMA phone with a unique electronic serial number (ESN). The set of spreading codes can thus be distributed during the registration phase. The use of time-varying spreading code is not necessary for running our protocol; however, our protocol adopts time-varying spreading codes in order to prevent attackers from learning a spreading code by correlating messages over time. Our protocol does not require regular feedback from receivers; however, our broadcast transmitter does need to be able to receive jamming report from receivers occasionally.

We present all necessary background of the paper in Section II, including an overview of two spread-spectrum techniques, the related work, and our attacker and system model. We then propose our keying protocol that mitigates jamming in a wireless broadcast network in Section III. We develop an optimization and show its optimality in Section IV. We explore the issue of false alarm in Section V. We simulated our protocols and present the results in Section VI and conclude in Section VII.

II. Background

In this section, we present two spread-spectrum techniques, the *fast-frequency-hopping code division multiple access* (FFH-CDMA) and the *direct-sequence code division multiple access* (DS-CDMA). These spread-spectrum techniques are used to illustrate and analyze our protocol. We then overview related work. We finally present our attacker and system models.

A. FFH-CDMA and DS-CDMA

FFH-CDMA and DS-CDMA are two distinct spread-spectrum implementations. In FFH-CDMA, a transmitter changes the frequency bands on which the signals are transmitted. In DS-CDMA, a transmitter maps each bit into a sequence of chips, which are then modulated and transmitted.

In a FFH-CDMA system, the entire spectrum of the communication system is divided into a number of frequency bands, and time is divided into time slots, the duration of which is much shorter than the time it takes to send 1 bit of information. Each user is assigned a *frequency-hopping pattern* that serves as his spreading code. In each time slot, the transmitter occupies a particular set of frequency bands and changes, at each time slot, to another set of bands according to his frequency-hopping pattern. The transmission made in a single time slot is referred to as a *chip*. The receiver receives the signal by monitoring the waveforms on the set of frequencies specified by the hopping pattern in each time slot and combining the waveforms.

In a DS-CDMA system, each bit is mapped to either 1 or -1, and each user is assigned a *pseudorandom code* of length η. To send a 1 bit, the transmitter transmits the pseudorandom code, and to send a -1 bit, the transmitter transmits the additive inverse of the code. To decode a bit, the receiver takes the inner product of the code and the signal it received; if the inner product is positive, then a 1 bit was sent, and if the inner product is negative, then a -1 bit was sent. (Intuitively, the inner product computes the correlation between the input and the code.) The DS-CDMA system relies heavily on the property of *code orthogonality*. Two pseudorandom codes are said to be orthogonal if their inner product is zero. Moreover, two pseudorandom codes of sufficient length have been shown to be asymptotically orthogonal [9]. That is, given two random pseudorandom codes of equal length, the expected ratio of their inner product and their length diminishes with respect to their length. Messages sent on orthogonal codes do not interfere with each other. For example, if one transmitter transmits bit b_1 on code c_1 of length η, and another transmitter transmits bit b_2 on an orthogonal code c_2, then the message received by any receiver is $\alpha_1 b_1 \mathsf{c}_1 + \alpha_2 b_2 \mathsf{c}_2$, where α represents the *path loss* from each transmitter to the receiver. A receiver using code c_1 will compute the inner product (denoted by the $\cdot$ operator) of c_1 and the received signal

$$\begin{aligned}\mathsf{c}_1 \cdot (\alpha_1 b_1 \mathsf{c}_1 + \alpha_2 b_2 \mathsf{c}_2) &= \alpha_1 b_1 (\mathsf{c}_1 \cdot \mathsf{c}_1) + \alpha_2 b_2 (\mathsf{c}_1 \cdot \mathsf{c}_2)\\ &= \alpha_1 \eta b_1\end{aligned}$$

which is not affected by the transmission of b_2.

Due to the orthogonality of frequencies and pseudorandom codes, FFH-CDMA and DS-CDMA both allow multiple users to transmit and receive simultaneously. For example, to simultaneously transmit the same packet on two frequency-hopping patterns that each specifies a single frequency band for every time slot, a transmitter divides its power across the two frequency bands specified in the patterns. If the two patterns have a time slot in which they share a band, then the transmitter can use its full power on a single band during that time slot. To simultaneously receive on two frequency-hopping patterns, a receiver simply monitors the two frequency bands. The ability of spread-spectrum systems to simultaneously transmit and receive has long been used in commercial systems such as IS-95 [7].

B. Related Work

Unintentional interference and jamming mitigation using CDMA has been studied at length [3]. Other physical-layer techniques, such as the use of multiple antennas and antenna nulling [10], have also been studied. Recently, researchers have also sought to avoid jamming by taking advantage of various properties of physical propagation. Bahn *et al.* proposed a particular coding scheme, the BBC code, such that when used with *indelible marks*, an energy-limited jammer cannot interfere with the message transmission indefinitely [11]. Strasser *et al.* proposed the uncoordinate frequency-hopping protocol [12], in which the transmitter seeks to finish its transmission before a jammer can find out on which frequency band the signal was transmitted. Follow-up studies have sought to extend the concept to other spreading techniques and to incorporate variable length of spreading code for faster decoding [13]. Our approach is different from these schemes in that any transmitter–receiver pair in our proposed jamming mitigation technique shares prior-agreed keys and spreading codes. We thus do not need to make any new assumptions about the computation ability of a jammer. For example, in the uncoordinated frequency-hopping scheme, the authors relied on the assumption that a jammer cannot timely detect the frequency band on which a packet of multiple bits is transmitted [12].

Asymmetric cryptography [14], such as RSA [15] and Diffie–Hellman [16], rely on the alleged asymmetry of certain computational functions to achieve public-key cryptography and digital signatures. Our work differs in that it overlays an inherently symmetric operation: wireless transmission. Other work has used time and delayed disclosure to provide asymmetry [17], [18]. If we wish to use digital signatures as spreading codes (as described by Kuhn [18]), we still need a jam-resistant way to provide receivers with a spreading code.

The effectiveness of jamming [19] and the difficulty of differentiating jamming from congestion [20] have previously been discussed, but no solutions were proposed to traverse the jammed area. In particular, Xu *et al.* [20] try to detect and avoid jammed regions.

To algorithmically detect and avert jamming, we adopt the key-tree structure proposed by several multicast key management methods. In particular, Wong *et al.* [21] proposed using key-graphs for secure group communication. As a special type of key-graph, the authors discussed an m-ary key-tree scheme where each node of an m-ary tree corresponds to a key, and each user possesses the keys corresponding to a leaf and all ancestors of that leaf. That is, each user possesses one key that is known only to himself and shares all the ancestor codes with a subset of other users. Our work uses a similar structure in managing the spreading codes between transmitter and receivers. We then propose a protocol that uses the code tree to provide upper-layer feedback to the lower layers and select a subset of spreading codes to use in a broadcast communication system.

C. Attacker and System Model

We make the standard assumption that any jammer is power-limited, but not necessarily energy-limited. That is, we assume that a jammer can transmit for as long as he wishes; however, the jammer cannot transmit with infinite power regardless of how short the time duration is. Specifically, we assume that a jammer is not powerful enough to saturate the analog-to-digital converter at the receiving end. We make no other assumptions on the physical ability of the jammer.

We assume a jammer cannot calculate the spreading code based on a message within the time duration to transmit 1 bit. We make no other limitations on the computation power of a jammer. Since our broadcast transmitter does not publicly select any subset of spreading codes to use, the number of codes a jammer needs to test to find the set of spreading codes in use is exponential with respect to the length of the spreading code. However, the time duration to send a bit is only linear with respect to the length of the spreading code. Therefore, our assumption is reasonable since the computation time is of different order than the transmission time.

When a jammer compromises a receiver, we allow the jammer to learn all the secrets known to that receiver. That is, if the compromised receiver shares some secret with another party, that secret is also known to the jammer after the receiver is compromised.

We assume that each receiver shares a set of spreading codes with the transmitter, which can be achieved by using an out-of-band authentication scheme.

III. Code Tree Scheme

In this section, we first present the challenges in making a power-efficient spread-spectrum broadcast system. We then present our code management scheme that adopts the structure of a binary key tree [21]. We then present a scheme that detects whether a particular spreading code is jammed. We also present how the transmitter can mitigate jamming when jamming is detected.

A. Symmetry of Spreading Codes

The current use of spreading codes in a spread-spectrum system is analogous to a *symmetric-key cryptosystem*, in which an encryption code and the corresponding decryption code are easily derivable from each other. For example, in the FFH-CDMA system, encoding and decoding both use the same hopping pattern. By keeping each hopping pattern a secret between the transmitter and receiver, the hopping pattern effectively serves as a cryptographic key for both encryption and decryption. In particular, without the knowledge of the hopping pattern in use, a jammer at each time slot must randomly choose a set of frequency bands on which to emit power. If the jammer selects too many bands, then its effective power in each band is substantially reduced. On the other hand, if the jammer fails to jam most of the frequency bands specified in the hopping pattern, then the legitimate signal will likely have a higher received power level than the jamming signal after decoding and is likely to be successfully received.

A spreading code can thus be viewed as a *secret key* between the sender and the receiver, such that a jammer without the key is unable to effectively jam a message sent using that code. This symmetry presents significant challenges to the design of a broadcast system: A symmetric key should not be shared, otherwise a single compromised user can jam in a way that cannot be rejected by using spread spectrum alone.

We present a protocol in which a broadcast transmitter possesses more knowledge than any proper subset of receivers, thereby creating an asymmetric system that allows detection and isolation of jammers.

B. Tree-Based Approach

In this section, we describe our approach to create an asymmetric system that allows detection and isolation of jammers in a spread-spectrum broadcast system. Each broadcast transmitter uses a structure similar to the multicast key tree proposed by Wong *et al.* [21]. Each transmitter builds a balanced binary tree of randomly generated spreading codes. The transmitter associates each legitimate receiver with a unique leaf in this binary tree and gives this receiver the spreading codes corresponding to that leaf and all ancestors of that leaf in the tree. For example, in

TABLE I
DEFINITION OF DIFFERENT TYPES OF COVERS

Term	Definition	Explanation
Cover	$\forall$ leaf c, $C \cap \text{ancestors}(c) \neq \emptyset$	Each receiver can decode at least one code
Disjoint	$\forall c_1, c_2 \in C$, $c_1 \notin \text{ancestors}(c_2)$	Each receiver can decode at most one code
Safe	$C \cap J = \emptyset$	No previously jammed code is used
Minimal Safe	$\forall c \in C$, $c = \text{root} \vee \text{parent}(c) \in J$	No smaller safe cover exists

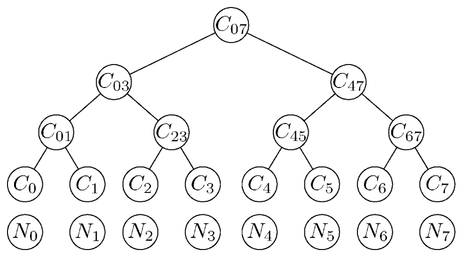


Fig. 1. Example code tree.

Fig. 1, user N_2 would have access to spreading codes C_2, C_{23}, C_{03}, and C_{07}.

In the initial phase of our protocol, a transmitter transmits to all receivers on a single spreading code; specifically, it would choose the spreading code corresponding to the root of the tree. Transmissions on this spreading code can be decoded by any legitimate (benign and compromised alike) receiver if the code is not jammed.

In general, in order to ensure that every receiver can decode a packet while minimizing the number of codes simultaneously used, the transmitter wants to transmit on a set of spreading codes such that any user can decode using exactly one spreading code in the set. We call such a set of spreading codes a *disjoint cover*. Once jamming has been detected on some spreading codes (we will discuss jamming detection in Section III-C), the transmitter should avoid using such spreading codes in the future.

To receive a message, each receiver simply decodes the signal using all the codes he knows. If a receiver does not have enough computation ability to decode using all codes he knows simultaneously, the receiver can sequentially try all codes, using only a subset of codes in each time instance.

Given a set of codes J on which jamming has been detected, we call a cover C *safe* relative to J if $C \cap J = \emptyset$; that is, if C contains no codes on which jamming was previously detected. Because each additional code used for transmission either increases the power consumption or reduces the received signal strength, we want to transmit on the smallest possible safe set. We call a safe cover *minimal* if no safe cover has fewer codes in it. In particular, for each code c in a minimal safe cover, either c is the root of the code tree or $\text{parent}(c) \in J$. A minimal safe cover must also form a disjoint cover. Table I summarizes these definitions.

Since we assume all codes are near-orthogonal to each other, and each benign receiver is given a unique leaf code, jamming a leaf code thus nullifies the benefit of spread spectrum for only one receiver, namely the receiver who is now considered compromised since it has lost its secret shared with the transmitter. To all other receivers that do not use such code, jamming that leaf code is similar to raising the noise floor. Consequently, we assume J does not contain any nonleaf codes. We let κ be the set of all nonleaf codes known to the jammer. Ideally, we would like $J = \kappa$ after finite time.

C. Jamming Detection Algorithm and Response

In this section, we present our algorithm to detect jamming on a particular spreading code. We then show how a broadcast system using our code tree can respond to detected jamming.

1) Jamming Detection: When the transmitter sends a packet, it will do so on the current minimal safe cover, on which no jamming had been previously detected, so that all legitimate receivers can decode the packet. In order to detect further jamming activities, the transmitter additionally transmits on a *test* spreading code that is randomly chosen from among the descendants of the cover. This redundant test spreading code allows the transmitter and receivers to cooperatively detect jamming on any spreading code in the cover that is an ancestor of the test spreading code. We call this ancestor code the *detectable* spreading code.

If no jammers are present, each receiver should get either one or two identical messages, the first encoded using one of the codes from the cover, and possibly a second encoded using the test code. If any receiver receives the second message without receiving the first, then he should suspect jamming on the detectable code. Any receiver detecting jamming in this manner should report that finding to the transmitter, for example by transmitting a JAMMING DETECTED message using the leaf code shared between the transmitter and the detecting receiver (because no jammer knows that leaf code). The transmitter can then spend a short time period listening for these jamming reports on the leaf codes corresponding to the set of receivers that can receive the test message. For example, if a parent code of two leaf codes was chosen as the test code, the transmitter listens on the two children leaf codes for any jamming report for some time after testing.

In some instances, jamming on the detectable code will not be detected. This can happen either when the message is also lost on the test code or when all benign users who hold the test code are absent.

Testing can be generalized so that a *set* of test codes are used at each step, thus allowing a *set* of detectable codes. For example, if the current disjoint cover in use is $\{C_{03}, C_{45}, C_{67}\}$, then the test code set of $\{C_{01}, C_4\}$ would make the detectable code set be $\{C_{03}, C_{45}\}$.

2) Response to Jamming: When a transmitter detects jamming, it will choose a different cover. In particular, if jamming is detected on some code c in the current cover, the transmitter will remove c from the cover and add the two children of c to the cover. For example, in Fig. 1, when jamming is detected on code C_{07}, the transmitter splits the cover into $\{C_{03}, C_{47}\}$. If jamming is further detected on C_{47}, the resulting cover would be $\{C_{03}, C_{45}, C_{67}\}$. To avoid false reports, jamming reports are only accepted from hosts that should know the spreading code c.

D. Loss Due to Jamming is Limited

In this section, we show that our protocol can effectively and efficiently mitigate jamming. Specifically, in an idealized environment, we show that any receiver will lose, with arbitrarily high probability, at most a finite number of messages before the jammers' effect on the system is similar to raising the noise floor. That is, after finite packet losses, any benign receiver would communicate with the transmitter using a spreading code not used by the jammers.

Theorem 3.1: For every benign receiver u and every $\varepsilon_u > 0$, there exists a finite loss limit λ_u such that u will lose at most λ_u packets with probability at least $1 - \varepsilon_u$, in the following idealized environment, before the jammer can only interfere in a noise-like fashion.

1) Jamming is successful on a code only if a jammer knows that code (i.e., jamming cannot overcome the processing gain).
2) Each subset of the tree is a possible test set (i.e., each such subset is tested with nonzero probability).
3) When only benign receivers know a test code, and jamming is perpetrated on the corresponding detectable code, at least one benign receiver detects the jamming with probability at least $p > 0$ (i.e., jamming detection sometimes works).

Proof: We prove this theorem in Appendix A. ■

IV. Tree Remerging Optimization

In our described jamming detection and response system, the size of the minimal safe cover is in the worst case on the order of the product of the number of jammers and the height of the binary tree. That is, $|C| = O(j \log_2(n))$, where j is the number of jammers and n is the number of legitimate receivers (both benign and compromised) in the broadcast system. In this section, we describe a tree remerging scheme that is equivalent to reassigning codes in order to merge two groups of receivers, where each group represents a subtree of the original code tree. Our tree remerging scheme allows a transmitter to split and reform a code tree to reduce the number of codes in the cover. Our remerging scheme is crucial in conserving the number of codes on which a transmitter must transmit simultaneously.

The intuition of our tree remerging scheme is based on the observation that if a code c is detected to be jammed, and one of its children c' is also detected to be jammed, then we can attribute the detected jamming activity on c to the same jammer that jammed c'. It may be the case that another jammer knows c'', but for the time being, we can assume code c'' is safe from jamming. Suppose we have two codes that are considered safe; then we can simply reassign a single new code for the two sets of receivers, thereby reducing the size of our minimal safe cover by 1.

We will show that even with such remerging, jamming will be detected at most $j\lceil \log_2 n \rceil$ times before the jammer can cause no more interference to the system than simply emitting noise. We will also show that a transmitter performing this remerging scheme sends on at most $2j + 1$ spread-spectrum codes simultaneously. Moreover, we show that any system that relies on using spreading codes to mitigate jamming while providing availability and termination requires the use of at least $j + 1$ codes.

A. Detailed Description

For the sake of explaining our optimization technique, instead of keeping a minimal safe cover, let each transmitter equivalently keep a set of trees $\mathcal{R}$.

The algorithm for managing the set of trees $\mathcal{R}$ is as follows. We start with the collection containing only the original tree. When jamming is detected on the root of a tree, the root is deleted, and the two children subtrees are inserted into the set of trees. The cover consists of the root of each of these trees (and is clearly disjoint). Because we insert a tree into $\mathcal{R}$ only when the parent of the root of this tree has been jammed, the roots of $\mathcal{R}$ form a minimal safe cover.

If a subtree T is in $\mathcal{R}$, but its most recent sibling $T' = \mathrm{sibling}(T)$ is not in the set of trees, then T' must contain a jammer. This is because the jamming event that caused T to be placed in $\mathcal{R}$ could also have caused T' to be placed in $\mathcal{R}$. The fact that T' is no longer in $\mathcal{R}$ indicates that jamming was detected on the code at the root of T'; that is, the root of T' is known to a jammer. It is therefore possible to remerge all sibling-free subtrees T into a new tree; Lemma 4.1 will show that even with such remerging, jamming will be detected at most $j\lceil \log_2 n \rceil$ times before the jammer can no longer affect normal users.

To ensure that the resulting remerged trees are balanced, we impose requirements on the empty subtrees contained in the tree. Any binary tree can be represented as a complete binary tree on which every leaf node is at equal depth. Some of these leaf nodes will be empty; if their sibling is also empty, the parent will likewise be empty. In general, we call a subtree *empty* if every leaf of that subtree is empty. We call a tree *full* if it contains no empty subtrees.

We represent each nonfull tree in this manner, and build it such that a tree of height h has at most one empty subtree of each height from 1 to $h-1$ when each empty subtree is counted only once (that is, a subtree of height h' is not counted as two subtrees of height $h'-1$). In addition, we do not allow an empty subtree of height $\lceil \log_2 n \rceil$ or greater, where n is the number of legitimate users. This is because there are at most n nonempty leaves in any tree.

To keep insertion efficient, we ensure that all the empty leaves are contiguous and located at the far right. This ensures that, when viewed from right to left, there is at most one empty subtree of any height (because any two empty subtrees of the same height would be next to each other and would therefore constitute an empty subtree of a height one greater than either subtree). Also, the heights of the empty subtrees are in descending order. All our algorithms will require the tree to start this way and will return the tree to this condition when complete. Our tree operations may change the ancestors of some tree nodes; in this case, we randomly generate new codes for all affected ancestor nodes and disseminate the new codes as described later in this section.

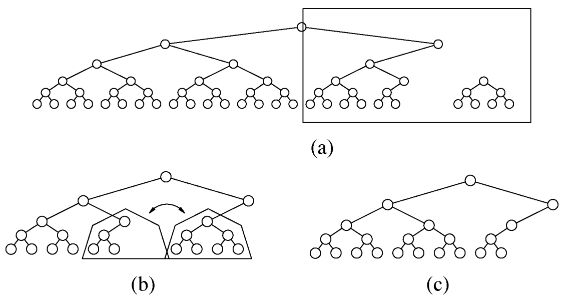


Fig. 2. Code tree is merged with the main tree. (a) Main tree and tree to be inserted. (b) and (c) show the right half of the tree in (a). (b) Code tree is merged by being inserted into the leftmost place that it would "fit." (c) Resulting main tree after subtrees are swapped to keep all nonempty nodes flush to the left.

To insert a full tree T' of height h' into a tree T of height h with e empty leaves, we first determine whether or not T' will fit (i.e., whether $e \le 2^{h'}$). If not, we increase the height of T by adding an empty subtree of height h to the right of T, making T height $h+1$ with $e+2^h$ empty leaves. We then recursively apply this algorithm until T' fits.

At this point, an empty subtree of height at least h' exists. Fig. 2(a) shows an example of a tree being merged into another tree. We insert T' into the leftmost place that it will "fit"—that is, on the left side of the smallest empty subtree of height at least h' (it may have height greater than h', in which case we will take an empty subtree of height $h'+\delta$ and turn it into δ empty subtrees of heights $h', h'+1, \ldots, h'+\delta-1$). The leftmost side of this tree will be the $e-(e \bmod 2^{h'})$th leaf, counting from the rightmost leaf. In our example, there are 10 empty leaves, and $h'=2$, so the leftmost leaf of the inserted tree is placed in the $10-(10 \bmod 2^2) = 10-2 = 8$th position counting from the right, as shown in Fig. 2(b). To maintain the property that all empty leaves reside at the right, if there are any empty leaves to the left of where T' was inserted (i.e., if $e \bmod 2^{h'} > 0$), we swap the root of T' with the node at the same level that is one to the left of the root of T'. We know that at most one such swap is necessary to ensure that all empty leaves are on the right-hand side because, if there were more than $2^{h'+1}$ empty leaves to the left of T', we would have placed T' farther left. In our example, this swap is shown in Fig. 2(c).

To insert a nonfull tree T' into a tree T, we break T' into maximally sized component subtrees and add each such subtree into T. When the root of a nonfull tree is found to have been jammed, we split the tree in half and insert both the left child and the right child into the tree set $\mathcal{R}$. If an empty subtree is in $\mathcal{R}$, we leave it in $\mathcal{R}$ until its sibling is removed from $\mathcal{R}$; at that point, we discard the empty subtree rather than remerging it.

B. *Disseminating New Codes*

When new tree nodes are created, new codes are also created. To disseminate these codes, the transmitter periodically broadcasts new ancestor codes on codes that were part of the original tree. For example, if c was a code on the original tree, and the main tree has new ancestors for c, the transmitter will broadcast the codes of those new ancestors using the code c, so that all receivers with code c can learn the new ancestor codes. Because of mobility and varying wireless propagation conditions, these broadcasts may need to be repeated periodically.

Another approach is to disseminate the current cover using the testing mechanism. When the base station transmits on a test code, the message includes all ancestor codes of that particular test code. Thus, whenever a code known to a receiver is selected for testing, that receiver is able to update the ancestor codes to which he has access. This dissemination technique is desirable since some receivers might not know the current cover and would redundantly report jamming detection when tested.

C. *Upper Bound on Number of Simultaneously Used Codes*

We call a tree $T \in \mathcal{R}$ *main* if the most recent sibling of T is not in $\mathcal{R}$. A tree set need only contain at most one main tree; otherwise, the two trees that both do not have siblings could merge. To see that a transmitter can mitigate jamming by using only $2j+1$ codes in the cover, we observe that every pair of siblings in $\mathcal{R}$ must contain at least one jammer (because jamming was detected on the parent of those two siblings). This means that excluding one main tree, there are at most another $2j$ trees in $\mathcal{R}$, for a total of $2j+1$ trees.

A transmitter might still need to use more than $2j+1$ codes due to false alarms. A false alarm happens when the message is lost due to noise on the cover but not on the test code. We will explore the false alarm issue in Section V.

Lemma 4.1: When tree remerging is done in the manner described in this section, after $j\lceil \log_2 n \rceil$ detections, the jammer is unable to jam further (that is, it will only know roots of $\mathcal{R}$ that are leaves).

Proof: We prove this lemma in Appendix B. ■

D. *Lower Bound on Number of Codes Required*

In this section, we show that any keying scheme that provides both availability and termination must use at least $j+1$ codes in the worst case, thereby showing the number of codes used simultaneously by our tree remerging protocol is a constant factor away from optimal. A broadcast system may choose to use other physical techniques to use fewer codes simultaneously [11], [12].

We define *availability* to be the property that each nonattacking receiver must eventually know at least one code in the cover, and that receiver must not have reported that code as being jammed. Intuitively, availability ensures that a nonattacking receiver is eventually able to receive each broadcast on an unjammed code. We also define *termination* to be the property that the cover eventually converges; that is, given one specific jammer strategy, there exists a time after which the transmitter makes no further changes to the cover. Our scheme provides termination: If a code c in the cover is known to a subset of receivers $\mathcal{N}_c \subseteq \mathcal{N}$ and is reported as being jammed, then never again is another code c' included in the cover when $\mathcal{N}_{c'} = \mathcal{N}_c$. Because the power set of receivers is finite, the cover eventually converges.

Theorem 4.2: In a network of n receivers where we do not *a priori* know the number of attackers, if we allow the network to exclude at most $k-1$ nonattacking receivers, then j jammers can force the use of $\lfloor \frac{j}{k} \rfloor + 1$ codes.

Proof: We prove our claim by showing two network configurations that are indistinguishable based solely on the jamming reports, thereby showing that it is impossible to identify and selectively stop service to a set of jammers using jamming reports only. Since jammers may report each other, the lower bound on the number of spreading codes used simultaneously is intuitively achieved by letting all benign receivers share one spreading code and letting every k jammers share one spreading code.

In the first network configuration, which we call network A, we consider j jammers $a_1, a_2, \ldots, a_j$; these jammers divide into sets of k elements $s_1 = \{a_1, \ldots, a_k\}, s_2 = \{a_{k+1}, \ldots, a_{2k}\}, \ldots, s_{\lfloor \frac{j}{k} \rfloor}$. All jammers always jam, and a jammer $a_f \in s_g$ reports only when the detectable code c is known to a jammer that is not in s_g. Let R_A denote the set of reports made in network A, so that $(N, r) \in R_A$ if and only if receiver N made a report r in network A.

In the second network configuration, which we call network B, we consider k benign receivers $s'_g = \{a'_{gk+1}, \ldots, a'_{(g+1)k}\}$ with the rest being jammers who jam continuously, and report exactly that subset of R_A that they can report. (Specifically, if $(N, r) \in R_A$ and $N \notin s'_g$, then N reports r in network B, so $(N, r) \in R_B$.) Furthermore, because every cover element is jammed, a benign receiver $a'_f \in s'_g$ will report jamming on detectable code c whenever c is known to any of the jammers (that is, known to a receiver not in s'_g).

The set of reports R_B made in network B is exactly equal to the set of reports R_A made in network A, so networks A and B are indistinguishable to the transmitter. Furthermore, every element of s_g is a jammer, but every element of s'_g is a benign receiver, and each set is indistinguishable from every other set. Therefore, the transmitter cannot exclude any set s_g or s'_g, and to ensure availability, the transmitter must send on a different code for each s_g, and send on one additional code for all the benign receivers, for a total of $\lfloor \frac{j}{k} \rfloor + 1$ codes. ■

We require availability for every single nonattacking receiver in our broadcast system, thus $k = 1$, and the minimum number of codes required for j attackers is $j + 1$. The requirement for termination means that R_A and R_B are finite, and therefore the attacker can force the use of $j + 1$ codes in finite time. This shows that our scheme, which uses $2j + 1$ codes in the worst case, is within a factor of two of optimal in terms of the *number of codes used simultaneously*.

V. False Alarms and Power Allocation

Due to the probabilistic nature of packet reception, a signal may be lost even when there is no jamming. When a jammer raises the interference level, the probability of loss increases further. Such losses raise the prospect of *false alarms*: Because our scheme detects jamming when the detectable code is lost and the test code is received, the loss of an unjammed detectable code could result in a false alarm. False alarms result in a larger-than-necessary cover since $J \not\subseteq \kappa$, where κ is the set of nonleaf codes known to jammer.

One option for coping with the effects of false alarms while achieving higher power efficiency is to periodically empty J. Emptying J resets the state of our system, allowing jammers to deny service again. In particular, resetting J also resets the loss counter in Section III-D, so that rather than bounded losses across the entire lifetime of the system, the reused protocol only limits the number of losses over a fixed period of time between resets. In the rest of this section, we explore the tradeoff between false alarms and missed detections in the context of power allocation, and show that parameters exist for which a low false alarm and a reasonable detection rate are achievable, which increases the amount of time between when J needs to be emptied.

We define the false-alarm rate as the probability of detecting jamming on a code when the jammer is jamming but does not jam that particular code; and we define the detection rate as the probability of detecting jamming on a code when the jammer jams that code.

For the following analysis, let the length of codes be η. Using the DS-CDMA decoding rule presented in Section II-A, the intended transmitted signal is a fixed value after decoding, namely η times the signal power of each chip, p_s. We model the noise as additive white Gaussian noise (AWGN); after decoding, the noise is simply a sum of independent zero-mean Gaussian variables, resulting in another zero-mean Gaussian variable with larger variance $N_0\eta$, where N_0 is the noise level. The jamming signal, without knowledge of the pseudorandom code, after decoding is a sum of independent Bernoulli random variables equiprobable in -1 and 1. However, a sum of Bernoulli variables is binomially distributed and can be modeled as a Gaussian variable, namely a zero-mean Gaussian variable with variance $p_j\eta$, where p_j is the power of the jammer.

Thus, if we let the length of codes be η, and the signal-to-interference-plus-noise power ratio be $\text{SINR} = \frac{p_s}{p_j + N_0}$, then the message bit, normalized with respect to the signal power, is approximately distributed $\approx N(\eta, \eta(\frac{1}{\text{SINR}}))$.

Whenever the random variable is negative, the bit is decoded incorrectly. Thus, the bit error probability is $P_e^{\text{bit}} \simeq Q(\sqrt{\eta \text{SINR}})$.

Since a transmitter in our broadcast system uses a set of spreading codes simultaneously, we provide careful treatment on the definitions of signal-to-interference-plus-noise ratio (SINR) when different signals are considered. We let SINR be the ratio between the *total* transmission power and interference power. We then let $\text{SINR}^{\text{cover}}$ be the ratio between the signal power on a particular *detectable code in the cover* and the interference power, taking into account self-interference. Finally, we let $\text{SINR}^{\text{test}}$ be the ratio between the signal power on a particular *test code* and the interference power, taking into account self-interference.

If we allocate a fraction of $1 - r$ of the transmission power to a test code and the remaining r to the corresponding detectable code, then the equivalent SINRs, taking self-interference into account, are

$$\text{SINR}^{\text{cover}} = \frac{r}{\frac{1}{\text{SINR}} + 1 - r}$$
$$\text{SINR}^{\text{test}} = \frac{1 - r}{\frac{1}{\text{SINR}} + r}.$$

To prevent a jammer from learning the spreading code by correlating messages, we use time-varying spreading codes so that each bit is spread using a different code. Li *et al.* propose a protocol in which the transmitter and receiver keep changing their spreading code by using a shared secret key and the AES protocol to encrypt the spreading code from previous round [5]. Without the shared secret key, the spreading codes would appear independent as guaranteed by the AES encryption.

Assuming each bit is encoded with independent spreading codes, we can calculate the probability that a message of length ℓ bits is successfully received. The false-alarm rate P_{F} can also be calculated

$$\begin{aligned} P_{\mathrm{d}}^{\mathrm{cover}} &= \left(1 - Q\left(\sqrt{\eta \mathrm{SINR}^{\mathrm{cover}}}\right)\right)^{\ell} \\ P_{\mathrm{d}}^{\mathrm{test}} &= \left(1 - Q\left(\sqrt{\eta \mathrm{SINR}^{\mathrm{test}}}\right)\right)^{\ell} \\ P_{\mathrm{F}} &= \left(1 - P_{\mathrm{d}}^{\mathrm{cover}}\right)\left(P_{\mathrm{d}}^{\mathrm{test}}\right) \end{aligned}$$

where $P_{\mathrm{d}}^{\mathrm{cover}}$ is the probability that the cover message is received, and $P_{\mathrm{d}}^{\mathrm{test}}$ is the probability that the test message is received.

To calculate the detection probability, we assume the jammer's message is random. That is, with probability $2^{-\ell}$, the jammer helps the system by transmitting an identical message. Thus, $P_{\mathrm{d}}^{\mathrm{cover}} \approx 1 - 2^{-\ell}$, and the detection rate is approximately $P_{\mathrm{D}} \simeq (1 - 2^{-\ell}) P_{\mathrm{d}}^{\mathrm{test}}$. We performed a Monte Carlo simulation for the case where $r = 0.25, 0.27, 0.29, \ldots, 0.75$, $p_j = 10$, $N_0 = 10$, and $\eta = 255$. The results were consistent with those predicted by above analysis and are shown in Fig. 3. Each point represents 25 runs of 25 000 messages, each 8 bits long; we plot the mean and 95% confidence intervals. Because each data point represents two variables (probability of detection and probability of false alarm), each point includes two error bars that indicate the confidence interval of each variable. For clarity, the x- and y-scales of this figure are different. We use one data point to show that the false-alarm rate can in fact be much smaller than the detection rate. At $r = 0.75$, the false-alarm rate is around 0.61%, while detection rate is around 71.9%.

Since the false-alarm rate significantly impacts the frequency our protocol needs to be reset, the optimal message length can be determined by choosing ℓ and the power ratio r so that the false-alarm rate is at or below the tolerable threshold and provides the highest detection rate.

VI. Evaluation

In this section, we describe the results of two simulation-based evaluations. In the high-level simulation, we assume the codes are completely orthogonal, and precisely model the received signal and noise levels after processing gain, using the resulting signal-to-noise ratio to select a bit error rate. In the low-level simulation, we model the use of random codes and tree remerging in order to ensure that the random codes provide sufficient interference cancellation, and that the tree remerging scheme does not substantially reduce the effectiveness of jammer detection.

A. High-Level Simulation Methodology

We simulate our basic jamming mitigation scheme without the remerging optimization in a high-level simulation that is based on the *ns-2* discrete-event network simulator [22]. We implement the following features into *ns-2*: simultaneous sending

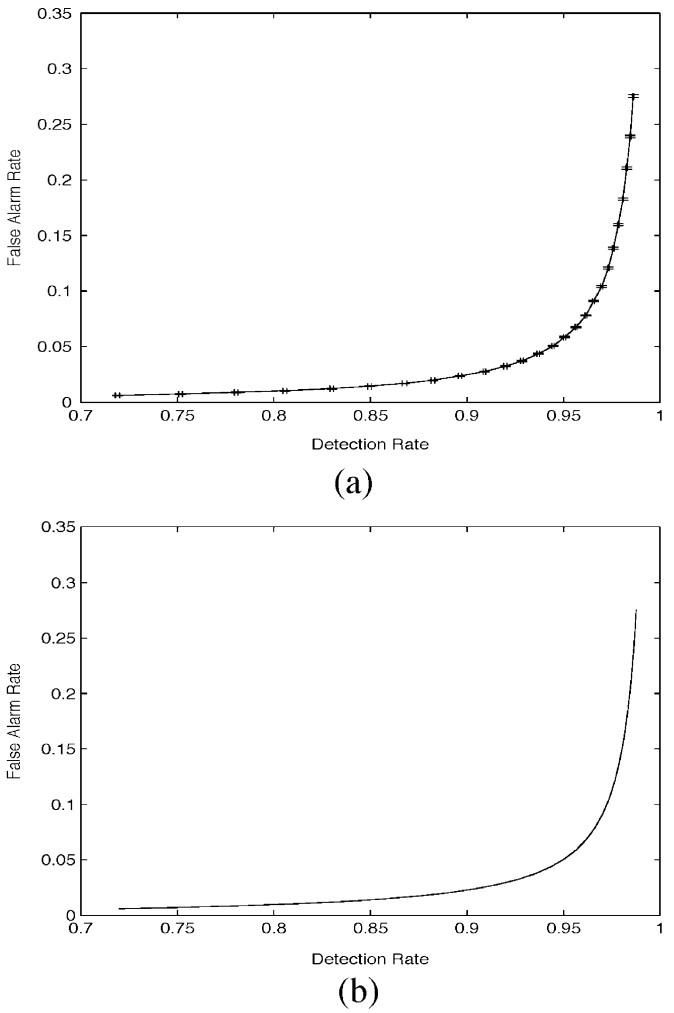


Fig. 3. False-alarm rate versus detection rate with different power allocation between a detectable code and a test code. For clarity, the x- and y-scales are different. (a) Simulated false-alarm rate versus detection rate. (b) Theoretical false-alarm rate versus detection rate.

of a single packet on multiple codes and simultaneous reception of multiple packets on multiple codes. We conduct our simulations on a 1500×300 m^2 area with 50 receivers.

For each incoming packet, we compute the maximum interference power experienced during the reception of the packet, including power contributed by any outgoing packets, the jammers, other incoming packets, and ambient noise (conservatively computed as half of the IEEE 802.11b carrier sense threshold from *ns-2*). We then compute the signal-to-noise ratio, including processing gain from the original packet, the energy cost of simultaneously sending on multiple codes, and any processing gain that the jammers receive as a result of transmitting on the same code. Based on this signal-to-noise ratio and the modulation scheme in use, we determine the resulting symbol error rate p. We then compute the probability that the packet would be lost, which is given by $1 - (1 - p)^{\frac{s}{b}}$, where s is the length of the packet in bits and b is the number of bits per symbol, and accept the packet with this probability.

We choose processing gains from 16 to 1024 to demonstrate the tradeoffs in choice of processing gain. Our modulation scheme is QPSK; we also consider 16-QAM and 64-QAM in preliminary experiments since those modulation schemes are used for higher data rates (those above 18 Mb/s) in IEEE 802.11a [6], [23]. However, preliminary experiments show that decreasing the processing gain and bits-per-symbol proportionally so as to achieve the same data rate would result in better performance, so we use QPSK for all of our experiments.

We determine our bandwidth based on an 80-MHz-wide channel, equivalent to four 802.11a channels. This should be achievable with modern hardware since 802.11a cards using two channels simultaneously have been on the market for several years. In addition, the 5.8-GHz ISM band used by 802.11a provides for 12 nonoverlapping channels. A modulation scheme that gives us b bits per symbol combined with processing gain η gives us a bit rate of $\frac{b \cdot 40 \times 10^6}{\eta}$. In QPSK, $b = 2$.

Our communications pattern is 20 broadcast streams, each sending four packets per second containing 512 bytes of data. This traffic rate could be used for highly compressed audio or for navigation data. We select our malicious jammers from receivers that were not transmitting in our communications pattern, thereby avoiding the possibility that a jammer might jam its own packets. We perform experiments for a number of jammers between 0 and 10.

We consider five jammer strategies in which jammers *collude* and share all codes in κ among themselves. Each strategy is executed on a per-packet basis. In each strategy, the attacker determines the subset of its codes on which the transmitter transmits. The jammer then allocates its power between that subset of codes. This model reflects a particularly strong attacker model where the attacker knows the subset of compromised codes currently in use; in practice, a jammer may not be able to hear each packet that it attempts to jam, and thus cannot determine the set of codes in use. In the first strategy, the jammer jams on a code that is orthogonal to all transmitted codes. This represents a best-case scenario, which cannot be improved upon by any code selection algorithm. We call this the "Noise Only," and any jammer that has compromised any receiver should be able to perform better. The second strategy represents a jammer that jams on a single code per transmission and chooses a code to jam that covers the most potential victims (that is, one that is closest to the root of the tree). In the event that multiple codes are closest to the root, it chooses only one such code. We call this the "Best Code" jammer because it allocates all of its power to the single best code. The third strategy is to jam all codes in that subset with equal amounts of power, which we call the "All Codes" strategy. The fourth strategy is to proportionally share the jamming power among all codes in the subset, where each code gets an amount of power proportional to the number of users potentially affected. We call this the "Prop Fair" strategy. The fifth strategy is the impossible (and worst-case) strategy where the jammer provides full jamming power to *each* code in the subset, which we call the "Worst Case" strategy. We also consider the "Worst Case" strategy when the sender uses no test codes, which we call "Ignore Jammer."

We do not use the MAC protocol from 802.11 because 802.11 defers transmission when it senses another transmission. A jammer within the carrier sense range of any device, then, would be able to prevent that device from transmitting, effectively preventing communication. We choose to use a simple (null) MAC protocol that passes each packet without modification, buffering, or delay between the network layer and the physical layer. All computations to determine collision and transmission delay are performed in the physical layer of the simulator.

TABLE II
SUMMARY OF SIMULATION PARAMETERS

Simulator	*ns-2*
Mobility Pause Time Maximum Speed	Random Waypoint 0 20 m/s
Traffic Pattern Packet Size	20 CBR sources, 4 packets/s 512 bytes of data per packet
MAC Protocol	None
Modulation Scheme Processing Gain (η)	QPSK 16, 32, 64, 128, 256, 512, 1024
Simulated Time Runs	2000 s per run 10 per parameter set

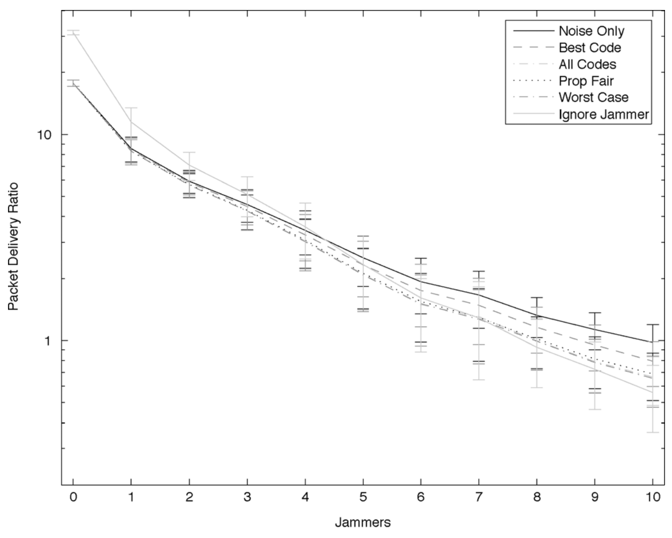
For simplicity of evaluation, in our simulation, each jammer transmits at a power level equal to the power level of a legitimate transmitter. However, our results also show what happens at increased jammer power. In particular, if a jammer transmits with twice as much power, the performance will be no worse than if there were two jammers at the same location because the aggregate jamming power level will be the same, but in the latter case, the jammers will know more about the code tree.

For each processing gain, jamming strategy, and number of jammers, we perform 10 simulation runs of the described scheme, each of which represents 2000 simulated seconds. We summarize these parameters in Table II.

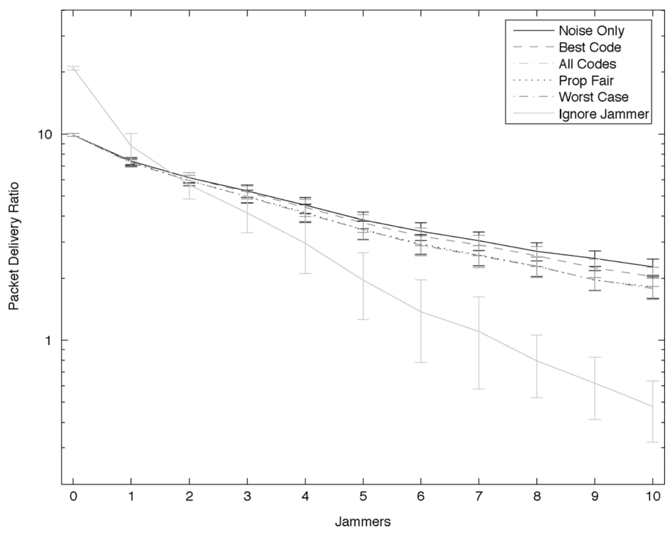
B. High-Level Simulation Results

Fig. 4 shows the results of our simulation runs. Each graph shows the *packet delivery ratio* (PDR), which is the number of packets received divided by the number of packets sent. Each PDR result is averaged over 10 simulation runs; the error bars represent the 95% confidence interval of this average. Because our communications pattern is broadcast, a single transmitted packet may be received by several different receivers. In particular, out of 50 receivers, any receiver other than the transmitter and the jammers could potentially receive the packet. However, because not all receivers are within wireless transmission range, the packet delivery ratio cannot reach 49, even when there are no jammers.

Fig. 4(a) shows the performance of our scheme when $\eta = 32$. At the left side of the graph, where there are few jammers, ignoring the jammer substantially outperforms the other four schemes; this is because when ignoring the jammer, no test codes are sent, increasing the effective transmission power and thereby increasing range. As the number of jammers increases, the strategy of ignoring the jammer becomes less advantageous; however, the difference between ignoring the jammer and avoiding codes used by the jammer remains slight, even at 10 jammers. This is because avoiding jammers requires spreading one's transmission power across multiple codes. At $\eta = 32$, the improved jamming rejection (a factor of 32) does not substantially overcome the reduced effective transmission power. The ineffectiveness of small η on jamming rejection is



(a)



(b)

Fig. 4. Packet delivery ratio of our high-level simulation using different lengths of spreading code and jamming strategy. (a) $\eta = 32$. (b) $\eta = 512$.

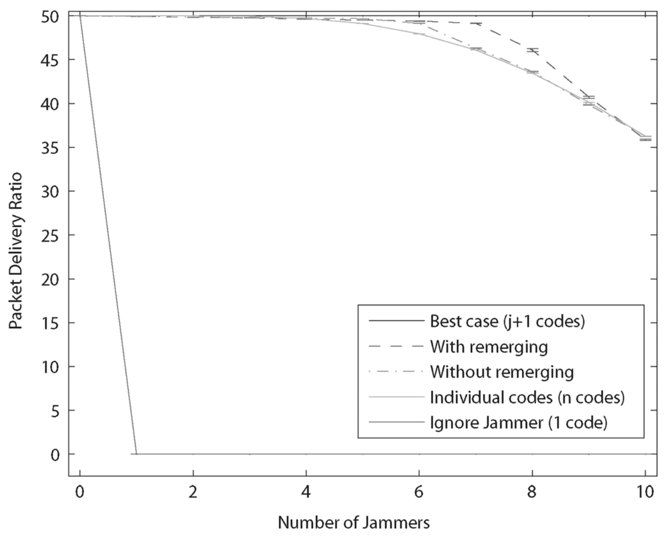


Fig. 5. Packet delivery ratio of our low-level simulation with different jammer strategy.

even more pronounced when $\eta = 16$. Though we do not present the results in detail due to space limitations, in small η scenarios, ignoring the jammer outperforms all other approaches, even with 10 jammers.

Fig. 4(b) shows the performance of our scheme when $\eta = 512$. As when $\eta = 32$, the "Ignore Jammer" approach provides substantial benefits when there are no jammers. However, as the number of jammers increases, our scheme provides substantial advantages regardless of jammer strategy. In fact, when $\eta = 512$, our scheme provides *almost all of the performance improvement possible* because any scheme choosing CDMA codes must necessarily perform below the "Noise Only" line, and any reasonable scheme will perform above the "Ignore Jammer" line for a sufficiently large η and number of jammers. (When forward error correction (FEC) is used, higher delivery rates can be achieved, but the use of FEC is orthogonal to the approach described here and is therefore applicable to all schemes, including ours and "Noise Only.")

C. Low-Level Simulation Methodology

In order to determine the effectiveness of using random codes and the tree remerging scheme, we implement our scheme including these two elements (which are missing from the high-level simulation above) in MATLAB. The simulation scenario consists of one base station, 0–10 collocated jammers, and 50 benign receivers that are all equidistant from the transmitter and the jammers. Each jammer is able to emit power equal to 10 times the total transmission power of the transmitter; as before, jammers that emit more power can be modeled by increasing the number of jammers. To minimize the effect of false alarm, we assumed a noise level that is -10 dB compared to the base station power. We simulate FFH-CDMA with 256 channels and 127 hops per bit across those channels. In this simulation, jammers do not collude (the high-level simulation considers collusion); rather, each jammer simply jams on its hopping pattern in the cover. For each number of jammers, we perform 10 tests in which 3840 256-bit packets[1] are transmitted by the base station.

D. Low-Level Simulation Results

Fig. 5 shows the results of our low-level simulation. Again, we compute the packet delivery ratio by dividing the number of packets received by the number of packets sent. For each jamming strategy and number of jammers, we plot the average and 95% confidence interval of the packet delivery ratio. Because we have 50 benign receivers in each scenario (in addition to the transmitter and jammers), and because all benign receivers are within wireless transmission range of the transmitter, the best possible result is a packet delivery ratio of 50. This graph shows that for less than six jammers, each of whom is able to transmit at 10 times of the power of the transmitter, our schemes provide superb packet delivery. Our basic tree coding scheme offers some benefit over naïvely transmitting using individual hopping patterns. Our remerging scheme then provides additional benefits

[1] The value 3840 is chosen by multiplying 10, the size of the system (64 users, out of which 50 are normal, 0–10 are jammers, and the rest are absent), and the log of the size of the system in base 2 (i.e., the height of the code tree).

that allow us to operate at roughly 10% higher PDR when there are seven jammers in the system than assigning each receiver individual hopping pattern. The effect of tree remerging is greatest for a medium number of jammers. When the number of jammers is small, the number of codes used simultaneously does not yet substantially affect the ability of receivers to decode the packet; when the number of jammers is large, the system suffers from high interference, and false alarm causes both of our systems to use many codes that are not actually jammed, substantially increasing the size of the cover, and to eventually degrade into the case of individual code assignment.

The performance results are consistent with the expected result. The best-case line reflects the use of $j+1$ codes. Without considering false alarms, our remerging scheme uses $2j+1$ codes, our basic scheme uses $j\lceil \log_2 n\rceil$ codes, and the scheme that allocates each user a different pattern uses n codes. Our scheme provides statistically significant improvements with 50 normal receivers, and we expect the improvements to grow if the network had more receivers since the number of codes in the cover at any time in our remerging key scheme is constant with respect to the number of benign receivers.

These results are somewhat better than those of the high-level simulation for four reasons. First, every receiver is within the transmission range of the transmitter, thus the performance is better even without any jammers. Second, jammer and transmitter powers are well balanced at each receiver (that is, there are no near–far problems that overwhelm certain victims with excessive jamming power). Third, jammers do not collude, which can reduce the effectiveness of jamming. Finally, this simulation uses much shorter packets, reducing the packet error rate given the same symbol error rate.

VII. Conclusion

Due to the symmetric nature of spread-spectrum codes and the inverse relationship between signal power and number of codes used simultaneously, it is difficult to extend the jamming resilience of spread-spectrum techniques from a point-to-point wireless communication system to a broadcast wireless communication system. In this paper, we provide a protocol that allows a broadcast communication system to dynamically change the spreading codes used by subsets of receivers so that some benign users can share a single spreading code, thereby conserving the number of spreading codes used simultaneously.

We show a lower bound on the number of spreading codes used simultaneously in order to mitigate jamming by relying only on keying and not other physical characteristics. We optimize our protocol so that it can mitigate jamming by using twice as many spreading codes simultaneously as the lower bound. We present simulation results to support our theoretical results and show that jamming can be effectively mitigated in a broadcast wireless system.

Appendix A
Proof of Theorem 3.1

Proof: As in Section III-B, we define J_m to be the set of codes on which jamming has been detected when the mth message is sent, and C_m to be the minimal safe disjoint cover relative to J_m. Also, we define κ to be the set of nonleaf codes known to jammers. Finally, we define H_m to be the set of historic events before the mth message is sent. Given a leaf code d unknown to jammers, a jammer cannot determine whether or not a message is sent using such code because the jammer cannot decode the message from the signal.

The mth message is then sent on a set of test codes T_m and a set of cover codes C_m. Because C_m is a minimal cover, the normal receiver u must know one cover code in C_m. If the jammers do not know this code, then jamming cannot be successful (Assumption 1), and receiver u loses no more messages; otherwise, with probability $P_j[m,u|H_m]$, the jammers decide to jam such a packet.

Let d be the leaf code of receiver u, and let $P[d\in T_m|H_m]$ denote the probability that d is chosen as a test code when the mth message is sent. Since jammers cannot determine whether d is used, $P[d\in T_m|H_m]$ is independent to the jammers' knowledge and observations. Therefore, if jammers decide to jam a particular code in the cover, such action will be detected with probability at least

$$D_m^u = p\cdot P[d\in T_m|H_m]\cdot P_j[m,u|H_m].$$

We do not consider the probability of false alarm $P_{\rm f}$: By Assumption 1, a jammer cannot successfully jam a detectable code without knowing that code, thus by our definition of false alarm in Section V, $P_{\rm f}=0$. Since $p>0$ by Assumption 3, and $P[d\in T_m|H_m]>0$ by Assumption 2, then $D_m^u=0$ if and only if $P_j[m,u|H_m]=0$. In other words, if the jammer decides to jam the mth packet with nonzero probability, the jammer would be detected also with nonzero probability.

Thus, for benign receiver u, the probability that messages $m_1, m_2, \ldots, m_{N_u}$ are lost *due to jamming* without detection is

$$\xi_u=\prod_{i=1}^{N_u}(1-D_{m_i}^u)\ \to 0\quad \text{as } N_u\ \to\infty.$$

Since the system removes a code every time jamming is detected, jamming can only be detected a finite number of times before $J_m\supseteq\kappa$, and the jammers can do no more harm to the system than random jamming. Therefore, there exists a finite number Δ_u such that after being detected Δ_u times, jammers can no longer jam receiver u. That is, a receiver u will lose at most $\lambda_u=\Delta_u N_u$ messages with probability $\varepsilon_u=(\xi_u)^{\Delta_u}$. ■

Appendix B
Proof of Lemma 4.1

Proof: Given a set of jammers, we define a *cost* and *potential* for a tree set $\mathcal{R}$. The cost of a single tree is the height of the tree times the number of jammers in the tree, and the cost of a tree set is the sum of the costs of the trees in the set. For any tree T, we denote its height as H_T, and the number of jammers in T as $|T^j|$. The cost of the tree set is then

$$\text{cost}(\mathcal{R})=\sum_{T\in\mathcal{R}}H_T\cdot|T^j|.$$

The potential of a tree in $\mathcal{R}$ depends on whether or not the sibling of a tree is in $\mathcal{R}$. If the tree is main (that is, it does not have a sibling in $\mathcal{R}$), its potential is

$$\text{potential}(T) = (\lceil \log_2 n \rceil - H_T) \cdot |T^j|$$

where n is the total number of legitimate receivers. Otherwise, the tree T and its sibling T' together have potential

$$\text{potential}(T, T') = (\lceil \log_2 n \rceil - H_T) \cdot (|T^j| + |T'^j| - 1).$$

The potential of a main tree allows it to grow to a height of $\lceil \log_2 n \rceil$ and shrink as far as possible without change to the main tree's contributions to the sum of cost and potential: When the tree increases in height by 1, then potential drops by the number of jammers in the tree, but cost increases by the number of jammers in the tree. Also, since there will never be more than n receivers in the tree, the height must be at most $\lceil \log_2 n \rceil$, so the potential of this tree can never be negative. For the rest of this proof, whenever we manipulate this tree, we assume that it is of height $\lceil \log_2 n \rceil$. In essence, whenever we manipulate this tree, we grow it to a height of $\lceil \log_2 n \rceil$, perform the operations we need, and shrink it back down to a tree of the correct height. These operations do not change the cost plus potential of the tree.

We now show that each time jamming is detected, the cost plus potential is reduced by at least one. If this is the case, then jamming can only be detected $j\lceil \log_2 n \rceil$ times. This is because the tree starts with $j\lceil \log_2 n \rceil$ cost and 0 potential, and neither cost nor potential can be negative from our definitions.

When the main tree $T \in \mathcal{R}$ is split due to jamming on its root code, we insert the two subtrees T_{L} and T_{R} into $\mathcal{R}$. Thus

$$\begin{aligned}\text{cost}(T) &= H_T|T^j| \\ \text{potential}(T) &= (\lceil \log_2 n \rceil - H_T)|T'^j| \\ \text{cost}(T_{\text{L}}, T_{\text{R}}) &= (H_T - 1)|T^j| \\ \text{potential}(T_{\text{L}}, T_{\text{R}}) &= (\lceil \log_2 n \rceil - (H_T - 1))(|T^j| - 1).\end{aligned}$$

The old cost plus potential is $\lceil \log_2 n \rceil |T^j|$, and the new cost plus potential is $\lceil \log_2 n \rceil (|T^j| - 1) + (H_T - 1)$; the total change in cost plus potential is thus $-\lceil \log_2 n \rceil + H_T - 1$. But $H_T \leq \lceil \log_2 n \rceil$, so $-\lceil \log_2 n \rceil + H_T - 1 \leq -1$. This shows that the cost plus potential is reduced by at least 1. In Fig. 6(a), the original cost was $4 \cdot 3 = 12$, and potential was 0. The new cost is $3 \cdot 3 = 9$, and potential is $1 \cdot (3 - 1) = 2$. The total cost plus potential is reduced by 1.

When a nonmain tree $T \in \mathcal{R}$ is split due to jamming on its root code, we consider also the effect on $T' = \text{sibling}(T)$. Now $T' \in \mathcal{R}$ because T is nonmain. Let $\beta = |T^j| + |T'^j|$. Once T is split, T' will become a main tree or merge with a main tree, which means that its cost plus potential will become $\lceil \log_2 n \rceil |T'^j|$, and the current cost of T' is $H_T |T'^j|$. When T is split, then, T' requires an addition of $\lceil \log_2 n \rceil |T'^j| - H_T |T'^j| = (\lceil \log_2 n \rceil - H_T)|T'^j|$ potential. The current potential of $\{T, T'\}$ is $(\lceil \log_2 n \rceil - H_T)(\beta - 1)$.

Since jamming is detected on the root of T, there must be at least one jammer in T; that is, $\beta - 1 \geq |T'^j|$. Thus, the current potential of $\{T, T'\}$ is at least $(\lceil \log_2 n \rceil - H_T)|T'^j|$.

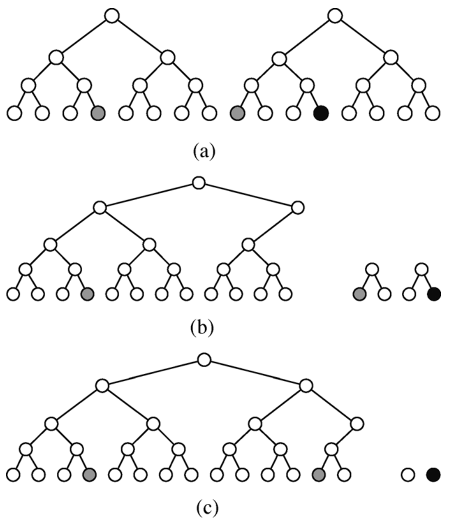


Fig. 6. Illustration of potential and cost of merging trees (from the proof of Lemma 4.1). Darkened nodes correspond to jammers. In this illustration, after jamming is detected $\log_2 n$ times, the solid jammer is isolated. (a) Jamming on root of depth 4 tree detected. (b) Jamming at height 2 detected, sibling remerged. (c) Jamming at height 1 detected, sibling remerged.

After providing the necessary potential to T', the remaining old potential is $(\lceil \log_2 n \rceil - H_T)(\beta - |T'^j| - 1)$. Splitting T into T_{L} and T_{R} decreases the cost by 1 for each jammer in T, so the total reduction in cost is $\beta - |T'^j|$. Of this reduction, we allocate $\beta - |T'^j| - 1$ to increase the potential to $(\lceil \log_2 n \rceil - H_T + 1)(\beta - |T'^j| - 1)$, which is the new potential of T_{L} and T_{R}, given the new height $(H_T - 1)$ and the number of jammers in $T(\beta - |T'^j|)$. The last unit of cost reduction is allocated to ensure that the cost plus potential is reduced by at least 1. For example, in Fig. 6(c), jamming is detected at a height of 2, causing a subtree of height 1 to remerge [the previous tree is shown in Fig. 6(b)]. The original cost was $4 \cdot 1 + 1 \cdot 2 = 6$, and potential was $(4 - 1)(2 - 1) = 3$. After jamming is detected, the new cost is $4 \cdot 2 + 0 \cdot 1 = 8$, and the new potential is 0, so the cost plus potential is reduced by 1. ■

ACKNOWLEDGMENT

The views and conclusions contained here are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ARO, the University of Illinois, or the U.S. Government or any of its agencies.

REFERENCES

[1] J. T. Chiang and Y.-C. Hu, "Cross-layer jamming detection and mitigation in wireless broadcast networks," in *Proc. 13th Annu. ACM MobiCom*, Montréal, QC, Canada, 2007, pp. 346–349.

[2] J. T. Chiang and Y.-C. Hu, "Dynamic jamming mitigation for wireless broadcast networks," in *Proc. 27th IEEE INFOCOM*, Phoenix, AZ, Apr. 2008, pp. 1211–1219.

[3] R. L. Pickholtz, D. L. Schilling, and L. B. Milstein, "Theory of spread spectrum communications—A tutorial," *IEEE Trans. Commun.*, vol. COM-30, no. 5, pt. 2, pp. 855–884, May 1982.

[4] V. Kawadia and P. R. Kumar, "Power control and clustering in ad hoc networks," in *Proc. IEEE INFOCOM*, San Francisco, CA, Mar. 2003, vol. 1, pp. 459–469.

[5] T. Li, J. Ren, Q. Ling, and A. Jain, "Physical layer built-in security analysis and enhancement of CDMA systems," in *Proc. IEEE MILCOM*, Atlantic City, NJ, Oct. 2005, pp. 956–962.

[6] *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Standard 802.11-1997, IEEE Computer Society LAN MAN Standards Committee, 1997.

[7] J. S. Lee, "Overview of the technical basis of Qualcomm's CDMA cellular telephone system design: A view of North American TIA/EIA IS-95," in *Proc. 9th IEEE ICCS*, Singapore, Nov. 1994, vol. 2, pp. 353–358.

[8] D. N. Knisely, S. Kumar, S. Laha, and S. Nanda, "Evolution of wireless data services: IS-95 to CDMA2000," *IEEE Commun. Mag.*, vol. 36, no. 10, pp. 140–149, Oct. 1998.

[9] A. J. Viterbi, *CDMA Principles of Spread Spectrum Communication*. Reading, MA: Addison-Wesley, 1995.

[10] B. Widrow, P. Mantey, L. Griffiths, and B. Goode, "Adaptive antenna systems," *Proc. IEEE*, vol. 55, no. 12, pp. 2143–2159, Dec. 1967.

[11] W. Bahn, L. Baird, and M. Collins, "The use of concurrent codes in computer programming and digital signal processing education," *J. Comput. Sci. Colleges*, vol. 23, no. 1, pp. 174–180, 2007.

[12] M. Strasser, S. Capkun, C. Pöpper, and M. Cagalj, "Jamming-resistant key establishment using uncoordinated frequency hopping," in *Proc. IEEE Symp. Security Privacy*, Berkley, CA, May 2008, pp. 64–78.

[13] T. Jin, G. Noubir, and B. Thapa, "Zero pre-shared secret key establishment in the presence of jammers," in *Proc. 10th ACM MobiHoc*, New Orleans, LA, 2009, pp. 219–228.

[14] R. C. Merkle, "Secure communications over insecure channels," *Commun. ACM*, vol. 21, no. 4, pp. 294–299, 1978.

[15] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[16] W. Diffie and M. E. Hellman, "Privacy and authentication: An introduction to cryptography," *Proc. IEEE*, vol. 67, no. 3, pp. 397–427, Mar. 1979.

[17] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler, "Spins: Security protocols for sensor networks," *Wireless Netw.*, vol. 8, no. 5, pp. 521–534, 2002.

[18] M. G. Kuhn, "An asymmetric security mechanism for navigation signals," in *Proc. 6th Inf. Hiding Workshop*, Berlin/Heidelberg, 2004, vol. 3200, Lecture Notes in Computer Science, pp. 239–252.

[19] T. X. Brown, J. E. James, and A. Sethi, "Jamming and sensing of encrypted wireless ad hoc networks," in *Proc. 7th ACM MobiHoc*, Florence, Italy, 2006, pp. 120–130.

[20] W. Xu, W. Trappe, Y. Zhang, and T. Wood, "The feasibility of launching and detecting jamming attacks in wireless networks," in *Proc. 6th ACM MobiHoc*, Urbana-Champaign, IL, 2005, pp. 46–57.

[21] C. K. Wong, M. Gouda, and S. S. Lam, "Secure group communications using key graphs," *IEEE/ACM Trans. Netw.*, vol. 8, no. 1, pp. 16–30, Feb. 2000.

[22] L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Y. Xu, and H. Yu, "Advances in network simulation," *IEEE Computer*, vol. 33, no. 5, pp. 59–67, May 2000.

[23] *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, High-Speed Physical Layer in the 5 GHz Band*, IEEE Std 802.11a-1999, IEEE Computer Society LAN MAN Standards Committee, 1999.

Jerry T. Chiang (S'10) received the B.S. degree in electrical engineering from the University of Washington, Seattle, in 2005, and is currently a Ph.D. candidate in the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Urbana.

His current research interests include denial-of-service attack mitigations in the lower layers of wireless networks.

Yih-Chun Hu (M'05) received the B.S. degree in computer science and pure mathematics from the University of Washington, Seattle, in 1997, and the Ph.D. degree in computer science from Carnegie Mellon University, Pittsburgh, PA, in 2003.

He is an Assistant Professor with the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Urbana. In his thesis work at Carnegie Mellon University, he focused on security and performance in wireless ad hoc networks. After receiving the Ph.D. degree, he worked as a Post-Doctoral Researcher with the University of California, Berkeley, doing research in the area of network security. His research interests include systems and network security.