

# Fundamental Limits on Secure Clock Synchronization and Man-In-The-Middle Detection in Fixed Wireless Networks

Jerry T. Chiang   Jason J. Haas   Yih-Chun Hu   P. R. Kumar   Jihyuk Choi

Department of Electrical and Computer Engineering  
University of Illinois at Urbana-Champaign  
{chiang2, jjhaas2, yihchun, prkumar, jchoi43}@illinois.edu

**Abstract**—In this paper we present fundamental results on secure clock synchronization and man-in-the-middle detection using only timing information. Under the assumption of affine clocks, we present a clock synchronization protocol that can operate on any channel on which data can be sent. We present a clock synchronization protocol from the literature and add verification steps on top of this protocol. These verification steps force man-in-the-middle attackers, who want to delay traffic between the endpoints and yet remain undetected, to impose only constant delays on packets. In a special case, we show that it is possible to identify and ignore attacker-delayed packets. We then show three different types of attackers: a half-duplex attacker that can always be caught using timing information alone, a double full-duplex attacker that can never be caught using only timing information, and a full-duplex attacker whose capability to perform man-in-the-middle attacks depends on its location relative to the endpoints and on the turnaround times of the endpoints. In particular, we prove that certain attackers are impossible to detect using only timing, and we construct defensive protocols that prevent all other man-in-the-middle delay attacks. A particularly noteworthy result is that a single attacker using the same radio technology as the endpoints can never successfully perform a man-in-the-middle attack to delay traffic. These results form a lightweight man-in-the-middle attack detection protocol, on top of which a wide variety of protocols can be built, including routing protocols and more sophisticated heavyweight protocols.

## I. INTRODUCTION

Wireless links are useful in a wide variety of applications. Though wireless networks are often considered to be mobile networks, there are many applications for fixed wireless networks, including wireless mesh networks, sensor networks, and industrial control networks, for which retrofitting a wired network would be impractical or prohibitively expensive. Compared to wired networks, wireless networks do not require as much extra infrastructure, such as in-wall wiring, wiring closets, and conduits in which to run wires. Thus, fixed wireless networks are often deployed due to quick setup and low cost.

Clock synchronization can be crucial to scheduling in a wireless network. For example, some sensor networks use a

sleep schedule to conserve power. If synchronization is lost or tampered with in such systems, nodes may waste power waking up and transmitting at times when other nodes are asleep. Thus, secure clock synchronization is a prerequisite for sensor networks to run efficiently in a hostile environment while keeping a sleep schedule. More generally, clock synchronization is necessary for applications involving wireless control. The stability of a feedback control loop is affected by delay in the loop, thus knowledge of the involved delays is important. Clock synchronization is thus desirable for a wide range of applications [1], [2], some of which are or will be deployed in potentially adversarial environments. Clock synchronization, which provides consistent timestamps, is also valuable for replay prevention. In this paper, we seek to limit an attacker's ability to delay packets on the link that he controls by using secure clock synchronization.

Wireless networks, e.g. sensor networks, are often deployed in hostile environments. In an adversarial setting, attackers may wish to set up a link that does not have the properties of the network in which it is embedded; this attack is called the *wormhole attack* [3], [4], [5]. Additionally, the attacker may wish to tamper with the characteristics of a link between two legitimate nodes; this attack is called the *man-in-the-middle attack*. An attacker performing either of these attacks does so in order to obtain some level of control over a link, and can consequently affect the bandwidth, latency, and loss characteristics of that link. We consider wormhole attacks to be a subset of man-in-the-middle attacks because creating new links is a subset of the general behavior of tampering with links.

Higher-layer protocols may experience substantial performance degradation when trusting the existence of links that are controlled by an attacker. When forwarding routing protocol messages, for example, a link may appear very good, but when forwarding data packets, the link may degrade substantially.

In this paper, we explore the limits of *using timing information alone, such as clock synchronization, to detect man-in-the-middle attackers* in fixed wireless networks without requiring location information. We present solutions to this problem where external location information is unavailable, for example, in a network where the power or cost of obtaining such information is prohibitive. In addition, we consider each link in isolation,

This material is based upon work partially supported by USARO under Contract Nos. W-911-NF-0710287 and W911NF-08-1-0238, and NSF under Contract Nos. ECCS-0701604, CNS-07-21992, CNS-0626584, and CNS-05-19535.

without regard to any information that might be gained from sharing information across links; we leave protocols that perform such sharing to future work.

In exploring the limits of this approach, we will show both positive and negative results. We will first show that under the assumption that nodes use affine clocks, any undetected man-in-the-middle attacker can impose only a constant delay for relayed traffic. We will then prove that in some cases, against some attackers, no timing-based solution can detect an optimal attacker. In every case for which we do not give an impossibility result, we design a protocol that allows us to detect the specified attacker.

The balance of this paper is organized as follows. We discuss related work, describe the problem we are considering formally, present our assumptions, and give our attacker model in Section II. In Section III-A, we present a clock synchronization protocol that enables legitimate nodes to detect any delaying attacker that imposes a non-constant delay. In Section III-B, we show that in the special case when an attacker controls at most some fraction of packets, attacker-delayed packets can be identified and ignored after a certain number of packets are exchanged by legitimate nodes. In Section IV, we show the fundamental limits on detecting man-in-the-middle attackers with timing info alone and present protocols that detect such attackers whenever possible. In Section V, we discuss practical issues involved in implementing our protocols. We conclude our work in Section VI.

## II. BACKGROUND

### A. Related work

In this paper, we address the problem where an attacker in a wireless network acts as a communications channel between the sender and receiver, enabling the attacker to control aspects of such communications. This is a classic example of the *wormhole attack*. We are also concerned with the problem where an attacker influences the properties, including the latency, of a wireless link, generally called a *man-in-the-middle attack*.

Certain previous approaches do not try to prevent wormholes as long as they are contained within a small enough region. Hu, Perrig, and Johnson [5] proposed countermeasures using two different kinds of leashes: geographic leashes and temporal leashes. These leashes are based either on GPS information or on timestamps. Our work differs from the leashes approach because while the leashes restrict the sender and receiver to be within a certain range, they *do not* prove that the link is free from attacker interference. Additionally, we do not assume that clocks begin being synchronized, as the temporal leashes approach does; we also do not require any access to external location determination services, as the geographic leashes approach does.

SECTOR [6] performs distance-bounding based on the rapid exchange of one-bit challenge messages. This approach allows both parties to bound the distance between themselves but does not assure that an attacker does not maliciously lengthen the distance or replay the packets; it merely limits the distance on which the attacker can perform such replays.

Some prior work uses non-timing information to detect wormholes. Hu and Evans [7] solve the wormhole attack using directional antennas to detect transmitter locations that are inconsistent with the physical directions in which the users should reside. Their scheme requires specialized equipment and does not use timing information. Many approaches take a graph-theoretic approach to detecting wormholes, thus relying on the existence of other links and also often on location information. These approaches [8], [9], [10], [11], do not use timing, and hence, are orthogonal to our approach.

The work most similar to our work is TrueLink [12], which shows that wormhole attacks are difficult to perform when using a modified version of the 802.11 Medium Access Control (MAC) protocol because of the real-time requirement for transmission of frames exchanged as part of a single packet. Our work extends TrueLink, showing that under their attacker model (which is our half-duplex attacker), attackers can always be detected, and showing that under a full-duplex attacker (which can allow for substantial wormholes under their scheme), attackers can sometimes be detected. In particular, our impossibility results are theoretically proven, and we develop protocols that detect the attacker in some instances where TrueLink cannot.

Previous work has also been done on clock synchronization within a trusted environment. Graham and Kumar [2] present fundamental limits on timestamp-based clock synchronization and develop a clock synchronization algorithm that handles some of the timing requirements necessary for control applications. Specifically, they show that location information alone is not sufficient to determine node-to-node communication delays and clock offsets for timestamp-based clock synchronization algorithms. Freris and Kumar [1] extend this result to networks of nodes that use affine clocks and show that it is possible to exactly determine both round-trip delay and clock skew relative to a reference clock or reference node. Additionally, the authors show in their Lemma 3 that after clock synchronization, a transmitting node can correctly predict the exact time when a receiving node will receive the transmitted packet on the receiving node's clock. We will make use of the result of this lemma extensively in designing our protocols.

### B. Problem Statement

This paper studies fundamental issues in two important problems in wireless networks. The first problem is that of *clock synchronization*. In order to use the resulting synchronization for a real-time control application, we need to have an assurance that the channel will continue to operate in the same manner afterward as during synchronization. Our *secure* clock synchronization protocol, described in Section III-A, assures these properties. In the special case when an attacker controls at most some fraction of packets, we show that a node can also identify and ignore attacker-delayed packets after using our secure clock synchronization protocol.

The second problem builds on top of our solution to the first problem. Once we have synchronized clocks across a link and ensured that the link will continue to behave consistently, even in the presence of an attacker, can we *detect that attacker*

using timing information *alone*? In particular, secure clock synchronization will give nodes their relative skews and the time at which one node needs to send a packet in order for it to reach the other node at some given time on the other node’s clock. When considering this link in isolation, can we use this link in a way such that we can detect the attacker? We do not consider solutions that require having location information, since location information is not always available. Additionally, Freris and Kumar [1] point out that having location information (e.g. via GPS) does not necessarily help determine total delay from sender to receiver, as the delay is composed of contributions from mechanisms besides propagation delay (e.g. packetization and network stack traversal). Our work is also orthogonal to graph-theoretic approaches, which is important because some networks have an insufficient number of links to make graph-theoretic approaches work and because a sufficiently powerful attacker could cause the exclusion of some legitimate nodes in such approaches.

In addition to protocols that solve our second problem of attacker detection, we are interested in the theory of when this problem is solvable. In fact, previous work has suggested some limited solutions to this problem, but our work expands this by showing the set of all cases where this problem is solvable. In particular, we consider three levels of attackers. For each level, we either show that a solution is impossible or we give a solution that detects the attacker.

### C. Assumptions

Since we are using only timing information to detect attackers, most of our assumptions relate to the timing of events within our network. We begin by analyzing the scenario where there is only one link, which can use the channel whenever it wishes.<sup>1</sup>

Because we consider only a single link, we assume that neither side of this link is malicious. If one endpoint were malicious, then there would be no need for the attacker to affect this link, since the attacking endpoint can affect the link directly anyway. As a result, we require an upper-layer protocol to handle the possibility of node compromise.

As in previous work [1], [13], we assume that each node has a clock that is *affine*; that is, given the reading  $a_i$  on some true clock somewhere, a node  $A$  will read a clock value of  $A(a_i) = S_A a_i + o_A$ , with  $S_A > 0$ , where  $S_A$  is called the *skew* and  $o_A$ , the *offset*. We assume that  $S_A$  and  $o_A$  are not initially known. We also assume that these nodes have the capability to accurately timestamp incoming and outgoing packets.

The pair of nodes sharing a link also share a private key. This keying assumption prevents traditional (cryptographic) man-in-the-middle attacks, and can therefore be used for authenticating packets and providing confidentiality. Moreover, the nodes send unpredictable and verifiable messages to each other, for example, by using CBC encryption and message authentication codes. The

<sup>1</sup>In a network with more than one link, links can share the channel through time-division multiple access (in which case the scheduler used must be aware of each user’s transmission time requirements), frequency-division or code-division multiple access (so all links can be used simultaneously), or a combination of these MAC protocols.

private key can be derived using a number of techniques, such as a public-key infrastructure, as long as the chosen technique is resilient to traditional man-in-the-middle attacks. Our attacker is computationally unable to break the cryptography chosen for the particular instance of the protocol. Furthermore, our attacker’s communications resources (transmitting and receiving) are limited, as described in Section II-D.

This shared key can also be used to derive a spread-spectrum key to mitigate physical-layer jamming attacks [14]. Denial-of-service attacks at the physical layer are, therefore, beyond the scope of this paper. We assume that an attacker cannot violate causality. Additionally, we assume that if an attacker delays a packet, that delay must result in the packet arriving at its destination after the time when the packet would have arrived otherwise (i.e. via a direct transmission). Because we consider only a single link, we do not contemplate a MAC protocol, nor do we consider attacks against MAC protocols.

We assume that the two endpoints do not move for the lifetime of the link and that in the absence of an attacker, the round-trip time between the endpoints remains constant. This also implies that the speed of the channel between the endpoints (in terms of latency, not bandwidth) does not change over time. Consequently, the one-way delay time experienced in each direction must also remain constant over time.

We assume that the endpoints have some limited ability to communicate. In particular, we allow an endpoint to transmit or receive for any amount of time it wishes, but after it finishes transmitting and begins to receive, or after it finishes receiving and begins to transmit, the endpoint must wait an amount of time, called the *turnaround time*, before it can effectively begin the new activity. Packets that arrive during the turnaround process are not received.

### D. Attacker Models

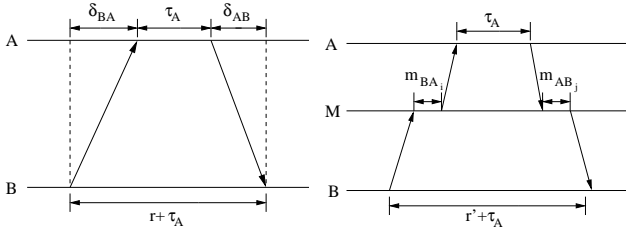
The concept of *half-duplex* and *full-duplex* are important to the understanding of our work and how we avoid certain kinds of attacks. We call a node *half-duplex*, if it can transmit or receive but cannot do both simultaneously. The endpoints described in Section II-C are examples of half-duplex nodes. We call a node *full-duplex*, if it can simultaneously transmit and receive but can concurrently handle the reception of at most one message and transmission of at most one message. We call a node *double full-duplex*, if it can concurrently transmit two messages and receive two messages.

We assume that an attacker is free to do as he pleases, except that he desires to remain undetected. In particular, the attacker is free to use additional hardware such as directional antennas, emissions detection hardware, etc. Also, the attacker’s radio has an infinitesimal turnaround time, that is, it takes a half-duplex attacker negligible time to switch from receiving to transmitting or vice versa.

Our intuition is that if the attacker is half-duplex or full-duplex, we can detect the attacker by *overwhelming the attacker’s radio hardware*. Specifically, a node’s radio hardware is overwhelmed when the transmit or receive capabilities of the node’s radio are exceeded. One example of when radio hardware is overwhelmed

TABLE I  
NOTATION

Symbol	Definition (units)
$\tau_A$	Turn-around time of node $A$ (s)
$\delta_{AB}$	The one-way delay from $A$ to $B$ (s)
$r$	Round-trip time without attacker (s)
$r'$	Measured round-trip time including delay by attacker (s)
$m_{AB_i}$	Attacker's delay of packet $i$ from $A$ to $B$ (s)
$o_A$	Clock offset of node $A$ (s)
$S_A$	Clock skew of node $A$
$S_{BA}$	Relative clock skew of node $B$ to node $A$ , $S_{BA} = S_B/S_A$
$\rho_{AB}$	The distance between nodes $A$ and $B$ (m)
$c$	The speed of light (m/s)



(a) Without attacker  
(b) With attacker  
Fig. 1. Message passing example of notation

is when a half-duplex node is transmitting a message and therefore unable to receive an incoming packet because both happen concurrently.

### III. SECURE CLOCK SYNCHRONIZATION

In the previous section, we provided a description of the man-in-the-middle attacker and introduced our assumptions. In this section, we will use the clock synchronization technique proposed by Freris and Kumar [1], and we show that a man-in-the-middle attacker can stay undetected only if he delays messages from one legitimate node to the other by a constant delay. We will use this property extensively in following sections.

For ease of description, Table I shows our notation, which is also partially illustrated in Figure 1. We will treat all colluding attacker nodes as a single unit, thus attacker  $M$  consists of one or more nodes spread throughout the network. For our clock synchronization protocol and Theorem III.1, the only limitation we place on the attacker model is that the attacker does not want to be detected.

Because we use affine clocks and can infer processing delay, we can place stringent limitations on what behavior the attacker can carry out without being detected. We include in the notion of processing delay all delays between when the clock synchronization application releases a packet and when the packet actually is transmitted by the radio, including, for example, packetization delay and MAC layer delay.

#### A. Clock Synchronization Protocol

We will now present the result of Freris and Kumar [1], which shows that after clock synchronization, legitimate nodes can send messages to each other and know exactly the time on the remote node's clock when the remote node receives the message. The synchronization is performed between 2 nodes,  $A$  and  $B$ , over a single link. The synchronization uses 4 messages, one sent from  $A$  to  $B$ , one sent from  $B$  to  $A$ , a second message from  $A$  to  $B$  and

a second message from  $B$  to  $A$ . Following the notation of Freris and Kumar, we denote the send and receive times as measured on the local clocks for message  $i$  as  $s_i$  and  $r_i$ , respectively. Assuming the endpoints use affine clocks, the time on node  $j$ 's clock at the true time  $a_i$  is  $j(a_i) = S_j a_i + o_j$ . After the message exchange described above, endpoints are able to calculate

$$\begin{pmatrix} r_1 \\ s_2 \\ r_3 \\ s_4 \end{pmatrix} = \begin{pmatrix} s_1 & 1 & 0 & 1 \\ r_2 & 0 & -1 & 1 \\ s_3 & 1 & 0 & 1 \\ r_4 & 0 & -1 & 1 \end{pmatrix} \begin{pmatrix} S_j \\ S_j \delta_{AB} \\ S_j \delta_{BA} \\ o_j \end{pmatrix}$$

The authors then note that the  $4 \times 4$  matrix in the above equation has a rank of 3. Thus, there is no unique solution for the vector of values on the right hand side of the above equation. However, the skew is uniquely determined by this system of equations. After some manipulation, the authors also show that nodes can predict the times at which a remote node will receive a message sent to it by the local node. This prediction is possible because the prediction equations only make use of the particular solution to the above equation:

$$\begin{aligned} r_k &= S_j^* s_k + S_j^* \delta_{AB}^* \quad (k \text{ odd}) \\ r_k &= S_j^{*-1} s_k + \delta_{AB}^* \quad (k \text{ even}) \end{aligned}$$

The starred quantities above denote particular solutions to the matrix system of equations presented above.

Using this clock synchronization protocol allows legitimate nodes to impose restrictions on would-be man-in-the-middle attackers, who want to delay packets passed between the two endpoints. We describe these restrictions in the following theorem.

**Theorem III.1.** *We assume a fixed network (constant delay between two neighbors) where each node has an affine clock, and any two neighbors can authenticate data transmitted by the other. Our synchronization protocol can eventually detect any attacker that imposes a delay that is not constant.*

*Intuition:* The received timestamp of packets sent from  $A$  to  $B$  are, in the absence of an attacker, affine in the sending timestamp of those packets, so any attacker-induced delay must also be affine in the sending time.  $B$  can calculate a skew from these timestamps, but if the calculated skew is smaller than the actual skew, then the attacker-induced delay is decreasing, and when it reaches zero, the attacker will be caught since he cannot acausally send a message. Furthermore, the product of the skews in both directions must equal one, so if the skew is increased in one direction, then it must be reduced in the other direction, so an attacker that wants to remain undetected may only impose constant delays.

*Proof:* Let  $a_1, a_2, \dots, a_k$  represent the true-clock transmission times of node  $A$ , and let  $b_1, b_2, \dots, b_k$  represent the true-clock transmission times of node  $B$ . Let  $\delta_{AB}$  be the normal (constant) delay from  $A$  to  $B$ , and  $m_{AB_i}$  be the delay that the attacker introduces on the  $i$ th packet from  $A$  to  $B$ . We define the affine transformation functions  $A(a_i) = S_A a_i + o_A$  and  $B(b_i) = S_B b_i + o_B$ , where the domains are all true clock

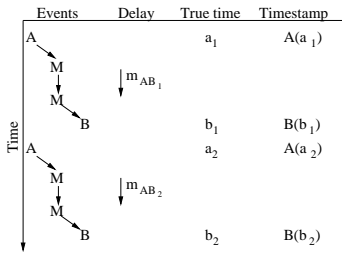


Fig. 2. Illustration of Theorem III.1

times and the ranges are the times measured by  $A$  and by  $B$ , respectively.

Node  $B$  can compute the difference between the sending and receiving timestamps of packet  $i$  using  $d_i = B(a_i + \delta_{AB} + m_{AB_i}) - A(a_i)$ . Using the expansion  $B(a_i + \delta_{AB} + m_{AB_i}) = B(a_i) + S_B(\delta_{AB} + m_{AB_i})$  and adding and subtracting  $S_{BA}A(a_i)$ , we get

$$d_i = S_{BA}A(a_i) - A(a_i) + B(a_i) - S_{BA}A(a_i) + S_B(\delta_{AB} + m_{AB_i})$$

We simplify using the definitions of  $B(a_i)$  and  $A(a_i)$

$$\begin{aligned} B(a_i) - S_{BA}A(a_i) &= (S_B a_i + o_B) - (S_B a_i + S_{BA} o_A) \\ &= o_B - S_{BA} o_A \quad \text{to get} \end{aligned}$$

$$d_i = (S_{BA} - 1)A(a_i) + S_B m_{AB_i} + o_B - S_{BA} o_A + S_B \delta_{AB}$$

Thus, when  $m_{AB_i} = 0$ , then  $d_i$  is affine in  $a_i$ , so the attacker must choose a  $m_{AB_i}$  that remains affine in  $a_i$  in order to remain undetected.

From any two  $d_i, d_j$  ( $i \neq j$ ) and their corresponding sending timestamps,  $A(a_i), A(a_j)$ ,  $B$  can compute its *perceived skew* relative to  $A$ ,

$$S'_{BA} = \frac{d_j - d_i}{A(a_j) - A(a_i)}$$

We know  $m_{AB_i}$  is affine, so it is either a constant, increasing, or decreasing function of  $A(a_i)$ . When it is increasing,  $S'_{BA} > S_{BA}$ , and when it is decreasing,  $S'_{BA} < S_{BA}$ .

If  $m_{AB_i}$  is decreasing, after finite time  $m_{AB_i} < 0$ , at which point the attacker would be caught due to a causality violation, so the attacker must choose  $S'_{BA} \geq S_{BA}$  to remain undetected. A parallel argument for packets sent from  $B$  to  $A$  shows that the attacker must also choose  $S'_{AB} \geq S_{AB}$  to remain undetected. Thus, to remain undetected the attacker chooses  $S'_{AB} S'_{BA} \geq S_{AB} S_{BA} = 1$ , but when  $S'_{AB} S'_{BA} \neq 1$ , we can detect an attack, thus to remain undetected, the attacker must choose  $S'_{BA} = S_{BA}$ , which implies that  $m_{AB_i}$  is constant. ■

After clocks are synchronized as described in this section, when  $A$  sends a packet, it does so notifying  $B$  of when to expect to receive that packet according to  $B$ 's clock. If the attacker ever delays a packet by an amount inconsistent with its prior delays, that misbehavior is instantly detected.

### B. Tolerating a Selectively Relaying Attacker

So far we have considered a link where the attacker completely controls the channel, that is, it can arbitrarily delay all packets. We will consider the following lossy model for the direct channel from the sender to the receiver: Out of every every  $n$  contiguous

packets sent by the sender, at least 1 is directly received by the receiver. We call this an *at least 1 out of  $n$*  channel. A network designer is free to set  $n$  arbitrarily to match the desired quality of service goals for the network. In the special case where the attacker does not control all packets sent on a link, that is, where some packets are directly receivable by the legitimate endpoint nodes, then we can improve upon our previous result of merely detecting the attacker. In this special case, we can continue to use a channel even in the presence of an attacker, while ignoring the attacker-delayed packets. This tolerance is also applicable to the useful situation when the attacker completely controls a link but then leaves and no longer affects a link.

We consider the situation where some packets are directly received by the endpoint nodes, and other packets that would otherwise be lost are relayed by the attacker to the endpoints after some delay. We even allow the attacker to somehow have the knowledge of which packets are directly received by the receiver and which are not, and thus the attacker can selectively relay only the packets which were not directly received. Using the result of Theorem III.1, the endpoints would be able to detect the attacker, if it did not delay packets by a constant amount. However, we will show below that the legitimate endpoint nodes can continue to use the at least 1 out of  $n$  channel.

If the attacker can cause fewer than 1 in  $n$  (e.g. 1 in  $n + 1$ ) but more than 0 packets to be directly received, then the attacker is caught. Hence, the link may be removed by Theorem III.1 because the attacker causes a non-constant delay and the quality of service criteria has not been met. If the attacker somehow causes 0 packets to be directly received, then this special situation collapses back to the case where the attacker has complete control. Since nodes have affine clocks and attacker-delayed packets do not arrive before the time they would have arrived without delay, plotting the receiver timestamps of packets against the transmitter timestamps included in packets results in the data points corresponding to the directly-received packets forming a line, and all other points lie above it. Such a line is a *supporting hyperplane* in this two-dimensional space. In other words, all packets that are directly received lie on a supporting hyperplane, and all relayed packets lie above that supporting hyperplane because of causality. Therefore, the endpoints can use this link, knowingly using only the directly received packets, after receiving a certain number of packets directly. The following lemma states our result formally.

**Lemma III.2.** *Consider a wireless link where an attacker cannot delay all packets, and at least 1 in every  $n$  contiguous packets are received through direct transmission from the endpoints. On this at least 1 out of  $n$  link, after the transmitting endpoint sends at most  $2n$  packets, the receiving endpoint node can knowingly use only directly received packets in spite of any attacker-induced delay.*

*Proof:* We will refer to Figure 3 to help illustrate various points in this proof. In this figure, the  $x$ 's represent relayed (delayed) packets and circles represent direct (non-delayed)

packets. The vertical line divides the first set of  $n$  packets (Set 1) from the second set of  $n$  packets (Set 2).

After  $2n$  packets, the receiving node calculates potential supporting hyperplanes formed by one point in each of Set 1 (i.e. packets 1 through  $n$ ) and Set 2 (i.e. packets  $n + 1$  through  $2n$ ). There is at least one directly received packet in each of these sets. Furthermore, the only hyperplane containing a packet in each of the two sets of packets and supporting the data points from below is the one formed by only directly received packets. We can prove this by breaking the receiver's choice of the two points (which form a line) down into four cases, one packet from each set of packets (listed as  $\langle$ Packet from set 1 $\rangle - \langle$ Packet from set 2 $\rangle$ , where each packet is listed as either R for Relayed or D for Direct): 1) R-R, 2) D-R, 3) R-D, and 4) D-D.

*Case 1:* In this case, the receiver calculates a line formed by two relayed packets, one from each of the two sets of  $n$  packets. We show that the line formed by these two packets cannot support from below all of the directly received packets. The R-R line is either parallel to or intersects the D-D line. If the lines are parallel, then the R-R line necessarily is supported from below by the D-D line because relayed packets are delayed by the attacker, moving those points in the direction of increasing receive time and off the D-D line. If the lines intersect, the R-R line may support from below direct packets from one set but not both sets because the R-R crosses the D-D line at only one spot, either to the left or to the right of the chosen relayed packets. The two lines cannot intersect between the two chosen relayed packets because we assume packets are not relayed such that they arrive before they would if they were received directly. If the intersection is to the right of the relayed packets, then the direct packets from Set 1 are necessarily below the R-R line because relayed packets can only be delayed. The same argument applies to the situation where the two lines intersect to the left of the two relayed packets, but in this case the direct packets in Set 2 are necessarily below the R-R line. Thus, the R-R line cannot support from below directly received packets from both sets.

*Case 2:* Consider a line formed by a directly received packet Set 1 and a relayed packet from Set 2. Because the intersection of the D-R line and the D-D line is at the directly received packet from Set 1, and because the relayed packet from Set 2 lies above the D-D line, lines formed by D-R packets cannot support from below the direct packets from Set 2.

*Case 3:* This case is the same as Case 2 above, except that the direct packets from Set 1 cannot be supported from below by the R-D line.

*Case 4:* The case where the supporting hyperplane is formed by two directly received packets is thus the only remaining choice for a supporting hyperplane that supports all packets from below.

After the exchange of these  $2n$  packets, the receiver only accepts packets received on the calculated supporting lower hyperplane, eliminating packets delayed by the attacker. ■

In summary, through Theorem III.1, we prevent a man-in-the-middle attacker from selectively delaying messages by other than a constant amount, and we enable nodes to choose a time at which

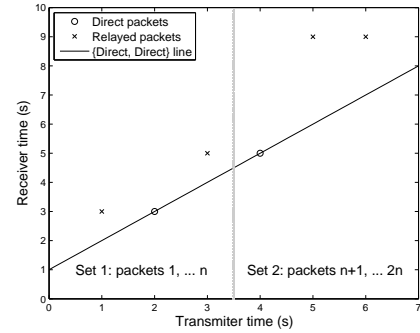


Fig. 3. Illustration of Lemma III.2 with  $n = 3$

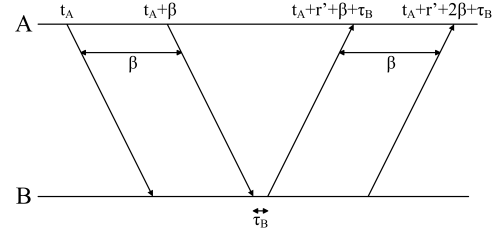


Fig. 4. Detecting half-duplex attacker

another node will receive a message. Additionally, inconsistent behavior by a man-in-the-middle attacker is instantly detected. Thus, by Lemma III.2, legitimate endpoint nodes can use a link without interference from an attacker.

#### IV. CHANNEL VERIFICATION

After synchronization, legitimate nodes can verify the channel to detect man-in-the-middle attackers with certain radio capabilities. In this section, we prove when a node can and when it cannot detect an attacker, if the only information available is timing information over a single link.

##### A. Detecting a Half-Duplex Attacker

In this section, we propose a protocol illustrated in Figure 4 that detects a half-duplex man-in-the-middle attacker. The intuition behind the protocol lies in the fact that the half-duplex attacker cannot simultaneously send and receive. Thus, if a message is sufficiently long, the attacker cannot delay all messages by a constant, which violates the synchronization result of Theorem III.1.

Let node  $A$  first perform clock synchronization with the intended recipient, node  $B$ . Let  $r$  be the round-trip time if there is no man-in-the-middle attacker between  $A$  and  $B$ , and let  $r'$  be the measured round-trip time between  $A$  and  $B$  with an attacker. If  $A$  and  $B$  can synchronize their clocks successfully as discussed in Section III-A, then by Theorem III.1, the attacker is undetected only if the attacker-induced delay in round-trip time,  $r' - r$ , is a non-negative constant. After clock synchronization,  $A$  starts timing at time  $t_A$ , and sends to  $B$  a signal of duration  $\beta > \frac{1}{2}(r' + \tau_B)$  including a timestamp, where  $\tau_B$  is the turnaround time of  $B$ . These steps are illustrated without an attacker in Figure 4. Upon receiving the signal from  $A$ ,  $B$  switches from receiving mode to transmitting mode. After the turnaround time  $\tau_B$ ,  $B$  also sends a signal of the same duration, along with a timestamp, to  $A$ .

Upon receiving the entire signal from  $B$ ,  $A$  calculates the elapsed time,  $T_{elapsed}$ . The total elapsed time calculated by  $A$  is expected to be the sum of the previously measured round-trip time,  $B$ 's turnaround time, and twice the signal duration. But  $r' + \tau_B < 2\beta$ , so  $T_{elapsed} = r' + \tau_B + 2\beta < 4\beta$ . In the following lemma, we prove that a half-duplex attacker will be detected using the described protocol.

**Lemma IV.1.** *The above protocol can detect a half-duplex man-in-the-middle attacker.*

*Proof:* A half-duplex attacker can only receive or transmit but not both simultaneously, hence the attacker needs at least four times the signal duration ( $4\beta$ ):  $\beta$  to buffer the signal from  $A$ , some additional amount for a delay,  $\beta$  to replay the signal to  $B$ ,  $\beta$  to buffer the signal from  $B$ , some additional amount to delay, and  $\beta$  to replay to  $A$  because the attacker cannot predict ahead of time when the messages will be sent. However, this elapsed time  $T'_{elapsed} \geq 4\beta$  exceeds  $T_{elapsed}$  expected by  $A$ . Thus, the attacker is detected. ■

### B. Detecting a Full-Duplex Attacker

Assume that the attacker has better technology than normal nodes, allowing it to handle full-duplex traffic. The attacker can simultaneously transmit and receive data unlike legitimate nodes, which are half-duplex. In this section, we show when it is impossible and when it is possible to detect a full-duplex attacker using timing alone.

1) *Impossible:* We now show that it is impossible for a pair of nodes to detect the presence of a man-in-the-middle attacker,  $M$ , if the distance between either endpoint and the attacker is less than half the product of that endpoint's turnaround time and the speed of light ( $\exists j \in \{A, B\}$  such that  $\rho_{Mj} < c\frac{\tau_j}{2}$ ).

**Lemma IV.2.** *With timing information alone, a full-duplex man-in-the-middle attacker cannot be detected by a pair of half-duplex nodes if the distance between one of the nodes and the attacker is strictly less than half the product of that node's turnaround time and the speed of light.*

*Proof:* Let the turnaround time of node  $A$  be  $\tau_A$ . If the attacker,  $M$ , is less than a distance of  $c\frac{\tau_A}{2}$  away from  $A$  (i.e.  $\rho_{MA} < c\frac{\tau_A}{2}$ ), then the attacker can always forward messages expected by a node.

If  $A$  is transmitting, then  $A$  will not be able to receive messages until  $\tau_A$  later since  $A$  is half-duplex. However, messages sent by  $A$  reach the attacker less than half the turnaround time after being sent (i.e.  $\frac{\rho_{MA}}{c} < \frac{\tau_A}{2}$ ). Thus, if  $A$  is transmitting at time  $t$  and this is detected by the attacker at  $t + \rho_{MA}/c$ , the attacker need not forward any messages to  $A$ . Moreover, if the attacker forwards messages to  $A$ , those messages would reach  $A$  at time  $t + 2\rho_{MA}/c < t + \tau_A$ . Therefore, only messages from  $A$  to  $B$  are important to the system when  $A$  is transmitting. On the other hand, if the attacker senses that  $A$  is not transmitting, then the attacker should forward the only messages in the system, that is, messages from  $B$  to  $A$ .

In summary, the attacker can always forward messages expected by a node by forwarding from  $A$  to  $B$  when  $A$  is

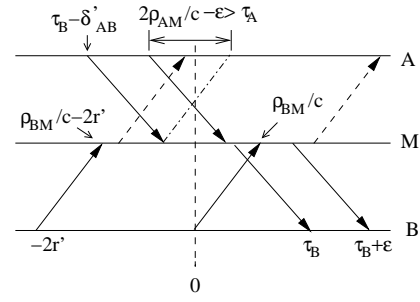


Fig. 5. Detecting certain full-duplex attackers

transmitting, and from  $B$  to  $A$  when  $A$  is not transmitting. The attacker is undetectable to  $B$  because  $B$  receives all of  $A$ 's messages. Similarly, the attacker is undetectable to  $A$  because  $A$  cannot tell that it is not receiving any messages. ■

**Corollary IV.3.** *With timing information alone, a pair of nodes  $A$  and  $B$  cannot detect any attacker if the measured round-trip time between them is shorter than both turnaround times, i.e.,  $r' < \tau_A$  and  $r' < \tau_B$ .*

*Proof:* Since packets are forwarded by the attacker, the attacker must be less than a distance of  $c\frac{r'}{2}$  away from one of the two nodes. However,  $\tau_A > r'$  and  $\tau_B > r'$ , thus the attacker is sufficiently close to at least one of two nodes to avoid detection by Lemma IV.2. ■

2) *Possible:* Now that we have shown an impossibility result, we develop a protocol that detects a full-duplex man-in-the-middle attacker in every other case. Given the distance between the attacker and each node is greater than half the product of each respective node's turnaround time and the speed of light ( $\rho_{Mj} > c\frac{\tau_j}{2}, \forall j \in \{A, B\}$ ), we propose a protocol that can detect the presence of a full-duplex man-in-the-middle attacker. The intuition behind our protocol is to force the attacker to both receive two messages simultaneously and to transmit two messages simultaneously.

First, the two nodes start the protocol by performing clock synchronization, as described in Section III-A. Nodes  $A$  and  $B$  then coordinate their transmissions so that they will overlap at any intermediate attacker, as long as the distance between the attacker and each endpoint is greater than half the product of that endpoint's turnaround time and the speed of light; that is,  $\forall j \in \{A, B\}, \rho_{jM} > c\frac{\tau_j}{2}$  (i.e.  $\tau_j < 2\rho_{jM}/c$ ). Thus,  $\forall j \in \{A, B\}, 2\rho_{jM}/c - \tau_j > 0$ , so we can define an  $\epsilon = \min_{j \in \{A, B\}} (2\rho_{jM}/c - \tau_j) > 0$ .

$B$  begins transmitting two round-trip times before some pre-defined 0 time on its clock and stops transmitting at time 0.  $A$  then sends a packet that arrives at the first time that  $B$  can receive, that is, at  $\tau_B$ ; to do so,  $A$  transmits a message of length  $\epsilon$  at time  $\tau_B - \delta'_{AB}$  (Section III-A shows that  $A$  can do this without knowing  $\delta'_{AB}$ ). We now show that these two messages must overlap at the attacker, and that the attacker needs to both simultaneously receive two messages and simultaneously transmit two messages in order to successfully defeat the challenge.

**Theorem IV.4.** *With timing information alone, a full-duplex man-in-the-middle attacker will be detected, if the distance*

between the attacker and each node is greater than half the product of each respective node's turnaround time and the speed of light.

*Proof:* First, we will show that the two messages will overlap for the entire duration of the short message, and that the overlapping portion is heard in its entirety at both endpoints. Second, we will show that the attacker,  $M$ , must transmit portions of both packets simultaneously. Since the attacker can neither simultaneously receive two messages, nor can it simultaneously transmit two messages, the attacker is necessarily detected.

Figure 5 illustrates the following discussion of how and when messages need to be exchanged to detect the attacker. To show that the two messages collide at the attacker is straightforward. First, we observe that  $B$  sends a message at  $-2r'$  so it arrives at the attacker at  $\rho_{BM}/c - 2r'$ .  $\rho_{BM}/c - 2r' < -r'$ , using  $r' > \rho_{BM}/c$ , and  $\rho_{BM}/c - 2r' < -\delta'_{AB}$  using  $r' > \delta'_{AB}$ . We get  $\rho_{BM}/c - 2r' < \tau_B + \rho_{AM}/c - \delta'_{AB}$  after we make use of the facts that  $\tau_B \geq 0$  and  $\rho_{AM}/c > 0$ . The attacker first receives the message from  $A$  at  $\tau_B + \rho_{AM}/c - \delta'_{AB}$ . The packet from  $A$  ends  $\varepsilon$  later at  $\tau_B + \rho_{AM}/c - \delta'_{AB} + \varepsilon$ . By substituting the definition of  $\varepsilon$ , we show that  $\tau_B + \rho_{AM}/c - \delta'_{AB} + \varepsilon \leq \tau_B + \rho_{AM}/c - \delta'_{AB} + 2\rho_{BM}/c - \tau_B$ . By rearranging we get,  $\tau_B + \rho_{AM}/c - \delta'_{AB} + \varepsilon \leq \rho_{AM}/c + \rho_{BM}/c - \delta'_{AB} + \rho_{BM}/c$ . Finally, using the fact that  $\rho_{AM}/c + \rho_{BM}/c < \delta'_{AB}$ , we arrive at  $\tau_B + \rho_{AM}/c - \delta'_{AB} + \varepsilon < \rho_{BM}/c$ . Since the attacker finishes receiving the message from  $B$  at  $\rho_{BM}/c$ , the message from  $A$  collides with the message from  $B$  at the attacker.

To show that both messages can be received by  $A$  and  $B$ , and thus the attacker needs to transmit both messages, we first observe that  $B$  obviously receives a message at time  $\tau_B$  since it stops transmitting at time 0.  $A$  transmits until  $\tau_B - \delta'_{AB} + \varepsilon$ , and is thus able to hear messages at  $\tau_B - \delta'_{AB} + \varepsilon + \tau_A \leq \tau_B - \delta'_{AB} + 2\rho_{AM}/c - \tau_A + \tau_A$  (using the definition of  $\varepsilon$ ), where  $\tau_B - \delta'_{AB} + 2\rho_{AM}/c - \tau_A + \tau_A = \tau_B - \delta'_{AB} + 2\rho_{AM}/c$ . In order for  $A$  to hear a message by  $\tau_B - \delta'_{AB} + 2\rho_{AM}/c$ , the attacker must transmit it by time  $\tau_B - \delta'_{AB} + \rho_{AM}/c$ , which is exactly the beginning of the message overlap at the attacker. In other words, for the attacker to successfully transmit both of the overlapping message segments while  $A$  cannot receive and before time  $\tau_B - \delta'_{AB} + \rho_{AM}/c$ , the attacker must impose a *negative delay* on the overlapping portion of the message from  $B$ . However, this packet, like every other packet, is checked for consistency with the time synchronization described in Section III-A, so the attacker would need to acausally send all packets from  $B$  to  $A$ . Since this is clearly impossible, the attacker will always be detected.

We have now shown that the attacker needs to receive two packets at the same time, but because it cannot do so, the attacker will be caught whenever multiple packets arrive simultaneously at the attacker.

We now also show that the attacker must transmit two packets simultaneously for a positive duration of time.  $A$  transmits its message to  $B$  from  $\tau_B - \delta'_{AB}$  to  $\tau_B - \delta'_{AB} + \varepsilon$  according to  $B$ 's clock.  $A$  does this by choosing to transmit such that the message begins arriving at  $B$  at  $\tau_B$ , which is possible because of our result shown in Section III-A, where a transmitter can transmit a

message such that the receiver will receive the message at a time chosen by the transmitter according to the receiver's clock. This message begins arriving at the attacker at  $\tau_B - \delta'_{AB} + \rho_{AM}/c$ . From the definition of  $\varepsilon$  we conclude that the attacker must begin relaying message from  $B$  to  $A$  as soon as he receives the packets from  $A$  in order to avoid any gap (and hence detection) between  $A$  finishes turning around and  $A$  begins receiving packets. That is, from the attacker's perspective, the attacker must transmit to  $A$  from  $\tau_B - \delta'_{AB} + \rho_{AM}/c$  until  $\delta'_{AB} - \rho_{AM}/c$ .

On the other hand, the attacker must forward  $A$ 's message to  $B$  such that  $B$  can start receiving at  $\tau_B$ . That is, from the attacker's perspective, the packet must be forwarded from the attacker to  $B$  from  $\tau_B - \rho_{BM}/c$  until  $\tau_B - \rho_{BM}/c + \varepsilon$ .

Since  $\delta'_{BA}$  is composed of the propagation delays from  $B$  to  $M$  and from  $M$  to  $A$ , the delay imposed by the attacker, and possibly other delays,  $\delta'_{BA} \geq \rho_{AM}/c + \rho_{BM}/c + m_{BA}$  by composition. We then apply this inequality and get

$$\tau_B - \delta'_{AB} + \rho_{AM}/c \leq \tau_B - \rho_{BM}/c - m_{BA} \leq \tau_B - \rho_{BM}/c$$

That is, the attacker must begin forwarding from  $B$  to  $A$  earlier than from  $A$  to  $B$ . On the other hand, we also apply the definition of  $\varepsilon$  to get

$$\tau_B + \varepsilon - \rho_{BM}/c \leq \rho_{BM}/c \leq \rho_{BM}/c + m_{AB} \leq \delta'_{AB} - \rho_{AM}/c$$

That is, the attacker must finish forwarding from  $A$  to  $B$  before from  $B$  to  $A$ . In conclusion, during the length of the packet from  $A$  to  $B$  (i.e.  $\varepsilon$ ), the attacker must be simultaneously transmitting two packets to avoid detection. Since a full-duplex attacker cannot transmit two messages simultaneously, the attacker is detected. ■

### C. Double Full-Duplex Attacker

**Lemma IV.5.** *An attacker equipped with a double full-duplex radio cannot be detected using timing information alone.*

*Proof:* A constant forwarding delay cannot be detected using timing alone because this delay is indistinguishable from additional normal propagation delay. A double full-duplex attacker can simultaneously receive from and transmit to both  $A$  and  $B$ . The double full-duplex attacker can thus receive a signal from  $A$ , delay it and replay it to  $B$ , and *simultaneously* receive a signal from  $B$ , delay it and replay it to  $A$ . A double full-duplex attacker can inject a constant delay, and thus, cannot be detected with timing information alone. ■

## V. DISCUSSION

With secure clock synchronization, our protocols can catch all half-duplex attackers. In addition, we can catch full-duplex attackers if the attacker is at least a distance of  $c \cdot \frac{1}{2}\tau$  away from legitimate nodes.

These results indicate that an attacker that wishes to carry out a man-in-the-middle attack and not be detected will require superior hardware to what the legitimate nodes in the network possess. For example, if legitimate nodes are only half-duplex, the attacker requires full-duplex hardware. Similarly, if legitimate nodes have hardware with fast enough turnaround time, or have

TABLE II  
TURN-AROUND TIME

Network	802.11a	UMTS TDD	WiMax TDD
$\tau$	$2\mu s$	$25\mu s$	$5\mu s$

full-duplex hardware, then the attacker requires hardware that achieves double full-duplex performance.

To assess the practicality of our protocol for detecting full-duplex attackers, we consider  $\tau$  values for real networks. Table II shows  $\tau$  values specified in IEEE 802.11a [15], UMTS TDD mode [16] and WiMax TDD [17]. In IEEE 802.11a,  $\tau$  is  $2\mu s$ , implying that a full-duplex attacker is detectable if the distance between the normal nodes and the attacker is greater than  $c \cdot \frac{1}{2}\tau = 300$  m, as described in Section IV-B. Based on this result, a shorter turnaround time is needed to secure against a full-duplex attacker. However, when  $\tau$  is sufficiently small, or the distance is sufficiently large, such as in Low Earth Orbit (LEO) satellites [18], and the minimum distance bound on the attacker's proximity to a node holds, the attacker is forced to have highly sophisticated hardware to avoid detection, even if legitimate nodes have half-duplex radios.

In Frequency Division Duplexing (FDD) networks, simultaneous transmission and reception are possible, that is, legitimate nodes have full-duplex radios. FDD-WCDMA networks use this technology [19]. In such networks,  $\tau = 0$ , and the attacker is forced to have twice the radio capacity of normal users.

When legitimate nodes and attackers have the same radio capacity, the attacker can always be detected. We have shown this result for the half-duplex and full-duplex cases. For cases where the radio capacity is greater than full-duplex, the scenario can be reduced to multiple attackers, one each on each full-duplex or half-duplex channel. Thus, the attacker can always be caught when legitimate nodes have the same radio capabilities as the attacker.

## VI. CONCLUSION

We have presented a protocol that securely synchronizes clocks over any link by passing timestamped messages. This protocol ensures that any man-in-the-middle attacker can only introduce constant delay. Our protocol also enables legitimate nodes to determine their clock skews and predict when another node will receive a packet sent to it. For the special case where some packets are directly receivable on a link, we showed how nodes can identify and ignore attacker-delayed packets after exchanging a certain number of packets. We then studied the impact of three different types of attackers: half-duplex attackers that can either transmit or receive, full-duplex attackers that can both transmit and receive but only one channel at a time, and double full-duplex attackers that can transmit two and receive two messages simultaneously. We showed that half-duplex attackers can be detected easily, that double full-duplex attackers are undetectable based on timing alone, and that full-duplex attackers may or may not be detectable depending on the relation between the turnaround time and the attacker's location. For each of these attackers, we developed tight impossibility results describing

when an attacker cannot be detected, and we gave algorithms for each scenario that is not impossible that will detect attackers. One interesting result is that when an attacker uses the same technology as legitimate nodes, the attacker is always detectable.

## REFERENCES

- [1] N. M. Freris and P. R. Kumar, "Fundamental Limits on Synchronization of Affine Clocks in Networks," in *Proceedings of the 46th IEEE Conference on Decision and Control*, December 2007, pp. 921–926.
- [2] S. Graham and P. Kumar, "Time in general-purpose control systems: the control time protocol and an experimental evaluation," *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, vol. 4, pp. 4004–4009 Vol.4, Dec. 2004.
- [3] P. Papadimitratos and Z. Haas, "Secure routing for mobile ad hoc networks," in *Proc. the SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNSD '02)*, San Antonio, TX, Jan. 2002.
- [4] K. Sanzgiri, B. Dahill, B. Levine, C. Shields, and E. Belding-Royer, "A secure routing protocol for ad hoc networks," in *Proc. the 10th IEEE International Conference on Network Protocols (ICNP'02)*, Paris, France, Nov. 2002, pp. 78–87.
- [5] Y.-C. Hu, A. Perrig, and D. Johnson, "Packet leases: a defense against wormhole attacks in wireless networks," in *Proc. the 22th IEEE Conference on Computer Communications (INFOCOM'03)*, San Francisco, CA, Mar. 2003, pp. 1976–1986.
- [6] S. Čapkun, L. Buttyan, and J. Hubaux, "SECTOR: Secure tracking of node encounters in multi-hop wireless networks," in *Proc. the ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN'03)*, Fairfax, VA, Oct. 2003, pp. 21–32.
- [7] L. Hu and D. Evans, "Using directional antennas to prevent wormhole attacks," in *Proc. the 11th Network and Distributed System Security Symposium (NDSS'04)*, San Diego, CA, Feb. 2004, pp. 131–141.
- [8] I. Khalil, S. Bagchi, and N. B. Shroff, "LITEWOP: A lightweight countermeasure for the wormhole attack in multihop wireless networks," in *Proc. the International Conference on Dependable Systems and Networks (DSN'05)*, Yokohama, Japan, Jun. 2005, pp. 612–621.
- [9] I. Khalil, S. Bagchi, and N. Shroff, "MOBIWOP: Mitigation of the wormhole attack in mobile multihop wireless networks," in *Proc. the 2nd International Conference on Security and Privacy in Communication Networks (SecureComm'06)*, Baltimore, MD, Aug. 2006, pp. 1–12.
- [10] R. Poovendran and L. Lazos, "A graph theoretic framework for preventing the wormhole attack in wireless ad hoc networks," *Wireless Networks*, vol. 13, no. 1, pp. 27–59, Feb. 2007.
- [11] R. Maheshwari, J. Gao, and S. R. Das, "Detecting wormhole attacks in wireless networks using connectivity information," in *Proc. the 26th IEEE Conference on Computer Communications (INFOCOM'07)*, Anchorage, AL, May 2007, pp. 107–115.
- [12] J. Eriksson, S. Krishnamurthy, and M. Faloutsos, "TrueLink: A practical countermeasure to the wormhole attack in wireless networks," in *Proc. the 14th IEEE International Conference on Network Protocols (ICNP'06)*, Santa Barbara, CA, Nov. 2006, pp. 75–84.
- [13] B. Scheuermann, W. Kiess, M. Roos, F. Jarre, and M. Mauve, "Error Bounds and Consistency of Maximum Likelihood Time Synchronization," Heinrich Heine University, Düsseldorf, Germany, Tech. Rep. TR-2008-001, February 2008.
- [14] R. Pickholtz, D. Schilling, and L. Milstein, "Theory of spread-spectrum communications—a tutorial," *IEEE Trans. Commun.*, vol. 30, no. 5, pp. 855–884, May 1982.
- [15] I. S. 802.11, *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*. IEEE Std 802.11, 2007.
- [16] H. Holma and A. Toskala, *WCDMA for UMTS: Radio Access for Third Generation Mobile Communications, 3rd Edition*. John Wiley & Sons, 2004.
- [17] B. Dropping, "WiMAX ready for prime time," *Wireless Design & Development*, 2007.
- [18] M. Sturza, "LEOs—the communications satellites of the 21st century," in *Proc. Northcon'96*, Seattle, WA, Nov. 1996, pp. 114–118.
- [19] S. K. Wong, S. W. Lee, and M. L. Sim, "RF transceiver reference design for third generation W-CDMA cellular handset," *Consumer Electronics, IEEE Transactions on*, vol. 51, no. 2, pp. 371–378, May 2005.