Combating Insider Attacks in IEEE 802.11 Wireless Networks with Broadcast Encryption

Joseph Soryal*, Irippuge Milinda Perera†, Ihab Darwish*,
Nelly Fazio*†, Rosario Gennaro*†, and Tarek Saadawi*†

*The City College of CUNY
{jsoryal00@,idarwis00@,fazio@cs.,rosario@cs.,saadawi@}ccny.cuny.edu
†The Graduate Center of CUNY
iperera@gc.cuny.edu

***Abstract*—The IEEE 802.11 protocols are used by millions of smartphone and tablet devices to access the Internet via Wi-Fi wireless networks or communicate with one another directly in a peer-to-peer mode. Insider attacks are those originating from a trusted node that had initially passed all the authentication steps to access the network and then got compromised. A trusted node that has turned rogue can easily perform Denial-of-Service (DoS) attacks on the Media Access Control (MAC) layer by illegally capturing the channel and preventing other legitimate nodes from communicating with one another. Insider attackers can alter the implementation of the IEEE 802.11 Distributed Coordination Function (DCF) protocol residing in the Network Interface Card (NIC) to illegally increase the probability of successful packet transmissions into the channel at the expenses of nodes that follow the protocol standards. The attacker fools the NIC to upgrade its firmware and forces in a version containing the malicious code.**

In this paper, we present a distributed solution to *detect* and *isolate* the attacker in order to minimize the impact of the DoS attacks on the network. Our detection algorithm enhances the DCF firmware to enable honest nodes to monitor each other's traffic and compare their observations against honest communication patterns derived from a two-dimensional Markov chain. A channel hopping scheme is then used on the physical layer (PHY) to evade the attacker. To facilitate communication among the honest member stations and minimize network downtime, we introduce two isolation algorithms, one based on identity-based encryption and another based on broadcast encryption. Our simulation results show that the latter enjoys quicker recovery time and faster network convergence.

***Index Terms*—Broadcast encryption, Byzantine attack, DoS attack, identity-based encryption, IEEE 802.11, Markov chain.**

I. Introduction

The IEEE 802.11 Distributed Coordination Function DCF [1] protocol specifies two mechanisms to perform packet transmission. The default mechanism is a two-way handshaking method referred to as "basic access". This mechanism employs immediate transmission of an acknowledgement (ACK) packet by the destination node after a successful reception of a packet transmitted by the sender.

The second mechanism (on which we focus in this paper) features a four-way handshaking procedure called "request-to-send (RTS)/clear-to-send (CTS)", which proceeds as follows. Prior to transmitting a packet, a node "reserves" the channel by sending a special RTS short frame as shown in Fig. 1. The available destination node responds to an RTS frame with a CTS frame, which is followed by data packet transmission by the sender node and a concluding acknowledgement (ACK) packet by the destination node.

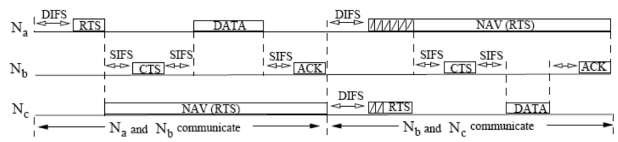


Fig. 1. N_a and N_c are contending to talk to N_b.

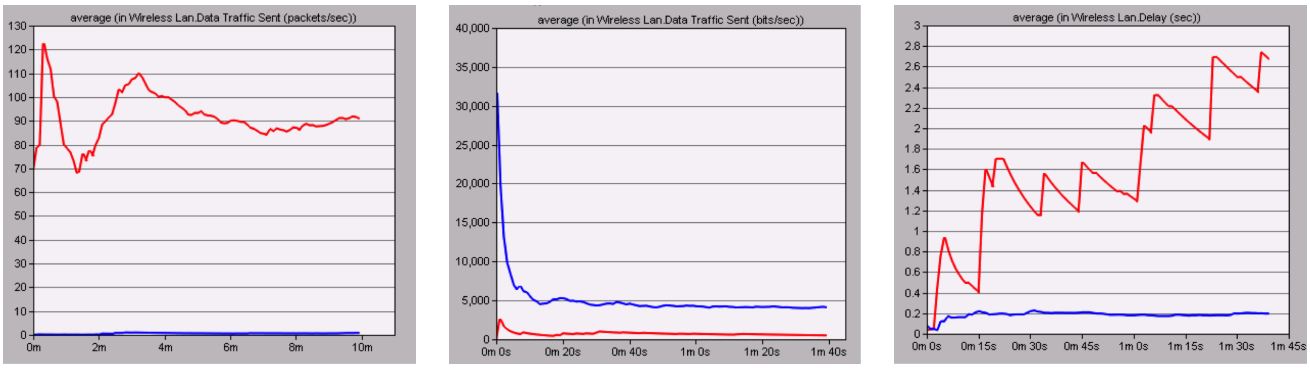
Under cooperative behaviour, the RTS/CTS mechanism increases the network throughput by reducing the duration of a collision when long messages are transmitted. To prevent disruption by outsiders, fully distributed ad-hoc wireless networks employ group access control mechanisms (like WEP or WAP), but once authenticated, member stations are trusted to follow the DCF protocol. A Byzantine station attack occurs when a trusted node gets compromised and starts acting maliciously. As a sample scenario, consider a group of users at a conference who use their smartphones and tablets to exchange files directly, and suppose that one of them turns rogue after passing the authentication steps. The compromised node follows the RTS/CTS mechanism but continuously pretends to have valid information to send so as to disrupt communication in the network—a denial-of-service attack.

In the collision avoidance mechanism, the Binary Exponential Backoff (BEB) algorithm [1] is used to regulate the back-off times for each node before attempting to transmit packets. Nodes that have packets to transmit computes a back-off value bw based on the Contention Window cw as follows:

$$\mathsf{bw} = \mathsf{int}(\sigma \times r \times \mathsf{cw}),$$

where σ is the slot time, r is a randomly generated value uniformly distributed between 0 and 1, and $\mathsf{cw}_{\mathsf{min}} < \mathsf{cw} < \mathsf{cw}_{\mathsf{max}}$ are the minimum and maximum values for the contention window cw.

Each node calculates back-off times in the range $[0, \mathsf{cw}_{\mathsf{min}} - 1]$. Then, when the medium becomes idle, after an additional



(a) (b) (c)

Fig. 2. (a) Traffic Sent (packets/second) by a node when following the standards (blue) vs. when mounting the attack (red). (b) Traffic Sent (bits/second) by an honest node under normal conditions (blue) vs. when an attacker is present (red). (c) Packet transmission delay (seconds) under normal conditions (blue) vs. when an attacker is present (red).

guard period, each node decrements its back-off timer until the medium becomes busy again or until its timer value reaches zero. If the timer has not reached zero and the medium becomes busy, the node freezes its own timer. The process continues until the timer is decremented to zero, at which point the first packet in the transmit queue is sent out. In case of a successful transmission, the receiving node will acknowledge the packet by sending an ACK packet to the sending node. The sending node will then set its cw to its initialization value of $\mathsf{cw}_{\mathsf{min}} - 1$. On the other hand, an attacker disregards the BEB algorithm and backs off only by one slot every time it encounters a collision.

Our Contribution. In this paper, we build on the work of [2] (discussed in Sect. III) to attain faster recovery time and shorter network convergence after the detection of the attacker. Our goal is to *mitigate* the DoS attack by giving innocent nodes at least a small window of communication, rather than letting the attacker completely occupy the channel. The results obtained by simulations as presented in Sect. V show that our new isolation algorithm reduces the attacker's impact drastically. The improvements are very significant when compared to the isolation algorithm introduced in [2]. To increase the effectiveness in network convergence and recovery time, we incorporate *broadcast encryption* techniques in our attacker isolation mechanism.

II. Attack Impact

The sole purpose of the DoS attacker is to disrupt the communication among the legitimate nodes by capturing the channel. The attacker randomly picks a node inside the network and starts communicating with it following the four-way RTS/CTS handshake mechanism so that it appears as a legitimate node to other nodes in the group. The packets sent by the attacker do not contain any useful information and once it reaches the transport layer in the receiving node, these packets are discarded since there is no communication session associated with them. The attacker substantially increases the probability of the transmitted packets by only backing off one slot time disregarding the IEEE 802.11 standards every time it has a packet to transmit. To show the impact of the DoS attack, we coded the attacker's behavior in OPNET [3] simulator and studied its effect on the honest nodes. The network configurations and parameters of our simulations are detailed below.

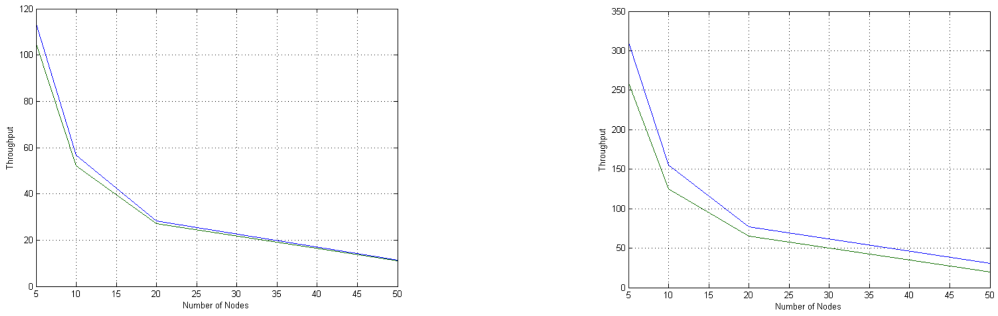
Figure 2a shows the difference between the traffic sent in packet/second when the a node is following the IEEE 802.11 standards (blue line) and when a node is mounting the Byzantine attack (red line). We observe that the number of packets sent when the attack mode is active is about eighty times higher than the number of packets sent under normal conditions.

Figure 2b shows the traffic sent by an honest node in bits/second. The blue line shows the traffic sent under normal circumstances (i.e., all the other nodes in the system are also honest nodes). When an attacker is present (red line), the transmission rate of the honest node falls from about 5000 bits/second to about 500 bits/second. Thus, the attacker causes severe throughput reduction and bandwidth utilization.

Figure 2c shows the delay in seconds for an honest node. It is very clear that when the network is under attack, the delay increases exponentially. This may also cause buffer overflows and dropped data packets, the typical elements of a DoS attack.

III. Detection Algorithm

The first step to combat an attack on the network is to detect the attacker. The detection algorithm [2] depends on modifying the IEEE 802.11 DCF firmware to enable the nodes to monitor the traffic by each node and compare it to the threshold values derived by solving a two-dimensional Markov chain [2], [4]. The maximum theoretical throughputs are determined by solving the Markov chain where all nodes are under saturation condition. The theoretical values are the maximum number of packets that a single node can transmit over time into the channel in the presence of the same number of nodes during the communication session. The threshold is determined based on a moving average to the transmitted nodes to eliminate the



(a) (b)

Fig. 3. Comparison (throughput vs. the number of nodes present in the coverage area) between theoretical (blue) and simulated (green) throughput values when using DSSS (a) and OFDM (b).

TABLE I
DETECTION THRESHOLDS (PACKETS/SECOND) EMPLOYED IN THE SIMULATIONS.

Number of Nodes	OFDM	DSSS
5	305	115
10	115	55
20	60	28
50	20	17

possibility of false positives due to the bursty nature of the transmissions.

The detection algorithm resides in all the node, and it is run simultaneously when the network communication is in session. The OPNET code was modified to implement this algorithm to facilitate the simulations. Each node listens to the network and creates a number of buckets equal to the number of transmitting nodes. Also, each node solves the Markov chain according to the number of nodes in the network to determine the detection threshold values. Every RTS packet sent by a node is counted towards a moving average. Once a specific node goes above the determined threshold, it is flagged as an attacker. Next, the isolation algorithm described in Sect. IV begins execution.

To validate the theoretical values, first we solved the Markov Chain using Matlab [5] and obtained the theoretical throughputs for different numbers of nodes in the network. Then, we compared the obtained throughput values to the throughputs obtained by OPNET [3] simulations. Figures 3a and 3b presents the comparison between the theoretical (blue line) and simulation (green line) throughput values in packets/second when using Direct-Sequence Spread Spectrum (DSSS) [1] and Orthogonal Frequency-Division Multiplexing (OFDM) [1] modulation techniques, respectively. It is noticeable that the theoretical results are slightly higher. This is due to the inefficiencies in the wireless medium. Table I presents the detection thresholds derived from the theoretical values presented in Figs. 3a and 3b depending on the number of nodes present in the network. Using the theoretical values as detection baselines act as a guard to eliminate false positives.

IV. ISOLATION ALGORITHM

The main goal behind the isolation algorithm is to isolate the attacking node by having all the honest nodes switch to a different channel frequency. The isolation algorithm consists of three phases: *initialization phase*, *registration phase*, and *isolation phase*.

During the initialization phase the system is setup and it occurs once for the lifetime of the system. The registration phase occurs when a new node is given access to the system. During this phase, the node is given the information required to properly run the isolation phase. Notice that these two phases are executed by an authority responsible for setting up the nodes (i.e., by installing the firmware containing our modified IEEE 802.11 DCF MAC layer protocol). For simplicity, we call this authority the *registration authority*. The isolation phase is initiated at the completion of the detection algorithm, and it is also simultaneously run at all the nodes. During this phase, the nodes do the actual isolation of the attacker and move to a new channel frequency. It is important to note that the isolation phase does not require any coordination from a central authority, and this is a key requirement for our solution to be distributed.

We present two concrete methods for realizing the isolation algorithm, namely the *chain method* and the *broadcast method*. The novelty of our isolation algorithm is actually the broadcast method. The chain method is an identity-based encryption-based version of the regular public-key encryption-based isolation algorithm presented in [2]. It is given here for completeness and to facilitate easier comparison. Since both versions of the isolation algorithm rely on public-key cryptography, we provide a review of the required cryptographic background in Sect. A. In Sects. IV-A and IV-B, we present the details of the chain method and the broadcast method, respectively. Next, in Sect. IV-C, we present a brief analysis of the two techniques.

A. The Chain Method

The chain method is based on an IBE-CCA-secure scheme such as that of Boneh and Franklin [6]. The idea in a nutshell is as follows. Let H_i denote the honest node with the i-th highest

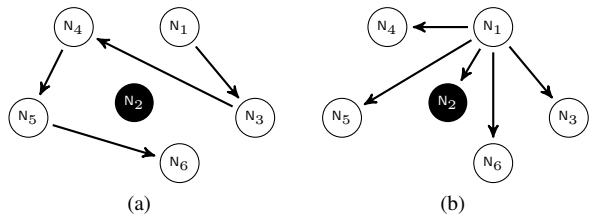


Fig. 4. An example showing the flow of messages in the chain method (a) and in the broadcast method (b). The node N_2 is the attacker.

MAC address. During the initialization phase, the registration authority initializes the IBE scheme. During the registration phase, the nodes are given the IBE secret keys corresponding to their MAC addresses. The isolation phase goes as follows. H_1 first picks a new channel frequency uniformly at random, sends that frequency to H_2 encrypted under the MAC address of H_2, and hops to the newly chosen channel frequency. After receiving the encrypted channel frequency, H_2 decrypts it, sends it to H_3 re-encrypted under the MAC address of H_3, and hops to the decrypted channel frequency. As shown in Fig. 4a, this chain of messages continues until all the nodes except the attacker receive the new channel frequency and hop to that frequency. Although the attacker can still eavesdrop on the encrypted messages, the security of the IBE scheme guarantees that he is unable to obtain the new channel frequency.

Let $\Pi = (\mathsf{Setup}, \mathsf{Extract}, \mathsf{Encrypt}, \mathsf{Decrypt})$ denote an IBE-CCA-secure scheme and 1^λ denote the security parameter. We give the formal details of the chain method below. As explained earlier, the registration authority runs the initialization and registration phases. The nodes only run the isolation phase at the completion of the detection algorithm.

Initialization: Compute $(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda)$ and save $(\mathsf{MPK}, \mathsf{MSK})$ for later use.

Registration: Let j denote the new node and M_j denote its MAC address. Compute $sk_{\mathsf{M}_j} \leftarrow \mathsf{Extract}(\mathsf{MPK}, \mathsf{MSK}, \mathsf{M}_j)$ and return $(\mathsf{MPK}, sk_{\mathsf{M}_j})$ to the new node.

Isolation: Let j denote the current node and M_j denote its MAC address. Proceed as follows:

1) If M_j is the highest MAC address,
   a. Let k be the honest node with the next highest MAC address M_k
   b. Pick a new channel frequency f at random
   c. Compute $c_k \leftarrow \mathsf{Encrypt}(\mathsf{MPK}, \mathsf{M}_k, f)$
   d. Send c_k to node k and hop to frequency f
2) Otherwise,
   a. Let i be the honest node with the previous highest MAC address M_i
   b. Wait for a ciphertext c_j from node i
3) When c_j is received,
   a. Compute $f := \mathsf{Decrypt}(\mathsf{MPK}, sk_{\mathsf{M}_j}, c_j)$
   b. If M_j is the lowest MAC address, hop to frequency f and *terminate* the isolation phase
   c. Otherwise, let k be the honest node with the next highest MAC address M_k
   d. Compute $c_k \leftarrow \mathsf{Encrypt}(\mathsf{MPK}, \mathsf{M}_k, f)$
   e. Send c_k to node k and hop to frequency f

B. The Broadcast Method

We now present the details of the second method of isolation, the broadcast method. In order to instantiate this method, we use a BE-CCA-secure scheme such as that of Dodis and Fazio [7]. The idea is as follows. The initialization and the registration phases follow analogously to the chain method. During the isolation phase, the honest node with the highest MAC address first picks a new channel frequency uniformly at random, encrypts that frequency under all the MAC addresses of the rest of the honest nodes, broadcasts that ciphertext to all the nodes (including the attacker), and hops to the new channel frequency. When the other honest nodes receive this ciphertext, they decrypt it to obtain the new channel frequency and then hop to that frequency. Figure 4b depicts this process. Notice that although the attacker receives the ciphertext, he is not able to decrypt it because he is not in the set of legal recipients of the ciphertext.

Let $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ denote an BE-CCA-secure scheme and 1^λ denote the security parameter. Let N denote the total number of nodes in the system. Given below are the formal details of the broadcast method. The nodes only run the isolation phase at the completion of the detection algorithm. The initialization and registration phases are run by the registration authority.

Initialization: Compute $(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda, N)$ and save $(\mathsf{MPK}, \mathsf{MSK})$ for later use.

Registration: Let j denote the new node and M_j denote its MAC address. Compute $sk_{\mathsf{M}_j} \leftarrow \mathsf{KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \mathsf{M}_j)$ and return $(\mathsf{MPK}, sk_{\mathsf{M}_j})$ to the new node.

Isolation: Let j denote the current node and M_j denote its MAC address. Proceed as follows:

1) If M_j is the highest MAC address,
   a. Let S denote the set of all the honest nodes and M_S denote the set of their MAC addresses
   b. Pick a new channel frequency f at random
   c. Compute $c \leftarrow \mathsf{Encrypt}(\mathsf{MPK}, \mathsf{M}_S, f)$
   d. Broadcast c and hop to frequency f
2) Otherwise,
   a. Let i be the honest node with the highest MAC address M_i
   b. Wait for a ciphertext c from node i
3) When c is received,
   a. Let S denote the set of all the honest nodes and M_S denote the set of their MAC addresses
   b. Compute $f := \mathsf{Decrypt}(\mathsf{MPK}, sk_{\mathsf{M}_j}, \mathsf{M}_S, c)$
   c. Hop to frequency f

C. Analysis

First, it is important to note that both the chain and the broadcast methods can handle multiple uncoordinated attackers attacking the network at the same time. In the case of chain method, each honest node always makes sure to skip any attacker found using the detection algorithm when choosing which node to send the encrypted channel frequency. As for broadcast method, the broadcasting node (i.e., the honest node with the highest MAC address) always makes sure to exclude the attackers from the set of recipients.

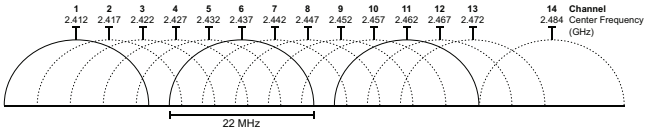


Fig. 5. Frequency to channel mapping for IEEE 802.11b/g [8].

Lets compare the running time of our isolation methods for N users under r uncoordinated attackers. Notice that in this case the chain method requires at least $N - r - 1$ rounds of communication for all the honest nodes to switch the channel frequency. However, due to the collisions with the attacker who is constantly trying to occupy the channel, the actual number of rounds required for the chain method could be much larger. Whereas in the broadcast method, it requires only one round since the broadcast ciphertext is sent to all the nodes (including the attackers) at once. This difference in the number of rounds is the main reason for the tremendous efficiency gain enjoyed by the broadcast method, and it is clearly shown in our simulation results in the following section.

V. Simulations & Results

A. Configuration

The nodes in the first round of simulations are configured to use DSSS modulation technique with IEEE 802.11b standards. DSSS operates in the 2.4 GHz band. Each channel has a width of 22. The rates defined in the IEEE 802.11 standard are 1 Mbps and 2 Mbps and the rates in the IEEE 802.11.b standard are 5.5 Mbps and 11 Mbps. Only the first 11 channels are used in the United States as shown in Fig. 5. Table IIIa lists the parameters configured in every node in the network during all the simulation runs. Multiple scenarios, regarding the number of nodes, are simulated to validate our algorithm. In all the simulation runs, the nodes are placed randomly in an area of 500 m $\times$ 500 m. All nodes are located within the same physical coverage. We modified the OPNET simulator to incorporate our algorithm with the IEEE 802.11 firmware in the simulator.

The second round of simulations operates on nodes using OFDM modulation technique. OFDM operates in the 2.4 GHz band. The rates supported by IEEE 802.11g are 6, 9, 12, 18, 24, 36, 48, and 54 Mbps. The channel to frequency mapping is shown in Fig. 5. Please note that Fig. 5 is just a schematic way of visualizing the channels and it does not reflect the reality of the OFDM sides which are sharper than the DSSS sides to reduce the interference between the channels.

The IBE-CCA-secure identity-based encryption scheme used for the simulation of the chain method is the one from Boneh and Franklin [6]. To simulate the broadcast method, we employed the BE-CCA-secure broadcast encryption scheme of Dodis and Fazio [7] which is based on the Subset Cover Framework of Naor et al. [9]. Table IIIb contains the parameters of these crypto systems employed in our simulations.

B. Results

Figures 6a to 6d show the comparison between the two isolation methods using DSSS modulation technique. The red lines represent the traffic (packets/second) sent in the chain method and the blue lines represent the traffic (packets/second) in the broadcast method. To prove the concept, we simulated the two methods with several network sizes (5, 10, 20, and 50 nodes) under a single attacker. The attacker starts the attack in all cases at the fifth second of the communication session. As seen in these figures, the broadcast method outperformed the chain method, and as a result, the network healed and started to re-communicate much faster reducing the impact of the attack. The same outcome can be seen in our simulations that use the OFDM modulation technique (Figs. 7a and 7b).

Figure 8 shows a setting where two attackers activated their attack mode in two different times (5th and 10th seconds of the communications session). Even in this scenario, our algorithm reacted effectively and minimized the impact of the attacks. Again, the broadcast method reacted much faster compared to the chain method leading to a shorter recovery time.

VI. Related Work

Because of the randomness in selecting a back-off value, detecting malicious back-off manipulation is a very challenging task [10]–[12] that has been the focus of much prior research. [11], [13] assumed Access Points (AP) to be trusted nodes that act as watchdogs in monitoring and controlling all other nodes and their back-off timers, which is a clear deviation from the IEEE 802.11 standards and thus harms interoperability. Our algorithm, on the contrary, is compatible with nodes running the original IEEE 802.11 standards. In [14], the authors assumed the usage of two sub-component modules for detecting the misbehaving nodes in two stages, namely the *throughput monitoring modules* for identifying the suspect greedy node and the *low power probing module* to identify the real misbehaving nodes. Serrano in [15] proposed a statistical method to detect misbehaving nodes via their re-transmission patterns, but his method depends on very tight clocks (in the order of microseconds) to follow randomly generated values, an approach prone to very high level of inaccuracy.

Our approach, on the other hand, can be distributed as in [2] and is designed to work in an environment with or without a central authority. [16] assumes that the attacker will cooperate in the attack avoidance mechanism which is hardly realistic for any network under attack. The authors of [17] introduce new parameters to indicate the level of cooperation from each node. In [18] the author proposes to analyze the distribution of inter-delivery times between two consecutive successful transmissions. This is a very challenging task because it requires very accurate clock readings (in the order of microseconds) to detect the selfish behavior.

VII. Conclusion

In this paper, an effective technique was presented to detect an attacker who manipulates the back-off timer by inserting a malicious code into the NIC's firmware to capture the

TABLE II

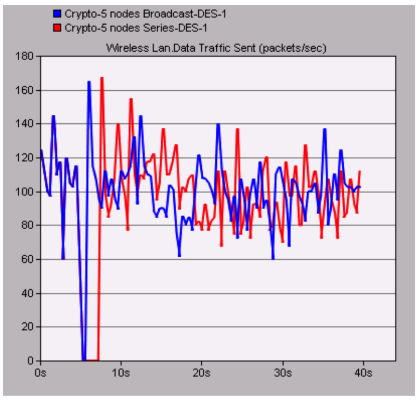
(A) NETWORK PARAMETERS. (B) CRYPTOGRAPHIC PARAMETERS FOR N NODES AND r ATTACKERS. ENCRYPTION/DECRYPTION TIMES FOR A SINGLE ROUND IS GIVEN FOR THE CHAIN METHOD.

Parameter	DSSS	OFDM
Slot Time (σ)	20 µs	9 µs
SIFS	10 µs	10 µs
DIFS	50 µs	28 µs
PHY Header	192, 96 µs	60 µs
MAC Header	28 B	246 b
ACK	14 B	134 b
CTS	14 B	134 b
RTS	20 B	182 b
Channel Bit Rate	11 Mbps	11 Mbps
$\mathsf{CW}_{\mathsf{min}}$, $\mathsf{CW}_{\mathsf{max}}$	31, 1023	15, 1023
Packet Size	8,000 b	10,000 b
Signal Extension	N/A	6 µs

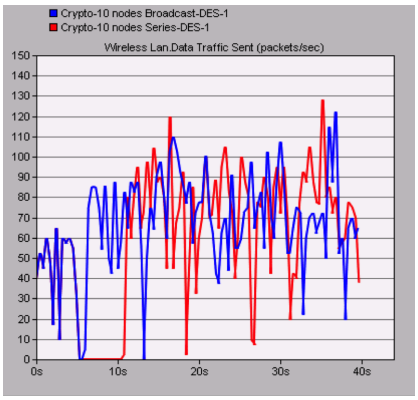
(a)

Parameter	Chain	Broadcast
Crypto System	[6]	[7]
MPK Length	342 B	342 B
MSK Length	28 B	28 B
sk Length	57 B	$57 \log(N+1)$ B
c Length	328 B	$328\, r \log(\frac{N}{r})$ B
Encryption Time	55.6 ms	54.2 ms
Decryption Time	45.1 ms	48.6 ms
Rounds	$N-r-1$	1

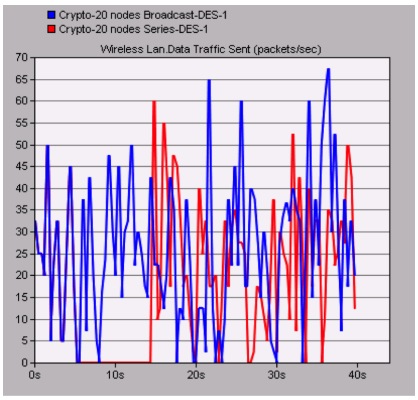
(b)



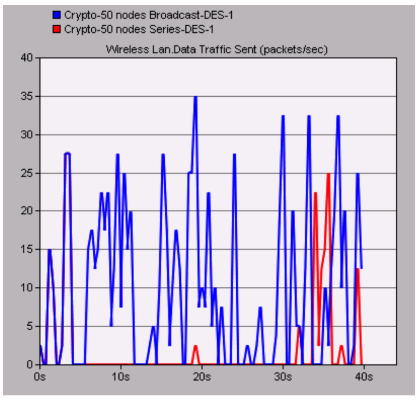
(a)



(b)

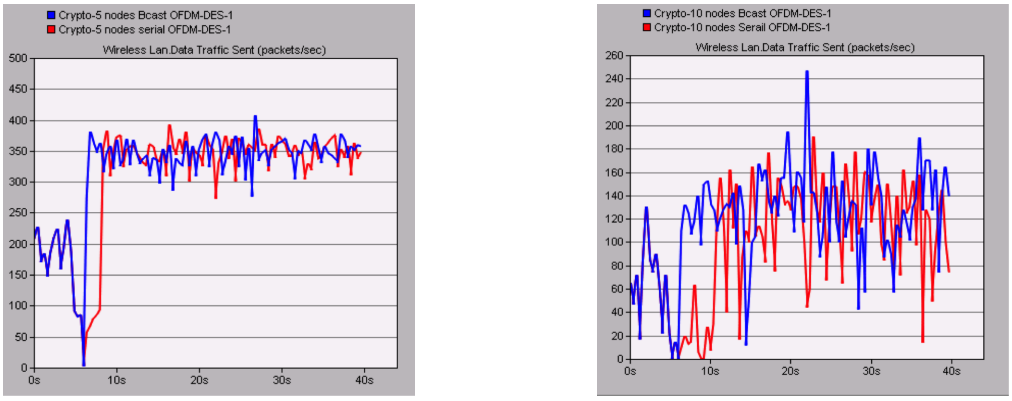


(c)



(d)

Fig. 6. Traffic Sent (packets/second) using DSSS for 5 (a), 10 (b), 20 (c), and 50 (d) nodes in chain method (red line) vs. broadcast method (blue line).



(a) (b)

Fig. 7. Traffic Sent (packets/second) using OFDM for 5 (a) and 10 (b) nodes in chain method (red line) vs. broadcast method (blue line).

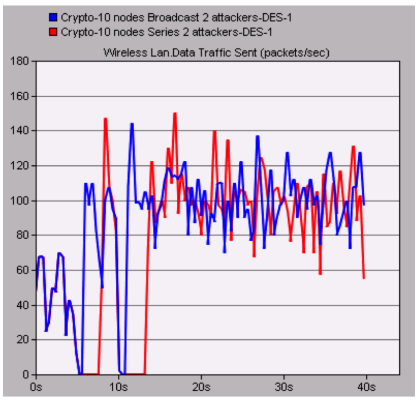


Fig. 8. Traffic Sent (packets/second) using DSSS for 10 nodes in chain method (red line) vs. broadcast method (blue line) under two attackers.

channel and prevent legitimate users from communicating. The algorithm presented is applicable to smartphones and tablets that connect to the Internet via the Wi-Fi technologies. The attacker investigated in this paper is a trusted insider node that has turned into a rogue node after passing all initial authentication steps.

Markov chain modeling results were used to set the detection baselines of the detection algorithm, and a new broadcast encryption-based channel hopping technique called the broadcast method was introduced to isolate the attacker and mitigate its impact of the attack. The OPNET simulation results indicated that the new isolation technique allows the nodes to recover from an attack much faster than the chain method [2] does. This tremendously reduces the downtime for the legitimate users.

Appendix

A. Identity-Based Encryption

Identity-based encryption (IBE) is a variant of public-key encryption in which the public key of a user is an arbitrary bit-string. This notion was originally proposed by Shamir in 1984 [19], and the first efficient and provably secure construction was proposed by Boneh and Franklin in 2001 [6]. Since then, there have been several IBE constructions proposed in the cryptographic literature (e.g., [20]–[22]). Given below is the formal definition of an IBE scheme.

Definition A.1: An IBE scheme, associated with an identity space $\mathcal{ISP}$, a message space $\mathcal{MSP}$, and a ciphertext space $\mathcal{CSP}$, is a tuple of probabilistic polynomial time (PPT) algorithms $(\mathsf{Setup}, \mathsf{Extract}, \mathsf{Encrypt}, \mathsf{Decrypt})$ such that:

- $(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda)$**:** Setup takes the security parameter 1^λ as input and outputs the master public key MPK and the master secret key MSK.
- $sk_I \leftarrow \mathsf{Extract}(\mathsf{MPK}, \mathsf{MSK}, I)$**:** Extract takes MPK, MSK, and an identity $I \in \mathcal{ISP}$ as inputs and outputs a secret key sk_I for I.
- $c \leftarrow \mathsf{Encrypt}(\mathsf{MPK}, I, m)$**:** Encrypt takes MPK, an identity I, and a message $m \in \mathcal{MSP}$ as inputs and outputs a ciphertext $c \in \mathcal{CSP}$.
- $m/\perp := \mathsf{Decrypt}(\mathsf{MPK}, sk_I, c)$**:** Given MPK, a secret key sk_I, and a ciphertext c, Decrypt either outputs a message m or the failure symbol $\perp$. Decrypt is assumed to be deterministic.

Correctness. For every $I \in \mathcal{ISP}$, and $m \in \mathcal{MSP}$, if sk_I is output by Extract $(\mathsf{MPK}, \mathsf{MSK}, I)$ then $\mathsf{Decrypt}(\mathsf{MPK}, sk_I, \mathsf{Encrypt}(\mathsf{MPK}, I, m)) = m$. $\Diamond$

Security. There are two main notions of security provided by identity-based encryption schemes: security against chosen-plaintext attack (IBE-CPA) and security against chosen-ciphertext attack (IBE-CCA). Informally, an IBE-CPA-secure scheme gives away no non-trivial information regarding the encrypted message. An IBE-CCA-secure scheme additionally guarantees that no PPT adversary that is given a valid challenge ciphertext is able to generate *another* valid ciphertext without the required secret key.

B. Broadcast Encryption

Conventional public-key encryption schemes allow secret transmission of data in one-to-one communication. The setting

of public-key broadcast encryption (BE), instead, allows one-to-many secret communication of data. Since the introduction by Fiat and Naor [23], this problem has also received significant attention from the cryptographic research community (e.g., [7], [9], [24]–[31]). The following is the formal definition of a BE scheme.

Definition A.2: A BE scheme, associated with a universe of users $U = [1, N]$, a message space $\mathcal{MSP}$, and a ciphertext space $\mathcal{CSP}$, is a tuple of PPT algorithms $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ such that:

- $(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\lambda, N)$**:** Setup takes the security parameter 1^λ and the number of users in the system N and outputs the master public key MPK and the master secret key MSK.
- $sk_i \leftarrow \mathsf{KeyGen}(\mathsf{MPK}, \mathsf{MSK}, i)$**:** KeyGen takes MPK, MSK, and a user $i \in U$ as inputs and outputs a secret key sk_i for the user i.
- $c \leftarrow \mathsf{Encrypt}(\mathsf{MPK}, S, m)$**:** Encrypt takes MPK, a set of receivers $S \subseteq U$, and a message $m \in \mathcal{MSP}$ as inputs and outputs a ciphertext $c \in \mathcal{CSP}$.
- $m/\perp := \mathsf{Decrypt}(\mathsf{MPK}, sk_i, S, c)$**:** Given MPK, a secret key sk_i, a set of receivers S, and a ciphertext c, Decrypt either outputs a message m or the failure symbol $\perp$. Decrypt is assumed to be deterministic.

Correctness. For every $S \subseteq U$, $i \in S$, and $m \in \mathcal{MSP}$, if sk_i is output by $\mathsf{KeyGen}(\mathsf{MPK}, \mathsf{MSK}, i)$ then $\mathsf{Decrypt}(\mathsf{MPK}, sk_i, S, \mathsf{Encrypt}(\mathsf{MPK}, S, m)) = m$. $\Diamond$

Security. Similar to identity-based encryption schemes, broadcast encryption schemes also provide two notions of security: security against chosen-plaintext attack (BE-CPA) and security against chosen-ciphertext attack (BE-CCA). BE-CPA security guarantees that the ciphertext does not leak any non-trivial information regarding the encrypted message even if the adversary is allowed to corrupt users (of course, excluding any corrupted user in the challenge ciphertext). BE-CCA security additionally guarantees that no PPT adversary that is given a valid challenge ciphertext is able to generate *another* valid ciphertext without any of the required secret keys corresponding to the users of the challenge ciphertext.

References

[1] IEEE Standards Association, "Part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications," IEEE, Tech. Rep., 2012.

[2] J. Soryal and T. Saadawi, "Byzantine attack isolation in IEEE 802.11 wireless ad-hoc networks," in *The 8th IEEE International Workshop on Wireless and Sensor Networks Security—WSNS*, 2012.

[3] Riverbed Technology, "OPNET, application and network performance," http://www.opnet.com.

[4] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 3, pp. 535–547, 2000.

[5] MathWorks, "MATLAB, the lanugage of technical computing," http://www.mathworks.com/products/matlab/.

[6] D. Boneh and M. K. Franklin, "Identity-based encryption from the weil pairing," in *Advances in Cryptology—CRYPTO*, 2001, pp. 213–229.

[7] Y. Dodis and N. Fazio, "Public-key broadcast encryption for stateless receivers," in *Digital Rights Management—DRM*, 2002, pp. 61–80.

[8] *Wireless Networking in the Developing World*, 2nd ed. http://wndw.net, 2007.

[9] D. Naor, M. Naor, and J. Lotspiech, "Revocation and tracing schemes for stateless receivers," in *Advances in Cryptology—CRYPTO*, 2001, pp. 41–62.

[10] J. Bellardo and S. Savage, "802.11 denial-of-service attacks: Real vulnerabilities and practical solutions," in *Proceedings of the USENIX Security Symposium*, 2003, pp. 15–28.

[11] M. Raya, J.-P. Hubaux, and I. Aad, "DOMINO: A system to detect greedy behavior in IEEE 802.11 hotspots," in *The 2nd International Conference on Mobile Systems, Applications, and Services—MobiSys*, 2004, pp. 84–97.

[12] S. Radosavac, A. A. Cárdenas, J. S. Baras, and G. V. Moustakides, "Detecting IEEE 802.11 MAC layer misbehavior in ad hoc networks: Robust strategies against individual and colluding attackers," *Journal of Computer Security*, vol. 15, no. 1, pp. 103–128, Jan 2007.

[13] A. M. Alsahag and M. Othman, "Enhancing wireless medium access control layer misbehavior detection system in IEEE 802.11 network," *Journal of Computer Science*, vol. 4, no. 11, p. 951, 2008.

[14] K. Pelechrinis, G. Yan, S. Eidenbenz, and S. Krishnamurthy, "Detection of selfish manipulation of carrier sensing in 802.11 networks," *IEEE Transactions on Mobile Computing*, vol. 11, no. 7, pp. 1086–1101, 2012.

[15] P. Serrano, A. Banchs, V. Targon, and J. Kukielka, "Detecting selfish configurations in 802.11 WLANs," *IEEE Communications Letters*, vol. 14, no. 2, pp. 142–144, 2010.

[16] V. N. Lolla, L. K. Law, S. V. Krishnamurthy, C. Ravishankar, and D. Manjunath, "Detecting MAC layer back-off timer violations in mobile ad hoc networks," in *The 26th IEEE International Conference on Distributed Computing Systems—ICDCS*, 2006, pp. 63–63.

[17] R. P. Bora, D. Harihar, and S. Sehrawat, "Detection, penalization and handling of misbehavior in ad hoc wireless networks," *IAENG International Journal of Computer Science*, vol. 33, no. 1, pp. 14–18, 2007.

[18] Y. Rong, *Detecting MAC Layer Misbehavior and Rate Adaptation in IEEE 802.11 Networks: Modeling and SPRT Algorithms*. ProQuest, 2008.

[19] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Advances in Cryptology—CRYPTO*, 1984, pp. 47–53.

[20] D. Boneh and X. Boyen, "Secure identity based encryption without random oracles," in *Advances in Cryptology—CRYPTO*, 2004, pp. 443–459.

[21] D. Boneh, C. Gentry, and M. Hamburg, "Space-efficient identity based encryption without pairings," in *IEEE Symposium on Foundations of Computer Science—FOCS*, 2007, pp. 647–657.

[22] B. Waters, "Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions," in *Advances in Cryptology—CRYPTO*, 2009, pp. 619–636.

[23] A. Fiat and M. Naor, "Broadcast encryption," in *Advances in Cryptology—CRYPTO*, 1993, pp. 480–491.

[24] J. A. Garay, J. Staddon, and A. Wool, "Long-lived broadcast encryption," in *Advances in Cryptology—CRYPTO*, 2000, pp. 333–352.

[25] D. Halevy and A. Shamir, "The LSD broadcast encryption scheme," in *Advances in Cryptology—CRYPTO*, 2002, pp. 47–60.

[26] Y. Dodis and N. Fazio, "Public-key trace and revoke scheme secure against adaptive chosen ciphertext attack," in *Public Key Cryptography—PKC*, 2003, pp. 100–115.

[27] Y. Dodis, N. Fazio, A. Kiayias, and M. Yung, "Scalable public-key tracing and revoking," in *ACM Symposium on Principles of Distributed Computing—PODC*, 2003, pp. 190–199, invited to the Special Issue of Journal of Distributed Computing PODC 2003.

[28] Y. Dodis, N. Fazio, A. Lysyanskaya, and D. Yao, "ID-based encryption for complex hierarchies with applications to forward security and broadcast encryption," in *ACM Conference on Computer and Communications Security—CCS*, 2004, pp. 354–363.

[29] D. Boneh, C. Gentry, and B. Waters, "Collusion resistant broadcast encryption with short ciphertexts and private keys," in *Advances in Cryptology—CRYPTO*, 2005, pp. 258–275.

[30] D. Boneh and B. Waters, "A fully collusion resistant broadcast, trace, and revoke system," in *ACM Conference on Computer and Communications Security—CCS*, 2006, pp. 211–220.

[31] C. Gentry and B. Waters, "Adaptive security in broadcast encryption systems (with short ciphertexts)," in *Advances in Cryptology—EUROCRYPT*, 2009, pp. 171–188.