CAS-HtBase: a new database for the study of HTs at the pre-silicon stage of ASICs

Konstantinos G Liakos
Department of Electrical and Computer Engineering
School of Engineering, University of Thessaly
Volos, Greece
kliakos@e-ce.uth.gr

Fotis C Plessas
Department of Electrical and Computer Engineering
School of Engineering, University of Thessaly
Volos, Greece
fplessas@e-ce.uth.gr

***Abstract*—Hardware Trojan (HT) consists a chip-level viruses which aim to leak encrypted information or degrade the performance of the infected device. They are a modification to the original design of a circuit and consist of two components, trigger, and payload. The trigger is the mechanism that calls the payload under rare conditions. The payload mechanism is responsible for the type of attack the infected circuit will receive. HTs can be inserted into any phase of the Application-Specific Integrated Circuits (ASICs) production chain. They can stay stealthy and be undetected. HTs viruses are a crucial issue in the field of electronics, with the potential to become an outbreak in the next years. A major problem is that, at the academic and research level, there is a general lack of data, a lack of uninfected benchmark circuits, a lack of the small type of uninfected and infected circuits, and a large imbalance in the amount of data between uninfected and infected circuits. We used and designed these limited benchmark circuits for the Gate Level Netlist (GLN) phase with a professional tool and extracted area, power, and time analysis features. Through these features, we created a synthetic data creation tool known as GAINESIS through which we created our CAS-HtBase. CAS-HtBase aims to provide the researchers with a new HT database based on GLN analysis exclusively for ASICs. The format of CAS-HtBase allows the development of Machine Learning (ML) models without conversions to the database.**

Keywords—database, application-specific integrated circuits, hardware trojan, gate-level netlist, machine learning models

I. INTRODUCTION

Hardware Trojans (HTs) viruses are one of the most important problems in hardware security. They are associated with unwanted changes that affect an Integrated Circuit (IC). They degrade the system, lead to destruction or and leak encrypted information through that. HTs can be inserted at any phase of IC development and remain inactive until activated by a rare activation condition. These viruses are stealthy with a wide range of mechanisms and sizes. The need for smaller and more sophisticated circuits is the reason for the existence of these viruses. Specifically, the design companies to reduce manufacturing costs, outsource to third-party foundries the design of their circuits. There for corruption reasons, the original designs can be modified, with the insertion of HTs.

HTs consists of two mechanisms. An activation mechanism is known as a trigger and an effective mechanism is known as a payload (Fig. 1). The trigger is activated under rare conditions like events or signals and the payload circuitry starts the attack on the infected circuit. These mechanisms are categorized into two activation logics, combinational and sequential. In combinational logic, the activation of HT on the infected circuit is based on the simultaneous existence of a set of rare events. While on sequential logic HTs require a series of rare events to activate. The type of attacks can be grouped into two main categories, general-purpose processor and cryptographic engine attacks. General-purpose processor attacks via the kernel, memory, or at lower levels of the processor, aim to the degradation or destruction of the infected system. While the cryptographic engine attacks aim to leakage of encrypted information through the infected system.

As mentioned, HTs can be inserted at any phase of IC development and remain inactive until activated by a rare activation condition. Ideally, the specific viruses should be detected at each stage of the pre-silicon, verification, and post-silicon, the fabrication production phase of an IC. The disadvantage is the necessity of the golden model for the designed IC, in both phases. The information is not always obtainable, particularly for designs based on Intellectual Properties (IPs) provided by mediator manufacturers. HTs attacks can be categorized according to the stage and production phase of the circuit they target and infect. Specifically, they can be categorized for the pre-silicon stage in four categories, register transfer level (RTL), GLN, placement and routing (PNR), and graphic database system II (GDSII). While in fabrication and testing for the post-silicon stage (Fig. 2). Depending on the phase that the virus is designed to target, the attacker can have specific benefits. In pre-silicon attacks, the attacker through the virus can gain full access to design files and source code or hack computer-aided design tools and scripts to produce a modified IC representation without changing the source code. On the other hand, in post-silicon attacks which take place after tape-out of the circuit, the attacker can add or delete components of the original IC through reverse engineering, IC metering, and layout geometry modification.

The majority of studies dealing with the development of methodologies as countermeasures against HT focus on FPGA circuits in the post-silicon stage. There is limited information and published studies on ASICs and especially for the pre-silicon stage of them. Generally, ASICs are challenging due to the variety of design phases, especially in the pre-silicon stage. Also, ASICs need professional tools for the design of each phase.

Studies relating to the post-silicon stage, develop approaches as countermeasures based on side-channel analysis features like area, power, and time. These approaches use techniques based on SCA features to detect alterations of physical characteristics like area, time, and power caused by HTs. If the original SCA values of an IC differ, then the circuit is infected. This is because when an HTs is partially or fully activated, the original infected circuit shows more interrupting activity compared to the original normal circuit because consumes extra power and has extra space [1][2][3][4][5][6]. Other approaches consist the simulated methodologies such as logic testing techniques. These techniques generate tests to activate HTs and propagate

979-8-3503-9958-5/22/$31.00 ©2022 IEEE

the HTs payload to principal outputs for comparison with the golden circuit. The difficulty with these methods is coming up with effective assays to activate HTs [7][8][9]. A new trend is the development of ML models for the classification of HT-infected and HT-free circuits. Specifically, these types of approaches developed ML-based classifiers for the classification of HTs in different phases of ICs development. ML-based approaches can be used for pre-silicon [10][11][12] as well and post-silicon stage [13][14][15]. Furthermore, there are auxiliary approaches the purpose of which is to enhance the effectiveness of the detection techniques against HTs for the pre-silicon [16][17][18] or post-silicon stage [19][20].

To develop countermeasures against HTs, the academic and research community needs data, from the original circuits and their modifications with HT viruses. The majority of the research in the field of HTs utilizes the Trust-HUB public library [21][22]. Trust-Hub is a circuit design public library that contains HT-free and HT-infected benchmarks. Trust-HUB database has four major limitations, the benchmarks are designed for field-programmable gate arrays (FPGA) and not for ASIC. There is a general lack of data, a lack of uninfected benchmark circuits, and a lack of the small type of circuits. The majority of the benchmarks are large in size without diversity, which means they are easier to detect. But actually, most HT viruses are very small in size and difficult to detect.

In this paper, we aim to introduce our new circuit design public database named as CAS-HtBase. Our database consists of the area and power analysis features from designed circuits at the GLN phase of the pre-silicon stage of ASICs. It is the first database exclusively for ASIC and specifically for the GLN phase. For the development of our CAS-HtBase used 18 HT-free and 923 HT-infected benchmarks. Also, the CAS-HtBase structure is designed for direct use in the development of ML-based.

II. CAS-HTBASE METHODOLOGY

A. Scheme of our Methodology

The analysis pipeline, described in the following sections, can be divided into the following distinct steps: a collection of the benchmarks in Verilog format; design of the GLN phase of each benchmark with a professional circuit design tool; generation and extraction of area, power and time analysis features based on in-house scripts; examination and creation of the CAS-HtBase. An overview of the analysis pipeline is depicted in Fig. 3.

B. Benchmarks Collection

The first step in the creation of CAS-HtBase was the exploration of all the free databases and libraries and the collection of all available benchmarks. Our exploration was based on studies related to HTs viruses and they report the repositories they relied on to implement their research. Repositories such as ISCAS' 85, ISCAS' 89, and ITC' 99 were investigated. Most of them were not active as they were set up for older conferences and are no longer operational. Some others did not contain the benchmarks reported or their database and the majority of the benchmarks were blank. The

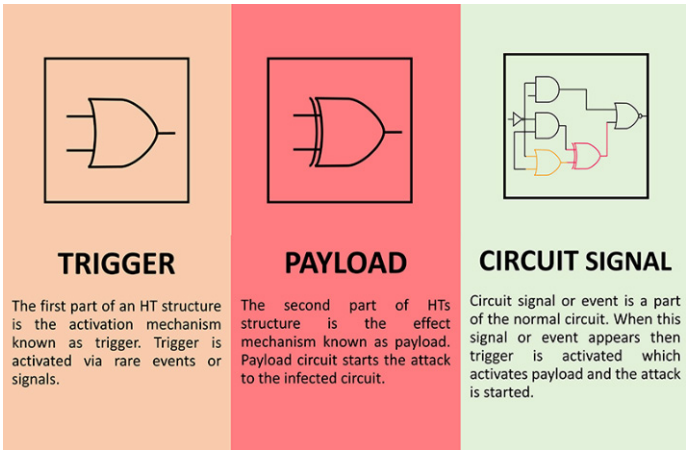

Fig. 1. Hardware trojan structure [23].

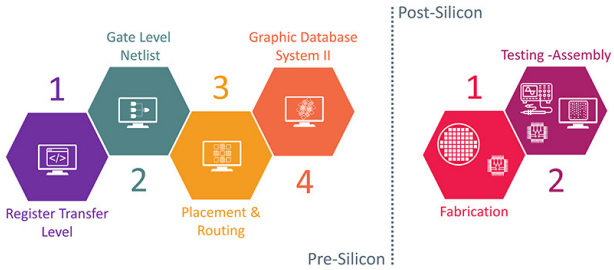

Fig. 2. Overview of ASIC production chains [23].

only operational repository was Trust-HUB, from which we collected all our samples. As mentioned, after the exploration of the Trust-HUB library we found four major limitations, there is a general lack of data, a lack of uninfected benchmark circuits, and a lack of small types of circuits, and the benchmarks are designed for FPGA and not for ASIC.

C. GLN Design

The majority of the studies are focused on FPGA circuits at the post-silicon stage. In the international bibliography, there is limited information from published studies, on the development of countermeasures against HTs on ASICs and especially for the pre-silicon stage. This is due to the complexity of ASICs. Specifically, ASICs have a variety of design phases especially, on the pre-silicon stage. Also, they need professional tools for the design of the phases. For the reasons set out above we focused on the study of ASICs and especially on the phases of the pre-silicon stage. We studied several phases of the pre-silicon stage of ASICs and we aimed at the GLN phase. The GLN phase consists of one of the most critical phases in circuit production chains. Because the designers utilize Verilog to compose netlists that describe the logical functionality of the circuit, based on logic gates and cells according to a specific cell library. These steps can be easily highjacked, making GLN the most vulnerable step in circuit production, in terms of HT infection susceptibility. We designed the GLN phase of the original and infected benchmarks via the Design Compiler NXT tool in combination with the FreePDK45nm free circuit library.

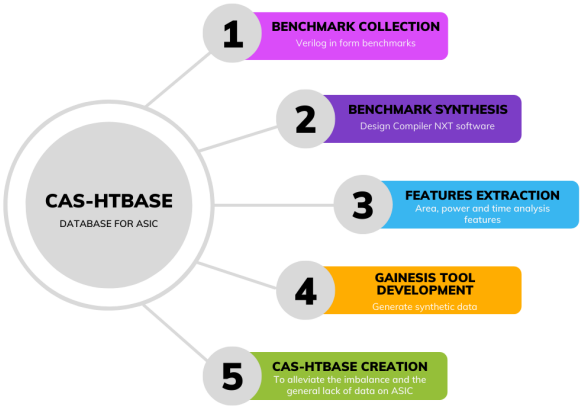

Fig. 3. Scheme of our CAS-HtBase methodology.

D. Feature Extraction

We designed in total 941 benchmarks, 18 original HT-free, and 923 modified HT-infected benchmarks. For each designed benchmark we created three files with information about the area, power, and time analysis of each benchmark. The elements inside the files were in a log form. Which made the elements unusable and unreadable. To solve these problems, we developed our in-house Python scripts and we parsed the essential information for each benchmark. For each benchmark in total were collected 51 features. Specifically, 13 areas, 37 power, and 1-time analysis features (Table I).

TABLE I. INITIAL 51 FEATURES

Analysis	Feature
Area	Number of ports
	Number of nets
	Number of cells
	Number of combinational cells
	Number of sequential cells
	Number of buf/inv
	Number of references
	Combinational area
	Buf/Inv area
	Non-combinational area
	Total cell area
Power	Cell Internal Power
	Net Switching Power
	Total Dynamic Power
	Cell Leakage Power
	Register Internal Power
	Register Switching Power
	Register Leakage Power
	Register Total Power
	Sequential Internal Power
	Sequential Switching Power
	Sequential Leakage Power
	Sequential Total Power
	Combinational Internal Power
	Combinational Switching Power
	Combinational Leakage Power
	Combinational Total Power
	Total Internal Power
	Total Switching Power
	Total Leakage Power
	Total Power
Performance	Slack

TABLE II. FINAL 11 FEATURES

Analysis	Feature
Area	Number of ports
	Number of nets
	Number of cells
	Number of sequential cells
	Number of references
Power	Net Switching Power
	Total Dynamic Power
	Combinational Switching Power
	Combinational Total Power
	Total Switching Power
	Total Power

E. Data Examination & Finalization

The next step was the examination and finalization of the data. As mentioned, our initial dataset consisted of 51 GLN analysis features. First, we examined the data for missing values. Missing values can lead to wrong statistics during ML modeling and for this reason, it is mandatory to check them when a new database is going to be created. From the examination, it was occurred that there are no missing values in the dataset. Next, we examined the distribution of the dataset, but from the examination it was occurred that 40 of 51 features consisted exclusively from zero values. As a result, from the 51 features remained 11 features. Of them, 5 features are based on area analysis, and 6 features are based on power analysis (Table II).

F. Lack of Data and GAINESIS tool

As mentioned there is a general lack of data which also affected and our CAS-HtBase. In order to be able to solve this problem we developed our GAINESIS tool [23]. Our GAINESIS tool is a tool based on generative adversarial networks (GANs) [24], which is able to generate new synthetic data. For its development we relied on the 941 benchmarks and the 11 features. Initially, based on these benchmarks we developed seven ML classifiers, gradient boosting (GB) [25], logistic regression (LR) [26], k-nearest neighbor (KNN) [27], support vector machine (SVM) [28], random forest (RF) [29], multilayer perceptron (MLP) [30] and extreme gradient boosting (XGB) [31]. The best performed classifier was the GB-based. Next, we used those benchmarks and we developed four GAN-based models, GAN, conditional generative adversarial network (CGAN) [32], Wasserstein generative adversarial network (WGAN) [33] and Wasserstein conditional adversarial network (WCGAN) [34]. Based on those models we created and compared three different in size synthetic datasets and developed new GB-based classifiers. The best performed synthetic data produced from our WCGAN-based model. Our new GB-based classifiers were developed based on a combination of the synthetic data with the original real data. In our final step we compared our initial classifier with our new classifiers. Our best performed new classifier managed to increase the performance compared with the initial classifier. The final results presented that the data generated by our GAINESIS tool was able to develop robust classifiers.

III. RESULTS

A. CAS-HtBase Distribution

CAS-HtBase consists of HT-free and HT-infected benchmark circuits, synchronous and asynchronous from real and synthetic data. Specifically, HT-free benchmarks retrieved from the design of the 14 synchronous benchmarks: AES, B-15, ETHERNET MAC-10GE, MEM-CTRL, PIC-16F84, RS-232, S1423, S13207, S15850, S35932, S38417, S38584, VGA-LCD &WB-CONMAX and 4 asynchronous: C2670, C3540, C5315, and C6288. In total 18 HT-free

benchmark circuits. The HT-infected benchmark circuits consisted of modifications of the original HT-free benchmarks and are in total 923 HT-infected benchmarks. Specifically, the majority of the benchmarks came from the design of the S13207 circuit, with 15.4% (145 of 941), C2670 circuit with 14.8% (139 of 941), and C6288/C5315 circuits with 11.8% respectively (222 of 941), S15850 circuit with 11.5% (108 of 941) and C3540 circuit with 10.6% (100 of 941). S1423 and S35932 circuits consist of 9.2% of our database with 91 and 65 benchmarks respectively. The minority of the benchmarks coming from the circuits: AES with 2.3% (22 of 941), RS232 with 2% (20 of 941), B-15/ETHERNET MAC-10GE with 0.5% respectively (10 of 941), PIC-16F84/S38417/S38584/WB-CONMAX with 0.4% respectively (16 of 941) and MEM-CTRL/VGA-LCD with 0.2% per circuit (4 of 941) (Fig. 4).

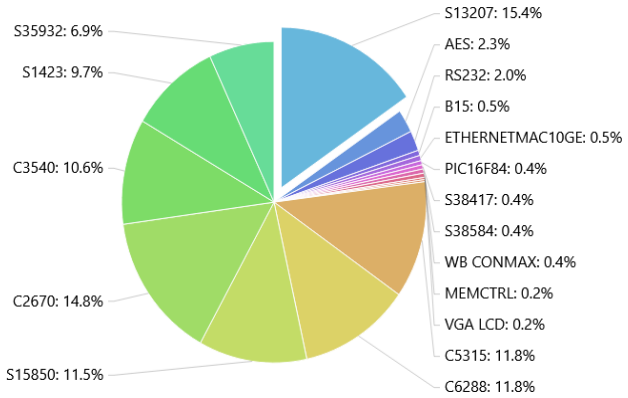

Fig. 4. CAS-HtBase benchmark distribution.

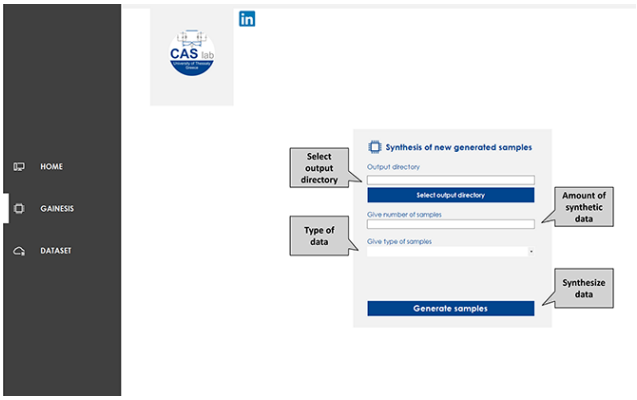

Fig. 5. CAS-HtBase, "GAINESIS" menu.

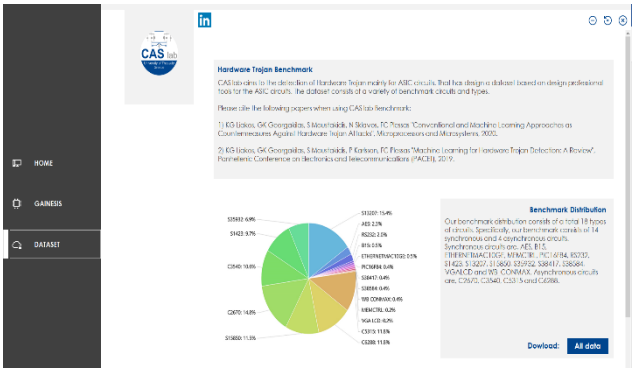

Fig. 6. CAS-HtBase, "DATASET" menu.

B. *CAS-HtBase and GAINESIS*

Interested users can download our CAS-HtBase in its original form through our laboratory's website or use the GAINESIS tool and create the amount of data they need. Our tool consists of three menus, home page, synthesis-GAINESIS and dataset menu. Home page menu provides general information about our laboratory. The GAINESIS menu is used to produce new synthetic data based on our original CAS-HtBase. Through this menu the users can choose the output directory of the new synthetic data. Next, they can specify the number of the new synthetic data which they want to create as well the type of the data, infected or uninfected and to proceed to synthesis of the new data (Fig. 5). Finally, through our dataset menu the users can see more information about our original CAS-HtBase or download it to the initial form (Fig. 6).

C. *CAS-HtBase Format*

An important factor that we considered when we designed our CAS-HtBase was to be used directly without conversions for the development of statistical measurements and ML models. CAS-HtBase format is to comma-separated values (CSV) file, to be able for the users to import it and read it directly as a data frame. CAS-HtBase consists of 13 elements, circuit name, and circuit label: zero and one (0 - 1). The zero number indicates that the benchmark is HT-infected and one number that is HT-free. The other features are the five areas and six power analysis features for each benchmark. We added as an element the circuit label to be able the users to develop supervised or unsupervised ML models, or to use them as an extra feature to their supervised models.

IV. CONCLUSION

A major problem in the research and academic community dealing with HT viruses is the general lack of data, the lack of uninfected reference circuits, the lack of small uninfected and infected circuits, and the large imbalance in the amount of data between uninfected and infected circuits. The work becomes even more difficult combined with the limited information and published studies for the ASICs and especially for the pre-silicon stage of them. For these reasons, we designed these limited benchmark circuits for the GLN phase to create a database exclusively for ASICs.

CAS-HtBase has been designed to accommodate all the experimental benchmark circuits, aiming at the creation of a holistic database consisting of features from all the pre-silicon phases of the design of ASICs. The main purpose of this database is to be studied and used by the research and academic community for further study of HTs viruses in ASICs. This attempt consists of the initial version of the holistic CAS-HtBase and is composed exclusively of GLN analysis features. In total CAS-HtBase consists of 941 benchmarks, 18 HT-free and 923 HT-infected. 4 Asynchronous and 14 synchronous benchmark circuits and 11 GLN features, 5 area & 6 power analysis features.

CAS-HtBase format is .csv for direct usage for statistical measurements and ML-based models' development. Through our CAS-HtBase in combination with our GAINESIS tool we can be alleviate the imbalance problem, the general lack problem and the lack of uninfected circuits problem for GLN-phase, but not the lack of small type of circuits.

CAS-HtBase and GAINESIS tool can be found on our site: https://caslab.e-ce.uth.gr/ToolsandDatabases.html. Our CAS-HtBase and our tool are free with open access to the public and can be used by academics – researchers or the general public.

In the future we will create our small-in-size circuits, aiming to solve the lack of diversity that is present in free benchmark circuits, and through these, we will upgrade our CAS-HtBase. Also, we will focus to create databases for other pre-silicon IC production phases such as RTL, PNR, and GDSII. The main aim is to provide a holistic generative database for the pre-silicon stage of the ICs production chain.

REFERENCES

[1] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar, "Trojan detection using IC fingerprinting," 2007, doi: 10.1109/SP.2007.36.

[2] J. Aarestad, D. Acharyya, R. Rad, and J. Plusquellic, "Detecting trojans through leakage current analysis using multiple supply pad IDDQs," *IEEE Trans. Inf. Forensics Secur.*, 2010, doi: 10.1109/TIFS.2010.2061228.

[3] R. Rad, J. Plusquellic, and M. Tehranipoor, "A sensitivity analysis of power signal methods for detecting hardware trojans under real process and environmental conditions," *IEEE Trans. Very Large Scale Integr. Syst.*, 2010, doi: 10.1109/TVLSI.2009.2029117.

[4] F. Koushanfar and A. Mirhoseini, "A unified framework for multimodal submodular integrated circuits trojan detection," *IEEE Trans. Inf. Forensics Secur.*, 2011, doi: 10.1109/TIFS.2010.2096811.

[5] C. Lamech, R. M. Rad, M. Tehranipoor, and J. Plusquellic, "An experimental analysis of power and delay signal-to-noise requirements for detecting trojans and methods for achieving the required detection sensitivities," *IEEE Trans. Inf. Forensics Secur.*, 2011, doi: 10.1109/TIFS.2011.2136339.

[6] K. Xiao, X. Zhang, and M. Tehranipoor, "A clock sweeping technique for detecting hardware trojans impacting circuits delay," *IEEE Des. Test*, vol. 30, no. 2, pp. 26–34, 2013, doi: 10.1109/MDAT.2013.2249555.

[7] R. S. Chakraborty, F. Wolff, S. Paul, C. Papachristou, and S. Bhunia, "MERO: A statistical approach for hardware Trojan detection," 2009, doi: 10.1007/978-3-642-04138-9_28.

[8] A. Waksman, M. Suozzo, and S. Sethumadhavan, "FANCI: Identification of stealthy malicious logic using boolean functional analysis," 2013, doi: 10.1145/2508859.2516654.

[9] J. Zhang, F. Yuan, L. Wei, Y. Liu, and Q. Xu, "VeriTrust: Verification for hardware trust," *IEEE Trans. Comput. Des. Integr. Circuits Syst.*, 2015, doi: 10.1109/TCAD.2015.2422836.

[10] K. Hasegawa, M. Oya, M. Yanagisawa, and N. Togawa, "Hardware Trojans classification for gate-level netlists based on machine learning," 2016, doi: 10.1109/IOLTS.2016.7604700.

[11] K. G. Liakos, G. K. Georgakilas, and F. C. Plessas, "Hardware Trojan Classification at Gate-level Netlists based on Area and Power Machine Learning Analysis," 2021, doi: 10.1109/ISVLSI51109.2021.00081.

[12] S. P. Moustakidis, K. G. Liakos, G. K. Georgakilas, and N. Sketopoulos, "A novel holistic approach for hardware trojan detection powered by deep learning (HERO)," *Attract'20*. 2020.

[13] C. Bao, D. Forte, and A. Srivastava, "On application of one-class SVM to reverse engineering-based hardware Trojan detection," 2014, doi: 10.1109/ISQED.2014.6783305.

[14] C. Bao, Y. Xie, Y. Liu, and A. Srivastava, "Reverse engineering-based hardware trojan detection," in *The Hardware Trojan War: Attacks, Myths, and Defenses*, 2017.

[15] M. Xue, J. Wang, and A. Hux, "An enhanced classification-based golden chips-free hardware Trojan detection technique," 2017, doi: 10.1109/AsianHOST.2016.7835553.

[16] H. Mardani Kamali, K. Zamiri Azar, K. Gaj, H. Homayoun, and A. Sasan, "LUT-Lock: A novel LUT-based logic obfuscation for FPGA-Bitstream and ASIC-hardware protection," 2018, doi: 10.1109/ISVLSI.2018.00080.

[17] H. Salmani, M. Tehranipoor, and J. Plusquellic, "A Novel Technique for Improving Hardware Trojan Detection and Reducing Trojan Activation Time," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 20, no. 1, pp. 112–125, Jan. 2012, doi: 10.1109/TVLSI.2010.2093547.

[18] B. Khaleghi, A. Ahari, H. Asadi, and S. Bayat-Sarmadi, "FPGA-based protection scheme against hardware trojan horse insertion using dummy logic," *IEEE Embed. Syst. Lett.*, vol. 7, no. 2, pp. 46–50, 2015, doi: 10.1109/LES.2015.2406791.

[19] X. T. Ngo, J. L. Danger, S. Guilley, Z. Najm, and O. Emery, "Hardware property checker for run-time Hardware Trojan detection," *2015 Eur. Conf. Circuit Theory Des. ECCTD 2015*, pp. 1–4, 2015, doi: 10.1109/ECCTD.2015.7300085.

[20] F. Khalid, S. R. Hasan, O. Hasan, and F. Awwad, "Runtime hardware Trojan monitors through modeling burst mode communication using formal verification," *Integration*, vol. 61, no. October 2017, pp. 62–76, 2018, doi: 10.1016/j.vlsi.2017.11.003.

[21] H. Salmani, M. Tehranipoor, and R. Karri, "On design vulnerability analysis and trust benchmarks development," 2013, doi: 10.1109/ICCD.2013.6657085.

[22] B. Shakya, T. He, H. Salmani, D. Forte, S. Bhunia, and M. Tehranipoor, "Benchmarking of Hardware Trojans and Maliciously Affected Circuits," *J. Hardw. Syst. Secur.*, 2017, doi: 10.1007/s41635-017-0001-6.

[23] K. G. Liakos, G. K. Georgakilas, F. C. Plessas, and P. Kitsos, "GAINESIS: Generative Artificial Intelligence NEtlists SynthesIS," *Electron.*, vol. 11, no. 2, 2022, doi: 10.3390/electronics11020245.

[24] I. J. Goodfellow *et al.*, "Generative adversarial nets," 2014, doi: 10.3156/jsoft.29.5_177_2.

[25] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Ann. Stat.*, 2001, doi: 10.1214/aos/1013203451.

[26] J. Berkson, "Application of the Logistic Function to Bio-Assay," *J. Am. Stat. Assoc.*, 1944, doi: 10.1080/01621459.1944.10500699.

[27] T. M. Cover, "Estimation by the Nearest Neighbor Rule," *IEEE Trans. Inf. Theory*, 1968, doi: 10.1109/TIT.1968.1054098.

[28] C. Cortes and V. Vapnik, "Support-Vector Networks," *Mach. Learn.*, 1995, doi: 10.1023/A:1022627411411.

[29] L. Breiman, "Random forests," *Mach. Learn.*, 2001, doi: 10.1023/A:1010933404324.

[30] S. K. Pal and S. Mitra, "Multilayer Perceptron, Fuzzy Sets, and Classification," *IEEE Trans. Neural Networks*, 1992, doi: 10.1109/72.159058.

[31] T. Chen and C. Guestrin, "XGBoost," 2016, doi: 10.1145/2939672.2939785.

[32] M. Mirza and S. Osindero, "Conditional Generative Adversarial Nets Mehdi," *arXiv1411.1784v1 [cs.LG] 6 Nov 2014 Cond.*, 2018.

[33] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," 2017.

[34] S. Qin and T. Jiang, "Improved Wasserstein conditional generative adversarial network speech enhancement," *Eurasip J. Wirel. Commun. Netw.*, 2018, doi: 10.1186/s13638-018-1196-0.