Secure Device Identification Using Multidimensional Mapping

Supriya Yadav
School of Engineering and Digital Arts
University of Kent
Canterbury, UK
sy227@kent.ac.uk

Gareth Howells
School of Engineering and Digital Arts
University of Kent
Canterbury, UK
W.G.J.Howells@kent.ac.uk

Abstract— **In this paper we investigate several potential hardware features from multiple devices for suitability during the employment of a device identification. The generation of stable and unique digital identity from features is challenging in device identification because of the unstable operation environments that implies the features employed are likely to vary under normal operating conditions. To address this, we introduce a novel multi-dimensional key generation technology which maps from multi-dimensional feature space directly to a key space. Furthermore, normalized distributions of features give the necessary data to model the characteristics, from which we derive intra-sample device feature distributions, and correlate the distinct features to generate a secure key to identify the device.**

Keywords—Security, ICMetric ,Authentication ,Key generation,Multidimensional space.

I. INTRODUCTION

In common cybersecurity parlance, the importance of strong authentication is more than ever before. Regulatory requirements, such as EU eIDAS, are also mandate robust authentication. Cybercrime cost the global economy as much as $600 billion in 2017 and a significant part of these attacks are related to weak authentication in one way or the other [16]. Under the current circumstances, traditional approaches like hardware tokens are expensive to deploy and manage and are ineffective against some threats [16]. The challenge we face is deploying a technology that is both easy to use yet strong enough to protect against sophisticated attacks like malware, Man-in-the-Middle etc. ICMetrics is a secure software credential that combines protection for digital identities like that of a hardware smart card with ease of use, ease of distribution, and lower costs for deployment and maintenance. The ICMetrics is the 'something you have' and ICMetrics password (optional) is the 'something you know' necessary for two-factor authentication. As a software-based solution, the ICMetrics enables organizations to leverage the advantages of Public-Key Infrastructures (PKI) without the expense and management issues inherent with hardware-based secure key storage. ICMetrics can also operate and offer similar level of security in a non-PKI mode as well [3].

ICMetrics is a unique technology for deriving private keys based on the digital fingerprint (software and hardware configurations) of the device [2]. The novelty of the proposed system is that the measured characteristics need not remain absolutely constant but can fluctuate within a (configurable) defined range, thus allowing the software to operate in several different states whilst still ensuring that any illegal clone or malware attack is detected [1]. Such a system will offer the following significant advantages: 1) Eliminate the need to store any credential related sensitive data within the device, hence addressing the major weakness which can be used to circumvent the security offered by the system. 2) In a malware attack, ICMetrics behaviour analysis helps to detect tampering with the constitution of a software will cease authentication process. The novelty of the proposed system is that the measured characteristics need not remain absolutely constant but can fluctuate within a (configurable) defined range, thus allowing the software to operate in several different states whilst still ensuring that any illegal clone or malware attack is detected.
ICMetrics is defined as a two-step process [4]:

Calibration Phase

1. For all the devices or services, measure desired feature values that characterize the device or service.
2. Generate feature distributions for each feature illustrating the frequency of each occurrence of each discrete value for each sample device. This will allow the same digital signature to be generated from the normal variations of operation of the device concerned but ensure any abnormal variation fails to generate the correct digital signature.
3. Normalize the feature distributions generating normalization maps for each feature. These essentially relate the range of measured values for a given device to a fixed range of values chosen for that particular device feature. The absolute values of features are thus discarded and abstract virtual values are chosen in their place.

978-1-7281-5546-3/19/$31.00 ©2019 IEEE

Operation Phase

1. Measure desired systems features.
2. Apply the normalization maps to generate values suitable for key generation.
3. Apply the key generation algorithm to combine the normalised feature values into a single key.

The focus of this paper is to evaluate the feasibility of generating encryption key based on hardware features derived from the properties and behaviour of general-purpose computing devices [7]. In order to achieve this, we first investigate appropriate method of extracting hardware features and explore potential features that are suitable for key generation. Then, we evaluated a new multidimensional encryption key generation algorithm.

The rest of the paper is organised as follows. First, we describe criteria of ICMetrics features then we have used to extract feature values. Then, we explain the analysis of the data we have performed and provide interpretation of the results. Finally, we summarise the paper with some suggestions for future work.

II. Criteria

The properties of ICMetrics features have been explored and the following properties are desirable to identify the device uniquely:

1) The first is the data should correlate to each other because correlated features improve the robustness of the system and raise the feasibility of raw feature data.

2) The second desirable property of a feature is a low intra-sample variation. The more a feature value can vary, the harder the value is to map and the less stable the value is when contributing to key generation.

3) The final aspect of a feature to consider is inter-sample variation. This determines a feature's entropy and the larger the inter-sample variation, the larger the entropy. In other words, the derived key should have a property with low intra-sample variance (i.e. the values produced for the same device) but high intersample variance (i.e. the values produced for the different devices) with an ideal case being no inter-sample overlap of potential features.

These criteria for features are needed holistically in the multidimensional space including correlations. Next, the combination of the data needs to present a certain amount of discrimination in a multi-dimensional space, which means it should as less overlap as possible in the multi-dimensional space. Finally, we evaluated normalization and quantization of the feature values. Overall, this paper outlines the method of analysis and mathematical implementation in multidimensional space. In our future work will focus on developing a new binary key mapping algorithm to map a measured data from multi-dimensional space to a key vector and implement Shamir's Secret Sharing to increase the entropy of the system [10].

III. Feature Extraction

We evaluated some of the potential hardware performance features read by the MacBook Air, and identified the useful ones. In order to collect data each device runs an algorithm to find features that can provide an adequate dynamic range, obfuscation and variance. By default, features collected by devices are grouped into 3 main categories. These are CPU-related values like the performance of floating-point arithmetic, memory-related features like time taken to read memory, & hard disk-related features like the CPU usage when writing to disk. Also, we analyse the correlation between features and used as new features. In this paper, we investigated hardware features as a potential ICMetrics features. Each feature was collected 1000 times since it is sufficient to determine the probability distributions. Also, please note that this is the calibration phase and not that we need to capture 1000 samples to rebuild the key which would make the system infeasible.

IV. Mapping Methodology

This section introduces the algorithm for generating an encryption key which has the following four example features related to hard disk like the CPU usage when writing to disk. To generate an encryption key, it is necessary to develop suitable methods for combining selected features to produce unique basis number - an initial binary number unique to the devices from which actual encryption keys may be derived [6][7]. This basis number can then be used to generate encryption keys, for device authentication.

In order to increase this entropy, feature values from multiple features should be combined in order to produce a long basis number [9]. Feature values can be generated from both static and dynamic features (where the value may legitimately vary for a given device) but the process of doing so varies for each type. Since static features do not change with time, the measured value of the static feature can be used directly, since it is likely to remain the same each time it is sampled. This approach will not work for dynamic features, however, since it is likely that each time the feature is sampled, the feature will hold a different value. Instead, it is necessary to take many measurements of the feature, quantize the measured values into discrete values, and generate a frequency distribution for that feature.

A. Feature Combination

Once feature values have been generated for all device features, it is necessary to combine them in such a way that they produce a suitably long basis number, with sufficient entropy to be used for key generation, and that is stable enough that it can be reliably reproduced. In other words, let us assume we take four feature set each feature set has four features each. We get a stable basis number from each of the four feature sets. Then we simply concatenate all the basis numbers to get a final basis number. Our approach to produce the stable basis number suppose we have four features (F1, F2, F3, F4) and each feature has n samples. We then add four sets of samples and get F5. We then take 20 random samples and calculate the average, take log value of the average and this log value is the base number. We repeat this process with another set of 20 random samples and derive the basis number. After repeating this process several hundred times, we then figure out whether all the basis numbers are same or not – if they are same, it is a stable basis number. Our results for these example features produced stable basis number for key generation.

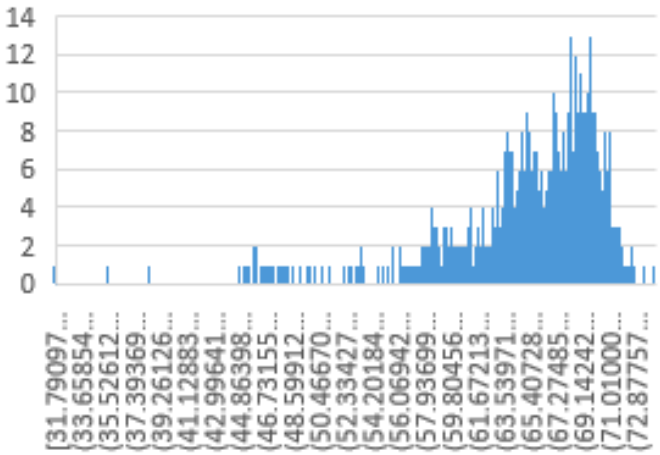
B. Feature Quantization and Normalization

978-1-7281-5546-3/19/$31.00 ©2019 IEEE

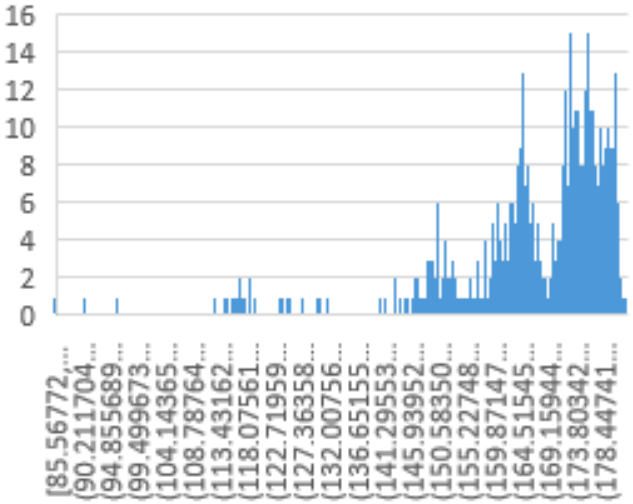
This system works in two phase process, first analyzing feature values for devices to produce a normalization map for the feature and subsequently employing the normalization maps to produce a code for identifying devices in multidimensional space. The basic concept of the normalization map is to map a measured series of feature data into a multidimensional space. In our previous work [11], normalization maps are linear based, mapping each individual feature to a vector and concatenating them together. The traditional strategy for generating an encryption key from a given feature distribution may involve quantizing the distribution into fixed subsets with each value within a given subset mapping to a single value. The goal of quantization is to normalize feature data, so the best quantization interval should exhibit the biggest inter sample variance between devices.

C. *Multimodal distributions*

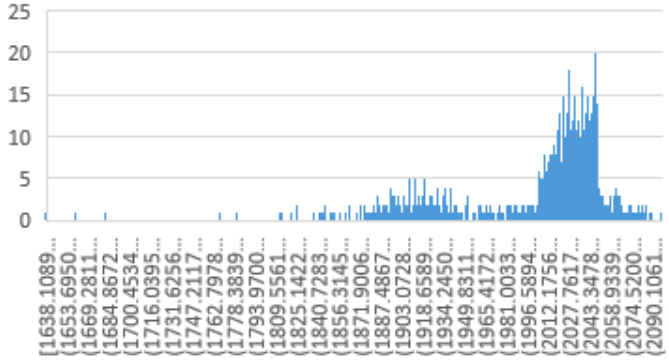
After quantization and normalization, the next step is to establish the form of the probability distribution, for example Gaussian, bimodal or multimodal in nature. It is possible that a set of data from a particular feature is mostly multimodal in nature, making it difficult to generate a basis number. Feature values that are multi-modal in distribution require careful consideration with regards to generating a stable key [9]. Before employing any mapping algorithm, bit manipulation on the feature involves a number of binary operations on the feature bytes. One solution to this problem is to divide the distribution into a series of Gaussian distributions, where each mode on the original distribution becomes the mode of its own Gaussian distribution. A simple approach to this problem is to apply a peak trough detection algorithm to the distribution, where the troughs split the multimodal distribution into separate Gaussian distributions with the peaks forming the modes [16]. Fig.1 shows the distributions of the four example features related to hard disk like the CPU usage when writing to disk for device 1 F1 & F2 shows multimodal distribution, and F3 &F4 shows Gaussian distribution and Table I shows the modes after applying peak-trough algorithm.



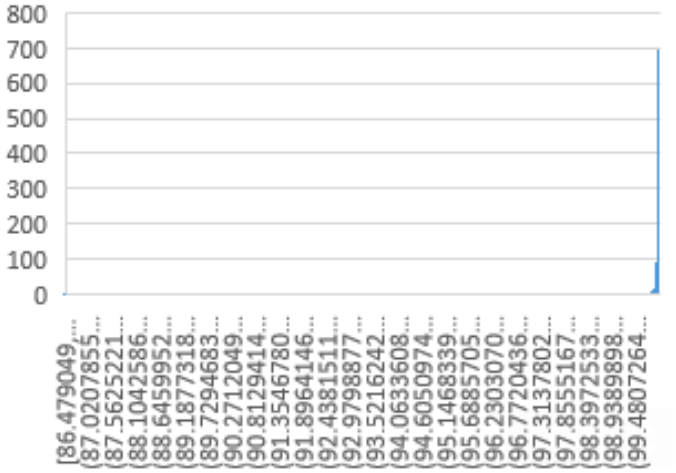
(a) Shows Gaussian Distribution



(b) Shows Multimodal Distribution



(c) Shows Bimodal Distribution



(d) Shows Gaussian Distribution

Fig. 1. Distributions of the four different features. The vertical axis is the frequency. The horizontal axis is the feature values

978-1-7281-5546-3/19/$31.00 ©2019 IEEE

TABLE I. MODES OF THE FOUR DIFFERENT FEATURES

Features Sets	*Device1 (Modes*	*Devices2(Modes)*	*Devices 3 (Modes)*
F1	5704.0623539999997	17093.88013000 0001	13458.91942
F2	0.25318800000000002	7.835268000000 0001	0.056325
F3	0.149087	0.467077999999999	0.025094000000000002
F4	0.39458599999999999	28.44702399999999	0.15968599999999999

V. EXPERIMENT & RESULTS

The experimental environment platform for extraction of the features consists of a number of stages: (1) A Software Application (2) Logging Feature Values for additional analysis required for ICMetrics. The implementation is described in the following subsections.

A. Software Application

The experimental platform for extracting ICMetrics features is iOS running on the MacBook Air. Data is collected from the hardware features (Memory, CPU) such as sequential output (block), MFlops, random Seeks etc –these features on macbook have not been investigated previously. For this research, we have employed:

- XCODE (iOS Developer Tools) build Version 8.2.1 (8C1002) including Interface Builder and an application used to construct graphical user interface.
- Three MacBook Air.
- Data collected from the general-purpose computing devices.
- Python Code and Microsoft Excel used for data analysis.

The method for collecting the feature values must be controlled such that we can determine what causes the features to behave as they do during the analysis. The sample values output for analysis have a timestamp to determine the fluctuation in feature values. This is to allow conclusions to be drawn between the presence of background processes for a system resource and the influence they can have on the various candidate features being analyzed hence it is not only system processes that could potentially affect low-level hardware feature values. User-controlled processes could also alter the distribution of a feature. To assist with this challenge, the status of the device is monitored and recorded when samples are read for analysis. This information can be very useful when developing mapping functions for feature values, and one reason why it is a good idea to run feature collection over a long period of time. When values of interest are found to have an effect on other feature values, it presents the opportunity to use these internal relationships or feature correlations, as a first-class feature. These features generally offer more natural obfuscation and are found to be more reliable than individual features, so it is common they make strong candidate features for employing in an ICMetric system.

In this analysis we have different devices, first we calculate frequency distribution of all devices and then we apply a peak-trough detection algorithm to the distribution. Here the peak-troughs split the multimodal distribution into separate Gaussian distributions with the peaks forming the modes and then we use multivariate normal probability density function to calculate the probability of the sample associated with that mode [12]. In our experiment, the features from all devices have Gaussian and multimodal distribution. For calculating the probability, we take the samples from each server, calculate the mean and covariance of the modes within the distribution of the current devices. For example, if the device has bimodal distribution then we have two modes and each mode have its own mean and covariance. For this, first we determine in which mode the current sample falls into and then we calculate the probability of the sample and repeat the same process for other modes. We then take same sample from another device and see if that sample from other device lies in which mode of first device and then we calculate the probability of the sample. If the probability from second device is low as compare to first device, that means first device is correctly identified based on probability and we repeat the same process for' n' devices. After that we look for the boundaries in the 4d space to locate which area in space belongs to which device and then we write a mapping function to identify the devices area correctly in space. We observed that the training data and testing data differentiated between devices are quite promising. In some cases, we got 97% and above correct results.

B. Correlation of Features

Correlated features are more desirable than singular features because the correlated features are likely to be more stable than the singular features as they represent a relationship rather than a specific range, such that there is less intra-sample variance thus increasing reproducibility of the generated key. In other words, in a given device, a non-correlated feature could have any range of values but the relationship between two tends to be more stable as indicate by the correlation. Another significant aspect of correlated features is their ability to help distinguish devices. Singular features have a higher change of having an overlap when the possible range for the feature is analyzed across multiple devices. Singular features are features that are measured directly from the device rather than being derived. Correlated features add an extra step when trying to recreate the values, as the correlated values must be generated and cannot be read directly from a device. Importantly, each correlated feature can itself be used as a feature, which has the benefit of increasing the entropy of the key generated by the ICMetrics algorithm [13]. For instance, Table II shows the correlation of the same features combinations from different devices. The correlation of F1-F2 from device1 is 0.964728227 and the correlation of device2 is 0.738532807. This shows a great difference between Device1 and Device2. Although the coefficient of Devuce3 is 0.982909775, which it shows a small difference compared to Device1, but it still distinguishable. For F2-F4, Device1 and Device2 show similarity. Device3 shows enormous disparity between Device1 and Device2. In this case, Device3 is distinguishable, but Device1 and Device2 are quite close. In this situation, we can still distinguish them according to the Pearson correlation distribution.

978-1-7281-5546-3/19/$31.00 ©2019 IEEE

TABLE II. CORRELATION OF FEATURES

Correlation of Features	*Device1*	*Device2*	*Device3*
F1F2	0.964728227	0.738532807	0.982909775
F1F3	0.155117596	0.351856621	0.886997405
F1F4	0.283791595	0.34646151	0.961830229
F2F3	0.224913722	0.343722645	0.872258654
F2F4	0.350919526	0.342947973	0.959656282
F3F4	0.767960793	0.886801301	0.94689632

Finally, the features of a device can be logically categorized into specific sets, which we call mini-ICMetrics. Each set contains features, which share similar traits or are affected by the same modifications of a device. The creation of the mini-ICMetrics feature sets allows for fault tolerance system to be implemented into the ICMetrics key generation process by employing them as points on a polynomial combined via Shamir Secret Sharing Algorithm [10]. This fault tolerance is achieved by using Shamir's Secret Sharing cryptographic algorithm to allow a fixed minimum number of shares mini-ICMetrics to be required to produce the same key. In this case, a mini-ICMetrics is created for each category used in the key generation process. The number of correct categories required to reconstruct the key can be defined in the algorithm, thus allowing that the robustness and entropy of the key and can be adjusted as required for the specific the circumstances. The incorporation of this enhancement introduces robustness with the key generation process by allowing a pre-defined number of feature sets to generate the correct component ICMetric value, defining the number of sets required as the error tolerance value. Thus, if the tolerance number of sets is not reached, the system fails, not meeting the secret sharing reconstruction threshold and a different key will be produced and measures to protect the data would be taken. Conversely, if the number of categories that was correct was greater than or equal to the tolerance, the system would produce the correct ICMetric key and a practical system could be implemented to adapt to the changes in the failed categories so that they could become part of the accepted range for the features in that category, where that is desirable in order to deal with acceptable changes in the system.

VI. Conclusion

This paper has explored the hardware data as an ICMetrics feature and investigated how the number of samples of the feature values being employed affects the ICMetrics system's performance. We observed that data differentiated between devices quite nicely, however the total entropy using these features was not as strong as current encryption keys so to increase strength of encryption key, we could use additional hardware features.

Overall, this paper outlines the method of analysis and mathematical implementation in multidimensional space. In our future work will focus on developing a new binary key mapping algorithm to map a measured data from multi-dimensional space to a key vector and implement Shamir's Secret Sharing to increase the entropy of the system.

References

[1] 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.

[2] E. Papoutsis, G. Howells, a. Hopkins, and K. McDonald-Maier, "Key Generation for Secure Inter-Satellite Communication," Second NASA/ESA Conf. Adapt. Hardw. Syst. (AHS 2007), pp. 671–681, Aug. 2007.

[3] B. Ye, G. Howells, and M. Haciosman, "Investigation of Properties of ICmetrics in Cloud," in Emerging Security Technologies (EST), 2013 Fourth International Conference on, 2013, pp. 107–108.

[4] 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.

[5] A.Hopkins, K. Mcdonald-Maier, and G. Howells, "Device to generate a machine specific identification key." Google Patents, 2013.

[6] R. Tahir, H. Hu, D. Gu, K. McDonald-Maier, and G. Howells, "A scheme for the generation of strong cryptographic key pairs based on ICMetrics," in Internet Technology And Secured Transactions, 2012 International Conference For, 2012, pp. 168–174.

[7] Y. Kovalchuk, H. Hu, D. Gu, K. McDonald-Maier, D. Newman, S. Kelly, and G. Howells, "Investigation of Properties of ICmetrics Features," in Emerging Security Technologies (EST), 2012 Third International Conference on, 2012, pp. 115–120.

[8] Y. Kovalchuk, K. McDonald-Maier, and G. Howells, "Overview of ICmetrics Technology-Security Infrastructure for Autonomous and Intelligent Healthcare System.," Int. J. U-& E-Service, Sci. Technol., vol. 4, no. 3, 2011.

[9] G. Howells, E. Papoutsis, A. Hopkins, and K. McDonald-Maier,"Normalizing Discrete Circuit Features with Statistically Independent values for incorporation with in a highly Secure Encryption System," in Adaptive Hardware and Systems, 2007. AHS 2007. Second NASA/ESA Conference on, 2007, pp. 97–102.

[10] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612-613, 1979.

[11] E. Papoutsis, G. Howells, A. Hopkins and K. McDonald-Maier, "Integrating Feature Values for Key Generation in an ICmetric System," *NASA/ESA Conference on Adaptive Hardware and Systems*, pp. 82-88, 2009.

[12] Y. Kovalchuk, W.G.J. Howells, H. Hu, D. Gu, K.D. McDonald-Maier, "ICmetrics for Low Resource Embedded Systems", Proceedings of the third International Conference on Emerging Security Technologies, 2012.

[13] K. Appiah, X. Zhai, S. Ehsan, W. M. Cheung, H. Hu, D. Gu, K. McDonald-Maier, and G. Howells, "Program Counter as an Integrated Circuit Metrics for Secured Program Identification," in *Emerging Security Technologies (EST), 2013 Fourth International Conference on*, 2013, pp. 98–101.

[14] W. G. J. Howells, E. Papoutsis, and K.D.McDonald-Maier, Novel Techniques for Ensuring Secure Communication for Distributed Low Power Devices , in IEEE, NASA/ESA Conference on Adaptive hardware and Systems, AHS 2006. 2006:Instanbul, Turkey. p. 343-350.

[15] XiaojunZhai ,Kofi Appiah, Shoaib Ehsan ,Wah M Cheung,Gareth Howells,Huosheng Hu ,Dongbing Gu, Klaus McDonald-Maier "Detecting Compromised Programs for Embedded System Applications" in International Conference on Architecture of Computing Systems ARCS 2014 pp 221-232

[16] https://securityintelligence.com/news/global-cost-of-cybercrime-exceeded-600-billion-in-2017-report-estimates/

978-1-7281-5546-3/19/$31.00 ©2019 IEEE