Adaptive Real-time Trojan Detection Framework through Machine Learning

Amey Kulkarni, Youngok Pino and Tinoosh Mohsenin
Department of Computer Science & Electrical Engineering
University of Maryland, Baltimore County

***Abstract*—Hardware Trojans inserted at the time of design or fabrication by untrustworthy design house or foundry, possesses important security concerns. With the increase in attacker's resources and capabilities, we can anticipate an unexpected new attack from the attacker at run-time. Therefore, the challenge is not only to reduce hardware overhead of added security feature but also to secure design from new attacks introduced at real-time. In this work, we propose a Real-time Online Learning approach for Securing many-core design. In order to prevent unexpected attacks, many-core provides feed-back to online learning algorithm based on core information and its behavior to incoming data packet. The proposed Online Learning approach updates the model run-time at each data transfer based on feed-back from many-core. For demonstration, Online Machine Learning model is initially trained with two types of (known) attacks and Trojan free router packets and then unexpected attack is introduced later at run-time. The results show that, feed-back based Online Machine Learning algorithm has 8% higher overall detection accuracy and an average of 3% higher accuracy for unexpected attacks at each interval of 1000 test records than Supervised Machine Learning algorithms. The proposed feed-back based Trojan detection framework is demonstrated using a custom many-core architecture integrated with "Modified Balanced Winnow" Online Machine Learning algorithm on Xilinx Virtex-7 FPGA. Post place and route implementation results show that, secured many-core architecture requires 4 extra cycles to complete data transfer. The proposed architecture achieves 3% reduction in area and 50% less latency overhead as compared to previous published work. Furthermore, we evaluate our framework for many-core platform by employing seizure detection application as a case study.**

Hardware Security, Trojan Detection, Many-Core Design, Machine Learning

I. Introduction

Increased focus on R&D and reduction in time-to-market window in most of the semiconductor companies a new trend has started to rely on Third Party Intellectual Properties (3PIP) and outsourcing fabrication process. This raises serious security concern about Hardware Trojan inclusions in recent years. The Trojan detection can be performed at design-time, test-time and run-time. In this paper we assume that Trojan is inserted at design or fabrication phase by untrustworthy third party vendors. Trojans inserted by an untrustworthy person in 3PIP design team or foundry can escape design and test-time detection methods.

Most of the researchers focus on Trojan detection techniques that are targeted to specific set of attacks, but increase in attackers resources and capabilities, a clever attacker can include hardware Trojan which can trigger new type of unexpected attack at run-time. Therefore, Trojan detection technique should be efficient in detecting unexpected attack. In this paper, we propose a real-time Trojan detection architecture and test setup for a custom many-core design as a case study, that considers different features to detect malicious inclusions while transferring packet from one core to the other. In supervised ML algorithms, the model is trained for expected set of attacks making it rigid to unexpected attacks. Therefore, we propose an Online learning Machine Learning technique to detect unexpected attacks at run-time.

The main contribution includes:

- Trojan Detection analysis on Supervised ML algorithms such as Support Vector Machine (SVM) and K-Nearest Neighbor (K-NN) and its comparison to Modified Balanced Winnow (MBW) Online learning algorithm.
- Feature data set generation for many-core (with 64 processing cores) architecture, feature selection based on correlation analysis and hardware implementation feasibility.
- Fully placed and routed implementation of the proposed secured many-core architecture on Xilinx Virtex-7 FPGA.
- Case study of seizure detection application mapping on many-core architecture with the proposed real-time Trojan detection framework.

II. Background

In this paper, we propose run-time anomaly detection technique for a many-core platform by using ML techniques. The proposed security framework is application and network independent, and we assume that IP cores, processing cores and memories are secured similar to Fiorin et.al. [1]. Therefore, the many-core is attacked only through communication network.

A. Many-Core Architecture and Routing Packet Generation

The many-core architecture consists of 64 cores routed through 4-ary tree architecture (with hierarchical router). The many-core architecture is scalable and details of many-core has been discussed in [2]. To communicate between cores, many-core uses *IN* and *OUT* instructions. The *OUT* instruction initiates inter-core communication. It specifies the location of data and the core number to transmit, and the location to write the data in the receiving core. Whereas, to read data from another core processor, *IN* instruction is used.

B. Trojan Insertion Methodology

The Trojan is implemented at design-time and internally activated based on number of core-to-core transfers and clock

cycles. We target Denial-of-Service (DoS) attack, where in a specific core under attack is made unavailable.

- Traffic Diversion Attack : Under this attack, the router selects a random core to transfer the data.
- Route looping attack: Under this attack, the packets are routed back to the sending core. The Sending core is made unavailable to other communicating cores, thereby causing latency in other core transfers.
- Core Spoofing Attack: This attack transfers all packets to randomly chosen (address) destination. The attack saturates the core and makes it unavailable for other cores.

III. Machine Learning Algorithms

A. Feature Extraction and Optimization

Collecting relevant data based on hardware behavior analysis is the first and most important step in this research. In a good ML data-set, each feature must contribute to the class i.e better correlation between feature and the class, but not among the features. Relevant feature selection will aid both, increasing accuracy of Trojan detection and hardware implementation. In this work, we selected following features based on feature correlation analysis:

- Source Core : Source Core ID [1]
- Destination Core : Destination Core ID [1]
- Packet Transfer Path : Packet transfer between the two cores has a unique path which alters in case of Trojan.
- Distance : At each router hop, distance *vector* is incremented by 1. For example, when core 11 is transferring packet to core 62, distance *vector* is incremented at $R0\_2$ (Distance=1), $R1\_0$ (Distance=2), $R2\_0$ (Distance=3), $R1\_3$ (Distance=4), $R0\_3$ (Distance=5). Since there will be six vertices (5 router hops and 1 processing core) and five edges, highest distance is 6.

B. Modified Balanced Winnow Algorithm (MBW)

In order to learn unexpected attacks, we implement online Trojan detection which is performed using mistake driven learning model. In mistake driven online learning model, for each test record X_t, learner makes prediction $\hat{y}_t \in \{1,-1\}$. The prediction is based on function f, which is calculated based on score and threshold (θ) parameter. After prediction learner receives true label $y_t \in \{1,-1\}$ from feedback. If the prediction is incorrect it updates the model ($w_t \rightarrow w_{t+1}$). We use "MBW" algorithm [3] for detecting unexpected attacks at run-time. It is based on three parameters, promotion parameter α, demotion parameter β, and a threshold parameter θ_{th}. The weight vector model (w_t) is a combination of two parts: positive model u_t and negative model v_t. The score function ($\hat{y}_t$) is $score = \langle X_t, u_t\rangle\langle X_t, v_t\rangle\theta_{th}$, where $\langle X_t, w_t\rangle$ is the inner product of vectors X_t and w_t.

C. Analysis of Machine Learning Algorithms

In-depth analysis of the SVM as Trojan Detection technique is performed in [4], [5]. To demonstrate detection accuracy for unexpected attack, the model is trained with two types of attacks, namely core spoofing and route looping attack. The training data consists of 7260 train records with 10 features and a binary class label, in which 33% are Trojan Free train records and 67% are Trojan infected train records. However, testing is performed with three types of attack, core spoofing, route looping and traffic diversion attack. Traffic diversion attack is considered as unexpected attack and introduced only during test. The test consists of 10055 records, in which 25% are Trojan free records and remaining are Trojan infected records. Among 7541 Trojan infected test records, 42% test records are infected by traffic diversion (unexpected) attacks. Figure 1 shows the comparison between detection accuracy of Offline and Online ML algorithms. Figure 2 shows detection accuracy after an interval of 1000 records. In each 1000 test feature records, there are 316 test feature records consisting of unexpected route looping attacks. It shows that MBW online learning algorithm performs better than other discussed ML algorithms for unexpected Trojan detection. In last interval, MBW has 96% detection accuracy for unexpected attacks. It can be observed that, there is an increase in detection accuracy of unexpected attack for MBW algorithm by an average of 3%. Overall accuracy for MBW is 5% to 8% higher than SVM and K-NN respectively. Table I shows the hardware complexity of discussed machine learning algorithms. For SVM algorithm, the complexity is calculated when the algorithm is trained offline, whereas K-NN cannot be trained offline. It can be observed that SVM and MBW have comparable complexity though MBW is an online ML algorithm.

[1] Core ID is known to only corresponding core, and router does not have knowledge of it

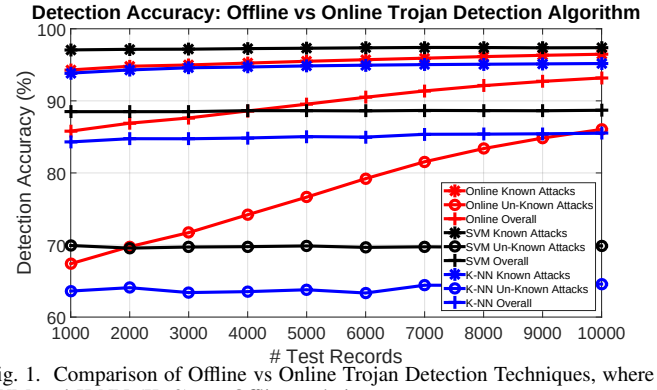

Fig. 1. Comparison of Offline vs Online Trojan Detection Techniques, where SVM and K-NN (K=3) are Offline techniques

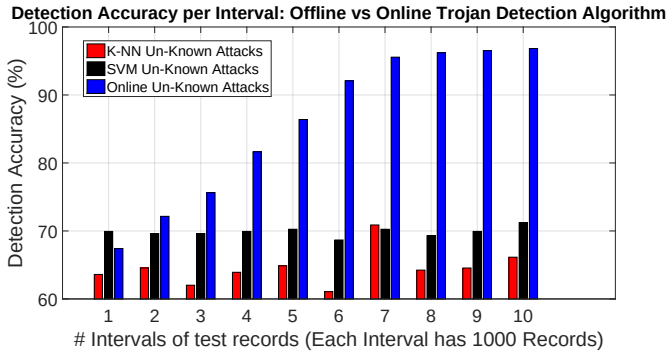

Fig. 2. Detection Accuracy Analysis per Interval for Unexpected Attacks

TABLE I
HARDWARE COMPLEXITY ANALYSIS OF ML ALGORITHMS WHEN TRAINED OFF-LINE, WHERE n SIZE OF TRAINING DATA, m SIZE OF TEST DATA AND p FEATURES

Algorithm	Multiplications	Comparisons	Memory Requirement
SVM	$p \times m$	-	p
K-NN	$p \times n \times m$	$log_2 n$	$n \times p$
MBW	$p \times m$	$3 \times m$	$2 \times p$

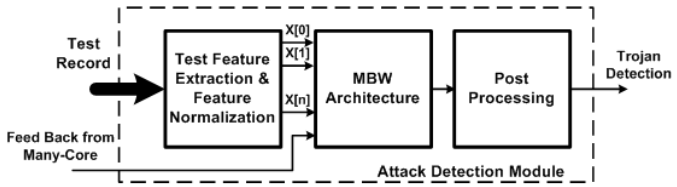

Fig. 3. Kernels of Attack Detection Module, where MBW is Modified Balanced Winnow online learning algorithm

IV. HARDWARE IMPLEMENTATION AND CASE STUDY

In this paper we propose feedback based Trojan detection framework. The efficiency of the proposed framework is demonstrated using a custom many-core architecture for a seizure detection algorithm. To detect attacks we implement "Attack Detection Module" (ADM) consisting of MBW online machine learning algorithm. It detects an attack on router packet before it enters the processing core. The Feature_Sample is transferred from the processing cores to the ADM which detects Trojan infection based on updated MBW model. In case of an attack, processing core sends the feedback, and if the decision from ADM is incorrect then it updates MBW model. In this section we discuss hardware implementation of MBW algorithm and test set up, which include interface with many-core architecture and Trojan insertion module.

A. Hardware Implementation of Modified Balanced Winnow Algorithm

The heart of the ADM is MBA algorithm. In order to reduce hardware complexity, we train model offline by using "Golden Data Set". It is created by randomly injecting Trojans in packets based on core spoofing and route looping attacks as discussed in Section II-B and labeling them based on hardware behavior. The "Golden Data Set" consists of 7260 train records with 10 features and a binary class as discussed in Section III-C. Figure 3 shows the kernels of ADM, it consists of three kernels i.e. Feature Extraction, MBW architecture, and Post Processing kernel. The architecture of the MBW algorithm is shown in Figure 4. All the parameters i.e α,β,θ and margin (M) are stored in register memory. It consists of three logic blocks i.e. Prediction logic, Feedback Control logic and Model Update logic. The Prediction Logic is used for predicting test record, which calculates the score based on positive and negative models, and threshold parameter (θ). Feedback Control Logic takes decision to update the model. It consists of three comparators, which concurrently compares feedback input from many-core with predicted output. In case of an incorrect prediction, feedback control logic enables

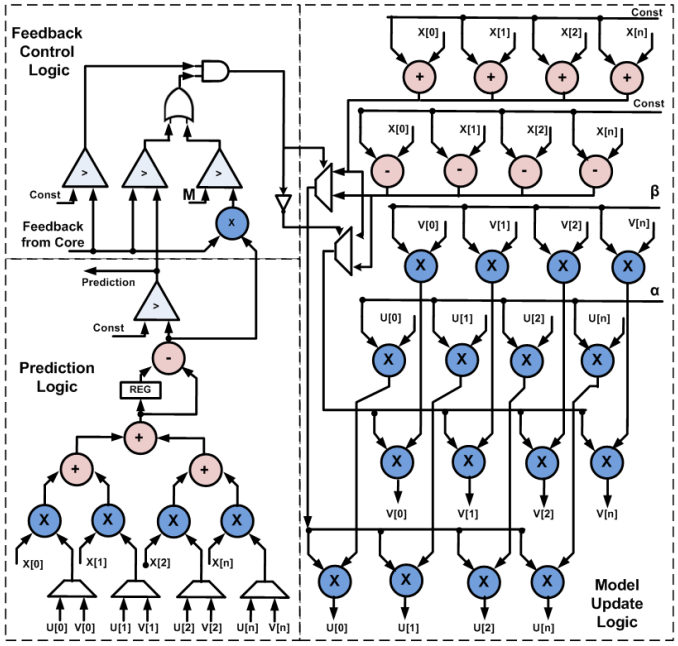

Fig. 4. Architecture for Modified Balanced Winnow (MBW) Algorithm

Model Update Logic. The Model Update Logic updates the weight vectors.

B. Case Study: Seizure Detection Application

In order to demonstrate the efficiency of the proposed security framework, we implement a bio-medical seizure detection algorithm. Seizure Detection algorithm consists of three main kernels: 1. Detection Phase 2. Analysis Phase 3. Energy Band Separation Phase. The seizure detection algorithm requires 61 cores and takes 2.1μS to execute. Mapping of seizure detection algorithm is dominated by communication instructions (56%) with Compute-to-Communication ratio of 0.8. The details of many-core implementation of seizure detection algorithm can be studied in [2].

C. Hardware Test Setup

To demonstrate detection accuracy of the proposed framework we map Seizure detection algorithm on many-core. Figure 5 shows the many-core test setup implemented on Xilinx Virtex-7 FPGA. To reduce the communication latency, we implemented Distributed ADM where each ADM targets four cluster of cores (i.e. for each 16-cores). The test setup consists of three important modules : 1. Many-Core Architecture, 2. Attack Detection Module and 3. Attack Insertion Module. The Seizure Detection algorithm generates 458 inter-cluster and 1088 intra-cluster traffic patterns for each seizure window.

In a typical security state, for each data transfer among cores Feature_Sample is updated at every router hop. At the final router hop, Feature_Sample is transferred to ADM and finally based on prediction, it either drops or accepts the packet. The state exits when it receives core enable signal from ADM. For each core to core communication, *Router Packet* is generated, which consists of two information 1. Routing information 2. Core information. Routing information has an address of the destination, which is changed under an attack. Core information contains data to be transferred,

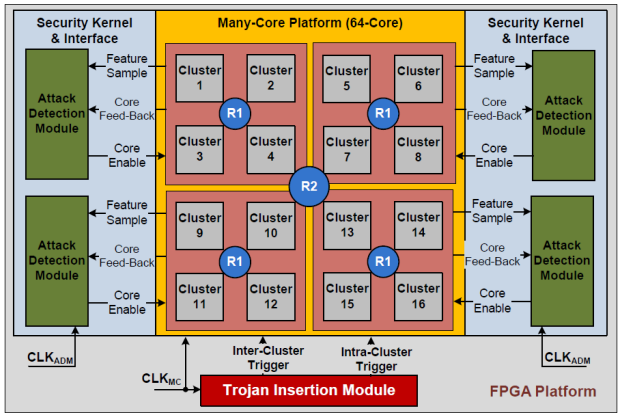

Fig. 5. Test Setup for Many-Core Platform (64-Core), where ADM-Attack Detection Module, Each cluster consists of 4 Cores, R1 - Level-1 router, R2-Level-2 router, CLK_{ADM} - Clock for ADM (188MHz), CLK_{MC} - Clock for Many-Core platform (98MHz)

Opcode generated information and Register/Memory Location. The Core information does not change in case of an attack, hence it can be used as a tool to detect an attack. However, the Core information is evaluated in core and therefore attack is detected once it corrupts the program flow. Thus, we use this tool as feedback to ADM. The Trojan is detected before it enters the processing core. Therefore, for an unexpected attack Prediction Logic fails to detect the attack until model does not learn unexpected attacks.

V. Implementation Results

A. Trojan Detection Accuracy of Online Learning Algorithm for Seizure Detection Algorithm

The Trojan detection accuracy for Seizure detection application on many-core platform is evaluated by Triggering attacks at real-time by using "Trojan Insertion Module". Seizure detection algorithm mapping execute 1546 communications for each window of 256 samples/second. We perform analysis on first 10 windows of data from each channel. For every Seizure window, we trigger Trojans on 75% of communications (i.e. total of 1169 communication instructions are attacked), among which 58% are "known" (modeled) attacks and remaining are "unexpected" attacks. Accuracy analysis over 10 windows of seizure data processing and detection on many-core platform. For the first seizure window ADM evaluates only 491 communications which are infected by "unexpected" attacks and its detection accuracy is 67%. However at the 10^{th} seizure window ADM learns pattern of "unexpected" attack through 4912 infected communication packets, thus its detection accuracy is increased to 94%.

B. FPGA Implementation Analysis

The many-core test setup comprises of 64 processing cores and hierarchical routers. The test set up is fully placed and routed in Xilinx Virtex-7 FPGA. The many-core operates at 92 MHz and packet can be transferred to the farthest core in 24 cycles. The ADM operates at 188 MHz and takes 12 cycles to execute all three logic blocks, whereas the Trojan detection takes 7 cycles, which includes execution cycles needed for Prediction Logic and Post Processing block. Table II shows area analysis for Attack Detection Module (ADM), Many-Core processor comprising 64 processing cores and hierarchical router with and without ADM to show the security overhead. The security kernel overhead is due to ADM and peripheral combinational logic. Security kernel adds 4 cycles to each data transfer. The area overhead analysis includes the additional bus for Feature_Sample. Therefore, Feature_Sample will introduce area overhead and does not affect bandwidth of the router. The seizure detection algorithm requires 5.6μS to execute with the proposed security framework at an average Trojan detection accuracy of 93%.

TABLE II
Area Analysis on Xilinx Virtex-7 FPGA

Logic Utilization	Many-Core With ADM	Many-Core Only	ADM Only	Available
Slice Count	55,132	55,072	58	75,900
Register Count	49,576	49,472	94	607,200
LUT Count	142,147	142,008	123	303,600
Distributed Memory Count	11,244	11,244	-	130,800

VI. Conclusion

In this paper we propose low overhead online learning of unexpected attacks for a custom many-core architecture. We assume that processing cores and memories are safe and anomaly is included only through communication transfers. We built training data set based on hardware feature analysis and Trojan insertion effects. Detection accuracy of unexpected attacks for 1. Support Vector Machines 2. K-Nearest Neighbors and 3. Modified Balanced Winnow Algorithms is examined. For demonstration, we train machine learning model with two types of attacks and introduce new type of attacks at run-time. Modified Balanced Winnow algorithm has 5% to 8% higher attack detection accuracy as compared to Support Vector Machines and K-Nearest Neighbors. To test our architecture, Attack Insertion Module is implemented to insert condition based attack. The architecture is fully placed and routed on Xilinx Virtex-7 FPGA. The proposed architecture takes only 4 extra cycles to detect an attack at run-time. Compared to previous published Trojan detection architecture, the proposed architecture achieves 3% reduction in area overhead.

References

[1] L. Fiorin, S. Lukovic, and G. Palermo, "Implementation of a reconfigurable data protection module for noc-based mpsocs," in *Parallel and Distributed Processing, 2008. IPDPS 2008. IEEE International Symposium on*, April 2008, pp. 1–8.

[2] A. Kulkarni, A. Jafari, C. Sagedy, and T. Mohsenin, "Sketching-based high-performance biomedical big data processing accelerator," in *Circuits and Systems (ISCAS), 2016 IEEE International Symposium on*, 2016.

[3] V. R. Carvalho and W. W. Cohen, "Single-pass online learning: Performance, voting schemes and online feature selection," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '06. New York, NY, USA: ACM, 2006, pp. 548–553.

[4] A. Kulkarni, Y. Pino, M. French, and T. Mohsenin, "Real-time anomaly detection framework for many-core router through machine learning techniques," in *ACM Journal on Emerging Technologies in Computing (JETC)*. New York, NY, USA: ACM, 2016.

[5] A. Kulkarni, Y. Pino, and T. Mohsenin, "Svm-based real-time hardware trojan detection for many-core platform," in *Quality Electronic Design (ISQED), 2016 17th International Symposium on*, March 2016.