# Enhanced Wireless Channel Authentication Using Time-Synched Link Signature

Yao Liu, Peng Ning
North Carolina State University, Raleigh, NC 27695
{yliu20, pning}@ncsu.edu

*Abstract*— **Wireless link signature is a physical layer authentication mechanism, which uses the unique wireless channel characteristics between a transmitter and a receiver to provide authentication of wireless channels. A vulnerability of existing link signature schemes has been identified by introducing a new attack, called *mimicry attack*. To defend against the mimicry attack, we propose a novel construction for wireless link signature, called *time-synched link signature*, by integrating cryptographic protection and time factor into traditional wireless link signatures. We also evaluate the mimicry attacks and the time-synched link signature scheme on the USRP2 platform running GNURadio. The experimental results demonstrate the effectiveness of time-synched link signature.**

## I. Introduction

Wireless physical layer security is becoming increasingly important as wireless devices are more and more pervasive and adopted in critical applications. For example, implantable medical devices (IMD) such as pacemaker may grant access to an external control device only when it is close enough [15], thus making it critical to verify the physical proximity of the control device. There have been multiple proposals recently to provide enhanced wireless security using physical layer characteristics, including fingerprinting wireless devices (e.g., [3]), authenticating and identifying wireless channels (e.g., [14], [19]), and deriving secret keys from wireless channel features only observable to the communicating parties (e.g., [12]).

Among the recent advances in wireless physical layer security is (wireless) link signature. Link signature uses the unique wireless channel characteristics (e.g., the multi-path effect) between a transmitter and a receiver to provide authentication of the wireless channel. Three link signature schemes [8], [14], [19] have been proposed so far. Since its introduction, link signature has been recognized as a wireless channel authentication mechanism for applications where wireless channel characteristics are unique (e.g., [3], [12]).

A vulnerability of existing link signature schemes has been identified by introducing a new attack called *mimicry attack* [9], [10]. Traditional link signature schemes [8], [14], [19] assumed that "an attacker cannot 'spoof' an arbitrary link signature" and that the attacker "will not have the same link signature at the receiver unless it is at exactly the same location as the legitimate transmitter" [14]. However, it was shown in [10] and [9] that a mimicry attacker *can* forge an *arbitrary* link signature as long as it roughly knows or can estimate the legitimate signal at the receiver's location, and the attacker does not have to be at exactly the same location as the legitimate transmitter in order to forge its link signature.

To defend against the threat identified in this paper, we develop a new link signature scheme, which is called time-synched (i.e., time synchronized) link signature. Time-synched link signature integrates cryptographic protection as well as time factor into the wireless physical layer features, and provides an effective countermeasure against mimicry attacks. We also perform an extensive set of experimental evaluation of the mimicry attacks and the time-synched link signature scheme on the USRP2 platform [11] running GNURadio [1]. Our experiments confirm that the mimicry attacks against the previous link signature schemes are a real threat and demonstrate that the newly proposed time-synched link signatures are effective in mitigating those attacks.

## II. Preliminaries

### A. Multi-path Effect and Link Signature

Wireless signal usually propagates in the air along multiple paths due to reflection, diffraction, and scattering [14]. As a result, a receiver may receive multiple copies of the signal on different paths, each of which may have a different delay due to the path it traversed on. The received signal is the sum of these time delayed signal copies. Each path imposes a *response* (e.g., distortion and attenuation) on the signal traveling along it [14], and the superposition of all responses between two nodes is referred to as a *channel impulse response* [6].

The multi-path effects between different pairs of nodes are usually different, and so are the channel impulse responses [14]. Due to this reason, a channel impulse response between two nodes is also called a *link signature*, and has been proposed to provide robust location distinction and location-based authentication [14], [19]. Specifically, to determine if a received signal is from the desired location/channel of the transmitter, the receiver estimates the link signature of the received signal and compares it with *reference link signatures*, which are estimated when the receiver has known signals from the desired location/channel. The received signal is accepted only if the estimated link signature is similar to the references.

### B. Estimating Channel Impulse Responses

Channel impulse responses are usually estimated using training sequences [17]. Specifically, the transmitter converts the training sequence (i.e., a sequence of bits) into $M$ physical layer symbols (i.e., complex numbers that are transmission units at the physical layer [6]). The transmitter then sends the $M$ symbols to the wireless channel, while the receiver feeds the corresponding received symbols and the same training sequence into a channel estimator to estimate the channel impulse response. Two types of estimators are generally used: least-square (LS) estimator and linear minimum mean squared error (LMMSE) estimator [2]. The training sequence can be pre-shared [17] or reconstructed from the received signal [14].

### C. Mimicry Attack

Let $\mathbf{y}_t$ and $\mathbf{y}_a$ denote the received symbols from the transmitter and the attacker, respectively. The attacker's goal in the mimicry attack is to make $\mathbf{y}_a$ approximately the same as $\mathbf{y}_t$. Thus, when the receiver attempts to extract the link signature from the attacker's symbols $\mathbf{y}_a$, it will get a link signature similar to the one estimated from $\mathbf{y}_t$.

The attacker needs to meet two requirements to launch a mimicry attack: First, the attacker needs to roughly know the received symbols $\mathbf{y}_t$. Second, the attacker needs to manipulate its own symbols, such that when the manipulated symbols arrive at the receiver, they are similar to $\mathbf{y}_t$ (i.e., $\mathbf{y}_a \approx \mathbf{y}_t$). The detailed steps for the attacker to achieve both goals are shown in [10] and [9].

## III. Time-synched Link Signature

In this section, we develop a novel time-synched link signature to defend against the mimicry attack. A key feature of this new mechanism is the integration of cryptographic protection and time factor into wireless link signatures.

### A. Assumptions and Threat Analysis

**Assumptions:** We assume that there are a *Transmitter* and a *Verifier*, who share a secret key $K$ that is only known to them. The Transmitter sends physical layer *frames* to the Verifier, who then verifies if these frames are directly transmitted by the Transmitter. We assume that the attacker can eavesdrop, overhear, and jam wireless communications. However, we assume that the attacker cannot compromise the Transmitter or the Verifier, and thus does not know their secret.

**Threat Analysis:** Let us first understand what new challenges the mimicry attack brings given the existing network security tools. First of all, note that we can simply add digital signatures or Message Integrity Code (MIC) into each frame. As a result, the frames forged by the attacker can be easily detected through authentication of message content. Thus, the remaining threat is from the frames that are originally generated by the Transmitter but forwarded by the attacker. Note that the frame forwarded by the attacker is the same as the original frame generated by the Transmitter at the bit level, but different at the symbol level.

Moreover, with replay attack detection mechanism such as sequence numbers, if the Verifier can receive the original frames sent by the Transmitter, it can easily identify frames forwarded by the attacker as duplicates and discard them. Thus, the unresolved threats are from the following two cases: (1) when the attacker can jam and replay the Transmitter's frames (jam-and-replay attack [5]), and (2) when the Transmitter and the Verifier are out of communication range, but the jammer forwards frames from the Transmitter to the Verifier.

In this paper, we focus on the unresolved threats, assuming existing mechanisms such as cryptographic authentication and sequence numbers can be used. In the following, we clarify the attacker's capabilities in forwarding frames.

We assume the attacker may launch *frame repeater attacks*. That is, the attacker may receive a frame sent by the Transmitter and then forward it to the Verifier. Such frame repeaters are widely available commercially (e.g., 802.11 repeaters).

The attacker may also launch physical layer *symbol repeater attacks*. That is, the attacker can observe the transmission of each physical layer symbol, which may represent one or multiple bits in the frame, and then forward the symbol to the Verifier directly. Such repeaters can be developed using noise canceling techniques and proper positioning of antennas [4]. Compared with frame repeater attacks, symbol repeater attacks are much harder to defend against.

Link signatures are specific to wireless communication channels, and usually require a training phase. The attacker may target at either the *training phase* to mislead the Transmitter and the Verifier about their link signature, or the *operational phase* when the link signature is used for physical layer authentication. Thus, a secure link signature has to protect both the training and the operational phases.

### B. Design Strategy

The fundamental reason for the mimicry attack is that the attacker can establish a set of equations based on (1) the knowledge of the training sequence and (2) the Transmitter's signal (i.e., physical layer symbols) at the Verifier's location. These allow the attacker to manipulate the transmitted physical layer symbols so that a forged frame has a valid link signature.

**Initial Idea:** To defend against this attack, our strategy is to deprive the attacker at least one of these two pieces of information. It is in general very difficult to prevent a passive attacker from receiving signals (and then extracting valid link signatures). However, it is possible to prevent the attacker from knowing the training sequences. Thus, our initial idea is to use *unpredictable*, *dynamic*, and *authenticated* training sequences for extracting link signatures from wireless packets (frames).

**Detecting Frames Forwarded by Attackers:** It is not hard to realize that simply using unpredictable, dynamic, and authenticated training sequences is still insufficient. The attacker can receive and analyze the Transmitter's signal to learn the training sequence, and forge link signatures by manipulating and forwarding frames received from the Transmitter.

To handle this threat, we propose to bring "time" into the scheme. We assume the Transmitter and the Verifier have synchronized clocks. (Our scheme will include a time synchronize component to meet this assumption.) The Transmitter may include a timestamp in the transmitted frame, which indicates the time when a particular bit or byte called the *anchor* (e.g., the Start of Frame Delimiter (SFD) field [7]) is transmitted over the air. We assume that the Transmitter can use authenticated timestamping techniques (e.g., [18]) to ensure that the timestamp precisely represents the point in time when the anchor is transmitted. Upon receiving a frame, the Verifier can use this timestamp and the frame receiving time to estimate the frame traverse time. An overly long time indicates that the frame has been forwarded by an intermediate attacker.

**Defending against Physical Layer Symbol Repeater Attacks:** A physical layer symbol repeater attack is much harder to detect than frame repeater attacks. If the attacker knows where the training sequence is located in the frame, she can start repeating the physical layer symbols right after receiving the symbols for the training sequence. This reduces the delay that the attacker has to tolerate to the transmission time of the training sequence, which could be much shorter than the transmission time of the entire frame.

To defend against such physical layer symbol repeater attacks, we propose to integrate a third idea into the scheme, that is, to make the location of the training sequence *unpredictable until the end of the frame transmission*. Specifically, we insert the training sequence at a *random* location in the payload, and place this location, which can be represented as the offset from the start of the frame header, at the end of the frame. In order for a physical layer symbol repeater to mimic the link signature of the Transmitter, she has to manipulate the physical layer symbols corresponding to the training sequence in a frame. If the location of the training sequence is not revealed until the end of the frame, the attacker will have to wait until the end of the transmission to learn it. This forces a physical layer symbol repeater attack to degenerate into a frame repeater attack.

**Minimum Frame Length:** If a frame is too short, the Verifier may have difficulty seeing the delay caused by a frame repeater. One solution is to pad extra bits into the frame if the frame length is less than a minimum frame length.

The minimum frame length can be determined based on the errors of the time synchronization and time measurement. Assume the maximum errors in clock discrepancy and transmission time are $e_\delta$ and $e_\tau$, respectively, and the maximum time measurement errors in the Transmitter and the Verifier are $e_T$ and $e_V$, respectively. Thus, the maximum error that the Verifier has to tolerate is $e_{all} = e_\delta + e_\tau + e_T + e_V$. Assume that the data rate of the wireless communication is $R$. It is easy to see that when the frame length is greater than the minimum frame length $L_{min} = R \cdot e_{all}$, the Verifier is guaranteed to detect frames forwarded by frame repeaters.

It has been demonstrated in an implementation of Radio Frequency (RF) distance bounding protocol [16] that nanosecond processing delay is feasible to achieve. The time-synched link signature requires much less precision in time synchronization. For example, even assuming $e_{all}$ is between $1\mu s$ and $10\mu s$, in a 54 Mbps 802.11g wireless network, $L_{min}$ will range between 7 bytes and 68 bytes.

**Overall Design:** Figure 1 illustrates how these ideas can be integrated into a physical layer protocol. A physical layer frame typically consists of a series of preamble symbols, the frame header, and the payload. To detect frames forwarded by attackers, we include in each frame a timestamp $t_s$, which indicates the transmission time of the frame. To defend against physical layer repeater attacks, we include the randomly generated offset $P$ of the training sequence in each frame at the end of the frame (to force the attacker to wait until the end of frame transmission).

Assume the Transmitter and the Verifier share a secret key $K$. We piggyback the authentication of the frame with the generation of the unpredictable, dynamic, and authenticated training sequence. Specifically, we propose to use the MIC of the entire frame as the training sequence $\mathbf{x}$. In situations where there is a mismatch between the MIC and the training sequence (e.g., when a longer training sequence is needed), we can simply generate the training sequence as $\mathbf{x} = F(K, t_s)$, where $F$ is a pseudo-random generator, and compute the frame MIC separately. The use of $K$ and $t_s$ makes $\mathbf{x}$ dynamic and unpredictable, and the frame MIC allows $\mathbf{x}$ to be authenticated.

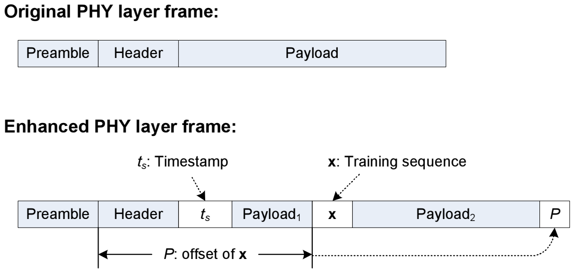

Fig. 1. PHY layer frame: Dynamic training sequence with random offset

In the following, we present the details of the training and the operational phase in time-synched link signature. The security analysis of the proposed time-synched link signature can be found in [9].

### *C. Training Phase*

The training phase is intended for the Verifier to collect enough information from the Transmitter so that the Verifier can verify the link signatures of the future frames from the Transmitter. The Verifier should obtain the valid link signature from the Transmitter whenever the link signature may change. This can be accomplished by executing the training phase protocol periodically or whenever one of them moves.

In the training phase, the Verifier needs to synchronize its clock with the Transmitter, and obtain the link signature for the current communication channel. Moreover, it needs to confirm that there is no successful attack during the training phase.

We use the classic time synchronization technique (e.g., [13]) to estimate the clock discrepancy between the Transmitter and the Verifier as well as the frame traverse time. We refer to the point in time when the anchor (e.g., the SFD field) in a frame is transmitted or received as the *transmission time* or the *receiving time* of this frame. Specifically, the Verifier sends a *request frame* to the Transmitter, and at the same time records the frame transmission time $t_1$ in the Verifier's local clock. When the Transmitter receives the request frame, it records the receiving time $t_2$ of this frame, and then sends a *reply frame* to the Verifier, in which $t_2$ and the transmission time $t_3$ of the reply frame (in the Transmitter's clock) are included. Finally, the Verifier receives the reply frame and records the receiving time $t_4$ in its clock. The clock discrepancy $\delta$ between the Verifier and the Transmitter and the one-way frame traverse time $\tau$ can then be estimated as $\delta = \frac{(t_2-t_1)-(t_4-t3)}{2}$ and $\tau = \frac{(t_2-t_1)+(t_4-t_3)}{2}$ [13].

Figure 2 shows the training phase protocol between the Transmitter and the Verifier.

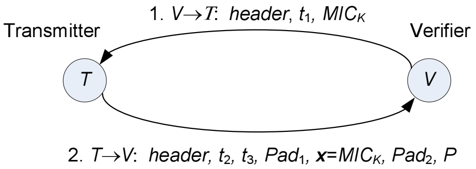
1. V→T: header, $t_1$, $MIC_K$
Transmitter
Verifier
T
V
2. T→V: header, $t_2$, $t_3$, $Pad_1$, x=$MIC_K$, $Pad_2$, P

Fig. 2. Training phase protocol

**Training Request:** The Verifier sends the first training request frame to the Transmitter, which includes the frame header, the transmission time $t_1$ of this frame, and the frame MIC that covers the entire frame (excluding the preambles). Upon receiving of the request frame, the Transmitter immediately records the receiving time $t_2$ of the frame, and authenticates the request frame by verifying the MIC.

**Training Reply:** Upon verifying a training request frame, the Transmitter should send back a training reply frame. The Transmitter should include time $t_2$ and the actual transmission time $t_3$ of the reply frame in the frame. The Transmitter also pads the frame payload to at least the minimum frame length $L_{min}$ and randomly selects an offset $P$ to place the training sequence as discussed earlier. The Transmitter then leaves a placeholder (e.g., all 0's) in place of the training sequence and computes the frame MIC using the shared key $K$. Finally, the Transmitter places the frame MIC as the training sequence $\mathbf{x}$ in the reply frame and sends it over the air.

Once the Verifier receives the training reply frame, the Verifier computes the clock discrepancy $\delta$ and the one-way transmission time $\tau$. If $\tau$ is greater than a threshold $\tau_{max}$, which is the maximum possible direct transmission time, the Verifier should consider the reply frame as possibly forwarded by the attacker and discard it. Otherwise, the Verifier locates the frame MIC by following the offset $P$ at the end of the frame, authenticates the frame MIC using the shared key $K$, and uses the frame MIC (i.e., the training sequence $\mathbf{x}$) to extract the link signature. The Verifier may run the training phase several times to get a better quality link signature.

### D. Operational Phase

Once the Verifier obtains the clock discrepancy and the valid link signature from the Transmitter, they can start the operational phase, during which the Verifier uses this link signature to verify frames that require physical layer authentication.

**Transmitter:** To defend against the threats discussed in Section III-A, the Transmitter follows the design shown in Figure 1. Specifically, the Transmitter randomly selects an offset in the frame payload to include the field for the training sequence. places the offset $P$ at the end of the frame, and computes the frame MIC using the shared secrete key $K$, with a placeholder (e.g., all 0's) for the training sequence. The Transmitter then uses the frame MIC as the training sequence $\mathbf{x}$, puts it in the frame, and sends the frame over the air. Similar to the training phase, the Transmitter estimates the frame transmission $t_s$ based on the current time and the estimated duration for the deterministic MIC computation.

**Verifier:** When the Verifier receives the frame, it immediately records the receiving time $t_r$. The Verifier then retrieves the frame transmission time $t_s$ from the received frame and estimates the frame traverse time $\tau = t_s - t_r - \delta$, where $\delta$ is the clock discrepancy between the Verifier and the Transmitter learned in the training phase. If $\tau$ is greater than the threshold $\tau_{max}$, the maximum possible direct transmission time, the Verifier should consider the frame possibly forwarded by the attacker and discard it. Otherwise, the Verifier locates the frame MIC by using the offset $P$ at the end of the frame, verifies the frame MIC using the shared key $K$, and then uses the frame MIC as the training sequence to extract the link signature. Finally, the Verifier compares this link signature with the one derived during the training phase. The frame is accepted if this link signature does not deviate from the valid one learned in the training phase. Otherwise, the frame is considered forged and discarded.

## IV. Experimental Evaluation

We have implemented the link signature scheme in [14], the basic mimicry attack, and the newly proposed time-synched link signature. We have also implemented the frame repeater attack, which can be used along with the mimicry attack. Our prototype uses USRP2 [11], which are equipped with AD and DA converters as the RF front ends, and XCVR2400 daughter boards operating in the 2.4 GHZ range as transceivers. The software toolkit is GNURadio [1].

### A. Evaluation Methodology

**Evaluation Scenarios:** Our prototype system consists of a transmitter, a receiver, and an attacker. The receiver is 15 meters from the transmitter. Each node is a USRP2 connected to a commodity PC. The receiver estimates the received link signatures and compares them with the transmitter's link signature. We consider three scenarios in our evaluation: (1) *normal scenario*, (2) *forgery scenario*, and (3) *defense scenario*. In a normal scenario, the attacker simply sends original symbols to the receiver. In both the forgery and the defense scenarios, the attacker launches the mimicry attack, during which it transmits manipulated symbols to the receiver. However, the forgery scenario uses the previous link signature scheme in [14], while the defense scenario uses the newly proposed time-synched link signature scheme.

**Evaluation Metrics:** Intuitively, the attacker wants to reduce the difference between its own link signature and the transmitter's link signature, whereas the defense method aims to increase this difference to alert the receiver. Thus, the link difference between both the attacker's and the transmitter's link signatures can visually reveal the impact of mimicry attacks and the effectiveness of the defense method. The method of calculating link differences is given in [14].

### B. Evaluation Results

We now show how mimicry attacks affect the link difference, false alarm rate, detection rate, and the tradeoff between the detection and the false alarm rates in the normal, forgery, and defense scenarios.

In each evaluation scenario, the receiver first measures a set $\mathcal{H}$ of $N = 50$ link signatures of the transmitter in the training phase. It then collects 450 link signatures of the attacker and calculates the link difference $d_{a,\mathcal{H}}$ for each. Moreover, the receiver collects another 450 link signatures of the transmitter, and calculates the link difference $d_{t,\mathcal{H}}$ for each of them.

Figures 3, 4, and 5 show the link differences for the attacker $d_{a,\mathcal{H}}$ and the transmitter $d_{t,\mathcal{H}}$ in the normal, forgery, and defense scenarios, respectively. Figure 3 shows that in the normal scenario $d_{a,\mathcal{H}}$ is generally larger than $d_{t,\mathcal{H}}$. Moreover, Figure 6 shows the histograms of $d_{a,\mathcal{H}}$ and $d_{t,\mathcal{H}}$. Most of the transmitter's link difference is less than 0.15, whereas most of the attacker's link difference is larger than 0.15. Thus, based on the link difference, the receiver can achieve a high accuracy in distinguishing between the transmitter and the attacker.

In the forgery scenario, the attacker launches mimicry attacks to make its own link signatures similar to the transmitter's link signatures. Figure 4 shows that $d_{a,\mathcal{H}}$ decreases to the same level as $d_{t,\mathcal{H}}$, and $d_{a,\mathcal{H}}$ and $d_{t,\mathcal{H}}$ substantially overlap with each other. The histogram of $d_{a,\mathcal{H}}$ (i.e., the top graph in Figure 7) shows that the link difference distribution of the attacker is very close to that of the transmitter. The mimicry attack reduces the link difference between the attacker and the transmitter, leading to high false negative rate at the receiver.

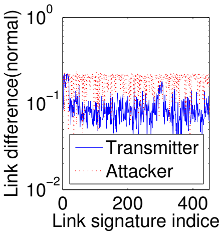

Fig. 3. Normal

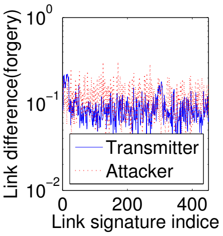

Fig. 4. Forgery

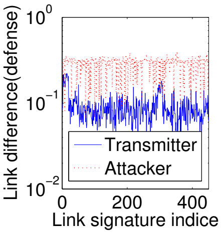

Fig. 5. Defense

In the defense scenario, as indicated in Figure 5, the use of time-synched link signature increases the link difference $d_{a,\mathcal{H}}$ for the attacker. In particular, the mean value of $d_{a,\mathcal{H}}$ under the defense and forgery scenarios are 0.2847 and 0.1170, respectively. The histogram of $d_{a,\mathcal{H}}$ in the defense scenario (i.e., the bottom graph in Figure 7) shows that the link difference computed from a majority of forged signatures is larger than 0.15. Thus, the receiver can again distinguish between the transmitter and the attacker with low error rate.

## V. Conclusion

A mimicry attacker can forge a transmitter's link signature if she knows *approximately* the legitimate symbols at the receiver. To defend against the mimicry attack, we proposed the time-synched link signature scheme by integrating cryptographic protection and time factor into wireless features. Our experimental results demonstrated both the feasibility of mimicry attacks and the effectiveness of the proposed method.

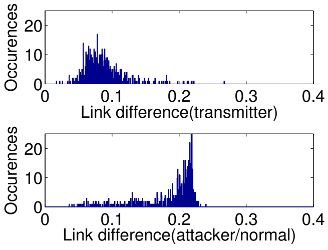

Fig. 6. Histograms of link difference for the transmitter's and the attacker's link signatures in normal scenario

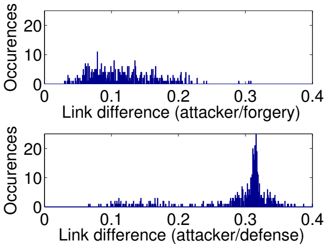

Fig. 7. Histograms of link difference for the attacker's link signatures in forgery and defense scenarios

## Acknowledgements

This work is supported by the National Science Foundation (NSF) under grant CNS-1016260. The authors would also like to thank the anonymous reviewers for their helpful comments.

## References

[1] GNU Radio - The GNU Software Radio. `http://www.gnu.org/software/gnuradio/`.

[2] M. Biguesh and A. B. Gershman. Training-based mimo channel estimation: A study of estimator tradeoffs and optimal training signals. *IEEE Transaction on Signal Processing*, 54(3):884–893, March 2006.

[3] V. Brik, S. Banerjee, M. Gruteser, and S. Oh. Wireless device identification with radiometric signatures. In *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking (MobiCom '08)*, pages 116–127, 2008.

[4] J. Choi, M. Jain, K. Srinivasan, P. Levis, and S. Katti. Achieving single channel, full duplex wireless communication. In *Proceedings of the 16th ACM Mobicom (Mobicom '10)*, September 2010.

[5] S. Ganeriwal, S. Capkun, C. Han, and M. B. Srivastava. Secure time synchronization service for sensor networks. In *Proceedings of 2005 ACM Workshop on Wireless Security (WiSe 2005)*, pages 97–106, September 2005.

[6] A. Goldsmith. *Wireless Communications*. Cambridge University Press, New York, NY, USA, 2005.

[7] IEEE Std 802.15.4-2003. IEEE standard for information technology – telecommunications and information exchange between systems – local and metropolitan area networks – specific requirements – part 15.4: Wireless medium access control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area networks (LR-WPANs).

[8] Z. Li, W. Xu, R. Miller, and W. Trappe. Securing wireless systems via lower layer enforcements. In *In Proceedings of ACM Workshop on Wireless Security (WiSe'06)*, 2006.

[9] Y. Liu and P. Ning. Mimicry attacks against wireless link signature and defense using time-synched link signature. Technical Report TR-2011-17, NC State University, Computer Science Department, July 2011.

[10] Y. Liu and P. Ning. Poster: Mimicry attacks against wireless link signature. In *Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS'11)*, 2011.

[11] E. R. LLC. The USRP product family products and daughter boards. `http://www.ettus.com/products`. Accessed in April 2011.

[12] S. Mathur, W. Trappe, N. Mandayam, C. Ye, and A. Reznik. Radiotelepathy: extracting a secret key from an unauthenticated wireless channel. In *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking (MobiCom '08)*, 2008.

[13] D. Mills. Internet time synchronization: The network time protocol. *IEEE Transactions on Communications*, 39(10):1482–1493, 1991.

[14] N. Patwari and S. K. Kasera. Robust location distinction using temporal link signatures. In *MobiCom '07: Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pages 111–122, New York, NY, USA, 2007. ACM.

[15] K. Rasmussen, C. Castelluccia, T. Heydt-Benjamin, and S. Čapkun. Proximity-based access control for implantable medical devices. In *Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS '09)*, 2009.

[16] K. Rasmussen and S. Čapkun. Realization of rf distance bounding. In *Proceedings of the USENIX Security Symposium*, 2010.

[17] R. Safaya. A multipath channel estimation algorithm using a kalman filter. `http://www.ittc.ku.edu/research/thesis/documents/rupul_safaya_thesis.pdf`.

[18] K. Sun, P. Ning, C. Wang, A. Liu, and Y. Zhou. TinySeRSync: Secure and resilient time synchronization in wireless sensor networks. In *Proceedings of 13th ACM Conference on Computer and Communications Security (CCS '06)*, pages 264–277, October/November 2006.

[19] J. Zhang, M. H. Firooz, N. Patwari, and S. K. Kasera. Advancing wireless link signatures for location distinction. In *MobiCom '08: Proceedings of the 14th ACM international conference on Mobile computing and networking*, New York, NY, USA, 2008. ACM.