URSI GASS 2021, Rome, Italy, 28 August - 4 September 2021

Using Dynamic Operational features to Identify Embedded Devices

Pooja R. Khanna* (1) and Gareth Howells (1)
(1) University of Kent, UK, CT2 7NT, pk327@kent.ac.uk

Abstract

This paper investigates the use of system memory features as the characteristics of simpler embedded devices. The ICmetrics technology is based on the hardware and software features of the systems in order to generate the unique identifiers. Usually memory features are considered very volatile as they are considered very unstable usually. But, in this case the memory features are very crucial as they tell us more about the system function. Hence, these features were used for analysis and a multivariate gaussian model was created and validated by comparing testing sample against the training data with 98% accuracy amongst 6 devices. This confirms the usefulness of the system memory features in adding a layer of security to the devices.

1 Introduction

In the recent years, the world has seen an exponential growth in the IoT sector which means there has been a tremendous increase in the connected devices. As the products are getting cheaper, simpler and accessible, they are more prone to cyber/hacking attacks. Hence, the need to secure the IoT devices is of a grave concern. The ICMetrics technology is one of the novel techniques to generate an encryption key based on the hardware and software characteristics for the embedded devices [1]. The other encryption algorithms use the pre-stored encryption keys/identifiers which would be easily accessible if used on simpler embedded technologies. Hence, the ICMetrics (Integrated Circuit metrics) extracts the hardware/software information from the system. This technique captures the data from the system from its own environment, working in various states which ensures the detection of anything infecting the system via changes observed in the running system [2].

This technique is mainly used in complex, off-the shelf devices but this paper explores using this technique for rather simpler embedded systems to understand if these devices can be classified by the hardware information derived from it. ICMetric procedure is divided in two main sections [3]:

1. **Calibration Phase:**
    a. This Phase occurs just once with each of the devices.
    b. Measure the features we need and make the feature distributions
    c. Generate normalization maps for collected features.
2. **Operation Phase:**
    a. Capturing the features from the devices whose encryption key is needed.
    b. Generate the suitable values for creating unique identifiers by applying normalization maps.
    c. Apply the key generation method.

In this paper we explore the possibility of identifying simpler embedded device like raspberry pi. The Raspberry pi is chosen as it supports close to limitless applications and is a simpler device than ones used before for ICMetrics.

In this paper we investigate if the simpler devices like raspberry pi can be uniquely identified based on their hardware characteristics. By measuring these features, we evaluate if we are 1. Able to uniquely identify the device by designing a simple model and 2. If step 1 is successful, generate the unique key. Here, we are only exploring step 1 as we are using raspberry pi's memory usage features, as it would be a point of reference for all the devices and we would have same features for each one. As applications vary, if sensor data is used, the sensor data cannot be compared if we have different sensors for two devices.

The following sections of the paper describes our hardware, software and the conditions the data was collected in. The analysis of the data collected, building a model and how the classifier works. Lastly, the conclusion section also suggests the direction of exploration for forthcoming work.

2 Experimental Setup

This section contains the entire setup of the data collection. A software application, code written to collect the features, storing the data locally and later extracting for more statistical analysis.

2.1 Software Used

The Raspberry Pi has a Linux based Operating System.

- Data is collected from the pi via Node-Red software which does the logging.
- Raspberry Pi 2 with 16GB SanDisk Memory card.
- 6 devices used with updated Raspbian desktop version
- Data saved in .txt form and exported to excel for statistical processing i.e. calculating covariance, mean, correlation

- Python for data cleaning and excel for removing the characters from the

The figure below shows the code/data logging flow which will then be utilized to extract the useful features and do the processing. For each device, this is the calibration phase which collects the data.

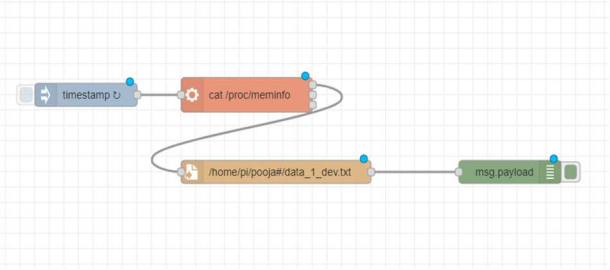

Figure 1 Node-Red flow for data collection

2.2 Feature Analysis

The pre-analysis was done in the excel. In total, 37 memory features were extracted. Some of them were static i.e. very little use. Here we extract the system memory features as it gives us an insight on when the system is being overused or idle, as Pi constantly tries to free its unused space so makes the device efficient. Hence, identifying and evaluating the useful features from the data collected. The features selected must be uniform amongst all the devices. The correlation is of high significance while choosing the features as they provide higher level complexity than the singular features [4]. The embedded memory run-time usage data is investigated and is mapped to generate the normalization maps. We collected about 2000 samples from each device and will randomly divide the data into 80:20 ratio for training and test set.

Table 1-6 show the simple mean and correlation between the features for Device 1-6 respectively:

Table 2. Correlation between features and their respective mean for Device-1.

Features	F1	F2	F3	F4	Mean
F1	1				313098
F2	0.799446	1			67288.4
F3	-0.02049	0.204417	1		48859.4
F4	-0.70584	-0.48270	0.08581	1	32089.4

Table 2. Correlation between features and their respective mean for Device-2.

Features	F1	F2	F3	F4	Mean
F1	1				501050
F2	0.2595487	1			150545.
F3	0.0443142	0.852108	1		49988.3
F4	-0.404908	-0.82359	-0.54614	1	30490.0

Table 3. Correlation between features and their respective mean for Device-3.

Features	F1	F2	F3	F4	Mean
F1	1				713596
F2	-0.027850	1			151854
F3	-0.861135	0.501389	1		45067.4
F4	-0.956858	0.174126	0.936759	1	26630.5

Table 4. Correlation between features and their respective mean for Device-4.

Features	F1	F2	F3	F4	Mean
F1	1				171088
F2	0.884214	1			51039.3
F3	0.763742	0.840440	1		32834.3
F4	-0.80707	-0.89515	-0.84112	1	19580.9

Table 5. Correlation between features and their respective mean for Device-5.

Features	F1	F2	F3	F4	Mean
F1	1				352285
F2	0.8361488	1			112156
F3	0.435537	0.696946	1		35627
F4	-0.885897	-0.89783	-0.48535	1	19187.7

Table 6. Correlation between features and their respective mean for Device-6.

Features	F1	F2	F3	F4	Mean
F1	1				346930
F2	0.9091056	1			165940
F3	0.9266794	0.945521	1		40081.5
F4	-0.940902	-0.97379	-0.94095	1	19445.1

Here the four features here are derived from the system memory hence, even though they are all raspberry Pi's we have a variety of correlations between the features based on their operation. The 'mean' of the features F1, F3 and F4 are similar for a few devices so the data can be wrongly classified. Hence it is important to check the nature of the distribution. We also used the standard multivariate gaussian classifier, which assumes the distribution is unimodal and got between 50-60% accurately classified test data. Therefore, a necessity to check the nature of the distribution.

3 Feature Mapping Technique

This section provides the information on how to map the feature values to potentially identify the device based on its behavior.

3.1 Feature normalization and distribution

Initially we generate the normalization maps to detect an identifier for an 'x' named device. This will later be classified as one of the known/unknown device data. The main concept of creating a normalization map is to map the

collected data onto a multidimensional space. In [5] generated normalization maps specific feature to a vector and concatenating them via linear mapping. As we know all the samples are inter-related, i.e. can be analyzed independently to gain information but cannot be used independently. The next step is to find the probability distribution of every feature individually for all the devices.

To understand the distribution of the features, we create the probability density function graphs for each feature, which signifies the data is unimodal, bimodal or multimodal in nature [6]. These multimodal features can be interpreted as different operational phases of the circuit. Incorporating the highly multi-modal features are a vital part of employing an ICMetrics based system [7-9].

3.2 Addressing Multimodality

Multi-modal features are normalized, so they can be treated as the gaussian distributions. In papers [5,10] the mapping was linear in nature. For the multi-dimensional space, we use peak – trough detection [11] to separate the modal clusters and converting the data into gaussian format. This will create multiple gaussians and each mode acts as a distribution of its own [12]. Considering multi-modal features that are derived from raspberry Pi's hardware, will allow to create a generalized scenario and make the classification/identification more efficient.

The table below shows the number of modes each feature of each device. This gives an insight of what the modal boundaries look like after the data has been applied to a peak - trough algorithm.

The paper describes the data from 6 Raspberry Pi's. Four features are selected from the recorded 37 features. Feature values for each circuit take a unique subset of values. Some of the features might overlap but have a different correlation with the other samples. Hence, we divide the dataset into training and testing samples to calibrate the system.

3.3 Identifying modal Clusters

Identifying modal clusters per distribution is the first step of classification. Each of the sample set are read and checked if they belong to one of the distributions. Every value from the set is read one by one from beginning and checked which distribution (device) it belongs to. If they do belong to that distribution, the value will be identified/belong to one of the modal clusters from that device. When a feature set from different device is encountered, we will not find the values being identified or belonged to device 1 so all the clusters are exhausted. So, moved on to comparing with the next device distribution and this will be done for all the test feature sets. Exhausting all the test sets, there will some left which are not allocated within the known sample set. Hence, most plausible device is allotted using probabilities calculated by each of the generated Gaussian distributions. Similarly, used in paper

The necessary parameters like the Mean, Variance and Probability function for each discrete sample set. The equations are used to generate the Multivariate gaussian model in-order to get the probability against each training set to assign the device number to it.

$$\overline{X} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{1}$$

$$\sigma^2 = \sum_{i=1}^{N} P(x_i)(x_i - \mu)^2 \tag{2}$$

$$P(x) = \frac{1}{\sigma\sqrt{(2\pi)}} e^{\frac{-(x-\mu)^2}{(2\sigma^2)}} \tag{3}$$

The Figure 2 below presents the step by step approach to implement the classifier which takes care of n-modal data.

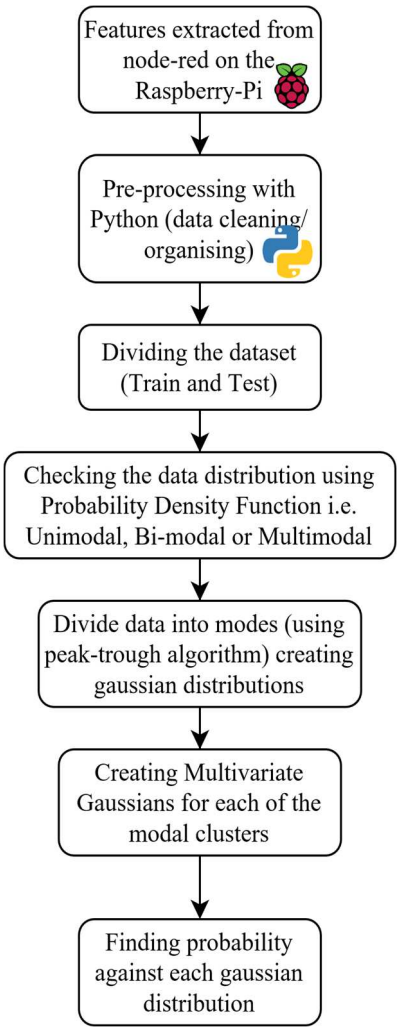

Figure 3. Flowchart to implement Classifier.

5 Conclusion

This paper explores the system memory usage data as ICMetrics features. Memory features are usually volatile, but for smaller embedded devices, these features are very crucial as they are running only targeted applications when deployed. We also observed that it is important to understand the nature of distribution per feature, to identify and classify the data correctly. There is a need to properly deploy modal clusters as separate gaussians to effectively take in account the multi-modal nature of the data to uniquely identify the system. This data will be used to produce effective encryption keys soon to add a layer of security on the simpler embedded devices.

6 References

1. Y. Kovalchuk and K. McDonald-Maier and G. Howells, 'Overview of ICmetrics Technology – Security Infrastructure for Autonomous and Intelligent Healthcare System.' International Journal of u- and e- Service, Science and Technology, 4. 49 - 60. ISSN 2005-4246, 2011.

2. R. Tahir and K. McDonald-Maier, "Improving Resilience against Node Capture Attacks in Wireless Sensor Networks using ICMetrics," in Emerging Security Technologies (EST), 2012 Third International Conference on, 2012, pp. 127–130.

3. R. Tahir, H. Hu, D. Gu, K. McDonald-Maier, and G. Howells"Resilience against brute force and rainbow table attacks using strong ICMetrics session key pairs," in Communications, Signal Processing, and their Applications (ICCSPA), 2013 1st International Conference on,2013, pp. 1–6.

4. S. Yadav and G. Howells, "Analysis of ICMetrics features/technology for wearable devices IOT sensors," 2017 Seventh International Conference on Emerging Security Technologies (EST), Canterbury, 2017, pp. 175-178, doi: 10.1109/EST.2017.8090419.

5. G. Howells, E. Papoutsis, A. Hopkins and K. McDonald-Maier, "Normalizing Discrete Circuit Features with Statistically Independent values for incorporation within a highly Secure Encryption System," Second NASA/ESA Conference on Adaptive Hardware and Systems (AHS 2007), Edinburgh, 2007, pp. 97-102, doi: 10.1109/AHS.2007.78.

6. S. D. Baba, S. Yadav and G. Howells, "SortAlgo-Metrics: Identification of Cloud-Based Server Via a Simple Algorithmic Analysis," 2019 Eighth International Conference on Emerging Security Technologies (EST), Colchester, United Kingdom, 2019, pp. 1-6, doi: 10.1109/EST.2019.8806214.

7. S.W. Baik, R. Baik: Adaptive image classification for aerial photo image – Article – Computer Science, Artificial Intelligence – AI 2004: Advances in Artificial Intelligence, Proceedings Lecture Notes In Artificial Intelligence 3339: 132 – 139 2004.

8. T. Kubota: Massively parallel networks for edge localization and contour integration – adaptable relaxation approach – Article – Computer Science, Artificial Intelligence – Neural Networks 17 (3): 411 – 425 APR 2004.

9. C. Schmid: Weakly supervised learning of visual models and its application to content-based retrieval – Article – Computer Science, Artificial Intelligence – International Journal Of Computer Vision 56 (1-2): 7 – 16 Sp. Iss. SI, JAN-MAR 2004.

10. E. Papoutsis, W. G. J. Howells, A. B. T. Hopkins, and K. D. McDonald-Maier, "Integrating Feature Values for Key Generation in an ICmetric System", IEEE NASA/ESA Conference on Adaptive Hardware and Systems (AHS-2009), 2009, p. 5.

11. K. Harmer, G. Howells, W. Sheng, M. C. Fairhurst, and F. Deravi, "A Peak-Trough Detection Algorithm Based on Momentum", International Congress on Image and Signal Processing (CISP2008). Sanya, Hainan, China, May 2008

12. S. Yadav and G. Howells, "Secure Device Identification Using Multidimensional Mapping," 2019 Eighth International Conference on Emerging Security Technologies (EST), Colchester, United Kingdom, 2019, pp. 1-5, doi: 10.1109/EST.2019.8806218.

13. E. Papoutsis, G. Howells, A. Hopkins and K. McDonald-Maier, "Integrating Multi-Modal Circuit Features within an Efficient Encryption System," Third International Symposium on Information Assurance and Security, Manchester, 2007, pp. 83-88, doi: 10.1109/IAS.2007.27.