Taming Cross-Layer Attacks in Wireless Networks: A Bayesian Learning Approach

Liyang Zhang, Francesco Restuccia, *Member, IEEE*,
Tommaso Melodia, *Fellow, IEEE* and Scott M. Pudlewski, *Member, IEEE*

Abstract— Wireless networks are extremely vulnerable to a plethora of security threats, including eavesdropping, jamming, and spoofing, to name a few. Recently, a number of next-generation cross-layer attacks have been unveiled, which leverage small changes on one network layer to stealthily and significantly compromise another target layer. Since cross-layer attacks are stealthy, dynamic and unpredictable in nature, novel security techniques are needed. Since models of the environment and attacker's behavior may be hard to obtain in practical scenarios, machine learning techniques become the ideal choice to tackle cross-layer attacks. In this paper we propose FORMAT, a novel framework to tackle cross-layer security attacks in wireless networks. FORMAT is based on Bayesian learning and made up by a *detection* and a *mitigation* component. On one hand, the attack detection component constructs a model of observed evidence to identify stealthy attack activities. On the other hand, the mitigation component uses optimization theory to achieve the desired trade-off between security and performance. The proposed FORMAT framework has been extensively evaluated and compared with existing work by simulations and experiments obtained with a real-world testbed made up by Ettus Universal Software Radio Peripheral (USRP) radios. Results demonstrate the effectiveness of the proposed methodology as FORMAT is able to effectively detect and mitigate the considered cross-layer attacks.

Index Terms—Security, Wireless, Cross-layer, Bayesian, Machine Learning, USRP, Software-defined Radio.

1 Introduction

Owing to the broadcast nature of the transmission medium, wireless networks are extremely vulnerable to a plethora of security threats, including eavesdropping, denial-of-service (DoS), spoofing, message falsification/injection, and jamming, just to name a few [1]. Traditionally, security threats are tackled with mechanisms which are implemented at different network layers [2–7]. On the other hand, this approach does not address the newly emerging threats of *cross-layer attacks* [8–16], which jointly utilize multiple "sub-attacks" on different layers to achieve a single, well-coordinated target.

Since their introduction, cross-layer attacks have evolved into ever complex forms (we refer the reader to Section 2 for a detailed literature review). *In this paper, we focus on a new family of cross-layer attacks that leverage cross-layer interactions in wireless protocols to improve their effectiveness.* Specifically, it is well known that in many network protocols functionalities such as power allocation, channel selection, and routing decisions are jointly optimized with a common objective [17–19], resulting in layers that are closely coupled with each other. The adversary can then exploit this weakness by attacking a *helping* layer using small-scale activities, having instead a *target* layer as objective. As long as the helping and target layers are coupled, the attack will lead to the defender's *responsive* change on the target layer. With carefully-tuned attack activities and objectives, the defender's reaction will favor the attacker's objective. Software-defined radios further facilitate such attacks by easing the manipulation of physical-layer dynamics [20].

To illustrate this type of cross-layer attacks, we now provide some examples.

1) *MAC Poisoning.* Let us suppose a node has two frequency channels (f_0 and f_1) dynamically available for communication. Let us also suppose f_0 is experiencing more interference than f_1, thus the node is using frequency f_1. The target of the adversary is decreasing the node's throughput. In case of a single-layer attack, the attacker may directly jam the physical layer on f_1. However, in case of cross-layer attack, the attacker may influence the medium access control (MAC) layer by jamming the channel reservation messages or periodically falsifying such messages on frequency f_1, and thus inducing the node to eventually switch to f_0. As a consequence, this strategy will eventually lead the node n to experience a lower throughput.

2) *Hammer-and-Anvil [15].* Let us consider a wireless multi-hop network as illustrated in Fig. 1, where node n wants to minimize the end-to-end delay to the sink.

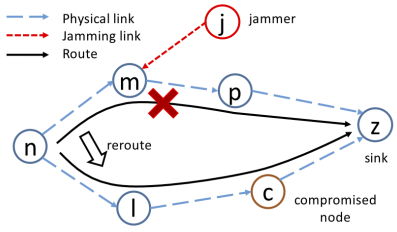


Fig. 1. Illustration of hammer-and-anvil attack.

The attack takes place by using a jammer and a compromised node, which we assume is able to undermine communication with selective forwarding, traffic analysis, or

- *L. Zhang, F. Restuccia, and T. Melodia are with the Department of Electrical and Computer Engineering, Northeastern University, Boston, MA, 02215 USA (e-mail: {liyangzh, frestuc, melodia}@ece.neu.edu).*
- *S.M. Pudlewski is with the Air Force Research Laboratory, RITF, Rome, NY, 13440 USA (e-mail: scott.pudlewski.1@us.af.mil)*

Manuscript received Month X, YYYY; revised Month Y, ZZZZ.

decryption, among others. The jammer aims at redirecting traffic to the compromised node. To this end, it selectively jams links that do not lead to the compromised node. With appropriate jamming, the link between n and m will be degraded so much that m is no longer the best next hop for n, Thus, n may select another next hop with a higher utility, which also leads to the compromised node.

3) *TCP Timeout* [21]. In this attack, the adversary aims at stealthily disrupting existing TCP flows of a node. To this end, an attacker can forward a DoS flow intentionally tuned in accordance with the TCP timeout mechanism. Specifically, the attacker can send high-rate but short-duration bursts with a round-trip time (RTT) scale length and a retransmission timeout (RTO) scale period. As a result, the victim will be forced to enter timeout state repeatedly, and be throttled to near-zero throughput. It has been shown in [12, 14, 22] that physical and MAC layer attacks can be used to create DoS with the same pattern.

The examples above highlight the unique characteristics of cross-layer attacks. Most importantly, they leverage small-scale activities at the helping layer to achieve significant damage at the target layer. This implies that the attacker can achieve the same goal with relatively small-scale activities, and therefore remain *undetected*. On the other hand, existing attack detection methods often assume that attacks are conducted always in the same manner and always have the same objective, and that large-scale attacks have to be conducted in order to get significant results [1]. As we have pointed out, this is not necessarily true in cross-layer attacks. Therefore, developing a *detection* and *mitigation* algorithm able to detect and counteract small-scale, dynamic cross-layer attack activities becomes paramount.

This paper makes the following novel contributions:

• We propose a *FramewORk to learn and Mitigate cross-layer security ATtacks* (in short, FORMAT) in wireless networks. FORMAT uses a novel detection scheme based on a particular kind of machine learning [23] technique called Bayesian learning [24], which we use to train classifiers based on multi-layer features to detect and mitigate small-scale malicious activities in the helping layer [25, 26]. Since cross-layer attacks are stealthy, dynamic and unpredictable in nature, machine learning addresses these challenges by detecting and mitigating the attack "on the fly" without requiring a model of the attacker's behavior.

• To evaluate the framework on practical use-case scenarios, we apply the framework to solve the MAC poisoning attack as well as the hammer-and-anvil attack and the TCP timeout attack. Experimental results obtained with simulations and a practical testbed implemented with Ettus USRP software-defined radios [20] show that our framework improves significantly with respect to the state of the art and is able to detect and mitigate such attacks effectively.

2 Related Work

Cross-layer attacks in networks have been the subject of extensive study over the last few years [8, 10–16, 22].

Compared to traditional single-layer attacks, cross-layer attacks jointly utilize multiple "sub-attacks" at different network layers. They can be broadly categorized into two types, depending how the sub-attacks are utilized. In the first type of cross-layer attacks, sub-attacks work *in parallel* towards a common goal. Thamilarasu *et al.* [8] argue that an adversary can jointly use collision on link layer, packet dropping and misdirection on network layer to perform a Denial of Service (DoS) attack. Two cross-layer attacks for cognitive radio networks are considered by Wang *et al.* [13]. In the first one, the objective of reducing channel utilization is achieved by using Report False Sensing Data (RFSD) attack on physical layer and Small Backoff Window (SBW) attacks on MAC layer. The second one aims at creating interference to the primary users (PU). To this end, an RFSD attack is performed to prevent the secondary users (SU) from detecting the PUs, then a routing manipulation attack is used to redirect packets to the SUs close to the PU. Djahel *et al.* [11] investigate an attack aiming at establishing fake symmetric links in MANETs.

Other cross-layer attacks leverage actions on different layers in a more *intertwined* way. In [10], the authors unveil an attack against MANET monitoring. The attacker can spoof MAC layer ACK frames to reduce the reputation of some nodes at the MANET monitoring tool. In this way, the attacker can manipulate the routing in the network, which is based on the reputation of the nodes. Hasan *et al.* [16] consider an attack in GSM networks, where the control messages are eavesdropped to extract the information useful for launching other attacks such as jamming and Base Transceiver Station (BTS) cloning. TCP's congestion control mechanism is well-known to be vulnerable to DoS attacks, as discussed in Section 1 example (3). In [12] and [14], the authors consider cognitive radio networks and argue that the objective of forcing TCP timeout can be achieved with physical/link layer attacks such as Primary User Emulation (PUE) and Objective Function Attack (OFA).

The benefit of considering multi-layer information in attack detection has attracted significant attention. There is a series of works on Intrusion Detection System (IDS) using cross-layer features [13, 27–31], for both single-layer attacks and the first type of cross-layer attacks. The idea is to monitor features on multiple layers so that different single-layer attacks (or parallel sub-attacks for a cross-layer attack) can be detected with a higher accuracy. The unique characteristics of the second type of cross-layer attacks, *i.e.*, the interactions between layers are not considered, therefore they cannot be applied directly to these cross-layer attacks. Bayesian learning has also been widely used for attack detection in network security [32–34]. However, conversely from ours, previous work has focused on single-layer attacks and parallel cross-layer attacks.

3 The FORMAT Framework

In this section, we describe in detail our the FORMAT framework. We first provide some necessary background notions on Bayesian learning, then, we illustrate an overview of the framework. Next, we discuss the threat model and the components of FORMAT in Section 3.3 and 3.4.

3.1 Background

Let us now introduce some definitions and terminology that will be used throughout the paper. Let us define x as a data

point and θ as the parameter of the data point's distribution, *i.e.*, $x \sim p(x \mid \theta)$. We also define α as the hyper-parameter of the θ parameter distribution, *i.e.*, $\theta \sim p(\theta \mid \alpha)$. We define $\mathbf{X}$ as the sample, which is composed by a set of n observed data points, *i.e.*, $x_1, \ldots, x_n$. Finally, we define $\tilde{x}$ as a new data point whose distribution is to be predicted.

Bayesian inference [35] works as follows. Let us define as *prior* the distribution of the parameter θ before any data is observed, *i.e.* $p(\theta \mid \alpha)$. The prior distribution might not be easily determined. In this case, we can use the Jeffrey's prior to obtain the posterior distribution before updating them with newer observations. Let us define $\mathrm{L}(\theta \mid \mathbf{X}) = p(\mathbf{X} \mid \theta)$ as the *likelihood* (also called sampling distribution). The *marginal likelihood* (also called the "evidence") is the distribution of the observed data marginalized over the parameter(s), which is derived as follows:

$$\overbrace{p(\mathbf{X} \mid \alpha)}^{marg.likelihood} = \int_\theta \overbrace{p(\mathbf{X} \mid \theta)}^{likelihood} \cdot \overbrace{p(\theta \mid \alpha)}^{prior} \mathrm{d}\theta.$$

The *posterior* is the distribution of the parameter(s) after taking into account the observed data. This is determined by Bayes' rule, which forms the heart of Bayesian inference:

$$\overbrace{p(\theta \mid \mathbf{X}, \alpha)}^{posterior} = \frac{\overbrace{p(\mathbf{X} \mid \theta)}^{likelihood} \cdot \overbrace{p(\theta \mid \alpha)}^{prior}}{\underbrace{p(\mathbf{X} \mid \alpha)}_{marg.likelihood}} \propto p(\mathbf{X} \mid \theta) \cdot p(\theta \mid \alpha).$$

Bayesian learning predicts the posterior predictive distribution of a new data point marginalized over the posterior, as follows:

$$\overbrace{p(\tilde{x} \mid \mathbf{X}, \alpha)}^{prediction} = \int_\theta \overbrace{p(\tilde{x} \mid \theta)}^{likelihood} \cdot \overbrace{p(\theta \mid \mathbf{X}, \alpha)}^{posterior} \mathrm{d}\theta.$$

We point out that proposing new Bayesian learning methodologies is out of the scope of this paper. Instead, the focus of this paper is to develop a theoretical framework based on Bayesian learning able to detect and mitigate multiple cross-layer attacks.

3.2 An Overview of FORMAT

We describe the proposed FORMAT framework in the following subsections. The main symbols that will be used in the following discussions are summarized in Table 1.

TABLE 1
Summary of main symbols.

Symbol	Description
$\mathbf{V}$	network state vector
$\mathbf{O}$	observations vector
$\mathbf{S}$	legitimate node strategy vector
$\mathbf{A}$	attacker strategy vector
$U(\cdot)$	defender's utility
$G(\cdot)$	attacker's gain
$C(\cdot)$	classifier
B	*a priori* knowledge on the attack features
α, β	weights of security and performance

Fig. 2 provides a high-level overview of the proposed FORMAT framework, which aims to enable legitimate wireless network nodes to detect and mitigate cross-layer security attacks. To achieve this goal, FORMAT implements a detection and mitigation scheme based on Bayesian learning, described as follows.

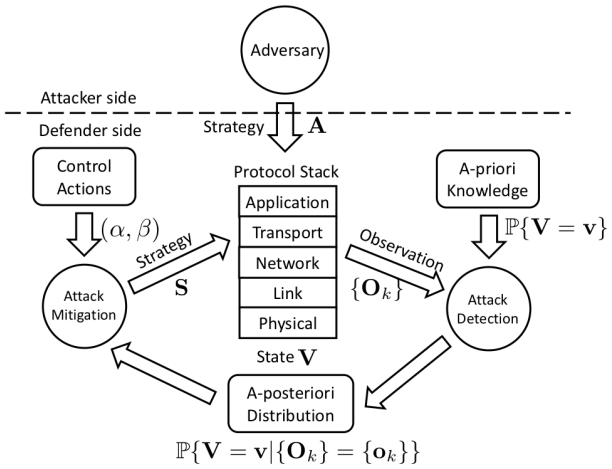


Fig. 2. Attack detection and mitigation framework.

3.2.1 System and Threat Model

We consider a wireless network with a set of legitimate users (*i.e.*, the *defenders*) and an adversary (*i.e.*, the *attacker*). We represent the network state as a set of variables that are referred to multiple network layers, which may include signal-to-interference-plus-noise ratio (SINR) of a link, the channel access probability of a node, the quality of a route, and so on. Let us define $\mathbf{V}$ as the vector holding the network state. The state vector $\mathbf{V}$ may be affected by the strategies of the defenders and the attacker, including power allocation, scheduling, and next hop selection, among others. By denoting the strategies for a defender and an attacker as $\mathbf{S}$ and $\mathbf{A}$, respectively, the vector $\mathbf{V}$ can be expressed as $\mathbf{V} = g(\mathbf{S}, \mathbf{A})$, with $g(\cdot)$ representing a certain functional relationship.

As in [36–38], we assume that both the attacker and the defender(s) are *rational* and decide their strategies iteratively.[1] We define the interval between two updates of the attacker's strategy as one *strategy updating period*. The strategy updating iteration is shown in Fig. 3, where subscripts such as i and $i+1$ are used to distinguish different time periods.

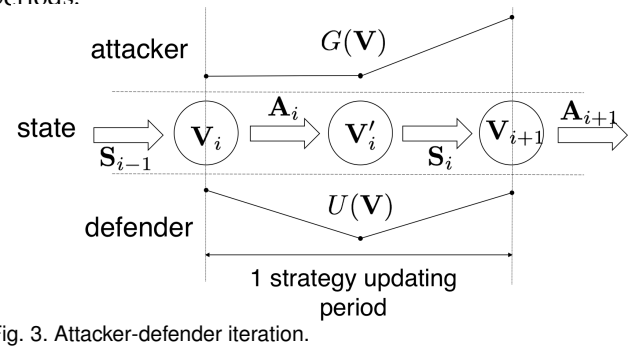


Fig. 3. Attacker-defender iteration.

We assume the defender implements a strategy involving multiple network layers, in which networking variables

1. As a more sophisticated model, the two parties may make their decisions simultaneously. From the perspective of game theory, a minimax rule should be adopted in decision making in this case. However, since minimax rule requires that a player has the knowledge of the possible strategies of the other, it is difficult to adopt it in our model, in which the strategy of the attacker is assumed to be unknown and difficult to detect. Therefore, we will limit the discussion to iterative decision making, and leave the simultaneous decision making scenario to future work.

are jointly optimized by considering a *utility* function $U(\mathbf{V})$. The defender's next strategy is thus decided by solving a cross-layer optimization problem, which takes as input the current network state. More formally,

$$\mathbf{S}_{i+1} = \arg\max_{\mathbf{S}} U(\mathbf{V}_i) = \arg\max_{\mathbf{S}} U(g(\mathbf{S}, \mathbf{A}_i)). \quad (1)$$

Note that the defender does not need to know the attacker's strategy $\mathbf{A}$. All it needs is the network state $\mathbf{V}$, which is a function of $\mathbf{A}$. The detection of $\mathbf{V}$ is one of the objectives of the framework, and will be shown in detail in the following subsections.

In the considered threat model, the attacker launches a cross-layer attack in the following way, as shown in Fig. 3. Suppose the network is at a state $\mathbf{V}_i = \{v_i^1, \ldots, v_i^K\}$, resulting in an attack gain of $G(\mathbf{V}_i)$. The attacker aims at changing some state $v^t, 1 \leq t \leq K$ on a target layer so that the new state $\mathbf{V}_{i+1} = \{v_{i+1}^1, \ldots, v_{i+1}^K\}$ yields a higher attack gain $G(\mathbf{V}_{i+1})$. This objective can be done by directly changing the state component v_i^t to v_{i+1}^t by attacking with strategy $\mathbf{A}_{\text{single}}$ (single-layer attack), but it may be costly. Cross-layer attack strategy $\mathbf{A}_{\text{cross}}$, on the other hand, chooses a helping layer that is closely coupled with the target layer in the underlying wireless protocol, but much easier to attack, and changes the network state v^h on it. An intermediate state $\mathbf{V'}_i = \{v_i'^1, \ldots, v_i'^K\}$ with a changed $v_i'^h$ is resulted, lowering the utility $\mathbf{U}(\mathbf{V}')$ of the defender. The defender always tries to optimize its utility $\mathbf{U}$ according to the underlying protocol. Therefore, it chooses a new strategy $\mathbf{S}_i$ based on (1) for the new state $\mathbf{V}'$ in response. The coupling between layers implies that the state v^t will be changed along with v^h. Therefore, as long as the helping layer is carefully chosen, the resultant new state $\mathbf{V}_{i+1}$ may increase both the defender's utility $\mathbf{U}$ and the attacker's gain $\mathbf{A}$.

The indirect attack approach that exploits cross-layer interactions in the underlying wireless protocols distinguishes cross-layer attacks from single-layer (and the "parallel" cross-layer attacks, see Section 2 for details), and poses unique challenges. First, by choosing a helping layer with drastic interactions with the target layer, the attacker may only need to change v_i^h by a small scale to create an intermediate state $\mathbf{V}'$, in which the defender has to change $v_i'^t$ significantly and results in a state $\mathbf{V}_{i+1}$ in favor of the attacker. In other words, the attack objective can be achieved with small-scale attack activities at a different layer, therefore the attack is exceptionally stealthy. Second, in the second stage of the attack, the responsive action of the defender optimizes its own utility while also increasing the attack gain inadvertently. This suggests simple defense mechanisms diverting from the attacked state (e.g., $\mathbf{V}_{i+1}$ in Fig. 3) may also introduce degradation in utility. Therefore, the security performance trade-off becomes imperative for these attacks.

FORMAT is composed of two parts, the attack detection component and the attack mitigation component. We will introduce them in detail in the following subsections.

3.3 Attack Detection

The attack detection component faces the challenge of stealthy attack activities, and needs to detect slight changes in network states. To this end, we use Bayesian learning to compute the belief that a specific attack is taking place. Specifically, it first estimates the network states resulted by the attack activities based on observed events; then, the result is mapped to possible attacks according to their features by using a set of classifiers that have been trained to recognize the attacks. For the sake of simplicity, in the following we will omit subscripts such as i and $i+1$ since the following discussion only involves one strategy updating period.

The attack activities are directly reflected by the network state changed by the attacker. However, since $\mathbf{V}'$ is only slightly different from $\mathbf{V}$, the defender must be able to detect slight changes in the network state. According to Bayesian learning, the more evidences about an event are accumulated, the closer the result is to the real distribution of the variable. Therefore, the attack detection component will be able to construct a hypothesis with high confidence based on consecutive but not decisive evidences, which perfectly suits the scenario we are considering. Specifically, $\mathbf{V}$ is viewed as a random variable with an unknown distribution. This assumption is valid, since the strategy of the attacker is static in one strategy updating period. Note that some environmental variables may also affect $\mathbf{V}$, but they can be generally assumed to be either constant (e.g., network topology), or a random variable with a static distribution (e.g., channel fading) during one strategy updating period.

To learn the distribution of $\mathbf{V}$, the defender records a set of observable events during a time period. Such events may include a successful (or failed) reception of a bit (or packet), the result of a channel contention, and the end-to-end delay of a message, among others. We denote an event as a random variable (r.v.) $\mathbf{O}_k$. These events reveal different states such as SINR of a link, channel access probability, and quality of a route, but in a probabilistic way. That is, an event $\mathbf{O}_k = \mathbf{o}_k$ happens for a given network state $\mathbf{V} = \mathbf{v}$ with a probability equal to

$$\mathbb{P}\{\mathbf{O}_k = \mathbf{o}_k | \mathbf{V} = \mathbf{v}\} = f(\mathbf{o}_k, \mathbf{v}). \quad (2)$$

We assume the function $f(\mathbf{o}_k, \mathbf{v})$ is available to the defender, by either theoretical formulation (e.g., bit error probability), or training. By assuming that a sequence of independent events $\{\mathbf{o}_k\}_{k=1,\ldots,K}$ have happened, it follows that

$$\mathbb{P}\{\{\mathbf{O}_k\}_{k=1,\ldots,K} = \{\mathbf{o}_k\}_{k=1,\ldots,K} | \mathbf{V} = \mathbf{v}\} = \prod_{k=1}^{K} f(\mathbf{o}_k, \mathbf{v}), \quad (3)$$

which leads to the following a posteriori distribution

$$\mathbb{P}\{\mathbf{V} = \mathbf{v} | \{\mathbf{O}_k\} = \{\mathbf{o}_k\}\} =$$

$$= \frac{\mathbb{P}\{\{\mathbf{O}_k\} = \{\mathbf{o}_k\} | \mathbf{V} = \mathbf{v}\}}{\int_{\mathbf{v}} \mathbb{P}\{\{\mathbf{O}_k\} = \{\mathbf{o}_k\} | \mathbf{V} = \mathbf{v}\} \cdot \mathbb{P}\{\mathbf{V} = \mathbf{v}\} \, \mathrm{d}\mathbf{v}} \cdot \mathbb{P}\{\mathbf{V} = \mathbf{v}\}, \quad (4)$$

where $\mathbb{P}\{\mathbf{V} = \mathbf{v}\}$ is some *a priori* distribution of $\mathbf{V}$, which represents the ideal network state (*i.e.*, without attacks). Note that such quantity is often available – for example, the distribution of SINR on a link can be derived from the fading model, the channel access probability for any node in a network running CSMA/CA is approximately the same, and so on. If accurate knowledge is not available, it is often possible to know some information on it, such as the functional form, and the range of its values [24].

With the *a posteriori* distribution in Equation (4) available, the defender is now aware of the attack activities of the attacker. However, the defender needs information regarding the type of attack that is taking place. To this end, we employ a classifier taking as input both the *a posteriori* distribution of the network state, and features of attacks that the network is prone to. Let us define B as the features of an alleged attack, which represents *a priori* knowledge on typical network state under this attack. Thus, we formally define an attack classifier as follows:

$$C(\mathbb{P}\{\mathbf{V}=\mathbf{v}|\{\mathbf{O}_k\}=\{\mathbf{o}_k\}\},B) \underset{\text{Not attacked}}{\overset{\text{Attacked}}{\lessgtr}} C_{\text{th}}, \tag{5}$$

where C_{th} is a threshold that is set based on an *a priori* training of the system (e.g., by using supervised learning). We would like to point out here that, as a general framework, FORMAT does not, and is unable to specify the kind of classifier to be used or how to train it. When implementing the wireless system, the designer should choose experiment with different classifier (e.g., K Nearest Neighbor (KNN, Support Vector Machine (SVM), Logistic Regression) and choose the best one according to experimental data. The choice of threshold C_{th} depends on the attack and the classifier. For example, if SVM is used, then the classifier is a linear function of $\mathbf{V}$ in the form of $\mathbf{w}^T\mathbf{V}$, *i.e.*, a weighted sum of each network state in the vector $\mathbf{V}$ with a weight vector $\mathbf{w}$. In this case, the threshold is just a vector $\mathbf{b}$ such that $\mathbf{w}^T\mathbf{V}=\mathbf{b}$ is a hyperplane separating most points with the considered attack from most points without the attack.

3.4 Attack Mitigation

Since multiple sub-attacks may be jointly utilized, the mitigation component faces the challenge of a smaller space for strategy. Simple defense mechanisms aiming at one of the sub-attacks may fall to the pocket of another, and therefore suffers in either security or performance. To address this, we let the attack mitigation component to decide a strategy $\mathbf{S}_k$ that optimizes the trade-off between the effort that must be put forth by the defender (which includes the incurred cost of such effort), and the desired security level. To this end, we define a security-performance function:

$$\begin{aligned} h(\mathbf{S}_k,\mathbf{A}_k) &= \alpha\cdot E[U(\mathbf{V}_k)] - \beta\cdot E[G(\mathbf{V}_k)] \\ &= \alpha\cdot E[U(g(\mathbf{S}_k,\mathbf{A}_k))] - \beta\cdot E[G(g(\mathbf{S}_k,\mathbf{A}_k))], \end{aligned} \tag{6}$$

where $E[\cdot]$ is the expectation of a random variable. The expected utility $E[U(\mathbf{V}_k)]$ represents performance and negative attack gain $-E[G(\mathbf{V}_k)]$ represents security. Note $E[U(\mathbf{V}_k)]$ is available since a defender is able to evaluate the performance of its own strategy. The exact value of attack gain $E[G(\mathbf{V}_k)]$ may not be directly measurable, but an estimation is usually available for a specific attack, as long as the pattern in attack activities is acquired. We have shown in Section 4 and 5 how it can be estimated for the two use cases. The mitigation engine decides the optimal strategy by solving

$$\underset{\mathbf{S}_k}{\text{maximize}} \quad h(\mathbf{S}_k,\mathbf{A}_k) \tag{7}$$

$$\text{subject to} \quad E[U(\mathbf{V}_k)] \le U_{\text{th}} \tag{8}$$

$$E[G(\mathbf{V}_k)] \le G_{\text{th}}, \tag{9}$$

where α and β are control variables set by the network operator. By adjusting their values, the defender can regulate the desired trade-off point between performance and security. Constraints (8) and (9) represent the desired performance and security levels.

The values of α and β largely depend on the considered application. For example, video streaming applications usually require high performance on data rate but low security level, whereas a wireless sensor network collecting scientific data usually requires high security level but may tolerate low data rate.

4 Use-case Example: MAC Poisoning

To demonstrate the practical use of the FORMAT framework, we consider two examples of cross-layer attacks. Specifically, the analysis is aimed at illustrating how the framework works in practical scenarios and analyzing its performance. The first example is the MAC poisoning attack, which was briefly introduced in Section 1 and formally defined as follows.

MAC poisoning attacks take place in multi-channel cognitive networks, where the set of available channels is defined as $\mathcal{F}$. There is a set $\mathcal{N}$ of secondary users (SU) that need to access the medium. To this end, an SU $n\in\mathcal{N}$ needs to decide the channel and transmission power to use, which are denoted as vectors $\boldsymbol{\delta}_n=\{\delta_n^f\}_{f\in\mathcal{F}}$ and $\mathbf{P}_n=\{P_n^f\}_{n\in\mathcal{F}}$, where

$$\delta_n^f = \begin{cases} 1 & n \text{ chooses channel } f \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

We assume that to access the channel the SUs leverage a contention based MAC protocol. Thus, SU n needs to compete with other SUs for the access of a channel f once it is chosen. Let us define $\pi_n^f(\delta_n^f,\delta_{-n}^f)$ as the channel access probability. By considering the average SINR as utility function, the strategy of n can be formulated as the optimization problem (11) - (14).

In Equation (14), quantity γ_n^f denotes the achieved SINR on channel f, which is affected by P_n^f as well as the ambient noise W_f. Due to the heterogeneity in primary users (PUs), the noise level and maximum allowed transmitting power may differ from channel to channel, resulting in different achievable SINR. We also assume flat fading, so the average channel gain is the same for each $f\in\mathcal{F}$.

$$\underset{\delta_f,P_f}{\text{maximize}} \quad \textstyle\sum_{f\in\mathcal{F}}\delta_n^f\cdot\pi_n^f(\delta_n^f,\delta_{-n}^f)\cdot\gamma_f(P_n^f,W_f) \tag{11}$$

$$\text{subject to} \quad \delta_n^f\in\{0,1\} \tag{12}$$

$$\textstyle\sum_{f\in\mathcal{F}}\delta_n^f = 1 \tag{13}$$

$$P_n^f \le P_n^{\max} \tag{14}$$

The objective function is optimized by selecting the channel with maximum $\pi_n^f(\delta_n^f,\delta_{-n}^f)\cdot\gamma_n^f(P_{\max}^f,W_f)$. As a consequence, an attacker may degrade the defender's SINR by either raising the interference plus noise level directly, or by lowering the access probability of channels with high achievable SINR so that the defender selects a channel with low achievable SINR. We define the second strategy as a MAC poisoning attack.

A feasible way to lower the channel access probability is to purposely jam the RTS/CTS of the defender. In such case, a failed attempt of an RTS/CTS exchange is considered a collision, and the defender needs to attempt to

access the channel with a doubled-size contention windows. Therefore, the channel access probability of all other nodes will increase, whereas the one of the defender decreases. Compared to directly raising the interference plus noise level, this method costs much less energy of the attacker.

4.1 Attack Detection

The problem of unfair channel access detection has been analyzed in literature. A scenario with selfish nodes that deliberately lower their contention window size is studied in [39], where the authors propose a detection scheme that compares the throughput of different nodes. Under the assumption that the throughput is proportional to channel access probability, if the ratio of two nodes exceeds 1 for some margin ϵ during an observation time window, the node with the higher throughput is considered a selfish one. However, this approach is based on short-time observations that ultimately fail to reveal a statistical characteristic of the malicious behavior. Indeed, the very reason to introduce a time window and a tolerance is to tackle short-time unfairness. Even with this assumption, the method is still not able to detect a selfish node with slightly deviated behavior, as discussed in the paper. Therefore, the method does not fit our scenario, as the attacker may achieve its goal by creating a channel access probability slightly under the fair share for the defender. In [40], the authors use a sequential probability ratio test (SPRT) to establish a null or an alternative hypothesis. However, the simple hypotheses assumption cannot be applied to our case, where the domain of the estimated parameter, *i.e.*, channel access probability, is a continuous set, and it is hard to establish the likelihood for observations of a simple binary hypothesis.

These issues are successfully tackled by our FORMAT framework, which aims at establishing the probability distribution of the network states on a continuous set, rather than probability ratio of simple hypotheses. Moreover, consecutive observations can be utilized recursively to improve the accuracy of the distribution. Applied to the MAC poisoning attack, we have the channel access probabilities π_n^f as the network states, and the objective is to construct *a posteriori* distribution for it based on a series of observable events. For readability, we will eliminate the subscripts and superscripts n and f.

Since the channel access probability affects channel contentions directly, the outcomes can be used as observable events. Let us denote the k-th event as

$$e_k = \begin{cases} 1, & \text{node } n \text{ wins channel } f \text{ in } k-\text{th competition}, \\ 0, & \text{otherwise.} \end{cases} \tag{15}$$

For a sequence of channel competition $\{e_k\}_{k=1,\dots,K}$, we derive the *a posteriori* distribution of π as follows:

$$\mathbb{P}\{\pi = x|\{e_k\}_{k=1,\dots,K}\} = \frac{\mathbb{P}\{\{e_k\}_{k=1,\dots,K}|\pi = x\}}{\int_y \mathbb{P}\{\{e_k\}_{k=1,\dots,K}|\pi = y\}\mathrm{d}y} \cdot \mathbb{P}\{\pi = x\}. \tag{16}$$

Since the contention results are independent, the likelihoods in (16) can be computed recursively as follows:

$$\mathbb{P}\{\{e_k\}_{k=1,\dots,K}|\pi = x\} = \prod_{k=1,\dots,K} \mathbb{P}\{e_k|\pi = x\}. \tag{17}$$

Note the conditional probability for an individual event is given by

$$\mathbb{P}\{e_k|\pi = x\} = \begin{cases} x, & e_k = 1, \\ 1-x, & e_k = 0. \end{cases} \tag{18}$$

The *a priori* distribution can be initialized with a high peak at $p = 1/|\mathcal{N}|_f$, with $\mathcal{N}_f \subset \mathcal{N}$ as the set of nodes contending for channel f. Since RTS/CTS can be overheard, the node is able to estimate them.

The probability for a node n to be attacked is revealed by the *a posteriori* distribution. Similar to [39], the underlying fact that all contending nodes should have the same share of channel access implies a classifier can be designed as follows:

$$\mathbb{P}\{\pi_n^f \leq \tfrac{1}{|\mathcal{N}_f|} - \epsilon\} \underset{\text{Attacked}}{\overset{\text{Not Attacked}}{\lessgtr}} \mathbb{P}_{\text{th}}. \tag{19}$$

The proposed method is able to detect any $\pi < 1/|\mathcal{N}|$. On the other hand, for the sake of simplicity, there is a minimum granularity on the domain of π_n^f, and the margin ϵ is introduced to represent it. Algorithm 1 summarizes the proposed detection algorithm.

Algorithm 1 Bayesian Learning Based MAC Poisoning Attack Detection

1: Given a node n and a channel f, *a priori* distribution $\mathbb{P}\{\pi_n = x\}$;
2: **while** the K-th contention completes **do**
3: Update $\mathbb{P}\{\pi_n|\{e_k\}_{k=1,\dots,K}\}$, according to (16) and e_n^K;
4: **if** The classifier (19) decides that node n is attacked **then**
5: Break;
6: **end if**
7: **end while**

4.2 Attack Mitigation

To tackle an existing attack, the defender n may choose from the following strategies:

1) Switch to another channel f' with a lower achievable SINR;
2) Transmit overlapping other nodes' transmissions with probability $\hat{\pi}_n^f = 1/|\mathcal{N}_f| - \pi_n^f$, in such a way that the real channel access probability is compensated to the fair share of $1/|\mathcal{N}_f|$.

We will denote them as $\mathbf{S}_{f'}$ and $\mathbf{S}_f$. Note that $\mathbf{S}_f$ does not harm fairness, as long as the detection result is correct. However, depending on the tuning of the classifier, the accuracy of detection result may change. Therefore, there is a possibility that the defender will disrupt a legitimate transmission. Meanwhile, the transmission of other nodes degrades the SINR of the overlapping transmission, too. Therefore, the defender should optimize along a trade-off line of both performance (the achievable SINR) and security (collision with others). Denoting the achievable SINR for an overlapping transmission as $\hat{\gamma}_n^f$, the performance function can be formulated as

$$U(\mathbf{S}) = \begin{cases} \pi_n^f \gamma_n^f + \hat{\pi}_n^f \hat{\gamma}_n^f, & \mathbf{S} = \mathbf{S}_f, \\ \pi_n^{f'} \gamma_n^{f'}, & \mathbf{S} = \mathbf{S}_{f'}. \end{cases} \tag{20}$$

The security function describes the probability of collisions with legitimate transmissions resulted from a false positive. Obviously, there is no risk of collision if $\mathbf{S}_{f'}$ is used, and we may normalize the security value for this strategy as 1. For *a posteriori* distribution $\mathbb{P}\{\pi_n^f|\{e_k\}_k\}$, we derive that $1 - \mathbb{P}\{\pi_n^f \leq 1/|\mathcal{N}_f| - \epsilon\}$ is the probability that node n is not attacked, or, in other words, the probability that

TABLE 2
Simulation scenario.

channel	f_1	f_2
power limit (dBm)	10	15
channel gain (dB)	-75	-75
noise level (dBm)	-85	-75
number of other nodes	4	3
achievable SINR (dB)	20	15
fair channel access probability	0.2	0.25

the "defensive" action is unjustified. Therefore, the security function is

$$G(\mathbf{S}) = \begin{cases} \mathbb{P}\{\pi_n^f \leq 1/|\mathcal{N}_f| - \epsilon\}, & \mathbf{S} = \mathbf{S}_f, \\ 1, & \mathbf{S} = \mathbf{S}_{f'}. \end{cases} \quad (21)$$

The node can then mitigate the attack by solving

$$\underset{\mathbf{S} \in \{\mathbf{S}_f, \mathbf{S}_{f'}\}}{\text{maximize}} \quad \alpha \cdot U(\mathbf{S}) + \beta \cdot G(\mathbf{S}), \quad (22)$$

with weights α and β representing the preference on security or performance.

This mitigation scheme can be integrated with the detection and results in an improved Algorithm 2.

Algorithm 2 Joint Detection and Mitigation Algorithm for MAC Poisoning Attack

1: Given a node n and a channel f, *a priori* distribution $\mathbb{P}\{\pi_n = x\}$;
2: **while** The K-th contention completes **do**
3: Based on e_n^K, update $\mathbb{P}\{\pi_n | \{e_k\}_{k=1,\dots,K}\}$, according to (16);
4: **if** $\alpha \cdot U(\mathbf{S}_f) + \beta \cdot G(\mathbf{S}_f) < \alpha \cdot U(\mathbf{S}_{f'}) + \beta \cdot G(\mathbf{S}_{f'})$, for $f' \neq f \in \mathcal{F}$ that maximizes $\alpha \cdot U(\mathbf{S}_{f'}) + \beta \cdot G(\mathbf{S}_{f'})$ **then**
5: Switch to channel f';
6: Break;
7: **end if**
8: **end while**

4.3 Performance Evaluation

We evaluate the efficacy of the framework on the MAC poisoning attack by simulating the following scenario. There are 2 channels, denoted as $\mathcal{F} = \{f_1, f_2\}$. The parameters for each channel are listed in Table 2. Apparently, without the attack, the optimal strategy is to use channel f_1, since it gives the utility value of 4 dB, compared to 3.75 dB for channel f_2. To launch a MAC poisoning attack, the attacker needs to lower the channel access probability π_1 to a value less than 0.1875.

Detection Results. For the sake of simplicity, we show the detection results for the MAC poisoning attack on channel f_1. We simulate the channel contention process by assigning the channel f_1 to each contending nodes according to its channel access probability. To be specific, with K contentions, $\pi_m K$ contentions are won by node $m \in \mathcal{N}_{f_1}$. For the sake of simplicity, the domain of the random variable π_n is set to a discrete set starting from 0 to 1 with a step of 0.001. To leverage the advantage of small granularity, the margin in the classifier is also set to $\epsilon = 0.001$. For the *a priori* distribution, we use a Gaussian distribution centered around the ground truth value of 0.2, with a variance 0.05^2 to account for the error, *i. e.*, $\pi \sim N(0.2, 0.05^2)$.

Fig. 4 (a) depicts the *a posteriori* distribution as a function of the number of events. The results conclude that the *a posteriori* distribution converges to the actual distribution as the number of observed events increasing. When the number of observed events reaches 5000, the curve lies mostly to the left of the discrimination line $x = 1/|\mathcal{N}| - \epsilon$. To further illustrate this point, we compare the discrimination functions of the proposed detection method and that of [39], which is considered the state of the art. Specifically, for the proposed detection, it is the left-hand side of (19); for [39], it is π_n/π_m, for $m \neq n \in \mathcal{N}_{f_1}$. Results are shown in Fig. 4 (b) and conclude that the discrimination function of [39] does not improve with more available observations, conversely from the proposed one. Note the absolute value of the two discrimination functions are not directly comparable, since different classifiers are used. Therefore, the proposed detection method provides more and more accurate result with evidences accumulated.

Mitigation results. Let us now show how the mitigation scheme achieves performance-security trade-off with different detection results. The previous detection results are used to compute the security function (21). For the performance function (20), we use the same settings as in Table 2. The achievable SINR with concurrent transmission is set to $\hat{\gamma}_n^{f_1} = 16.99$ dB, i,e., half the SINR achievable without overlapping transmission. Since the value of the security function is between 0 and 1, we also normalize the performance function with the maximum achievable value without the attack so that they are comparable. The values are shown in Fig. 4 (c) with different settings of α and β.

With the weight of security $\beta \neq 0$, the performance-security function for $\mathbf{S}_1$ increases with number of observable events, and exceeds that of strategy $\mathbf{S}_2$ at some point. The explanation is that, with more and more evidences accumulated, the detection accuracy increases, and thus the strategy of overlapping transmission becomes more and more convenient. This implies that, when it is unclear if an attack is taking place, a defender can either choose to stay on the channel with the risk of harming other legitimate transmissions, while accumulating more evidence, or migrate to another channel.

5 Use-case example: Hammer-and-anvil

In this section, we consider another specific cross-layer attack, called the hammer-and-anvil attack [15]. Although the attack has been briefly discussed here, due to space limitations the reader can refer to [15] for details.

5.1 Brief Introduction

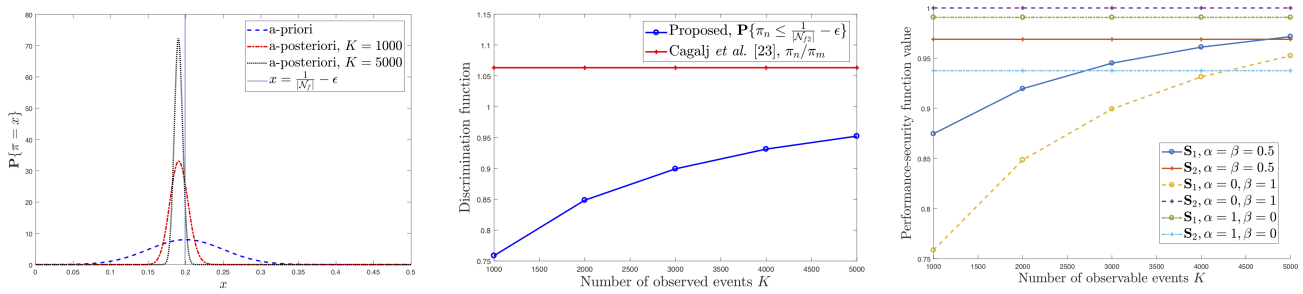
As shown in Fig. 1, the considered wireless multi-hop network consists of a set $\mathcal{N}$ of nodes. Data is generated at source nodes and forwarded to the sink $z \in \mathcal{N}$ in a hop-by-hop manner. A node $n \in \mathcal{N}$ must decide its power allocation $\mathbf{P}_n = \{P_n^1, \dots, P_n^{|\mathcal{F}|}\}$ on the set $\mathcal{F}$ of channels, as well as the next hop $\hat{m}_n$. Therefore, its strategy can be written as $\mathbf{S}_n = \{\mathbf{P}_n, \hat{m}_n\}$.

The objective of node $n \in \mathcal{N}$ is to minimize the expected end-to-end delay from itself to the sink, represented as $T_n(\mathbf{P}_n, \hat{m}_n)$. To this end, a joint optimization of $\mathbf{P}_n$ on physical layer, and $\hat{m}_n$ on network layer is performed, with the optimal strategy

$$\{\mathbf{P}_n^*, \hat{m}_n^*\} = \arg \max_{\mathbf{P}_n \in \mathcal{P}_n, \hat{m} \in \mathcal{V}_n} -T_n(\mathbf{P}_n, \hat{m}_n), \quad (23)$$

where $\mathcal{P}_n$ and $\mathcal{V}_n$ are the set of possible power allocation, and the neighbor set of n, respectively.

As a remark, the queuing model in [15] states that the average delay of link $n \to \hat{m}_n$ is a decreasing function of



(a) (b) (c)

Fig. 4. MAC poisoning attack: (a) *a priori* and *a posteriori* distributions of the channel access probability for f_1, ground truth: $\pi = 0.1875$; (b) discrimination function output with different numbers of observed events; (c) performance-security function values for the two strategies with different α and β.

the achievable throughput of the link, which is expressed as

$$\mu_n(\hat{m}_n) = \sum_{f \in \mathcal{F}} \log_2 \left(1 + \frac{P_n^f H_n^f(\hat{m}_n)}{I_n^f(\hat{m}_n) + \eta_n^f(\hat{m}_n)}\right), \quad (24)$$

with $H_n^f(\hat{m}_n)$, $I_n^f(\hat{m}_n)$, and $\eta_n^f(\hat{m}_n)$ denoting the channel coefficient of link $n \to \hat{m}_n$, the interference and noise at $\hat{m}_n$, respectively.

Equation (24) establishes the ground for hammer-and-anvil attack to work. The attacker is made up by a jammer j and a compromised node $c \in \mathcal{N}$. The compromised node can undermine communication in various ways. For the sake of generality, we assume the activities are not observable by an outsider, meaning there is no way to distinguish it from legitimate nodes.

The jammer aims to redirect traffic to the compromised node. To this end, it selectively jams links that do not lead to the compromised node. For example, if j jams the link $n \to m$, the achievable throughput $\mu_n(m)$ will be degraded, resulting an increased $T_n(m)$. With appropriate jamming, the utility is degraded so that, for $\forall \mathbf{P}_n \in \mathcal{P}_n$, $-T_n(\mathbf{P}_n, m) < -T_n(\mathbf{P}'_n, l)$, for l and some $\mathbf{P}'_n$. Then n will choose l instead of m as the next hop.

The strategy of the attacker is simply the power allocation of the jammer, $\mathbf{P}_j$, and its attack gain is the input data rate of the compromised node.

5.2 Attack Detection

It is shown in [15] that the attack objective can be achieved by small-scale jamming. Therefore, it is difficult to accurately detect the attack using traditional jamming detection methods based on packet delivery ratio (PDR). We will adopt the detection method in the framework to address this challenge.

In the following, we will focus on link $n \to m$, and thus eliminate the subscripts n and m. Moreover, since all the channels are i.i.d., we will also eliminate f. We still retain the subscript j to denote the power and channel coefficients involving the jammer j. Therefore, we use P, P_j to denote the power allocation of n and j on f. Furthermore, H, H_j represent the channel coefficients from n and j to m, on channel f, respectively.

To detect the attack, we focus on the network state of the interference $I = P_j H_j$, as it is the direct variable affected by jamming. For the observable events, we choose the reception of bits at the receiver, since it reveals the interference. We denote an event as e_k and formally define it as

$$e_k = \begin{cases} 1, & \text{bit } k \text{ is received correctly,} \\ 0, & \text{bit } k \text{ is not received correctly.} \end{cases} \quad (25)$$

The conditional probability for an event e_k given an interference value, shown as

$$\mathbb{P}\{e_k | I = i\} = \mathbb{P}\{e_k | \gamma = \frac{PH}{i + \eta}\}, \quad (26)$$

is essentially the bit error probability given the SINR γ, with Gaussian noise η. (26) is readily available for additive white Gaussian noise (AWGN) channels, and can be obtained through training in other environments.

Let us assume a sequence of K bits is transmitted during a strategy updating period. Therefore we have a sequence of events $\{e_k\}_{k=1,\ldots,K}$ and

$$\mathbb{P}\{\{e_k\}_{k=1,\ldots,K} | I = i\} = \prod_{k=1}^{K} \mathbb{P}\{e_k | I = i\}. \quad (27)$$

Thus, the *a posteriori* distribution of I can be estimated as

$$\mathbb{P}\{I = i | \{e_k\}_{k=1,\ldots,K}\} = \frac{\prod_{k=1}^{K} \mathbb{P}\{e_k | I = i\}}{\int_j \prod_{k=1}^{K} \mathbb{P}\{e_k | I = j\} \mathbb{P}\{I = j\} \mathbf{d}j} \cdot \mathbb{P}\{I = i\}. \quad (28)$$

The expression in (28) provides a procedure to estimate the interference of link $n \to m$ on channel f. Starting with some *a priori* distribution $\mathbb{P}\{I = i\}$, once a sequence of bits have been received (correctly or incorrectly) on channel f, the receiver computes the *a posteriori* distribution according to the corresponding likelihood for events $\{e_k\}_{k=1,\ldots,K}$. This is done recursively for every transmission. For practical reasons, the integration in (28) is approximated by a summation over a finite set $\mathcal{I}$. Then, it can be rewritten as

$$\mathbb{P}\{I = i | \{e_k\}_{k=1,\ldots,K}\} = \frac{\prod_{k=1}^{K} \mathbb{P}\{e_k | I = i\}}{\sum_{j \in \mathcal{I}} \prod_{k=1}^{K} \mathbb{P}\{e_k | I = j\} \mathbb{P}\{I = j\}} \cdot \mathbb{P}\{I = i\}. \quad (29)$$

When the number of channels is large, it may not be practical to compute (29) for each channel in every transmission. In this case, the receiver can select some channel(s) from $\mathcal{F}$ to update.

For attack mapping, without observable behavior of the compromised node, the most obvious feature is the small jamming scale. We design the classifier as

$$\mathbb{P}\{I_{\text{lwr}} \leq I \leq I_{\text{upp}}\} \underset{\text{Attacked}}{\overset{\text{Not Attacked}}{\lessgtr}} \mathbb{P}_{\text{th}}. \quad (30)$$

If the *a posteriori* probability that the interference is between a range is higher than a probability threshold, the link is considered to be jammed. The thresholds I_{lwr}, I_{upp}, and $\mathbb{P}_{\text{th}}$ correspond to the feature B in (5). The network management entity can adjust these thresholds according

to its information on whether the attack exists, how far the jammer is likely to be, and so forth. The resulting jamming detection scheme is described in Algorithm 3.

Algorithm 3 Bayesian learning-based jamming detection

Given node $m \in \mathcal{N}$ and *a priori* distribution $\mathbb{P}\{I_m^f = i\}$;
while true do
  if A node $n \in \mathcal{N}$ selects m as the receiver and wins the channel competition **then**
    n and m conduct channel estimation and get $H_{nm}^f, \forall f \in \mathcal{F}$;
    Based on H_{nm}^f, n decides the strategy and transmits;
    m selects a $\mathcal{F}' = \{f \subset \mathcal{F} : P_n^f \neq 0\}$;
    for $f \in \mathcal{F}'$ **do**
      Compute the likelihood in (27) based on the bit reception events;
      Update the *a posteriori* probability in (29);
    end for
  end if
  if *a posteriori* distribution satisfies convergence condition; **then**
    Break;
  end if
end while
if (30) holds for a certain number of $f \in \mathcal{F}$ **then**
  Node m is considered jammed;
else
  Node m is considered not jammed;
end if

5.3 Attack Mitigation

After the attack is detected, we further apply the proposed framework to generate a mitigation scheme. Therefore, we define a performance-security function as in (6) and optimize it in the fashion of (7) - (9).

The performance component is simply the negative of delay $-T_n$. The security component is the negative of the attack gain, *i.e.*, the amount of data that is routed to the compromised node c. However, this value may not be accurately measured, since generally the location of the compromised node is not detectable. In fact, a compromised node is detectable only if it performs attacks with observable activities, such as packet dropping for blackhole attack. For the sake of generality, we do not assume such behaviors, and rely solely on the detection results of small-scale jamming for security level estimation. Specifically, we approximate the risk, *i.e.*, the negative of security, for a strategy with the likelihood that it leads to the compromised node. Since it is expected that the jammer "drives" data to the compromised node, it is reasonable to assume that the compromised node is on an alternative route. Therefore, denoting the risk as R_n, we set it to 1 for the strategy involving rerouting, and a small value $\epsilon > 0$ for the strategy staying on the currently jammed route.[2] The optimization problem can then be formulated as

$$\text{maximize} \qquad -\alpha T_n - \beta R_n. \tag{31}$$

To elaborate the possible defense mechanisms, we assume that l is the best next hop candidate for n with jamming, while the jammed node m is the second best one, in the following discussion.

2. The latter is not set to 0 because there is a chance that a downstream node of the currently jammed route reroutes to the compromised nodes.

Two simple strategies for n are (i) to reroute to l or (ii) to keep routing to m. We denote these strategies as $\mathbf{S}_l$ and $\mathbf{S}_m$, respectively. The delay for the two strategies are then $T_n(\mathbf{S}_l) = T_n(l)$ and $T_n(\mathbf{S}_m) = T_n(m)$, respectively. We set $R_n(\mathbf{S}_l) = 1$ and $R_n(\mathbf{S}_m) = \epsilon$, with a small $\epsilon \in (0, 1]$ to represent the normalized security level. Since $R_n(\mathbf{S}_l) >> R_n(\mathbf{S}_m)$, a desirable compromise between performance and security may not be available with these two strategies alone.

A more sophisticated strategy that may enable a better compromise can be designed based on a secure network coding strategy [41], which we denote as $\mathbf{S}_{\text{SNC}}$. We construct the secure network coding scheme as follows:

1) Choose a suitable integer r and get the message vector X from $GF^{(r)}(q)$, for some q;
2) Choose a suitable r-dimensional linear network code E on $GF^{(r)}(q)$;
3) Encode the vector X by multiplying it with the encoding matrix, and get the encoded vector $Y = EX$;
4) Among the r elements of the encoded vector Y, the first N_l are transmitted through l, and the remaining N_m are transmitted through m, ($N_l + N_m = r$).

GF stands for Galois Field. According to Theorem 2 in [41], there exists a code E guaranteeing that neither l nor m can decode the messages if $\dim(E) = r$ and $\max(\dim(V_l), \dim(V_m)) < r$, with V_l and V_m being the linear spans of the encoding vectors corresponding to l and m, respectively. The symbol $\dim$ stands for dimension.

Since N_l and N_m out of $N_l + N_m$ encoded messages are transmitted through l and m respectively, the achievable performance for this strategy is

$$T_n(\mathbf{S}_{\text{SNC}}) = \frac{N_l T_n(l) + N_m T_n(m)}{N_l + N_m} + \frac{1}{\lambda_n}(N_l + N_m - 1). \tag{32}$$

The second term on the RHS represents the additional delay introduced by waiting for all the $N_l + N_m$ messages to arrive to the destination before encoding. Since it is guaranteed that neither m nor l can decode the message, the risk is $R_n(\mathbf{S}_{\text{SNC}}) = 0$.

To sum up, node n chooses a strategy among (i) reroute to l, (ii) keep sending through m, and (iii) use secure network coding, according to the performance risk values for each of them. The corresponding performance-security functions are

$$h_n(\mathbf{S}_l) = -\alpha \hat{T}_n(\mathbf{S}_l) - \beta, \tag{33}$$

$$h_n(\mathbf{S}_m) = -\alpha \hat{T}_n(\mathbf{S}_m) - \beta\epsilon, \tag{34}$$

$$h_n(\mathbf{S}_{\text{SNC}}) = -\alpha \hat{T}_n(\mathbf{S}_{\text{SNC}}). \tag{35}$$

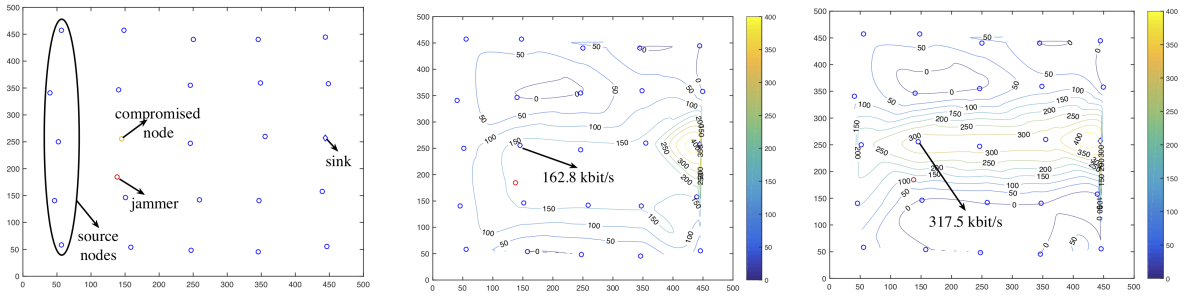
Note that the performance is normalized to be comparable to the risk, which takes values in $[0, 1]$. Then, the problem (which can be solved using Algorithm 4) is

$$\underset{\mathbf{S} \in \{\mathbf{S}_l, \mathbf{S}_m, \mathbf{S}_{\text{SNC}}\}}{\text{maximize}} \; h_n(\mathbf{S}). \tag{36}$$

5.4 Performance Evaluation

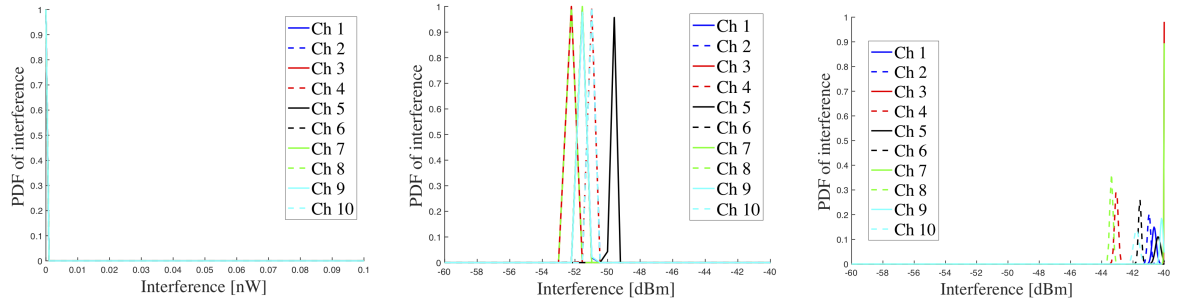
We now evaluate the performance of the FORMAT framework when tackling the hammer-and-anvil attack through both simulations and testbed experiments.

Simulation settings. A network of 25 nodes is randomly generated in a $500\ \text{m} \times 500\ \text{m}$ area, with one sink and one compromised node. There is a jammer whose location is set in accordance to the location of the compromised node. A



(a) (b) (c)

Fig. 5. Example scenario: (a) topology; (b) traffic map without jamming; (c) traffic map with jamming, $P_j^{\max} = 10$ mW. Traffic in (b) and (c) are shown in contour maps, with the lines representing contours for the traversing data rate. The numbers show the data rate in $kbits/s$. Values for points without a node are interpolated for visual purpose.



(a) (b) (c)

Fig. 6. Detection results for small-scale jamming: (a) $P_j^{\max} = 0$; (b) $P_j^{\max} = 10\,\mathrm{mW}$; (c) $P_j^{\max} = 100\,\mathrm{mW}$.

Algorithm 4 Secure network coding based attack mitigation

Given a jammed link $n \to m$:
n finds the best next hop candidate $l \in \mathcal{V}_n$, and the optimal secure network code (N_l, N_m);
if $(35) \geq \max[(33), (34)]$ **then**
  proceed with the secure network coding scheme;
else if $(34) \geq \max[(33), (35)]$ **then**
  Transmit through m;
else
  Transmit through l;
end if

typical example is shown in Fig. 5(a). For convenience, we label the nodes with numbers.

Data sessions are generated at the leftmost 5 nodes. The generation rates are randomly set, with a mean value of 80 kbits/s. Each legitimate node has power budget $P^{\max} = 1$ W. There are 10 mutually orthogonal channels, with bandwidth of 10 kHz each. All channels have a path loss with exponent 3 and i.i.d. Rayleigh fading with parameter 0.5. The spectral density of noise is 1×10^{-8} W/Hz. The power budget of the jammer is set to $10 - 100$ mW, *i.e.*, only $1/100 - 1/10$ as high as that of the legitimate nodes. Even with this level of jamming power, the attack is effective. This can be verified in 5 (b) and (c), which show traffic maps corresponding to the example topology. Considering the simulation settings, even with a jammer that only causes interference around 10 times the noise level, a significant amount of data can be "driven" to the compromised node. Therefore, the efficiency and stealthiness is verified.

Detection results. Since we focus on small-scale jamming, the considered interference range for the distribution $\mathcal{I}$ is set to $[0, 1 \times 10^{-7}]$ W, with a step size of 1×10^{-9} W. The proposed jamming detection scheme can effectively detect the interference caused by small-scale jamming attack. For node 7, which is the target node of the jammer, the *a posteriori* distribution of I is shown in Fig. 6, for $P_j^{\max} = 0$, $P_j^{\max} = 10$mW, and $P_j^{\max} = 100$mW. We assume uniform *a priori* distribution $\mathbb{P}\{I = i\}$. We observe that the interference can be effectively and accurately estimated in all cases. For $P_j^{\max} = 0$, which means no jamming attack, the distribution converges to a single peak at $I = 0$ for all channels, and provides a perfect estimation. For $P_j^{\max} = 10$ mW, the *a posteriori* distribution of I concentrates at around -51 dBm. Considering that the distance from the jammer to node 7 is 40.2 m, the average jamming-caused interference is -50.5 dBm. Therefore, the result is accurate. For $P_j^{\max} = 100$ mW, the *a posteriori* distribution converges to the largest possible value in $\mathcal{I}$, *i.e.*, $I = -40$ dBm, since in this case the jammer creates relatively large interference.

To provide a better understanding, we list the corresponding bit error rates (BER) in Table 3.[3] We observe that the difference in BER with or without jamming is small. This implies that a detection scheme based on BER only may not work well. Therefore, the proposed detection scheme can better discriminate between jammed and unjammed scenarios with small-scale jamming. This is confirmed by

3. It is couter-intuitive to observe a BER for $P_{\mathrm{j}}^{\max} = 100\,\mathrm{mW}$ slightly smaller than that for $P_{\mathrm{j}}^{\max} = 10$ mW. This is due to the fact that we have considered a scenario with high mutual interference between different nodes, and the overall interference includes both the interference caused by jamming and that caused by other nodes. According to [15], with the hammer and anvil attack, the traffic in the network is re-organized, with a portion rerouted to faraway nodes. With a higher jamming power, more traffic is rerouted, and the interference from other nodes is lowered. Therefore, a similar SINR (and consequently a similar BER) is resulted. With the high mutual interference setting, and the fact that jamming only contributes a portion of the overall interference, Table. 3 still reveals small-scale jamming.

TABLE 3
BER of node 7 for different jamming power

Jamming power budget $P_j^{\max}$ (mW)	0	10	100
Bit error rate (%)	3.84	9.84	9.78

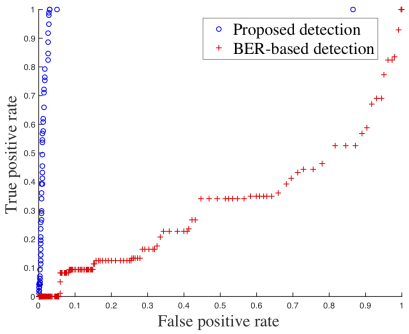


Fig. 7. ROC curves of the proposed detection and BER-based detection methods.

the receiver operating characteristic (ROC) curves shown in Fig. 7.

To obtain the ROC curve, we ran simulations for different topologies and varied the jamming power budget between 10 mW and 100 mW with a step of 10 mW. For each simulation, the learned distribution of interference is fed to the classifier defined in (30). We point out that, there is not yet a detection method for the considered attack to compare with. We try to compare the ability of attack activity detection, *i.e.*, jamming detection, with traditional method. Therefore, we set the upper bound I_{upp} to infinity for fairness. True positive rates (TPR) and false positive rates (FPR) are recorded for the threshold $I_{\text{lwr}} \in [0, 1\times10^{-4}]$mW, with a step of 1×10^{-6} mW. For comparison, we also plot the ROC curve for BER-based jamming detection, with the classifier

$$\text{BER} \underset{\text{jammed}}{\overset{\text{unjammed}}{\lessgtr}} \text{BER}_{\text{th}}, \tag{37}$$

and the threshold varying from 0 to 0.5, with a step of 0.001. Comparing the two ROC curves, we find that the proposed detection scheme results in a large area under the curve (AUC), and can thus achieve very reliable detection results in most cases. Conversely, the BER-based detection method has much smaller AUC, and thus is inferior to the proposed method.

Mitigation results. To demonstrate the performance of the mitigation scheme, we used traces from the previous simulations. We identified the jammed link in each topology, *i.e.*, $n \to m$, and found the best next hop candidate l for n. For the strategies of rerouting to l, staying on m, and using secure network coding, we computed the performance-risk functions defined in (33), (34) and (35), for different values of (α, β). The results are shown in Fig. 8.

For $\alpha = 1, \beta = 0$, *i.e.*, when performance is the only concern, secure network coding is not as good as using the best unjammed link. However, it still achieves considerable gain compared to the jammed link. When risk is taken into account, the benefits of secure network coding become obvious. For $\alpha = \beta = 0.5$, *i.e.*, when performance and risk are equally important, secure network coding outperforms the other two since it provides a better performance-risk balance. For $\alpha = 0, \beta = 1$, *i.e.*, when the only concern is risk, the relationship between the three strategies is the same as in the previous case, but the benefits of secure network coding compared to using the jammed link become marginal, since it is unlikely that a link leading to a compromised node is jammed.

TABLE 4
Detection result and corresponding relative BER

Detection Result	No	Yes	Yes
Relative BER	0.27%	1.28%	3.92%

5.5 Testbed Evaluation

In real scenarios, there may not be a closed-form relationship between BER and SINR, *i. e.*, the likelihood for an event to happen, shown in (2), may not be readily available. In this case, a training process is required. We conducted testbed experiments where we first trained a BER curve from the transmission results without jamming; then, based on the curve, we ran the jamming detection scheme with reactive jamming to show its accuracy. We used two USRP N210 [20] to form a transmitter-receiver pair, and another USRP X310 [20] is used as the jammer, emitting interference. We used a channel with bandwidth of 10 kHz.

In the training period, the jammer constantly emits interference with varying power. Meanwhile, the transmitter-receiver pair conducts normal transmissions. The receiver estimates the SNR and calculates the corresponding BER by comparing the received file with the original one. Multiple points are obtained, based on which a BER-SINR curve is generated by using exponential fitting. The curve is shown in Fig. 9.

To test the accuracy of the proposed detection, we run more experiments with the reactive jamming model. The receiver is able to estimate the SINR without jamming. Along with the real bit reception events, the distribution of interference can be learned. We use the interference-to-noise ratio (INR) instead of absolute interference level to reduce the error. In Fig. 10, we show the resulting PDF for different jamming scenarios. Since the real interference level at the receiver is unknown, we use the BER to distinguish the scenarios. Note that there is an irreducible BER of 1.15% caused by the multipath effect, representing the lowest BER the transmitter-receiver pair is able to achieve.

Since our objective is to detect small-scale jamming, we set $\text{INR}_{\text{th}} = 0$ dB, and used $Pb(\text{INR} > \text{INR}_{\text{th}}) > 0.9$ as the classifier. If the interference is at least at the same level of noise with probability 0.9, then it is considered to be jammed. With this classifier, the detection results and the relative BER (real BER - irreducible BER) are shown in Table 4. The proposed scheme is able to detect small scale jamming resulting in very low relative BER, as long as it is greater than 1%, which verifies the effectiveness of the detection scheme.

6 Use-Case Example: TCP Timeout

In this section, we show how the framework can be applied to TCP timeout attacks. We will focus on the detection. For this specific attack, the security performance tradeoff is not obvious, and it can be mitigated by simply dropping the packets of the detected DoS flows.

We consider a router n with multiple TCP flows aggregated. There may be DoS flows among them, and the

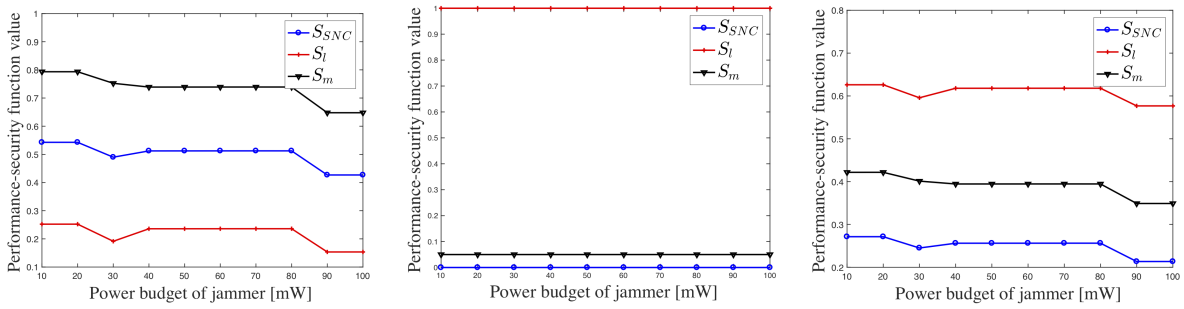


Fig. 8. Optimal performance-security tradeoff with different settings: (a) $\alpha = 1, \beta = 0$; (b) $\alpha = 0, \beta = 1$; (c) $\alpha = 0.5, \beta = 0.5$.

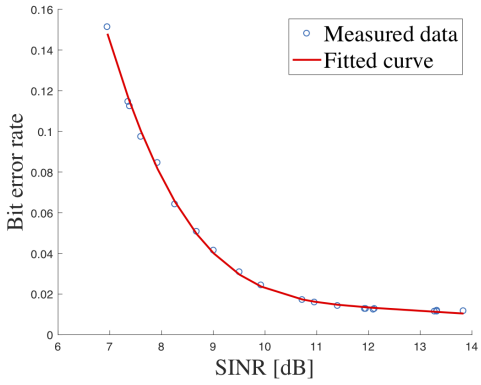


Fig. 9. Trained BER curve.

objective is to detect if one of the flows is a DoS flow performing TCP timeout attack. As is shown in [21], an important characteristic of the DoS flow is the periodic arrival process. Therefore, the burst inter-arrival time reveals if the attack is taking place, and thus can be used as the observations. We will denote the inter-arrival time between the k-th burst and $(k+1)$-th burst as Δ_k.

A normal flow and a DoS flow differ on the arrival process, which composes the network state needed for the detection of this attack. According to [21], to successfully launch the attack, a DoS flow should have a period T at the scale of $minRTO$, *i.e.*, the minimum retransmission timeout value. Moreover, the normalized thoughput of the aggregated TCP flows with a DoS flow of period T is given by

$$\rho(T) = \frac{\lceil \frac{minRTO}{T} \rceil \cdot T - minRTO}{\lceil \frac{minRTO}{T} \rceil \cdot T}. \tag{38}$$

Therefore, it is reasonable to assume the inter-arrival time of a malicious flow is a fixed value $T = minRTO$, *i.e.*, for a DoS flow, the likelihood to observe inter-arrival time $T = t$ is:

$$\mathbf{P}\{T = t\} = \begin{cases} 1, & t = minRTO, \\ 0, & otherwise. \end{cases} \tag{39}$$

According to [42], the interarrival time of a normal TCP flow fits Weibull distribution, *i. e.*,

$$\mathbf{P}\{T = t\} = \frac{1}{a}\left(-\frac{t}{a}\right)^{c-1} e^{-\left(\frac{t}{a}\right)^c}. \tag{40}$$

Note the shape parameter c varies with application type and operating time. In other words, different c values represent different types of normal TCP flows. Therefore, we can model the parameter as a random variable C and use it directly as the network state to be detected. To cover the case of DoS flow, we let $C = 0$ represent the arrival process with period $T = minRTO$. Formally, $C = c$ represents

$$c = \begin{cases} 0, & \text{DoS TCP flow}, \\ \text{other}, & \text{normal TCP flow with parameter } c, \end{cases} \tag{41}$$

For each flow, with a-priori distribution of C, we can update the a-posteriori with the observation of a series of inter-arrival time $\{\Delta_k\}_{k=1,\ldots,K} = \{\delta_1, \ldots, \delta_K\}$, following

$$\begin{aligned} &\mathbf{P}\{C = c | \{\Delta_k\}_{1,\ldots,K} = \{\delta_k\}_{1,\ldots,K}\} \\ &= \frac{\prod_{k=1}^{K} \mathbf{P}\{\Delta_k = \delta_k | C = c\} \mathbf{P}\{C = c\}}{\int_t \prod_{k=1}^{K} \mathbf{P}\{\Delta_k = \delta_k | C = c\} \mathbf{P}\{C = c\}}. \end{aligned} \tag{42}$$

The a-priori distribution $\mathbf{P}\{C = c\}$ can be acquired by sampling the interarrival time for normal TCP flows in the network, with the probability for $C = 0$ given by the belief that an attacker exists. In the simulations, we borrow the a-priori distribution for $C > 0$ directly from [42], which represents the histogram of AT&T WorldNet modem users during a 12 minutes period. The distribution is normalized by 0.95, *i. e.*, the a-prori belief on the existence of a DoS flow is set to 0.05. The distribution is shown in Fig. 11 (a).

We run simulations for both normal and DoS TCP flows. Normal TCP flows are drawn with Weibull distribution with $c = 0.68$, and DoS flows are generated with an interval of $minRTO = 1$ s.[4] Since there are always small discrepancies between the real distribution and theoretical distribution, and the attacker may not use a period strictly equal to $minRTO$, we intentionally introduce a Guassian random error with zero mean and a standard deviation of 100 ms. The a-posteriori distributions of C for normal flow and DoS flow are shown in Fig. 11 (b) & (c). Clearly, the detection scheme is able to distinguish normal flows ($c > 0$) and DoS flows ($c = 0$). Interestingly, according to [21], the traditional detection methods do not perform well for the attack with a period exactly equals to $minRTO$, therefore, we can verify the advantage of the proposed detection method.

7 Conclusions

In this paper, we have proposed FORMAT, a framework to detect and mitigate cross-layer attacks in wireless networks based on Bayesian learning. The learning-based detection is able to utilize observations of multiple layers to generate a hypothesis on whether a certain attack is taking place, while the mitigation scheme uses optimization to achieve the desired security-performance trade-off under

4. It is suggested that $minRTO = 1$ s, as argued in [7]

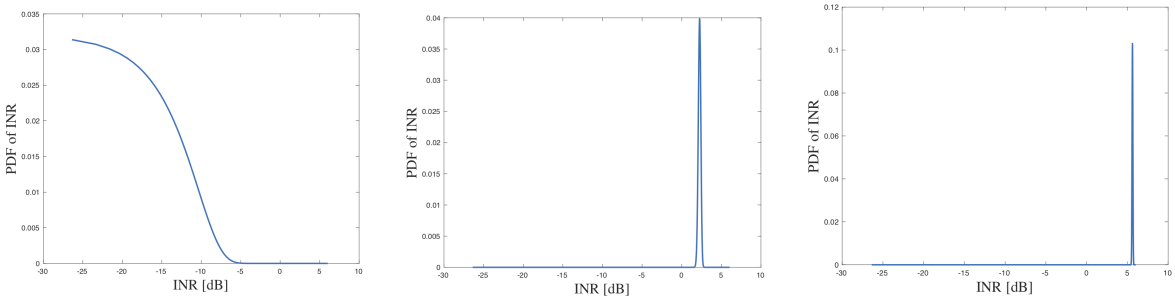


Fig. 10. Detection results for small-scale jamming, with real BER: (a) 1.42%; (b) 2.43%; (c) 5.07%.

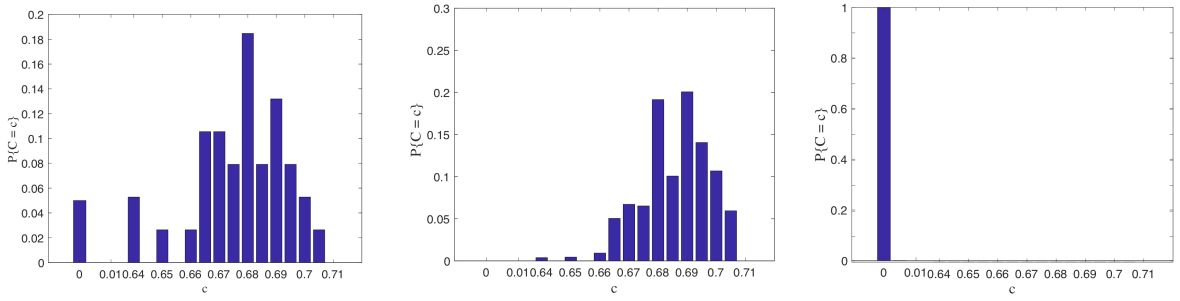


Fig. 11. Distribution of C: (a) a-priori; (b) a-posteriori, normal TCP flow; (c) a-posteriori, DoS TCP flow.

different requirements. We extensively evaluated the FORMAT framework by considering two state-of-the-art cross-layer attacks. Both simulation and testbed results conclude that the FORMAT achieves superior detection scheme to traditional single-layer approaches and thus tackles cross-layer attacks effectively.

ACKNOWLEDGEMENT

We thank the anonymous reviewers for their comments, which have helped us improve the quality of our manuscript significantly. This material is based upon work funded by the Air Force Research Laboratory under Contract No. FA8750-15-3-6001-NU and by National Science Foundation under Grant CNS-1618727.

REFERENCES

[1] Y. Zou, J. Zhu, X. Wang, and L. Hanzo, "A Survey on Wireless Security: Technical Challenges, Recent Advances, and Future Trends," *Proceedings of the IEEE*, vol. 104, pp. 1727–1765, Sep. 2016.

[2] T. Jin, G. Noubir, and B. Thapa, "Zero Pre-shared Secret Key Establishment in the Presence of Jammers," in *Proc. of ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, New Orleans, LA, USA, May. 2009, pp. 219–228.

[3] C. Popper, M. Strasser, and S. Capkun, "Anti-Jamming Broadcast Communication Using Uncoordinated Spread Spectrum Techniques," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 28, pp. 703–715, Jun. 2010.

[4] Y. Liu, P. Ning, H. Dai, and A. Liu, "Randomized Differential DSSS: Jamming-Resistant Wireless Broadcast Communication," in *Proc. of IEEE International Conference on Computer Communications (INFOCOM)*, San Diego, CA, USA, Mar. 2010, pp. 1–9.

[5] A. Cassola, T. Jin, G. Noubir, and B. Thapa, "Efficient Spread Spectrum Communication without Preshared Secrets," *IEEE Transactions on Mobile Computing*, vol. 12, no. 8, pp. 1669–1680, Aug. 2013.

[6] A. Rajandekar and B. Sikdar, "A Survey of MAC Layer Issues and Protocols for Machine-to-Machine Communications," *IEEE Internet of Things Journal*, vol. 2, no. 2, pp. 175–186, 2015.

[7] A. Mukherjee, S. A. A. Fakoorian, J. Huang, and A. L. Swindlehurst, "Principles of Physical Layer Security in Multiuser Wireless Networks: A Survey," *IEEE Communications Surveys Tutorials*, vol. 16, no. 3, pp. 1550–1573, Third 2014.

[8] G. Thamilarasu, A. Balasubramanian, S. Mishra, and R. Sridhar, "A Cross-Layer Based Intrusion Detection Approach for Wireless Ad Hoc Networks," in *Proc. of IEEE International Conference on Mobile Adhoc and Sensor Systems*, Washington, DC, USA, Nov. 2005.

[9] M. Shao, S. Zhu, G. Cao, T. L. Porta, and P. Mohapatra, "A Cross-Layer Dropping Attack in Video Streaming over Ad Hoc Networks," in *Proc. of International Conference on Security and Privacy in Communication Netowrks (SecureComm)*, Istanbul, Turkey, Sep. 2008, pp. 25:1–25:8.

[10] V. Toubiana and H. Labiod, "A Cross Layer Attack Against MANET Cooperation Enforcement Tools," in *IEEE International Conference on Networks*, Dec. 2008, pp. 1–5.

[11] S. Djahel, F. Nat-Abdesselam, and A. Khokhar, "A Cross Layer Framework to Mitigate a Joint MAC and Routing Attack in Multihop Wireless Networks," in *IEEE Conference on Local Computer Networks*, Zurich, Switzerland, Oct 2009, pp. 730–737.

[12] O. León, J. Hernandez-Serrano, and M. Soriano, "A New Cross-Layer Attack to TCP in Cognitive Radio Networks," in *Proc. of International Workshop on Cross Layer Design (IWCLD)*, Palma de Mallorca, Spain, Jun. 2009, pp. 1–5.

[13] W. Wang, Y. Sun, H. Li, and Z. Han, "Cross-Layer Attack and Defense in Cognitive Radio Networks," in *Proc. of IEEE Global Telecommunications Conference (GLOBECOM)*, Miami, FL, USA, Dec. 2010, pp. 1–6.

[14] J. Hernandez-Serrano, O. León, and M. Soriano, "Modeling the Lion Attack in Cognitive Radio Networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2011, pp. 2:1–2:10, Jan. 2011.

[15] L. Zhang and T. Melodia, "Hammer and Anvil: The Threat of A Cross-Layer Jamming-aided Data Control Attack in Multihop Wireless Networks," in *Proc. of IEEE Conference on Communications and Network Security (CNS)*, Florence, Italy, Sep. 2015, pp. 361–369.

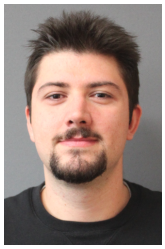
[16] K. Hasan, S. Shetty, and T. Oyedare, "Cross Layer Attacks on GSM Mobile Networks Using Software Defined Radios," in *IEEE Annual Consumer Communications Networking Conference (CCNC)*, Las Vegas, NV, USA, Jan 2017, pp. 357–360.

[17] X. Lin, N. B. Shroff, and R. Srikant, "A Tutorial on Cross-Layer Optimization in Wireless Networks," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 24, pp. 1452–1463, Aug. 2006.

[18] L. Ding, T. Melodia, S. Batalama, J. Matyjas, and M. Medley, "Cross-Layer Routing and Dynamic Spectrum Allocation in Cog-

nitive Radio Ad Hoc Networks," *IEEE Transactions on Vehicular Technology*, vol. 59, pp. 1969–1979, May. 2010.
[19] S. Pudlewski, N. Cen, Z. Guan, and T. Melodia, "Video Transmission over Lossy Wireless Networks: A Cross-Layer Perspective," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, pp. 6–22, Feb. 2015.
[20] Ettus Research, "Ettus Research Products," https://www.ettus.com, 2017.
[21] A. Kuzmanovic and E. W. Knightly, "Low-Rate TCP-Targeted Denial of Service Attacks and Counter Strategies," *IEEE/ACM Transactions on Networking*, vol. 14, no. 4, pp. 683–696, Aug. 2006.
[22] K. Bian, J.-M. Park, and R. Chen, "NIS01-6: Stasis Trap: Cross-Layer Stealthy Attacks in Wireless Ad Hoc Networks," in *Proc. of IEEE Global Telecommunications Conference (GLOBECOM)*, San Francisco, CA, USA, Nov. 2006, pp. 1–5.
[23] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2016.
[24] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2000.
[25] F. Restuccia, S. D'Oro, and T. Melodia, "Securing the internet of things in the age of machine learning and software-defined networking," *IEEE Internet of Things Journal*, pp. 1–1, 2018.
[26] F. Restuccia, N. Ghosh, S. Bhattacharjee, S. K. Das, and T. Melodia, "Quality of information in mobile crowdsensing: Survey and research challenges," *ACM Transactions on Sensor Networks (TOSN)*, vol. 13, no. 4, p. 34, 2017.
[27] Y. Liu, Y. Li, and H. Man, "Short Paper: A Distributed Cross-Layer Intrusion Detection System for Ad Hoc Networks," in *International Conference on Security and Privacy for Emerging Areas in Communications Networks (SECURECOMM)*, Sep. 2005, pp. 418–420.
[28] X. Liu, G. Noubir, R. Sundaram, and S. Tan, "SPREAD: Foiling Smart Jammers Using Multi-Layer Agility," in *IEEE International Conference on Computer Communications (INFOCOM)*, Anchorage, AK, USA, May. 2007, pp. 2536–2540.
[29] X. Wang, J. S. Wong, F. Stanley, and S. Basu, "Cross-Layer Based Anomaly Detection in Wireless Mesh Networks," in *Proc. of International Symposium on Applications and the Internet*, Bellevue, WA, USA, Jul. 2009, pp. 9–15.
[30] R. Shrestha, K.-H. Han, D.-Y. Choi, and S.-J. Han, "A Novel Cross Layer Intrusion Detection System in MANET," in *IEEE International Conference on Advanced Information Networking and Applications*, Perth, Australia, Apr. 2010, pp. 647–654.
[31] R. Song, H. Tang, P. C. Mason, and Z. Wei, "Cross-Layer Security Management Framework for Mobile Tactical Networks," in *Proc. of IEEE Military Communications Conference (MILCOM)*, San Diego, CA, USA, Nov. 2013, pp. 220–225.
[32] K. C. Nguyen, T. Alpcan, and T. Başar, "A Decentralized Bayesian Attack Detection Algorithm for Network Security," in *Proc. of the IFIP International Information Security Conference*, Boston, MA, USA, 2008, pp. 413–428.
[33] N. Forti, G. Battistelli, L. Chisci, and B. Sinopoli, "A Bayesian Approach to Joint Attack Detection and Resilient State Estimation," in *Proc. of the IEEE Conference on Decision and Control (CDC)*, Las Vegas, NV, USA, Dec. 2016, pp. 1192–1198.
[34] R. F. Fouladi, C. E. Kayatas, and E. Anarim, "Frequency Based DDoS Attack Detection Approach Using Naive Bayes Classification," in *Proc. of the International Conference on Telecommunications and Signal Processing (TSP)*, Vienna, Austria, Jun. 2016, pp. 104–107.
[35] A. P. Dempster, "A Generalization of Bayesian Inference," *Classic Works of the Dempster-Shafer Theory of Belief Functions*, vol. 219, pp. 73–104, 2008.
[36] R. H. Gohary, Y. Huang, Z.-Q. Luo, and J.-S. Pang, "A Generalized Iterative Water-Filling Algorithm for Distributed Power Control in the Presence of a Jammer," *IEEE Transactions on Signal Processing*, vol. 57, no. 7, pp. 2660–2674, Jul. 2009.
[37] B. Wang, Y. Wu, K. J. R. Liu, and T. C. Clancy, "An anti-jamming stochastic game for cognitive radio networks," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 4, pp. 877–889, Apr. 2011.
[38] K. Firouzbakht, G. Noubir, and M. Salehi, "On the Capacity of Rate-adaptive Packetized Wireless Communication Links Under
[39] M. Cagalj, S. Ganeriwal, I. Aad, and J.-P. Hubaux, "On Selfish Jamming," in *Proc. of the ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec)*, Tucson, Arizona, USA, 2012, pp. 3–14.
Behavior in CSMA/CA Networks," in *Proc. of IEEE Conference on Computer Communications*, Miami, FL, USA, Mar. 2005, pp. 2513–2524.
[40] Y. Rong, S.-K. Lee, and H.-A. Choi, "Detecting Stations Cheating on Backoff Rules in 802.11 Networks Using Sequential Analysis," in *Proc. of IEEE International Conference on Computer Communications (INFOCOM)*, Barcelona, Spain, Apr. 2006, pp. 1–13.
[41] N. Cai and R. W.-H. Yeung, "Secure Network Coding on a Wiretap Network," *IEEE Transactions on Information Theory*, vol. 57, pp. 424–435, Jan. 2011.
[42] A. Feldmann, *Characteristics of TCP Connection Arrivals*. Wiley-Blackwell, 2002, ch. 15, pp. 367–399.

Liyang Zhang is a Ph.D. student in the Department of Electrical and Computer Engineering at Northeastern University, Boston, MA, USA. He received his B.S. and M.S. in Electrical Engineering from Tsinghua University (China) and State University of New York at Buffalo in 2008 and 2014, respectively. His research interests are in the modeling and analysis of cross-layer attacks and advanced jamming, countermeasure framework design for wireless networks against cross-layer attacks.



Francesco Restuccia (M'16) received his Ph.D. in Computer Science under Prof. Sajal K. Das from Missouri University of Science and Technology, USA in 2016. Currently, he is an Associate Research Scientist in the Department of Electrical and Computer Engineering of Northeastern University, USA. His research interests lie in the modeling, analysis, and experimental evaluation of wireless networked systems, with applications to pervasive computing and the Internet of Things. He has served as a TPC Member of IEEE INFOCOM (2018), IEEE WoWMoM (2017-2018), and IEEE LCN (2016-2018), and is a reviewer for several ACM and IEEE conferences and journals. He is a Member of the IEEE and the ACM.

Tommaso Melodia (M'07, SM'16, F'18) is an Associate Professor with the Department of Electrical and Computer Engineering, Northeastern University, Boston, MA, USA. He is the Director of Research for the PAWR Project Office, a public-private partnership that is developing four city-scale platforms for advanced wireless research in the United States. His research focuses on modeling, optimization, and experimental evaluation of wireless networked systems, with applications to 5G Networks and Internet of Things, software-defined networking, and body area networks. He is a Fellow of the IEEE and a Senior Member of the ACM.

Scott M. Pudlewski (M'12) received his B.S. in Electrical engineering from the Rochester Institute of Technology, Rochester, NY, in 2008, and his M.S. and Ph.D. degrees in electrical engineering from the University at Buffalo, The State University of New York (SUNY), Buffalo, NY in 2010 and 2012, respectively. He is currently an Electronics Engineer at the Air Force Research Laboratory in Rome, NY, USA. His main research interests include video transmission and communications, networking in contested tactical networks, convex optimization, and wireless networks in general. He is a Member of the IEEE.