

A Timing Channel Spyware Robust to MAC Random Back-off

Yousra Alkabani, *Student Member, IEEE*, Negar Kiyavash, *Member, IEEE*,
Farinaz Koushanfar, *Member, IEEE* and Todd Coleman, *Member, IEEE*

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.

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 timing channel resulting from inter-arrival 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 non-standard 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 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 inter-arrival times of packets to leak data from the chip.
- We introduce the first *covert* channel encoding that practically attains rates near the fundamental limits utilizing some methodologies developed in [2].
- 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 non-standard 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 [3].

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 [4]. Much of this work has concentrated on detection of foundry-inserted Trojans (malware), which is typically done by invasive or noninvasive study of the post-silicon ICs and comparison with the original design files to detect the modifications to the manufactured chips [5]. 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 [6]. The threat of insertion of spyware in communication and security hardware modules has been identified [7]. Recently, King et al. [8] 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 pre-synthesis 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 [9]–[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 [16], [17], the recent work has focused more on detection of covert timing channels [18], [19].

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.

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 [20].

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 Figure 1 shows the basic steps of the CSMA/CA at the transmitting radio.

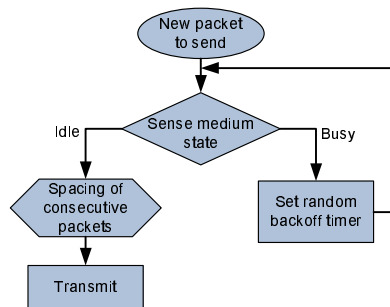


Fig. 1. Flow of the CSMA/CA packet transmission with exponential back-off time.

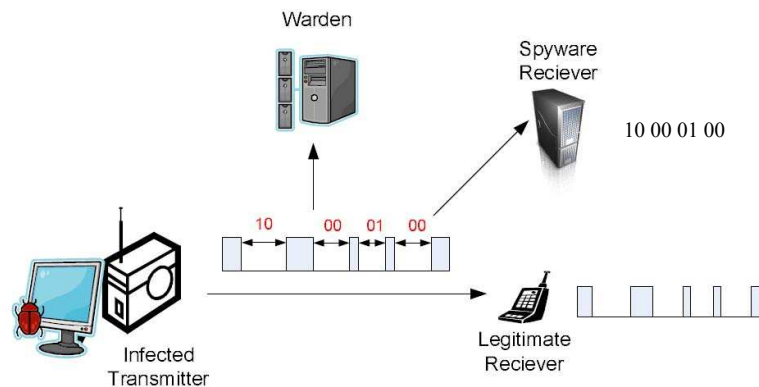


Fig. 2. Covert communication: A warden inspects the communication, but is unable to decode messages modulated by the packet timings.

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 inter-frame 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 a certain counter state. 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.

IV. COVERT COMMUNICATION WITH TIMINGS

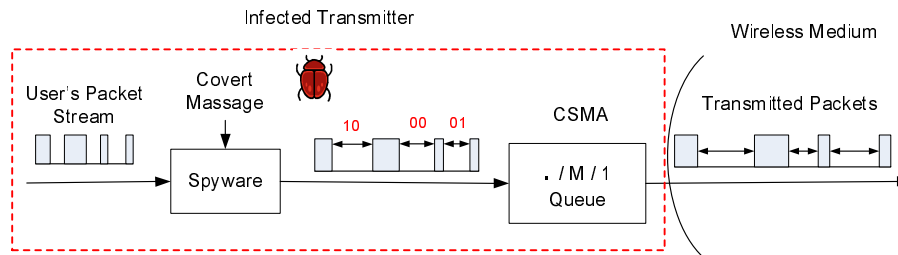


Fig. 3. Bits are encoded into the arrival times of the packet sequence originating from the user's packet stream. The packets are sent over the wireless medium with the augmented inter-arrival times.

Historically, timing channels are synonymous with covert channels [9]–[15], [18], [19]. 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. Figure 2 illustrates a covert timing channel between an infected transmitter and a legitimate receiver. The *warden* (a.k.a 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 inter-arrival 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 i th packet arrival (departure) of the queue, we have that:

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

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

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 inter-arrival times of the packets, the CSMA/CA protocol acts as “noise” to the packet timings, as illustrated in Figure 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 [21], 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]

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

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 queueing 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 in-circuit spyware. In short, a coset code construction [22] followed by shaping is used [22]–[25]. 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 inter-arrival time $z_i \in \mathbb{R}_+$ using a table-lookup:

$$z_i = B_i(x_i).$$

This idea exploits the well-known result [22] 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 non-uniform, exponentially distributed, inter-arrival 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 [26], and the inverse CDF $F_Z^{-1}(\cdot)$ of an exponentially distributed

random variable:

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

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

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 inter-packet times of the departure process (as depicted in Figure 4).

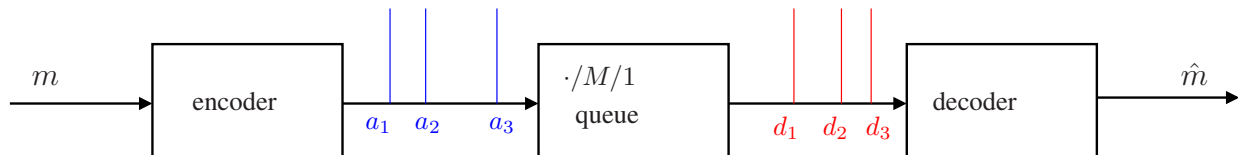


Fig. 4. A block diagram of the spyware encoder and decoder. The spyware receiver decodes the message using the packet's timing sequence (red).

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. Figure 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.

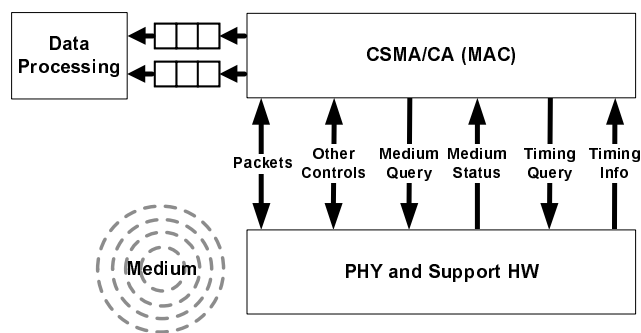


Fig. 5. Interaction between CSMA/CA and the PHY.

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 Figure 6.

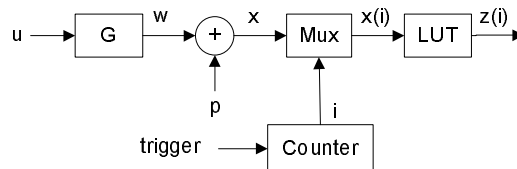


Fig. 6. Block diagram of the encoder circuit.

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 just 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 inter-arrival 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 inter-arrival 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 above circuitry of Section V is mainly combinational and can be interwoven and concealed within the sea of gates in the PHY and the communication signal processing circuitry. Only a few latches are used for the counter and the LUT that can also

be effectively embedded within the regular combinational logic in the circuit. The overhead presented by the encoder is a function of the size of G . However, while a smaller G provides a smaller overhead, there is a tradeoff between the size of G and the accuracy of decoding of the encoded message. An example implementation of the encoder and the tradeoff between the overhead and accuracy of the decoder is shown in Table I of Section VII-A.

Our implementation method renders the removal attack almost impossible: it is well known that reverse-engineering the functionality of the circuit by accessing the gate-level information and even the layout information is not practically possible [6]. This is because the synthesis step of the design which transforms the functional specifications into the gate-level consists of several optimizations that do not maintain the functional relationships. Note that circuit verification cannot reveal the spyware either. The circuit is verified for having the same functionality as the protocol specification by means of checking the integrity of the protocol states. Verification does not generally detect (verify) the unknown added states. Furthermore, the verification criteria is often set by the designer who is also the spyware implementer in our case and can effectively hide the spyware from the other verifiers.

Note that the currently available Trojan detection methods are unable to detect our circuit level manipulations and spyware. All of the Trojan detection methods are based on the assumption that the original circuit's hardware description (the netlist or GDSII files that the designer sends to the foundry) are available at the detector [5]. The adversarial model is that the foundry is changing the circuitry by adding components. The designer gets the ICs back, and can test to see if the IC is implementing its original design. In our model, the designer is modifying the circuit herself and ultimately decides the number and arrangement of gates. Furthermore, given the complexity of the state-of-the-art designs, reverse-engineering is not practical.

Let us assume 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 minimum number of gates or particular arrangement of them that can be computed for a given specification, so there would be little relevance between the new entity's design and the original circuitry without the spyware. Thus, this attack is not at all valid on our spyware.

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 [16], [17], the recent work has focused more on detection of covert timing channels [18], [19]. 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 non-parametric 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. Specifically, the KS-test measures the maximum distance between the empirical cumulative distribution functions (CDF) of the legitimate

and the test traffic,

$$\max_x |F_T(x) - F_L(x)|, \quad (4)$$

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 [18] which examines whether the variance in the inter-arrival times remains small. Specifically, the test traffic is separated into non-overlapping windows of size w packets. For each window i , the standard deviation σ_i of the inter-arrival 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.

Other special purpose detection tests that are designed for specific covert timing channels include Cabuk et al.’s [18] ϵ -similarity for IP covert timing channels and Berk et al.’s [19] mean-max ratio to test for binary or multi-symbol covert timing channels. The mean-max ratio test assumes that the legitimate inter-packet 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

Original		Small		Medium		Large	
		#	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%

TABLE I

THE OVERHEAD INTRODUCED BY THE ENCODER IMPLEMENTATION ON A WARP NODE FOR THREE ENCODING MATRIX SIZES.

The hardware-based covert channel spyware was implemented using the Wireless Open Access Research Platform (WARP) [3], which is a scalable, extensible programmable wireless platform

that enables advanced wireless network prototyping. 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 can be detected by 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 $\frac{k}{n} \log_2 Q = 0.4$ bits/packet, and $\lambda = 0.225$ 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.

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

In this subsection, we study the transparency of the spyware circuitry. This is done by evaluating the cost of integrating the hardware encoder on the FPGA. The field size (Q) 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 then inserted within the WARP platform physical blocks on the FPGA.

Table I shows the overhead introduced by the encoder implementation on a WARP node. 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 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. Also, the 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. 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.

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 [27]), 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.

Figure 7 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 Figure 7 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 the traffic with spyware covert channels for various network settings. We use the different traffic flows to generate inter-arrival

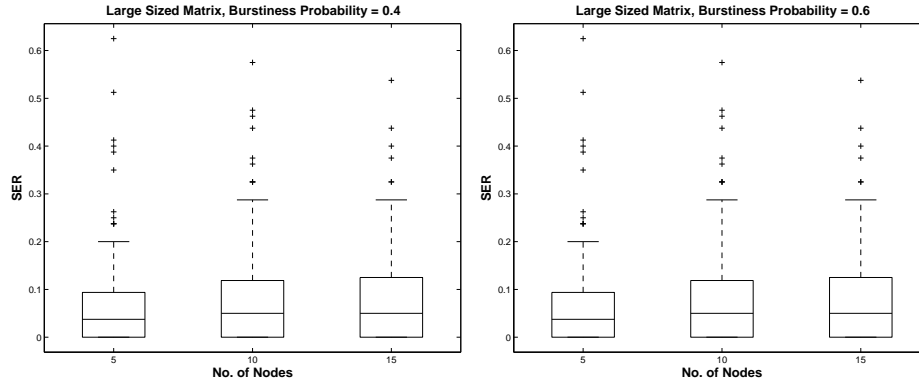


Fig. 7. Boxplots of the symbol errors across multiple trials, using the large encoding matrix with burstiness probabilities $p \in \{0.4, 0.6\}$ with 200 runs.

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

TABLE II

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

times that are used as inputs to two covert channel detection tests: Komolgorov-Smirnov and regularity tests of Section VI-B.

Table III illustrates results for the Komolgorov-Smirnov test where, over a collection of 1000/2000 packets, we randomly injected an external trigger activating covert transmission of $n = 20$ (small), or $n = 40$ (medium), or $n = 80$ (large) consecutive packets. The false alarm event occurs when all packets transmitted do not have a trigger but are erroneously deemed to have been containing spyware covert timing information. The rightmost column illustrates the false alarm probability (P_{FA}) for the 1000/2000 packet transmission scenarios. Meanwhile, the detection event pertains to the scenario when an external trigger is randomly injected into the packet stream and it is successfully detected using the hypothesis testing procedure. Detection probabilities (P_D) are given for small, medium, and large matrix sizes. In all cases, the threshold

for the hypothesis testing procedure was the deviation of the Komolgorov-Smirnov statistic, given by (4), beyond the 95% confidence interval. As we can see, this approach successfully evades detection: P_D and P_{FA} are close to one another in virtually all scenarios. We note that for larger matrix sizes, the deviation between P_D and P_{FA} increases, thus illustrating a tradeoff between error tolerance robustness and detection probability.

Number of packets	P_D			P_{FA}
	Small	Medium	Large	
1000	.033	.052	.112	.031
2000	.036	.049	.069	.026

TABLE III

DETECTION (P_D) AND FALSE ALARM (P_{FA}) PROBABILITIES FOR USING A KOMOLGOROV-SMIRNOV TEST ON THE PACKET INTER-ARRIVAL TIMES.

Packets	Small		Medium		Large	
	Covert	Non-covert	Covert	Non-covert	Covert	Non-covert
1000	0.36	0.3	0.36	0.3	0.35	0.3
2000	0.4	0.37	0.38	0.3	0.37	0.3

TABLE IV

REGULARITY TEST FAILS TO DETECT THE PRESENCE OF COVERT COMMUNICATION (WINDOW SIZE $w = 100$ PACKETS).

Table IV illustrates the results for the regularity test with non-overlapping windows of size $w = 100$ packets over a collection of 1000/2000 packets. Again we randomly injected an external trigger activating covert transmission of $n = 20$ (small), or $n = 40$ (medium), or $n = 80$ (large) consecutive packets. As mentioned earlier in Section VI-B, the regularity test of Cabuk et al. [18] examines whether the traffic is more regular than it is expected with legitimate traffic. Specifically, it tests whether the variance of the inter-arrival times remains small. As seen in Table IV, the regularity test completely fails to detect 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 pre-synthesis, 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. It can potentially be done by the use of “change point detection” approaches [28], [29] where they attempt to find the location in time where inter-arrival 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 [22] 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 inter-packet

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 inter-packet times are i.i.d.

REFERENCES

- [1] V. Anantharam and S. Verdú, “Bits through queues,” *IEEE Trans. on Information Theory*, vol. 42, no. 1, pp. 4–18, 1996.
- [2] T. Coleman and N. Kiyavash, “Sparse graph codes and practical decoding algorithms for communicating over packet timings in networks,” *Conference on Information Sciences and Systems (2008)*, pp. 447–452, 2008.
- [3] “WARP: Wireless Open Access Research Platform,” 2008, <http://warp.rice.edu/trac/>.
- [4] “Defense Science Board (DSB) study on high performance microchip supply,” 2005, http://www.acq.osd.mil/dsb/reports/2005-02-HPMS_Report_Final.pdf.
- [5] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar, “Trojan detection using ic fingerprinting,” in *IEEE Symposium on Security and Privacy*, 2007, pp. 296–310.
- [6] Y. Alkabani and F. Koushanfar, “Active hardware metering for intellectual property protection and security,” in *USENIX Security*, 2007, pp. 291–306.
- [7] S. Adee, “The hunt for the kill switch,” May 2008, <http://www.spectrum.ieee.org/may08/6171>.
- [8] S. King, J. Tucek, A. Cozzie, C. Grier, W. Jiang, and Y. Zhou., “Designing and implementing malicious hardware,” in *USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET)*, 2008.
- [9] B. Lampson, “A note on the confinement problem,” *Communications of ACM*, vol. 16, 1973.
- [10] “Trusted computer system evaluation,” U.S. Department of Defense, Tech. Rep. DOD 5200.28-STD, 1985.
- [11] A. Wagner and V. Anantharam, “Information theory of covert timing channels,” in *NATO/ASI Workshop on Network Security and Intrusion Detection*, 2005.
- [12] M. Kang, I. Moskowitz, and D. Lee, “A network pump,” *IEEE Trans. on Software Engineering*, vol. 22, no. 5, pp. 329–338, 1996.
- [13] V. Anantharam and S. Verdú, “Reflections on the 1998 information theory society paper award: Bits through queues,” *IEEE Information Theory Society Newsletter*, pp. 100–102, December 1999.
- [14] S. Servetto and M. Vetterli, “Communication using phantom covert channels in the internet,” in *IEEE International Symposium on Information Theory*, 2004, p. 229.
- [15] X. Luo, E. W. Chan, and R. K. C. Chang, “Covert communications in TCP burstiness,” Hong Kong Polytechnic University, preprint, 2007.
- [16] M. Kang and I. Moskowitz, “A pump for rapid, reliable, secure communication,” in *ACM conference on Computer and communications security (CCS)*, 1993, pp. 119–129.
- [17] J. Giles and B. Hajek, “An information-theoretic and game-theoretic study of timing channels,” *IEEE Trans. on Information Theory*, vol. 48, no. 9, pp. 2455–2477, 2002.
- [18] S. Cabuk, C. Brodley, and C. Shields, “IP covert timing channels: Design and detection,” in *ACM conference on Computer and Communications Security (CCS)*, 2004, pp. 178–187.
- [19] 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.
- [20] “IEEE 802.11 LAN/MAN Wireless LANS Standard,” 2007, <http://standards.ieee.org/getieee802/802.11.html>.

- [21] R. Sundaresan and S. Verdú, “Robust decoding for timing channels,” *IEEE Trans. on Information Theory*, vol. 46, no. 2, pp. 405–419, 2000.
- [22] R. G. Gallager, *Information Theory and Reliable Communication*. New York: John Wiley & Sons, 1968.
- [23] A. Bennatan and D. Burshtein, “Design and analysis of nonbinary LDPC codes for arbitrary discrete-memoryless channels,” *IEEE Trans. on Information Theory*, vol. 52, no. 2, pp. 549–583, 2006.
- [24] C. Fragouloi, R. D. Wesel, D. Sommer, and G. Fettweis, “Turbo codes with nonuniform QAM constellations,” in *IEEE International Conference on Communication*, 2001, pp. 70–73.
- [25] F.-W. Sun and H. van Tilborg, “Approaching capacity by equiprobable signaling on the Gaussian channel,” *IEEE Trans. on Information Theory*, vol. 39, no. 5, pp. 1714–1716, 1993.
- [26] J. Ziv, “On universal quantization,” *IEEE Trans. on Information Theory*, vol. 31, no. 3, pp. 344–347, 1985.
- [27] T. Richardson and R. Urbanke, *Modern coding theory*. Cambridge University Press.
- [28] A. Tartakovsky and V. Veeravalli, “Asymptotic analysis of Bayesian quickest change detection procedures,” *IEEE International Symposium on Information Theory*, 2002.
- [29] A. Tartakovsky, B. Rozovskii, R. Blažek, and H. Kim, “Detection of intrusions in information systems by sequential change-point methods,” *Statistical Methodology*, vol. 3, no. 3, pp. 252–293, 2006.