IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 11, 2019, accepted January 23, 2019, date of publication February 5, 2019, date of current version March 5, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2896479

A Multi-Layer Hardware Trojan Protection Framework for IoT Chips

CHEN DONG[1,2], (Member, IEEE), GUORONG HE[1,2], XIMENG LIU[1,2], (Member, IEEE), YANG YANG[1,2], (Member, IEEE), AND WENZHONG GUO[1,3], (Member, IEEE)

[1]College of Mathematics and Computer Science, Fuzhou University, Fuzhou 350116, China
[2]Key Laboratory of Information Security of Network Systems, Fuzhou University, Fuzhou 350116, China
[3]Fujian Provincial Key Laboratory of Network Computing and Intelligent Information Processing, Fuzhou University, Fuzhou 350116, China

Corresponding author: Wenzhong Guo (fzugwz@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61672159, Grant 61872091, Grant U1804263, and Grant 61702105, in part by the Foundation of the Education Department of Fujian Province under Grant JAT170099, and in part by the Science Foundation of the Fujian Province under Grant 2018J01793.

ABSTRACT Since integrated circuits are performed by several untrusted manufacturers, malicious circuits (hardware Trojans) can be implanted in any stage of the Internet-of-Things (IoT) devices. With the globalization of the IoT device manufacturing technologies, protecting the system-on-chip (SoC) security is always the keys issue for scientists and IC manufacturers. The existing SoC high-level synthesis approaches cannot guarantee both register-transfer-level and gate-level security, such as some formal verification and circuit characteristic analysis technologies. Based on the structural characteristics of hardware Trojans, we propose a multi-layer hardware Trojan protection framework for the Internet-of-Things perception layer called *RG-Secure*, which combines the third-party intellectual property trusted design strategy with the scan-chain netlist feature analysis technology. Especially at the gate level of chip design, our *RG-Secure* is equipped with a distributed, lightweight gradient lifting algorithm called lightGBM. The algorithm can quickly process high-dimensional circuit feature information and effectively improve the detection efficiency of hardware Trojans. In the meanwhile, a common evaluation index F-measure is used to prove the effectiveness of our method. The experiments show that *RG-Secure* framework can simultaneously detect register-transfer-level and gate-level hardware Trojans. For the trust-HUB benchmarks, the optimized lightGBM classifier achieves up to 100% true positive rate and 94% true negative rate; furthermore, it achieves 99.8% average F-measure and 99% accuracy, which shows a promising approach to ensure security during the design stage.

INDEX TERMS Internet of Things, protection framework, hardware security, hardware Trojan.

I. INTRODUCTION

With the achievements of wireless communication, sensor technology, embedded system application, and microelectronics technology, the Internet of things (IoT) technology has been widely valued by the governments, academia and industry of various countries due to its vast application prospect [1]. For example, appointment registration and remote consultation driven by IoT technologies can promote the development of smart healthcare. By embedding and integrating IoT devices into industrial systems (such as power grids, bridges, highways, and buildings), the process of production can be managed more precisely and dynamically. IoT applications can significantly improve the industrial productivity and relationship between human and nature.

Although the IoT is developing rapidly, the security of the IoT become extremely prominent (mainly for industrial applications). The perception layer is the source of all IoT data [2]. It is also the foundation of the entire IoT architecture. The physical security of sensor layer devices will be more threatened than the transport layer and application layer of the Internet of things. Due to the wide distribution of sensors in agricultural and industrial environments, sensitive information may be directly captured by the enemy if no one checks the sensors for a long time. So, most IoT devices in circulation are vulnerable to malicious attacks. At present, some scholars have pointed out that wireless embedded medical devices such as pacemakers and insulin pumps in the market generally have available security holes [35].

The associate editor coordinating the review of this manuscript and approving it for publication was Tawfik Al-Hadhrami.

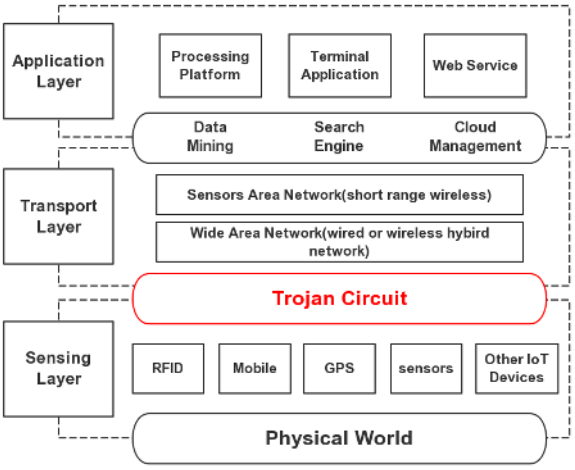

FIGURE 1. Logical architecture for IoT and Malicious circuit implantation location.

Some HT circuits or computer virus can cause disruption to cyber physical systems (for example: industrial control systems) [3]. The defense science council reported in March 2017 that U.S. military weapons systems may have been injected with HTs [4]. It is easy for a malicious attacker to insert a hardware Trojan that affects the normal operation of the system or steal sensitive information [5]. A variety of intelligent devices, from environmental sensors to cameras, are used to obtain data [6], [7]. Therefore, some malicious circuits are often embedded in the SoC of IoT devices (such as video monitors, smartphones and autonomous vehicles) — see Fig.1. In general, HTs infect the SoC of sensing layer devices. The structure of HTs is extremely uncomplicated, including trigger blocks and payloads. Typically, the hardware Trojans does not contain any state information. The malicious attackers have full control over their HT triggers and implant various types of HTs which would be extremely hard to detect by traditional verification techniques. Furthermore, SoC in circulation is a complex heterogeneous system composed of multiple third-party IP cores. It further increases the risks faced by IoT devices. In the past few years, some works have been devoted to SoC security. However, Trojan-detection techniques in the 3PIPs are difficult to distinguish Trojan nets completely because of the small size and concealment of HTs. Some Tojans even need to be analyzed manually. Moreover, formal verification techniques for RTL/gate-level cannot take the merit of high efficiency and accuracy at the same time. Some malicious third party suppliers even colluded to jointly manufacture HTs to evade detection. So, how to design safe and reliable SoC secure strategies and Trojan detection technology are critical tasks for researchers.

In this paper, we design a multi-layer hardware Trojan protection framework called *RG–Secure*, which tackles the main security threat of IoT terminals — hardware Trojans lives in RTL and gate level. Our *RG–Secure* architecture is established by combining the excellent experience of previous works. Compared with other approaches, our framework introduces scan-chain netlist features to improve the detection accuracy of gate-level HTs. In addition, we propose three SoC design strategies to prevent Trojan IPs from untrusted vendors. Experiments prove that our framework can achieve high detection accuracy, wide applicability and suitable for very large scale integrated circuits (VLSI). The contributions are summarized as follows:

- We propose *RG–Secure*, a fast and applicative hardware Trojan protection framework for third-party IP cores based on HTs in RTL and gate level. Our *RG–Secure* can find HTs by multilayer analysis rather than directly analyzing the functionality based on netlists. It is suitable for multi-type hardware Trojan detection.
- Based on the particularity of the SoC structure, we make attempts to detect HTs in RTL. Introducing three security design strategies for untrusted third-party IP cores to prevent HTs infection. At the same time, our improved design strategies can reduce the space complexity and overhead of the original safety design.
- At gate level, some gate-level Trojans cannot be converted to scan-chain netlist in the functional testing phase. We put forward two new features called suspicious trigger module and suspicious observe point through analyzing the difference between Trojan netlist and scan-chain netlist. Compared with the traditional circuit feature analysis methods, our scan-chain analysis technology has higher detection accuracy.
- In order to deal with high-dimensional circuit information, we learn from the idea of feature analysis of HT detection technologies and propose a gate-level HT detection method based on LightGBM algorithm. Experiments show the effectiveness of our framework using 15 benchmarks from the trust-hub [36].

The rest of the paper is organized as follows. In Section II, the related works are introduced. In Section III, the design goals and problem models of this paper are described. The multi-layer hardware Trojan protection framework is proposed *RG–Secure* in Section IV. Experimental results are analyzed in Section V. The Section VI is the conclusion.

II. RELATED WORK

To address the risks that HTs bring, HT detection technology is a promising direction. Various classification algorithms and security designs have been proposed.

Hicks *et al.* [8] proposed a technique called Unused Circuit Identification (UCI) to find lines of RTL code that have not yet been executed during the test. Zhang *et al.* [9] proposed a hardware verification method called VeriTrust. The verification module can automatically identify suspicious gates which are driven by HT. However, some of Trojan gates are not calculated based on function input. Farahmandi *et al.* [10] proposed a HT detection technique based on symbolic algebra. Xiao and Forte *et al.* [11] briefly summarized the research situation and development trend of HTs in recent years, while Karri *et al.* [12] sorted out the development process of the

concept of hardware Trojans. There are also HT detection technologies based on the analysis of HT features proposed by Salmani *et al.* [13] and Oya *et al.* [14]. But the security workload is added to the computational effort imposed by feature extraction and classification algorithms. Existing HT detection techniques for SoC can be roughly classified into two categories: formal verification and circuit feature analysis.

A. FORMAL VERIFCATION

If agreement can be reached between IP vendors and SoC integrators, specific design rules are assigned to IP cores at outsourcing time. Rajendran *et al.* [15], [16] proposed a technique to formally verify malicious modification of critical data in 3PIP by hardware Trojans. Their approach takes advantage of 3PIP from diverse vendors. No need for ''golden'' IP (A standard IP core without hardware Trojans implant) as a reference. However, their method leads to high area and power cost, and it is not able to prevent the hardware Trojans that are implanted at gate level. Besides, authors also put forward a formula for information leakage paths, as shown in formula 1.

$$P \models (s_0 = o) \vee (\neg s_0 = o), \quad \forall s_0 \in \{0, 1\}. \tag{1}$$

where, s_0 is the binary representation of private information (e.g., user password, private key). o is any output path (e.g., output port). The authors hold that this formula detect if the sensitive information is leaked. However, researchers have proved [17] that the method is ineffective in finding information leakage Trojans by inclusive formal verification(IFV) tool of Cadence. As shown in formula 2. If there is no information path between U_1 and U_n. The IFV tool will return a false positive result.

$$P \models \exists(U_1 = U_n) \vee (\neg U_1 = U_n). \tag{2}$$

Love *et al.* [18] and Guo [19] proposed a set of circuit safety attributes and systematically studied a proof-carrying-based framework. In the framework, IP cores are accompanied by formal proof of security attributes. But, it may hard to detect other unexpected features introduced by HTs. Formal verification method does not directly detect HT, but attempts to evaluate the reliability of IP core, and designs IP cores through trust strategies to improve the security of integrated circuits. In addition, the technology can isolate infected third-party IP through software scheduling protocol [20] or the correlation of input signals [21].

B. CIRCUIT FEATURE ANALYSIS

Circuit feature information can reflect the health of SOC. Hasegawa and Oya [22], Hasegawa and Yanagisawa [23], and Hasegawa *et al.* [24] extracted features from the hardware Trojans to identify dangerous netlist and normal netlists. The method constructs a machine learning classifier to identify the Trojans. It has high detection efficiency and low cost. However, the detection accuracy and algorithm stability depends on the selection of Trojan features.

Many schemes detect unused circuit modules in various ways to mark suspicious circuits or nets, such as UCI [8], Verirust [9] and FANCI [25]. These methods are simple to implement, but it is so specific that existing benchmarks may not be general. Additionally, the technologies cannot guarantee Trojan detection so that a small number of suspicious gates need to be manually analyzed by SoC integrators. Methods' limitations have been discussed in details in papers [26], [27].

Rajendran *et al.* [26] proposed a hardware Trojan detection and recovery method based on gate-level netlists. The method utilizes the controllability and observability to show the distance relative to the standard circuit. Circuit feature analysis can be applied to behavior or structural code to find suspicious assertions or modules. Other literatures [29], [30] used quantitative measurement or scoring mechanism to label dubious nets or gates.

Characteristics in circuits also include the side channel information of SoC. The main idea is to detect the path delay and power consumption of integrated circuits [31]. A method is designed to look up the difference in circuit overhead called MERO. The power component of leaking a bit K_i is shown in the formula 3, $s_{K_i}(n)$ represents the time-shift correlation of information flow. N is the number of bits in the binary representation of private information.

$$P_{K_i}(t) = \sum_{n=2}^{N} [K_i \oplus (s_{K_i}(n) - s_{K_i}(n-1)) \cdot P_{0\to1}(t)]. \tag{3}$$

However, Chakraborty and Pagliarini [32] proved that when the Trojan triggers probability is less than 10^{-3}, MERO will be difficult to detect Trojans.

Yoshimizu [33] attempted to detect the hardware Trojans with circuit path delay. A temperature-based HT classification technology [34] is an example of the circuit feature analysis. However, it is a pity that ''standard'' samples are difficult to obtain in reality, and the detection efficiency of side channel depends on the precision of the instrument. Side channel analysis needs to collect information frequently, and it is weak to deal with the interference of process noise. So the de-noising of by-pass noise may affect the original signal again. That is why the side channel approach is hard to popularize.

III. PROBLEM FORMULATION

In this section, the problem formulation of our *RG–Secure* is introduced, including threat model, security model and goals of paper.

A. THREAT MODEL

Hardware Trojans can be divided into several types: information disclosure, change of functionality, and denial of service. When Trojan triggers are not activated, HTs can coexist with the original circuit without affecting the normal function. Once triggers release a rare signal or condition, the payload is activated to perform a malicious behavior. Generally speaking, the HTs are implanted in the circuit at the design stage.

Only a small group of HTs are implanted in the manufacturing phase.

The realized function R of a SoC (formula 4) is defined three parts: operation α performed by a chip, sets of input ports I and output ports O. R is composed of different circuit subfunctions r_n. These subfunctions break down the complex system-on-chip into a simple circuit structure (formula 5).

$$R = \{\alpha, I, O\}, \tag{4}$$

$$R = r_1 \cup r_2 \cup r_3 \cup \cdots \cup r_n = \bigcup_{n=1}^{N} r_n, \tag{5}$$

These malicious activities of HTs undoubtedly break the original circuit structure and produces unwanted behaviors (formula 6,7).

$$R = R_{secur} \cup (R_h \cup r_h), \tag{6}$$

$$R = r_1 \cup r_2 \cup r_3 \cup \cdots \cup r_n \cup r_h = \bigcup_{n=1}^{N} r_n \cup r_h. \tag{7}$$

Here, each type of HTs locate in RTL can be expressed as I_R, C_R and D_R. Similar to the gate-level HTs is I_G, C_G and D_G. r_h represents the hardware Trojan circuits. If ξ is used to indicate the trigger signal of Trojan, the existence form of hardware Trojans in SoC will be characterized by the following equations (formula 8,9). When the Trojans receive the trigger signal ($\xi = true$), it will perform the predetermined behavior. Otherwise, the hardware Trojan is in static state ($r_h = \phi$).

$$r_h = \phi \quad if\ \xi = false, \tag{8}$$

$$r_h = \{I_R, C_R, D_R, I_G, C_G, D_G\} \quad if\ \xi = true. \tag{9}$$

In this paper, our framework plays a vital role in the pre-silicon phase of SoC security design. It is suitable for multi-type HT detection. This method work on combining the third-party IP trusted design strategy with the netlist feature analysis technology and ensuring the security of RTL and gate-level netlist. In other words, it guards netlists instead of physical layout stage. Hence, it cannot detect HTs inserted in the layout level.

B. SECURITY MODEL

Due to the increasing number of IoT application, SoC architecture has become immensely complicated. Our method considers two stages of HT in SoC: RTL Trojans and gate level Trojans (I_R, C_R, D_R, I_G, C_G, D_G). Information leakage hardware Trojans I_R and I_G mainly attack crypto processor and graphics controller. D_R and D_G can lead to circuit damage or denial of service. They are usually located in bus controller, general processor and system functions. C_R and C_G basically change the circuit function and they can be implanted at any stage of the SoC.

Thus, the physical security of perception-layer devices will be more seriously threatened than before. It is tough to find a universal method which can defense all kinds of HTs. Both formal verification and circuit feature analysis have their own pros and cons. To protect the safety of SoC in IoT devices, we overcome these limitations by combining the two technologies. In the process of SoC integration, RTL is the most vulnerable to hardware Trojan attacks. Security design strategies proposed for 3PIP can prevent and detect part of the circuits infected by HTs in RTL. In the next stage, SoC integrators synthesize the RTL description into a gate-level netlist. Unfortunately, there is still a Trojan threat at the gate level. Circuit feature analysis technologies do well in detecting hardware Trojans injected at the gate level. So a multi-layer hardware Trojan protection framework is designed for the IoT perception layer SoC, as illustrated in Fig.2.

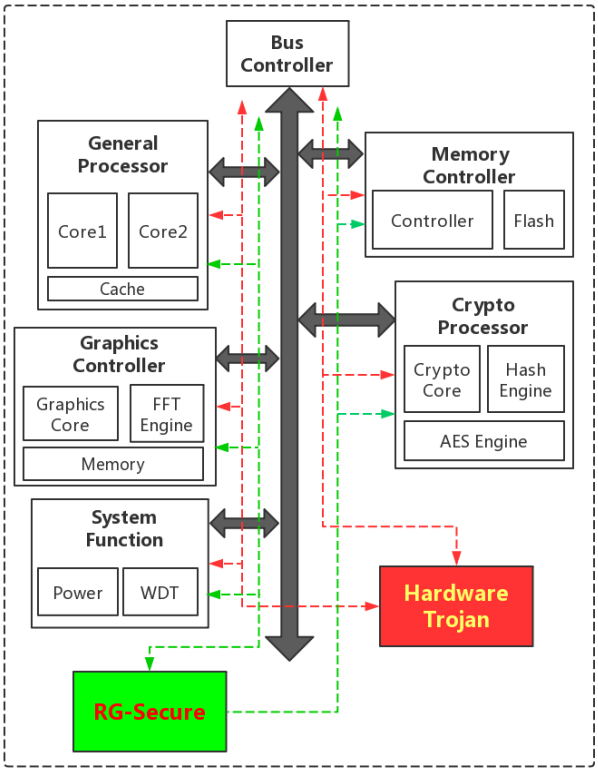

FIGURE 2. RG-Secure Model Based on SoC chips in IoT equipments.

C. DESIGN GOALS

Our goal is to create a hardware Trojan protection framework for IoT Chips. Specifically, we have the following goals:

- **Applicability**. Our *RG–Secure* is to cope with the complex and various HTs. The designed framework is able to fully protect the safety of SoC.
- **Accuracy**. Accuracy of *RG–Secure* framework should be accurate enough to detected hardware Trojans without omission. Also, all normal circuit nets cannot be misjudged.
- **Efficient**. Compared with other detection methods, the time complexity of our algorithm cannot be too high. It is necessary to ensure timeliness as well as accuracy.

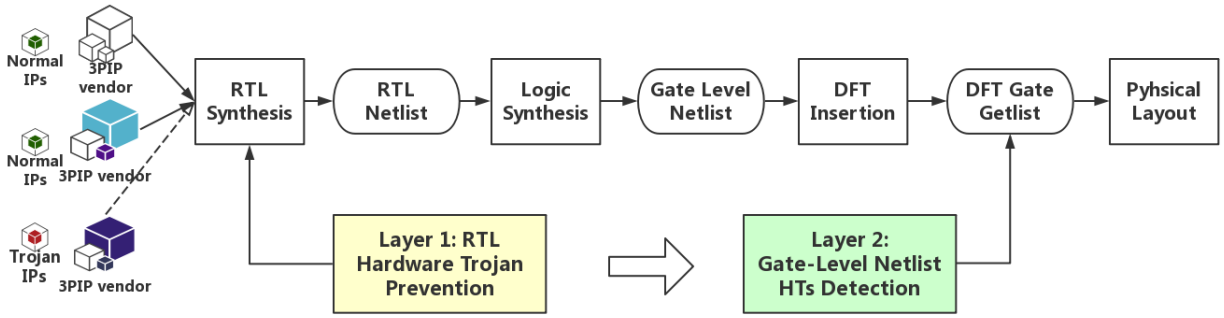

FIGURE 3. Overall framework of *RG–Secure*.

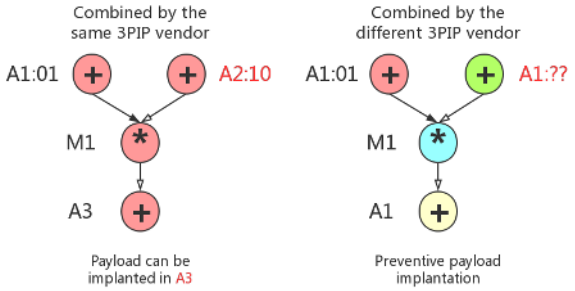

FIGURE 4. Control data flow graph of strategy 1.

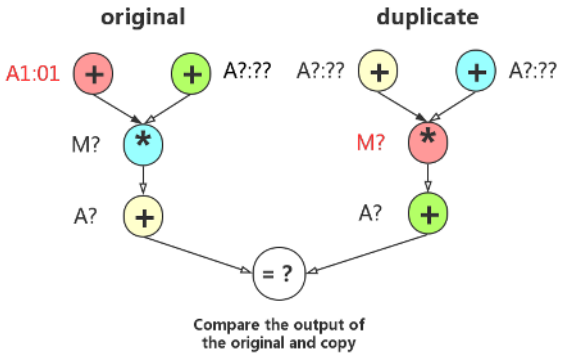

FIGURE 5. The CDFG of duplication for detection.

IV. SECURITY FRAMEWORK FOR RTL AND GATE LEVEL

In this section, we introduce the proposed multilayer hardware Trojan protection framework — *RG–Secure*, including RTL-Trojan prevention and gate-level Trojan detection.

A. TRUSTED DESIGN BASED ON REGISTER-TRANSFER-LEVEL NETLIST

RTL is the phase of chip manufacturing, which is the easiest stage to be implanted hardware Trojans. In this part, a trusted approach is presented to prevent the Trojan threat.

In order to build trustworthy SoCs, we incorporate some security strategies into the advanced integrated operations to operator assignment phase. Fig.3 shows the overall framework of our RG–Secure. As shown in Fig.3, the design strategies for RTL hardware Trojan prevention is the first layer of our approach. Typically, an IP vendor sells multiple 3PIPs. Malicious IP vendors may implant hardware Trojans in their products (Trojan IPs). They design a distributed HT by coordinating Trojan triggers. In response to two or more Trojan IPs form the same vendors, we introduced *Strategy 1*.

Strategy 1 (IP Diversity): Use IP cores from different third-party IP vendors whenever possible. Because malicious attackers are difficult to predict and control the output of 3PIPs from different vendors. They are unable to determine the placement of Trojan trigger modules and load modules. An example shown in Fig.4, circles of different colors represent disparate 3PIP suppliers (red circles represent malicious IP vendors). Netlists manufactured by single 3PIP vendor is more susceptible to infection than multiple vendors. It is because IP components from the same 3PIP vendor can quickly form a Trojan trigger signal. Attackers set the location of HT trigger as they wish. In Fig.4, collusion between a child and parent operation in the same vendor. The output '10' of $A2$ is acquired by opponents. In the way, Trojan payloads are implanted in $A3$ and launched when receiving the signal of '01*10' of $A1$ and $A2$. However, if we employ 3PIPs from different vendors to integrate netlists, the output of $A2$ will be hard to predict. Malicious attackers are unable to determine a place that Trojan payload can be implanted.

The above method only applies to the case that there is only one malicious vendor. For the collusion of multiple malicious vendors, we also bring in *Strategy 2*.

Strategy 2 (IP Randomness): Randomly selecte 3PIP at the design stage, the number of 3PIP within a function is guaranteed to completely random. It can prevents collusive attacks between suppliers unless all the vendors are hardware Trojan manufacturers. There is no doubt that the possibility almost zero. However, in order to prevent the hardware Trojans generated by random combination of IP, we take inspiration from Rajendran's work. Adding a duplication for verifying whether there is a malicious circuit in design.

Strategy 3 (Duplication for Detection): Create a copy with the same function as the original circuit. Certainly, it also follow *Strategy 1* and *Strategy 2*. Duplication is used to detect whether the output of an operation contains a hardware Trojan. The technique can find at least one malfunction of a given 3PIP. The CDFG of duplication for detection shown in Fig.5.

In Fig.5, both the original and copy are composed by different IP vendors. The attackers only get the output result of $A1$. If the subsequent operation cannot be obtained, the malicious modification of the circuit will become very difficult. In addition, for preventing the occurrence of random events (exactly choose the same IP), we compare the original output with duplication. Once finding a difference, manufacturers can believe there is a malicious circuit in the operation. Assuming the total number of suppliers is N, the number of untrusted suppliers or colluding attacks is D. The probability that HTs may escape is $(D/N)^4$ in our method (Footnote: we do not consider the total of IPs). So, the more types of suppliers we choose, the safer our proposed design approach will be.

Our *RG–Secure* is based on Rajendran *et al.*. However, we abandon redundant design constraints and remove isolation operations. It is because removing some restrictions helps reduce the total number of IPs and production costs. In a nutshel, these above strategies are the first layer of our *RG–Secure* framework. It mainly cope with some HTs in RTL which change the functionality or deny service. Unfortunately, it is weak to detect HTs that leakage information or come from gate level.

B. SCAN-CHAIN FEATURE ANALYSIS BASED ON GATE-LEVEL NETLIST

In the section, we come up with another layer of our framework for the above threats — Scan-Chain Feature Analysis based on Gate-Level netlist (in Fig.3).

To evaluate the effectiveness of integrated circuits, the circuit function needs to be tested in the design process. For IP cores, the scan chain is a conventional technology of Design for Testability (DFT). It is supported by all commercial EDA softwares. Based on DFT, we introduce how to extract Trojan features from scan-chain netlists and classify HT nets using LightGBM algorithm.

1) SUSPICIOUS TRIGGER MODULE

Identifying suspicious trigger modules of HT is the first step in our feature analysis. After circuits are designed, manufacturers only detect them through its input and output ports. However, in most cases, it is impossible to rely on a single test vector to detect chip functionality completely. The scan chain is able to turn a difficult-to-test sequential circuit into an easy-to-test combined circuit which is shown in Fig.6.

In scan chain technology, each normal sequence element (such as D Flip-Flop) will be replaced with a scan sequence element (such as Scan D Flip-Flop). By connecting the output (Q) of each Scan D Flip-Flop (*SDFF*) with the input interface (*SI*) of next (*SDFF*), a scan chain is formed. In Fig.6, *Primary Input x* is the original signal of chips. *Test Input* accesses a test excitation signal, and *Test Output* represents the output signal of the new combinatorial circuit. We can determine whether there are defects or faults in the circuit through the values of *Scan out*.

In order to avoid the scan phase, some types of hardware Trojans will not be converted to scan elements (As shown in Fig.7). In Fig.7, An information leaking hardware Trojan can reveal important information about the circuit by Trojan leakage path. To evade functional validation, it still needs to add HTs rarely triggered. So, the Trojan trigger module must be different from the standard scan chain trigger module.

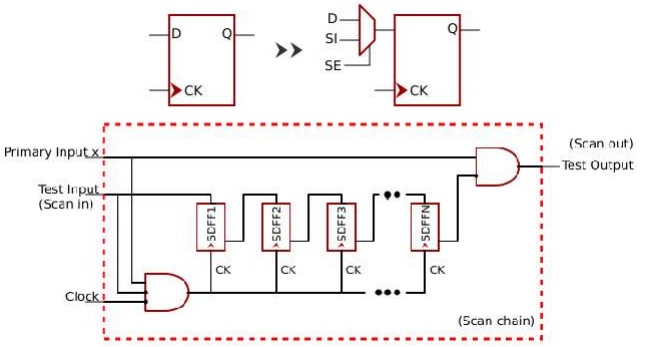

FIGURE 6. Scan chain generation.

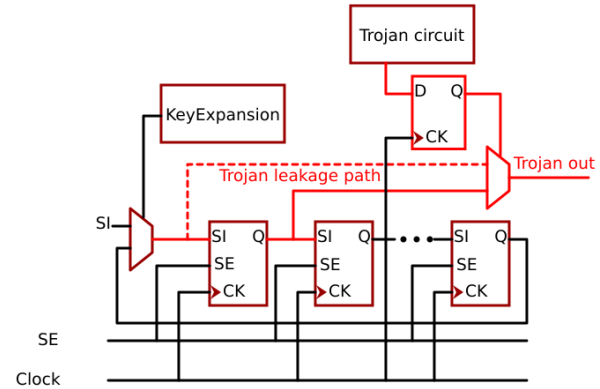

FIGURE 7. An information leakage Hardware Trojan.

Therefore, a new feature has been defined for such triggers called suspicious trigger module, which is used to look for suspicious Trojan trigger modules in the scan chain. If our method finds any triggers which cannot be converted to scan modules, it will be considered as HT trigger units.

2) SUSPICIOUS OBSERVE POINT

Based on Feature 1, some of Trojan circuits can be identified conclusively. However, It is far from enough to find the trigger module of Trojans. In order to avoid suspicious nets are not missed, we also need to identify the Trojan nets which are hidden in the normal netlist (such as *Trojan leakage path* or *Trojan out* in Fig.7).

The second step of our gate-level detection is to find some special nets, which are not only the input of scan triggers (*SDFF*) but also the input of main output (*Test Output*). Becasue if an attacker wants to steal critical information, the signal must pass through the final output. However, scanning chain can convert a sequential circuit into a combination circuit, the circuits will shift a chain structure. Therefore, HT injection will partially affect the linear relationship of the

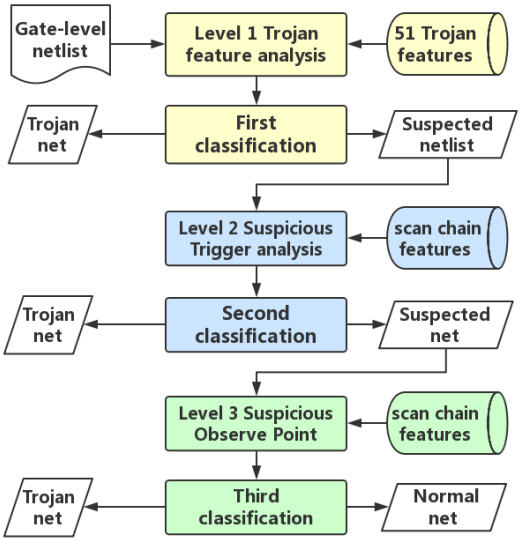

FIGURE 8. Flow of our implementation of the three-level feature analysis.

chain structure. We position this kind of nets as Suspicious Observe Point. For example, if we discover an observation point which is both the input of the next scan trigger and the main input of the circuit then we can conclude that there exists a suspicious leakage path or Trojan out. Generally, and the HT nets will cause information leaks or functional changes.

C. LIGHTGBM-BASED HARDWARE TROJAN CLASSIFICATION

In this section, we combine scan-chain features with the HT features summarized in previous work [22], and explain in detail why we use lightGBM algorithm to classify.

The feature analysis technique is presented in Algorithm.1 and shown in Fig.8. The algorithm, first takes a gate-level netlist as input. Next, initializes statistics (Suspicious Trigger Module, Suspicious Observe Point and Trojan net). Then, the algorithm classifies the netlist for the first time based on 51 feature databases [23] (Line 5). Apparently, some of HTs cannot be detected. Here, we adds scan-chain features to all netlists. At the same time we use conditional selection strategies to analyze suspicious netlists. Now, for each *net* $\in$ netlist, the algorithm judges whether it is suspicious (Line 10, Line 14). Once the net is judged to be dangerous, then it will be marked (Line 7, Line 13, Line 17). Suspicious Trigger Module and Suspicious Observe Point will increase at the same time (Line 12, Line 16). Finally, count all marked *net* as Trojan nets (Line 22).

In order to attain efficient HT detection results, we implement a three-level HT detection, as illustrated in Fig.8, including HT feature analysis, suspicious trigger analysis, and suspicious observation point analysis. At the first level, the netlists are classified for the first time based on 51 features without considering scan-chain features. The 51 circuit features mainly include: fan in/out features of circuit, multiplexer features, flip flop features, the number of circuit loops

Algorithm 1 Scan-Chain Feature Analysis

1: **Procedure** Scan-Chain Feature Extraction
2: **Input:** gate-level netlist
3: **Output:** suspicious trigger module, suspicious observe point, Trojan net
4: **Initialization:** STriModule, SObsPoint, TrojanNet
5: **for** *net* $\in$ *netlist* **do**
6: *analyze_Trojan*(*netlist*)
7: **if** *analyze_Trojan* $= 1$ **then**
8: mark *net*
9: **else**
10: *analyze_input*(*net*, "*test_se*")
11: **if** *analyze_input* $< 1''$ **then**
12: *STriModule* $=$ *STriModule* $+ 1$
13: mark *net*
14: **else** *analyze_input*(*net*, *PO*)
15: **if** *analyze_input*(*net*, *PO*) $= 1''$ **then**
16: *SObsPoint* $=$ *SObsPoint* $+ 1$
17: mark *net*
18: **else** break
19: **end if**
20: **end if**
21: **end if**
22: **end for**
23: *TrojanNet* $=$ *all_markednet*

and constant information (such as grounding or constant signal). The specific features can be referenced in [23]. Some hardware Trojans that have a large area are distinguished at this level. In the next level, we increase the Suspicious Trigger Module feature to the original HT feature library and train the rest of suspicious netlists. At this stage, Trojan nets are classified with special Trojan triggers. In the third level, the Suspicious Observe Point features are introduced to distinguish between suspected trojan leakage paths. In the level, a possible signal leakage net can be detected. Through the above three stages, we can eliminate the gate-level details and keep important circuit faeture information. The three-level frame aims to merge the scan-chain features and improve the detection accuracy.

In the whole experiment, *RG–Secure* choose to employ lightGBM algorithm. Because the features of integrated circuits is very sparse, especially for the very large scale integrated circuit. Reading and training data repeatedly will consume a lot of time. Therefore, general supervised machine-learning methods are difficult to apply. LightGBM is a gradient boosting framework that uses tree based learning algorithms. It transforms continuous eigenvalues into discrete histograms to reduce the feature dimension and improve the optimization of training speed.

D. ANALYSIS OF TIME COMPLEXITY

In our feature analysis technique, some symbols are defined firstly, which will be used in the time complexity analysis.

TABLE 1. The classification results.

HT	Type	TP	FN	FP	TN	TPR	TNR	Precision	F-measure	Accuracy
RS232-T1000	Change funcation	214	7	0	36	96.8%	100%	100%	98.4%	97.3%
RS232-T1100	Change funcation	221	3	1	33	98.7%	97.1%	99.5%	99.1%	98.4%
RS232-T1200	Change funcation	222	15	0	20	93.7%	100%	100%	96.7%	94.2%
RS232-T1300	Change funcation	227	4	0	27	98.3%	100%	100%	99.1%	98.4%
RS232-T1400	Change funcation	221	1	1	29	99.5%	96.7%	99.5%	99.5%	99.2%
RS232-T1500	Change funcation	219	0	3	37	100%	92.5%	98.6%	99.3%	98.8%
RS232-T1600	Change funcation	227	0	1	29	100%	96.7%	99.6%	99.8%	99.6%
s38147-T100	Denial of Service	5787	1	0	13	99.9%	100%	100%	99.9%	99.9%
s38147-T200	Denial of Service	5791	1	0	12	99.9%	100%	100%	99.9%	99.9%
s38147-T300	Denial of Service	5818	6	5	6	99.9%	54.5%	99.9%	99.9%	99.8%
s35932-T200	Denial of Service	6391	0	1	11	100%	91.7%	99.9%	99.9%	99.9%
s35932-T300	Denial of Service	6412	2	2	9	99.9%	81.8%	99.9%	99.9%	99.9%
s35932-T100	Leak Imformation	6391	0	2	10	100%	83.3%	99.9%	99.9%	99.9%
s38584-T200	Leak Imformation	7305	0	2	167	100%	98.8%	99.9%	99.9%	99.9%
s38584-T300	Leak Imformation	7275	0	1	976	100%	99.9%	99.9%	99.9%	99.9%

The bold values represent the best value

The sample size of data is N, the number of features is M. In LightGBM algorithm, the number of trees is K, and the depth of the tree is D.

For the Trojan features extraction stage, the time complexity calculation is very common and easy, so we do not explain them in detail. The worst time complexity is $O(N^2)$ as shown in Algorithm 1. The addition of scan-chain features has little effect on the time complexity of the whole feature extraction algorithm. Therefore, there is no significant difference in time between our feature extraction algorithm and [20] and [21].

At the Trojan nets classification stage, since the LIght-GBM algorithm requires a global sort at the beginning of the classification, the sorting complexity is $O(NMlog(N))$. And the algorithm builds a single tree complexity is $O(NMKD)$. Consequently, the overall time complexity of this step is $O(NMlog(N)) + O(NMKD) + O(N^2)$. Although the complexity looks high, our method uses a leaf-wise growth strategy with depth limitations, and it is possible that the depth D and number of trees K are extremely low. Therefore, our method reflects higher efficiency than other machine learning detection algorithms.

V. EXPERIMENTAL RESULTS

A. EXPERIMENTAL SETUP

Our framework is used to detect Trojans in Trust-hub [36] benchmark netlists. We divided them into three types: information leakage, denial of service and change of functionality. Note that, our *RG–Secure* works at RTL Trojan netlists and gate-level Trojan netlists. Therefore, it is not suitable for other HTs detection. The framework is built by the Python language. The feature extraction experiments are carried out on an Ubuntu server with an Intel i7-4720HQ central processing unit (CPU) running at 2.6GHz and 16GB memory. The classification experiments are carried out on a Win10 server with an Intel i5-6500 CPU running at 3.2GHz and 4GB memory. The details are shown in Table 2.

We evaluate the effectiveness framework using performance measure. There are five values to assess the results: the true positive rate (TPR), the true negative rate (TNR), the precision, the F-measure and the accuracy. In the process of experiment, we use cross validation method to verify the effectiveness of our method.

TABLE 2. Experimental setup.

CPU	Operating System	Memory(GB)	Language	Step
i7-4720HQ	Ubuntu	16G	C++	Feature extraction
i5-6500	Window10	4G	Python	Trojan classification

TABLE 3. The average of the classification algorithm.

Benchmark Type	TPR	TNR	Precision	F-measure	Accuracy
Change funcation	98.1%	97.6%	99.6%	98.8%	98%
Denial of Service	99.9%	85.6%	99.9%	99.9%	99.9%
Leak Imformation	100%	94%	99.9%	99.9%	99.9%

B. RESULTS OF TROJAN DETECTION

In RTL prevention, we introduce the copy comparison method [15]. So our *RG–Secure* can achieve the same effect as Rajendran *et al.* that most of HTs based on changing function can be detected in the first layer. The power overhead and area overhead of our framework are shown in the Fig.9. (Footnote: we assume that the 3PIPs from different suppliers have the same power and area). In the Trojan prevention, because there are no additional IP cores as references, its overhead are almost the same as the original design. Once synthesized with the detection constraint, our operations' average overhead is doubled from the original design. It is because a number of operations are bound to the same IPs in the RTL Trojan decetion. However, compared with the traditional high-level synthesis methods [15], [16], our security framework replaces RTL hardware Trojan isolation mechanism with scan-chain feature analysis. Because the traditional 3PIP security methods are not suitable for the HTs which implanted in the gate level. The addition of circuit characteristic analysis not only reduces the overhead of traditional security design by nearly 50%, but also improves the efficiency of HT detection. Here, we mainly verify the result of gate-level HT detection.

TABLE 4. Comparison results of the three methods.

HT	TPR			TNR			Precision			F-measure			Accuracy		
	[22]	[23]	**Ours**	[22]	[23]	**Ours**	[22]	[23]	**Ours**	[22]	[23]	**Ours**	[22]	[23]	**Ours**
RS232-T1000	24%	98.2%	**96.8%**	100%	100%	**100%**	92.3%	87.8%	**100%**	96%	93.5%	**98.4%**	99.1%	98.4%	**97.3%**
RS232-T1100	25%	98.6%	**98.7%**	78%	52.8%	**97.1%**	92.1%	82.6%	**99.5%**	94.6%	64.4%	**99.1%**	98.8%	93.4%	**98.4%**
RS232-T1200	55%	100%	**93.7%**	91%	94.1%	**100%**	100%	100%	**100%**	90.3%	97%	**96.7%**	98.1%	99.4%	**94.2%**
RS232-T1300	65%	100%	**98.3%**	86%	100%	**100%**	100%	100%	**100%**	98.2%	100%	**99.1%**	99.7%	100%	**98.4%**
RS232-T1400	15%	100%	**99.5%**	100%	100%	**96.7%**	97.6%	100%	**99.5%**	94.3%	100%	**99.5%**	98.4%	100%	**99.2%**
RS232-T1500	47%	100%	**99.6%**	82%	97.4	**92.5%**	97.4%	97.4%	**98.6%**	96.1%	97.4%	**99.3%**	99.1%	99.4%	**98.8%**
RS232-T1600	28%	98.6%	**100%**	97%	96.6%	**96.7%**	91.7%	87.5%	**99.6%**	83%	91.8%	**99.8%**	97.2%	98.4%	**99.6%**
s38147-T100	98%	100%	**100%**	83%	41.7%	**100%**	100%	100%	**100%**	15.4%	58.8%	**99.9%**	99.8%	99.9%	**99.9%**
s38147-T200	74%	100%	**100%**	93%	40%	**100%**	100%	100%	**100%**	23.5%	57.1%	**99.9%**	99.8%	99.8%	**99.9%**
s38147-T300	94%	100%	**100%**	100%	97.7%	**54.5%**	57.1%	100%	**99.9%**	27.6%	98.9%	**99.9%**	99.3%	100%	**99.8%**
s35932-T200	88%	100%	**100%**	67%	8.3%	**91.7%**	50%	100%	**99.9%**	14.3%	15.4%	**99.9%**	99.8%	99.8%	**99.9%**
s35932-T300	97%	100%	**99.9%**	100%	94.6%	**81.8%**	45.8%	100%	**99.9%**	36.1%	97.2%	**99.9%**	99.4%	100%	**99.9%**
s35932-T100	99%	100%	**100%**	80%	73.3%	**83.3%**	80%	100%	**99.9%**	40%	84.6%	**99.9%**	99.8%	99.9%	**99.9%**
s38584-T200	93%	99.9%	**100%**	91%	64.6%	**98.8%**	0%	88.2%	**99.9%**	0%	74.5%	**99.9%**	98.1%	99.3%	**99.9%**
s38584-T300	93%	99.9%	**100%**	97%	97.5%	**99.9%**	100%	99.2%	**99.9%**	4.8%	98.3%	**99.9%**	99.6%	99.9%	**99.9%**
Average	66.3%	99.7%	**99.1%**	89.7%	77.2%	**92.9%**	80.3%	96.2%	**99.8%**	54.3%	81.9%	**99.5%**	99.1%	99.2%	**99%**

The bold values represent the best value

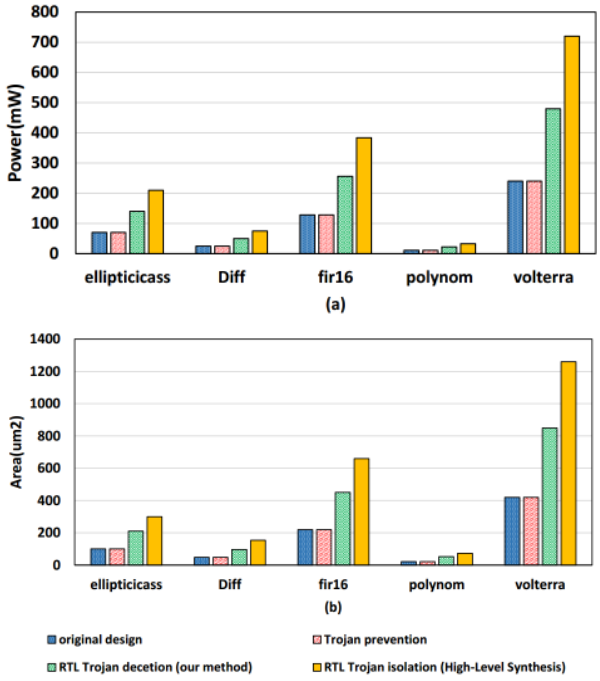

FIGURE 9. The power overhead (a) and area overhead (b) of RG-secure framework in different chip operations.

In the experiments, we analyze the performance of the proposed detection framework and compare them with the other machine learning classifiers proposed in [22] and [23]. The results of HT detection by our proposed three-level feature analysis is shown in Table 1. For the detection methods of hardware Trojan, what is essential is the recall rate (TNR) of the nets of hardware Trojan. Twelve of the 15 sets of netlist information have TNRs above 90%, only one hardware Trojan (s38147-T300) has a TNR below 80%. Because the histogram algorithm in LightGBM sacrifices certain segmentation accuracy for training speed and memory space, it is sensitive to outliers. All kind of netlists are correctly identified to be HTs in these cases. At the same time, our detection algorithm can guarantee the lowest misjudgment rate. In Table 1, all the benchmarks realized 93% or more TPRs (the probability of correctly identifying as a normal net). Especially for all netlists, the average of accuracy, precision and F-measure can reach above 99%. It can prove the effectiveness of our framework classification. The experiment results mean that HTs from all of the scan-chain features can be classification effectivly.

Then, we calculated the average value of each index, used to represent the framework in each type of hardware Trojan classification effect (in Tbale 3). As shown in Table 3, the average TPR of three kinds of HTs become 98% above. However, for HTs which are denial of service, the average TNR is reduced from 97.6% to 85.6%. It is because most of denial of service HTs are small in size, and the Trojan nets located at the boundary are easily judged as standard net mistakenly. However, for the mere detection of these HTs, the average TNR is considered acceptable. Since we have almost got 100% precision in more than half of benchmarks, the whole process of detection can ensure there is no normal net be misjudged. In terms of F-measure and Accuracy, most benchmarks reached more than 98%. As it happens, the classification method can effectively solve the defects existing in the first layer of *RG–Secure* framework.

If the detection is only at the gate-level stage, the netlists infected at the RTL will be converted into the gate-level neliset through logical synthesis firstly. It will increase the detection cost and reduce the accuracy of gate-level detection. That's why we designed the framework as two layers. However, the conclusion may not be universal because the benchmarks available in the Trust-Hub suite are very limited.

C. COMPARISON WITH OTHER METHODS

Next, the effectiveness of our method is analyzed by comparing it with other algorithms. Here, we systematically

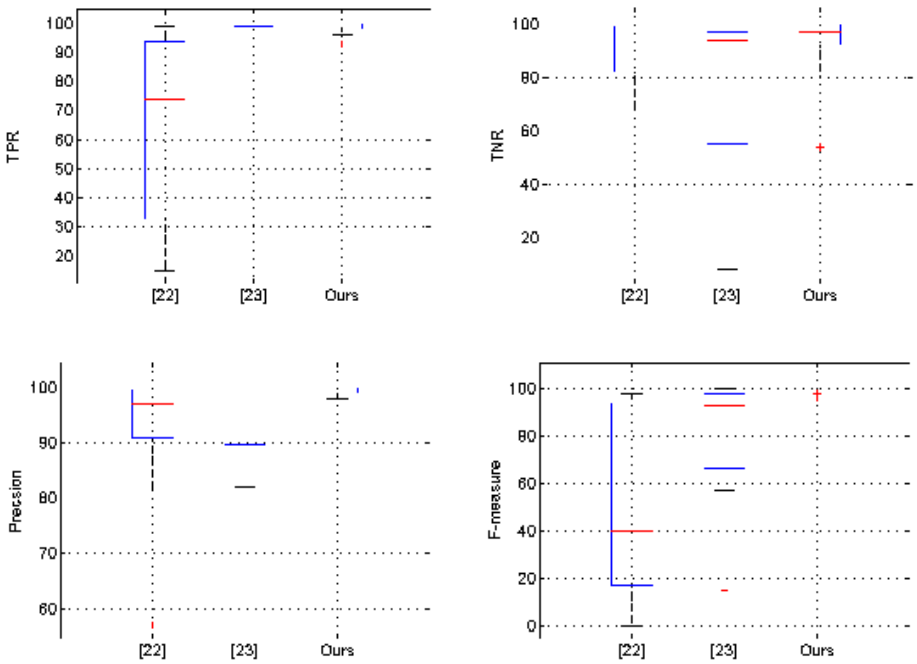

FIGURE 10. Box plots of Trojan Detection algorithms.

analyze the advantages and limitations of *RG–Secure* framework.

In the experiment, SVM [22] and RF [23] machine learning algorithms are compared with our method. Table 4 shows the comparison results of three detection methods. To show the advantage of our security method more clearly, we demonstrate the box plot of each algorithm in Fig.10 (Footnote: We ignore the accuracy, because the detection accuracy of three methods is basically the same). From the box plots, we can observe the classifier's average TPR, precision and F-measure become higher than the existing methods. When merging scan-chain features, the data distribution and median of the boxplots increased significantly in Fig.10. In particular, the true negative rate (TNR) and F-measure are obviously improved. It is because the addition of features enables the machine learning algorithm to distinguish the trojans and normal netlist more accurately. Comparing with RF, the average TNR and F-measure of our method increase by nearly 15% and 20% respectively. As for SVM, the average TNR increase by nearly 5% and average F-measure by nearly 40%. For other metrics, lightGBM can achieve the same or better results as other methods. In order to further verify the validity of scan-chain features, we applied the original 51 features [22] and scan-chain features to lightGBM algorithm at the same time to analyze their impact on the results (in Fig.11).

For the TPR in the Fig.11, the introduction of scan-chain features does not affect the classifier's judgment. In particular, for some benchmarks (RS232-T1300, RS232-T1500, s35932-T100), scan-chain features are helpful for the correct decision of the common nets. Their TPN increase by 3%-5% or so.

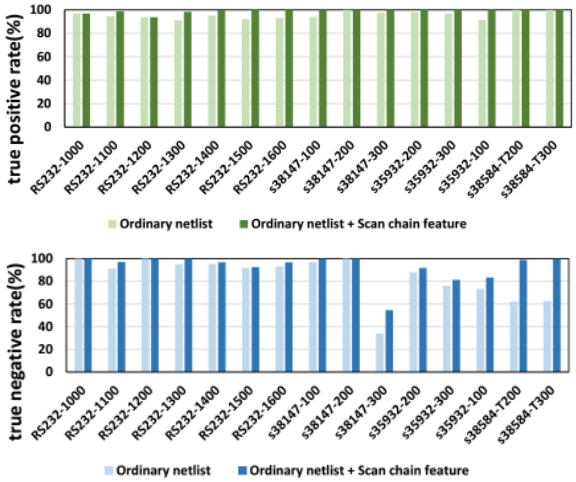

FIGURE 11. Validity analysis of scan chain features.

As for the TNR from our experimental results (Fig.11), with the addition of scan-chain features, most benchmarks improve in the detection results of HT nets. Especially for s38584-T200 and s38584-T300, their TNR increase from 62.5% and 63.1% to 98.8% and 99.9%. s38584-T200 and s38584-T300 are both information-leakage Trojans, their hidden HT trigger modules cannot be transferred to scan triggers, so they are easily identified as suspicious by the framework. It is why TNR increase significantly compared with other methods. The results can prove that the method is suitable for the second phase of the security framework, as a supplement to the first layer of RTL layer security.

Some regrets, we find that the TNR in s38147-T300 cannot reach the standard. It is because the netlist has a special structure and trigger module, which causes the s38147-T300 netlist to not be effectively classified. Small parts of nets are at the boundary between the normal netlists and the Trojan netlists. Our framework also hard to distinguish them well due to the randomness of the classifier. It is why some hardware Trojans cannot be judged correctly. In the future, we will focus on the study of individual netlists and boundary nets to maximize the TPR and TNR of our method.

VI. CONCLUSION

In this paper, we discussed the limitations in development of hardware Trojan detection technologies for modern SoC design. To improve these deficiencies, a multi-layer hardware Trojan protection framework — *RG–Secure* was proposed. Both RTL and gate-level security is guaranteed by fusing the third-party IP trusted design strategies with the scan-chain netlist feature analysis technology. Experiments showed that the method achieved up to 100% TPR and 94% TNR, furthermore, achieved 99.8% average F-measure and 99% accuracy, only one HT had a detection rate below 60%. Our *RG–Secure* framework is a promising method to ensure SoC security during the design stage.

Although the security framework is efficient and effective, there are still some disadvantages. In the future, we will not only improve the TPR and TNR of individual netlists but also design more advanced SoC security framework which can settle layout-level hardware Trojan threats.

REFERENCES

[1] R. Khan, S. U. Khan, R. Zaheer, and S. Khan, "Future Internet: The Internet of Things architecture, possible applications and key challenges," in *Proc. 10th Int. Conf. Frontiers Inf. Technol.*, Dec. 2013, pp. 257–260.

[2] F. Conti *et al.*, "An IoT endpoint system-on-chip for secure and energy-efficient near-sensor analytics," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 64, no. 9, pp. 2481–2494, Sep. 2017.

[3] A. Nourian and S. Madnick, "A systems theoretic approach to the security threats in cyber physical systems applied to stuxnet," *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 1, pp. 2–13, Jan./Feb. 2018.

[4] S. Dupuis, M.-L. Flottes, G. Di Natale, and B. Rouzeyre, "Protection against hardware trojans with logic testing: Proposed solutions and challenges ahead," *IEEE Design Test*, vol. 35, no. 2, pp. 73–90, Apr. 2018.

[5] M. Elleuchi, M. Boujeleben, M. Abid, and M. S. Bensaleh, "Securing RPL-based Internet of Things applied for water pipeline monitoring," in *Proc. 25th Int. Conf. Softw., Telecommun. Comput. Netw.*, Sep. 2017, pp. 404–410.

[6] R. Román-Castro, J. López, and S. Gritzalis, "Evolution and trends in IoT security," *Computer*, vol. 51, no. 7, pp. 16–25, 2018.

[7] L. Riliskis, H. Shafagh, and P. Levis, "POSTER: Computations on encrypted data in the Internet of Things applications," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur.*, 2015, pp. 1668–1670.

[8] M. Hicks, M. Finnicum, S. T. King, M. M. K. Martin, and J. M. Smith, "Overcoming an untrusted computing base: Detecting and removing malicious hardware automatically," in *Proc. IEEE Symp. Secur. Privacy*, May 2010, vol. 41, no. 3, pp. 159–172.

[9] J. Zhang, F. Yuan, L. Wei, Z. Sun, and Q. Xu, "VeriTrust: Verification for hardware trust," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 34, no. 7, pp. 1148–1161, Jul. 2015.

[10] F. Farahmandi, Y. W. Huang, and P. Mishra, "Trojan localization using symbolic algebra," in *Proc. 22nd Asia South Pacific Design Automat. Conf.*, Feb. 2017, pp. 591–597.

[11] 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. Automat. Electron. Syst.*, vol. 22, no. 1, pp. 6:1–6:23, 2016.

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

[13] H. Salmani, "COTD: Reference-free hardware trojan detection and recovery based on controllability and observability in gate-level netlist," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 2, pp. 338–350, Feb. 2017.

[14] M. Oya *et al.*, "Hardware-trojans rank: Quantitative evaluation of security threats at gate-level netlists by pattern matching," *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, vol. E99-A, no. 12, pp. 2335–2347, 2016.

[15] J. Rajendran, O. Sinanoglu, and R. Karri, "Building trustworthy systems using untrusted components: A high-level synthesis approach," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 24, no. 9, pp. 2946–2959, Sep. 2016.

[16] J. Rajendran, H. Zhang, O. Sinanoglu, and R. Karri, "High-level synthesis for security and trust," in *Proc. IEEE 19th Int. On-Line Test. Symp.*, Jul. 2013, pp. 232–233.

[17] A. Nahiyan, M. Sadi, R. Vittal, G. Contreras, D. Forte, and M. Tehranipoor, "Hardware trojan detection through information flow security verification," in *Proc. IEEE Int. Test Conf.*, Oct./Nov. 2018, pp. 1–10.

[18] E. Love, Y. Jin, and Y. Makris, "Proof-carrying hardware intellectual property: A pathway to trusted module acquisition," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 1, pp. 25–40, Jun. 2011.

[19] X. L. Guo, R. G. Dutta, P. Mishra, and Y. Jin, "Scalable SoC trust verification using integrated theorem proving and model checking," in *Proc. IEEE Int. Symp. Hardw. Oriented Secur. Trust*, May 2016, pp. 124–129.

[20] S. M. H. Shekarian and M. S. Zamani, "A trust-driven placement approach: A new perspective on design for hardware trust," *J. Circuits, Syst. Comput.*, vol. 24, no. 8, 2015, Art. no. 1550115.

[21] C. Liu, J. Rajendran, C. Yang, and R. Karri, "Shielding heterogeneous MPSoCs from untrustworthy 3PIPs through security- driven task scheduling," in *Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Nanotechnol. Syst.*, Oct. 2013, pp. 101–106.

[22] K. Hasegawa, M. Oya, M. Yanagisawa, and N. Togawa, "Hardware Trojans classification for gate-level netlists based on machine learning," in *Proc. IEEE 22nd Int. Symp. On-Line Test. Robust Syst. Design*, Jul. 2016, pp. 203–206.

[23] K. Hasegawa, M. Yanagisawa, and N. Togawa, "Trojan-feature extraction at gate-level netlists and its application to hardware-Trojan detection using random forest classifier," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2017, pp. 1–4.

[24] K. Hasegawa, M. Yanagisawa, and N. Togawa, "Hardware Trojans classification for gate-level netlists using multi-layer neural networks," in *Proc. IEEE 23rd Int. Symp. On-Line Test. Robust Syst. Design*, Sep. 2017, pp. 227–232.

[25] A. Waksman, M. Suozzo, and S. Sethumadhavan, "FANCI: Identification of stealthy malicious logic using Boolean functional analysis," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2013, pp. 697–708.

[26] J. Rajendran, V. Vedula, and R. Karri, "Detecting malicious modifications of data in third-party intellectual property cores," in *Proc. 52nd ACM/EDAC/IEEE Design Automat. Conf. (DAC)*, Jun. 2015, vol. 7, no. 3, pp. 1–6.

[27] C. Sturton, M. Hicks, D. Wagner, and S. T. King, "Defeating UCI: Building stealthy and malicious hardware," in *Proc. IEEE Symp. Secur. Privacy*, Jul. 2011, pp. 64–77.

[28] X. Wang, H. Salmani, M. Tehranipoor, and J. Plusquellic, "Hardware trojan detection and isolation using current integration and localized current analysis," in *Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Syst.*, Oct. 2008, pp. 87–95.

[29] M. Oya *et al.*, "A hardware-trojans identifying method based on trojan net scoring at gate-level netlists," *IEICE Trans. Fundam. Electron., Commun. Comput.*, vol. E98-A, no. 12, pp. 2537–2546, 2015.

[30] M. Oya, Y. Shi, M. Yanagisawa, and N. Togawa, "A score-based classification method for identifying Hardware-Trojans at gate-level netlists," in *Proc. Design, Automat. Test Eur. Conf. Exhib.*, Apr. 2015, pp. 465–470.

[31] R. S. Chakraborty, F. Wolff, S. Paul, C. Papachristou, and S. Bhunia, "*MERO*: A statistical approach for hardware trojan detection," in *Proc. Int. Workshop Cryptograph. Hardw. Embedded Syst.*, Aug. 2009, pp. 396–410.

[32] R. S. Chakraborty, S. Pagliarini, J. Mathew, S. R. Rajendran, and M. N. Devi, "A flexible online checking technique to enhance hardware trojan horse detectability by reliability analysis," *IEEE Trans. Emerg. Topics Comput.*, vol. 5, no. 2, pp. 260–270, Apr./Jun. 2017.
[33] N. Yoshimizu, "Hardware trojan detection by symmetry breaking in path delays," in *Proc. IEEE Int. Symp. Hardw.-Oriented Secur. Trust (HOST)*, Jul. 2014, pp. 107–111.
[34] J. Zhong, J. Wang, and H. Ding, "Temperature-variation-based hardware Trojan detection through ring oscillator," *Electron. Lett.*, vol. 52, no. 15, pp. 1302–1304, 2016.
[35] F. M. Tabrizi and K. Pattabiraman, "Formal security analysis of smart embedded systems," in *Proc. 32nd Annu. Conf. Comput. Secur. Appl.*, Dec. 2016, pp. 1–15.
[36] *Trust-HUB*. [Online]. Available: http://www.trust-hub.org

CHEN DONG received the B.S. and M.S. degrees from the College of Mathematics and Computer Science, Fuzhou University, China, in 2002 and 2005, respectively, and the Ph.D. degree in computer science from the Computer School, Wuhan University, China, in 2011. She was a Visiting Researcher with the University of California at Los Angeles, from 2015 to 2016. She is currently an Assistant Professor with the College of Mathematics and Computer Science, Fuzhou University. Her research interests include artificial intelligence, hardware security, intelligent computing, and integrated circuit physical design.

GUORONG HE received the B.S. degree in computer science and technology from the Hunan University of Chinese Medicine, in 2017. He is currently pursuing the master's degree with the College of Mathematics and Computer Science and the Fujian Provincial Key Laboratory of Information Security of Network Systems, Fuzhou University. His research interests include hardware security, intelligent computing, and artificial intelligence.

XIMENG LIU (M'–) received the B.Sc. degree in electronic engineering and the Ph.D. degree in cryptography from Xidian University, Xi'an, China, in 2010 and 2015, respectively. He is currently a Full Professor with the College of Mathematics and Computer Science, Fuzhou University. He is also a Research Fellow with the School of Information System, Singapore Management University, Singapore. He has published more than 100 papers on the topics of cloud security and big data security, including papers in the IEEE TRANSACTIONS ON COMPUTERS, the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, the IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, the IEEE TRANSACTIONS ON SERVICE COMPUTING, and the IEEE INTERNET OF THINGS JOURNAL. His research interests include cloud security, applied cryptography, and big data security. He served as a member of the ACM and CCF. He served as a Program Committee Member for several conferences, such as the 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, the 2017 IEEE Global Communications Conference, and the 2016 IEEE Global Communications Conference. He was a recipient of the "Minjiang Scholars" Distinguished Professor Award, the "Qishan Scholars" Award in Fuzhou University, and the ACM SIGSAC China Rising Star Award, in 2018. He served as a Leader Guest Editor for *Wireless Communications and Mobile Computing*.

YANG YANG received the B.Sc. and the Ph.D. degrees from Xidian University, Xi'an, China, in 2006 and 2011, respectively. She is currently an Associate Professor with the College of Mathematics and Computer Science, Fuzhou University. She has been a Research Fellow (Postdoctoral Fellow) under the supervision of R. H. Deng with Singapore Management University. She has published more than 60 papers in IEEE TIFS, IEEE TDSC, IEEE TSC, IEEE TCC, and IEEE TII. Her research interests include information security and privacy protection.

WENZHONG GUO received the B.S. and M.S. degrees in computer science and the Ph.D. degree in communication and information system from Fuzhou University, Fuzhou, China, in 2000, 2003, and 2010, respectively, where he is currently a Full Professor with the College of Mathematics and Computer Science. He currently leads the Network Computing and Intelligent Information Processing Lab, which is a key Lab of Fujian Province, China. His research interests include cloud computing, mobile computing, and evolutionary computation.