A Timing Channel Spyware for the CSMA/CA Protocol

Negar Kiyavash, *Member, IEEE*, Farinaz Koushanfar, *Member, IEEE*, Todd P. Coleman, *Senior Member, IEEE*, and Mavis Rodrigues

***Abstract*—This paper presents the design and implementation of spyware communication circuits built into the widely used carrier sense multiple access with collision avoidance (CSMA/CA) protocol. The spyware components are embedded within the sequential and combinational communication circuit structure during synthesis, rendering the distinction or dissociation of the spyware from the original circuit impossible. We take advantage of the timing channel resulting from transmission of packets to implement a new practical coding scheme that covertly transfers the spied data. Our codes are robust against the CSMA/CA's random retransmission time for collision avoidance and in fact take advantage of it to disguise the covert communication. The data snooping may be sporadically triggered, either externally or internally. The occasional trigger and the real-time traffic's variability make the spyware timing covert channel detection a challenge. The spyware is implemented and tested on a widely used open-source wireless CSMA/CA radio platform. We identify the following performance metrics and evaluate them on our architecture: 1) efficiency of implementation of the encoder; 2) robustness of the communication scheme to heterogeneous CSMA/CA effects; and 3) difficulty of covert channel detection. We evaluate criterion 1) completely theoretically. Criterion 2) is evaluated by simulating a wireless CSMA/CA architecture and testing the robustness of the decoder in different heterogeneous wireless conditions. Criterion 3) is confirmed experimentally using the state-of-the-art covert timing channel detection methods.**

***Index Terms*—Timing channels, covert channels, hardware Trojan.**

I. Introduction

RAPID technological advances in wireless communication and the growth in the number and diversity of applications and services raise the demand for data-integrity and communication security. The complexity of the emerging applications and platforms introduce newer vulnerabilities that need to be carefully identified, analyzed, and addressed.

The emerging 802.xx standard protocols must be continually enriched with security features to meet the new and more critical application demands. For instance, the IEEE 802.22 is a cognitive radio standard being developed to bring broadband access to less populated rural areas by using vacant TV channels. To address the complex adaptive nature required by the technology, a cognitive radio device is typically implemented by a general purpose computer processor that also runs radio application software. As a result, it is susceptible to spyware and malicious software or hardware.

In this paper, we present a new transparent and robust covert communication scheme that can reliably spy the chip data to outside entities by exploiting the timing channel resulting from interarrival times of the legitimate transmitted packets. Our scheme can be embedded within any medium access control (MAC) protocol that avoids collision in packet retransmissions by using an exponential back-off rule. For instance, the IEEE 802.11 Wireless Local Area Network (WLAN) standard's main MAC layer protocol, Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA), is an example of a widely used protocol that employs such a rule.

The effects of the CSMA/CA's back-off rule perturbs packet timings by virtue of the queuing and hence introduces a nonstandard noisy channel that interferes with timing-based communication purposes. Although the maximum rate of information exchange of such channels (i.e., capacity) was characterized in [1], only recently have practical encoding/decoding methods to instantiate the theoretical limit with low (i.e., linear) complexity been introduced [2]. However, to date, there have been no practical *covert* information encoding methods with low encoder/decoder complexity to instantiate the findings of [1]. Here, we establish the first reliable covert timing channel that is also robust to the effects of CSMA/CA's back-off rule.

We identify the following performance metrics and evaluate them on our architecture:

1) Transparency of the spyware circuitry: This criterion pertains to the efficiency of implementation at the encoder to prevent the detection of the spyware circuitry.
2) Robustness: This criterion tests the robustness of the communication scheme to heterogeneous CSMA/CA effects which act as noise.
3) Transparency of the covert channel: This criterion measures the difficulty of the covert channel detection.

We evaluate criterion 1) completely theoretically. Criterion 2) is evaluated by simulating a wireless CSMA/CA architecture and

Manuscript received September 20, 2012; revised December 23, 2012; accepted December 24, 2012. Date of publication January 11, 2013; date of current version February 14, 2013. This work was supported in part by AFOSR under Grant FA9550-11-1-0016, Grant 727 AF Sub TX 0200-07UI, and Grant FA9550-10-1-0573, and in part by NSF Grant CCF 10-54937 CAR and Grant CCF 10-65022. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. C.-C. Jay Kuo.

N. Kiyavash is with the Department of Industrial and Enterprise Systems Engineering, and the Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, Urbana-Champaign, IL 61801 USA (e-mail: kiyavash@illinois.edu).

F. Koushanfar is with the Department of Electrical Engineering and Computer Engineering, Rice University, Houston, TX 77005 USA (e-mail: farinaz@rice.edu).

T. P. Coleman is with the Department of Bioengineering, University of California, San Diego, CA 92093 USA (e-mail: tpcoleman@ucsd.edu).

M. Rodrigues was with the Department of Electrical and Computer Engineering, and the Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, Urbana-Champaign, IL 61801 USA (e-mail: mavis.rodrigues@gmail.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIFS.2013.2238930

testing the robustness of the decoder in different heterogeneous wireless conditions. Criterion 3) is confirmed experimentally using state-of-the-art covert timing channel detection methods.

The main contributions of this paper are as follows:

- We present the first design of spyware integrated within the communication circuitry of wireless CSMA/CA that exploits the timing channel resulting from interarrival times of packets to leak data from the chip.
- We introduce a *covert* channel encoding that utilizes the coding methodology developed in [2]. This methodology allows us to encode timing information in presence of the worst nonnegative additive noise, the exponential noise. [3].
- Our spyware employs a low-complexity error-correcting code framework that is robust to timing perturbations resulting from the CSMA/CA's collision avoidance back-off strategy. This back-off strategy introduces a nonstandard queuing noisy channel that interferes with timing-based communication purposes.
- We show the difficulty of the spyware detection, both at the hardware level and at the packet timing level, from both theoretical and practical perspectives.
- To substantiate our theoretical findings and to evaluate the overhead, the spyware is implemented and tested on the widely used WARP wireless radio platform [4].

The remainder of the paper is organized as follows. The related literature is surveyed in Section II. The preliminaries are described in Section III. Section IV introduces our theoretical results for covert channel embedding. In Section V, we devise the new spyware embedding algorithm. Detection of the spyware in circuitry and at the communication channel is presented in Section VI. Comprehensive experimental results for implementation, simulations, and detection are demonstrated in Section VII. We conclude in Section VIII.

II. Related Work

Research in hardware malware has recently emerged, especially after the DoD's Defense Science Board comprehensive report on the subject in 2005 [5]. Much of this work has concentrated on detection of foundry-inserted Trojans (malware), which is typically done by invasive or noninvasive study of the postsilicon ICs and comparison with the original design files to detect the modifications to the manufactured chips [6]. Note that the existing hardware Trojan detection mechanisms are not able to detect our spyware because our assumptions are radically different (we will elaborate on this issue in Section VI). A number of methods for protection of ICs and third-party IP cores against piracy have been invented [7]. The threat of insertion of spyware in communication and security hardware modules has been identified [8], [9]. King *et al.* [10] have designed and implemented malicious hardware that can get hidden login access or can steal the security keys on a microprocessor. The malicious hardware is embedded at the micro-architecture level by addition of a block of gates and does not modify the internals of the architectural blocks. The key advantage of our presynthesis spyware integration is that it is not distinguishable/removable from the radio's internal structure implementing the protocol's state-transitions.

Historically, timing channels are synonymous with covert channels [11]–[15]. Covert channels are mechanisms for communicating information in ways that are difficult to detect. While the focus of earlier work has been mainly on disrupting or completely eliminating covert timing channels [13], [16], the recent work has focused more on detection of covert timing channels [17]–[19]. Moreover, timing side channels pose formidable security and privacy threats [20]–[22].

III. Background

In this section, we first provide a brief background on the employment of the CSMA/CA protocol in the widely used IEEE Wireless Standard 802.11. Afterwards, we summarize the methods for triggering the spyware. Lastly, the incentives for designer spyware insertion are discussed.

A. CSMA/CA in IEEE 802.11

The MAC layer of the IEEE Standard 802.11 for Wireless Local Access Networks (WLANs) uses CSMA/CA for collision avoidance [23].

The main idea behind the CSMA/CA collision avoidance strategy is that the channel is always sensed before any transmission. If the channel is sensed busy, the sender waits until the channel is idle and then goes through a random back-off period before retrying. The random back-off period ensures fairness among contending transmissions. More precisely, the transmission occurs only if, during a fixed period of time, equal to a distributed interface state (DIFS), the channel is sensed idle. If the sensed channel is busy during or immediately after the DIFS, the station continues monitoring the channel until the channel is sensed idle for a DIFS period. At this time, the CSMA/CA generates a random back-off interval before transmitting to avoid the possible collisions by minimizing the probability of collision with packets transmitted by other radios. Even if the medium is sensed idle during the DIFS period, a radio waits a random back-off time between consecutive transmission of two packets to avoid channel capture. The flowchart in Fig. 1 shows the basic steps of the CSMA/CA at the transmitting radio.

Moreover for each packet transmission, an exponential back-off method is used and to signal a successful packet transmission, an ACK is transmitted by the destination radio. This ACK is automatically transmitted at the end of the packet, after a short-time interframe space (SIFS). Since the combination of the SIFS and channel delay is shorter than DIFS, the channel cannot be detected idle for a DIFS until the end of an ACK.

B. Triggering the Spyware

The spyware is activated upon the arrival of a trigger, which may come from either internal or external sources. The spyware does not need to be active all the time. We call the active duration of a spyware the *spying interval*. An internal spyware trigger comes from within the hardware. The two most obvious choices for an internal incitement are the states of the registers in the design, and the clock. The states of the register can be utilized in a number of ways—for example, by arriving at a certain internal state of the communication controller, or upon reaching

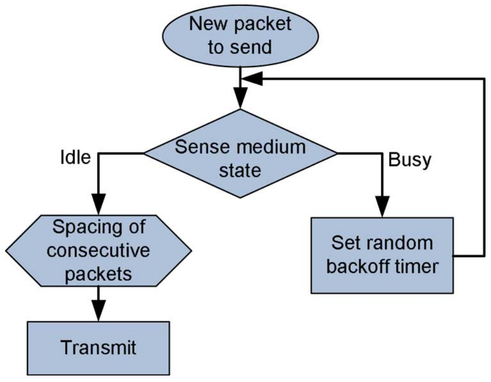

Fig. 1. Flow of the CSMA/CA packet transmission with exponential back-off time.

a certain counter state. Recent research has suggested the possibility of designing challenging (hard-to-detect) Trojans and triggers [24]. Since we implement the trigger at the hardware level and integrate it within the design, it is impossible to detect these types of signals by studying the radio output.

The external triggers have to find a way for the outside source to reach the spyware. The most common method is via the communication channel. Note that other hardware interfaces to the outer world, such as sensors or power lines, can be used as extrinsic stimuli. The covert communication channel can also be used for signaling the trigger. For example, the radio can be intentionally kept busy for certain time intervals.

An important observation is that the external trigger does not have to be sent via the covert channel. For example, the receiver can be manipulated such that a certain combination of the input signals would not create an interrupt to the layers above. At the same time, the carrier sense signal can be raised high by the spyware, disallowing the legitimate messages to route via the communication channel. Unless the secret trigger combination is known and the radio is under physical tests observing its activity, detecting this type of external trigger is extremely difficult, if not impossible. Note that the cost of triggering can be negligible. For example, very often a wire or a single gate is used for triggering [25].

C. *Incentives for Designer Spyware*

Application Specific Integrated Circuit (ASIC) is the predominant technology used for designing power-efficient and low-cost communication circuitry. At least three parties are involved in a typical ASIC supply chain scenario: a design company, a third party fabrication house, and the end user. The spyware (Trojan) discussed in this paper is inserted by the design company during the circuit synthesis which could be applied to both ASIC and Field-Programmable Logic Array (FPGA) technologies. Note that the inclusion of designer spyware is not limited to the untrusted companies for malicious purposes. Contemporary designers often use the backdoors implemented by the spyware for digital rights management purposes.

Several research and development efforts in the area of hardware security have focused on the problem of hardware Trojan where the treat model has been postsynthesis Trojan insertion by the ASIC fabrication house [25], [26]. Since postsynthesis modification of the design blueprint is prohibitively expensive, the fabrication house could only insert the added Trojan components within the white spaces on the blueprint; the white spaces are unavoidable since they are a side-product of automatic synthesis and layout processes. This attack could have the same impact on the communication packets as the designer's spyware. However, this attack may be detectable by the design company who has access to the original design specifications postsynthesis. The designer could use the side-channel tests to compare the measurements from the fabricated chip with the original design simulations [27].

Side-channel tests are inapplicable for the designer spyware detection discussed in this paper. Our spyware is inserted during the synthesis, while the Trojan-free presynthesis specifications are never shared with other parties in the IC supply chain. The spyware and its trigger can be designed to be *point functions* , and thus they can be efficiently obfuscated (hidden) within the exponential state-space of the design [7], [24], [28]. As long as the circuit performs the intended input/output tasks, the hidden functionality of the spyware cannot be distinguished postsynthesis because of its point function properties.

IV. Covert Communication With Timings

Historically, timing channels are synonymous with covert channels [11]–[15], [17], [18], [29]. Covert channels are mechanisms for communicating information in ways that are difficult to detect. Packet networks are designed with the goal of communicating through packet contents and their headers and not the timings. Hence, the timing channel induced by the inter packet timings provides a side channel that can be utilized for covert communications. Fig. 2 illustrates a covert timing channel between an infected transmitter and a legitimate receiver. The *warden* (aka eavesdropper) sees the exchange of packets but fails to realize the covert communication in packet timings. On the other hand, the spyware receiver, which has side information pertaining to the parameters of the encoding scheme, (see Section IV -B) can decode the message conveyed through timings.

As mentioned in Section III, the CSMA/CA protocol results in random delays in the packet interarrival times. We will model the impact of the CSMA/CA as a first-come first-served (FCFS) queuing system. In particular, a special case of a FCFS queuing timing channel is the "exponential server timing channel (ESTC)", where service times are independent and identically distributed (i.i.d.), and memoryless: in continuous-time, they are exponentially distributed of rate μ—with probability density function (PDF) $P_{S_i}(s) = \mu e^{-\mu s}$, and in discrete-time they, are geometrically distributed of rate μ—with probability mass function (PMF) $P_{S_i}(s) = \mu(1-\mu)^{s-1}$. In either case, by defining a_i (d_i) to be the time of the ith packet arrival (departure) of the queue, we have that:

$$P\left(\underline{d}^n | \underline{a}^n\right) = \begin{cases} \prod_{i=1}^{n} P_{S_i}(s) & \text{if } d_i \geq a_i, \ \forall\, i \in \{1,\ldots,n\} \\ 0 & \text{otherwise,} \end{cases} \tag{1a}$$

$$s_i = d_i - \max(a_i, d_{i-1}). \tag{1b}$$

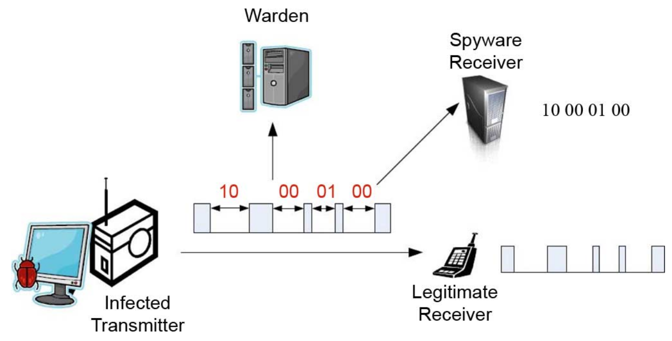

Fig. 2. Covert communication: A warden inspects the communication, but is unable to decode messages modulated by the packet timings.

Ideally, in the absence of any queuing, the information in the timings between successive packets could be used for unlimited rates of communication. However, by adding random delays to the interarrival times of the packets, the CSMA/CA protocol acts as "noise" to the packet timings, as illustrated in Fig. 3. It is the job of the spyware receiver to decode the transmitted message by virtue of the departure process of the queue. The ESTC has the smallest capacity among all FCFS queuing channels with the same average service rate [30], hence a covert communication scheme that can survive the ESTC is most likely robust to other types of queuing noise.

The maximum rate of communication for an ESTC with an arrival process of rate λ satisfies [1]

$$C(\lambda) = \lambda \log_2 \frac{\mu}{\lambda}, \quad \lambda < \mu, \quad \frac{\text{bits}}{\text{s}} \tag{2}$$

where the maximum of (2) is achieved with an input Poisson process at a rate $\lambda^* = e^{-1}\mu$. However it was not clear until recently [2] how to develop practical codes that can withstand the noise from the queueing channels and approach the fundamental limits given by (2). In this work, we will for the first time use similar code constructions for the purpose of *covert* communication over a timing channel introduced due to queuing effects of CSMA/CA.

A. Encoder Structure

The coding schemes of [2] involves an algebraic code construction that allows for a simple encoder, thus lending itself nicely to the design of an in-circuit spyware. In short, a coset code construction [31] followed by shaping is used [31]–[34]. Specifically, we first use a sparse graph coset code (H, s):

$$\mathcal{C} = \{x : Hx = s\}$$

where H is an m by n matrix, s is a length-m vector, and all algebraic operations are defined over the finite field $\mathbb{F}_Q$. The encoder module uses the generator-representation of the code, where a length-k $(k = n - m)$ information vector is mapped to a length-n codeword vector x by use of an n by k generator matrix G and a length-n vector p:

$$x = Gu + p.$$

Thus, in order to perform these operations, only standard finite field arithmetic is required. Lastly, a "shaping" operation is performed to map each $x_i \in \mathbb{F}_Q$ to an interarrival time $z_i \in \mathbb{R}_+$ using a table-lookup:

$$z_i = B_i(x_i).$$

This idea exploits the well-known result [31] that the ensemble of random linear coset codes produces i.i.d. code symbols x_i uniformly distributed over $\mathbb{F}_Q$. So B_i is chosen to map a uniformly distributed random variable, $x_i \in \mathbb{F}_Q$, to a nonuniform, exponentially distributed, interarrival time $z_i \in \mathbb{R}_+$. This can be done via the use of a "dither" D_i, used widely in quantization for provably good performance [35], and the inverse CDF $F_Z^{-1}(\cdot)$ of an exponentially distributed random variable:

$$U_i(x) = \left[\frac{\mathcal{R}(x)}{Q} + D_i\right]_{\bmod 1}, \tag{3a}$$

$$B_i(x) = F_Z^{-1}(U_i(x)) \tag{3b}$$

where $\mathcal{R}(x)$ denotes interpreting $x \in \mathbb{F}_Q$ as a member of $\mathbb{R}_+$. Note that the details of the dither are immaterial to the encoder/decoder pair: B_i is simply a length-Q vector.

B. Decoder Structure

The coding scheme of [2] exploits the graphical structure of the probabilistic dynamics of a queuing system (1a) that enables a low-complexity message-passing decoder. The decoder module resides in software at the spyware receiver, and upon receiving the packets, it decodes the transmitted message using the interpacket times of the departure process (as depicted in Fig. 4).

In this scheme, the concatenation of the graphical model of the error-correcting code (specified by the sparse matrix H, the syndrome s, the shaping operator B_i), and the probabilistic dynamics of a queuing system lead to an aggregate graphical model that is highly structured. This enables the use of standard low-complexity message-passing algorithms (belief propagation, also termed the sum-product algorithm) as a viable decoding solution that produces low bit-error rates even close to the capacity.

V. Implementation

The CSMA/CA which constitutes the communications MAC layer may be implemented in software or hardware. Either way, The MAC needs to interact with the physical layer (PHY) and the hardware components that provide the necessary information for the correct operation of the protocol. The PHY layer can be modified without affecting the upper layer MAC or notifying the changes to any of the higher layers, or even providing false information to them. Fig. 5 shows different ways of interaction between the MAC and the PHY layer. The MAC queries the PHY for the status of the medium and timing information. As can be seen on the figure, the PHY can be modified to mislead the MAC by sending false timing information to it, or even notify the MAC that the medium is busy when it is idle. Perhaps the most important mean for spying the information at this layer is that the front transmission and reception of the packets must be done through the PHY.

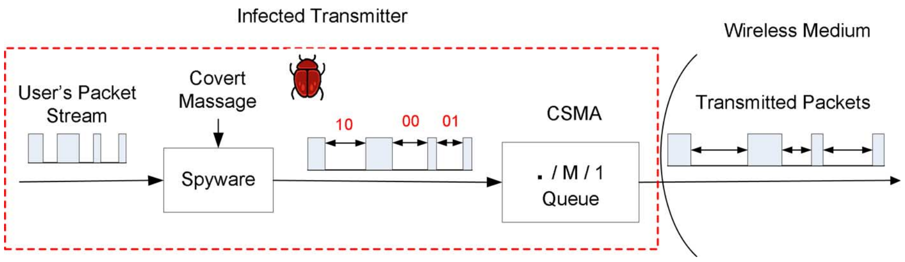

Fig. 3. Bits are encoded into the arrival times of the packet sequence originating from the user's packet stream. The packets are sent over the wireless medium with the augmented interarrival times.

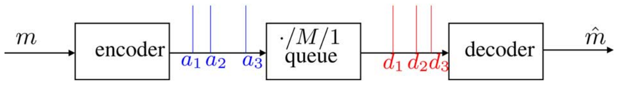

Fig. 4. Block diagram of the spyware encoder and decoder. The spyware receiver decodes the message using the packet's timing sequence (red).

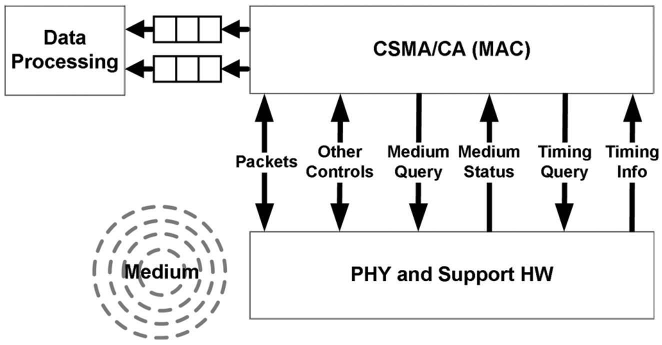

Fig. 5. Interaction between CSMA/CA and the PHY.

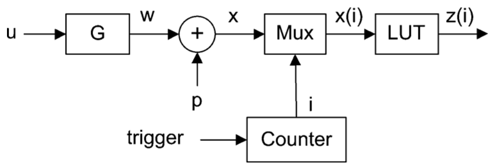

Fig. 6. Block diagram of the encoder circuit.

In our implementation of the spyware, we modify the PHY so that the timing of the sent packets are altered in order to covertly send the stolen data to the outside world. To achieve this goal, we implement the encoder explained in Section IV-A. The block diagram of the encoder circuit is shown in Fig. 6. The encoder circuit was implemented on a reconfigurable radio platform that implements a programmable baseband architecture. We used the WARP platform developed at Rice University for our implementation [4]. This programmable baseband radio contains an FPGA board that can be reconfigured. An advantage of the FPGA implemented in WARP (compared to the simple FPGAs with only lookup tables) is that there it also contains dedicated adders and multipliers that facilitate implementation of various DSP and coding algorithms.

By programming the FPGA, both the generator matrix G and the length-n vector p are hard coded in the PHY layer. G is generated for a specific finite field $\mathbb{F}_Q$. Algorithm 1 shows the steps for constructing the circuit representing multiplications by G. First, a template table T is constructed for $\mathbb{F}_Q$. T has Q entries representing small circuits that perform addition and multiplication of constants from 0 to $Q-1$ with a variable in $\mathbb{F}_Q$. The circuits in the template table are simple XOR gates, constants, or wires. The circuit is generated by the loop in Lines 5 to 7 in the algorithm.

Algorithm 1 Algorithm for Encoding With the Generator Matrix G

```
Require: $Q$: the field size to be used, $G : n \times k$ matrix over
    $\mathbb{F}_Q$, T : $Q \times Q$ matrix over $\mathbb{F}_Q$.
1: Input: $u$, the input vector to the circuit of size $k$
2: Output: $w$, the output vector of the circuit of size $n$
3: for $i = 1$ to $n$ do
4:     $W(i) = 0$
5:     for $j = 1$ to $k$ do
6:         $W(i) = W(i) \oplus T(G(i,j), u(j))$
7:     end for
8: end for
```

The final circuit is a network of XOR gates. The output of this circuit is then added to the constant vector p. This step is also hard coded in the circuit by inverting some of the bits of w to generate x. Each element in x represents an index for a specific interarrival time value stored in the LUT. The elements of x are traversed using a counter. The counter is reset when the trigger is raised. Every time a packet is successfully transmitted, the counter is incremented for generating the next interarrival time.

VI. Spyware Detection

The spyware can be distinguished either by realizing the presence of the spyware circuitry on the chip or by detecting the covert timing channel. We next show that the neither of these two measures are feasible.

A. Detection of Spyware Circuitry

It should be noted that the circuitry of Section V is interwoven and integrated within the physical layer functions. A High Level Description (HDL) Hardware language is used for programming the boards. The HDL code is then compiled and synthesized to hardware components and automatically placed on the circuit. We note that the circuit elements used for realizing the algorithm and blocks shown in the previous section are mostly combinational. There are very few sequential elements that use the LUTs, for example the counters. The overhead of the encoder is a function of the size of G. While a smaller G provides a smaller overhead, there is a trade-off between the size of G

and the accuracy of decoding of the encoded message. An example implementation of the encoder and the trade-off between the overhead and accuracy of the decoder is shown in Table I of Section VII-A. Note that detection of spyware by the power consumption is not feasible since its impact can be easily obfuscated, since the spyware circuit can be actively consuming power without sending any information out. Therefore, it cannot be used for spyware detection purposes. The specific power consumption number is very dependent on the implementation. In our simulated implementation, it was less than 0.5% when the trigger was constantly on. This number is below the noise floor for the measurable power from the supply pin of WARP boards and is not detectable.

The class of attacks that are addressed in this paper are channel-based, where the communication packet timings can be probed. This is because of our spyware implementation scenario. The designer is the implementer of the covert channel and therefore, others who are users would only utilize the implementation. We believe that this is a reasonable assumption for at least three reasons. First, for many products, the manufacturers may provide a backdoor to the design that would often remain unnoticed. The method provided in this paper can be used as a possible backdoor for the implementer. The reason is that the typical users are not provided with the blueprint of the circuitry to be able to investigate the designs as those are often proprietary information and trade secrets. There have been notable cases where it was revealed that the manufacturer has intentionally implemented a backdoor to the design, so this attack is realistic and has been reported. Second, after packaging, the internals of many electronic products are clandestine to the users and cannot be easily debugged and tested. It is true that in our implementation, we use a programmable platform to evaluate our method. One disadvantage or reconfigurability might be that it can be easily reprogrammed and maybe even probed to a certain degree. However, we emphasize that in real products there is a possibility of implementing such channels in ASIC which cannot be reprogrammed or removed. Third, the fabrication house attack scenarios are drastically different than those presented in our paper as discussed in Section III-C.

Let us also discuss a scenario in which another entity designs a circuit with exactly the same functionality. Even this circuit would not provide a good comparison point, because of the degree of freedom in selecting the design components. In other words, there is no certain measure for the expected power or timing that can be computed for a given high level specification, so there would be little relevance between the new entity's design and the original circuitry without the spyware. Thus, the circuit-level attack is not applicable to our spyware scenario.

B. Detection of Covert Communication

The countermeasures to covert timing channels come mainly in two flavors: detection and disruption. The detection of covert timing channels is based on statistical tests that can distinguish between legitimate and covert traffic. The disruption (e.g., jamming) of covert timing channels is done by an active warden that aims at preventing the covert communication usually with the undesired side effect of degraded system performance. While the focus of earlier work has been mainly on disrupting or completely eliminating covert timing channels [13], [16], the recent work has focused more on detection of covert timing channels [17], [18]. This is mainly because the disruption of timing channels is usually costly (either in terms of resources used to active warden or the degradation of the system performance). One possible disruption is random delaying of the traffic, which reduces the performance (e.g., by reducing the capacity or increasing the error rates) of covert timing channels. Notably, our wireless timing channel is susceptible to injection of false packet events by an external entity. For instance, a second node that transmits cover traffic claiming the identity of the node containing the spyware. However, such a countermeasure is both costly in terms of the attacker resources and the lost capacity of the legitimate users of the channel. Therefore, it is highly desirable to employ disruption mechanisms only after one has verified the presence of a covert channel reliably. Moreover, often cryptographic measures can deal with such scenarios. Specifically in our case once the presence of spyware deduced covert channels is detected, the warden will eliminate the covert channel by eradicating the spyware.

We check the detection resistance of our scheme against the current state-of-the-art covert timing channel detection tests.

- **Kolmogorov–Smirnov (K-S) Test.** The KS-test is a nonparametric and distribution free test that evaluates the equality of one-dimensional distributions of probability. The objective is to determine if two datasets (or one dataset and a reference distribution) differ significantly. For the purpose of detection of covert channels, *legitimate traffic* (in the absence of covert communication) is compared to the data collected in a situation where it is suspected that a covert communication is happening. We shall call the samples from the later situation as *test traffic*. If the test traffic is sufficiently different from the legitimate traffic, it is declared that a covert channel has been detected. The KS-test measures the maximum distance between the empirical cumulative distribution functions (CDF) of the legitimate and the test traffic,

$$\max_{x} |F_T(x) - F_L(x)|, \tag{4}$$

  where $F_T(x)$ and $F_L(x)$ denote the empirical CDFs of the test and legitimate traffic.
- **Regularity Tests.** Regularity tests are based on the assumption that the covert traffic is more regular than the legitimate traffic. One example of such a test is Cabuk *et al.*'s heuristic regularity test [17] which examines whether the variance in the interarrival times remains small. Specifically, the test traffic is separated into nonoverlapping windows of size w packets. For each window i, the standard deviation σ_i of the interarrival times is computed. If the standard deviation of the pairwise differences between all pairs σ_i and σ_j for $i < j$ remains small, the traffic is considered highly regular and hence the presence of a covert channel is declared.
- **Entropy Tests.** Entropy-based detection test assumes that the presence of a covert timing channel has an effect on the entropy of a legitimate signal. To examine the entropy, the

TABLE I
OVERHEAD INTRODUCED BY THE ENCODER IMPLEMENTATION ON A WARP NODE FOR THREE ENCODING MATRIX SIZES. THE OVERHEAD IS REPORTED IN TERMS OF NUMBER OF ADDED SLICES/GATES TO THE FPGA

Original		Small (20×8)		Medium (40×16)		Large (80×32)	
		#	Overhead	#	Overhead	#	Overhead
Slices	24,518	24,531	0.05%	24,643	0.50%	24,993	1.94%
Gates	18,741,445	18,741,598	0.00%	18,742,966	0.01%	18,747,184	0.03%

interarrival times are binned into Q bins which are used to calculate the empirical probability densities of patterns of size m in the sequence. A pattern is defined to be a sequence of bin numbers. The estimated entropy of patterns of size m is calculated using these probability densities. The regularity of the data can be measured from the entropy rate. Similarly a slightly different version of test using the conditional entropy estimates can be calculated. For more details see [19].

Other special purpose detection tests that are designed for specific covert timing channels include Cabuk *et al.*'s [17]ϵ-similarity for IP covert timing channels and Berk *et al.*'s [18] mean–max ratio to test for binary or multisymbol covert timing channels. The mean–max ratio test assumes that the legitimate interpacket delays follow a normal distribution which is often not true for real network traffic.

In Section VII, we show that none of the above tests can reliably detect the presence of covert communication.

VII. EXPERIMENTAL RESULTS

The hardware-based covert channel spyware was implemented using the Wireless Open Access Research Platform (WARP) [4], which is a scalable, extensible programmable wireless platform that enables advanced wireless network prototyping. The WARP board contains reconfigurable hardware (FPGA) and can be connected by a UBS port to a PC running Windows. The encoder is implemented in hardware and its overhead when integrated within the WARP platform is reported for different encoder sizes in Section VII-A. The decoder is implemented in software using Matlab, since we do not study the spying receiver's architecture. We also do network level simulations where the communication among the different nodes introduces interfering network traffic. The goal is to evaluate the accuracy of decoding in presence of random interference as well as to study whether the covert flow survive traffic analysis. Our studies require two types of experimental setups: hardware setup and software simulation setup. The details of the hardware setup are as follows:

- The Vertix2P7 FPGA on the WARP platform is used for the encoder implementation. The spyware encoder is implemented in Verilog and integrated within the physical layer blocks on the FPGA.
- CSMA/CA MAC layer protocol is implemented on the PowerPC hardcore processor embedded in the WARP FPGA.
- The physical layer blocks with the exception of the analog front-end are all implemented on the WARP FPGA.
- A field size Q $= 4$ and GF(4) operations are used for encoding.
- Three different sizes of the encoding matrix (G) are implemented: 20×8, 40×16, and 80×32.

The details of the simulation setup are as follows:

- Aside from the spyware encoder and its intended receiver, there are K other nodes (possibly interfering) in the wireless network.
- The transmitting nodes send out the packets according to a Bernoulli process, with burstiness probability p.
- The spyware information is encoded in the timings of the packets, as discussed in Section IV-A, for a fixed encoder rate of $k/n \ \log_2 Q \ = \ 0.4 \text{ bits/packet}$, and $\lambda = 0.225 \text{ packets/sec}$.
- The collision management scheme in the CSMA/CA protocol introduces queuing of the packets—thus acting as a queuing timing channel—as discussed in Section III.
- The spyware decoder senses the transmission times of the wireless radio—which act as the departure process of the queue—and uses the iterative probabilistic decoder discussed in Section IV-B.

The timing diagrams in Fig. 7 demonstrate the signal timings resulting from our WARP/FPGA implementation of the encoding matrix 20×8. Fig. 7(a) contains the timing plot after initiating the CSMA/CA protocol by a trigger signal, denoted as *TX Ready* . In Fig. 7(b) we show the initiation of the CTS time-out after sending an RTS packet. Fig. 7(c) illustrates back-off after a successful frame transmission. The control state of the protocol is shown by the bottom waveform in all our timing diagrams. As expected, the spyware impact is covert and not detectable in timing signals. As we discussed in Section I, the performance of our spyware is evaluated with regards to three criteria: 1) transparency of the spyware circuitry, 2) robustness of the communication scheme to heterogeneous CSMA/CA effects, and 3) transparency of the covert communication scheme.

A. Transparency of the Spyware Circuitry

Here, we study the transparency of the spyware circuitry which is done by evaluating the cost of integrating the hardware encoder on the FPGA. The field size is set to 4 for the implementation. A template table implementing both addition and multiplication in $GF(4)$ is constructed and a C program is written to generate the encoder corresponding to any matrix G of arbitrary size in Verilog. The encoder is inserted within the WARP platform physical blocks on the FPGA.

Table I shows the overhead introduced by the encoder implementation on a WARP node in terms of FPGA added slices and gates. The first column in Table I is labeled *original* and presents the original area of the wireless node components on WARP. The area is given in terms of the number of slices used on the FPGA and the estimated number of gates in the design. The next two columns show the area of the wireless node on the FPGA when we integrate an encoder with small matrix G of size 20×8. The percentage overhead in area in terms of FPGA slices is 0.05%, while the overhead in terms of the extra number of

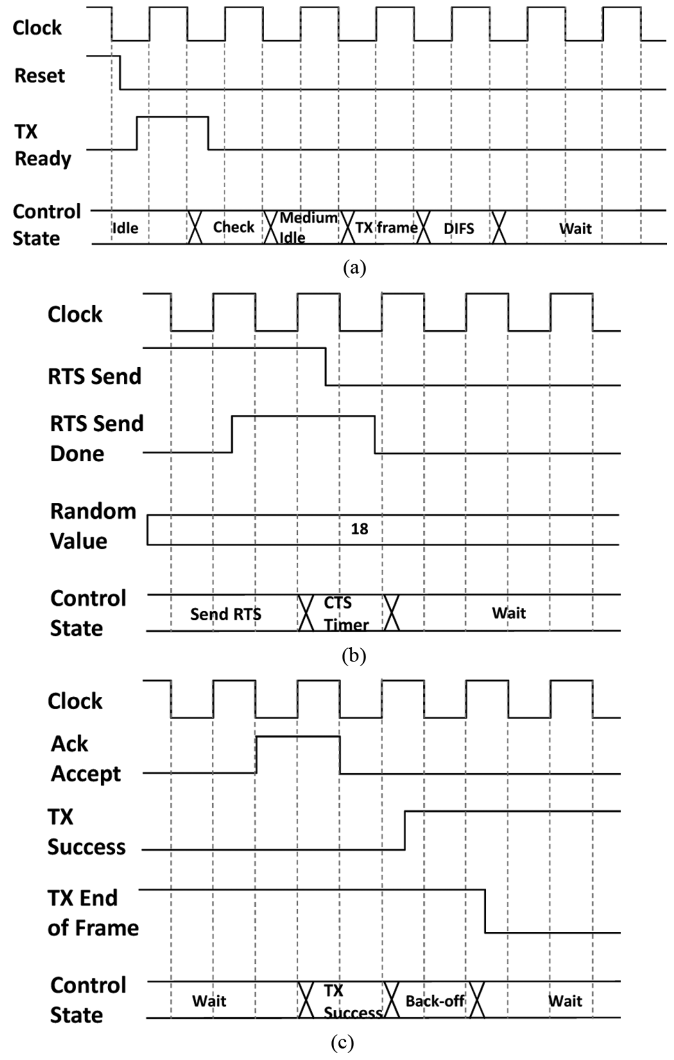

Fig. 7. Timing diagram of WARP/FPGA implementation.

gates is negligible. The next four columns show the area and the percentage overheads when using a matrix G with size 40 × 16, and 80 × 32, respectively. Naturally, the overhead increases as the size of the matrix increases. Due to the very small area overhead, the integration of the encoder has no impact on the overall power of the WARP platform. The spyware introduced in this paper is so small compared to the rest of the circuitry that we are able to insert it on noncritical paths in our implementation. Thus, our encoder does not affect the overall delay of the digital circuitry implementing the radio because it does not affect the critical path of the digital design. The critical path timing is what determines the speed of the chip and is the timing information often provided within the chip specifications. Again we emphasize that the overhead is only measurable by the original designer, and the original unaltered files are never shared with any entity, including the foundry.

B. Robustness of the Communication Scheme to Heterogeneous CSMA/CA Effects

In this subsection, we study the robustness of the spyware communication in terms of the symbol error rate (SER) at the decoder. The SER is affected by the queuing effects introduced by the CSMA/CA due to varying network conditions. We generate multiple levels of interfering traffics on the network by simulating different number of nodes that communicate in the range of both the sender and receiver. It should be noted that the only purpose of the network traffic is to introduce interference that can affect the times of arriving packets carrying the spyware information. Table II shows the average SER at the decoder. These average SER values are shown for various numbers of interfering nodes, namely values of $K \in \{5, 10, 15\}$, as well as burstiness probability $p \in \{0.4, 0.6\}$ for three encoding matrix sizes $(k, n) \in \{(8, 20); (16, 40); (32, 80)\}$ corresponding to small, medium, and large on the table respectively.

TABLE II
AVERAGE SER OF THE SPYWARE DECODER VERSUS NUMBER OF CONGESTING NODES. THE STUDY IS DONE FOR DIFFERENT PROBABILITIES OF SENDING PACKETS OVER THE WIRELESS MEDIUM, AND THE MATRIX SIZES

Number of nodes	Burstiness prob. = 0.4			Burstiness prob. = 0.6		
	Small	Medium	Large	Small	Medium	Large
5	.16	.08	.08	.13	.09	.05
10	.14	.10	.09	.14	.10	.07
15	.14	.09	.08	.13	.09	.08

Although the SER values are only on the order of 10^{-2}, we bring attention to the specific context of this application: a short code-length, externally triggered, and hardware implementation. Since this approach uses linear coset codes (which have extremely small undetected error probability [36]), an error event is essentially equivalent to the condition $H\hat{x} \neq s$. Thus, the spyware receiver can simply externally retrigger the transmission at a random time in the future. As it can be expected, Table II shows that as the length of the code (size of the encoding matrix) increases, the average symbol error rate at the decoder improves.

Fig. 8 illustrates boxplots of the SER across multiple trials. The lower edge of the rectangular plot corresponds to $e_{.25}$, the 25th percentile; the upper edge pertains to $e_{.75}$, the 75th percentile. The 50th percentile, or median, appears as the horizontal line between the lower and upper bounds. The upper 'tail' of the box corresponds to the vertical line extending beyond the 75th percentile, which is of length $1.5(e_{.75} - e_{.25})$. All outliers extending beyond the 'tail' are explicitly drawn with the '$+$' symbol. We see from Fig. 8 that the statistical structure of the SER is highly concentrated near 0, in such a way that cannot be evidenced by merely the mean SER. Specifically, the 25th percentile of the errors is 0. In terms of externally triggered retransmissions, this means that simple "majority-rules" decoding schemes will work particularly well with this approach. More specifically, while the average SER is of the order of 10^{-2}, around 25% of the time, the decoder receives symbols with zero error and thus a majority decoding rule after 2 or 3 transmissions will result in perfect decoding.

C. Transparency of the Covert Communication Scheme

In this subsection, we study the detectability of the covert information hidden in the network traffic. This is done by generating regular traffic and comparing it to spyware's covert traffic for various number of interfering nodes. The simulated traffic was generated and tested using Matlab on a 64-bit

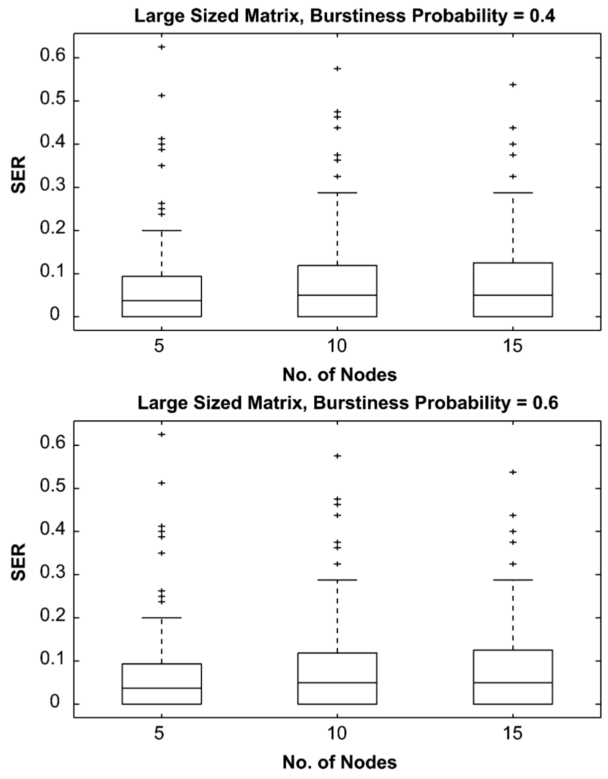

Fig. 8. Boxplots of the symbol errors across multiple trials, using the large encoding matrix with burstiness probabilities $p \in \{0.4, 0.6\}$ with 200 runs.

Six-Core AMD Opteron(tm) Processor 8431. To test the visibility of covert traffic, we compared two types of simulated traffic: (i) Regular traffic: simulated traffic generated using a Bernoulli distribution with burstiness probability p with no covert message embedded, and subjected to the CSMA/CA back off; (ii) Covert traffic: simulated traffic with a covert message embedded. This covert message was embedded using the scheme in Section IV-A and the specifications in the beginning of Section VII and in Section VII-C. This traffic was then subjected to the CSMA/CA queuing effects for K number of interfering nodes. Interarrival times from the legitimate and covert traffic were used as inputs to three covert channel detection tests: Kolmogorov–Smirnov, regularity, and entropy tests of Section VI-B.

Table III gives the results for the Kolmogorov–Smirnov test when a covert message was inserted in flows consisting of 10000 packets. In each of these flows, we randomly injected an external trigger activating covert transmission consisting of $n = 20$ (small), $n = 40$ (medium), or $n = 80$ (large) consecutive packets in every interval consisting of 250, 500, 1000 or 2000 packets. In all cases, the threshold for the hypothesis test was the deviation of the Komolgorov–Smirnov statistic beyond the 99% confidence interval, and the results are an average of over 10000 trials. When the covert message was inserted in every 1000 or 2000 packets, our approach was successfully undetectable—the detection probability, P_D and the false alarm probability, P_{FA} (fixed at 1%) are close to one another. When injecting the covert transmission within every 250 and 500 packets, the transmission becomes more visible, although the detection rates remain low except when $n = 80$ over 250 packets, in which case most of the covert transmissions are detected.

TABLE III
DETECTION (P_D) PROBABILITIES FOR USING A KOMOLGOROV–SMIRNOV TEST ON THE PACKET INTERARRIVAL TIMES (FALSE ALARM PROBABILITY FIXED AT 1%)

Number of packets	P_D		
	Small	Medium	Large
250	0.0357	0.2820	0.9022
500	0.0164	0.0280	0.3288
1000	0.0114	0.0195	0.0258
2000	0.0106	0.0122	0.0206

TABLE IV
DETECTION (P_D) PROBABILITIES FOR THE REGULARITY TEST (WINDOW SIZE $w = 100$ PACKETS AND FALSE ALARM PROBABILITY FIXED AT 1%)

Number of packets	P_D		
	Small	Medium	Large
250	0.315	0.874	0.948
500	0.147	0.370	0.885
1000	0.058	0.108	0.228
2000	0.029	0.065	0.105

Table IV illustrates the results for the regularity test with nonoverlapping windows of size $w = 100$ packets over a collection of 250, 500, 1000 or 2000 packets. The results are averaged over a 1000 trials and the threshold was set beyond the 99% confidence interval. The number of possibly interfering nodes and the burstiness probability were chosen to be $K = 5$ and $p = 0.4$, respectively.

Again, we randomly injected an external trigger activating covert transmission of $n = 20$ (small), $n = 40$ (medium), or $n = 80$ (large) consecutive packets. As mentioned earlier in Section VI-B, the regularity test of Cabuk *et al.* [17] examines whether the traffic is more regular than it is expected with legitimate traffic. Specifically, it tests whether the variance of interarrival times remains small. As seen in Table IV, the regularity test completely fails to detect the covert communication when the covert message was inserted in every 1000 or 2000.

In the following entropy detection tests, in accordance with recommendations of [19], we used bin sizes of $Q = 5$ and $Q = 65,536$, subsequently for the conditional entropy and the entropy tests. Tables V and VI shows the detection probabilities, P_D (P_{FA} fixed at 1%) for the two entropy-based detection tests on an infected traffic consisting of 10000 packets, where the covert transmission was inserted in every interval consisting of 250, 500, 1000 or 2000 packets. The external trigger activating covert transmission of $n = 20$ (small), $n = 40$ (medium), or $n = 80$ (large) consecutive packets was done at random positions in each interval. The threshold was set beyond the 99% confidence interval and the results are an average over 1000 trials. The number of possibly interfering nodes and the burstiness probability were chosen to be $K = 5$ and $p = 0.4$, subsequently.

To see the effect of number of possibly interfering streams on the entropy-based, we tested the detection and false alarm probabilities when $n = 20$ (small), $n = 40$ (medium), or $n = 80$ (large) packets containing a covert transmission were sent. In this case, we did not embed the packets in a larger transmission sequence. Table VII illustrates the results for $K = 5, 10, 15$ interfering nodes still with a burstiness probability of $p = 0.4$ for

TABLE V
DETECTION PROBABILITIES, P_D FOR ENTROPY TEST ON THE PACKET INTERARRIVAL TIMES IN PRESENCE OF RANDOM TRIGGER (FALSE ALARM PROBABILITY FIXED AT 1%)

Number	P_D		
of packets	Small	Medium	Large
250	0.206	0.319	0.419
500	0.144	0.245	0.441
1000	0.058	0.082	0.178
2000	0.061	0.064	0.090

TABLE VI
DETECTION PROBABILITIES, P_D FOR CONDITIONAL ENTROPY TEST ON THE PACKET INTERARRIVAL TIMES IN PRESENCE OF RANDOM TRIGGER (FALSE ALARM PROBABILITY FIXED AT 1%)

Number	P_D		
of packets	Small	Medium	Large
250	0.248	0.646	0.877
500	0.094	0.232	0.656
1000	0.043	0.050	0.148
2000	0.046	0.056	0.066

TABLE VII
DETECTION (P_D) AND FALSE ALARM (P_{FA}) PROBABILITIES FOR CONDITIONAL ENTROPY TEST VERSUS NUMBER OF POSSIBLY INTERFERING STREAMS (k)

Number of possibly	P_D			P_{FA}		
interfering streams (K)	Small	Medium	Large	Small	Medium	Large
5	.0095	.0130	0.0195	.0120	.0095	.0070
10	.0030	.0020	0.0120	.0030	.0020	0.020
15	.0003	.0020	0.0250	.0003	.0020	.0040

the conditional entropy case. The results for entropy test were similar and hence omitted for sake of brevity. It can be seen that as number of interfering nodes increase the detection performance of the conditional entropy test decreases. One possible explanation is that because CSMA back off protocol engages more often to ensure collision avoidance, more packet timings are altered from their regular pattern, resulting in more randomness. This increase randomness in the streams seems to impair the detection test.

As clearly demonstrated by our simulations, none of the three detection tests can reliably detect presence of covert communication. In fact, often times the false alarm rates are comparable, if not higher, than the detection rates. However, note that the best transparency results are achieved when smaller number of covert packets are inserted more sporadically into the traffic, thus effectively reducing the transmission rate of the covert communication.

VIII. CONCLUSION AND FUTURE WORK

We have shown a new methodology for the design and implementation of spyware communication circuits that exploits the CSMA/CA MAC protocol properties to covertly leak the chip data to an intended spyware receiver. Our proposed scheme is robust to the collision avoidance techniques in the CSMA/CA protocol of the MAC layer. We interweave the encoder in the states and transitions of the wireless transmitter presynthesis, such that distinction and uncoupling of the spyware from the radio's physical layer is impossible. The spying act is initiated upon an occasional external or internal trigger. The proof-of-concept implementation is demonstrated on the widely used Wireless Open Access Research Platform (WARP). Three metrics were used for evaluation of the spyware performance: 1) efficiency of the encoder implementation, which was theoretically demonstrated; 2) robustness of the communication method to CSMA/CA effects, which was done by simulating the wireless architecture across a range of parameters; 3) difficulty of covert channel detection that was shown to be resilient against the known covert timing channels detection methodologies. While there exists a trade-off between accuracy, overhead and detectability, our experimental results show that our implemented spyware simultaneously meets the desired above metrics.

Some possible future directions include detection of the "trigger" time, or the time at which covert communication begins. One may think about implementing the spyware in other timing aspects of the wireless network such as rate adaptation or sleeping mode. It can potentially be done by the use of "change point detection" approaches [37] where they find the location in time where interarrival times drawn from a distribution F_0 switch to a distribution F_1. However, if the circuit-level design criterion permits more sophisticated shaping schemes, even the change-point detection will fail. For instance, the encoder's shaping technique (3) of Section IV can be used to design provably undetectable covert communication schemes. At a high level, the idea is that the dithering scheme discussed in Section IV has two important properties:

- as we are using a random sparse graph linear coset code, it follows from [31] the x_i symbols are independent and identically distributed.
- the dithering technique in (3) allows each z_i to be i.i.d. with distribution given by $F_Z(z)$ used in the shaping.

It follows theoretically that any statistical test that models normal traffic as i.i.d. with interpacket timings of distribution F_Z, will provably fail. More specifically, the inverse CDF, $F_Z^{-1}(\cdot)$, can be simply shaped to any desirable legitimate traffic model, as long as the interpacket times are i.i.d.

REFERENCES

[1] V. Anantharam and S. Verdu, "Bits through queues," *IEEE Trans. Inf. Theory*, vol. 42, no. 1, pp. 4–18, Jan. 1996.

[2] T. Coleman and N. Kiyavash, "Sparse graph codes and practical decoding algorithms for communicating over packet timings in networks," in *Proc. Conf. Inform. Sci. and Syst.*, 2008, pp. 447–452.

[3] S. Verdu, "The exponential distribution in information theory," *Problems Inf. Transmission*, vol. 32, no. 1, pp. 86–95, 1996.

[4] WARP: Wireless Open Access Research Platform, 2008 [Online]. Available: http://www.dupont.com/mcm/techinfo.html

[5] Defense Science Board (DSB) Study on High Performance Microchip Supply 2005 [Online]. Available: http://www.acq.osd.mil/dsb/reports/2005-02-HPMSn Reportn Final.pdf

[6] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar, "Trojan detection using IC fingerprinting," in *Proc. IEEE Symp. Security and Privacy*, 2007, pp. 296–310.

[7] Y. Alkabani and F. Koushanfar, "Active hardware metering for intellectual property protection and security," in *Proc. USENIX Security*, 2007, pp. 291–306.

[8] Y. Jin and Y. Makris, "Hardware Trojans in wireless cryptographic ICs," *IEEE Design Test Comput.*, vol. 27, no. 1, pp. 26–35, Jan./Feb. 2010.

[9] S. Adee, The Hunt for the Kill Switch May 2008 [Online]. Available: http://www.spectrum.ieee.org/may08/6171

[10] S. King, J. Tucek, A. Cozzie, C. Grier, W. Jiang, and Y. Zhou, "Designing and implementing malicious hardware," in *Proc. USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET)*, 2008, pp. 1–8.

[11] B. Lampson, “A note on the confinement problem,” *Commun. ACM*, vol. 16, 1973.

[12] A. B. Wagner and V. Anantharam, “Information theory of covert timing channels,” presented at the NATO/ASI Workshop on Network Security and Intrusion Detection, Yerevan, Armenia, Oct. 2005.

[13] M. Kang and I. Moskowitz, “A pump for rapid, reliable, secure communication,” in *Proc. ACM Conf. Comput. and Commun. Security (CCS)*, 1993, pp. 119–129.

[14] V. Anantharam and S. Verdu, “Reflections on the 1998 information theory society paper award: Bits through queues,” *IEEE Inf. Theory Soc. Newsletter*, pp. 100–102, Dec. 1999.

[15] X. Luo, E. W. Chan, and R. K. C. Chang, *Covert Communications in TCP Burstiness*. Hong Kong: Hong Kong Polytechnic Univ., 2007.

[16] J. Giles and B. Hajek, “An information-theoretic and game-theoretic study of timing channels,” *IEEE Trans. Inf. Theory*, vol. 48, no. 9, pp. 2455–2477, Sep. 2002.

[17] S. Cabuk, C. Brodley, and C. Shields, “IP covert timing channels: Design and detection,” in *Proc. ACM Conf. Comput. and Commun. Security (CCS)*, 2004, pp. 178–187.

[18] V. Berk, A. Giani, and G. Cybenko, Detection of Covert Channel Encoding in Network Packet Delays Dartmouth College, Dept of Computer Science, Tech. Rep. TR2005536, 2005.

[19] S. Gianvecchio and H. Wang, “Detecting covert timing channels: an entropy-based approach,” in *Proc. 14th ACM Conf. Comput. and Commun. Security*, 2007, pp. 307–316.

[20] S. Kadloor, X. Gong, N. Kiyavash, T. Tezcan, and N. Borisov, “A low-cost side channel traffic analysis attack in packet networks,” in *Proc. Commun. and Inform. Syst. Security Symp. (IEEE ICC 2010)*, Cape Town, South Africa, 2010.

[21] X. Gong, N. Kiyavash, and N. Borisov, “Fingerprinting websites using remote traffic analysis,” in *Proc. 17th ACM Conf. Comput. and Commun. Security*, 2010, pp. 684–686.

[22] S. Kadloor, X. Gong, N. Kiyavash, and P. Venkitasubramaniam, “Designing router scheduling policies: A privacy perspective,” *IEEE Trans. Signal Process.*, vol. 60, no. 4, pp. 2001–2012, Apr. 2012.

[23] *LAN/MAN Wireless LANS Standard*, IEEE 802.11, 2007 [Online]. Available: http://standards.ieee.org/getieee802/802.11.html

[24] S. Wei, K. Li, F. Koushanfar, and M. Potkonjak, “Hardware Trojan horse benchmark via optimal creation and placement of malicious circuitry,” in *Proc. Design Automation Conf. (DAC)*, 2012, pp. 90–95.

[25] M. Tehranipoor and F. Koushanfar, “A survey of hardware Trojans: Taxonomy and detection,” *IEEE Design Test Comput.*, vol. 27, no. 1, pp. 10–25, Jan./Feb. 2010.

[26] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor, “Trustworthy hardware: Identifying and classifying hardware Trojans,” *IEEE Computer*, vol. 43, no. 10, pp. 39–46, Oct. 2010.

[27] F. Koushanfar and A. Mirhoseini, “A unified framework for multimodal submodular integrated circuits Trojan detection,” *IEEE Trans. Inf. Forensics Security*, vol. 6, no. 1, pp. 162–174, Mar. 2011.

[28] F. Koushanfar, “Provably secure active IC metering techniques for piracy avoidance and digital rights management,” *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 1, pp. 51–63, Feb. 2012.

[29] Trusted Computer System Evaluation, U.S. Department of Defense, Tech. Rep. DOD 5200.28-STD, 1985.

[30] R. Sundaresan and S. Verdu, “Robust decoding for timing channels,” *IEEE Trans. Inf. Theory*, vol. 46, no. 2, pp. 405–419, Mar. 2000.

[31] R. G. Gallager, *Information Theory and Reliable Communication*. Hoboken, NJ, USA: Wiley, 1968.

[32] A. Bennatan and D. Burshtein, “Design and analysis of nonbinary LDPC codes for arbitrary discrete-memoryless channels,” *IEEE Trans. Inf. Theory*, vol. 52, no. 2, pp. 549–583, Feb. 2006.

[33] C. Fragouloi, R. D. Wesel, D. Sommer, and G. Fettweis, “Turbo codes with nonuniform QAM constellations,” in *Proc. IEEE Int. Conf. Commun.*, 2001, pp. 70–73.

[34] F.-W. Sun and S. van Tilborg, “Approaching capacity by equiprobable signaling on the Gaussian channel,” *IEEE Trans. Inf. Theory*, vol. 39, no. 5, pp. 1714–1716, Sep. 1993.

[35] J. Ziv, “On universal quantization,” *IEEE Trans. Inf. Theory*, vol. 31, no. 3, pp. 344–347, May 1985.

[36] T. Richardson and R. Urbanke, *Modern Coding Theory*. Cambridge, U.K.: Cambridge Univ. Press.

[37] A. G. Tartakovsky and V. V. Veeravalli, “Asymptotic analysis of Bayesian quickest change detection procedures,” in *Proc. IEEE Int. Symp. Inform. Theory*, 2002, p. 217.

Negar Kiyavash (S’99–M’06) is an assistant professor in the Department of Industrial and Enterprise Systems Engineering (ISE) at the University of Illinois at Urbana-Champaign, USA. She received the Ph.D. degree in electrical and computer engineering from the University of Illinois at Urbana-Champaign in 2006. Her research interests include information theory and statistical signal processing with applications to security and network inference.

Dr. Kiyavash is a recipient of the NSF CAREER and AFOSR YIP awards.

Farinaz Koushanfar (S’99–M’05) received the Ph.D. degree in electrical engineering and computer science from University of California Berkeley, in 2005.

She is currently an Associate Professor with the Department of Electrical and Computer Engineering, Rice University, Houston, TX, USA, where she directs the Texas Instruments DSP Leadership University Program. Her research interests include adaptive and low power embedded systems design, hardware security, and design intellectual property protection.

Prof. Koushanfar is a recipient of the PECASE Award, the ACM SIGDA Outstanding New Faculty Award, the National Academy of Science Kavli Foundation fellowship, the ARO, ONR, and DARPA YIP, and the NSF CAREER Awards, and a best paper award at Mobicom.

Todd P. Coleman (S’01–M’05–SM’11) received the Ph.D. degree in electrical engineering from the Massachusetts Institute of Technology (MIT), Cambridge, in 2005.

He was a postdoctoral scholar in neuroscience at MIT and MGH during the 2005–2006 academic year. He was an Assistant Professor in ECE and Neuroscience at the University of Illinois from 2006–2011. He is currently an Associate Professor in Bioengineering and director of the Neural Interaction Laboratory at UCSD. His research is highly interdisciplinary and lies at the intersection of bio-electronics, neuroscience, medicine, and applied mathematics.

Dr. Coleman is a science advisor for the Science & Entertainment Exchange (National Academy of Sciences).

Mavis Rodrigues received the Masters degree in electrical and computer engineering from University of Illinois at Urbana-Champaign, USA, in 2012. Her research interests are in timing channels and their application to network security.