Covert Communication Channels Based On Hardware Trojans: Open-Source Dataset and AI-Based Detection

Alán Rodrigo Díaz Rizo, Abdelrahman Emad Abdelazim, Hassan Aboushady, Haralampos-G. Stratigopoulos

Covert Communication Channels Based On Hardware Trojans: Open-Source Dataset and AI-Based Detection

Alán Rodrigo Díaz-Rizo, Abdelrahman Emad Abdelazim, Hassan Aboushady and Haralampos-G. Stratigopoulos
Sorbonne Université, CNRS, LIP6, Paris, France

***Abstract*—The threat of Hardware Trojan-based Covert Channels (HT-CCs) presents a significant challenge to the security of wireless communications. In this work, we generate in hardware and make open-source a dataset for various HT-CC scenarios. The dataset represents transmissions from a HT-infected RF transceiver hiding a CC that leaks information. It encompasses a wide range of signal impairments, noise levels, and HT insertions, facilitating a robust evaluation of HT-CC attack models and defenses. We also propose a deep learning-based HT-CC detection defense that achieves excellent accuracy on the dataset. It is an one fit all solution that circumvents the cost of integrating several distinct defenses to deal with all known HT-CC scenarios.**

I. Introduction

The horizontal business model of the Integrated Circuit (IC) supply chain creates security vulnerabilities that can be exploited by a knowledgeable adversary to perform an attack. A preoccupying attack is Hardware Trojans (HTs) defined as a malicious modification of the hardware performed by an adversary [1].

For wireless ICs, a HT may be embedded inside the RF transmitter to implement a Covert Channel (CC), referred to as HT-based Covert Channel (HT-CC). A CC refers to a communication channel that is hidden within a legitimate signal transmission and is designed to operate stealthily and secretly. Fig. 1 shows the threat model of a HT-CC. The CC reveals sensitive information from the transmitter, a.k.a. Alice, such as a cipher key. The information is leaked to an eavesdropper's receiver, a.k.a. Eve, without the nominal receiver, a.k.a. Bob, realizing it. Numerous works have showcased implementations of HT-CCs and have proposed defense strategies to detect the CC either at time zero using testing or on-line at run-time [2]–[14]. These works will be analyzed in more detail in Section II. In this work, we make the following contributions:

- We generated a *dataset of RF transmissions carrying a CC*, originating from a HT-infected transmitter. We included four main HTs found in the literature. The dataset is generated on hardware using the Software Defined Radio (SDR) bladeRF board from Nuand. For the first time we make such as dataset open-source and publicly available.
- We propose a novel *deep learning-based defense* that detects HT-CCs at run-time and, in addition, it is capable of pinpointing the underlying HT mechanism.

It should be noted that, herein, we consider only HT-CCs in the context of wireless technologies. There also exist HT-CCs that induce physical side-channels to convey secret information in short range [15]. Moreover, there exist HTs for analog ICs, applicable to RF transceivers as well, that target as payload performance degradation and denial-of-service and not information leakage via a CC [16].

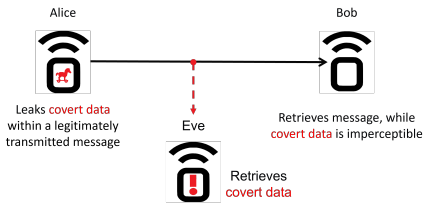


Fig. 1: Threat model of a HT-CC.

The rest of the article is structured as follows. In Section II, we discuss the prior art on HT-CCs. In Section III, we describe the specific HT-CC attacks considered for the dataset and their implementation details. In Section IV, we describe the dataset acquisition experiment. In Section V, we describe the proposed deep learning-based defense. In Section VI, we demonstrate the defense using the generated dataset. Section VII concludes this article.

II. Related work on HT-CCs

Table I provides a concise summary of existing HT-CC attack models and corresponding defenses. In general, a HT-CC attack works by driving the bits of the stolen information to the HT mechanism, which then hides them inside the legitimate transmission into a CC. The HT mechanism can be inserted into the Medium Access Control (MAC) protocol [2], into the digital baseband physical (PHY) layer [3]–[9], or its payload mechanism can be partially applied to the Analog Front-End (AFE) [10]–[14]. The four HT-CC attack models that are included in the generated dataset are indicated with a check mark in the last column of Table I and will be described in detail in Section III.

The aforementioned works also examine resilience to various defenses, finding in most cases a working defense, as shown in the third column of Table I. All these defenses operate at post-silicon during test time or run-time and can be classified as standard or ad-hoc. The former look whether the IC complies with the communication protocol and the latter look specifically for HT activity. Standard defenses include measuring Signal-to-Noise Ratio (SNR), Error Vector Magnitude (EVM), and Bit Error Rate (BER), examining compliance

TABLE I: HT-CC attack models and defenses.

Ref.	Attack model	Defense	Present in the dataset
[2]	Modifies the Medium Access Control (MAC) layer Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) protocol to leak data into the timings of the transmitted packet sequence.	Evades statistical tests that detect covert timing channels. No other defense is studied.	
[3], [9]	Encodes leaked data on the I/Q mapping and hides the encoding by introducing imperfections to the transmitted signal.	Certain tests, such as Error Vector Magnitude (EVM), show a distinguishing behavior compared to HT-free operation.	✓
[4]-1	Leaks data by introducing an additional phase shift into all Short Training Sequence (STS) symbols of the preamble.	Analysis of the preamble constellation.	✓
[4]-2	Leaks data by introducing artificial Carrier Frequency Offset (CFO) into each Orthogonal Frequency-Division Multiplexing (OFDM) symbol.	Analysis of CFO changes over time.	
[4]-3, [5]	Leaks data in extra camouflage subcarriers added to the OFDM signal.	Decode the signal field to determine if the number of subcarriers is correct.	
[4]-4, [6]	Leaks data into replaced parts of the OFDM Cyclic Prefix (CP).	Compare the last 16 samples of an OFDM symbol with its CP; Spectral analysis.	
[7]	Leaks data by substituting some legitimate data in the Forward Error Correction (FEC) block.	Channel noise profiling.	
[8]	Leaks data through amplitude modulation (denoted by α) of some subcarriers in the STS of the preamble.	Evades any known defense for α <15%.	✓
[10]–[12]	Leaks data by modulating the amplitude and/or frequency of the transmitted signal.	Statistical Side-Channel Fingerprinting (SSCF) [11], [12]; Adaptive Channel Estimation (ACE) [12]; Use hardware dithering as a prevention mechanism [17]; Checking compliance of an invariant side-channel fingerprint [18].	✓
[13]	Leaks data using spread spectrum techniques.	Spectral analysis.	
[14]	Leaks data into controlled artificial RF impairments.	No defenses are studied.	

with the spectral mask specifications, and analyzing I/Q constellation diagrams. Ad-hoc defenses include Statistical Side-Channel Fingerprinting (SSCF) [11], [12], Adaptive Channel Estimation (ACE) [12], channel noise profiling [7], and checking the compliance of invariant side-channel fingerprints [18]. None of these defenses is systematically performed at run-time on wireless ICs since it requires adding dedicated hardware modules inside the wireless IC, which increases area and power consumption.

In addition, proactive measures can be taken during the design phase aiming at preventing the attack. Techniques that can be used include encrypting the PHY layer to prevent man-in-the-middle attacks [19] and locking the functionality of the RF transceiver, rendering it key-dependent, so as to make it difficult for the attacker to insert the HT [20], [21]. Another strategy is to challenge the operational principles of HTs, aiming to neutralize their impact. An example here is the hardware dithering technique proposed in [17].

Finally, there exist many generally applicable techniques aiming at exposing HT mechanisms hidden into a design [1], i.e., at pre-silicon using testing, formal verification, and information flow tracking and at post-silicon using reverse engineering, etc.

III. Implemented HT-CC attacks description

Herein, we describe in detail the four different HT-CCs that constitute the dataset. In all cases, the CC is implemented in IEEE 802.11, a.k.a. WiFi, systems. The CC is composed of a *covert* message hidden within the *cover*, i.e., nominal or legitimate, signal of the WiFi network.

A. Cover signal

The cover signal consists of Orthogonal Frequency-Division Multiplexing (OFDM) IEEE 802.11 frames. The Physical layer Protocol Data Unit (PPDU) frame format of an OFDM IEEE 802.11 transmission consists of several OFDM symbols. These symbols are divided into 3 parts, namely preamble (a.k.a. SYNC), header (a.k.a. SIGNAL), and payload (a.k.a. DATA). The preamble section is composed of two different training symbol sequences, namely a Short Training Sequence (STS) and a Long Training Sequence (LTS). Fig. 2 shows the PPDU of an IEEE 802.11 transmission as defined in the IEEE 802.11 standard [22].

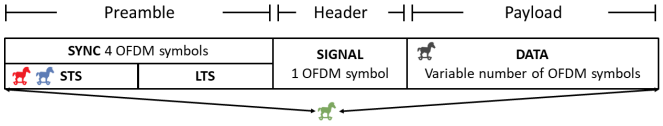


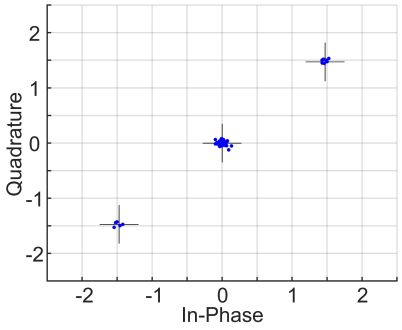
Fig. 2: PPDU frame format of an OFDM IEEE 802.11, a.k.a. WiFi, transmission. The red, blue, black, and green Trojan horses depict the location of the covert message within the cover signal for HT1-CC, HT2-CC, HT3-CC, and HT4-CC, respectively.

B. Covert message generation

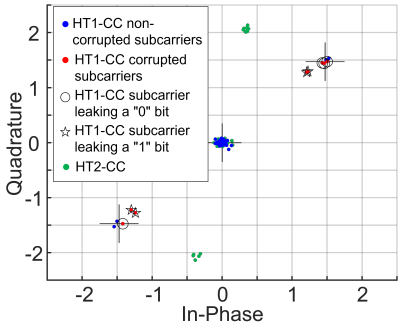
1) A first approach is to hide the HT inside the PHY layer and make it modulate the STS of the preamble of the transmitted frame, as shown with the Trojan horses placed in the STS field of the frame in Fig. 2. The frequency-domain representation of the STS, denoted by STS_F, is composed of 64 complex values, i.e., having real (I) and imaginary (Q) components, also called subcarriers or frequency bins, from the alphabet $\{-1.472 - 1.472j, 0, 1.472 + 1.472j\}$. The 64 subcarriers are indexed from -32 to 31, and there are 12 non-zero subcarriers, as shown in Table II. For instance, Fig. 3a shows a constellation diagram of the nominal STS_F where the transmitted values, plotted as blue dots, fall near the expected constellation points, i.e., $\{-1.472-1.472j, 0, 1.472+1.472j\}$, indicated by black plus signs. The time-domain representation of the STS, denoted by STS_t, is derived by performing an Inverse Fast Fourier Transform (IFFT) on the STS_F. The PPDU STS is composed of two OFDM symbols and it is obtained

TABLE II: Frequency-domain representation of the STS (STS_F) within the preamble of a WiFi frame.

STS_F index (k)	Complex-value (I,Q)
-24, -16, -4, 12, 16, 20, 24	1.472+1.472j
-20, -12, -8, 4, 8	-1.472-1.472j
others	0



(a) Nominal STS constellation diagram according to the standard [22].



(b) CC-infected STS constellation diagram in the case of HT1-CC [8] and HT2-CC [4].

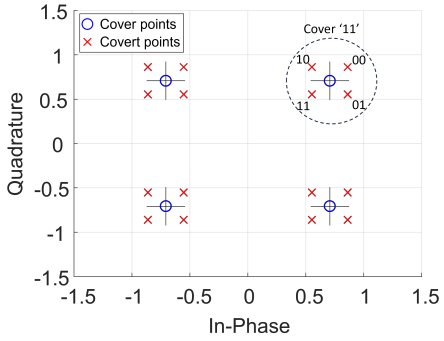
Fig. 3: Covert message hidden in the preamble. Additive White Gaussian Noise (AWGN) has been added to the signals such that the subcarriers are not superposed for illustration purposes.

by concatenating two and a half STS_t. For simplicity, in the rest of the paper we will refer to STS_F as STS.

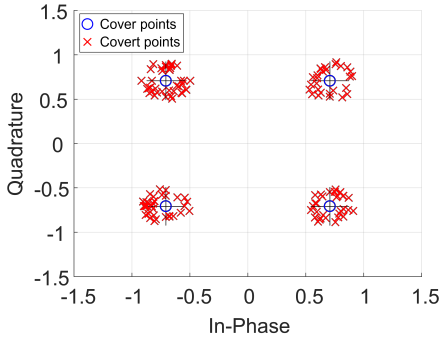
We consider two implementations of this attack type, referred to as HT1-CC and HT2-CC, shown with the red and blue Trojan horses in Fig. 2, respectively.

a) HT1-CC [8]: The data are leaked through minute amplitude modulations of non-zero subcarriers of the STS. These subcarriers are called *corrupted subcarriers*. In particular, the amplitude of a corrupted subcarrier is reduced below a threshold level by multiplying it with $\alpha < 1$ when the leaked bit is '1'. Otherwise, the amplitude is preserved for a leaked bit '0'. In our implementation, we chose an amplitude modulation of 10%, i.e., $\alpha = 0.9$, and there are 8 corrupted subcarriers per STS, thus the CC leaks 8-bits per transmitted frame. Fig. 3b shows an example constellation diagram of the CC-infected STS where the leaked byte is composed of four '0' bits and four '1' bits.

b) HT2-CC [4]: The leaked data is encoded through a controlled counter-clockwise phase shift in all the STS subcarriers with respect to the CC-free constellation points. The number of possible phase shifts varies depending on the number of bits intended to be encoded and the shift amount in binary corresponds to the leaked byte. The CC implemented



(a) Possible initial displacement of the cover point to create the covert point.



(b) Possible final positions of the covert point.

Fig. 4: "Dirty" constellations.

in the dataset leaks 8 covert bits per STS or per transmitted frame, resulting in 256 possible phase shifts. Fig. 3b shows an example constellation diagram.

2) HT3-CC [3]: A second approach, called "dirty" constellations, is to hide the HT inside the PHY layer and make it modulate the covert message within the PPDU DATA field of the cover signal, as shown in Fig. 2 by a black Trojan horse. This approach takes advantage of hardware impairments and noisy channel conditions. More specifically, the PPDU DATA is composed of a variable number of OFDM symbols depending on the frame type and payload length. In our implementation, the OFDM symbols comprise 48 QPSK modulated subcarriers. Let us consider the QPSK constellation diagram in Fig. 4a. The attack can leak two bits per QPSK subcarrier. It chooses a subcarrier and displaces its constellation point according to the two bits being leaked. Essentially, using a QPSK cover point as origin there are four QPSK covert points. For example, if we want to leak bits '01' we can displace the upper right point (i.e., cover point '11') to the bottom right position (i.e., covert point '01'), as shown in Fig. 4a. Then, several operations are applied to the covert point to reduce the probability of detecting the CC: (a) the covert point approaches symmetrically around the origin up to a distance equal to that of the 64-QAM; (b) its position is randomized with a Gaussian distribution within a dispersion radius $r = \sqrt{2/42}$; (c) it is rotated within r with a monotonically increasing angle θ with steps of 15^o. Moreover, to avoid detecting HT activity, even when the HT is inactive, intentional distortions are added to the transmitted signal increasing the average EVM up to 10 dB within the allowed limits of the IEEE 802.11 standard.

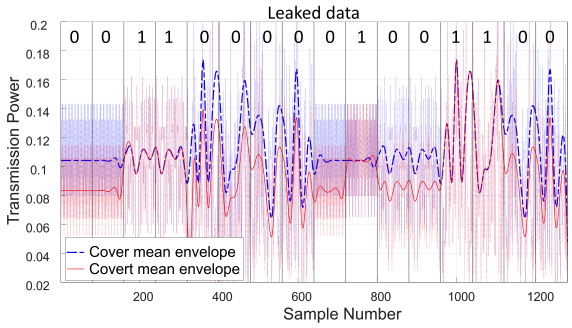


Fig. 5: Mean envelope of the CC-free signal and the CC-infected signal with the HT4-CC attack. The signal mean envelope is reduced for a leaked bit '0', while it remains unchanged for a leaked bit '1'.

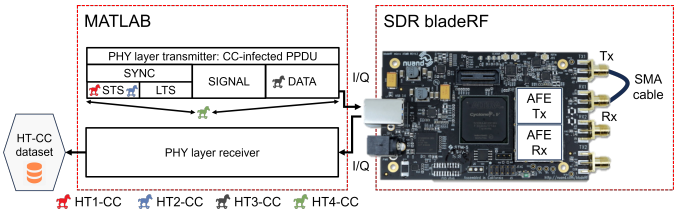


Fig. 6: Experimental setup for dataset generation.

Therefore, it is worth noticing that a device infected with HT3-CC will always have a higher BER compared to a CC-free device. Fig. 4b depicts the outcome after these operations. In our implementation, we use 5 "dirty" subcarriers per OFDM symbol, thus 10 bits are leaked per OFDM symbol.

3) HT4-CC [12]: The fourth considered attack model inserts the HT into the AFE leaking data through minute modifications in the amplitude of the transmitted signal. In this way, the covert message is spread across the transmitted frame, as shown with the green color Trojan horse in Fig. 2. More specifically, RF transmitters use multiple programmable Variable Gain Amplifiers (VGAs) in the transmission chain to satisfy linearity and achieve desired performance specifications. A Serial Peripheral Interface (SPI) controls these VGAs through the PHY layer. The attack leaks secret information by systematically modifying the gain of the VGAs, creating minute variations in the transmit power, according to the leaked bits. In our implementation, 8 bits are leaked per transmitted frame, where the amplitude is reduced by 20% for a leaked bit '0', while it remains unchanged for a leaked bit '1', as illustrated in Fig. 5.

IV. HT-CC Dataset Acquisition

The dataset is generated using an SDR bladeRF board from Nuand. The hardware acquisitions are performed using a single board, by connecting the RF transmitter with the RF receiver in loopback mode via an SMA cable, as shown in Fig. 6. The PHY layer implemented in MATLAB prepares the PPDU frames of the transmitted signal, shown in Fig. 2, then the frames are transmitted using the RF transmitter of the board. While a CC-free transmission is composed of PPDU frames as defined by the IEEE 802.11 standard [22], a CC-infected transmission has a CC embedded into the PPDU frames leaking secret information as described in Section III.

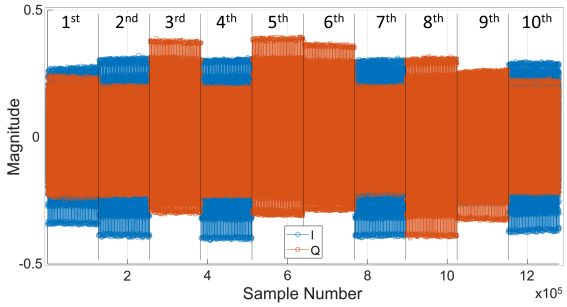


Fig. 7: Ten concatenated acquisitions forming a multi-acquisition of the CC-free signal.

The dataset is organized into 5 parts corresponding to the 4 HT-CCs described in Section III, denoted by HTX-CC, X=$\{1, \cdots, 4\}$, and the CC-free signal. Each part consists of 16 elements representing 2 acquisitions with 8 different SNR values ranging from 1 to 29 dB with a step of 4 dB. Each element consists of 2000 fixed-length OFDM IEEE 802.11 frames, where each frame contains 2×640 real-value samples corresponding to the two I/Q branches.

The leaked message is common to all attack models. It is formed of random binary data, it has a length of 11 bytes, and it is repeated continuously.

As the received signal passes through different digital and analog hardware components, every received sample is affected by hardware impairments that impact the signal at baseband and RF, e.g., flicker noise, quantification error, DC offset (DCO), IQ imbalance (IQI), carrier frequency offset (CFO), phase noise, and jitter. Moreover, although the SMA connection reduces channel impairments, the test environment is not noise-free, thus the SMA cable loopback connection is affected by various noise sources, such as thermal noise or interference from other signals. Such impairments in the dataset signals are not compensated for after acquisition. However, a single long acquisition of 2000 frames has similar hardware and RF impairments conditions for all received frames. A richer dataset should take into account diverse hardware and RF impairments conditions, as well as hardware temperature. Hence, we performed ten smaller acquisitions of 200 frames and concatenated them to form a single 2000 frames multi-acquisition dataset element. In this way, each dataset element comprises signals with differently distributed non-idealities. As an example, Fig. 7 shows the concatenation of the 10 acquisitions of I/Q samples for the CC-free signal in the dataset.

The dataset is made open-source and is downloadable from: https://github.com/alandr918/Hardware-Trojan-Covert-Channel-dataset.

V. AI-based HT-CC detection

As shown in Table I, for most HT-CC attacks specific detection countermeasures have shown to work. Most of these countermeasures are too costly to be implemented at run-time within the nominal receiver or they do not operate on-line requiring data collection first, which results in detection

latency. In any case, the defender is forced to combine simultaneously many of these countermeasures so as to provide maximum security against the multitude of HT attack models and their ever-increasing sophistication level, which would rise exponentially the cost of the defense, eventually making the cost prohibitive.

In this work, we propose a novel run-time Machine Learning (ML)-based defense against HT-CCs. SSCF [11], [12] is the only defense among those mentioned in Section II that employs ML. SSCF consists in training an one-class classifier, e.g., a Support Vector Machine (SVM), in a feature space composed of parametric measurements referred to as side-channel fingerprints, e.g., transmitted power. The classifier is trained using data from golden HT-free devices only. The allocated classification boundary encloses the footprints of the HT-free devices, anticipating that the footprints of HT-infected devices will lie outside this boundary, distinguishing in this way HT-free from HT-infected operation. In general, due to process variations within the transmitter hardware, noise, and channel effects, it is extremely challenging to extract a feature space wherein the overlapping between the HT-free and HT-infected classes is small enough so as to achieve a good classification accuracy. While SSCF has shown to be efficient for HT-CCs systematically distorting the transmission power [10], [11], using transmission power as parametric measurement has failed to detect other HT-CCs that do not violate any wireless protocol or circuit specifications, such as the HT1-CC [8] or HT4-CC [12].

As a first step, we trained a one-class SVM using as features raw received data instead of relying on parametric measurements. The SVM uses the Radial Basis Function (RBF) kernel and the unsupervised learning employs the CC-free frames in our dataset. For each frame the I/Q samples are concatenated to make a 1280-dimensional input feature. Principal Component Analysis (PCA) is applied for feature dimensionality reduction. A number of principal components is kept such that 90% of data variation is explained. This results in a 253-dimensional input. The frames are randomly divided into 70% for training and 30% for testing. The result is that SVM shows poor prediction accuracy on this binary classification problem that never exceeds 75% for a given SNR value with an average of 66.4% across all SNR values.

To this end, we conclude that deep learning is rather required if we want to detect the CC from raw data. We propose to train a Deep Neural Network (DNN) classifier that takes as input directly the incoming received frames. The frame is encoded as a 2×640 "image". In our experiment, we employ a Convolutional Neural Network (CNN), which is the most popular choice for image classification problems, and we leverage the generated dataset to perform the CNN training. We formulate two classification problems: (a) binary classification, i.e, the classifier distinguishes CC-infected from CC-free transmissions, where the HTX-CCs are combined into a single CC-infected class; and (b) multi-class classification, where the classifier learns a more challenging task to predict 1 out of 5 classes, namely CC-free and HTX-CC, X=$\{1, \cdots, 4\}$,

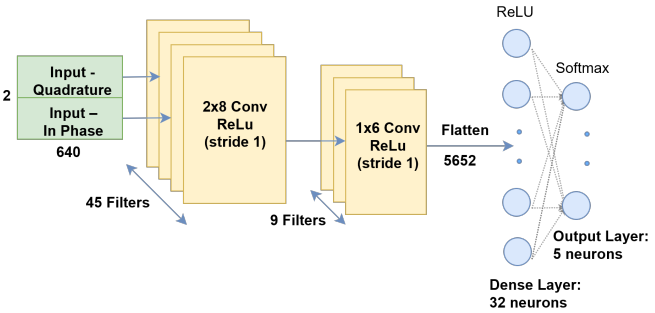


Fig. 8: CNN architecture.

i.e., the classifier in this case not only distinguishes CC-free from CC-infected transmissions, but as an auxiliary benefit it pinpoints the underlying HT mechanism within the infected transmitter.

All the considered HT-CCs are always on. In this case, Bob in Fig. 1 can use its transmitter to re-transmit a few frames to the cloud where the CNN inference will take place. If a CC is detected, then the communication is halted and Bob warns the inconspicuous Alice that it has been compromised and that it is leaking sensitive data to Eve. Note that the defense can also be placed inside Alice, but a knowledgeable attacker can manipulate the CNN prediction to suppress the alert. If the HT-CC is enabled within Alice for a short time period or it is switched-on periodically, then it is not a viable option for Bob to re-transmit every frame to the cloud as the power consumption penalty will be non-negligible. Bob can send periodically frames to the cloud to check for CC presence, but this is likely to result in detection latency or missing the CC if it is not permanently activated. In this scenario, an AI hardware accelerator needs to be eventually included into the RF transceiver chip to perform the CNN inference in real-time to detect the CC whenever it is activated.

VI. Results on Dataset

The WiFi protocol requires a minimum SNR of 10 dB for QPSK-modulated signals. While this technically allows for a connection, it results in poor quality and slow performance. Dropping below this threshold makes the connection unsuitable. Ideally, a minimum SNR of 20 dB to 25 dB is recommended for robust and dependable communication in WiFi [22]. While the SNR range of interest is above 20 dB, we use SNR values as low as 1 dB to investigate the capability of the CNN classifier to detect the HT-CC in unfavorable conditions where the communication faces a high BER.

We started with a classical CNN architecture and we followed a trial-and-error approach to optimize its size, i.e., number of layers, number of feature maps in convolutional layers, and number of neurons in fully-connected layers, without jeopardizing the classification accuracy. The resultant CNN architecture having a total of 184265 synaptic weights is shown in Fig. 8. Training was performed on Kaggle using the Keras framework and the Adam optimizer with a learning rate of 0.001.

Fig. 9 shows the CNN classification accuracy as a function of the SNR for the binary and multi-class classification

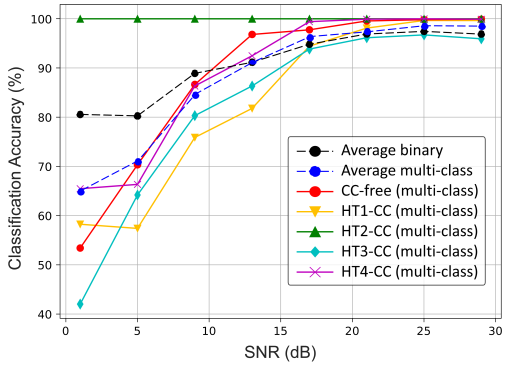


Fig. 9: CNN classification accuracy as a function of SNR.

problems. For the multi-class problem, it also shows the per-class accuracy for the 5 classes, namely CC-free and HTX-CC, X=$\{1, \cdots, 4\}$. The trend is that the accuracy increases with SNR. This monotonic behavior is explained by the observation that the stronger the signal is, the easier it becomes for the CNN to spot the CC within the signal. The CNN shows excellent accuracy for both classification problems in the range of interest of SNR>20 dB. It demonstrates over 95% accuracy for detecting the CC-free and any HT-CC transmission, with HT2-CC being the easiest to detect and HT3-CC the most challenging to detect. The average accuracy for the multi-class problem reaches 99% for SNR>25 dB which is required for practical communication. Even for the lowest SNR of 1 dB, where the CC data cannot be recovered reliably by the attacker, the CNN still performs surprisingly well achieving an accuracy of over 80% for the binary problem. For the multi-class problem, the accuracy reaches 90% for SNR>10 dB. Regarding inter-class misclassification, in the case of SNR=25dB, there are no false positives, that is CC-free transmissions are always correctly identified, while HT1-CC, HT3-CC, and HT4-CC are misclassified as CC-free with probabilities 0.5%, 3.1%, and 0.1%, respectively. Misclassifications between HTX-CCs are less probable. The only cases observed are HT3-CC being misclassified into HT4-CC and HT4-CC into HT1-CC, both with probability 0.1%. These results prove that the proposed AI-based defense can accurately not only detect a CC-infected communication, but also classify the underlying HT mechanism within the infected transmitter that enables the CC.

VII. Conclusion

We generated on hardware and made publicly available a dataset of RF transmissions carrying a CC, originating from a RF transmitter that is infected with four different HT mechanisms. The dataset is augmented with CC-free RF transmissions and can be readily used to evaluate the effectiveness of HT-CC detection defenses. We also proposed a novel single run-time defense which consists of a CNN embedded on the receiver side that monitors the raw transmission signal and detects the CC and, in addition, it predicts the type of HT mechanism inside the transmitter. The CNN is shown to achieve 99% detection accuracy on the dataset for the SNR range of interest.

References

[1] K. Xiao, D. Forte, Y. Jin, R. Karri, S. Bhunia, and M. Tehranipoor, "Hardware Trojans: lessons learned after one decade of research," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 22, no. 1, pp. 6:1–6:23, Dec. 2016.

[2] N. Kiyavash, F. Koushanfar, T. P. Coleman, and M. Rodrigues, "A timing channel spyware for the CSMA/CA protocol," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 3, pp. 477–487, Mar. 2013.

[3] A. Dutta, D. Saha, D. Grunwald, and D. Sicker, "Secret agent radio: Covert communication through dirty constellations," in *Information Hiding*, M. Kirchner and D. Ghosal, Eds., Berlin, Heidelberg, 2013, pp. 160–175, Springer Berlin Heidelberg.

[4] J. Classen, M. Schulz, and M. Hollick, "Practical covert channels for WiFi systems," in *Proc. IEEE Conf. Commun. Netw. Secur. (CNS)*, Sep. 2015, pp. 209–217.

[5] Z. Hijaz and V. S. Frost, "Exploiting OFDM systems for covert communication," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, Oct./Nov. 2010, pp. 2149–2155.

[6] S. Grabski and K. Szczypiorski, "Steganography in OFDM symbols of fast IEEE 802.11n networks," in *Proc. IEEE Secur. Priv. Workshops*, May 2013, pp. 158–164.

[7] K. S. Subraman, A. Antonopoulos, A. A. Abotabl, A. Nosratinia, and Y. Makris, "Demonstrating and mitigating the risk of an FEC-based hardware trojan in wireless networks," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 10, pp. 2720–2734, Feb. 2019.

[8] A. R. Díaz-Rizo, H. Aboushady, and H.-G. Stratigopoulos, "Leaking wireless ICs via hardware trojan-infected synchronization," *IEEE Trans. Dependable Secure Comput.*, vol. 20, no. 5, pp. 3845–3859, Sept. 2023.

[9] K. Grzesiak, Z. Piotrowski, and J. Kelner, "A wireless covert channel based on dirty constellation with phase drift," *Electronics*, vol. 10, no. 6, Mar. 2021.

[10] 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.

[11] Y. Liu, Y. Jin, A. Nosratinia, and Y. Makris, "Silicon demonstration of hardware trojan design and detection in wireless cryptographic ICs," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 4, pp. 1506–1519, Apr. 2017.

[12] K. S. Subramani, N. Helal, A. Antonopoulos, A. Nosratinia, and Y. Makris, "Amplitude-modulating analog/RF hardware trojans in wireless networks: Risks and remedies," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 3497–3510, Apr. 2020.

[13] S. Chang, G. Bhat, U. Ogras, B. Bakkaloglu, and S. Ozev, "Detection mechanisms for unauthorized wireless transmissions," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 23, no. 6, pp. 70:1–70:21, Nov. 2018.

[14] K. Sankhe *et al.*, "Impairment shift keying: Covert signaling by deep learning of controlled radio imperfections," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, Nov. 2019, pp. 598–603.

[15] L. Lin, T. Güneysu M. Kasper, C. Paar, and W. Burleson, *Trojan Side-Channels: Lightweight Hardware Trojans through Side-Channel Engineering*, Berlin, Germany: Springer, 2009.

[16] M. Elshamy *et al.*, "Digital-to-analog hardware Trojan attacks," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 69, no. 2, pp. 573–586, Feb. 2022.

[17] C. Kapatsori, Y. Liu, A. Antonopoulos, and Y. Makris, "Hardware dithering: A run-time method for trojan neutralization in wireless cryptographic ICs," in *Proc. IEEE Int. Test Conf. (ITC)*, Oct./Nov. 2018.

[18] Y. Liu, G. Volanis, K. Huang, and Y. Makris, "Concurrent hardware trojan detection in wireless cryptographic ICs," in *Proc. IEEE Int. Test Conf. (ITC)*, Oct. 2015.

[19] J. Chacko *et al.*, "Physical gate based preamble obfuscation for securing wireless communication," in *Proc. Int. Conf. Comput. Netw. Commun. (ICNC)*, Jan. 2017, pp. 293–297.

[20] A. R. Díaz-Rizo, J. Leonhard, H. Aboushady, and H. Stratigopoulos, "RF transceiver security against piracy attacks," *IEEE Trans. Circuits Syst., II, Exp. Briefs*, vol. 69, no. 7, pp. 3169–3173, Jul. 2022.

[21] A. R. Díaz-Rizo, H. Aboushady, and H.-G. Stratigopoulos, "Anti-piracy design of RF transceivers," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 70, no. 1, pp. 492–505, Jan. 2023.

[22] IEEE, "IEEE standard for information technology—telecommunications and information exchange between systems local and metropolitan area networks—specific requirements - part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications," *IEEE Std 802.11-2016 (Revision of IEEE Std 802.11-2012)*, pp. 1–3534, 2016.