

Intrusion Aware System-on-a-Chip Design with Uncertainty Classification

Te-Shun Chou[^], Sharon Fan[^], Wei Zhao[^], Jeffrey Fan[^], Asad Davari[¶]

[^]Department of ECE, Florida International University, Miami, Florida, USA

[¶]Department of ECE, West Virginia University Institute of Technology, Montgomery, WV, 25136, USA

Abstract— In this paper, we have proposed a System-on-a-Chip (SoC) architectural design to avoid potential intrusion or attacks from external devices. Either using misuse detection or anomaly detection techniques to design intrusion detection systems, a large amount of traffic data is needed to be collected in advance for analysis. However, it is not feasible in the limited resources available in SoC systems. We propose to incorporate fuzzy clustering technique along with Dempster-Shafer theory into our intrusion detection design to solve uncertainty problems caused by ambiguous and limited information. Also, the k-NN technique is applied to speed up the detection process. We compare the results of our proposed approach with those of k-NN classifier, fuzzy k-NN classifier and evidence-theoretic k-NN classifier. It indicates that our approach is able to achieve higher detection rates than those from the other three classifiers, thus is more useful in the implementation of intrusion aware mechanism in SoC design.

Index Terms—System-on-a-Chip (SoC), intrusion detection, fuzzy logic, Dempster-Shafer theory

I. INTRODUCTION

Nowadays, the security of computers and networks becomes extremely important. In the past, a variety of techniques have been proposed to the task of detecting the intruders' activities. They are mainly categorized into two groups: anomaly detection techniques and misuse detection techniques. Misuse detection techniques model patterns of known attacks [1] [2]. By simply matching signatures of traffic records with previous well defined attack patterns, activities can be declared as intrusions if any mismatch happens. However, it only can detect known intrusions. Whenever a novel attack is discovered, it is necessary to spend a number of hours or days on the development of this new attack signature and then to update it manually into the intrusion detection system. Moreover, the update of such signatures of intrusion is infeasible in System-on-a-Chip (SoC) architecture because of limited resources available in SoC architecture. Thus, the system is more vulnerable to any potential intrusions or attacks.

While misuse detection is achieved by modeling known attack signatures, on the contrary, anomaly detection models

normal or expected behavior of computer users. It uses techniques such as data mining [3] [4], genetic algorithms [5] [6], or neural networks [7] [8] to look for malicious activities by comparing the observed network data with acceptable learned behaviors. If the data diverge from the learned normal behavior, the data are classified as attacks.

Figure 1 is a simple demonstration of SoC architecture without intrusion. It contains some CPU (such as ARM9), a BUS system (i.e. AMBA bus), and some accessories allocated to the system. This basic illustration reveals some fundamental understanding of an overall SoC architectural construction.

Under the condition of no intrusion, the SoC system, such as H.264/AVC [9], encodes the captured incoming streaming data, i.e. video image, from the sensors outside the chip, and transmits the coded information through a USB or Wi-Fi network interface. The CPU reads the instructions from the system memory (the System Memory to CPU line in Figure-1, 2, 3 and 5) and manipulates all the working modules simultaneously.

II. SoC SYSTEM UNDER INTRUSION

In case of an attack, Figure 2 and 3 describe the potential intrusion data path and hijacked CPU program flow in SoC

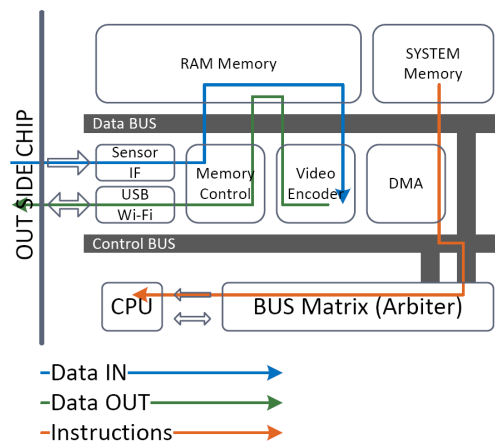


Figure 1. Default SoC Data Path (NO intrusion)

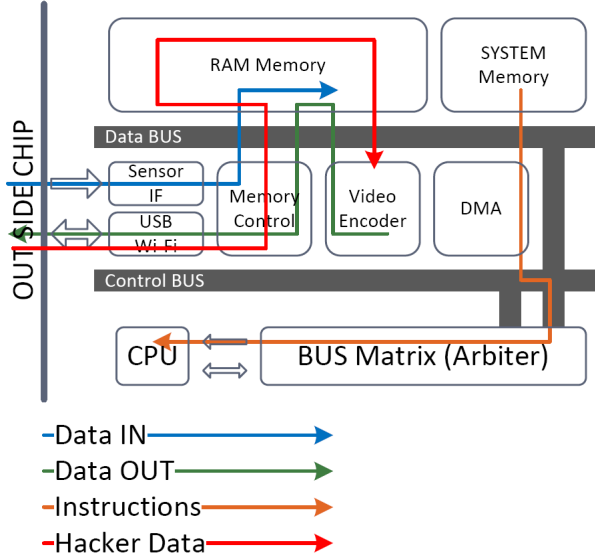


Figure 2. Potential Data Path under Attacks

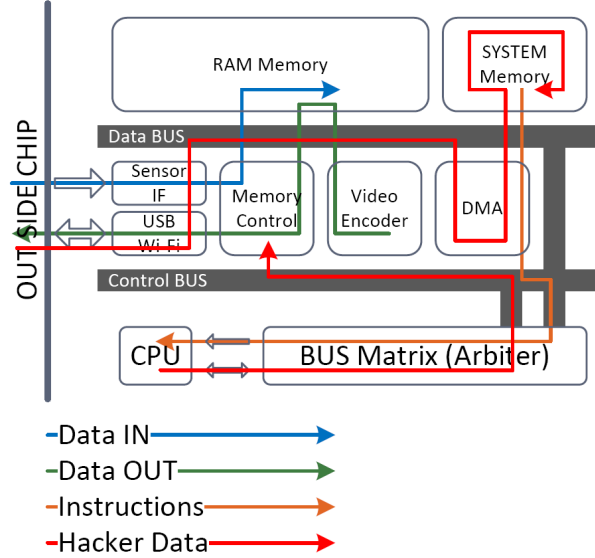


Figure 3. Potential CPU Programs under Attacks

architecture. One potential intrusion path in SoC architecture is through the bidirectional port like USB or Wi-Fi interface as shown in Figure 2. The USB and Wi-Fi ports usually have their own Direct Memory Access (DMA) to send the data to the dedicated memory with specific memory location by the address header through the BUS Matrix (Arbiter). However, in case that intrusion attacks the system, the address multiplexer may lead the “malicious code” into the memory, even into the system memory. Then, the memory is refreshed with wrong data in there. The other potential attack is that if the standard memory is affected as shown in Figure 3, the true and accurate video image may be replaced by the corrupted data. Because the corrupted data is random in nature, the video encoder may need to spend huge efforts in encoding, but output nothing except random signals. Therefore, the bandwidth is wasted and the system is constantly busy doing nothing. Things may become worse if the intrusion hijacks the program by breaking into the system memory. The CPU may follow the instructions given by the hacker. It will be easy for hackers simply to modify some register values to the memory controller. When the memory controller crashes, the system itself crashes.

III. PROPOSED APPROACH

Intrusion detection in fact is a classification task. In our work, the goal is to identify *Denial of Service (DoS)*, *Probe*, *User to Root (U2R)*, and *Remote to Local (R2L)* attacks from the intrusion detection benchmark data set, i.e. *DARPA Intrusion Detection Evaluation data set KDD99* [10]. This paper applies an intrusion detection technique, called fuzzy belief k-nearest neighbors (k-NN) anomaly detection by

using k-NN technique [11] to reduce the computation time in application to the architectural design of SoC system. The approach uses a combination of fuzzy clustering technique [12] [13] and Dempster-Shafer theory [14] [15] since both of them have merit of resolving the uncertainty problems caused by limited and ambiguous information during a decision process.

Let's assume the available information in the training set from either USB or Wi-Fi contains N network traffic connections, and each of them is composed of n distinct features with positive numeric values. We denote the training set as T , the training connection as x , and the set of features in each connection as F . The class set is L and it includes a number of p possible classes.

Because a training connection sometimes could not be crisply defined as normality or abnormality in classification, we apply fuzzy c-Means clustering technique to deal with the above uncertainty. When the clustering operation is finished, we can obtain a set of cluster centers C and a membership partition matrix U . Within a vector (connection) of U , the membership grades are treated intuitively to be our degrees of confidence on classes that a connection can belong to. Consequently, we can build p decision rules from a connection and each one consists of a number of feature values F , a class label l , and a confidence value α .

$$R_U = \{r_U\} \text{ where } r_U : \langle F_i, l_j \rangle, \alpha \quad (1)$$

where i is the connection number and j is the class number. The confidence values are in proportion to the correspondent membership grades that connection belongs to certain classes.

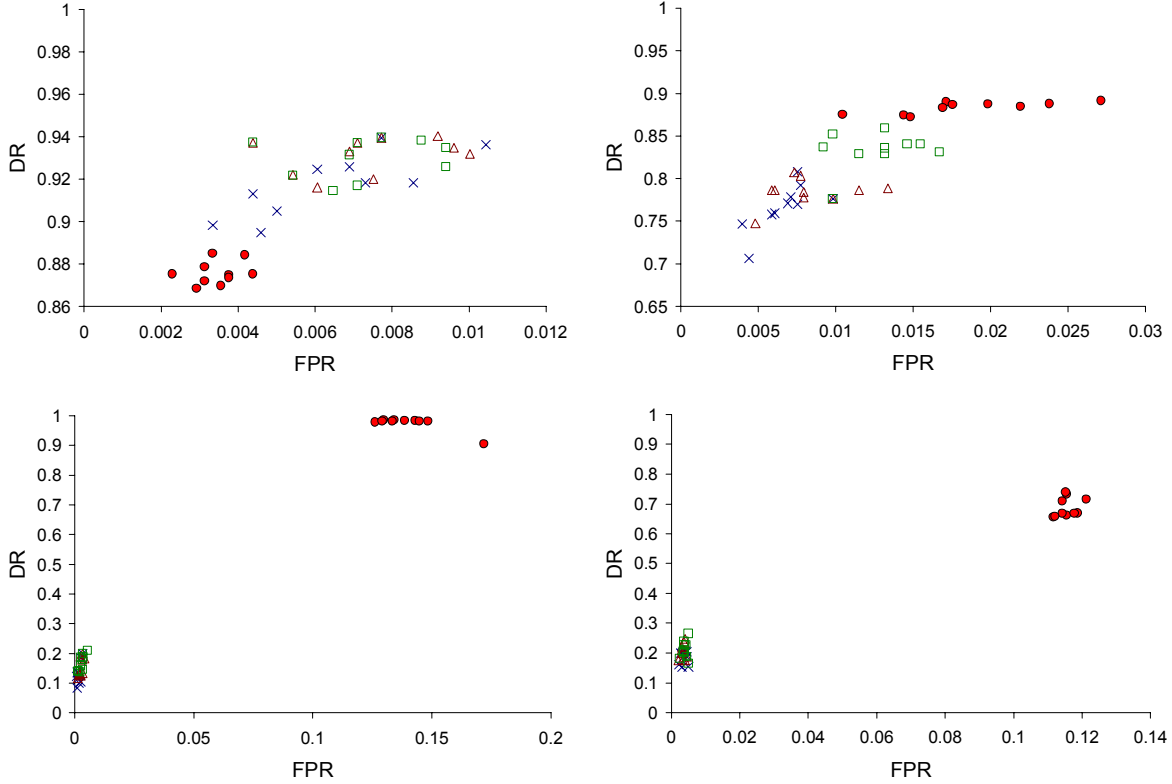


Figure 4. ROC Graphs of *Normal-DoS* (top left), *Normal-Probe* (top right), *Normal-U2R* (bottom left), and *Normal-R2L* (bottom right) Data Sets
x: k-NN, Δ : Fuzzy k-NN, \square : Evidence-Theoretic k-NN \bullet : Fuzzy Belief k-NN

In addition to the rules created from membership partition matrix U , a number of p rules are generated from the cluster centers. In each rule, the antecedent part includes n values of a cluster center and the corresponding class label. The degree of confidence is designated to 1 because we have full confidence that the cluster center should belong to that partitioned class without any doubt.

$$R_c = \{r_c\} \text{ where } r_c : \langle c_j, l_j \rangle, \alpha = 1 \quad (2)$$

Now let's assume v be an incoming connection to be classified. In order to classify it into the correct class, Dempster-Shafer theory [14] [15] is used to measure and combine pieces of evidence derived from the set of decision rules. This theory starts by defining the set of class labels L as the *frame* of the problem domain. The possible subset A of L represent hypothesis that one could present evidence. To classify v means to assign it to one of members in L .

For classifying v into the correct class, we treat the set of decision rules as pieces of evidence that alters our degrees of belief on which class v should belong to. If the distance is large between v and a decision rule, it implies that the rule only has a little influence on v . On the other hand, we have stronger belief that v should belong to the same class of the rule if v is "close" to it, which means the distance has a smaller value. Here, we apply k-NN rule to find the most

informative k nearest training connections of v . By using these k connections, we then can find the corresponding decision rules. Also, we use weighted k-NN rule [16] to assign different weights w to these rules in order to differentiate the degrees of importance. By adapting Dempster-Shafer theory, the degree of belief is quantified by *mass function* which is denoted as m .

$$m(l_q) = w \cdot \alpha \quad (3)$$

where q is the class number. Up to this stage, each rule creates a belief assignment indicating the degree that v belongs to a certain class. Nevertheless, we need to notice that a belief should also be designated to the frame (with every class labels). The reason is that only part of our beliefs is committed to single classes for a given training connection, and the rest of our belief should be assigned to the whole class set. According to Dempster-Shafer theory, the summation of all mass functions inferred from one training connection is equal to 1. Thus, the belief belonged to the frame becomes one minus the summation of beliefs of all single classes.

$$m(L) = 1 - \sum_{i=1}^p m_i(l_q) \quad (4)$$

Generally speaking, the mass function is a piece of evidence that supports certain hypothesis concerning to the class

member of a rule. When more evidences appear with same class label, these evidences can be integrated to generate a single belief function which represents the total support for the same class. Now assume that there are two mass functions m_1 and m_2 induced by distinct items of evidences X and Y . By using *Dempster Rule of Combination*, these two independent evidences can be fused into a single belief function that expresses the support of the hypotheses in both evidences. The combination result is called *orthogonal sum* of m_1 and m_2 and noted as $m = m_1 \oplus m_2$.

$$m(Z) = \frac{\sum_{X \cap Y = Z} m_1(X) \cdot m_2(Y)}{\sum_{X \cap Y \neq \emptyset} m_1(X) \cdot m_2(Y)} = \frac{\sum_{X \cap Y = Z} m_1(X) \cdot m_2(Y)}{1 - \sum_{X \cap Y = \emptyset} m_1(X) \cdot m_2(Y)} \quad (5)$$

After having all the *fused mass functions*, the final decision is made by introducing the *pignistic probability function*. [14] [15] It is illustrated as follows:

$$Bp(l_q) = m(l_q) + \frac{m(L)}{p} \quad (6)$$

where q is the class number and p is the number of classes. The function quantifies our beliefs to individual classes with pignistic probability distribution. For making an optimal decision, v is assigned to a class with the highest pignistic probability. Hence, the incoming patterns of information can be classified as intrusion or non-intrusion activities in SoC systems.

IV. EXPERIMENTAL RESULTS

The data for our experiments is the *KDD99* data set that is made up of a large number of network traffic connections. Each connection is represented with 41 features plus a label of either normal or a type of attack. Totally 39 attack types are included and are fall into four main classes, *DoS*, *Probe*, *U2R*, and *R2L*. The sizes of the original training and testing sets are reduced by removing the duplicated connections.

The experiments are performed on the binary (normal/attack) classification. To minimize the inaccuracy and variation factor of experiment results, 10 trials are performed in every detection task. In each trial, only a very small amount of connections are randomly selected from reduced training and testing sets. It not only speeds up the classification process but also simulates the uncertainty caused by lack of network traffic information. The four training sets have 545 *DoS* attacks, 213 *Probe* attacks, 52 *U2R* attacks, and 99 *R2L* attacks, respectively and each set has a number of 878 normal connections. The four testing sets have 235 *DoS* attacks, 268 *Probe* attacks, 215 *U2R* attacks, and 291 *R2L* attacks, respectively and each set has 479 normal connections.

For detecting the attacks, training and testing are performed in each trial. In the training phase, the four classifier, k-NN [11], fuzzy k-NN [17], evidence-theoretic k-NN [18], and fuzzy belief k-NN, are constructed using the limited and ambiguous training data. The testing data are

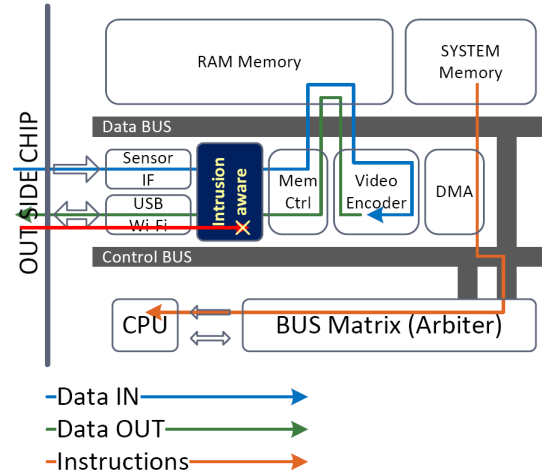


Figure 5. Intrusion Aware Module in SoC design

then fed into the trained classifier to identify intrusions in the testing phase.

We evaluate the performances using distinct numbers of k nearest neighbors, false positive rate (FPR) and detection rate (DR). The false positive rate is the percentage of normal connections that are incorrectly identified as attacks. The detection rate is the percentage of attacks that are correctly identified. Figure 4 shows the Receiver Operating Characteristics (ROC) graphs that plot FPRs on the X axis and DRs on the Y axis.

In the tasks of detecting *DoS* and *Probe* attacks, the differences in both FPRs and DRs are very slight for k-NN, fuzzy k-NN, and evidence-theoretic k-NN classifiers. Since *DoS* and *Probe* attacks usually have frequent sequential patterns that are different from the normal connections, they can be easily separated from normal activities and thus all of three classifiers can achieve low FPRs and high DRs. On the contrary, *U2R* and *R2L* attacks do not have any intrusion only frequent sequential patterns. They are embedded in the data portions of the packets and normally involve only a single connection. Also, many attacks in *Normal-U2R* and *Normal-R2L* testing sets do not exist in their corresponding training sets. Therefore, most machine learning approaches would fail to achieve high DRs. In our experiments, those three classifiers only achieve around 15% and 20% DRs in *U2R* and *R2L* attacks, respectively. Although they have very low FPRs, it is because they treat most network traffic data as normal connections either they are normal or malicious activities.

In summary, the overall performances of fuzzy belief k-NN classifier are better than those of other three k-NN based classifiers in detecting four categories of attacks. By applying such uncertainty classification into the proposed SoC system, we may detect the potential intrusion activities earlier, thus manage to do some counter-managements. As shown in Figure 5, we propose to add an “intrusion aware” module between USB/Wi-Fi module and memory controller. Some examples of these counter-attacks include limiting the

Wi-Fi port from bi-directional to unidirectional, cutting of intrusion path in the video compression, isolating the use of system memory, and sending signals to alert other systems from attacks, etc.

V. CONCLUSIONS

In this paper, we have proposed SoC architectural designs based on the fuzzy belief k-NN anomaly detection technique. During the decision process, ambiguous of users' activities were imitated by fuzzy clustering technique. Uncertainty caused by limited information was simulated by incorporating only a small amount of network traffic data for analysis and solved by the use of Dempster-Shafer theory. For verifying the performance of our proposed classifier, the other three k-NN based classifiers, k-NN, fuzzy k-NN and evidence-theoretic k-NN, were compared. The experimental results demonstrate that our approach has a superior performance to the other three classifiers, thus may be applied to SoC design to prevent potential intrusions or attacks.

REFERENCES

- [1] N. Bashah, I. B. Shanmugam, and A. M. Ahmed, "Hybrid Intelligent Intrusion Detection System," *Transactions on Engineering, Computing and Technology*, vol. 6, pp. 291-294, June 2005.
- [2] M. Sabhnani and G. Serpen, "KDD Feature Set Compliant Heuristics Rules for R2L Attack Detection," *International Conference in Computer Security and Management*, Las Vegas, Nevada, pp. 310-316, June 2003.
- [3] S. Chebrolu, A. Abraham, and J. P. Thomas, "Feature Deduction and Ensemble Design of Intrusion Detection Systems," *Computers Security*, vol. 24, no. 4, pp. 295-307, 2005.
- [4] W. Lee and S. J. Stolfo, "A Framework for Constructing Features and Models for Intrusion Detection Systems," *ACM Transactions on Information and System Security*, vol. 3, no. 4, pp. 227-261, 2000.
- [5] W. Li, "Using Genetic Algorithm for Network Intrusion Detection," *Proceedings of the United States Department of Energy Cyber Security Group 2004 Training Conference*, Kansas City, Kansas, CD-ROM Proceedings, May 2004.
- [6] D. Song, M. I. Heywood, and A. N. Zincir-Heywood, "Training Genetic Programming on Half a Million Patterns: An Example from Anomaly Detection," *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 3, pp. 225-239, 2005.
- [7] K. M. Faraoun and A. Boukelif, "Neural Networks Learning Improvement Using the K-Means Clustering Algorithm to Detect Network Intrusions," *International Journal of Computational Intelligence*, vol. 3 no. 2, pp. 161-168, 2006.
- [8] L. Vokorokos, A. Balaz, and M. Chovanec, "Intrusion Detection System Using Self Organizing Map," *Acta Electrotechnica et Informatica*, vol. 6, no. 1, 2006.
- [9] J. V. T. of ITU-T and I. J. 1, "Draft itu-t recommendation and final draft international standard of joint video specification (itu-t rec. h.264 iso/iec 14496-10 avc)," Document JVT-GO50, December 2003.
- [10] KDD99 archive: The Fifth International Conference on Knowledge Discovery and Data Mining.
URL: <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>
- [11] E. Fix and J. L. Hodges, "Discriminatory Analysis: Nonparametric Discrimination: Consistency Properties," Report Number 4, Project Number 21-49-004, USAF School of Aviation Medicine, Randolph Field, Texas, 1951.
- [12] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, New York, 1981.
- [13] J. C. Dunn, "A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters," *Journal of Cybernetics*, vol. 3, pp. 32-57, 1973.
- [14] A.P. Dempster, "A Generalization of Bayesian Inference," *Journal of the Royal Statistical Society, Series B*, vol. 30, pp. 205-247, 1968.
- [15] G. Shafer, *A Mathematical Theory of Evidence*, Princeton, University Press, Princeton, NJ, 1976.
- [16] S. A. Dudani, "The Distance-Weighted k-NN Rule," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 6, no. 4, pp. 325-327, 1976.
- [17] M. Keller, M. R. Gray, and J. A. Givens Jr., "A Fuzzy k-Nearest Neighbor Algorithms," *Transactions on Systems, Man and Cybernetics*, vol. SMC-15(4), pp. 580-585, 1985.
- [18] T. Denoeux, "A k-Nearest Neighbor Classification Rule Based on Dempster-Shafer Theory," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 25, no. 5, pp. 804-813, May 1995.