Exploring Novel Device Authentication Techniques for General Computing Devices

Supriya Yadav

A Thesis submitted to the University of Kent
for the Degree of Doctor of Philosophy
in Electronic Engineering

January 2023

Acknowledgements

I am sincerely grateful to the people who supported me all through my PhD.
My supervisor, Prof. Gareth Howells, for his advice, guidance, and consistent support throughout the research. He has been an invaluable source of constant encouragement. I would also like to thank Dr. Konstantinos Sirlantzis, my second supervisor, for his help and guidance throughout my research.

I would also like to thank Chris Miles for the extensive discussions and invaluable advice he provided on many occasions and for his continuous encouragement. A special thanks to my husband, Govind Yadav, who saw potential in me and offered advice related to PKI concepts. Without this support, I would have never endeavoured to pursue this journey.

I would like to thank my brother, Pushpander Singh, for his support. Finishing this PhD would not have been possible without his help. I am eternally grateful to my father, mother, and sister for their prayers, support, and motivation throughout my research.

Finally, I must mention my children, Riddhi and Anantshri, whose smiles have made me happy even when I was feeling down.

Abstract

Secure device authentication is one of the top challenges worldwide from a security and privacy point of view. For the provisioning of security services, cryptographic methods have traditionally relied on keys stored in the devices. These keys are vulnerable to attack since they are seldom protected.

This thesis investigates the feasibility to enhance device security. The recommended framework makes use of novel Integrated Circuit Metrics (ICMetrics) technology, which leverages measurable features and properties of a device. Low level device features are used to build an identity for the device through the use of the ICMetrics. This technology specialises in deriving strong device identity to prevent all forms of skimming and malware attacks.

Firstly, the research contribution is to examine the suitability of employing various low level behavioural characteristics or features derived from wearable and general computing devices. The novelty offered by this research enables the utilization of dynamic features instead of solely relying on static features. Additionally, the feature characteristics need not remain absolutely constant but are free to vary within deduced parameters, thus allowing the software to operate in several states and on a variety of platforms. To increase the complexity of the generated ICMetrics, the extracted feature values are subjected to statistical and mathematical analysis. Another fundamental problem solved by ICMetrics is the generation of stable and unique digital identities from features that are unstable. Potential features that might be used for device identification were the initial point of focus, which was followed by a study of the feature extraction strategy and multimodal properties. The modular dataset made it easier to assess how reliable the device identification was. The security system is analysed and tested during this phase in order to measure its efficacy. In other words, it is tested using a dataset that was captured directly from the computing devices. The accuracy rate and confusion matrix, are calculated in this phase. The investigation showed that the suggested model outperformed all other model for identifying devices. The accuracy results obtained for the second and third feature sets of the proposed model are 91.5%, 92%, and 80.3% respectively.

The thesis also investigates the effectiveness of employing measured hardware features mapped into the frequency domain for device identification. Discrete Wavelet Transform (DWT) coefficients are used as differentiating features in the approach. In this thesis, the proposed model of multivariate Gaussian distribution is used to describe the analysis process and its mathematical application.

Hardware characteristics were investigated. Wavelet-based features were leveraged. The analysis and comparison of classifiers revealed that they behave differently on the same dataset. Overall, wavelet features outperform raw features, and Sym2 and DB2 are the two wavelets that perform the best.

Finally, because the sample data was stored on the device, an efficient technique for data security had to be implemented. A decision was taken to employ the homomorphic encryption (HE) algorithm. The method fulfils the requirements for data protection.

Publication List

Journal:

1. Yadav, S., Khanna, P.R. and Howells, G. ‘Device Authentication Using Wavelet Based Features. International Journal for Information Security Research (IJISR), 2022. DOI:10.20533/ijisr.2042.4639.2022.0120.

Conference:

1. Yadav, S., Khanna, P.R. and Howells, G. ‘Robust Device Authentication Using Non-Standard Classification. International Conference for Internet Technology and Secured Transactions’ (ICITST), London, UK, 7-9 December 2021.
2. Yadav, S., Khanna, P.R. and Howells, G. ‘Device Identification Using Discrete Wavelet Transform’. IEEE International Conference on Engineering and Emerging Technologies (ICCET), Istanbul, Turkey, 27-28 October 2021.
3. Yadav, S. and Howells, G. (2019). ‘Secure Device Identification Using Multidimensional Mapping’. IEEE Eighth International Conference on Emerging Security Technologies (EST), Colchester, UK,22-24 July 2019.
4. Baba, S.D., Yadav, S. and Howells, G. (2019). ‘SortAlgo-Metrics: Identification of Cloud-Based Server Via a Simple Algorithmic Analysis’. IEEE Eighth International Conference on Emerging Security Technologies (EST), Colchester, UK,22-24 July 2019.
5. Yadav, S. and Howells, G. (2017). ‘Analysis of ICMetrics features/technology for wearable devices IoT sensors’. IEEE Seventh International Conference on Emerging Security Technologies (EST), Canterbury, UK, 06-08 September 2017

Table of Contents

List of Acronyms

AMD	Advanced Micro Devices
ARM	Advanced RISC Machine
CPU	Central Processing Unit
ECU	Engine Control Unit
FIPS	Federal Information Processing Standards
HSM	Hardware Security Modules
IAM	Identity Access management
ICMetrics	Integrated Circuit Metrics
MIM	Machine Identity Management
MIP	Machine Identity Protection
MITM	Man in The Middle
PKI	Public Key Infrastructure
SSH	Secure Shell Protocol
TPM	Trusted Platform Module
AES	Advanced encryption standard
BWAA	Basic Web Application Attacks
DES	Data encryption standard
IMEI	International Mobile Equipment Identity
KMIP	Key Management Interoperability Protocol
KMS	Key Management Systems
MIME	Multipurpose Internet Mail Extensions
MPC	Multi Part Computation
PUF	Physical Unclonable Functions
RSA	Rivest-Shamir-Adleman
SSL	Secure sockets layer
TCU	Telematics control unit
TPM	Trusted Platform Module
MFLOPS	Mega Floating-Point Operations per Second
HDD	Hard Disk Drive
MVGD	Multivariate Gaussian Distribution
SVM	Support Vector Machines

GMM	Gaussian Mixture Models
LDA	Linear Discriminant Analysis
LR	Logistic Regression
QDA	Quadratic Discriminant Analysis
WPC	Wavelet packet coefficients
BFV	Brakerski/Fan-Vercauteren
CKKS	Cheon, Kim, Kim and Song

List of Tables

List of Figures

Chapter 1

Introduction

Our identity is everything: who we are, what we have access to, etc. Every day, we rely on our identities. Today, with ever increasing cyber fraud, securing online identities has become a major focus area for organisations across the globe. Humans are just a speck in the identity network ecosystem. The problem was amplified when machine identities outnumbered humans. Just like humans, machines also need identities to authenticate and securely communicate with each other and with humans. The IAM tools are better equipped to handle human traffic; however, we need similar tools, or rather more sophisticated ones, to handle machine identities. So, what is a machine ID? Simply put, they're your cryptographic assets, like X.509 certificates, SSH keys and certificates, symmetric keys, code signing certificates, and other secrets and non-human identities.

With constantly rising cyber scams, securing online identities has become a main focus for organisations throughout the world and almost 14 billion dollars were spent on identity access management (IAM) in 2021 [1]. IAM relies on tools like multi-factor authentication, single sign-on, and privileged access management.

Identity fraud can wreak havoc on societies and economies, and this crime is often committed to facilitate other crimes such as credit card fraud or money laundering, mail fraud, bank fraud, wire fraud, etc. These frauds affect not only individual citizens and the nation's economy, but they are a national security threat as well. There are different types of Machine Identity Management (MIM) solutions in the market, for example, hardware and software-based techniques; they have limitations and vulnerabilities, which are detailed in Chapter 2 (Section 2.9). However, the need of the hour is robust Machine Identity Protection (MIP) [2].

Hardware-based solutions are relatively expensive. In a post-pandemic workforce, remote connections have greatly increased, and this has created additional security concerns for CISOs that cannot be met with ever tightening budgets. Large organizations have to choose between key protection and productivity. Unfortunately, today it is all too likely that organisations are relying on the underlying native security offered by a device's operating system, the device hardware itself, and the microprocessor providers.

The primary challenge with these MIP techniques is that they can be compromised by advanced attacks like MITM, malware, etc. If deployed correctly; PKI is, by far, the most robust technology [3]. In the world of PKI, a credential is a combination of a digital certificate and a private key. Conventional key protection techniques have fundamental flaws. For example, software based key protection approaches like .pfx or. p12 files or browser-based certificate stores have issues like being prone to brute-force attacks, can be skimmed, and software containers cannot be strongly bound to a machine. The traditional hardware backed mechanisms to protect the private key is through Hardware Security Modules (HSM), smart cards, TPM chips, and key management solutions. HSM security is, of course, very high (often at the FIPS level); however, applications connect to the HSM via username and password and/or client certificates stored in .pfx or. p12 files [4]. Often, these can be hacked with simple social engineering tricks. Further, these are dump authenticators since they will sign any hash value so long as the container password is correct. Hardware containers like cryptographic smart cards, TPMs, which have limitations, hardware, and logistic cost, requires driver installation. To protect against fraudulent data manipulation, the protection of encryption keys should be guaranteed.

A majority of main stream crypto key providers have been successfully attacked multiple times, such as the Pegasus attack against WhatsApp encryption keys (iOS and Android) [5], Jeff Bezos iPhone hack, Meltdown and Spectre (Intel, ARM, AMD, Linux/Windows, and so on) [6], and, more recently, SGAxe and Crosstalk targeting Intel H/W [7].

To avoid total dependence on native security features, one can adopt an effective 'layered' security approach. ICMetrics offers all the capabilities and benefits to provide a strong deterrent against the above-mentioned threats and attacks.

Integrated Circuit Metrics (ICMetrics) is a novel machine identity protection technology for deriving unique private keys based on the operating characteristics of digital systems (a combination of software and hardware configurations) using properties or features derived from their own construction and behaviour capable of assuring both their authenticity and freedom from malware [8][9] [10].

A significant amount of such fraud can be tackled effectively if there is a robust way to link users' physical identities to their online identities and the credentials are strongly bound to their devices.

ICMetrics can play a crucial role here. ICMetrics is a software client that reads various dynamic and static (hardware and software) feature values of a device and generates a unique identifier for the device. This unique identifier is used to generate key pair, of which the private key is not stored permanently on the device or in the database. Every time a cryptographic operation is required, the ICMetrics client reads these feature values and reconstructs the private key. If the ICMetrics client is skimmed, then on a rogue device, the feature values will differ from what the ICMetrics client expects, which will result in a failed cryptographic operation. This technique eliminates offline brute force attack [8][9] [10].

The novelty of the proposed system is that the measured characteristics do not need to remain constant but are free to vary, thus allowing the software to operate in several states and on a variety of platforms while still ensuring that any skimming attacks or malware are detected via unacceptable changes in the operating parameters. This technique thus represents a promising new approach for generating unique keys for systems enabling the protection of cryptographic keys. This technology can identify pattern features with highly non-standard distributions derived from machine level behavioural characteristics, which ensures that a private key is not easily compromised [8][9] [10].

Such a system will offer the following unique selling proposition:

1) The elimination of the need to store any type of sensitive data (or template) for validating the service, thereby directly addressing the major flaw that the data is accessed and used to breach the system's security.

2) Prevent malware from taking over devices and tampering with software configuration. Malware infection will cause its ICMetrics behaviour to change, which will prevent its further utilisation.

A system for implementing device identification is proposed in this thesis. ICMetrics technology is investigated in the proposed system to identify devices. The ICMetrics technique generates a unique identification for a device using internal properties of the device. It can be used to offer authentication and attack detection services. ICMetrics is generated automatically as needed and is then discarded, with no user input [8][9] [10].

1.1 Research Motivation

An internet-connected device is always at risk when modern social engineering techniques, naive users, and other factors are combined. Personal and confidential information can be compromised with just one successful attempt. Therefore, it is strongly advised that confidential data be encrypted in order to reduce the fraudulent use of the compromised data and protect its validity and integrity.

Data encryption, which transforms plain-text data into seemingly random data, is a crucial instrument for data protection. An encryption key is used to encrypt plain text, and depending on whether symmetric or asymmetric encryption is being used, the same or a different key is used to decrypt the data, respectively. Hence, securing these keys is of prime importance. When the security of the encryption keys cannot be completely ensured, the integrity of encryption is called into question.

Apart from encryption, cryptographic keys can be leveraged for authentication and digital signatures as well. These serve the purposes of identification and non-repudiation. The cryptographic keys are often secured by passwords, and since they tend to be brief and contain alphanumeric characters, they are weak in nature. Therefore, key creation and seamless usage has been suggested in recent years without the user being aware of the encryption key or passphrase.

There are currently more online devices than users [11]. This indicates that many of us have multiple internet-connected devices. It is impossible to overstate the significance of security as more commonplace items like watches and televisions gain internet connectivity. Computational devices are still insecure despite ongoing development in the realm of cryptography. Every year, there are several reports of computer security incidents where systems are attacked, resulting in monetary loss, data theft, and even a threat to life[12]. The availability of cheap computing power and improved communication have made enemies more powerful than before. The goal of cyber-attack may be to jeopardise security or simply to cause a minor annoyance. Traditional cryptography systems rely on the user having access to these keys, which breaks down if the secrets are not kept private. Furthermore, keys cannot offer non-repudiation because they might be misplaced, lost, or stolen.

An enemy may view device communication as an alluring setting full of equipment and linkages. Therefore, attackers will try to acquire access by taking advantage of security or system design weaknesses. Security procedures and techniques that call for cryptographic keys presumptively keep the key secret. Traditional non-hardware-based cryptosystems have a flaw because there are numerous ways for an opponent to obtain the keys. So, any security-based system's Achilles heel is cryptographic key theft. It is imminently necessary to adopt a new strategy for the supply of security in light of the possibility that an attack on cryptographic keys could result in failed security.

An alternative method for protecting a system is to enhance the device identity. ICMetrics offers a unique way to measure characteristics of a target device by leveraging features and their combination. In this research, a security system based on ICMetrics technology is suggested. This system collects properties and features from a devices' behaviour and characteristics in order to uniquely identify and secure a system based on that identification. A device can create an identity using the ICMetrics technology in order to provide authentication and a number of additional security functions [9] [10].

Additionally, some security measures rely on the keys that are kept in storage to enable safe information. Such methods have weaknesses because they leave the security of any data protected by the keys exposed in an instance that they are compromised. Consequently, it is crucial to use security techniques to defend these systems against intruders. By utilising a device property to generate an ICMetrics number that will be used for the identification, ICMetrics technology has been developed as a means of preventing key theft [9] [10].

1.2 Scope and Research Objective

This thesis offers a novel method to address the device security issues raised above and develops a methodology to showcase how device characteristics can be extracted and captured to produce a unique digital identity. Utilizing these unique characteristics in various models, helps to determine the identity of a device.

The aim is to reproduce unique identifiers for a particular device to identify it in a large device base. Firstly, the features that can be used for device identification are looked into. The technology is then used to standardise the characteristics. Multimodal features are examined and converted to a format that traditional statistical models can easily understand. One example is the division and reordering of multi-model features into a normal form. The effectiveness and viability of the features that have been gathered are examined next. For the purpose of identifying the configurations that result in the best performance, all features are mapped into a multi-dimensional space.

The unique generation of a digital identity by harvesting device features overcomes major issues regarding secure data transfer, such as spoofing, cloning, and MITM attacks. As the identity of the device becomes more difficult to mimic, it reduces the significant threat posed to communication systems. The technique offers mathematical analysis and device characteristic modelling, which allows for the actual feature values to change. As a result, a fundamental concept for generating identity keys under variable conditions is implemented. The cryptographic keys generated using the ICMetrics technology can be leveraged for authentication, confidentiality, and integrity [8][9] [10]. The framework that is being presented examines the ICMetrics technology in two ways: first, as a foundation for creating cryptographic keys for device identification, and second, as a means of preventing key theft.

The objectives are:

1. Developing a security system for device identification based on ICMetrics technology, which depends on features that distinguish each device and generates a unique number called an ICMetrics number that is used for device identification.
2. Robust authentication of devices is based on the internal behavioural characteristics of the associated hardware and software. The technology can prevent MITM, brute force, malware takeover of device etc.
3. Demonstrating how device features can be leveraged to increase security and implement device identification. It is possible to identify a device using data collected by internal behavioural characteristics of the related hardware and software.
4. Examine features in the frequency domain for identifying devices and determining a unique ICMetrics.
5. Device identification using homomorphic encryption.
6. System performance parameters, such as the confused matrix, accuracy rate are tested and evaluated.

1.3 Research Challenges

Based on the description above, the main challenges of this research are listed below

1. It is crucial to minimize intra-sample variability at all stages of the procedure. This is required to achieve the best results for all pattern recognition tasks. If the intra-sample variation is not reduced at every level, it will progressively increase.

2. The variation of the extracted feature vectors should also be decreased by lowering the variance at the beginning of the process. This should make it possible to develop matching and classification algorithms that require less variance modelling during matching.
3. Given the difficulty in generating a stable unique identifier for individual devices, the challenge was how to deploy multimodal features in multi-dimensional space.
4. The standard pattern recognition system may struggle to compute features with non-standard distributions. The research was focused on challenge to normalise those features and apply them to the ICMetrics system. As is well known, an unstable set of features will result in an unstable pattern recognition system. Stability is far more crucial for a security system than other conventional classification schemes. It will be quite difficult to figure out how to apply those elements of an unstable environment to a security system.
5. Performance can also be enhanced by extracting a group of features that share specific traits. However, when implementing an automatic pattern recognition system, such feature selection is a crucial challenge that must be taken into account. A feature, or combined feature set, must be highly discriminative (i.e., have low variation between signals obtained from the same origin and high variation between signals obtained from different origins), consistently reproducible, and invariant to affine transformations and scale in order to yield the best results.
6. Unfortunately, it is not easy to pinpoint the specific features or feature combinations that deliver high inter-sample and low intra-sample variance. However, because of the considerable variability of the data, it may not always be possible to consistently extract a feature that is highly discriminative. As a result, finding an appropriate representation of the feature is just as crucial as finding the feature itself.

Despite the fact that ICMetrics has been studied for a while, this thesis is innovative in that it uses, for the first time, new strategy to leverage unique hardware features and combining them to significantly enhance device identity. This research also utilizes discrete wavelet features, and homomorphic encryption to protect template data.

1.4 Research Questions

A set of research questions is developed to further investigate this area and identify the opportunities and constraints for device authentication. The goal of the thesis is to develop a framework for evaluating and quantifying performance on various platforms across various contexts, usage tasks, and scenarios. This research seeks to provide answers to the following important questions:

1) Is it possible to enhance device identification to 80% or above from the physical characteristics of general computing devices?

By analysing various operating characteristics of digital systems associated with their software, hardware configurations, and behavioural properties, for example, CPU usage performance, memory I/O, hard disk speed, usage of apps, keystrokes, tabs opened in browsers, etc., one can model pattern features, which can ensure that a unique private key is constructed serving the purpose of robust device authentication. These features are plotted on a polynomial, and if malware infects the device and the presence of additional software (malware) violates the polynomial, the authentication fails, preventing malware takeover of the device.

2) What elements of the devices have an impact on the system's overall performance?

When using the embedded sensors on the device to capture feature data, it is essential to examine the stability of these over time and in different environments. Because every user's machine usage has a different impact on the features that are extracted (as each user uses their machines differently), it is possible to draw connections between the existence of background processes for a system resource and the potential impact those activities may have on the various candidate features under consideration. Understanding whether and how these factors affect authentication performance can lead to the development of robust authentication methods and devices. Therefore, a study on the impact of these factors on device identification has been conducted and presented in this thesis.

3) How can various behavioural characteristics of a device be used to increase identification accuracy even more?

Enhancing the underlying security is essential because of the rising threat of spoofing and MITM attacks on devices. Understanding the technological viability of establishing a system, as well as its potential to increase identification accuracy in the context of challenges unique to devices, was essential for achieving this. In this work a thorough analysis and comparison of the device identification performance based upon multimodal features has been conducted, along with a variety of traditional classifiers for benchmarking.

1.5 Thesis Structure

There are a total of six chapters in this thesis. The areas in which this thesis study was conducted are listed below.

Chapter 2 starts with an overview of security requirements. Then it presents current State of the Art in Authentication techniques for example PUF, TPM, MPC. Following that, discussed the background of ICMetrics and related works. Next the focus was on extensive competitive analysis and a comparison of various different types of authentication technologies. Finally, the chapter ends with security issues in cybersecurity physical attacks on devices.

Chapter 3 describes the ICMetrics Process, the feature extraction concept, feature modelling, introduce feature analysis, and multimodal distribution. The second half of the chapter explains the operation phase of the system and the different methodologies and algorithms of the system. Lastly, it describes the experiments, results, and conclusion.

Chapter 4 starts with the introduction of wavelets, types of wavelets used for analysis, and related work. Then it describes a system overview, experimental methodology, and classifiers, and ends the chapter with experimental results and a conclusion.

Chapter 5 commences with a brief focus on related works, then goes on to describe homomorphic encryption and explain the proposed system. Lastly, it focuses on experiments conducted and concludes the chapter.

Chapter 6 is the last chapter, which summarises the findings of the research. Additionally, future work is discussed.

Chapter 2

Literature Review & Related Works

2.1 Introduction

Today's cutting-edge computer environment views security and data privacy as one of the most important factors impeding the widespread adoption of large-scale general computing devices. Determining the security level of a particular device and being able to comprehend device specific vulnerabilities are therefore essential. Protecting the hardware and software from attacks is currently an issue. Security refers to defending computer systems from harm and unauthorised entry by malicious actors and software, such as malware. One of the most frequent forms of attacks on information systems, for instance, is the computer virus. Security tools are required to find and eradicate the malware once it has taken up residence in a computer system.

Attacks on a device are often conducted with knowledge of the device's limitations. Cybersecurity experts understand that one of their responsibilities is to reduce the impact of security threats. When sharing data and resources, it's crucial that both communicating devices are protected against threats. As per Verizon’s Data Breach Investigations Report 2022, a vast majority of cyber-attacks involving Web applications use stolen credentials [13]. This highlights significance of robust authentication techniques.

This chapter highlights existing device authentication techniques i.e., Physically unclonable functions, Trusted Platform Module (TPM), Hardware Security Module (HSM), Multi Party Computation (MPC). The deficiencies in the current state of art and competitive analysis are examined in detail. The chapter then describes the ICMetrics technology's design ideas, novelty, earlier work related to ICMetrics, and the chapter also includes a comprehensive analysis on device attacks.

The purpose of this research study is to examine how the novel technique, ICMetrics can be used to address credential related security problems. It is essential to have a fundamental understanding of the terminology and evaluation standards currently used in these domains. Once the user authenticates, there are crucial cryptographic operations required to ensure the confidentiality of data. All such operations were discussed and supported by ICMetrics.

The rest of the chapter is organised as follows:
Section 2.2 discusses security requirements. Section 2.3 will explore the current start of the art in authentication techniques. Section 2.4 discusses the background of ICMetrics, section 2.5 on ICMetrics related works, section 2.6 focuses on competitive analysis, section 2.7 will discuss a comparison of various other types of authentication technologies. Finally, Section 2.8 discusses cybersecurity physical attacks on devices and Section 2.9 summarize the chapter.

2.2 Security Requirements

The goal of security is to safeguard system data in order to preserve it against modification and theft and to ensure its availability, confidentiality, and integrity. It covers securing computer hardware, software, data, and connections. [14]

The word ‘security’ is used to refer to a variety of situations, including those in which there are no risks or threats, hazards are prevented, or confidence has been attained. At the level of both individuals and organisations, attaining security is necessary in various contexts. The level of security varies from one organisation to other since ensuring security requires people with expertise and experience. It is crucial for users to take a systematic approach, which entails analysing, planning, implementing, and maintaining a necessary security system, in order to improve security for companies and individuals [15].

Security has a variety of objectives, as shown below [14]:

Authentication: The process of confirming a person's or a device's identification.

Confidentiality: In order to avoid unauthorised access to the information, confidentiality refers to keeping the information private.

Availability: This refers to the data's readiness for use. A system malfunction results from delayed access to information.

Integrity: Integrity is the quality of preventing information from being manipulated or destroyed. Information can be altered or sabotaged if there is a lack of integrity.

Non repudiation: This guarantees that sender cannot ever refute signing the message.

Adversaries are able to take advantage of system flaws in order to get unauthorised access. In order to ensure the security of any system, both its hardware and software must be secure.

2.3 Current State of the Art in Authentication Techniques

The emphasis in this section was to understand existing device authentication techniques to address the core security requirements such as PUF, MPC, TPM, and PKCS #12 file-based storage, as well as their vulnerabilities and limitations. The emphasis was on how ICMetrics overcomes these challenges.

2.3.1 Physical Unclonable Functions (PUF)

PUF is usually implemented by utilising a specific hardware component in a device, which serves as the basis for a digital finger print [16]. SARM-based PUF is one way of doing it. Here is how it works:

1) A software client is deployed on each device. And at the silicon level, this client reads the unique submicron physical characteristics of a chip's SRAM and, using this as a digital fingerprint, generates an asymmetric key pair. It is a scientifically proven fact that each SRAM (even of the same model and produced by the same manufacturer) is absolutely unique, resulting in a very high entropy. A digital certificate is then issued in a standard way that serves identity purposes [17].

2) Since PUF client is required to generate this private key, if skimmed, this client will be required on a rogue device as well to generate the same private key. And since the physical

characteristics of SRAM will change, the same private key will not be generated, effectively making it impossible to skim.

There are two types of randomness with PUF:

- Intrinsic Randomness: PUFs rely on randomness that is intrinsically present in the physical device introduced at the time of its manufacture. The advantage of this type of PUF is that no special fabrication techniques are required to produce them, making them easier and cheaper to integrate into an IC.

- Explicitly Introduced Randomness: PUFs that use a source of randomness that is intentionally added to the device, such as an optical system, have the advantage that they are generally able to produce basis numbers with higher entropy than intrinsic randomness PUFs and tend to be more environmentally independent since the source of randomness was chosen with these parameters in mind. However, special fabrication techniques are often required to integrate these sources of randomness into the device, increasing its manufacturing difficulty and cost.

PUFs produce their basis number by implementing the challenge response authentication method: the PUF is challenged with a stimulus applied to the device's input, which is mapped (via the randomness in the internal structure of the device) to an output called the response. While different stimuli will produce different responses, the PUF's response to a given stimulus should always be the same. Since this challenge response mapping is defined by randomness in the internal structure of the device, it can be challenging to spoof or clone a properly implemented PUF.

Although PUFs provide an adequate source of basis number generation, there are several disadvantages when compared to an ICMetrics approach. One disadvantage is that a system implementing PUFs requires specialised hardware, implemented at the design phase, making it difficult to apply retroactively if it is decided that additional security is needed. This is in contrast to an ICMetrics approach, which aims to generate a basis number directly from characteristics already present on the device, allowing it to be implemented in software even after the device has been manufactured. In addition, a PUF based system is only as secure as the PUF hardware; if a given PUF is found to be vulnerable (as many have been), then the entire security system has been broken; it is not easy to patch a hardware problem. Since an ICMetrics system is more adaptable, even if a particular implementation of the system is found to be vulnerable, it may be possible to apply a software fix to resolve the problem [18].

2.3.1.1 Vulnerabilities

1) PUF, in general, has known vulnerabilities.
2) PUF itself does not have anti malware capabilities and is vulnerable to attacks by malware running on the system's (micro) processor. It is the device owner’s or manufacturer’s responsibility to protect the boot sequence.
3) Since Intrinsic PUF technology depends exclusively on dedicated hardware, if ever a vulnerability is found with this component, as happened with the Spectre and Meltdown vulnerabilities (which affected all microprocessor manufacturers), a large number of keys deployed in the field might have to be revoked and renewed [19].

2.3.1.2 Limitations

1) A very common PUF implementation relies on SRAM. And it does not work on devices that block access to SRAM, such as Intel, Apple, etc. – that’s a significant chunk of enterprise security devices.

   Further, in the IoT space, SRAM PUF suitability is limited only to IoT devices. However, if the customer is focused on securing PKI keys across the entire IoT eco system, like servers, computers, mobile devices, and the applications that interact with the IoT device, then this technology is ruled out [20] [21].

2.3.2 Trusted Platform Module (TPM)

TPM is an in-built hardware chip (dedicated microcontroller) designed to secure cryptographic keys in such a way that the private key is not exportable. Because TPM is built into the device, third-party applications and services can use it right away.

TPM follows the RSA principles of holding a private, inaccessible key for either encryption or decryption. An attacker cannot remove a TPM chip and insert it into another computer, as it is fixed to specific hardware [22].

2.3.2.1 Vulnerabilities

1) Being hardware, if there is a vulnerability detected in library implementation, the devices need to be physically recalled - a logistical nightmare. For example, in October 2017, it was reported that a code library developed by Infineon, which had been in widespread use in its TPMs, contained a vulnerability known as ROCA, which allowed RSA private keys to be inferred from public keys. As a result, all dependent systems were vulnerable to compromise, such as identity theft or spoofing. And one country, Estonia, ended up recalling 750,000 ID cards. Software implementations are relatively easy to fix via an OTT update.
2) TPMs are prone to a cold boot attack (or, to a lesser extent, a platform reset attack). This is a type of side channel attack in which an attacker with physical access to a computer is able to retrieve encryption keys from an operating system while in operation after using a cold reboot to restart the machine. The attack relies on the data persistence property of DRAM and SRAM to retrieve memory contents that remain readable for a few minutes after shutdown [22] [23].
3) There have been several other attacks reported over the years, for example, a design flaw in the TPM 2.0 specification for the static root of trust for measurement (SRTM).

2.3.2.2 Limitations

a) Not all devices support TPMs. For example, Apple has not shipped a device with TPM since 2006.

b) TPMs are not economically viable for some IoT devices. For 100K-200K devices, it costs around USD 4-5 per TPM. This makes it difficult for manufacturers in certain segments with cut-throat competition, for example, smart thermostats.

c) Trusted Computing Group (TCG) has faced resistance to the deployment of this technology in some areas, where some authors see possible uses not specifically related to Trusted Computing that may raise privacy concerns. The concerns include the abuse of remote validation of software (where the manufacturer decides what software is allowed to run) and possible ways to follow actions taken by the user being recorded in a database that are completely undetectable to the user [23] [24].

The ICMetrics approach is not tied to a specific hardware platform; it is applicable across various hardware platforms. With ICMetrics, there is no chip for an attacker to remove, as the encryption keys are generated from the device features [25] [26].

Furthermore, in an ICMetrics approach, the feature value derived from a basis number and various combined features, which can be considered a private encryption key, is purely generated on request [27]. In contrast, in the case where a TPM is not cleared before a clean installation of a new operating system on a computer, under the principles of attestation, where a witness verifies authenticity, due to the reuse of similar TPM values by another operating system, the TPM encryption key generation is not purely unique, even though a high level of machine security is provided. The ICMetrics system is more unique by providing key encryption values on request at runtime, based on a series of attested combined feature distributions in the calibration phase.

2.3.3 Multi Party Computation (Shared Secret Approach)

MPC splits an asymmetric key into multiple shares, one of which is stored on the end entity device and the rest at the servers (on-premises) which support HSMs. Every time a cryptographic operation is required, all the shares are combined on the server and the operation is performed; all these shares are never combined on end-point devices. During the crypto operation, the end-entity share is exposed in device memory for a fraction of a second [28].

This end-entity share is protected by three asymmetric keys – the first key encrypts the end-entity share, the second signs every message send to MPC server and the third decrypts in-coming messages sent by the server.
They use the on-board security features offered by the device for securing the end-entity share and three asymmetric keys. These include the trusted execution environment for Android and the security enclave on iOS. Further, they can utilise native mobile biometric authentication schemes for end-user certificate use cases.

Every time the end-entity share is used, it is refreshed or rotated, primarily to protect against skimming attacks. They seem to support the risk engine (geo-location) for additional security.

2.3.3.1 Vulnerabilities

1. End-entity shares being exposed to memory can lead to a malware based skimming attack that can take a memory dump and then skim the sensitive data.
   [ICM advantage [25]: it effectively protects against malware attacks]

2. If end-entity share can be compromised, it is also equally likely that the three asymmetric keys can be compromised as well!
   [ICM advantages [27]: ICM has the capability to bind the private key to the device and does not need additional keys to enhance security.]

3. MPC technology itself does not offer anything at all to secure the private keys at the device level; there is total dependence on native security features, all of which have been breached, for example the Pegasus attack against WhatsApp encryption keys, Jeff Bezos iPhone hack, etc.
   [ICM advantage [27]: ICM offers very robust capability to protect and bind the private key to the device and does not depend on native security features. That said, native features can be used if the customer wishes to have additional security.]

4. If the device is compromised by malware, it can potentially also skim the new, refreshed key even before the victim can use it. There is no way for the server to know whether malware has taken over a device.
   [ICM advantage [26]: There is no need to rotate or refresh the keys or have short-lived certificates. ICMetrics itself effectively protects the keys from being skimmed.]

5. Geo-location only helps if the fraudster and victim are separated far enough to be identified as such. What if the fraudster is located in close proximity to or in the same building as the victim? Worse, it could be malware or a bot running on the victim's machine!
   [ICM advantage: ICM does not have any dependency on geo-location.]

2.3.3.2 Limitations

1) Since there is a dependency on the server to compute MPC, this technology is not suitable for SSL certificate key protection due to additional latency - every time an SSL handshake is required at the server, MPC client will need to communicate with the server, each of which can

support approx. 1000 concurrent users [29]. For higher concurrency on web servers, MPC technique will need multiple servers; hence, it will have significant infrastructure overheads! Additionally, this technology is not suitable for devices that are deployed in air-gap environments, for example, some IoT devices.

[ICM advantage: Since this technology runs entirely on the device, there is absolutely no dependency on servers, databases, etc. ICM does not store device feature lists or values locally, nor does it store the actual private key in file [26].]

2) Since IoT devices usually do not have on-board security features, this technology is not suitable for most IoT use cases.

   [ICM advantage: This technology doesn't depend on native security features, but rather provide themselves, which converts a standard device into a smart card or HSM.]

3) Since the end-point share is refreshed or rotated every time it is used, a new certificate will be required as well (a one-time certificate). Hence, they are not at all suitable for SSL certificates or for IoT devices with limited computational power and a lack of CSR generation capability.

   [ICM advantage [27]: There is no need to rotate or refresh the keys or have short-lived certificates. ICMetrics itself effectively protects the keys from being skimmed.]

4) Geo-location-based checks will further introduce latency for SSL certificates and are therefore not suitable.

   [ICM advantage: ICM does not have any dependency on geo-location.]

2.3.4 PKCS #12 file-based storage

PKCS #12 defines storing multiple cryptography objects as a single file. It is commonly used to store a private key with its X.509 certificate. These files have the. p12 or .pfx file extension. Being a software form factor, it is easy to deploy and supported by all standard applications and operating systems. This crypto container supports password-based access [30]. There are several use cases that rely on file-based storage of keys and certificates. Here are a few examples:

1. Digital signing and encryption: S/MIME, data encryption, EU eIDAS digital signature solutions, KMS, etc.

2. Client authentication (VPN, HSM, Certificate Enrolment clients, 2FA solutions, KMS, etc.).

2.3.4.1 Vulnerabilities

[1] PKCS #12 files cannot be locked or bound to a machine. To get remote access, usually there is an application username and password combined with a PKCS #12 client certificate. Assuming login credentials are compromised by standard attacks like phishing, smishing, or vishing, or by social engineering, insider attacks, etc., and malware then skims the client certificate file, the bad guys can gain access to the network or sensitive target keys. So, access to military-grade secrets is protected by flimsy authentication schemes!

[2] This key container supports password-based access; however, once skimmed, it is prone to an off-line brute force attack.

[3] Dumb authenticator: will sign any hash value so long as the .pfx/. p12 password is correct. It does not have any additional intelligence to detect threats.

2.3.4.2 Limitations

There are no limitations with PKCS #12 as such; however, certain implementations have overhead. For example, KMIP (Key Management Interoperability Protocol), etc. Here, the high-value 'target' key (for the end-user or server) is stored at a secure centralised military-grade crypto server (FIPS or Common Criteria), instead of the insecure end-entity device (a computer or server). In order to access the target key, the client authenticates to the crypto server and either (a) downloads the target key temporarily to perform the crypto operation, which is insecure, or (b) the crypto operation is performed remotely. This method, although secure, has latency, and for certain use cases it is not suitable, for example for hard disk encryption, where the data volumes are significantly higher and cannot be transferred remotely. As a result, on-device secure key storage is required [31]. The solution to all these vulnerabilities and limitations is discussed below.

2.4 Background of ICMetrics

Integrated Circuit Metrics (ICM) is a highly sophisticated device identification technology [25] [32]. It is essentially a software client that reads various dynamic and static hardware and software feature values from a device and calculates a polynomial. Each of these feature values is a point on the polynomial, which gives an ICM value, which in turn is the basis of a digital fingerprint, also known as device ID. This ICM value is used to generate the key pair, and the private key is not stored permanently on the device or in the database [25] [32]. Every time a crypto operation (for example, authentication) is required, ICMetrics reads these feature values, ensures it's the genuine device, reconstructs the digital fingerprint, and reconstructs the private key. If the ICM client is skimmed, then on a rogue device, the feature values will obviously differ from what the ICM client expects. In this situation, depending on the use case, either ICM can halt the cryptographic operation completely (for example, in the SSL use case). Or to trick the bad actor, ICM can deliberately construct an incorrect private key (client authentication use case). This forces the bad actor to use the incorrect key, which will eventually result in a failed cryptographic operation. This technique also eliminates 'offline brute force' attacks [33] [34].

By using the ICMetrics technology, there is no need to store the keys or any associated templates because the ICMetrics and keys are generated when required and discarded thereafter. Doing so discourages attackers since there is no cryptographic key present on the system. The concept of ICMetrics is analogous to human biometric technology: the identification of individuals using their varying physical and behavioural characteristics, such as fingerprints, iris patterns, voice, etc. Similarly, the ICMetrics technology proposes using device features to identify every device uniquely and this is achieved without the need for stored templates or associated data. This quality means that the ICMetrics technology can be used for preventing key theft, impersonation, and spoofing based attacks on computation systems [35]. In this technology, ICMetrics keys are generated when needed and then deleted afterward, there is no need to store the keys or other related templates when using the ICMetrics technology. Since no cryptographic key is present on the system, doing so deters attackers [36].

The formation of an ICMetrics is a complicated process, mainly because the ICMetrics is created using both explicit and implicit features. The features used for ICMetrics generation have a significant impact on the security of an ICMetrics based system. For example, since a device MAC address may be recovered with the help of a network surveillance tool like Wireshark, it is not a good candidate. Because of this, with lot of devices, a number low level features are used to generate the ICMetrics. The benefit of using low level traits is that they are difficult for an adversary to predict or duplicate [37].

The presence of malware (software) will result in an additional (invalid) polynomial point and thus fails to calculate the correct ICMetrics value, resulting in the private key not being reconstructed and thus the crypto operation failing. Though the default communication interface would be REST APIs, ICMetrics can be embedded into the device firmware, making it suitable for IoT devices.

The technique effectively allows the use of the distinct characteristics that are generated even for identical hardware and software. Some examples of features can be location information, app usage patterns, and content changes on the device, etc., together with low level operating characteristics of the system itself.

A major novelty and security enhancing feature of the ICMetrics system is that the measured characteristics need not remain absolutely constant but are free to vary within deduced parameters, thus allowing the software to operate in several states and on a variety of platforms while still ensuring that any illegal clone or malware infecting the software is detected as a result of unacceptable changes in the operating parameters. This also acts as a security enhancing feature in that an attacker cannot easily derive the information needed to reproduce the characteristics needed to break the code by observing the device, especially as characteristics whose acceptable values vary over time are typically employed [26] [27].

Two steps are required for the production of an ICMetrics system: the calibration phase and the operation phase.

Phase I: Calibration (once for each new device type)

1. For each sample device or system type, measure the desired feature values, which offer higher entropy.
2. Generate feature distributions describing the frequency of occurrence of discrete values for each sample system.
3. Normalize the feature distributions and generate normalisation maps for each feature.

Phase II: Operation (every time a device needs to authenticate)

1. Measure the desired system features.
2. Apply the normalisation maps to generate the correct values for key generation.
3. Apply the key generation algorithm and reconstruct the key.

These illustrations highlight some of the ICMetrics technology limitations:

1) The target test devices must be of same type (make and model), thus imposing limitation.
2) The stability of unique identifier is one of the limitations that depends on several factors, such as number of devices available for readings and the environment of their operation, features employed and mathematical equations.
3) Every new device type requires calibration, which is the current restriction on the calibration step. This, in turn, results in increased time and cost.
4) Limitation of existing ICMetrics system is that they do not focus on multimodal feature behaviour.

2.5 ICMetrics Related work

This section summarises previous studies involving the ICMetrics. As was mentioned in the background section, the technology was developed to identify the software and hardware features. The ICMetrics approach has been used in many earlier studies [33, 38, 39, 40, 41, 42] for their system.

The following list contains some of the most significant studies:

Hardware properties that can be leveraged for ICMetrics production have been uncovered in earlier investigations [43] [26] [25]. Experiments show that the Program Counter (PC) and Cycles Per Instruction (CPI) can be used to build an ICMetrics.

The Integrated Circuit Metrics (ICMetrics) technology was developed as a replacement for stored keys and as the foundation for a number of cryptographic applications [44] [45] [38]. The innovative idea and layout of the ICMetrics technology do not restrict its application as a substitute for stored keys. It is both feasible and advised to leverage a device's properties to generate an ICMetrics, which may subsequently be used to provide cryptographic services in a system, according to research [44] [45] [38] on the ICMetrics technology. The ICMetrics technique prevents key theft by completely doing away with the requirement for cryptographic keys to be kept.

In a research study [34] looked into intersatellite communications that operate in low power environments and generate encryption keys. It demonstrates methods that make it possible to create encryption keys based on characteristics or features that are actually connected to satellites, doing away with the need for key storage. It makes use of a prototyped feature extraction infrastructure circuit that is integrated into the embedded system-on-chip computing system of the satellite. It looked into the cache usage technique and the processor's software. Specifically, address and data value distributions.

A health care communication data encryption key strategy based on the ICMetrics was introduced by [46]. It developed metrics for the efficacy of the programme and the data caches using performance counters. These features included address and data buses from a system on chip and data values and programme addresses from a processor's data transaction. The features are then adjusted using a normalisation method so that they may be used by a standard pattern recognition system. Finally, a key for encryption was produced by combining all the attributes. The outcome of this study showed that it is possible to create an encryption key using the traits that were taken from systems on chip devices. The author [47] looked into the processor's programme counter and cycles per instruction. They then used these features to use the SOM to find anomalous behaviours (a self-organising map). An embedded device identification system utilising ICMetrics is presented in this study. It was suggested that SOM and the ICMetrics system may work together since different behaviours can be represented by different fundamental numbers and the key cryptography method can produce a variety of encryption keys as a result.

The findings show that the suggested strategy can identify unidentified behaviours that are not in the training set with an accuracy of over 98.4%. For embedded architectures such as function call sequences, internal control, and instruction streams within each function, the suggested work offers protection at many levels.

In addition, [48] investigated the programme counter as a potential source for extracting ICMetrics features and compared it to two tracing approaches, stepping and sampling, to acquire feature values. According to the findings of this study, PC values that may have been attained during execution do not provide for a strong ICMetrics feature. Instead, a powerful ICMetrics system might be developed using PC logs, sequence analysis, and frequency analysis.

Another piece of literature has suggested two strategies for guaranteeing device security [49]. The first form of security attempts to authenticate the user wearing the device, while the second technique strives to identify the gadget and secure it. These are two distinct paradigms, the first of which solely guarantees the wearer's security. The second approach secures the device, increasing the wearer's security. A faulty security implementation can result from focusing just on the device or the user, since cryptographic keys are held on a device.

The emphasis of the following chapters will be on enhancing ICMetrics through the handling of multimodal features in frequency distribution and safeguarding template feature data.

2.6 Competitive Analysis Matrix

This section provides in depth comparison between ICMetrics and various authentication technologies

Parameter	**Description**	**ICM**	**MPC**	**PUF**	**PKCS#12**	**TPM**	**HSM/ KMIP**
Robust end-point key protection	Does MIP technology provide strong key protection on its own or does it rely on third parties? For example, underlying device security features, etc.	Y	N	Y	N	Y	Y
Anti-malware capability	Can the technology itself protect a device from malware takeover? If not, after the takeover, the malware can skim the keys and impersonate the device or user. A robust MIP technology, ideally, should not rely on third party anti-malware solutions to secure the device.	Y	N	N	N	N	N
Database dependence	For the technology to operate, is there any dependence on a database at all? If so, the solution a) will not work in air-gapped networks and b) will incur costs such as database security (all eggs in one basket/single throat to choke)	N	Y	N	N	N	N

Prone to Inside Fraud	Is the solution prone to any form of insider fraud, for example, dependence on passwords?	N	Y	N	Y	N	Y
Latency	Is latency a barrier to securing a large number of keys in high-traffic environments, such as SSL keys on webservers?	N	Y	N	N	N	Y
Bespoke Hardware	Does the technology rely on a specific hardware component in a device? If yes, what if this specific component is not available or access to this component is blocked by a device manufacturer. Ideally, the technology should have the capability to leverage various device components simultaneously and ignore others if need be.	N	N	Y	N	Y	Y
Support offline crypto-operation	Does the technology require network connectivity to perform cryptographic operations? If yes, it is not viable in offline environments, for example IoT.	Y	N	Y	Y	Y	N
Immune to offline brute force attack	Is the technology immune against offline brute force attacks?	Y	Y	Y	N	Y	Y
Dynamic Features	Does the technology support dynamic features? i.e., the ability to allow measured features to vary while still generating a stable key. For example, CPU usage, memory I/O, etc. These feature values vary constantly and, as such, contribute towards higher entropy for key pair generation. On the contrary, with static features, such as disk serial number, OS version, and so on, the feature value remains constant, resulting in lower entropy, making it easy to brute force and vulnerable to insider fraud.	Y	N	N/A	N/A	N/A	N/A
Data obfuscation	Does the technology use raw feature values or leverage statistical and mathematical analysis to obfuscate these values? Raw feature values are prone to MITM attacks. For example, consider a possible attack vector: (1) A malware reads static feature values on the target victim device, (2) It skims the Device ID client and these feature values, (3) on a rogue device, the malware intercepts the	Y	Y	N/A	N	N/A	N/A

	communication between the client and OS, and (4) it passes the feature values fetched in (1) above to the client, thus tricking it into believing that it's still deployed on a genuine device.						
User behavior analytics	Does the technology support user behavior analytics to prevent friendly fraud?	Y	N	N	N	N	N
Persistent storage of key	Is any sensitive data, for example keys or feature values, persistently stored locally within the device or in a database? If they are stored at either of the two, there is always a chance that malware or insider fraud can expose the details and compromise the security. Even if these values are stored in encrypted format, who has access to them, and how secure is the encryption key?	N	N	N	Y	N/A	N/A
Know feature list and value	Is the device's feature list and feature value length known to anyone in the world? If it is, a rogue administrator, malware, and a brute-force attack can possibly breach the security.	Y	N	N/A	N/A	N/A	N/A
Random feature list and value	Does the device feature list and feature value length remain constant on all devices deployed at a customer's premise? If yes, once the hacker figures out how to crack one device, all other devices are vulnerable. On the contrary, if these are random on all devices, such a domino effect can be avoided.	Y	N	N/A	N/A	N/A	N/A
Broad spectrum device and OS support	Does the solution support deployment on various devices and OS types?	Y	Y	N	Y	N	Y
OTA crypto library update	If there is a vulnerability detected in the crypto library implementation, can it be updated over-the-air?	Y	Y	Y	N/A	Y	Y
Prevent side-channel attacks	Does the solution offer a high level of assurance against side-channel attacks?	Y	Y	N	N	Y	Y

Table2.1 Competitive Analysis Matrix

2.7 Comparison of Various Other types of Authentication Technologies

Since authentication is a key element in securing any environment, in this section, our contribution is to compare various types of authentication technologies with ICMetrics.

2.7.1 Biometrics Vs. ICMetrics

Biometric Authentication	**ICMetrics**
Provide third-factor authentication	Provide two-factor authentication
No Legal Tangibility	PKI is recognized by most IT Acts
Only the authentication layer	Provide multiple layers
There is no device locking	Device Locking and Identification
There are only ten resets available	Multiple resets are possible
Difficult deployment and management	Easy deployment and management
Not transparent to users	100% transparent to users
Not Scalable and costly	Highly scalable and cost-effective
False Positives	No false positives
No digital signature or encryption	Digital Signature and Encryption Layer
Cannot prevent against phishing, pharming	Prevention against Phishing 1.0, 2.0, and Pharming

Table2.2 Biometrics [51] vs ICMetrics [25]

2.7.2 OTP Vs. ICMetrics

OTP	**ICMetrics**
Weak 2FA	Strong 2FA
Prone to password sharing	Not prone to password sharing
Prone to skimming attacks	Not prone to skimming
The user experience deteriorates	No change in user experience
Multiple copies of credentials	Single copy of credentials
Prone to fraudulent administrator (inside attack)	Not Prone to Fraudulent Admin
Prone to MITB and brute force	Can prevent MITB & brute force
Prone to vishing and smishing	Can Prevent Vishing and smishing
Cannot perform encryption	Supports encryption
Limited legal tangibility	PKI is recognized by major IT Acts
No digital Signature only supports Electronic Signature	Asymmetric cryptographic digital signatures are supported

Table2.3 OTP [52] vs ICMetrics [25]

2.8 Security Issues

Since our research is primarily focused on device authentication, this section studies Verizon's Data Breach Investigations Report 2022, which highlights key paths leading to information security breaches including credentials and phishing (authentication), exploiting vulnerabilities, and botnets, and no organisation is safe without a plan to handle them all [13]. Below figure 2.1 shows basic Web application attacks breaches

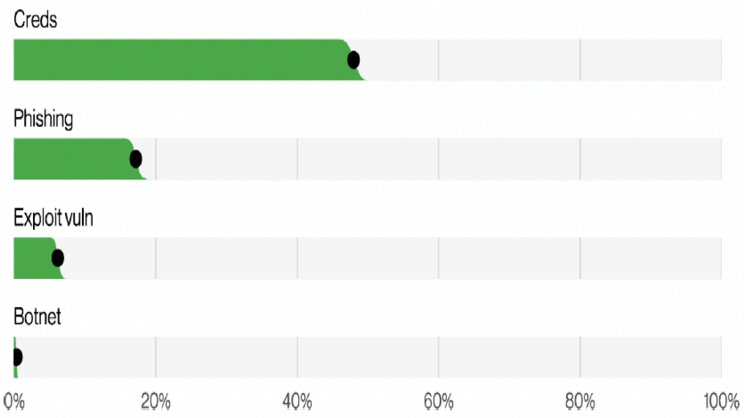


Figure 2.1 Top Action vectors in Basic Web Application Attacks breaches [13]

As described in figure 2.2, in Basic Web Application Attacks (BWAA), the report largely focuses on attacks that directly target an organization's most exposed infrastructure, such as Web servers. These incidents leverage one or both of two entry points: the use of stolen credentials or exploiting a vulnerability [13]. This pattern is still dominated by the use of stolen credentials to gain access to an organization's internet-facing infrastructure, such as web servers and email servers, and this is where ICMetrics can help [13].

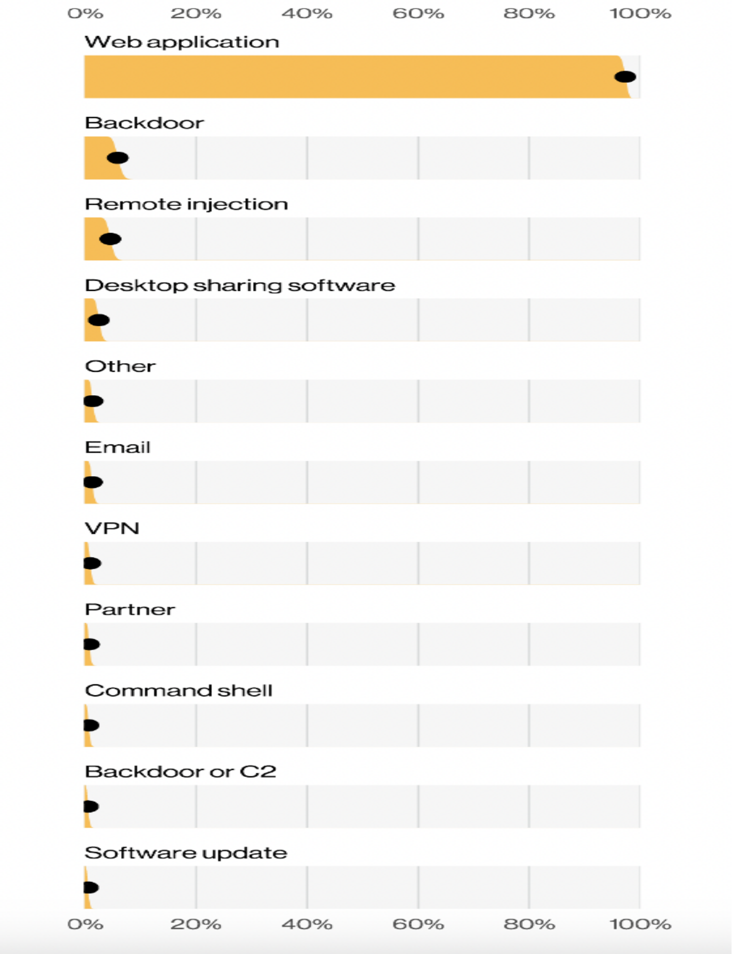


Figure 2.2 Top Action vectors in Basic Web Application Attacks breaches [13]

Figure 2.3 clearly displays how the vast majority of incidents involving Web applications use stolen credentials [13].

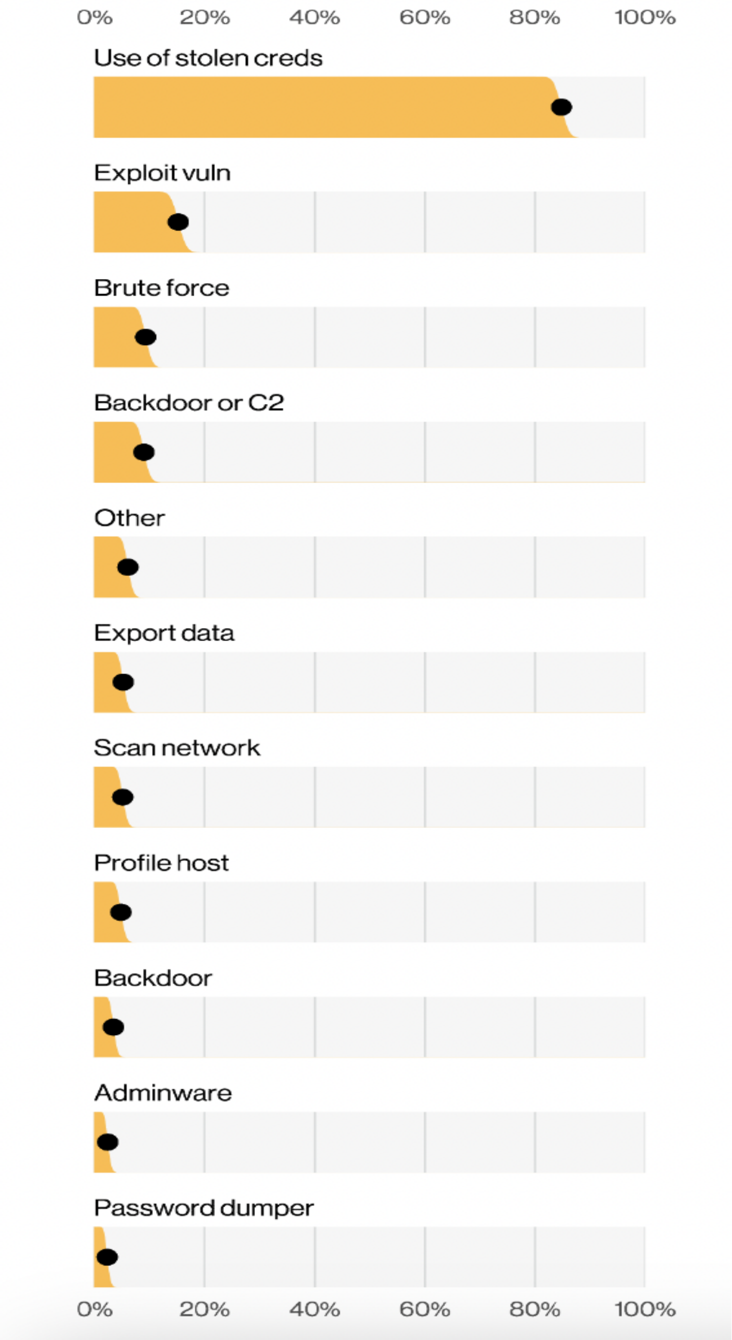


Figure 2.3 Shows incidents involving web application are using stolen credentials [13]

A system's flaws can frequently be used by adversaries to gain unauthorised access. Security is crucial as systems expand beyond the confines of homes and offices to more commonplace situations [53]. It is crucial to guarantee the security of any system's hardware and software components. A discussion of potential system attacks and how common they are in daily life is provided below.

2.8.1 Cyber security physical attacks on devices

In this section, the concern is with physical attacks on devices, where systems are vulnerable. This type of attack is carried out through penetration, monitoring, manipulation, modification, and substitution.

Physical tampering with hardware devices is a security issue. The processing and storing of data by hardware devices makes it crucial to safeguard hardware against attacks that could result in data theft or data modification [54]. If an adversary manages to physically access the cryptographic device or the area around it, such as smart card, then physical assaults are relevant [55].

According to research [56], physical devices can be tampered with using techniques like contactless radiation imprinting, contactless material removal, and probing. These attacks employ chemical and physical characteristics to gain unauthorised access to a system.

Data theft, forgery, and cloning are all possible outcomes of physical attacks [57] on systems. Data that has been captured is transferred to a cloned device, and a verifier is then persuaded that the device is legitimate. Cloned devices present a problem because their use can frequently go undiscovered. Cloning and counterfeiting can be prevented by implementing strong encryption and limiting access to the decryption keys [58].

Five attack scenarios are defined in [59]: penetration, monitoring, manipulation, modification, and substitution. These scenarios show the primary areas of physical attack.

An active, intrusive attack against the cryptography module is known as penetration. This entails breaching the module's cryptographic barrier. In order to find the secret keys kept inside the security module, it is intended to intercept data on internal communication lines or extract data from the memory.

Monitoring: The monitoring attack maintains the cryptographic border while being passive and non-intrusive. This class of attacks makes use of the cryptographic module's built-in leakage, such as through observing electromagnetic radiation. The two most common passive assaults based on monitoring are TEMPEST (Telecommunications Electronics Material Protected from Emanating Spurious Transmissions) investigations and side channel analysis.

Manipulation: A non-intrusive assault known as manipulation maintains the cryptographic barrier. The attacks mostly target the logical interface with the goal of inadvertently obtaining a service [59]. Changes in the surroundings may also be a component of manipulating attacks. For instance, the power supply and the ambient temperature may be used as harsh operating conditions for the cryptography module. These non-intrusive fault attacks fall under this heading.

Modification: Modification entails breaching the module's cryptographic border and is a direct, intrusive assault. The goal is to alter internal connections or the internal memories employed, as opposed to penetration attacks.

Substitution: Substitution entails removing the cryptographic module and replacing it with an imitation device that implements security functions differently. In this assault, the cryptographic border is not the main focus [59].

Next, the attack on communication is described in detail.

2.8.2 Attack on Communication

Through communication-based attacks, a malicious party can gain access to a network as a user or host and then obtain rights that allow the unauthorised use of authentication and authorisation. An attacker may try to seize the system's cryptographic keys once they have access. In an attack known as ‘IP spoofing’ [60] [61], the attacker fabricates IP addresses, which results in false IP packets. If done properly, an attacker can seize, redirect, alter, or remove data from the network. IP spoofing is particularly dangerous due to its online camouflage assault nature and difficulty in detecting it. In other types of communication attacks, malware finds vulnerable devices by looking for ones that are logged in with the factory default login and password. Once a device has been taken over, it is then employed as a bot to launch a Distributed Denial of Service (DDoS) attack by saturating a distant server with a significant amount of data [62] [63].

Eavesdropping is arguably the most frequent method of assaulting communication networks. Many wearable technology devices send data wirelessly, which makes them vulnerable to eavesdropping. By ensuring that the data is encrypted when it leaves a system, eavesdropping can be prevented [64].

The table 2.1 below highlights additional types of device-related communication channel attacks. As per UN regulation on vehicles with regards to cyber security management system, the following are the identified threats [65] [66].

High level and sub-level descriptions of vulnerability/ threat		*Example of vulnerability or attack method*
Threats to devices regarding their communication channels	Spoofing of messages or data received by the vehicle	**Spoofing of messages** by impersonation (e.g. 802.11p V2X during platooning, GNSS messages, etc.)
		Sybil attack (in order to spoof other vehicles as if there are many devices in vicinity
	Communication channels used to conduct unauthorized manipulation, deletion or other amendments to device held code/data	Communications channels permit **code injection**, for example tampered software binary might be injected into the communication stream
		Communications channels permit **manipulation** of device data/code
		Communications channels permit **overwrite** of device data/code
		Communications channels permit **erasure** of device data/code
		Communications channels permit introduction of data/code to the device (write data code)
	Communication channels permit untrusted/unreliable messages to be accepted or are vulnerable to session hijacking/replay attacks	Accepting information from an **unreliable or untrusted source**
		Man in the middle attack/ session hijacking
		Replay attack, for example an attack against a communication gateway allows the attacker to downgrade software or firmware of the device
	Viruses embedded in communication media are able to infect device systems	**Virus** embedded in communication media infects device systems
	Messages received by the device (for example X2V or diagnostic messages), or transmitted within it, contain malicious content	Malicious proprietary messages (e.g., those normally sent from OEM or component/system/function supplier)

Table 2.4 Highlights various other types of attacks on communication channels [65]

A description of cryptographic attacks has been presented in Section 2.2.3.

2.8.3 Cryptographic Attacks

An attacker's primary goal is to disassemble a cryptosystem and separate the plaintext from the cipher text. The attacker simply has to learn the secret decryption key to get the plaintext because the algorithm is already known to the general public.

As a result, he makes every effort to discover the cryptosystem's secret key. Once the attacker is able to determine the key, the system is regarded as broken or compromised.

According to research, cryptographic keys can be obtained through a wide variety of techniques, including brute force, malware, cold boot attacks, and more [67] [68] [69]. Key theft prevention can be a challenging undertaking because there are numerous assault tactics. Attackers try to capture cryptographic keys by taking advantage of system vulnerabilities or design faults. The following are some examples of potential key theft attacks.

By employing brute force, dictionary-based attacks, rainbow table assaults, man-in-the-middle attacks, etc., an attacker may try to break a cryptosystem. These types of attacks can be avoided by taking appropriate precautions, including increasing key size, integrating salts, and not using outdated algorithms [68].

Attackers may use another person's public key and claim it as their own. To demonstrate that the key is not being used as a fake identity, certification authorities need proof. There is an expanding certificate revocation list in web browsers as a result of certification authorities inadvertently issuing certificates to forgers in the past [73], [74].

The foundation of many cryptographic algorithms is algorithmic intractability, such as the use of huge prime numbers, factorability, etc. An adversary may find it simpler to attack the keys if the algorithm used to generate them is weak or poorly constructed. The keys must be produced by a reliable source, which is crucial. An enemy, for instance, may be able to generate a key by posing as a reliable source of information. By doing this, the attacker would not only be in possession of the keys but also have the ability to intentionally produce keys that lack the desired qualities [71], [72].

To get the keys from their owners, attackers may use psychological manipulation and persuasion [70]. The keys must be kept a secret from both insiders and outsiders at all times. Social engineering is a potent weapon that can be used to undermine security on multiple levels.

Even though the thesis' primary focus was on laptops, research on wearable internet of things devices were done in the beginning to evaluate their hardware. Cyber-attacks on wearable technology was investigated as part of this assessment.

2.8.4 Cyber Attacks on Wearable IoT device

Since wearable technology is being created and commercialised so quickly, security and risk management frequently seem to be neglected. Device vulnerabilities increase exponentially with the introduction of wearable technology. These devices have the potential to facilitate the compromise of data integrity, availability, and confidentiality for individuals and organisations if they are not properly controlled or governed [75]. Due to the individuality of each wearable device and the hazards associated with the data it gathers, including the user's location, the wearer is now more exposed than before [75].

Research [76] shows that as wearable technology becomes more popular, it seems that producers and developers are more concerned with boosting elements like design aesthetics and power consumption than with security dynamics. Due to a lack of authentication, authorization, and secure information transfer methods, wearable devices have a security and privacy issue [76], [77].

Gait recognition is one of several techniques for verifying the identity of a device's user. The idea of employing gait recognition to authenticate wearable technology has been thoroughly investigated [78], [79]. In their study [80], the authors propose a security system for Google Glass, an optical wearable. The authors describe a discrete security system that establishes user authenticity through a variety of user gestures. The concept is intriguing, but it lacks credibility because it depends on human input. The requirement that the user have prior Google Glass experience for enhanced accuracy is another flaw in the proposed approach.

Identification of a person via a system or device's built-in mechanism is called authentication. Without authentication, a hacker who is not a legitimate user of the hosting device can access resources, including services and information. At the moment, Fit bit smart watches lack an internal security system [76]. Fitbit can compromise someone's location and personal information without verification. Absence of strong device authentication can send a hacker to a network entry point where they could potentially abuse the system. If a vulnerability of this kind is effectively exploited, it can affect data storage other than that on a single device, allowing access to personal information, passwords, emails, and digital material. As a result, significant attacks and identity theft can occur [76], [81], [82], [83], and [84].

Research [76], [84] says some smart watches require a Bluetooth Low Energy (BLE) connection to another device before they may connect directly to the Internet. The IEEE 802.15 standard for wireless personal area networks now incorporates BLE (wireless personal area networks). Man-in-the-middle (MITM) attacks are more likely to succeed on connected devices with shaky connectivity or Bluetooth connections [76] [85]. According to the author [84], wearable devices inherit the same communication vulnerabilities that most Bluetooth devices do, such as message tampering, denial of service (DoS), and eavesdropping attacks.

According to a study [86], even wearable technology that has received widespread marketing can have weak security features that make hacking simple. The Fit bit tracker, which has 96 KB of RAM and an altimeter and accelerometer sensor built in, is the subject of this research. The Fit bit tracker's security is examined in the article, which demonstrates that the wearable technology can be attacked by taking advantage of flaws in its defences. The authors who reverse-engineered it, discovered that Fitbit lacks security features. The tracker, for instance, sends user credentials in plain text. Additionally, any HTTP data processing that occurs is likewise done in plaintext. The authors also show how, by attaching the tracker to moving objects, fake data may be created and injected into the device. Table 2.2 shows various types of attacks on devices [65].

High level and sub-level descriptions of vulnerability/ threat		*Example of vulnerability or attack method*
Threats to devices regarding their update procedures	Misuse or compromise of update procedures	Compromise of over the air software update procedures. This includes fabricating the system update program or firmware
		Compromise of local/physical software update procedures. This includes fabricating the system update program or firmware

High level and sub-level descriptions of vulnerability/ threat		*Example of vulnerability or attack method*
Threats to devices regarding unintended human actions facilitating a cyber attack	Legitimate actors are able to take actions that would unwittingly facilitate a cyber-attack	Innocent victim (e.g., owner, operator etc.) being tricked into taking an action to unintentionally load malware or enable an attack
Threats to device data/code	Extraction of device data/code	Extraction of copyright or proprietary software from device systems (product **piracy**)
		Extraction of cryptographic keys
	Manipulation of vehicle data/code	Illegal/unauthorized changes to **device's electronic ID**
		Identity fraud. For example, if a user wants to display another identity when communicating with other systems,
	Introduction of malware	Introduce **malicious software** or malicious software activity
	Cryptographic technologies can be compromised or are insufficiently applied	Combination of short **encryption keys** and long period of validity enables attacker to break encryption
Potential vulnerabilities that could be exploited if not sufficiently protected or hardened	Physical manipulation of systems can enable an attack	**Manipulation of electronic hardware**, e.g. unauthorized electronic hardware added to a device to enable 'man-in-the-middle' attack **Replacement of authorized electronic hardware** (e.g., sensors) with unauthorized electronic hardware

Table 2.5 Highlights various other types of attacks on devices [65]

2.8.5 Mitigation for device attacks

Below are the mitigations for device and communication channel attacks as highlighted in tables 2.3 [65].

Threats to ‘device communication channels’	Mitigation
Spoofing of messages (e.g., 802.11p, GNSS messages, etc.) by impersonation	The Device shall verify the authenticity and integrity of messages it receives
Sybil attack (in order to spoof other devices as if there are many devices)	Security controls shall be implemented for storing cryptographic keys (e.g., use of Hardware Security Modules)
Communication channels permit code injection into device held data/code, for example tampered software binary might be injected into the communication stream	The device shall verify the authenticity and integrity of messages it receives
Accepting information from an unreliable or untrusted source	The device shall verify the authenticity and integrity of messages it receives
Man in the middle attack / session hijacking	The device shall verify the authenticity and integrity of messages it receives
Replay attack, for example an attack against a communication gateway allows the attacker to downgrade software or firmware	
Interception of information / interfering radiations / monitoring communications	Confidential data transmitted to or from the device shall be protected
Virus embedded in communication media infects device systems	Measures to protect systems against embedded viruses/malware should be considered
Malicious messages e.g. infrastructure to device or device- device messages	The device shall verify the authenticity and integrity of messages it receives
Malicious diagnostic messages	
Malicious proprietary messages (e.g., those normally sent from OEM or component/system/function supplier)	

Mitigations to the threats which are related to ‘Update process’	Mitigation
Compromise of cryptographic keys of the software provider to allow invalid update	Security controls shall be implemented for storing cryptographic keys

Threats to ‘External connectivity and connections’	**Mitigation**
Corrupted applications or those with poor software security, used as a method to attack device systems	Software shall be security assessed, authenticated and integrity protected. Security controls shall be applied to minimize the risk from third party software that is intended or foreseeable to be hosted on the device
Extraction of cryptographic keys	Security controls shall be implemented for storing cryptographic keys e.g. Security Modules

Mitigations to the threats which are related to ‘Data loss / data breach from device’	**Mitigation**
Information breach. Personal data may be breached	Best practices for the protection of data integrity and confidentiality shall be followed for storing personal data.

Mitigations to the threats which are related to ‘Physical loss of data loss’	**Mitigation**
Damage caused by a third party. Sensitive data may be lost or compromised due to physical damages to device.	Best practices for the protection of data integrity and confidentiality shall be followed for storing personal data. Example Security Controls can be found in ISO/SC27/WG5
Loss from DRM (digital right management) conflicts. User data may be deleted due to DRM issues	
The (integrity of) sensitive data may be lost due to IT components wear and tear, causing potential cascading issues (in case of key alteration, for example)	

Table 2.6 Mitigation for device and communication channels [65]

2.8.6 Sky Go Report

This security research report shows how a real-world attack on a high-end luxury car was carried out. Then mitigation using ICMetrics is highlighted.

Sky-Go research demonstrated how remote cars can be hacked in the following phase [87]:

2.8.6.1 Use Case # 1: Remote access

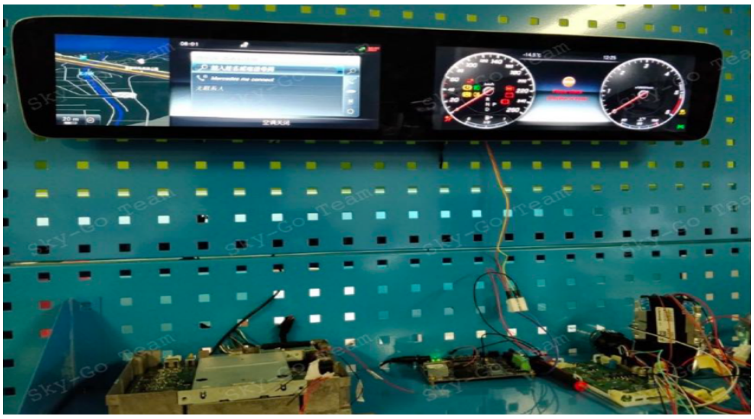


Figure 2.4 Test bench [87]

1) Dumping the firmware from NAND flash provides access to the file system.
2) Client Certificate: This file system exposes the client certificate in a .pfx file, password, and CA certificate for the backend server [87].

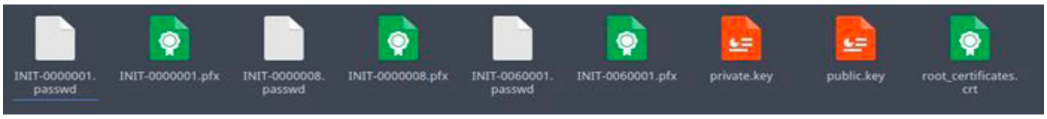


Figure 2.5 Certificates and Key pair [87]

[**ICM advantages**]: This technology does not rely upon insecure .pfx files to store the private key. In fact, the private key is not stored permanently. ICM offers a much more robust technique. Even if authorised access to the file system is possible, the private key cannot be used outside the Telematics control unit (TCU) because the TCU's digital fingerprint has changed [25], [34].

3) Access the Backend: Using the e-SIM from the car, connecting it to a local broadband router, and spoofing the IMEI number of the router (with APN (access point name) information from the car), the backend will trust the incoming connection. As per the research paper 'Car Backend is the core of Connected Cars'. As long as Car Backend services can be accessed externally, it means that Car Backend is at risk of being attacked [87]. 'The vehicles connecting to this car backend are in danger too'.

[**ICM advantages**]: An IMEI number, like the MAC address on a NIC card, is easily spoof able. To avoid this attack vector, possible techniques include a) using the TCU's private key to digitally sign the login request to the backend server and b) using eSIM as one of the components to calculate the TCU's digital finger print. When the eSIM is extracted from the

TCU, its digital fingerprint changes and ICMetrics fails to generate the correct private key, resulting in a failed rogue login [25] [35].

4) Protocol Analysis: Every time the TCU reboots, a new shared key is pushed to the car, which is equally vulnerable as the old key [87].
   [**ICM advantages**]: The fundamental reason for these short-lived keys is protection of the key. The current technique cannot assure private key protection; hence, they are renewed frequently. ICM can bind the private key to the TCU in a very effective way, so there is no need for 'short' key renewal. Furthermore, shared keys should be replaced with asymmetric keys to reduce management and security overheads on the database to store keys and encrypt shared keys [37] [27].

2.8.6.2 Use Case #2 – Field Update of ECU Software

The context: automobile manufacturers have a need to remotely upgrade the software of engine control units (ECUs) in a secure fashion. For enhanced security, the software is usually digitally code-signed to avoid a MITM attack. Such an attack can potentially push malware to the ECUs, which in turn can lead to severe consequences, such as fatally crashing the car, etc. HSM is typically used to protect the private key used for code signing [87].

The challenge: One possible attack vector could be replacing the code signing certificate with a rogue certificate, so that the ECUs will trust the code and execute malware. To explain further, after the code is signed by a private key at the server side, the ECU or TCU verifies it using the code signing certificate (public key), which is embedded in the file system as described in Use Case #1 above. If the bad actors replace this certificate with a rogue certificate, sign malware with their own (rogue) private key, and push it onto the ECU, the ECU will trust the malware and execute it [87].

The solution: ICMetrics client software can bind the manufacturer's code signing certificate (public key) on the file system such that it cannot be replaced by a rogue certificate or misused in any possible way. Effectively, manufacturers can implement a robust certificate store on the ECU device that cannot be tampered with. Although PKI is the de-facto standard for secure client authentication, if the manufacturer is not keen on implementing PKI, ICMetrics can also support similar high-level assurance in a non-PKI environment [34] [27].

2.9 Summary

As a precursor to device authentication technique, this chapter thoroughly examines existing authentication approaches and their limitations, including physically unclonable functions, TPM, MPC, and PFX. Then the chapter dives extensively into the cutting-edge ICMetrics technology and previous work related to ICMetrics. ICMetrics technology has been investigated as an important identity theft prevention method and as the foundation for a number of cryptographic services. In order to establish an identifier known as an ICMetrics, the technique suggests utilising unique features of a device. In order to produce an ICMetrics that is really unique, the ICMetrics technology relies on distinctive reproducible explicit and implicit characteristics. Additionally, the two stages of ICMetrics production have been provided for processing a variety of potential features that will be covered in the following chapter. Next, the focus was on extensive competitive analysis, and all possible relevant authentication technologies comparison were covered. Thereafter, the focus was on security concerns as we know a system can be attacked in a variety of ways, thus this chapter makes the case that attackers will try to take advantage of communication, physical, or cryptosystem flaws in order to get unauthorised access. Attacking a system in order to obtain the credential is a common objective of adversaries. Lastly the focus was on a very specific ECU attack vector. The extensive focus was on threat mitigation via ICMetrics highlighting at each step how ICMetrics helps in preventing the attack.

Chapter 3

General Computing Device Authentication

3.1 Introduction

The focus of this chapter is to evaluate the feasibility of identifying devices uniquely based on hardware features derived from the properties and behaviour of general-purpose computing devices [88]. In order to achieve this, appropriate methods of extracting hardware features were investigated and potential features that were suitable for identification were explored.

With the use of ICMetrics technology, a unique device identifier is created that can be used for a variety of security functions, including key generation, authentication, integrity, and privacy. Utilising ICMetrics technology, device identification is provided for the purpose of security services. Exploring suitable features to produce an individual device identification is necessary for a system based on ICMetrics technology.

In common cybersecurity parlance, the importance of strong authentication is greater than ever before. Regulatory requirements, such as EU eIDAS, also mandate robust authentication. As per the Department for Digital, Culture, Media, and Sport (DCMS), in the last 12 months, 39% of UK businesses identified a cyber-attack and 83% of these identified the most common threat vector as phishing attempts [89]. Further, 80% of companies experienced a breach related to a weakness in authentication. This costs them an average of $2.19 million per year [90]. Under the current circumstances, traditional approaches like hardware tokens are expensive to deploy and manage and are ineffective against some threats [91]. The challenge was to deploy a technology that is both easy to use and strong enough to protect against sophisticated attacks like malware, man-in-the-middle attacks, etc. ICMetrics is a secure software credential that combines protection for digital identities similar to that of a hardware smart card, and with its ease of use and distribution, it offers lower costs for deployment and maintenance. The ICMetrics is ‘something you have’ and the optional

ICMetrics password is ‘something you know’ necessary for two-factor authentication. As a software-based solution, ICMetrics enables organisations 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 a similar level of security in a non-PKI mode as well [92].

ICMetrics is a unique technology for deriving private keys based on the digital fingerprint (software and hardware configurations) of the device [93]. 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 while still ensuring that any illegal clone or malware attack is detected [94]. Such a system will offer the following significant advantages: 1) Eliminate the need to store any credential-related sensitive data within the device, thus addressing the major weakness that can be used to circumvent the security offered by the system. 2) In a malware attack, ICMetrics behaviour analysis helps detect tampering with the constitution of the software and will cease the authentication process.

The novelty of this research work was that new hardware features from different devices (MacBook Air, MacBook Pro) were investigated, shortlisted based on good feature criteria, and then analysed and applied in our proposed model. For benchmarking, conventional classifiers for device identification were used.

In this chapter, hardware feature data offered by devices are used to create an ICMetrics unique identifier. An identity can be generated by a device using the ICMetrics technology, which is used for various security applications. This chapter explores the potential of using device-generated features to provide an ICMetrics.

The rest of the chapter is organised as follows:
Section 3.2 describes the ICMetrics Process, Section 3.3 explains feature extraction concept, Section 3.4 focuses on feature modelling, Section 3.5 on feature analysis, Section 3.6 explains multimodal distribution, Section 3.6 details the operation phase of the system, Section 3.7 focus on system’s methodology and algorithms, Section 3.8 give experiment and result details and finally Section 3.9 highlights chapter’s conclusion.

These sections give us an in-depth understanding of the feature data characteristics, how it is gathered, and how to create an efficient classification model.

3.2 Proposed ICMetrics Security System

In the previous section, although a general description was provided, here the specific focus is on the adaptation of how each phase is applied for device identification.

ICMetrics technology relies on measurable features that are obtained from a certain system's characteristics. The emphasis is on making use of the hardware features that are provided in the new system.

It is necessary for this research to use Mac laptops, and the hardware attributes that can be extracted from these devices are used to create the ICMetrics security system. These characteristic data are utilised to generate an ICMetrics unique identifier, which is then used to identify devices. The following parts go into further detail about the suggested system:

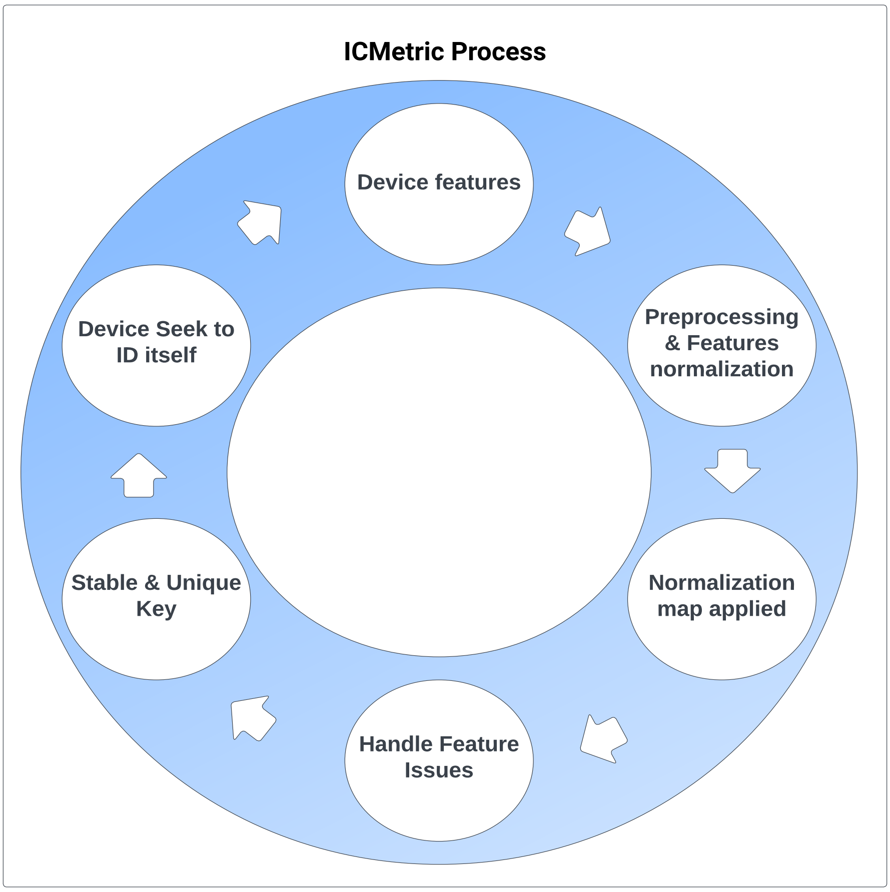


Figure 3.1 ICMetrics Process

The following sections will go into more detail about the calibration phase and the operation phase.

3.3 Feature Extraction

This research took into account the hardware features that could be gathered using general computing devices. Relevant features were retrieved from devices in this procedure to create feature vectors. These feature vectors are then used by classifiers to differentiate between the desired output unit and the input unit. These properties enable the classifier to discriminate between devices more clearly, simplifying categorization. The technique of extracting the most crucial information from raw data is known as feature extraction. The extraction of features should be simple, resource-efficient, and atomic.

3.3.1 Calibration

The calibration phase is useful in pre-production to extract suitable features with the goal of providing sufficient correlation when combined. By combining device features appropriately, an ICMetrics system can form a unique identifier for device identification. Calibration is carried out once per application domain. The suitability of device features depends on the nature of the device. In most cases, it will include surveying a device for a set number of days or periods and gathering stable values. In other cases, it may be variable features that are likely to change over time. This leads into static and dynamic features [50] [95].

The experimental data collection is described in this section, along with the scenarios and devices that were selected. It investigates the analysis of our collected data.

3.3.1.1 Devices

This was a very limited identification scenario. Even though there was lack of test devices, the research required a minimum of eight MacBook laptops for better indication of the performance of the feature and feature combination of the system. In order to find features that can provide sufficient dynamic range and variation, for example enough sample such that distribution is stable and shows clear pattern. MacBook Air and MacBook Pro were chosen since they allow access to both the low-level hardware and software present on each device. Since Apple produces MacBook, the hardware tends to be more homogenous, and based on these traits, the feature values may be segregated, in contrast to Windows, whose hardware is produced by various vendors and makes it difficult to differentiate the devices. To create the widest possible range of feature values, data from various Mac devices were gathered as part of the study of each feature. Some of the tested devices have various OS iterations, settings, and programmes. The same-model devices with identical chipsets in the collection were the most difficult to distinguish from one another using hardware and software features. The table below displays information on the device processor, memory, and storage.

Devices	**Processor**	**Memory**	**Storage**
MacBook Air (13-inch, Mid 2013)	1.3GHz dual-core Intel Core i5	8GB 1600 MHz DDR3	128 GB Flash Storage
MacBook Air (13-inch, Mid 2014)	1.4GHz dual-core Intel Core i5	8GB 1600 MHz DDR3	128 GB PCIe-based Flash Storage
MacBook Air (13-inch, Early 2015)	1.6GHz dual-core Intel Core i5	8GB 1600 MHz DDR3	128 GB PCIe-based Flash Storage
MacBook Air (13-inch, Early 2017)	1.8GHz dual-core Intel Core i5	8GB 1600 MHz DDR3	256 GB PCIe-based SSD
MacBook Air (13-inch, Early 2019)	1.6GHz dual-core Intel Core i5	8GB 1600 MHz DDR3	128 GB PCIe-based SSD
MacBook Pro (13-inch, M1 2020)	Apple M1 chip 8-core CPU with 4 performance cores and 4 efficiency cores 8-core GPU 16-core Neural Engine	8GB	256GB SSD
MacBook Pro (13-inch,2018)	2.3GHz quad-core Intel Core i5	8GB of 2133MHz LPDDR3	256GB SSD
MacBook Pro (13-inch,2019)	1.4GHz quad-core Intel Core i5	8GB	128GB SSD

Table 3.1 Device List

The calibration phase contains four parts, i.e., (a) data collection, (b) feature selection, (c) feature modelling, and d) feature analysis, which describe the processes sequentially undertaken prior to building a model.

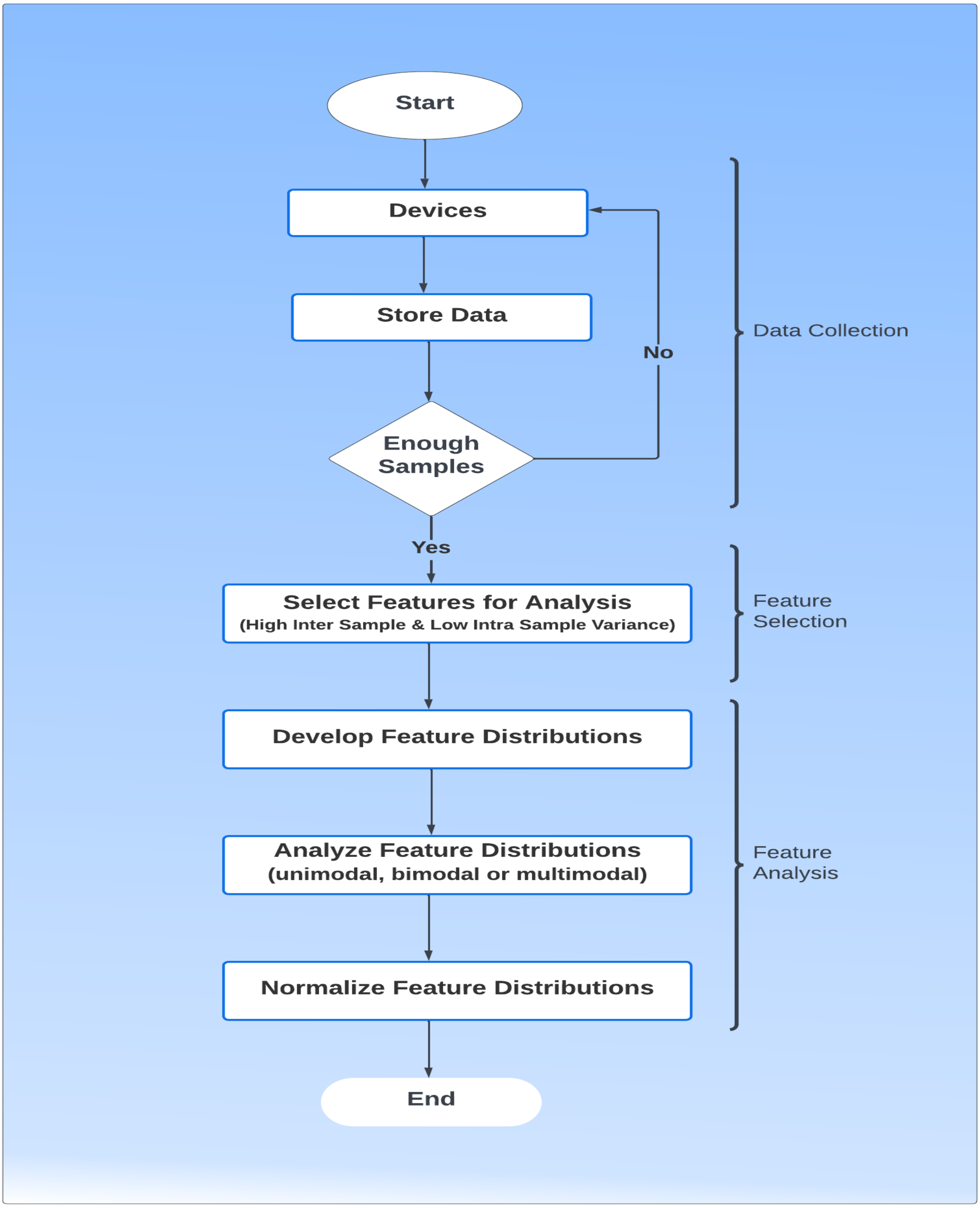

Start
Devices
Store Data
No
Enough Samples
Yes
Select Features for Analysis
(High Inter Sample & Low Intra Sample Variance)
Develop Feature Distributions
Analyze Feature Distributions
(unimodal, bimodal or multimodal)
Normalize Feature Distributions
End
Data Collection
Feature Selection
Feature Analysis

Figure 3.2 Flowchart of Calibration Phase

The device characterizations employed by the system are known as ‘features’. Features are a major part of the ICMetrics system, and the features utilised directly influence the strength of the security provided. The data collected from the devices described here are the Apple laptops, namely the MacBook Pro and MacBook Air. With weak features, i.e., features that do not change at all with the functionality and restrict the ICMetrics system in how much security it can offer, a feature value is ultimately used to identify a device. So, the more discriminative it is, the better when evaluating the security of an ICMetrics system [96] [88]. The values of features are determined by how the machine is used. Ideal feature candidates can provide the basis for a secure system that can guarantee an increase in the trust associated with existing security protocols. The analysis and mapping techniques allow the system to incorporate features whose values can change while still being able to transform these dynamic values to distinguish a device. To facilitate this, features that exhibit low intra-sample variance and high inter-sample variance are selected as priorities for the mapping process. Because the values of features employed in the ICMetrics system can change, the feature behaviour and the influences on that feature value need to be understood before an ICMetrics value can be reproduced consistently. The methodology of calibration phase (Figure 3.2) is described in detail below till section 3.5.

3.3.2 Data Collection

The contribution of this work was to focus on hardware features to identify the device uniquely. For this, features are required to provide distinguishability for similar devices using the same features for example MFLOPS, random seeks\sec, random seeks%cpu behaves differently on each device. Although the devices have same hardware, based on user usage, the performance level differs which help us to distinguish devices. The features not only have to provide sufficient variance but additionally; the features have to remain obscure to any unauthorized access, therefore the features need to come from a variety of sources on the device to prevent easy discovery of the features that are included for example in our analysis features are categorised from three different areas like hard disk performance, memory performance and CPU Floating point performance using different categories of features helps us provide sufficient variance and good obscurity.

In order to allow for a wide range of features, the particular focus of this study has been due to the great variety of devices it provides, which allows for an in-depth analysis of how the features affect the system. Thus, the data is gathered from multiple devices in order to fully ascertain the range of each particular feature's values. The devices tested included a variety of devices with identical chipsets of the same model. They were tested to obtain data from devices that would be difficult to distinguish based on the hardware and software alone. The data collection system was run on each device repeatedly. System variables such as the running processes, power settings, system updates, and user activity were monitored to examine the variance in the data produced.

In earlier research on ICMetrics generation, a variety of relevant qualities have been sought after. According to earlier research [122][123][124] on hardware features for ICMetrics production, a device ICMetrics can be produced using the programme counter and cycles per instruction. The bias in MEMS sensors was also exploited as a distinctive feature in a recent study [125]. The authors demonstrate statistically that there is a bias that is adequate to identify a specific wearable health gadget using the accelerometer and gyroscope. The authors set up a test bed of various sensors to demonstrate that, assuming the device and its internal environment are not corrupted, each sensor has a distinct and reproducible bias. An excellent feature will have a large inter-sample variance, providing a wide range of possible versions. A basic serial number with a very significant inter-sample variation would be an illustration of this since each device's serial number is entirely unique. Unluckily, these serial numbers likewise have very little obfuscation. As a result, both a feature's positive and bad aspects must be taken into account while making a decision.

Various features generate non-static values, which change while the device is in used. The amount of available RAM on the device right now is an illustration of such a feature. While the value should be fairly constant, it will change as programmes are launched (consuming RAM) or shut off (releasing RAM). How much these values change over a device's operational time is described by intra sample variation. The free RAM that is currently available on a device will always be located someplace along the axis but will occasionally move up and down. A specific code was designed to capture the data from the OS and it looks for intra and inter-sample variance. These dynamic operating conditions tends to make a device unique and they are modelled based on the frequency distribution of the feature behaviour. It is explained in detail below in section 3.5.1. This feature is too unstable to be recorded over time on its own, but it can be controlled if we define ranges of values, in this example ranges of 400MB. These ranges are all mapped to arbitrary values, and as long as the characteristic remains inside the range, the resulting arbitrary value will not change. The intra sample variance of the feature must be as low as it can be since features with a large intra sample variation fluctuate too much to be used in this manner. In the end, this is impossible to map and cannot provide a persistent value.

The data is collected using Python code in a monitored natural environment, which gives an insight into the behaviour of the features during the analysis. The features extracted are uniquely affected by each user’s machine usage (as each user uses their machines differently). This allows 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 analysed. The features were initially narrowed down through a variety of techniques, including the analysis of their variance and their correlations with other features to identify stable correlations that were distinctive to any set of devices. This will lead to a greater understanding of feature correlations per device in order to exploit their internal relationships. It is not just framework measures that might actually influence low-level hardware feature values. User-controlled cycles could likewise adjust the distribution of a feature. To help with this problem, the device is observed and recorded when data are read for analysis.

The selected features were then divided into sets in order to improve operational robustness by utilising Shamir's secret sharing to allow for controlled potential partial failure of the system whilst still retaining some security verification. These feature sets are more reliable than individual features, and generate a stronger base for applying the ICMetrics system [93].

A system's features are a result of its internal environment and can be divided into three categories: user data, system specific, and device settings. As stated in second paragraph of same section above, the methodology to select these categories is as follows:

1) User Data would assist in user behaviour analytics. For example, how they are using their machine in terms of how many tabs are open, what applications are running etc.
2) System specific data helps to uniquely identify the device for example CPU usage, memory I/O
3) Device settings are crucial since the configuration can be changed thus allowing flexibility to the users and at same time offers additional device features for enhanced security.

The next section discusses the three feature categories.

The internal setting of a system affects the features of that system. One can be tempted to start with obvious elements like a MAC address while thinking about system features. Despite being a system feature, the MAC address can be viewed as being weak because it is simple to acquire using typical network sniffing tools. ICMetrics can be generated using features that are specific to the system yet challenging for an attacker to capture. Every device has a CPU, and since every device has a different type of CPU, CPU information is used to create an ICMetrics measure. A number of characteristics of the CPU hardware, including the amount of memory assigned to buffers, the available memory, dirty memory, and shared memory, are employed in the development of the ICMetrics.

According to research [125], a device's internal settings can be used to calculate its ICMetrics. Serial numbers, calibration settings, inbuilt identifying modules, Bluetooth identification, SSID, etc. can all be used to do this. Using just one device setting can jeopardise the security of the ICMetrics implementation because device settings like Bluetooth identification are adjustable. Although a device's serial number is unique, it should still be used with caution because it is easy to fake them because they frequently appear on the outside of the device. The DS-2411 [126] is an illustration of a robust serial identification chip, on the other hand. This single line chip, which was lasered at the factory, offers serial identification-based services. Cloning, fabrication, and spoofing-based attacks are prevented since once the chip has been laser-engraved with a serial number, it cannot be changed in any manner. The calibration parameters of a device may also serve as the foundation for the ICMetrics of that device. This is a key feature for ICMetrics creation because the calibration of a sensor is frequently based on numerous variables. The hardware and software environments of identical devices that have been mass-produced by a single vendor will be comparable. Due to the availability of specific user data, if the devices are used by two distinct users, the internal environment will be sufficiently different. Once more, it's critical to build the ICMetrics generation on characteristics that are specific to a user and unpredictable to an enemy.

These three categories of features (hard disk performance, memory performance and CPU Floating point performance) were selected since they help to uniquely identify the devices based upon user behaviour. For example, on a similar type of device two users will have different sets of applications installed and running, data, etc.; hence, this helps to differentiate the devices.

The following features are in scope, which include a range of categories:

3.3.2.1 CPU Usage

CPU time (also known as process time) measures the amount of time a central processing unit (CPU) was put to work processing instructions from a computer programme or operating system, as opposed to, say, waiting for input/output (I/O) activities or going into low-power (idle) mode. Clock ticks or seconds are used to represent CPU time. CPU utilisation, also known as CPU time, is frequently expressed as a percentage of CPU capacity.

CPU time and CPU utilisation have two primary applications. The system's total level of bustle can be measured using the first application. The user can lag if the CPU use is beyond 70%. An insufficient amount of processing capacity is evident from such high CPU consumption. Either the CPU needs to be updated or the user experience needs to be scaled back, for instance by switching to graphics with lower quality or fewer animations. With the introduction of multitasking, the second application is to measure how the CPU is distributed among computer programmes. A piece of software that uses a lot of the CPU can be very resource-intensive or it might be broken—for instance, it might have become stuck in an endless loop.

3.3.2.2 Memory Performance

In other words, it's the measurement of all memory-related metrics on the host or device. These attributes were extracted because even the tiniest changes in the processes running on devices are reflected in them. One element of performance is memory bandwidth, or how quickly data can be written to or read from memory by the processor. Applications' loading and unloading times have a big impact on how quickly the OS can handle data. If memory bandwidth is low, the CPU can be waiting on memory to read or write data. Data that the CPU needs to read or write can be done quickly if memory bandwidth is high.

A simple, synthetic benchmark programme called STREAM (Sustainable Memory Bandwidth in High Performance Computers) determines the processing speed for simple vector kernels based on the sustainable main memory bandwidth in MB/s [97].

STREAM uses four kernels to inform its analysis:

1. ‘Copy’ computes transfer rates devoid of mathematical requirements.
2. The term ‘scale’ encompasses a simple mathematical operation.
3. To enable testing of multiple load/store ports on vector machines, ‘Sum’ adds a third operand.
4. ‘Triad’ enables the chaining, overlapping, or fusing of multiply/add operations.

3.3.2.3 Block I/O

The control and monitoring of tasks in cgroups' access to I/O on block devices is handled by the Block I/O (blkio) subsystem. Writing values to some of these pseudo files restricts access or bandwidth, while reading values from some of these pseudo files provides information on I/O operations [98].

The BLKIO subsystem has two policies for restricting access to I/O:

• Proportional weight division, which allows you to apply weights to certain cgroups and is implemented in the CFQ I/O scheduler.

I/O throttling (Upper Limit) is a policy that is used to set a maximum number of I/O operations that can be performed by a single device. This portion is set aside for each cgroup and is based on the cgroup's weight. As a result, a device is limited in how many read or write operations it can perform each second.

Currently, buffered write operations are incompatible with the Block I/O subsystem. Although it is primarily made for direct I/O, it may perform buffered read operations.

3.3.2.4 Floating Point Operations Per Second

FLOPS is a unit used to measure how many floating-point operations a microprocessor can perform in a second. In areas of scientific computing where floating-point calculations are required, FLOPS is useful. In certain situations, it is a more accurate statistic than counting instructions per second [99].

3.3.2.5 Input/output Operations Per Second (IOPS)

IOPS is a metric used to determine how many read and/or write operations a storage device can complete in a specific length of time. Block sizes on hard disk drives (HDD) are commonly 512 bytes or 4 KB. By itself, the IOPS statistic says nothing about how much data a drive can store. The maximum number of bits or bytes that can be devoted to a single I/O transaction, ‘block size,’ is what determines this amount together with the IOPS. For instance, when the IOPS value is the same, the drive with the larger block size may process more data (read or written).

IOPS may vary based on whether data is accessed sequentially or ad hoc. On HDDs in particular, IOPS is frequently higher for sequential writes due to the ease with which the disc head can access auxiliary blocks. On the other hand, random reads and writes require shifting the disc head to the necessary location. There may be differences in the read and write IOPS values [100].

Overall, 38 features were collected and named as shown in Table 3.1, out of which 17 were used for this analysis. The features that can provide critical information for characterization are kept, and the rest are ignored for this work.

Sr. No.	Feature Name	Feature Category
F1	max speed (MB/s) for copy function	Memory Performance
F2	avg. duration for copy function	Memory Performance
F3	quickest duration for copy function	Memory Performance
F4	longest duration for copy function	Memory Performance
F5	max speed (MB/s) for scale function	Memory Performance
F6	avg. duration for scale function	Memory Performance
F7	quickest duration for scale function	Memory Performance
F8	longest duration for scale function	Memory Performance
F9	max speed (MB/s) for add function	Memory Performance
F10	avg. duration for add function	Memory Performance
F11	quickest duration for add function	Memory Performance
F12	longest duration for add function	Memory Performance
F13	max speed (MB/s) for triad function	Memory Performance
F14	avg. duration for triad function	Memory Performance
F15	quickest duration for triad function	Memory Performance
F16	longest duration for triad function	Memory Performance

Sr. No.	Feature Name	Feature Category
F17	Norm. Resid	CPU Floating Point computing Power
F18	Resid	CPU Floating Point computing Power
F19	MACHEPX [1]	CPU Floating Point computing Power
F20	X[N]	CPU Floating Point computing Power
F21	Factor	CPU Floating Point computing Power

F22	Solve	CPU Floating Point computing Power
F23	Total	CPU Floating Point computing Power
F24	MFLOPS	CPU Floating Point computing Power
F25	Unit	CPU Floating Point computing Power
F26	Cray-Ratio	CPU Floating Point computing Power
F27	sequential output (per char) M/sec	Hard disk Performance
F28	sequential output (per char) %CPU	Hard disk Performance
F29	sequential output (block) M/sec	Hard disk Performance
F30	sequential output (block) %CPU	Hard disk Performance
F31	sequential output (rewrite) M/sec	Hard disk Performance
F32	sequential output (rewrite) %CPU	Hard disk Performance
F33	sequential input (per char) M/sec	Hard disk Performance
F34	sequential input (per char) %CPU	Hard disk Performance
F35	sequential input (block) M/sec	Hard disk Performance
F36	sequential input (block) %CPU	Hard disk Performance
F37	random Seeks /sec	Hard disk Performance
F38	random Seeks %CPU	Hard disk Performance

Table 3.2 Feature List

3.3.3 Feature Selection

The process of choosing a subset from the initial feature set based on the significance of the features is known as feature selection.

This work evaluates some of the potential hardware performance features read by the MacBook Air and identifies 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 three main categories. These are CPU-related values like the performance of floating-point arithmetic, memory-related features like the time taken to read memory, and hard disk-related features like the CPU usage when writing to disk. Also, the correlation between features and this value is used as a new feature. This research work focused on hardware features as potential ICMetrics features. Each feature was collected 1000 times since it is sufficient to determine the probability distributions.

Next, a feature set for a device was created that can be sensibly divided into individual sets. Each set includes features that have similar characteristics or are affected by the same changes in a device. These three feature sets consist of eight, six, and three features, respectively, in each set. Each feature set contains information to recognise low-level behaviours of the device. These features are dynamic in nature and will be more informative when building a model. These features are tied to each other to establish a unique relationship among features per device. In order to model this successfully, the first step was to perform very basic analysis to demonstrate the selection of the features in the table below (details in the following section).

Sr. No.	**Feature Name**	**Feature set Number**
F1	Maximum speed for copy function	1
F2	Maximum speed for scale function	1
F3	Maximum speed for add function	1
F4	Maximum speed for triad function	1
F5	Average duration for copy function	1
F6	Average duration for scale function	1
F7	Average duration for add function	1
F8	Average duration for triad function	1
F9	Sequential output (block)%CPU	2
F10	Sequential output (block)MB/sec	2
F11	Sequential output (rewrite)%CPU	2
F12	Sequential output (rewrite) MB/sec	2
F13	Sequential input (per char) %CPU	2
F14	Sequential input (per char) MB/sec	2
F15	Duration for add function	3
F16	Quickest duration for add function	3
F17	Longest duration for add function	3

Table 3.3 Lists New Dynamic Features selected to build an ICMetrics system

The following are the properties (related to raw features) that were explored to identify the devices uniquely:

A. Correlated features provide higher stability than individual features as they offer a predictability of range amongst them, this means that there is less intra-sample variance. These correlated raw feature sets contribute to building a robust system.
B. The lower intra-sample (samples of the same device) variance is needed means the more feature value can vary, the harder the value is to map and the less stable the value is when contributing to identify device uniquely.
C. Higher inter-sample (samples between two or more devices) variance contributes to the larger entropy of the system.

The high inter-sample variance and low intra-sample variance are examined to observe the potential overlap of the data between two or more devices [88].

3.3.4 Correlation of Features

This section explains the importance of correlated 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. 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 indicated by the correlation. Another significant aspect of correlated features is their ability to help distinguish devices. Singular features have a higher chance of having an overlap when the possible range for the feature is analysed across multiple devices. Singular features 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 [101]. For instance, Table 3.3 shows the correlation of the same feature combinations across different devices. The correlation of F1-F2 from device 0 is 0.964728227, and the correlation of device 1 is 0.738532807. This shows a great difference between Device 0 and Device 1. Although the coefficient of Device2 is 0.982909775, which is a small difference compared to Device0, it is still distinguishable. Device0 and Device1 are similar for F2-F4. Device2 shows an enormous disparity between Device0 and Device1. In this case, Device2 is distinguishable, but Device0 and Device1 are quite close.

Correlation of Features	**Device 0**	**Device 1**	**Device 2**
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

Table 3.4 Correlation of Features

3.4 Feature Modelling

The second contribution of this work is to analyse features in multidimensional space for device identification. To achieve this, the focus was on the following:

1 Feature Normalization

2 Feature Analysis

3 How to handle multimodal distribution

This section presents techniques that are used to model dynamic features that require statistical modelling to be used for unique device identification. Due to the fact that they are continuously changing, statistical features such as the mean and variance of a set of raw data are required in order to generate a stable unique identifier for identification, as these values are unlikely to change much with time. Since different approaches may be required for different feature sets, some may be normally distributed, some may conform to a multimodal distribution, etc.

The primary goal of an ICMetrics system is to generate a unique identifier for each device, which is derived from various device characteristics. This unique identifier can then be used to authenticate the device and detect changes in device operation. This unique identifier should have high intra-sample stability (on the same device) but low inter-sample stability (between different devices). In other words, a given device should always generate the same unique identifier, which should be unique to that device [93].

For dynamic features, it is likely that each time the feature is sampled, it will hold a different value. Instead, it is important to take numerous estimations of the feature, quantize the deliberate values into discrete values, and produce a frequency distribution for that feature [102]. One possible approach to extracting a feature value from a feature distribution is to map every value to a single value that is representative of the distribution, for example, the median of the set. This number would then be the feature value for that set. Since there are 17 unique features as of now, the entropy is 2^{17}.

3.4.1 Normalisation

In the calibration stage, features that are described in Table 3.2 have been utilized. The data is then sent through a quantization and normalisation process. If the data measured by the device is not-normally distributed, it may be necessary to normalise the data so that it can be used for device identification. One approach that can be used to achieve this is to map the values from the raw distribution to a set of values in a normal distribution [103]. Finally, a multidimensional normalisation map is produced based on normalised data. In the operation phase, measured data is mapped to a multidimensional normalisation map to form a unique identifier [104] [105].

3.5 Feature Analysis

To determine the best technique to describe the distinct behaviour of the features (per device) and to identify subtle differences between the datasets acquired from each device, simple statistical analysis to find the correlation, mean, standard deviation, and variance were undertaken. These are the first derived entities from the raw data. The probability density function (PDF) is applied to the data as part of the detailed analysis.

For detailed analysis, a probability distribution graph for each feature was generated to understand how the data is distributed in multidimensional space. The importance of visualising the data in multidimensional space helps differentiate between the overlapping data from different devices. This will infer the data to be unimodal, bimodal, or multimodal in nature.

The data should be analysed in a multidimensional space because the selected features are multimodal in nature. The distribution of the data is shown in the following section, along with a thorough explanation of how to deal with the data's multimodality for accurate classification. Addressing this multimodality will increase the probability of the devices being recognised correctly.

3.5.1 Multimodal Distributions

The previous section discussed the importance of using highly dynamic and multimodal aspects. Figure 3.3 below shows the distribution of one of the features across all devices.

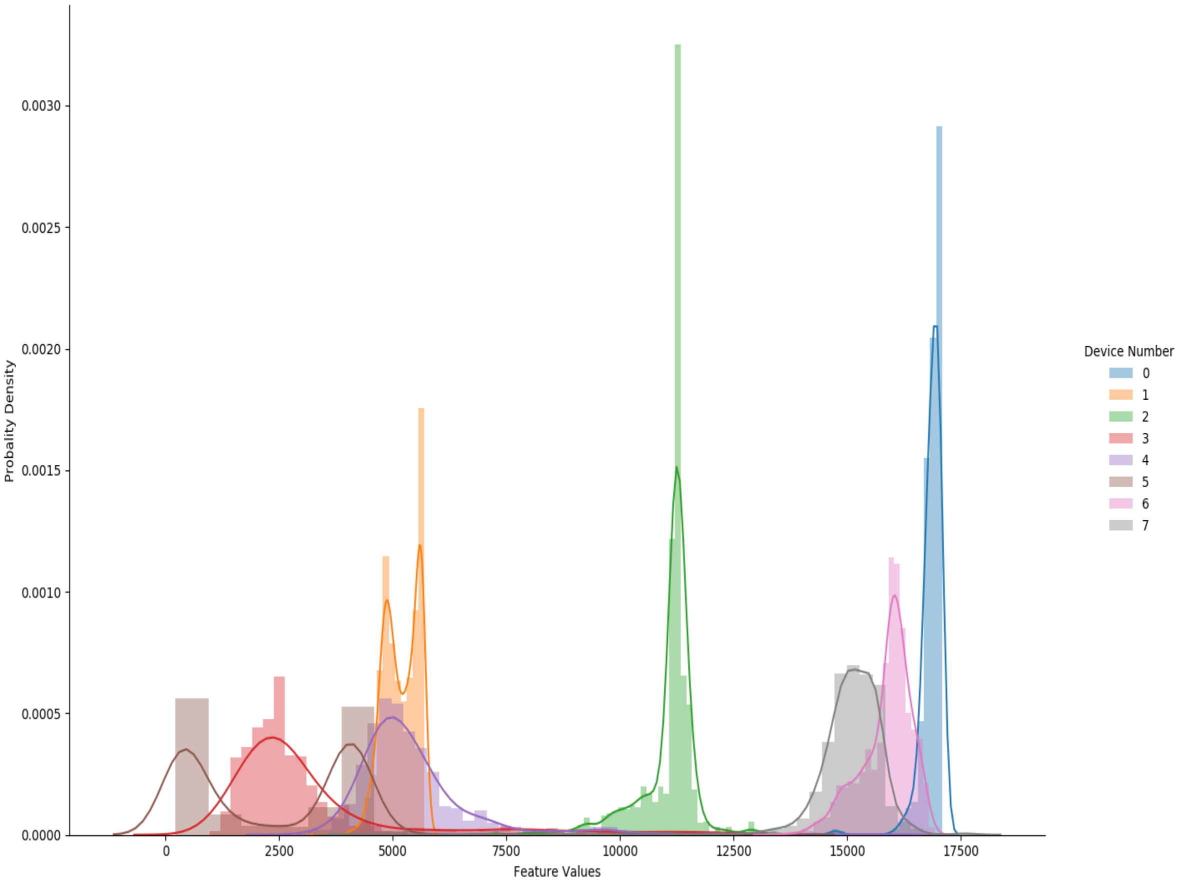


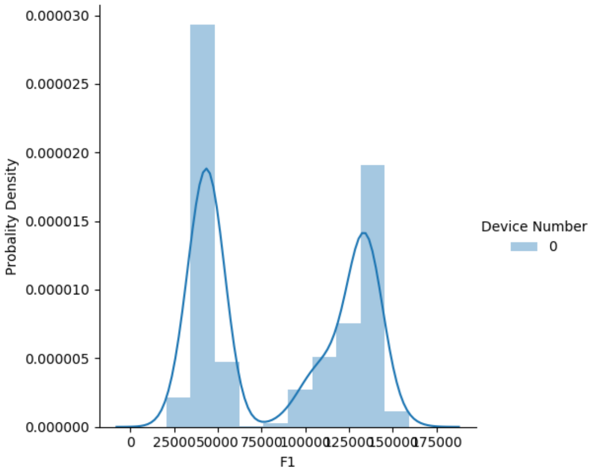
Figure 3.3 Probability density graph of feature (F1) for all devices

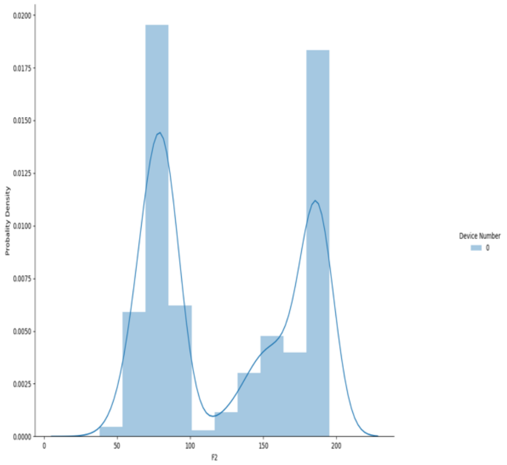
To analyse the data, the probability density function was applied to each selected feature, and the distribution of each feature across all devices was visualized. The device number in the graph denotes the total number of devices (8) in this case. The distribution of data across each device is presented using various colours. Understanding the distribution of F1 across all devices requires the representation of F1 on a single graph. Each device's Feature F1 is multimodal, and all of the data for all of the devices completely overlaps. Therefore, for efficient data modelling, it is crucial to take advantage of the connections between the features for effective modelling of the data.

To address the multimodal feature behaviour, the following steps are taken.

1. To determine the total number of modes for every feature on each device.

2. The peak-trough approach is used to separate these multimodal features into the number of modes they each represent. Here, the number of modes present in each feature will provide a corresponding number of thresholds. Modal boundaries will be produced by these thresholds. The modes for a device's F1 and F2 features are depicted in the fig below.





3. Establish a connection between the modes of these features. For instance, in a database with two features, F1 and F2, if F1 is bimodal (two modes are represented by F1-M1, F1-M2) and F2 is also bimodal (two modes are represented by F2-M1, F2-M2), as seen in the image above. Combining the modes for each feature can demonstrate their link. So, for Device 0, we obtain the following four modal combinations:

   I. [F1-M1, F2-M1]
   II. [F1-M1, F2-M2]
   III. [F1-M2, F2-M1]
   IV. [F1-M2, F2-M2]

4. The data in each of these combinations is unimodal in nature and exclusively represents device 0. The number of combinations produced by this process—four in the example above—depends on how many modes each feature has. It is carried out for each device with ‘n’ features. These modal mixtures are often known as ‘Converted Gaussians.’
5. The samples should then be sorted according to thresholds for each modal feature combination for that device. There might be a few empty combinations (i.e., combinations with no samples), which can be removed as soon as they are discovered.
6. For every device, these created modal combinations are also known as Converted Gaussians. Each device is represented by many converted Gaussians.

With the use of thresholds produced by the peak-trough method, the steps above address the multimodal features and transform them into Gaussian distributions. Next our focus is to take care of the second issue, which is feature overlap, by addressing the multimodality of the characteristics. By constructing the combination of modes that were described in step 3, we can demonstrate the relationship between the modes of features (these are specific to each device), which helps to distinguish the overlap of data between devices for each feature. For instance, consider the F2 for each device if the F1 values for devices 1 and 2 completely overlap. This shows that the overlapped values in F1 might not be related to the same values of F2 by taking into account the relationship between F1 and F2 for each device. Utilising the interdependence of each feature on each device, our aim is to take advantage of the unique hardware level operation of each device. The distribution of feature F2 across all devices is depicted in the figure 3.4 below and the graph shows that there are overlapped values for the same feature for different devices. There was a need to determine how each feature on each device interacts with the others. As visualised in the graph 3.4, devices 1 and 2 are bimodal and multi modal nature.

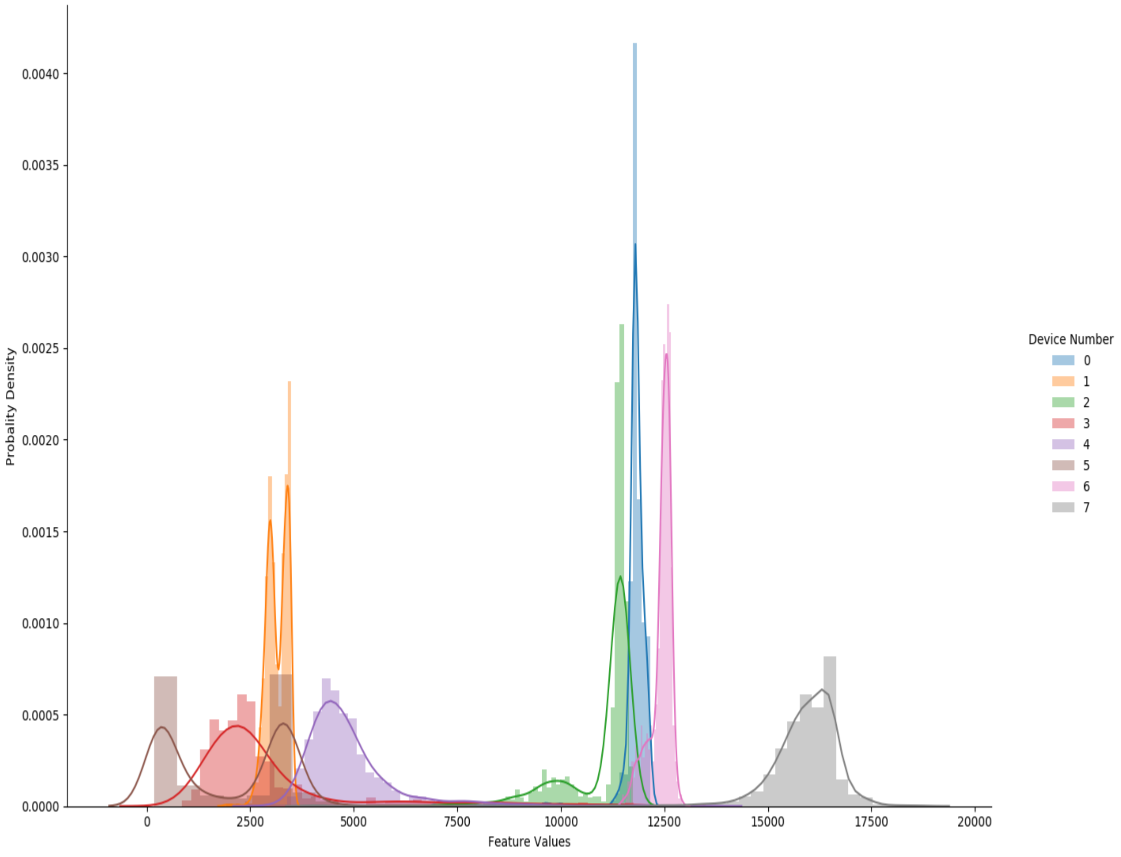


Figure 3.4 Probability density graph of feature (F2) for all devices.

Figures 3.5, 3.6, 3.7, 3.8, 3.9, and 3.10 show the third, fourth, fifth, sixth, and eighth feature probability density graphs for all devices. From the graph, we can observe how the same feature is acting differently across all devices. The relationship between the features must therefore be taken advantage of for effective data modelling. Graph 3.5 highlights devices 2 and 6 data are overlapping hence the focus was on relationship between the devices.

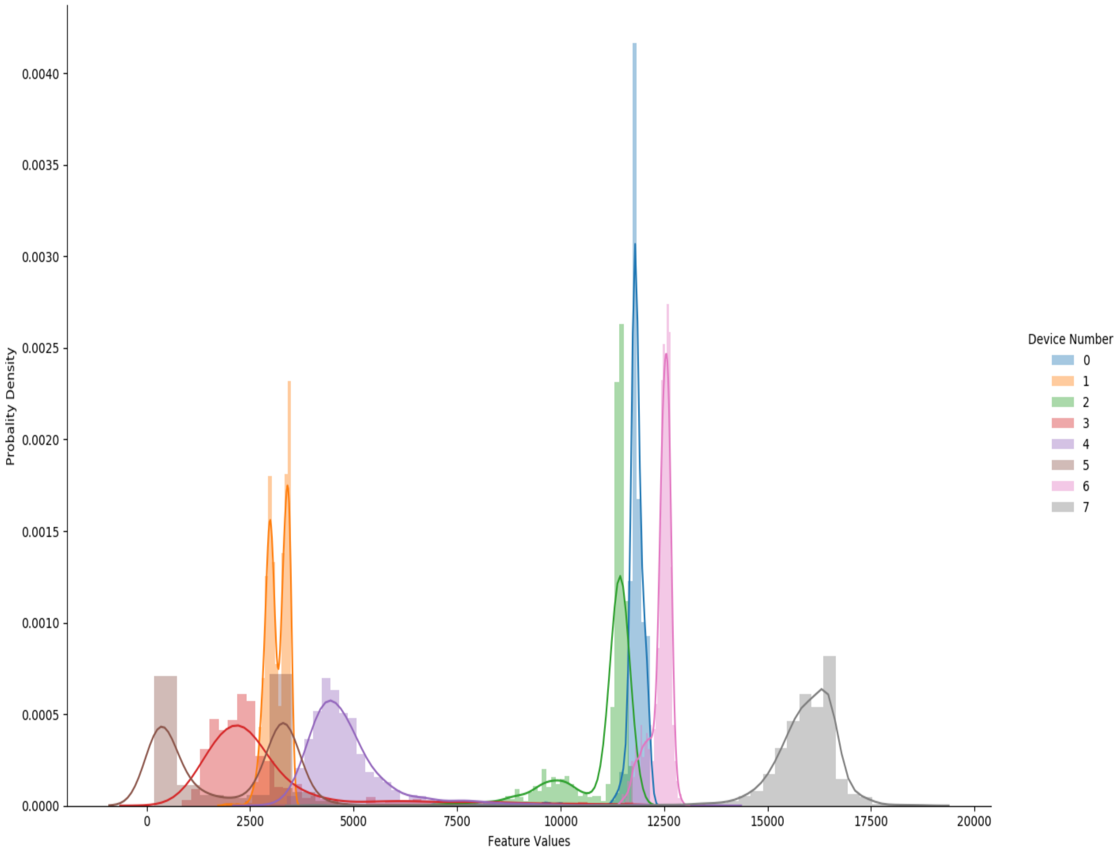


Figure 3.5 Probability density graph of feature (F2) for all devices.

In the graph 3.6, devices 4 is multimodal, 1 and 2 is bimodal in nature.

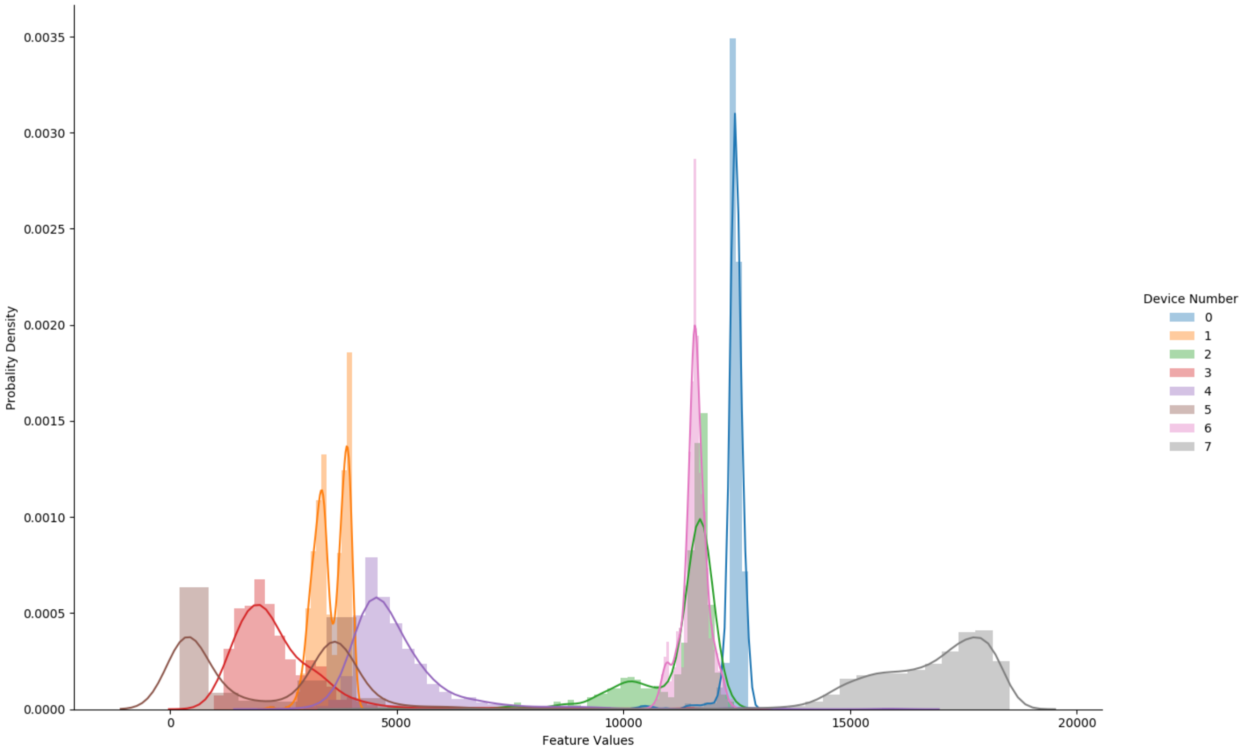


Figure 3.6 Probability density graph of feature (F3) for all devices.

In the below graph, devices 7 is multimodal and rest are all unimodal

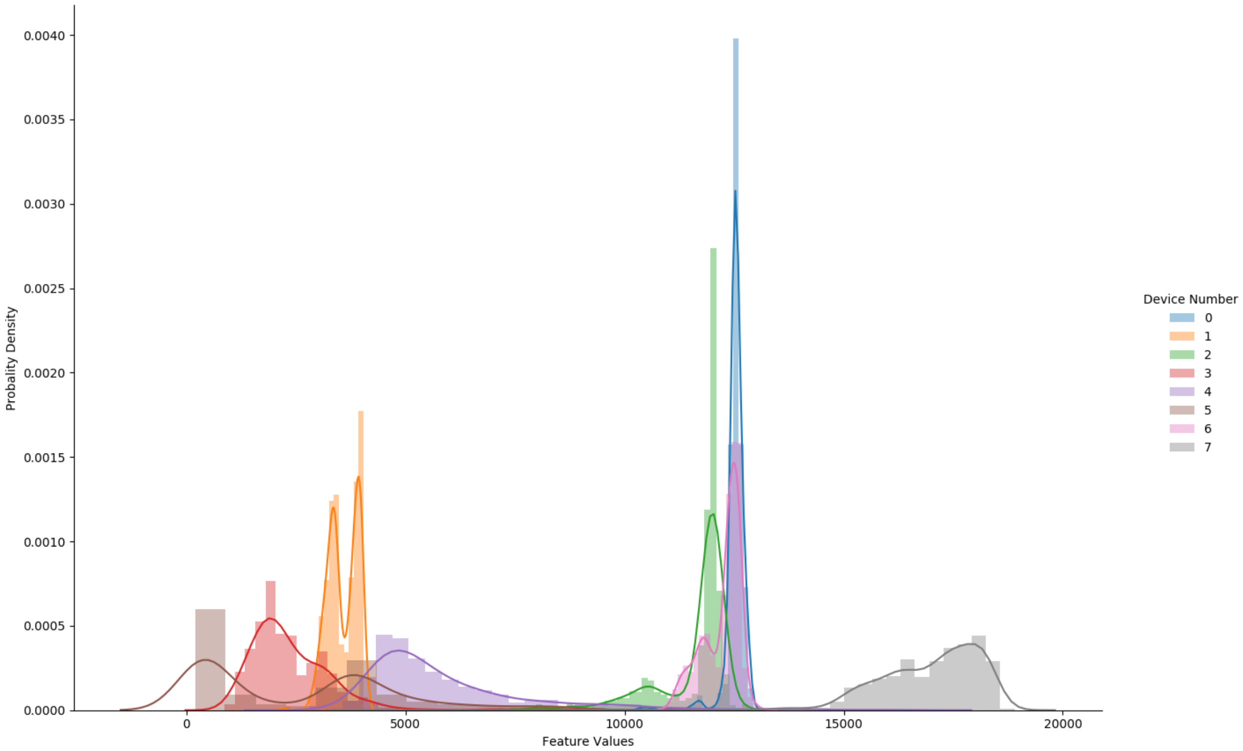


Figure 3.7 Probability density graph of feature (F4) for all devices.

As shown in the graph 3.8, devices 4, 6 and 7 are multi modal in nature.

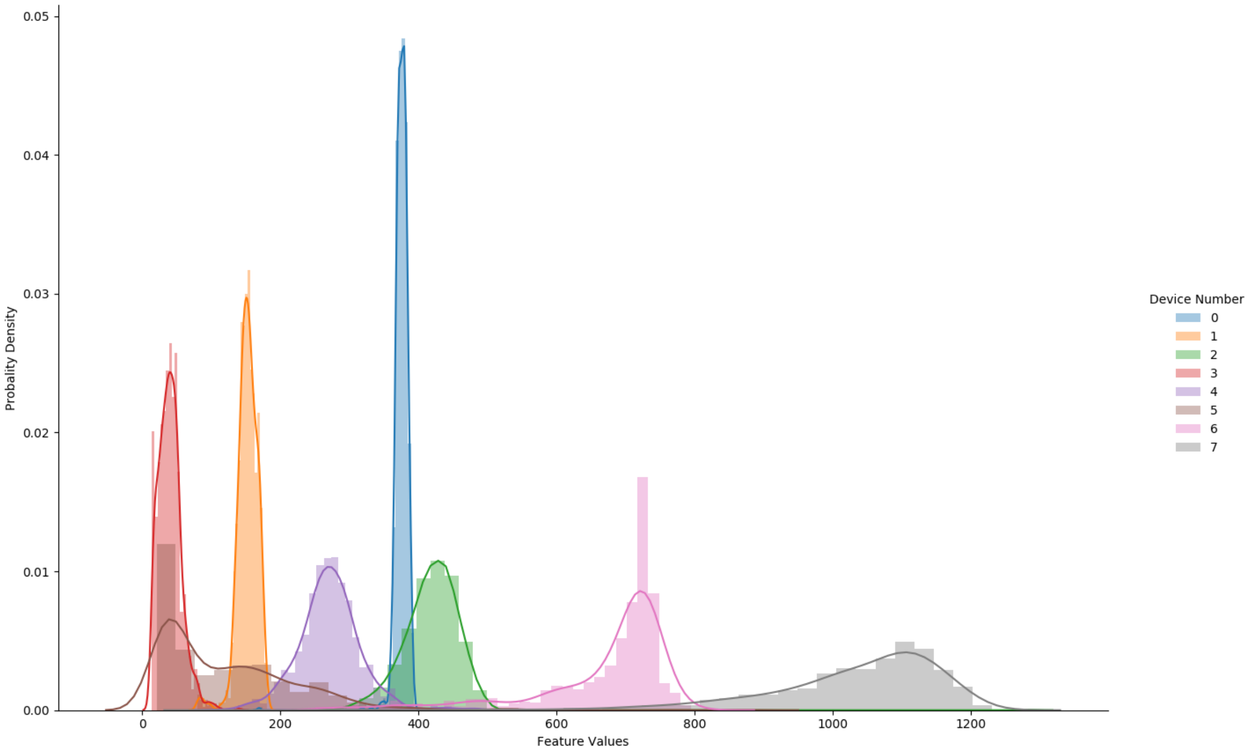


Figure 3.8 Probability density graph of feature (F6) for all devices.

As shown in the graph 3.9, devices 0, 2, 6 and 7 overlap significantly

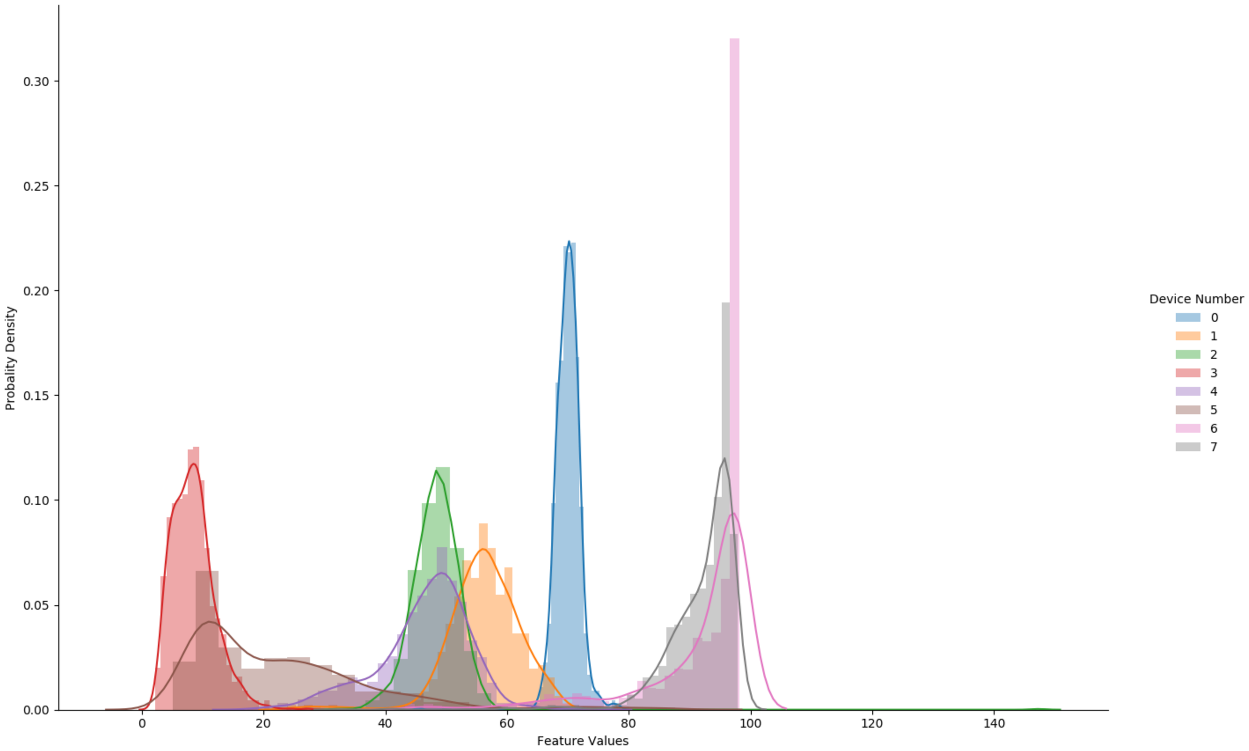


Figure 3.9 Probability density graph of feature (F7) for all devices.

Graph 3.10 represents all multi modal devices.

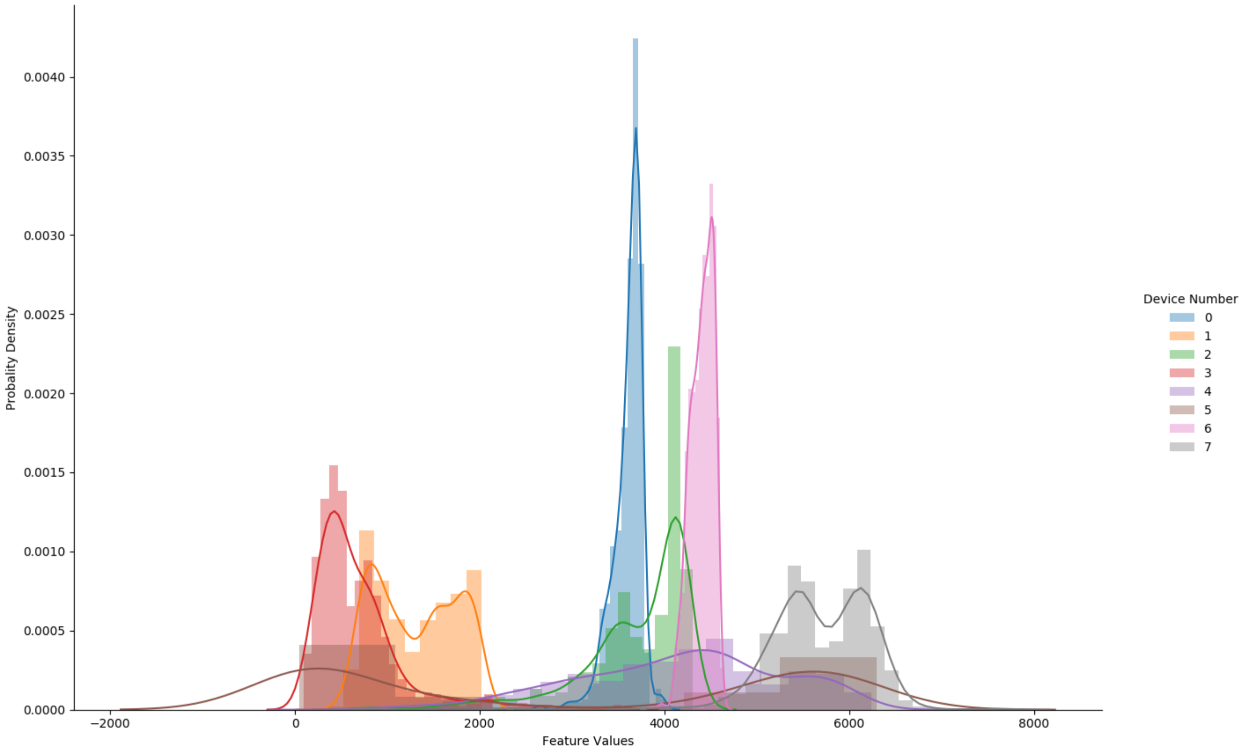


Figure 3.10 Probability density graph of feature (F8) for all devices.

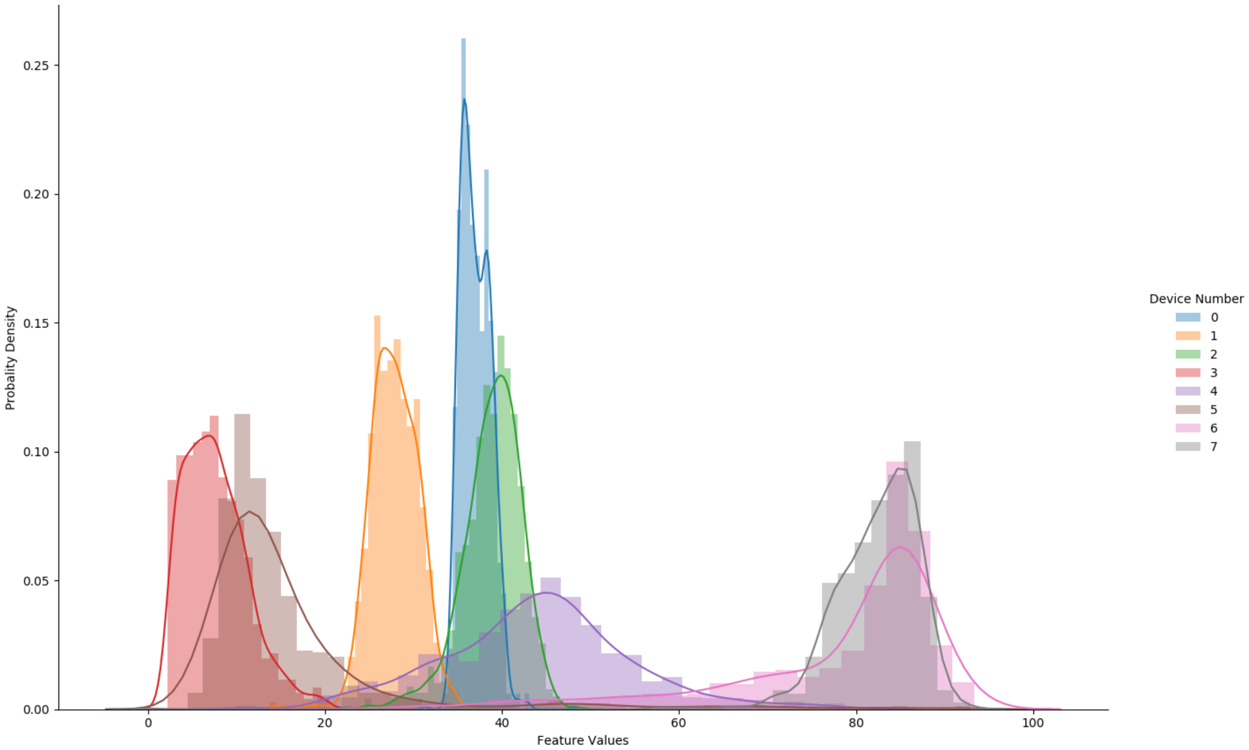


Figure 3.11 Probability density graph of feature (F9) for all devices

As shows in fig 3.2 - 3.10, if the feature data is distributed across two curves, it is bimodal and if its more than two curves, then its multi-modal distribution. To explain further, one feature on 2 different devices can have different distribution. For example, one target feature (CPU usage) on two identical (laptops) can have bimodal or multi modal distribution depending on application (Browser) running at the time of data collection. After looking at the frequency distribution of the feature, it became apparent that these are multimodal features, and hence the multimodal approach was decided. Figure 3.11 below depicts the multimodal feature process in detail.

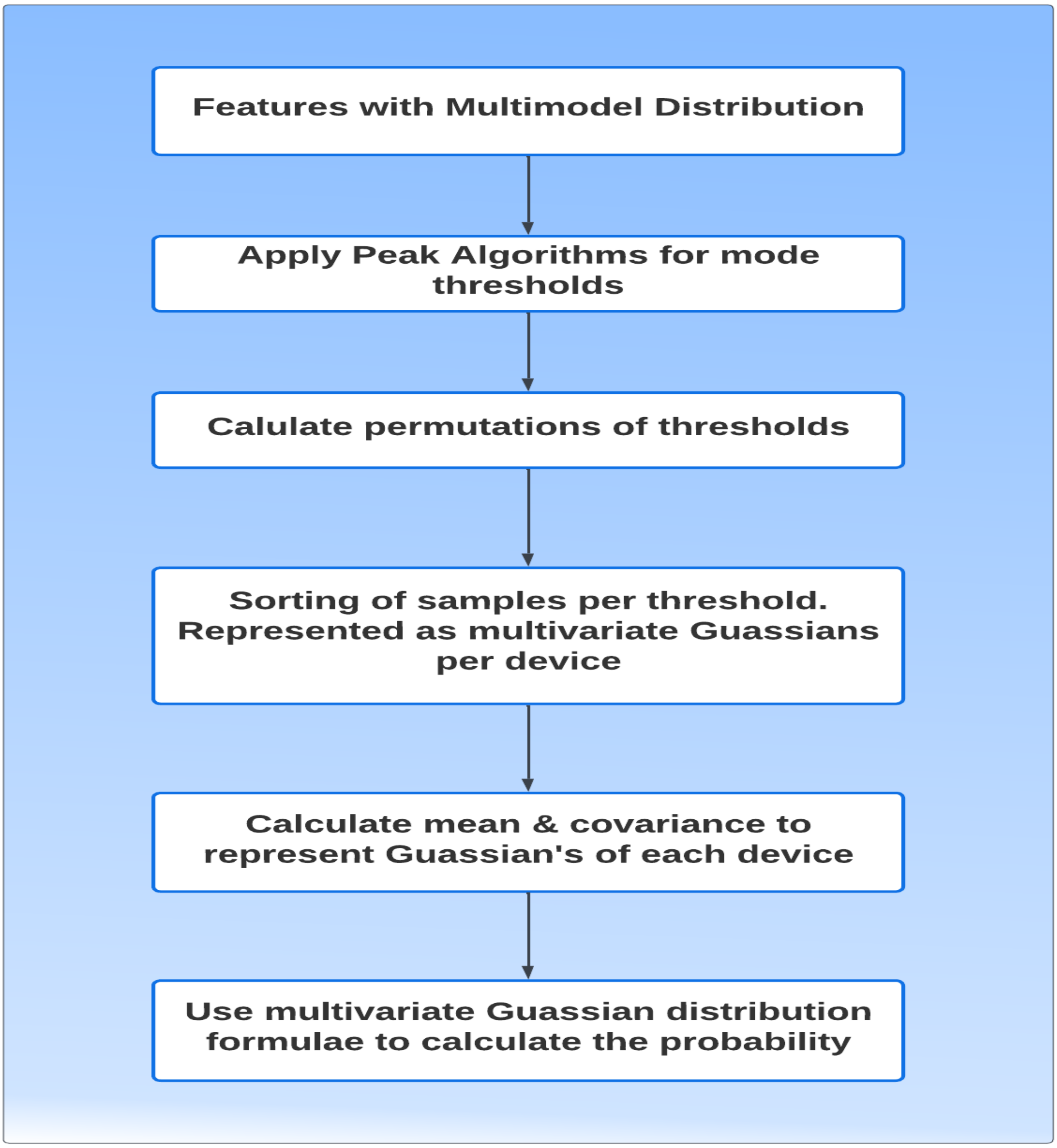

Features with Multimodel Distribution
Apply Peak Algorithms for mode thresholds
Calulate permutations of thresholds
Sorting of samples per threshold. Represented as multivariate Guassians per device
Calculate mean & covariance to represent Guassian's of each device
Use multivariate Guassian distribution formulae to calculate the probability

Figure 3.12 Multimodal Process

Multimodal process is defined below.

After feature analysis, it was concluded that a multimodal set of features does not generate a unique identifier for device identification. To address this challenge, there was a need to determine how many modes are available for each feature on each device. The distributions were subdivided into a series of components, each of which is approximately normal, and each mode on the original distribution become the mode of its own normal distribution. A simple approach to this problem is to apply a peak-trough detection algorithm to the histogram of each feature where the troughs split the multimodal distribution into separate normal distributions (converting this to unimodal) with the peaks forming the modes, to decrease the overlapping of data amongst devices.

The peak-trough algorithms take in the histogram data and divide the modes based on the troughs of a probability distribution graph [106]. This is used to create modes to associate samples with their respective permutations. This in turn will show the relationship between the features for each modal combination. Hence, examining these features in relation to each other creates a unique device print where these combinations are generated [108].

The above-mentioned procedures deal with multimodal features and transform them into Gaussian distributions using thresholds produced by the peak-trough technique. By addressing the multimodality of the characteristics, which is feature overlap, the second issue was solved. Establishing a relationship between the modes allows for the differentiation of data overlap between devices for each feature, so combining the modes demonstrated the relationship between the modes of features (these are unique per device) [107].

In this section, a thorough explanation of the handling of multimodal data and its significance is provided. The next section outlines the operation phase of the system.

3.6 Operation Phase

The operation phase starts each time an encryption key is required. For this, all features in the three-feature set (as described in Table 3.2) are dynamic in nature, which requires statistical or mathematical modelling for a unique identifier. In other words, a given device should always have a unique identifier, which is the primary goal of the ICMetrics system [101] [107].

The challenge with the ICMetrics technique is to generate a unique identifier that is used to authenticate and is formed of several device characteristics. Having just one characteristic change significantly may change the unique identifier, although the variation may still be consistent with the operation of the device. Subsequently, the device will fail to authenticate because basic approaches to combining feature values, like simple concatenation, don't allow for device characteristics to change.

One possible solution to this problem is to implement a secret sharing algorithm to combine feature values, which allows the unique identifier to be recovered even if a limited number of the device characteristics have failed.

3.6.1 Secret Sharing Scheme

In the case of Shamir's Secret Sharing algorithm [111], this is done by defining a polynomial where the y-axis intercept defines the unique identifier, and upon which all of the devices' feature values (at the time of calibration) lie. Since a polynomial can be defined if a given number of points are known (i.e., a straight line with 2 points, a parabola with 3 points, a cubic polynomial with 4 points, etc.), the y-axis intercept and therefore the unique identifier can be recovered even if a limited number of the characteristics fail. For example, a parabola with 5 total points would allow up to 2 points to be invalid, and the unique identifier can still be calculated correctly using the other 3 valid points available.

The next step is secret sharing in the ICMetrics key generation process. A common way is to generate a unique identifier to pass in as the X values and then calculate the associated Y values to create the points needed to reconstruct the unique identifier, or device identity. When it is required to reconstruct the secret, Y values can be fetched from where it was stored and ICMetrics values read to get the X values. Once the X and Y pairs are in place, reconstructing the secrets using interpolation is the next step. This way, correctly reconstructing the secrets is possible when enough ICMetrics values are valid.

The secrets can be split into several shares, with a threshold needed before the secret can be reconstructed. Generating a secret using some form of cryptographically secure RNG and then using ICMetrics to represent the points on the polynomial allows the secret to be reconstructed with valid ICMetrics, and also allow the key to be revoked if it gets compromised. It also enables us to set a level of tolerance in the system for difficult-to-map features, ensuring the ICMetrics system's reliability.

The advantage of this process is that the ICMetrics are not stored on the system, and the only values that are stored are one-half of the co-ordinates that are necessary to generate the polynomial that produces the ICMetrics. The halves that are stored on the system cannot be used to find out the polynomial that was used to generate them. Interpolation cannot be employed without the associated x value for each stored Y value, which means an attacker cannot derive the device identifier (unique identifier) from the stored data and has no way of knowing where on the X axis each point sits. Additionally, a new ICMetrics can be generated any time the system needs to be changed or reset by repeating the process of taking a new arbitrary basis value and passing in feature values to generate new Y values similar to existing Y values, or by dynamically adjusting Y values by using an offset.

3.7 Experimental Methodology

This section describes the dataset, pre-processing and standard classifiers used for this analysis.

3.7.1 Feature Dataset

MacBook Air and MacBook Pro were chosen since these devices allow access to the low-level hardware as well as the software contained on each device, allowing us to find features that can provide an adequate dynamic range and variance. Despite having MacBook models from 2013 to 2020 (section 3.3.1.1), the majority of them had an i5 processor, which allowed us to have uniformity in the features collected in section 3.3.2.

On contrary, Windows hardware is created by numerous vendors and makes it difficult to separate the devices. Apple's hardware tends to be more homogenous. Based on these traits, feature values were segmented. Memory performance, Hard disk performance, and CPU floating point make up the three categories for the dataset information that was collected from the devices. The analysis of each feature included collecting data from multiple Mac devices to generate as diverse a range for feature values as possible. Features generally form unusual distributions that cause a pattern recognition problem and require more complex pattern recognition methods. The patterns that were discovered were used to differentiate the feature values into multifaceted clusters that can dictate how distinguishable the devices are using that particular feature. If the feature provided enough inter-sample variance, it was selected for further examination. Furthermore, a feature that did not provide enough inter-sample variance could still be acceptable if a correlation could be found with another feature that produced suitably complex and distinct clusters when plotted in an n-dimensional space. Distinct clusters are easier to discover when multiple features are used in combinations, thus the requirement for multi-dimensional plotting. The goal of the experiment is to evaluate the hardware features for device identification. The extracted features are used during testing to determine the proposed system's accuracy rate and for benchmarking the results. Also performance metrics are computed to evaluate the effectiveness of the proposed ICMetrics security system, including the accuracy rate, precision, recall and F Measure.

3.7.2 Pre-processing /Normalizing

This stage is necessary to produce training and testing dataset from the device's extracted features. The mean and variance of each individual feature value are modified as part of the feature normalisation procedure in order to compare how much each characteristic contributes to the final match score. During this step, a min-max normalisation approach was applied [112].

3.7.3 Classifiers

This section explains the common linear and non-linear classification methods used on data gathered from devices (accessible with sklearn). The classification outcomes obtained using these methods are compared with the multimodality model we created. Standard classifiers are used to verify and compare how well each classifier performs on the data because the data is multi-modal in nature. The results are compared using the following classifiers:

The correct classification of the device is defined using our proposed model's multivariate Gaussian distribution [113] in order to evaluate the model's effectiveness as a classifier. In this proposed methodology, other three standard classifiers have been used for benchmarking, namely: logistic regression (LR) [115], linear discriminant analysis (LDA) [114] and support vector machines (SVM)[115]. These classifiers are used to compare the predictions. To accurately classify the test data based on three feature sets, the above-mentioned classifiers were applied to various device datasets. The performance of these classifiers is evaluated on the basis of accuracy, precision, recall, and F measure. The results section contains the analysed results based on these. For code implementation, Python has been used. In this section, we explain the various classification techniques.

3.7.3.1 Proposed Multivariate Gaussian Distribution (MVGD)

The statistical analysis needed for the production of ICMetrics numbers used for device identification is provided below. Because the feature readings are a discrete random variable, the process requires a probability density (x) function to calculate the precise value from the feature reading.

By taking multimodality into consideration, our goal was to model the data from each device as multiple multivariate Gaussian distributions. Create modal combinations to divide the data from each device into several converted Gaussians in order to address multimodality; these conversions are made with the use of modal threshold, as discussed in section 3.5.1. These transformed Gaussians can be seen as numerous multivariate distributions per device, similar to how it is shown in GMM. The mean and covariance matrices of each of these multivariate Gaussians serve as their representation. Next, use equation (1) to calculate the probability of each test sample compared to each distribution and keep a record of the probability that each devices produces. Last step is to repeat for each device, finding the maximum probability generated, which is our classifier's prediction data.

As per our knowledge in Gaussian Mixture Models (GMM) [161], the data is represented as ‘n’ mixture models; similarly, our classifier represents it as ‘n’ multivariate Gaussians per distribution. Assume that each sample has one of these multivariate Gaussian distributions.

The d-dimensional vector x is multivariate Gaussian in the event that it has a likelihood thickness capacity for the accompanying structure:

$$p(x; \mu, \sum) = \frac{1}{(2\pi)^{d/2}|\sum|^{1/2}} exp\left(\frac{1}{2}(x-\mu)^{T}\sum^{-1}(x-\mu)\right) \qquad (1)$$

if $'\mu'$ represents mean, ‘x’ represents particular sample reading from feature data and the covariance matrix ‘Σ’.
The mean vector μ is the assumption for x:

$$\mu = E[x] \qquad (2)$$

The covariance Σ is the assumption for the deviation of x from the mean:

$$\Sigma = E[(x-\mu)(x-\mu)^{T}] \qquad (3)$$

3.7.3.2 Linear Regression (LR)

Logistic Regression is a predictive analysis technique that uses linear classification when the dependent or target variable is dichotomous [117]. Since this is a standard classifier, our intent is to compare this with our classifier (multi-modal distribution) and check the performance of the system. Additionally, it offers multinomial categorical prediction, which in our instance classifies the data into the device number, or the target variable.

3.7.3.3 Linear Discriminant Analysis (LDA)

The LDA projects the data in higher dimensions onto a lower-dimension space (reducing dimensions). In LDA, the goal is to identify a linear combination of features that can distinguish between two or more object classes and assess the system's effectiveness with multimodal feature combination [118].

3.7.3.4 Support Vector Machine (SVM))

The SVM model has the capability to handle both regression and classification problems. Here the data is plotted and viewed in n-dimensional space, where n depicts the number of features. This is a non-linear classification technique that can separate the data into different classes via a decision plane. Hence the data, which seem to be linearly inseparable, are subjected to intricate mathematical functions called kernels, which effectively separate the data belonging to their respective classes [119]. The complexity of the model ensures higher accuracy and presents fewer possibilities of over-fitting [117].

The exploratory outcome depicts which classifier is best between them. The contribution of this research is to analyse multimodal features in multi-dimensional space and identify the device uniquely, and for benchmarking, use the standard classifier for comparison.

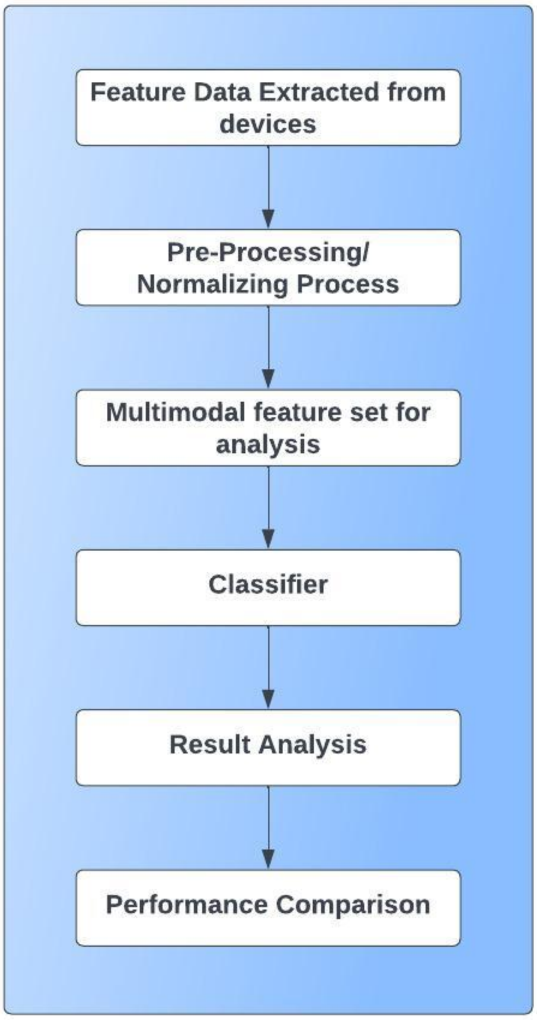

Feature Data Extracted from devices
Pre-Processing/ Normalizing Process
Multimodal feature set for analysis
Classifier
Result Analysis
Performance Comparison

Figure 3.13 Methodology Process

3.7.4 Multimodal Classifier Algorithms

The following steps summarise the proposed multi-modal classifier algorithm for the ICMetrics security system:

1. Read the data from all the devices.
2. Select features based on criteria for good features.
3. Normalize the data and then split it using k-fold (k = 10) cross-validation.
4. Divide features into three groups that share the same characteristics, and then determine a probability distribution for each feature.
5. Check if the distribution is unimodal, bimodal, or multimodal.
6. Apply a peak-trough identification algorithm to the distribution if the distribution is

multimodal. Here, the peak-troughs split the multimodal distribution into separate Gaussian distributions, with the peaks forming the modes.

7. Create modal combinations to split the data per device into many ‘Converted Gaussians’. These are transformed with the aid of a modal threshold that was built using peak-trough data.
8. The mean and covariance matrices of each of these multivariate Gaussian distributions are used to represent them.
9. Next, calculate the probability of each test sample associated with that mode using a multivariate normal probability density function.
10. Take samples from each device and compute the mean and covariance of the modes within the distribution of the current devices to determine the probability.
11. For example, if the device has a bimodal distribution, that means there are two modes, and each mode has its own mean and covariance.
12. For this, first determine which mode the current sample falls into, then calculate the probability of the sample, and repeat the same process for other modes.
13. Next, take the same sample set from another device and determine which mode of the first device that samples belong to, and then calculate the probability of the sample. If the probability from the second device is low as compared to the first device, that means the first device is correctly identified based on probability, and you repeat the same process for the other ‘n’ devices and then save the probability produced for each device.
14. Repeat this process for all devices, and the maximum probability of those results will serve as our classifier's prediction data.
15. Compare probability results from benchmarking classifiers.
16. Generate a confusion matrix and classification reports.

3.7.5 Performance Measures

Performance, accuracy and usability are three metrics that can be used to evaluate system features. Metrics were employed by the researchers to evaluate the system's performance. The system, which is based on the dataset collected from the general computing devices, is evaluated using a variety of performance metrics.

Four different alarm types - True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN) - must be calculated in order to quantify and assess the proposed system's performance. The measurements will be computed using the formula below [120].

TP- if we can prove that a unique identifier belongs to a specific device
TN- if we can prove that a unique identifier does not belong to a specific device
FP- if unique identifier identifies a device incorrectly
FN- if unique identifier incorrectly concludes that it’s not the specific device, however, in reality it is the device in question.

Classification Rate or Accuracy is given by the relation

$$\textbf{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \qquad (4)$$

Recall: Recall can be defined as the ability of the classifier to find all positive instances. It is defined as the ratio of true positives to the sum of true positives and false negatives.

$$\text{Recall} = \frac{TP}{TP+FN} \qquad (5)$$

Precision: Precision can be defined as the ability of the classifier not to label as positive a sample that is negative. It is defined as the ratio of true positives to the sum of true positives and false positives.

$$\text{Precision} = \frac{TP}{TP+FP} \qquad (6)$$

F-measure: F-measure can be defined as the harmonic mean of precision and recall. The F-measure corresponding to every class will tell you the accuracy of the classifier in classifying the data points in that particular class compared to all other classes.

$$\text{F-measure} = \frac{2*Recall*Precision}{Recall+Precision} \qquad (7)$$

3.8 Experimental Results

The proposed system's role in creating and improving the security methods for device identification is detailed in this section. To assess the system's performance, the proposed system is put to the test using features derived from the device. Performance measures, such as the confusion matrix, and accuracy rate are determined in this section.

As part of our experiment, multi-modal classifier was built since it offers higher level of security and is our novelty. To test the effectiveness of this classifier, benchmarking comparison was made with standard classifiers. To achieve this, the following experiments were conducted to establish that the above data can be measured and employed as feature values for the identification of individual device characteristics. Data gathering and statistical analysis have been automated using code written in Python.

The section provides a thorough analysis of the experimental findings associated with the proposed model, MVGD. The features discussed in section 3.3.2.5 are the subjects of the studies. This information is gathered from the hardware components of the MacBook Air and MacBook Pro (memory, CPU and hard drive). After analysis, this data provides us with a unique identifier for device identification. Eight devices with up-to-date software were used. Data gathered from the MacBook Air and Pro, Python code, and Microsoft Excel were used in this study. For this experiment, these devices are used, and for our analysis, each device includes a thousand samples. One of our experiments was to test the following features as shown in Table3.5 and 3.6 on multiple devices of the same type and conclude that a particular feature could be unimodal, bimodal, or multimodal on different devices. This is depicted in the tables below.

Device	Feature Name	Feature Mode Value	Mode Category
0	Mflops	[1729.6197731, 2029.6555919, 2254.682456]	Multimodal
	Random Seeks/sec	[53887.0, 145687.0, 237487.0]	Multimodal
	Random Seek%cpu	[70.02503879999999, 286.95473]	Bimodal
1	Mflops	[899.5161528, 948.082779]	Bimodal
	Random Seeks/sec	[51306.0]	Unimodal
	Random Seek%cpu	[74.953014, 90.994744]	Bimodal
2	Mflops	[1442.7346728, 1782.421908]	Bimodal
	Random Seeks/sec	[81121.4, 120248.20000000001, 159375.0]	Multimodal
	Random Seek%cpu	[194.854033]	Unimodal
3	Mflops	[1797.6023046999999, 2289.139028]	Bimodal
	Random Seeks/sec	[45609.2, 130894.0]	Bimodal
	Random Seek%cpu	[79.7744538, 210.394809]	Bimodal
4	Mflops	[1561.2914516, 2273.87369]	Bimodal
	Random Seeks/sec	[92743.2, 148842.0]	Bimodal
	Random Seek%cpu	[153.83213899999998, 218.29451]	Bimodal
5	Mflops	[814.2273418, 1427.335394]	Bimodal
	Random Seeks/sec	[3466.5, 6146.0]	Bimodal
	Random Seek%cpu	[4.127735400000001, 6.355079]	Bimodal
6	Mflops	[1525.785184, 1876.558732]	Bimodal
	Random Seeks/sec	[146585.0, 202439.0]	Bimodal
	Random Seek%cpu	[127.4858734, 213.639322]	Bimodal

7	Mflops	[683.2273122, 1980.172609]	Bimodal
	Random Seeks/sec	[295792.0]	Unimodal
	Random Seek%cpu	[103.0319376, 230.78943]	Bimodal

Table 3.5 Same three features with different number of modes in different devices

Device	Feature Name	Feature Mode Value	Mode Category
0	Sequential output (block)%CPU	[333.8403578, 393.821725]	Bimodal
	Sequential output (block)MB/sec	[77.671227]	Unimodal
	Sequential output (rewrite)%CPU	[3027.3687172, 3976.143141]	Bimodal
	Sequential output (rewrite) MB/sec	[33.659796, 42.961692]	Bimodal
	Sequential input (perchar)%CPU	[17.380394600000002, 20.018369]	Bimodal
	Sequential input (perchar)MB/sec	[97.57372260000001, 99.955529]	Bimodal
1	Sequential output (block)%CPU	[99.909929, 181.144981]	Bimodal
	Sequential output (block)MB/sec	[33.2123522, 70.531697]	Bimodal
	Sequential output (rewrite)%CPU	[1189.5236169999998, 2020.446923]	Bimodal
	Sequential output (rewrite) MB/sec	[35.187215]	Unimodal
	Sequential input (perchar)%CPU	[4.6886006, 8.756403]	Bimodal
	Sequential input (perchar)MB/sec	[54.3397778, 99.963561]	Bimodal
2	Sequential output (block)%CPU	[1283.202874]	Unimodal
	Sequential output (block)MB/sec	[147.342487]	Unimodal
	Sequential output (rewrite)%CPU	[1382.0247729999999, 4306.075873]	Bimodal
	Sequential output (rewrite) MB/sec	[48.88907]	Unimodal
	Sequential input (perchar)%CPU	[11.518076, 17.625869]	Bimodal
	Sequential input (perchar)MB/sec	[99.939266]	Unimodal
3	Sequential output (block)%CPU	[102.255239]	Unimodal
	Sequential output (block)MB/sec	[19.085119]	Unimodal
	Sequential output (rewrite)%CPU	[3538.445207]	Unimodal
	Sequential output (rewrite) MB/sec	[19.918016]	Unimodal
	Sequential input (perchar)%CPU	[1.256827, 2.659283, 3.711125]	Multimodal
	Sequential input (perchar)MB/sec	[9.952163500000001, 20.4312095, 28.290494]	Multimodal
4	Sequential output (block)%CPU	[624.562806]	Unimodal
	Sequential output (block)MB/sec	[74.72207]	Unimodal
	Sequential output (rewrite)%CPU	[6214.654154]	Unimodal
	Sequential output (rewrite) MB/sec	[78.222535]	Unimodal
	Sequential input (perchar)%CPU	[17.941423]	Unimodal
	Sequential input (perchar)MB/sec	[81.577747]	Unimodal
5	Sequential output (block)%CPU	[620.0750369, 876.800729]	Unimodal
	Sequential output (block)MB/sec	[54.5787354, 87.595625]	Bimodal
	Sequential output (rewrite)%CPU	[2547.5043626, 6294.058409]	Bimodal
	Sequential output (rewrite) MB/sec	[31.012234399999997, 92.964095]	Bimodal
	Sequential input (perchar)%CPU	[10.522833000000002, 20.989768]	Bimodal
	Sequential input (perchar)MB/sec	[100.005217]	Unimodal
6	Sequential output (block)%CPU	[362.5170858, 810.576401]	Bimodal
	Sequential output (block)MB/sec	[54.2516332, 98.151602]	Bimodal
	Sequential output (rewrite)%CPU	[4079.5603254, 4636.928499]	Bimodal
	Sequential output (rewrite) MB/sec	[93.369356]	Unimodal
	Sequential input (perchar)%CPU	[20.7158174, 21.523915]	Bimodal
	Sequential input (perchar)MB/sec	[99.877581, 99.968973]	Bimodal
7	Sequential output (block)%CPU	[1231.041954]	Unimodal
	Sequential output (block)MB/sec	[52.402349300000004, 78.43738570000001, 97.963663]	Multimodal
	Sequential output (rewrite)%CPU	[2899.0309488000003, 7735.746886]	Bimodal

	Sequential output (rewrite) MB/sec	[18.5745598, 92.863703]	Bimodal
	Sequential input (perchar)%CPU	[7.801117400000001, 25.793272]	Bimodal
	Sequential input (perchar)MB/sec	[20.746663800000004, 99.933559]	Bimodal

Table 3.6 Same six features with different number of modes in different devices

Here is an explanation of how permutation samples vary depending on certain feature modes on Mac devices. For another set of features (max speed (MB/s) for the copy function, quickest duration for the add function, quickest duration for the triad function). Feature 1 is bimodal; Feature 2 is multimodal; and Feature 3 is bimodal.

In this example, the above features are taken, and 12 permutations are identified. Half of them don't have samples associated with them.

Permutations	**Samples**
000	064
001	036
010	002
011	000
020	000
021	000
100	001
101	000
110	911
111	000
120	986
121	000

Table 3.7 Combinations with different numbers of samples

Empty combinations are not considered in our experiment, and combinations with higher sample values are useful for identifying the devices.

The experiment uses 17 features, which are collected from general computing devices and pre-processed. Features were extracted and passed to the model for training with 4 different algorithms. The data was divided into two parts for holdout accuracy estimates: 80% and 20%. The model was trained with 80% and tested with 20%. Each device randomly chooses a portion of its test data, which represents 20% of the total data set. The precision value for each device and classifier, as well as the overall accuracy for each classifier, are displayed in this section. The ability of a system to categorise a sample as positive when it is actually negative is known as precision. The results were validated using k-fold cross-validation with k = 10. K-fold cross-validation divided the data into k portions and used 1 portion as test data and the rest of the k-1 portion as training data.

In this system, hardware features are utilised for generating the unique identifier. In order to apply device identification, these readings are used in the construction of the ICMetrics security system. The retrieved features are used to assess the accuracy rate of the suggested system during testing. The suggested system is put to the test using a dataset taken from the hardware features to determine the accuracy rate and the four different sorts of alarms. The effectiveness of the suggested security strategy is assessed using cross validation. The accuracy of the classification for each device is displayed in the table below. Each table displays the classification outcomes for each common classifier in comparison to the (MVGD). Each linear classifier performs poorly when presented with the data. Because of this, the variation in the feature values and relationships among them are similar when viewed from a linear perspective, which accounts for the lower accuracy and precision values for the devices. When the data is subjected to MVGD, we take into consideration the multimodal character of the data, and higher accuracy results are observed. This demonstrates the need of looking at the data in a multidimensional space and taking into account the characteristics' multimodality.

Tables 3.8, 3.9 and 3.10 show the comparison of our proposed model MVGD with other standard classifiers LR, LDA, and SVM and evaluate the performance of the proposed model on the basis of accuracy, precision, recall, and F measure [120].

In all classifiers for the first feature set, which includes eight features related to the speed of a hard disk to copy, add, scale, and triad function, MVGD performs better; its accuracy is 91.5%; after that, SVM performs better; it holds 90% accuracy.

Classifier	**Accuracy**	**Precision**	**Recall**	**F Measure**
MVGD	91.5%	74.2%	73.1%	72.5%
LDA	87%	70.7%	69.7%	68.7%
LA	87%	69.2%	69.7%	68.9%
SVM	90%	73.4%	73.5%	72.9%

Table 3.8 For feature Set 1 classification performance of proposed model with standard classifiers using Training-Testing in 10-fold cross-validation setup.

For the second feature set, which includes six features related to hard disk like CPU usage when writing to disk and memory-related features like time taken to read memory, MVGD performs better and its accuracy is 92%. After that, SVM performs better and its accuracy is 90.5%.

Classifier	Accuracy	Precision	Recall	F Measure
MVGD	92%	74.1%	73.6%	73.4%
LDA	91.2%	74%	73%	72.5%
LA	91%	74.1%	73.4%	73%
SVM	90.5%	77.2%	77.2%	77.1%

Table 3.9 For feature Set 2 classification performance of proposed model with standard classifiers using Training-Testing in 10-fold cross-validation setup.

For the third feature set, which includes three features related to CPU-related values, like the performance of floating-point arithmetic, MVGD performs better; it holds 80.1% accuracy; after that, SVM performs better; its accuracy is 67.9%.

Classifier	Accuracy	Precision	Recall	F Measure
MVGD	80.1%	67.9%	64.6%	62.1%
LDA	57.8%	44.2%	47%	42.3%
LA	57.1%	52.6%	46.4%	44.1%
SVM	67.9%	59%	55.1%	50.3%

Table 3.10 For Feature Set 3 classification performance of proposed model with standard classifiers using Training-Testing in 10-fold cross-validation setup.

Below table show the confusion matrix for first feature set for MVGD classifier explained in table 3.8

D0	200	0	5	0	1	0	0	0
D1	0	193	0	0	0	0	0	0
D2	0	0	207	0	4	0	0	0
D3	0	0	0	197	1	2	0	0
D4	0	1	0	2	194	0	0	0
D5	0	59	0	25	34	83	0	0
D6	0	0	0	0	0	0	197	0
D7	0	0	0	0	0	1	0	194
	D0	**D1**	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**

Table 3.11 Confusion Matrix for first feature set for MVGD classifier explained in table 3.8

Below table show the confusion matrix for second feature set for MVGD classifier explained in table 3.9

D0	205	0	0	0	1	0	0	0
D1	0	187	0	0	0	6	0	0
D2	0	0	201	0	10	0	0	0
D3	0	0	0	192	0	8	0	0
D4	0	0	0	0	186	11	0	0

D5	0	0	0	87	0	114	0	0
D6	0	0	0	0	0	1	195	1
D7	0	0	0	0	0	4	0	191
	D0	**D1**	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**

Table 3.12 Confusion Matrix for first feature set for MVGD classifier explained in table 3.9

Below table show the confusion matrix for third feature set for MVGD classifier explained in table 3.10

D0	200	0	4	0	1	0	1	0
D1	0	193	0	0	0	0	0	0
D2	11	0	42	0	5	0	153	0
D3	0	0	0	196	1	3	0	0
D4	0	7	0	6	184	0	0	0
D5	0	78	0	22	17	84	0	0
D6	4	0	0	0	0	0	193	0
D7	1	0	1	2	0	0	0	191
	D0	**D1**	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**

Table 3.13 Confusion Matrix for third feature set for MVGD classifier explained in table 3.10

According to the experiment results, the proposed multivariate Gaussian distribution model outperforms the other three standard classifiers in the prediction of uniquely identifying devices. After applying secret sharing (explained in section 3.6.1), the results for ICMetrics device identification for multivariate Gaussian distribution classifier were 94%, 95% and 84% for the first, second, and third feature sets, respectively. This proves that our results improved statistically over the previous ones.

Classifier	**Feature Set**	**Accuracy**	**Precision**	**Recall**	**F Measure**
Proposed MVGD	8F	94%	93%	94%	94%
	6F	95%	94%	95%	95%
	3F	84%	85%	84%	84%

Table 3.14 Secret Sharing results for 8F,6F and 3F classification performance of proposed model using Training-Testing in 10-fold cross-validation setup.

The method proposed in this thesis can be utilised to provide device identification with a higher degree of accuracy, according to the results provided in the tables above.

3.9 Conclusion

ICMetrics is a device identification system that identifies a device based on its internal environment. This chapter includes a study on explicit and implicit features that can be applied to the development of device ICMetrics. This chapter provided an overview of ICMetrics technology and demonstrated how to use it for device identification. At the outset, an overview of the structure of the ICMetrics approaches is provided. This system is novel in that it takes features from a device and uses them to identify it. Following that is a discussion of a generic list of potential features that could be used for identification. Following that, the approaches of feature extraction technologies are demonstrated and explored. Handling multimodal characteristics was the second contribution. To convert unique distributions into conventional distributions, a multimodal algorithm is created. The peak-trough detection approach is used for multimodal features to identify each mode inside the feature. This multi-modal dataset aided in evaluating the robustness of the device identification methods under various devices scenarios. This dataset was heavily used in this thesis work to perform a variety of evaluations with multiple classifiers. Using this dataset, a thorough analysis of the identification accuracy was performed. The device was then identified using a classifier comparison analysis. The device identification technique is compared to four alternative classifiers. Samples were sorted out based on modes, and samples were verified against all training samples during testing; the probability was then calculated, and this was used to predict the classifier output.

Experimental results show that different classifiers behave differently on the same dataset. The analysis revealed that the proposed model MVGD outperformed all others for device identification because multimodal features were analysed. The accuracy results for feature set 1 for our proposed classifier are 91.5%; and for the second feature set, 92%; and for feature set 3, 80.3%. These findings demonstrate that our experiment was successful. This relates to the research question in Chapter 1. And our Shamir's Secret Sharing results based on ICMetrics device identification for the proposed model MVGD are quite promising. Overall, this chapter outlines the method of analysis and mathematical implementation using the proposed model of multivariate Gaussian distribution. The findings for ICMetrics device identification obtained using Shamir's secret sharing technique are finally presented.

Chapter 4

Frequency Domain Analysis & Classification

4.1 Introduction

Multimodal features have been employed to identify devices so far in our research. The system performance of this strategy was, however, constrained. There are three restrictions: 1) Excessive data noise and abrupt interruptions, 2) In multimodal distribution, bursts of modes frequently separate and drift apart from one another. It is typical to see that these bursts occur at regular intervals or within the modes, where high frequency changes are consistent while lower frequency changes are quite inconsistent. It is typical to find that certain high frequencies are consistent for a certain device and that low frequencies tend to dominate the later when things are separated into frequency and spatial domains. Our primary objective is to ascertain if low-frequency changes in the spatial domain conceal high-frequency changes inside modes for the various devices. They cannot be differentiated without being moved to the frequency domain and 3) Multimodal-based feature analysis requires longer processing time [126]. These restrictions have been explained in chapter 3 section 3.5.1

Frequency Domain Analysis and Classification provide a solution since they:

1) Lessened the commotion and noise.
2) When evaluated in the spatial domain, low frequency conceals high frequency fluctuations within modes that are constant across devices, according to our research. It is challenging to tell them apart without going into the frequency domain.
3) Faster computations.

To further validate these points, a detailed comparison of multi modal classifier results with standard classifier is highlighted in section 4.5. This technique further helps to add an additional security layer since their coefficient can be driven and used as an additional device feature. This novel way provides for even stronger device authentication.

The use of the wavelet transform for device identification is the focus of discussion in this chapter, as are wavelet-based features to identify such devices uniquely. Our aim was to study the impact of utilising different wavelet functions on the performance of the device identification system. Importantly, the features examined exhibit non-standard and multimodal distributions, which present a significant challenge to model and characterize.

The rest of the chapter is organised as follows: Section 4.2 details the related work on this topic; Section 4.3 focuses on the system overview; following this, Section 4.4 presents the methodology process; Section 4.5 presents experimental results; and Section 4.6 draws conclusions based on the results.

4.2 Related Work

This section progressed with our research based on wavelets covered in our experiments. Earlier wavelet-based image classification work is discussed. In [127], Harr and a bank of perceptron's used a database of 600 photos to classify images (300 for training and 300 for testing). For the training set, they achieve 81.7% right classification, and for the assessment test, 76.7%. Combining wavelet transformations for image classification is done in [128]. They claim that performance is almost 80%. In [129], classification is accomplished using Daubechies' wavelet transformations. For testing, 240 colour photos of aeroplanes were used, while 120 were used for instruction. 88% efficiency was the highest recorded. In this research, the author also offers a method based on the wavelet transform.

In order to synthesise the reconstructed images using the estimated detail matrix and information matrix provided by the wavelet transform, they employed DWT to estimate the detail matrix from the information matrix [130].

By applying several threshold approaches to increase the quality of the reconstructed image, they created a computationally efficient and effective algorithm using the Haar wavelet transform for compression of lossy images [131].

Wavelet transformation is one method that has been employed for feature extraction. The wavelet transform is frequently used to extract characteristics from non-stationary bioelectrical signals [132]. The discrete wavelet transform (DWT) was used in this study due to its popularity for the measuring and analysing time-frequency and spectral component fluctuation [133,134]. The technique has been widely applied [135,136]. It is helpful for evaluating transitory signals because it makes it possible to extract features that change over time [137,138].

The Haar wavelet can split data classes without considerably reducing the original data's content, which has the advantages of being quick and easy while dealing with memory efficiency [139]. The energy spectrum of Daubechies' wavelets is symmetric, centred around low frequencies, and effective for reducing the number of dimensions needed to classify images [140,141]. Symlets are almost symmetrical wavelets with Daubechies wavelet-like properties [142]. Coiflets use approximation properties based on the number of vanishing wavelet moments [143].

While the biorthogonal wavelet has linear phase filter banks with symmetric properties, and is useful for signal and image reconstruction, it has the advantage of a dual filter that corresponds to a fixed wavelet filter used for signal decomposition [144-146], since it is predicted that bioelectrical signals collected from smart watches will contain some noise, the most effective way to minimise the noise is to use a wavelet with a filter bank. The ability to extract a sample of the signal for feature extraction will be made possible by the signal's decomposition, which will be another benefit.

A biorthogonal wavelet has the benefit of a dual filter, which corresponds to a fixed wavelet filter used for signal decomposition; it is useful for signal and image reconstruction; and it has the benefit of linear phase filter banks with symmetric property [144–145]. The most efficient technique to reduce noise in signals obtained from devices, is to use a wavelet with a filter bank. Another advantage is that the signal decomposition will enable the extraction of a sample of the signal for feature extraction.

The Bior1.1, 1.3, 1.5, Bior2.2, 2.4, 2.6, 2.8, Bior3.1, 3.3, 3.5, 3.7, 3.9, Bior4.4, Bior5.5, and Bior6.8 wavelets are members of the biorthogonal wavelet family. A signal is divided into approximation and detail components using biorthogonal wavelet transformations. The Approximation and detail

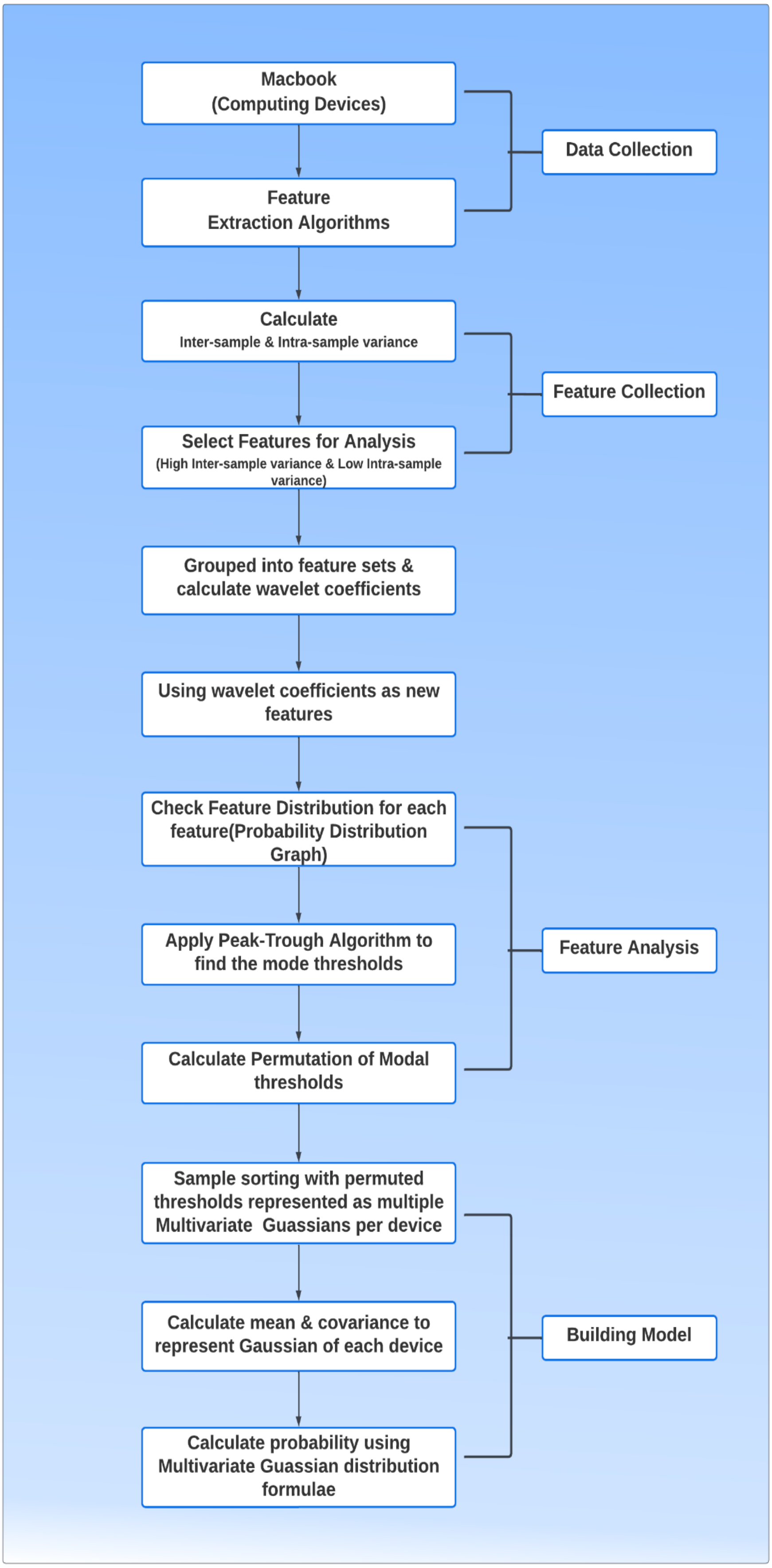
Coefficient contains pertinent data about a signal that can be used to extract features. Each n-level of the sub-band divides the bioelectrical signal into a high and low frequency signal component [153]. The parameters of the biorthogonal wavelet signal differ depending on how the signal is deconstructed or reconstructed.

4.3 System Overview

This section introduces the characteristics of computing devices and their classification based on device usage and its hardware. Our chosen platform is general purpose computing devices, as they have a wide range of applications that can be vulnerable to attacks. Hence, investigate the distinguishing device characteristics that can be used for device identification and explored a few device characteristics to see if they could effectively deliver as much information as possible [148].

This system's operation is divided into four stages

A. Criteria for Good Features: This portion of the paper introduces criteria for good features; this has already been detailed in Chapter 3.
B. Feature capture - At this stage, feature data was collected from the devices, as described in Chapter 3.
C. Feature selection - From all of the data collected, select features that meet the necessary criteria were chosen; this is described in Chapter 3.
D. Wavelet - Map the features into the frequency domain, generating wavelet coefficients and subsequently employing these coefficients as features for classification.



Macbook (Computing Devices)
Feature Extraction Algorithms
Data Collection
Calculate Inter-sample & Intra-sample variance
Select Features for Analysis (High Inter-sample variance & Low Intra-sample variance)
Feature Collection
Grouped into feature sets & calculate wavelet coefficients
Using wavelet coefficients as new features
Check Feature Distribution for each feature(Probability Distribution Graph)
Apply Peak-Trough Algorithm to find the mode thresholds
Calculate Permutation of Modal thresholds
Feature Analysis
Sample sorting with permuted thresholds represented as multiple Multivariate Guassians per device
Calculate mean & covariance to represent Gaussian of each device
Calculate probability using Multivariate Guassian distribution formulae
Building Model

Figure 4.1 Model of the Proposed System

This section explains the proposed model of the system step-by-step.

4.3.1 Data Collection

The dataset used for this experiment is the multi-modal dataset described in Chapter 3. A total of eight different devices with three different behavioural characteristics - hard disk performance, memory performance, and CPU floating point performance are used for this analysis.
The data was acquired under various usage scenarios as well. A code written in Python was used to capture this data explained in detail in Chapter 3.

4.3.2 Feature Selection

After reviewing the raw data in section 4.3.1, the static features were found there and deleted since they had the same values throughout the whole data collection stage and the same values for the devices. Additionally, these characteristics were removed in the preliminary exams.

After making this initial choice, our focus was on conducting statistical analysis to determine the correlation, mean, standard deviation, and variance in order to determine the best way to model the distinct behaviour of the features (per device) of the data that was collected and to identify minor variations between the datasets that were acquired from each device. These first entities were created using the raw data. And then, based upon the criteria for good features (discussed in chapter 3) features were selected. After feature selection, create feature sets, which are detailed in Chapter 3. In this same category of features, the sets are gathered for study to see how they might help us identify between devices and enhance system performance.

The novelty of this work is to use the above feature set data to calculate discrete wavelet coefficients and then use those coefficients as a new feature for device identification.

4.3.3 Feature Analysis

This section consists of a detailed analysis of the feature data. The first step is to check feature distributions for each feature (per device). The distribution of each characteristic is then shown on a probability density graph so that users can understand how they are distributed and whether any features on different devices overlap. It is also worth noting whether the features are multimodal, bimodal, or unimodal. Here our focus is on wavelet coefficient features derived from hardware feature sets explained in Chapter 3. The probability density graph of a feature with a multimodal distribution is shown in the figure 4.2 below. The next step is to build the model and compare the performance based on the classifier's accuracy.

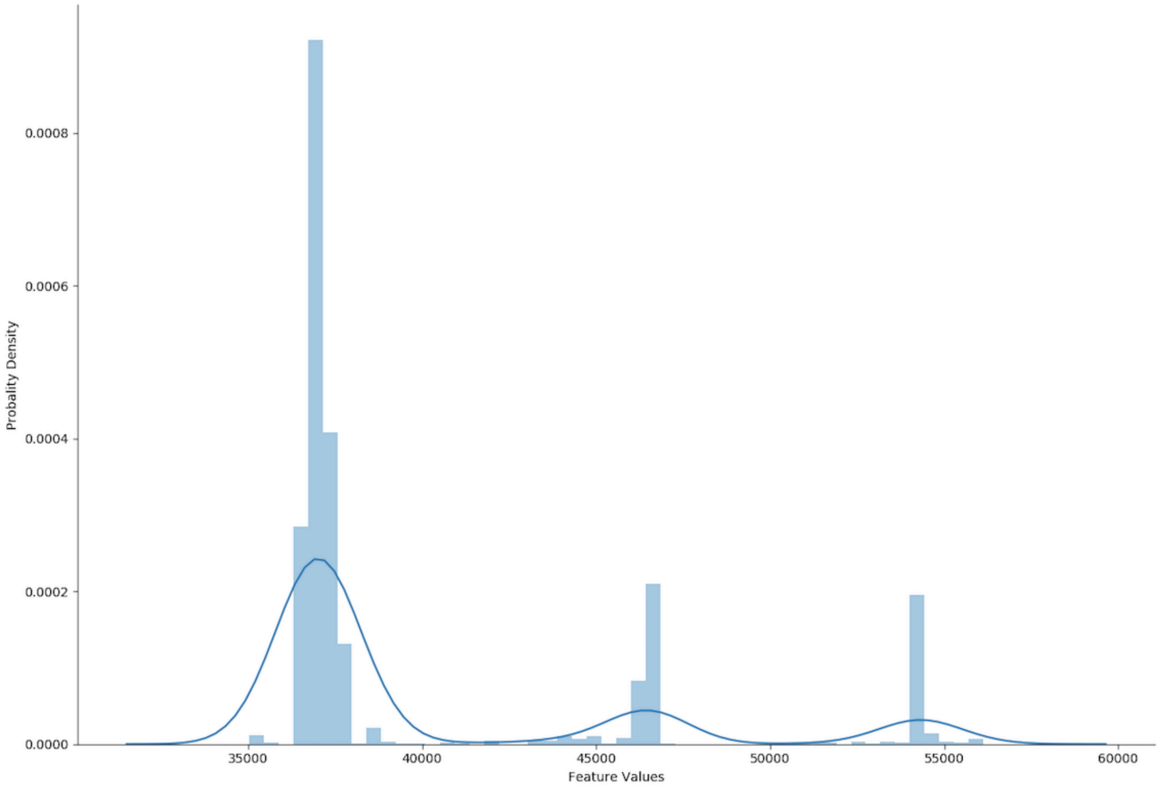


Figure 4.2 Probability density function graph of a feature having multimodal distribution

4.3.4 Discrete Wavelet Transform

Our contribution here is to use hardware features in the frequency domain for device identification. To do this, our choice was DWT because of the following advantage: DWT provides sufficient information for the analysis of original data with a significant reduction in computation time [149]. In DWT, wavelets are transient functions of short duration, that is, limited duration centring around a specific time. The DWT decomposes a dataset into numerous scales indicating different frequency bands, and at each scale, the position of the DWT can be determined at the important time characteristic with which the electrical noise can be recognised and successfully removed. The DWT is measured an appropriate mechanism for noise removal as an innovative substitute that changes ways of reducing noise in systems through the use of low-pass filters. For situations that need signal reconstruction, a discrete wavelet transform can be used [149]. Since there is no conversion from scale to frequency required, the DWT is simpler to calculate, and the wavelet coefficients are simpler to understand.

The advantages mentioned above are compelling reasons to investigate this novel method of exploring features in the frequency domain using a discrete wavelet transform, which helps us improve the accuracy of the system.

Figure 4.4 shows the process of wavelet decomposition. Performing the discrete wavelet transform (DWT) of a signal x is done by passing it through low-pass filters (scaling functions) and high-pass filters simultaneously [149].

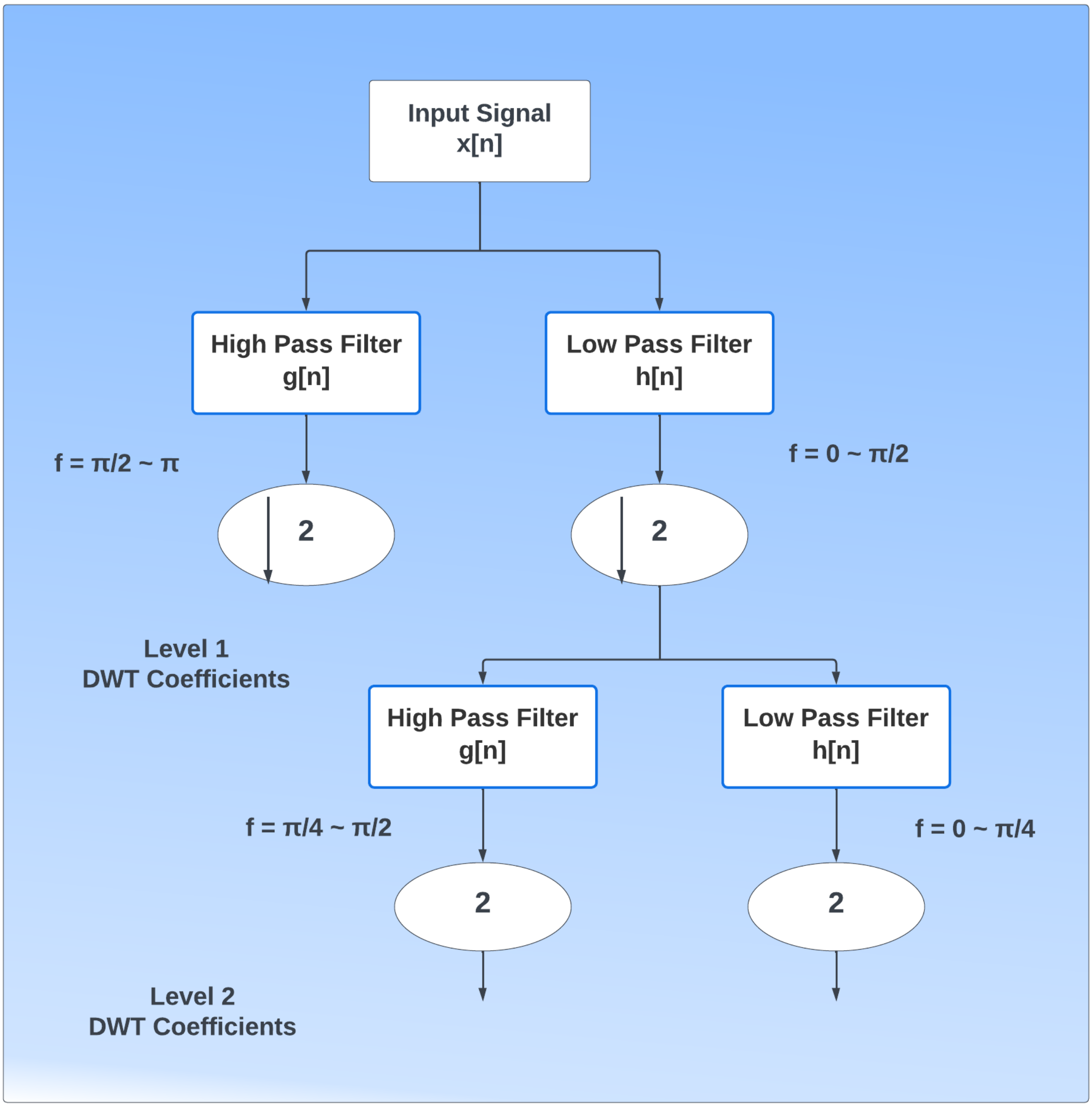


Figure 4.3 Process of Wavelet decomposition.

The results provide the detail coefficients (from the high-pass filter) and approximation coefficients (from the low-pass filter). The output of the low-pass filter is then subsampled by 2 and further processed by passing it again through a new low-pass filter and a high-pass filter with half the cut-off frequency of the previous one. This decomposition has halved the time resolution since only half of each filter output characterises the signal. Though, every output has half the frequency band of the input, the frequency resolution has been doubled. These filters are relevant to our data because of features like memory, CPU, hard disk, and using these filters, noise is reduced from the data and the high and low coefficient data is visualised, which helps us identify how our data is behaving, is capable of identifying the device, and speeds up the classification process [150].

An algorithm for DWT is the signal to be modified is x, the low-pass and high-pass filters are L and H respectively, and the number of filter bank iterations is n [150].

Step 1: Let's say feature vector X = [0,1,2,10,2,1,0,0].

Step 2: The first step of the wavelet decomposition is to split our information into two vectors of four components, say approximation coefficient a (1) and detail coefficient d (1).

Step 3: Calculate Approximation Coefficients

$$a(1) = \frac{X(2k)+X(2K+1)}{\sqrt{2}} \qquad (k= 0,1,2,3) \text{ ---------------------}(1)$$

Step 4: Calculate Detail Coefficients

$$d(1) = \frac{X(2k)-X(2K+1)}{\sqrt{2}} \qquad (k= 0,1,2,3) \text{ ---------------------}(2)$$

Step 5: Approximation Coefficients

$$a(1) = \frac{1}{\sqrt{2}},\frac{12}{\sqrt{2}},\frac{3}{\sqrt{2}},0$$

Step 6: Detail Coefficients

$$d(1) = -\frac{1}{\sqrt{2}}, -\frac{8}{\sqrt{2}},\frac{1}{\sqrt{2}},0$$

The next step is to use the step 5 and 6 approximations and detail coefficients as new wavelet features to identify the device uniquely. For further analysis, the same steps were used as discussed in sections 4.4.6.5.

4.3.5 Wavelet used for analysis

This section contains information about the wavelet used for feature extraction. Mother wavelets in WT play a significant role in the analysis of their shapes, which vary according to the application. Given the variety of mother wavelets, choosing one is a challenge for obtaining the most precise findings from the various analysis, which entails a strong correlation between the signal and the mother wavelet.

The wavelet families Daubechies, Coiflet, Haar, Symlet, and Bi-orthogonal were chosen as candidate wavelet functions. The variance technique is used as a selection criterion to choose the best wavelet function. This is related to the variance of the wavelet coefficients. As a mother wavelet, the wavelet with the highest sum of variance of wavelet packet coefficients can be used (WPC) [151]. In our system, the variance method is used for both feature and wavelet selection. In our analysis, different statistical measures were calculated, and variance is one of them.

These wavelets have been used, and their properties have been described below:

1. **Haar wavelet** are orthogonal, symmetrical, and compact.
2. **Daubechies'** wavelet lacks an unambiguous formulation, is orthogonal, asymmetrical, and introduces phase distortion.
3. **Coiflet** wavelet has regularity, is nearly symmetric, is orthogonal, and has compact support.
4. **Symlet** wavelet orthogonal, nearly symmetric, compact support, and regularity characterise
5. **A Biorthogonal** wavelet has regularity, compact support, symmetry, and orthogonality.

All these wavelets were used since they gave us better results. Without these, the output was approximately 5% less.

Wavelet coefficient features derived from different categories of hardware features like CPU, memory, and hard disk are vary based upon usage scenarios. The above properties of wavelets help us analyse the system's performance and improve its accuracy.

4.3.5.1 Haar

It is one of the most unsophisticated parts of the wavelet family. This is a theoretically simple, low-cost, easy to apply and memory-efficient wavelet transform. It uses just two scaling and wavelet function coefficients and decomposes a signal into two sublevels: one known as the average and the other as the difference. This wavelet family looks like a step function and is non-continuous in nature [152] [153]. The inability of the Haar wavelet transform technology to offer compression and noise removal for audio signal processing applications is a drawback [154] [155]. This wavelet helps us generate the results quickly with low computation power and reduces the feature dimension because of the way it calculates coefficients.

4.3.5.2 Coiflet

In general, wavelets have the largest number of vanishing moments for both scaling functions and wavelet functions for a given support width and are compactly supported wavelets. 'Coif' is the abbreviation for these wavelets. There are several orders in this family, including Coif1, Coif2, Coif3, Coif4, and Coif5 [3]. Coiflet was constructed with the vanishing moments of the wavelet function (phi) and scaling function (psi). The wavelet function has 2N moments, and the scaling function has 2N-1 moments equal to 0. These functions together have the support number 6N-1. The number of vanishing moments is highest in coiflets for a given support width, i.e., phi and psi [156]. The wavelet and scaling functions are both normalised by a factor. The scaling function of this family demonstrates the interpolating attributes that imply excellent approximations of polynomial functions at various resolutions. The symmetrical properties of coiflets are advantageous in signal analysis work due to their linear phase in the transfer function. It presents both time and frequency information as essential arrangements [152].

4.3.5.3 Daubechies

Daubechies's wavelets are capable of symmetry, with the energy scale focused around low frequencies and efficient for dimension reduction in data classification. Daubechies is a group of wavelets introduced by Ingrid Daubechies that are detached from the number of polynomial degrees that build up the wavelet, the number of instants lost, or the size of the filter coefficient used [152]. The number of polynomial degrees, the number of instants lost, and the size of the filter coefficient have values that are related to one another. The number of polynomial degrees and the number of lost instants have the same value, while the length of the filter coefficient is twice that value. Assume the degree of the polynomial that forms a Daubechies is 4, the number of lost instants is also 4, and the number of filter coefficients used is 8. These values are also taken into account by the name daubechies wavelet. For example, daubechies-1, daubechies-2, ... Daubechies-N. Several use the number of filter coefficients as a means of naming daubechies wavelets, for example, daubechies-2, daubechies-4, daubechies-6, ... daubechies-2N.

4.3.5.4 Biorthogonal

A biorthogonal wavelet has definite properties like perfect reconstruction and linear phase properties. It also has the benefit of a dual filter, which corresponds to a fixed wavelet filter used for signal decomposition. In a biorthogonal wavelet, there are two scaling functions and two different wavelet functions. A biorthogonal wavelet transforms split data into approximation and detail coefficients. The Approximation and Detail Coefficient holds appropriate information about a dataset from which features can be extracted. Data extracted from devices is expected to come with some level of noise; therefore, using a wavelet with a filter bank to reduce the noise will be most appropriate [157].

4.3.5.5 Symlet

The Symlet family wavelet is derived from the variations in the Daubechies family. These variations are the symmetry modifications, hence the name 'Symlet'. This family of wavelets is nearly symmetrical, biorthogonal, and orthogonal in nature. The properties of sym and db wavelets are similar and comparable. symN, where N is the order. The symN has seven different functions, from sym2 to sym8. These have the maximum number of vanishing moments for a compact support in time [158] [159].

The next step is to use the new wavelet features for analysis once the approximation and detail coefficients have been calculated. Wavelet features, which are multimodal in nature, are treated similarly to the detailed discussion in Chapter 3.

4.4 Experimental Methodology

The novelty of this work is to evaluate the proposed wavelet feature-based device identification using its potential as a basis for classifier accuracy. The experimental dataset contains the features explained in chapter 3, Table 3.2. The data is collected in a monitored environment where our goal is to track device activity during data collection. This gives an understanding of the behaviour of the features during the analysis. To understand the potential of the candidate features in the frequency domain, the hardware features used for evaluation are transformed into wavelet coefficients. These coefficients are then employed as features of the devices for classification.

4.4.1 Device Identification Phase

The first step in the device identification phase is to extract hardware features from all the devices. A total of 38 features from different categories were extracted, and then based upon selection criteria, 17 features were used for this analysis. In the next step, the same category of features is grouped into different sets, and feature sets from different categories are concatenated together to form a feature vector. This feature vector underwent feature normalisation and selection processes. The device identification was performed based on the proposed model classifier and three different classifiers (SVM, LDA, and LR), and a final result was generated for the incoming input sample.

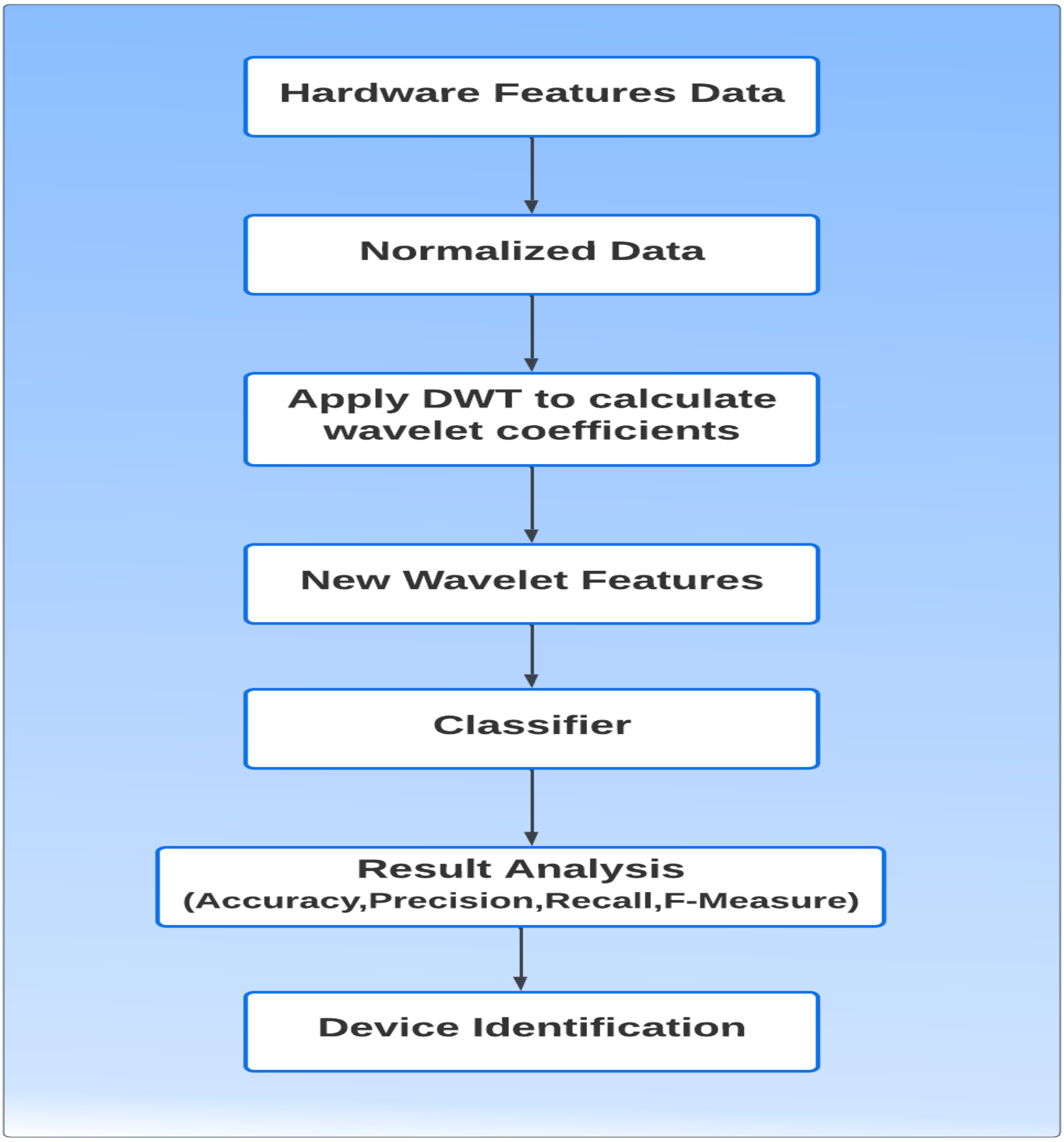

Hardware Features Data
Normalized Data
Apply DWT to calculate wavelet coefficients
New Wavelet Features
Classifier
Result Analysis (Accuracy,Precision,Recall,F-Measure)
Device Identification

Figure 4.4 Device Identification Phase

4.4.2 Feature Normalisation

Two feature vectors, X and Y, may have different ranges and distributions for individual feature values. As part of the feature normalisation process, the mean and variance of each individual feature value are adjusted in order to compare how much each feature contributed to the overall match score. This phase used a min-max normalisation strategy [160].

4.4.3 K-fold cross validation

After normalisation, k-fold cross-validation method was adopted to split the data into training and testing. A resampling technique called cross-validation is used to assess models using small data samples, like the one used in our experiment. The number of groups into which a given data sample is to be divided is indicated by the parameter 'k.' The following actions were taken while doing the stratified k-fold cross validation: samples from the feature set were mixed up at random. Ten folds were created for feature sets. For each unit of measurement breakdown, the model was fitted to the training set, and assessment was carried out on the test set, yielding an evaluation score. One group was set aside as the test data set, while the remaining groups were taken as the training data set. The number of folds was chosen as five owing to the limited number of samples.

4.4.4 Classifiers

This proposed method has the ability to offer strong device identification. In order to evaluate the model as a classifier, the correct classification of a unique identifier is required. In this process, a multivariate Gaussian distribution is leveraged.

In this proposed methodology, three standard classifiers for benchmarking were used, namely, logistic regression (LR), linear discriminant analysis (LDA), and support vector machines (SVM) [161] as explained in section 3.7.1. These classifiers are used to compare the predictions. Based on three feature sets, the test data was accurately classified using the aforementioned classifiers on various device datasets. As explained in the diagram above, our analysis was based on the wavelet coefficients as features for classification. The performance of these classifiers is evaluated on the basis of accuracy, precision, recall, and F measure. The results section contains the analysed results based on these. These classifiers code is implemented in Python.

This work represents a comparison amongst four classification techniques, evaluating which of these techniques is best suited to identify and classify the devices based on the collected data. In the next section the general algorithm that was used to model our proposed classifier is explained.

4.4.4.1 Algorithm for the proposed system

The algorithms below introduce the process of generating wavelet coefficients for classification.

Step 1: Read device data.
Step 2: Split device data into training and testing using k-fold.
Step 3: Determine the wavelet coefficients for the training split.
Step 4: Determine the wavelet coefficients for the test split.
Step 5: Repeat Steps 1-4 for all the devices.
Step 6: Verify training data against test data

1. Read the approximation and detail arrays one at a time.
2. Generate a probability density graph for each feature.
3. Analyse the distribution to see if it is unimodal, bimodal, or multimodal.
4. Apply Peak-trough to calculate mode thresholds if there are multiple modes in the distribution.
5. Use approximation array, threshold and test dataset
    a. Utilize the column threshold to compute permutation indices.
    b. Calculate permutation samples for training and testing.
    c. Calculate all probabilities for permutations.
6. Calculate the device probability.
7. Calculate the maximum probability.
8. Repeat steps 1-5 for each device and for each feature set.

Step 7: Apply the benchmarking classifier. Multivariate Gaussian Distribution [161], Logistic Regression [162], and Linear SVM [163] were used to predict the results' accuracy.
Step 8: Results are analysed based on accuracy, precision, recall, and F-measure.

4.5 Experimental Results

This section presents a discussion of the obtained experimental results with the standard classifiers mentioned above and then compares the results with raw feature data. Here, experiments use wavelet-based features (when raw feature data is subjected to a wavelet transform). This data, after analysis, gives us a unique identifier. Eight devices were used with updated software. For this work, data was collected from the MacBook Air and Pro, Python code, and Microsoft Excel. Each device contained thousands of samples for our analysis. The cross-validation method with a fold value of 10 has been used for training and testing phases. Consequently, all of the records that exist in the dataset will affect the training and testing of the classifiers.

Tables 4.1, 4.2, 4.3, 4.4, 4.5, 4.6 show the comparison of our proposed model's MVGD wavelet-based feature with other standard classifiers such as LR, LDA, and SVM and evaluate the performance of the proposed model on the basis of accuracy, precision, recall, and F measure, as explained in section 3.7.2.

Db2 Wavelet FS1 (Approximation)				
Classifier	**Accuracy**	**Precision**	**Recall**	**F Measure**
MVGD	93.2%	93%	93%	93%
LDA	89.2%	89.1%	88.2%	87.8%
LR	88.5%	88.1%	87.3%	87.1%
SVM	92.5%	92%	91.5%	91%
Db2 Wavelet FS2 (Approximation)				
Classifier	**Accuracy**	**Precision**	**Recall**	**F Measure**
MVGD	98%	97.9%	97.8%	97.7%
LDA	89.3%	89.8%	88.3%	87.3%
LR	80.4%	84.2%	79.1%	76.5%
SVM	91%	92.5%	90.6%	90%
Db2 Wavelet FS3 (Approximation)				
Classifier	**Accuracy**	**Precision**	**Recall**	**F Measure**
MVGD	81%	80.9%	80.8%	80.7%
LDA	46.9%	53%	48%	42%
LR	64.8%	62%	65%	57%
SVM	44.5%	64%	47.2%	40%

Table 4.1 Db2 wavelet (approximation) for all 3 feature sets classification performance of proposed model with standard classifiers using training-testing in 10-fold cross-validation setup.

While Haar compressed signals via averaging and differencing, Daubechies are an orthonormal wavelet that is compactly supported and maintains the energy of signals. In our experiment (Table 4.1 and Table 4.5) it has been found that Daubechies filters produce better classification outcomes than Haar but require more calculation time due to the longer support of scaling and wavelet coefficients.

Bior1.3 Wavelet FS1 (Approximation)				
Classifier	**Accuracy**	**Precision**	**Recall**	**F Measure**
MVGD	92%	91.9%	91.8%	91.7%
LDA	86.5%	88%	86%	85%
LR	87.3%	87%	87%	86%
SVM	90.9%	90%	89%	89%
Bior1.3 Wavelet FS2 (Approximation)				
Classifier	**Accuracy**	**Precision**	**Recall**	**F Measure**
MVGD	81.8%	78%	82%	78%
LDA	80.3%	82%	81%	81%
LR	76.4%	82%	78%	73%
SVM	80%	84%	82%	80%
Bior1.3 Wavelet FS3 (Approximation)				
Classifier	**Accuracy**	**Precision**	**Recall**	**F Measure**
MVGD	70.2%	76%	72%	69%
LDA	36%	36%	39%	30%
LR	47.2%	44%	51%	38%
SVM	54.9%	62%	57%	51%

Table 4.2 Bior1.3 wavelet (approximation) for all 3 feature sets classification performance of proposed model with standard classifiers using training-testing in 10-fold cross-validation setup

In Table 4.2 MVGD classifier performed best as compare to other classifier and out of three feature set, FS1 performed best in Bior 1.3. This is because FS1 features are related to memory performance of the system, which are largely dependent on how users use the machine.

Sym2 Wavelet FS1 (Approximation)				
Classifier	**Accuracy**	**Precision**	**Recall**	**F Measure**
MVGD	99%	98.9%	98.8%	98.7%
LDA	89.2%	89%	88%	88%
LR	88.5%	88%	87%	87%
SVM	91%	92%	91.5%	91%
Sym2 Wavelet FS2 (Approximation)				

Classifier	**Accuracy**	**Precision**	**Recall**	**F Measure**
MVGD	99%	98.9%	98.8%	98.7%
LDA	89.3%	90%	88%	87%
LR	80.4%	84%	79%	76%
SVM	91%	92%	91%	90%
Sym2 Wavelet FS3 (Approximation)				
Classifier	**Accuracy**	**Precision**	**Recall**	**F Measure**
MVGD	65%	64.9%	64.8%	64.7%
LDA	46.9%	53%	48%	42%
LR	64.8%	62%	65%	57%
SVM	44.5%	64%	47%	40%

Table 4.3 Sym2 wavelet (approximation) for all 3 feature sets classification performance of proposed model with standard classifiers using training-testing in 10-fold cross-validation setup.

In Table 4.3 MVGD classifier performed best as compare to other classifier and out of three feature set, FS1 and FS2 performed best in Sym 2. This is because FS1 and FS2 are related to memory and CPU performance of the system, which are largely dependent on how users use the machine.

Coif1 Wavelet FS1 (Approximation)				
Classifier	**Accuracy**	**Precision**	**Recall**	**F Measure**
MVGD	90%	90.00%	90.00%	89.00%
LDA	88.50%	89%	88%	87%
LR	88.60%	89%	88%	87%
SVM	89%	89%	89.00%	89%
Coif1 Wavelet FS2 (Approximation)				
Classifier	**Accuracy**	**Precision**	**Recall**	**F Measure**
MVGD	89%	89.00%	89.00%	88.00%
LDA	88.10%	87%	88%	87%
LR	87.00%	88%	88%	87%
SVM	88%	89%	87%	86%
Coif1 Wavelet FS3 (Approximation)				
Classifier	**Accuracy**	**Precision**	**Recall**	**F Measure**
MVGD	72%	77.00%	74.00%	71.00%
LDA	45.70%	49%	47%	40%
LR	59.80%	57%	61%	52%
SVM	58.50%	62%	61%	54%

Table 4.4 Coif1 wavelet (approximation) for all 3 feature sets classification performance of proposed model with standard classifiers using training-testing in 10-fold cross-validation setup

In Table 4.4 MVGD classifier performed best as compare to other classifier and out of three feature set, FS1 performed best in Coif1. This is because FS1 features are related to memory performance of the system, which are largely dependent on how users use the machine.

Haar Wavelet FS1 (Approximation)				
Classifier	**Accuracy**	**Precision**	**Recall**	**F Measure**
MVGD	90%	91.00%	89.00%	88.00%
LDA	87.30%	89%	86%	86%
LR	76.70%	81%	76%	73%
SVM	89%	89%	90.00%	91%
Haar Wavelet FS2 (Approximation)				
Classifier	**Accuracy**	**Precision**	**Recall**	**F Measure**
MVGD	92%	92.00%	91.00%	90.00%
LDA	90.10%	90%	90%	90%
LR	89.10%	90%	89%	88%
SVM	91%	91%	91%	90%
Haar Wavelet FS3 (Approximation)				
Classifier	**Accuracy**	**Precision**	**Recall**	**F Measure**
MVGD	71%	76.00%	71.00%	69.00%
LDA	31.30%	39%	33%	26%
LR	40.50%	40%	42%	33%
SVM	42.70%	49%	44%	36%

Table 4.5 Haar wavelet (approximation) for all 3 feature sets classification performance of the proposed model with standard classifiers using training-testing in a 10-fold cross-validation setup

In Table 4.5 MVGD classifier performed best as compare to other classifier and out of three feature set, FS2 performed best in Haar. This is because FS2 features are related to CPU performance of the system, which are largely dependent on how users use the machine.

Sym6 Wavelet FS1 (Approximation)				
Classifier	**Accuracy**	**Precision**	**Recall**	**F Measure**
MVGD	90%	91.00%	90.00%	89.00%
LDA	89.40%	89%	88%	88%
LR	84.60%	82%	83%	82%
SVM	89%	90%	88.00%	88%
Sym6 Wavelet FS2 (Approximation)				
Classifier	**Accuracy**	**Precision**	**Recall**	**F Measure**
MVGD	87%	86.00%	86.00%	85.00%

LDA	86.00%	87%	86%	85%
LR	81.90%	86%	82%	80%
SVM	86%	88%	85%	86%
Sym6 Wavelet FS3 (Approximation)				
Classifier	**Accuracy**	**Precision**	**Recall**	**F Measure**
MVGD	73%	79.00%	73.00%	70.00%
LDA	59.10%	59%	60%	55%
LR	50.10%	54%	51%	46%
SVM	71.00%	77%	72%	69%

Table 4.6 Sym6 wavelet (approximation) for all 3 feature sets classification performance of proposed model with standard classifiers using training-testing in 10-fold cross-validation setup

In Table 4.6 MVGD classifier performed best as compare to other classifier and out of three feature set, FS1 performed best in Sym6. This is because FS1 features are related to memory performance of the system, which are largely dependent on how users use the machine.

Raw Feature FS1 (without wavelet transform)				
Classifier	**Accuracy**	**Precision**	**Recall**	**F Measure**
MVGD	89.5%	89.00%	88.00%	88.00%
LDA	87.00%	88%	87%	86%
LR	88%	87%	88%	88%
SVM	88.5%	88%	87%	87.4%
Raw Feature FS2 (without wavelet transform)				
Classifier	**Accuracy**	**Precision**	**Recall**	**F Measure**
MVGD	80.5%	79%	78%	78%
LDA	79%	80%	78%	78%
LR	78%	77%	78%	77.4%
SVM	76%	76%	75%	75.4%
Raw Feature FS3 (without wavelet transform)				
Classifier	**Accuracy**	**Precision**	**Recall**	**F Measure**
MVGD	64.5%	63%	62%	62.4%
LDA	57.80%	55%	59%	53%
LR	63%	62%	61%	61.4%
SVM	56%	55%	55%	55%

Table 4.7 Raw feature (without wavelet transform) for all 3 feature sets classification performance of the proposed model with standard classifiers using training-testing in a 10-fold cross-validation setup

In Table 4.7 MVGD classifier performed best as compare to other classifier and out of three feature set, FS1 performed best. The conclusion is that wavelet-based features perform significantly better.

This paragraph explains the detailed analysis of the results in tables 4.2 - 4.7. These results are based upon wavelet features and raw features for each feature set, respectively. Six different mother wavelets were used to generate coefficients for wavelet features, and the results were compared to see which of six wavelets produced the best results.

Amongst all classifiers, MVGD worked best for all three feature sets for approximation in the cases of db2, bior1.3, Sym2, coif1, haar, and sym6 wavelets. For our analysis, three feature sets were used, out of which FS1 and FS2 accuracy percentages are the highest as compared to the third feature set. These feature sets contain the data related to disks, like CPU performance and disk speed, when read and write operations are performed. This can be unique to different devices; hence, the highest accuracy can be achieved from this feature set. Our wavelet feature-based results are better than non-wavelet-based results because when wavelets are used, a large amount of data is compressed and divided into two parts: high-pass and low-pass filter data. The compressed data are then used to do our analysis to uniquely identify the device, and the second point for improved wavelet results is the removal of noise from the data. Out of six wavelets, Sym2 is the best for the first and second feature sets and Db2 is the best for the third feature set for device identification. The MVGD classifier shows the best accuracy results.

From the experiment results, it was observed that the proposed model of multivariate Gaussian distribution using wavelet-based features performed better as compared to the other three standard classifiers in the prediction of uniquely identifying devices.

4.6 Conclusion

This chapter presents a novel wavelet feature based device identification scheme. In this work, hardware characteristic features were explored and wavelet-based features were used to identify electronic devices uniquely. This co-relates to research Question#1. The comparison and analysis of classifiers for the prediction of identifying the device were performed. This co-relates to research Question#3. The device identification technique is compared to four alternative classifiers. Experimental results show that different classifiers behave differently on the same dataset. Overall, this chapter outlines the method of analysis and mathematical implementation using the proposed model of multivariate Gaussian distribution.

Our conclusion was that device identification using wavelet features yields a higher percentage of accuracy in comparison with raw features. Overall, wavelet features give better results compared to raw features, and Sym2 and db2 perform the best out of the six wavelets.

Chapter 5

Device Authentication using Homomorphic Encryption

5.1 Introduction

Data transmission over the internet must be secure, especially if it contains sensitive or personal information. As previously mentioned, the scope of our investigation was only eight test devices in our setting. Additionally, since these devices were distant, this didn't allow us to run test scenarios in real time. As a result, the main test device (a Mac Laptop) had to be used to save all template data from eight machines. The sample data must be safeguarded because it is kept on the main device (in clear text). At the time of operation, these distant machines communicate the template data, which was vulnerable to interception while travelling. There was a need to create an easy approach to secure data. Data encryption is one way to protect the messages, but attackers may still attempt to steal the encryption key.

The solution to the above problem is homomorphic encryption (HE) algorithm, which allows:

1) Mathematical operations to be performed on encrypted data.
2) This avoids decryption of the sensitive data
3) The process is relatively faster (since decryption is not required)
4) Decryption key protection overheads are completely avoided

These are key beneficial characteristics that led us to decide on using HE in device authentication. As such, once the template data was available in primary machine, to protect it from attacks, it was encrypted. Lastly, when device identification was required, this data had to be operated in a secure manner. HE helped achieve this task. Our implementation of HE allowed ICMetrics code to run the identification logic on encrypted device data.

Talking about data, there are three different forms of it: stored, transmitted, and operated on. There are many algorithms available today to secure the first two data variations; however, few of these algorithms can operate on encrypted data, which is where homomorphic encryption comes in because it makes it possible to perform calculations on encrypted data. This means that data processing can be outsourced to a third party without the need to trust the third party to properly secure the data. Without the proper decryption key, the original data can't be accessed.

In general, data security must meet the three criteria listed below:

Confidentiality: One of the most important components of data security is confidentiality. Data confidentiality means that only authorised users are permitted access to the data. Data confidentiality guarantees that unauthorised users are kept at bay.

Data integrity: is the term used to describe safeguarding data from unauthorised change. Data integrity must be implemented in the cloud to prevent unauthorised data modification.

Data accessibility: Accessibility is a crucial component of data security. Data availability aims to provide clients with secure network access to their data at any time, from any location.

Data processing on encrypted data in cloud environments is a novel method for securing data. To create such a solution, new encryption techniques known as homomorphic encryption have been introduced. This technique permits the operation of encrypted data while guaranteeing data confidentiality during processing.

Since device data was collected from various sources and kept in clear text, a compromise would result in a spoofing attack, and the bad actor could impersonate the target device. To ensure the data on this device was not compromised, it had to be encrypted. Now, the question was, 'How to protect the data on the device or server?' The most viable option was to leverage file encryption tools.

To achieve this, the following are the alternative techniques that were considered:

A) Password based: the user defines the password at the time of file encryption. During decryption, the tool will prompt for the password. It is convenient, but it is not very secure because malware can read the file and attempt an offline brute force attack.

B) Token or dongle based: The decryption private key is available on a hardware token or dongle, which needs to be presented at the time of decryption. The token has a maximum of 10 PIN attempts; after which it initialises. This approach requires a token, however, and is much more secure.

HE does not require password or private key protection, resulting in lower overhead to manage and a better solution. Thus, it was the most viable choice for this purpose.

The term ‘big data analytics’ describes the capacity to efficiently ingest enormous amounts of data, analyse it, and draw conclusions and inferences from it [164]. How to maintain the security and privacy of the data while processing it is a significant concern for businesses that collect, transport, store, and use large data sets. Numerous studies have been done on the issues with securing data when it is in transit and at rest. How to preserve data, in particular big data, effectively and securely while it is being processed is the last unresolved problem. Because there aren't any effective methods for securing data privacy and security when it's processed remotely (in the cloud), data security breaches are frequently reported, sometimes in ways that are egregiously serious and other times in ways that seem insignificant. Examples of exceptionally serious data breaches include the 2017 Equifax data breach, which predominantly affected 143,000,000 Americans [165]. A large portion of the Equifax data that was obtained was in plaintext [166], probably for computational ease. Another especially catastrophic incident occurred in 2015 when the US Office of Personnel Management suffered a data breach, and more than 21.5 million records were stolen [167]. Other data breaches had a smaller effect but were nonetheless widely publicised.

The technique used to solve this problem is homomorphic encryption (HE), which is data processing delegation without granting access to it. HE enables operations to be performed directly on encrypted data without ever using the decryption key. Using HE, data is encrypted on the client side, pushed into the cloud, securely processed, and the results are sent back to the client for decryption. The alternative to this approach was to use a large number of real devices, which has been a limitation.

When compared to a password or token-based approach, HE has some advantages. There is no need for:

1) Password management. Hence, it offers a higher level of security.
2) Dongle management. As a result, the logistics and costs of hardware can be avoided.

Homomorphic encryption techniques offer creative ways to support calculations on encrypted data while protecting the content of private data. These methods do, however, have certain drawbacks, such as high computational costs and the requirement for custom adjustments for every case study.

Along with authentication, homomorphic encryption can be leveraged for device identification by performing mathematical operations on encrypted data, which will ensure the protection of the original data. This is the focus of discussion in this chapter. Features examined in Chapter 4 exhibit non-standard and multimodal distributions, which present a significant challenge to model and characterize. The details of the mathematical functions were investigated to see if there is a way to test samples and compare them for closeness to the training models for device identification using encrypted modelled data.

The rest of the chapter is organised as follows:
Section 5.2 focuses on related works, Section 5.3 describes homomorphic encryption, Section 5.4 explains the proposed system, Section 5.5 focuses on experiments, and Section 5.6 concludes the chapter.

5.2 Related Works

According to the related work described below, data security has been the subject of substantial investigation.

For monitoring chronic diseases, the study in [168] established an IoT based architecture with homomorphic encryption to protect against data loss and spoofing assaults. The findings imply that homomorphic encryption offers simple, affordable protection for private health information. For the protection of medical data, block chain technologies are also used in conjunction with homomorphic encryption.

To prevent potential password breaches, the author [169] proposes a novel authentication system based on the password authentication protocol, which uses two servers modified to store passwords. The El Gamal algorithm and DH are employed in this paper. The backup services are offered to maintain the service. Client data from server one is retained as a backup on server two, and vice versa. If one of the two servers were to shut down for whatever reason, the client would still need to get services from another server. This protocol offers protection from both active and passive assault [169]

In 2016, the author proposed a banking application for data security. The bank contains a large amount of confidential customer information that must be protected from unauthorised access, so the data must be kept confidential. In this paper, paillier HE is used to apply operations to encrypted banking information because it enables performing calculations on cipher text without the use of a secret key. This plan provides data security and confidentiality [170].

One of the most secure authentication methods, according to a paper [171] published in 2016, is the use of biometric validation. The biometric data is saved on the remote server in encrypted form. In this paper's proposed palm print authentication approach, the matching of the user input to the registered biometric data is computed in an encrypted domain based on pailliar homomorphic encryption. This plan is carried out effectively [171].

They proposed a security model for biometric verification in this study in 2017, and in this study, they propose a new verification scheme based on HE for template protection employing multi-biometric, with the Paillier homomorphic encryption scheme used for data encoding, processing, and decryption. By computing the original biometric data and the encrypted template, HE verification handles the sole cipher text. High accuracy rates can be seen in the results [172].

Encryption methods like homomorphic encryption are also used to protect medical data. Such institutions as hospitals and research institutes are developing technical solutions for sharing patient data in a privacy preserving manner. Two of these technical solutions are homomorphic encryption and distributed ledger technology. Homomorphic encryption allows encrypted patient data to be shared with other health care service providers while it is encrypted [173].

The authors of [174] suggest a homomorphic encryption-based online safe multiparty computation with patient information sharing to hospitals. In this paper [175], a homomorphic encryption model based on heart rate data was proposed and linked to a personal health information system. The findings show that, despite anticipated storage and network issues, the technique presented was successful in meeting the needs for secure data processing for 500 patients. The authors of [176] presented a data division scheme-based homomorphic encryption for wireless sensor networks. The findings demonstrate that data security and resource availability are mutually exclusive. By monitoring the patients' vitals with a simple encryption system, [177] demonstrates the applicability of homomorphic encryption. While encryption only occurs in medical facilities, sensor data like breathing and heart rate are encrypted using homomorphic encryption before being transmitted to an unreliable third party.

For monitoring chronic diseases, the study in [178] established an IoT based architecture with homomorphic encryption to protect against data loss and spoofing assaults. The findings imply that homomorphic encryption offers simple, affordable protection for private health information. For the protection of medical data, block chain technologies are also used in conjunction with homomorphic encryption. In their article [179], the authors suggested combining block chain and homomorphic encryption in intelligent transportation systems with autonomous healthcare monitoring to track pandemic infections. In a different work [180], they used homomorphic encryption to create a searchable distributed medical database on a block chain. The necessity to protect sensitive information is growing, which encourages the integration of several strategies.

The next section provides a detailed explanation of homomorphic encryption.

5.3 Homomorphic Encryption

According to the problem analysis in Section 5.2 and the overview presented above, it is crucial to use a homomorphic encryption approach to safeguard the confidentiality of data.

How organisations and individuals use and manage their data has fundamentally changed thanks to affordable cloud computing and cloud storage. Data can be conveniently saved in encrypted form using traditional encryption techniques like AES, which are incredibly quick. However, the owner of the data must download, decode, and act on the encrypted data locally, which can be expensive and logistically challenging. Alternatively, the cloud server must have access to the secret key, raising security issues. Because the cloud may directly process the encrypted data and only provide the encrypted output to the data owner, homomorphic encryption can greatly simplify this scenario. In more complex application scenarios, many parties with private data may be involved [181]. In these cases, a third party may carry out an operation and then deliver the findings to one or more of the participants for decryption.

Homomorphic encryption is being developed to protect the security of information that must be kept private.

Operations can be carried out directly on encrypted data (cipher text) using homomorphic encryption techniques without needing access to the unencrypted data (plaintext). The party with the private key ‘Alice’ can encrypt its input and pass it to the party with the public key ‘Bob,’ who can do the necessary operations on the encrypted data alone if the homomorphic encryption scheme is asymmetric, i.e., offers public and private keys. Due to Bob's inability to decrypt and view Alice's input, operations on encrypted data protect privacy. The procedures used to protect privacy are built around this property.

A fully homomorphic cryptosystem is one in which any action on the plaintext may be performed by performing an operation on the corresponding cipher text (FHE). Gentry [182], [183] made the first such cryptosystem suggestion in a ground-breaking paper. The structure meets the requirements for FHE, but it is found to be computationally inefficient to be used in practise [184], and creating FHE schemes that are computationally feasible is an ongoing field of research [185].

Two characteristics of the homomorphic encryption technique are additive and multiplicative. An algorithm that can calculate Enc (X1+X2) from Enc(X1) and Enc(X2) without knowing the values of X1 and X2 is said to have an additive property in homomorphic encryption [186]. An algorithm with multiplicative properties can generate Enc(X1*X2) from Enc(X1) and Enc(X2) without needing to know the values of X1 and X2 [186].

The basic architecture of homomorphic encryption is shown in Figure 5.1.

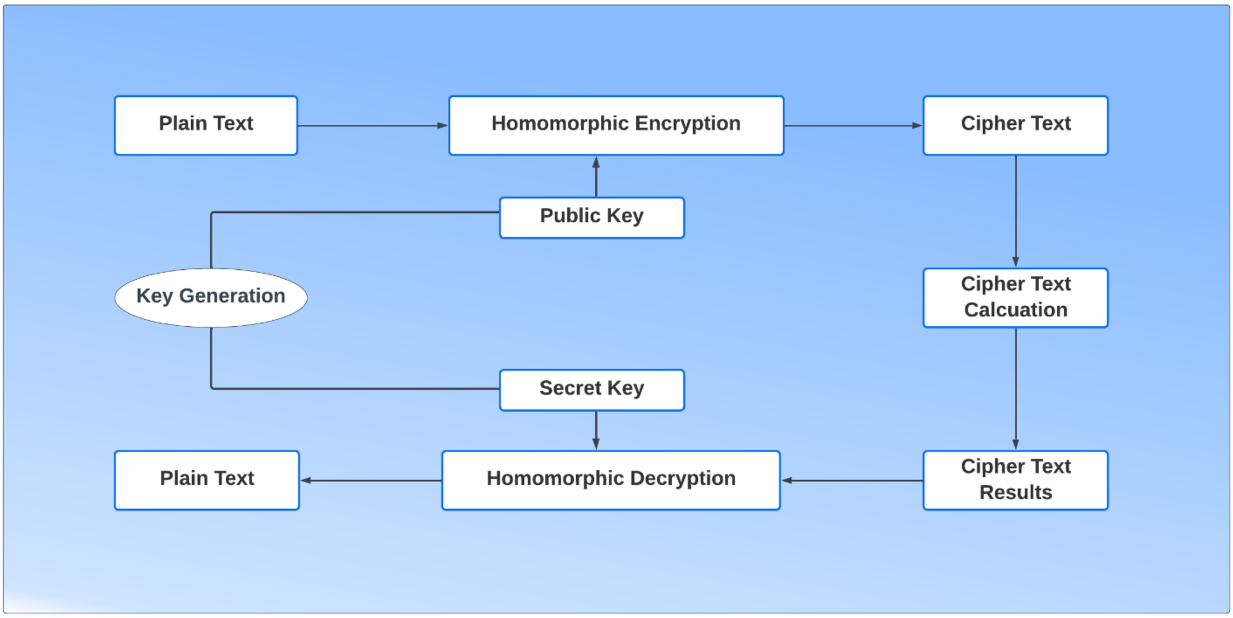


Figure 5.1 Homomorphic Encryption Process

The main algorithm utilised to represent our system is defined in the next section.

5.4 Proposed System

This section gives an overview of the proposed homomorphic encryption-based device identification system. To uniquely identify each device when data from different devices is generated and sent to our model, our model uses fully homomorphic encryption, which means that the data from these devices will always be in encrypted form throughout the computation. This makes the process of identifying devices safe and secure.

In our work, Microsoft Seal is leveraged, which is an open-source and highly optimised HE library developed by the Cryptography Research Group at Microsoft Research [187].

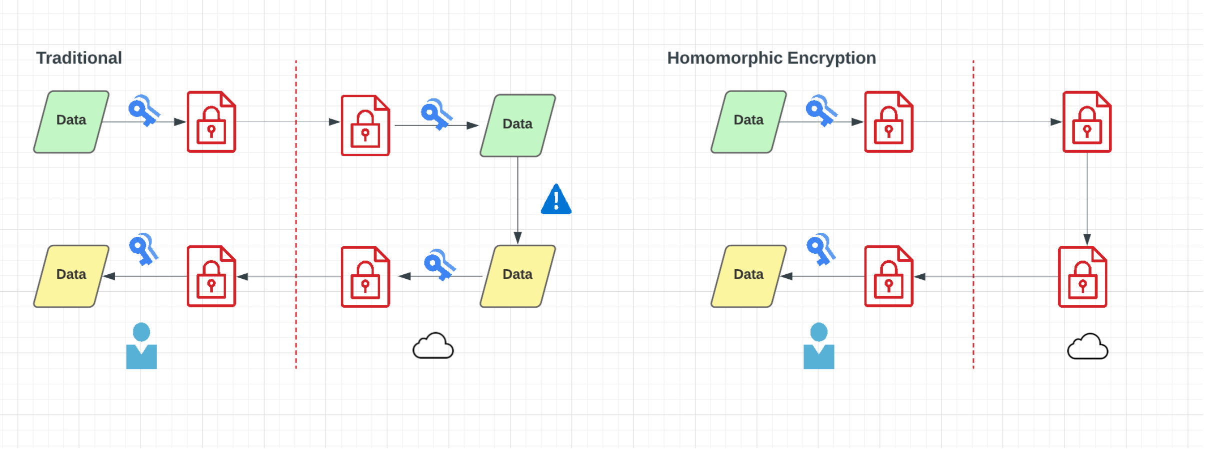


Figure 5.2 Traditional Encryption Vs Homomorphic Encryption

The BFV and CKKS algorithms are supported by Microsoft SEAL, which also enables additions and multiplications on encrypted integers or real numbers. Most of the time, using this technology, it is not possible to evaluate other operations on encrypted data, such as encrypted comparison, encrypted sorting, or regular expressions. Therefore, Microsoft SEAL should only be used to build cloud computation components of projects that require privacy. The BFV schemes allow modular arithmetic to be performed on encrypted integers [187].

This research work adopted the CKKS scheme proposed by Cheon et al. [188] [189], which supports the approximate calculation of real or complex numbers. Because data is typically represented by real numbers, the CKKS scheme, which can deal with real numbers, has received a lot of attention in a variety of applications, including evaluating machine learning models on encrypted data [190] [191] [192] [193]. Several studies have thus been carried out to optimise the CKKS scheme [194] [195] [196] [197] [198].

Figure 5.3 shows the step-by-step proposed system process. The proposed system consists of three main processes, which are described below: feature extraction, homomorphic encryption-based data protection, and device identification matching results.

5.4.1 Feature Extraction

This process has been explained in detail in Chapter 3, Section 3.3, where all data was pulled from devices and then used criteria for good features and shortlisted features that have high inter-sample variance and low intra-sample variance, and then normalised the data and split the data using k-fold (k = 10) into training and test data.

5.4.2 Homomorphic encryption-based data protection

The overview of our challenges with data protection, the solution and benefits of HE is explained in section 5.1. The fundamental challenge was limitation of test devices and as such, the test template data was kept on one main machine. To secure this crucial device data, while data computation was performed, a mechanism was required to address this concern. To achieve our goal of data protection from spoofing attacks, HE is leveraged, whereby test and training sample data is encrypted and, at the time of a cryptographic operation (for example, authentication), the encrypted data is used. Thus, the advantage is that spoofing and impersonation attacks can be avoided.

Microsoft offers a variety of APIs to aid in computation and support our model. The concept is briefed below.

The first step is to generate a key pair (public and private key) using:

[1] **get_seal ->** return the encryptor and encoder. Then encrypt training and test data using the public key [185].

[2] **compare_ciphertext ->** decrypt the ciphertext with private_key, compares it, and returns a Boolean, i.e., TRUE if ciphertext1 > cipherText2, False otherwise.

[3] **device_probability** -> decrypt the permutation test and training samples, and computes the mean, covariance and multivariate using the private_key and return the probability as a result.

The example below shows how get_seal interacts with data. The purpose of get_seal is to get an object from the Microsoft seal library to be used in encryption and decryption. get_seal consists of ckks_encoder, encryptor, scale, evaluator. To be able to encrypt, an instance of Encryptor needs to be constructed. The computations on the cipher texts are performed with the Evaluator class.

```
ckks_encoder = <seal.CKKSEncoder 0x7fa2a1a48470>

encryptor = <seal.Encryptor 0x7fa2a19ccb70>

evaluator = <seal.Evaluator 0x7fa2a199cc30>
```

Our model makes use of these APIs for data comparison on encrypted values. This helps in achieving data security since the data computation model is only known to work with encrypted values and is described in detail in Section 5.4.3.

5.4.2.1 Cipher Text API Comparison

Boolean comparison on cipher text is not straight forward and is not supported in the Microsoft SEAL library either; however, our probability model uses comparison while calculating probability based on input values. One way to implement comparison is to encrypt messages bit by bit and write a comparison circuit; however, this can be very inefficient from both a running time and data expansion point of view. So, our own logic has been enhanced and implemented to build this comparison circuit, which has enhanced the capability of our model to be more secure and robust.

Microsoft SEAL allows additions and multiplications to be performed on encrypted integers or real numbers. A comparison algorithm was built by leveraging the cipher text value additions capability, and our algorithm works as follows:

5.4.2.1.1 Comparison Algorithm

1. X and Y are two values that are encrypted. (In our research, this equates to cipher text data from 8 devices)
2. Produce a new random number M and encrypt it with EM. (Here, M is a random number to enhance security and EM is encrypted random number).
3. Perform the calculation result =X-Y+EM (In this step performs subtraction of cipher test and add encrypted random number).
4. Decrypt the result => R (Now the result is decrypted).
5. Subtract R- M from the result of the decryption. (This step simply subtracts the results from random number).
6. If (R- M) > 0; then X > Y; otherwise, Y > X. Lastly, we get the comparison done and conclude this data belongs to which device.

The purpose of each component is described below

Here is a compare cipher text example where two values, let's say X and Y, are encrypted.

X= cipher_text1

Y= cipher_text2

cipher_text1 = 0x7fa2a1a337b0

cipher_text2 = 0x7fa2a12a49b0

Afterward, generate random number M and encrypt it, i.e., r_encrypted.

r_encrypted = 0x7fa2a1ac3170

Consequently, the result is equal to X-Y+ EM.

result = 0x7fa2a1a28c30

The following step is to decrypt the result

plain = 0x7fa2a19a6af0

And the result from plaintext is an n-dimensional value as output.

Example: When X < Y

Assume x is 1000.1, y is 1999.1, and random Number is 16148.4.

r_plain = {Plaintext} <seal.Plaintext object at 0x7fde0994bab0>

r_encrypted = {Ciphertext} <seal.Ciphertext object at 0x7fde0992a670>

x_plain = {Plaintext} <seal.Plaintext object at 0x7fde0994a3b0>

x_encrypted = {Ciphertext} <seal.Ciphertext object at 0x7fde0999e870>

y_plain = {Plaintext} <seal.Plaintext object at 0x7fde099fecf0>

y_encrypted = {Ciphertext} <seal.Ciphertext object at 0x7fde099fc330>

result = {Ciphertext} <seal.Ciphertext object at 0x7fd5c2332030> (X-Y+RN)

plain = {Plaintext} <seal.Plaintext object at 0x7fd5c2326030>

output = 15149.4779 - 7151.8101

compare = {float64: ()} -998.9999999978227

ICMetrics specific flow is described below.

5.4.3 Device Identification Results

In the last stage, the results were compared based on the highest probability and used to identify the device uniquely.

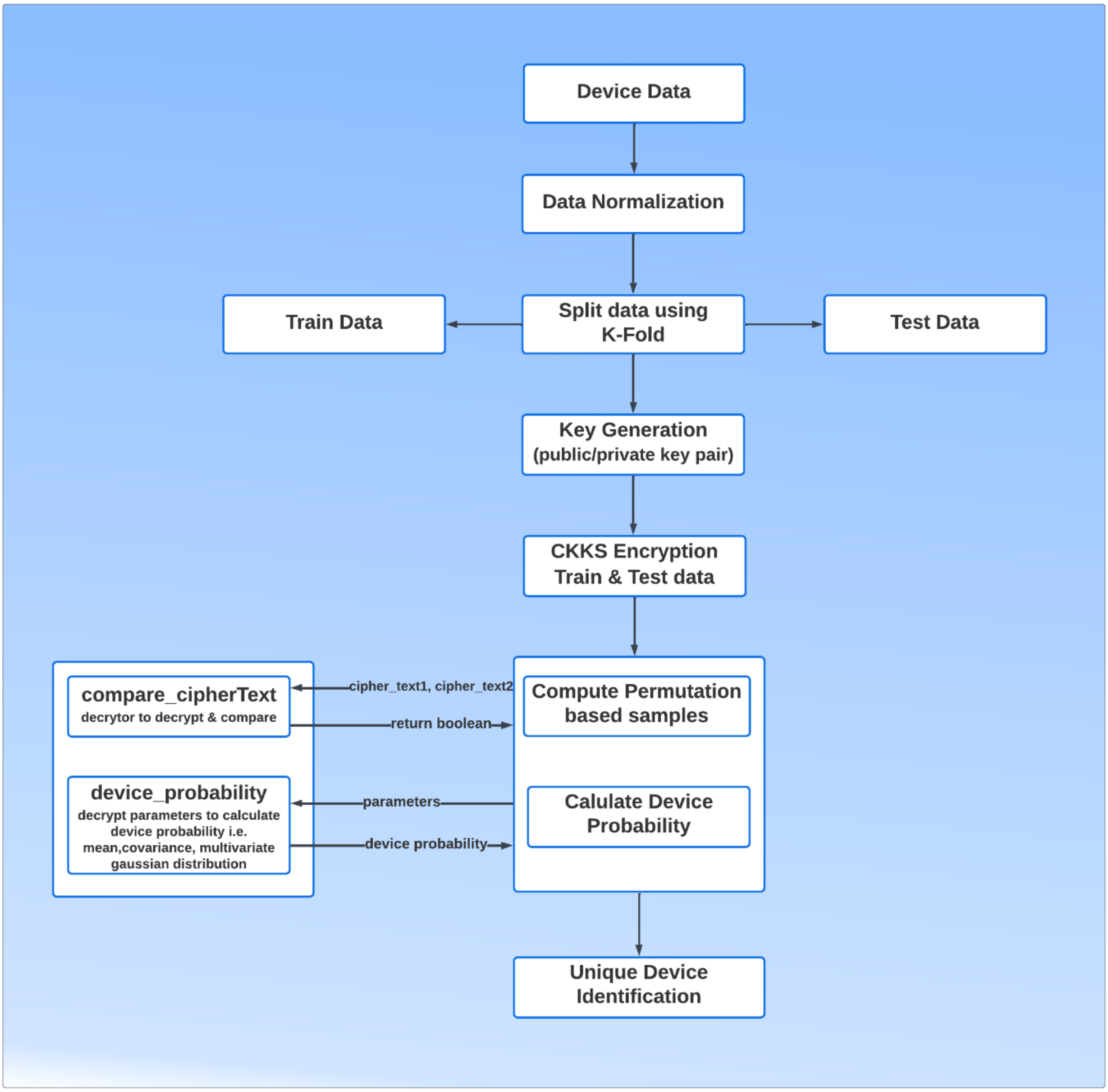

Device Data
Data Normalization
Train Data
Split data using K-Fold
Test Data
Key Generation (public/private key pair)
CKKS Encryption Train & Test data
compare_cipherText
decrytor to decrypt & compare
cipher_text1, cipher_text2
return boolean
Compute Permutation based samples
device_probability
decrypt parameters to calculate device probability i.e. mean,covariance, multivariate gaussian distribution
parameters
device probability
Calulate Device Probability
Unique Device Identification

Figure 5.3 Proposed System Process

5.4.4 Algorithm for the proposed system

The algorithms below introduce the process of a fully homomorphic encryption enabled model for device identification.

5.4.4.1 Algorithm

1. Read the data from all the devices.
2. Select features based on criteria for good features.
3. Normalize the data and then split it using k-fold.
4. Calculate the column threshold for training and testing and encrypt the threshold using CKKS FHE.
5. Encrypt test and training data.
6. Pass this encrypted training/test data to the model, which will return the device probability.
   A. The model will compute the permutation samples.
   B. Compute the device probability based on permutation samples after calculating the mean, covariance and multivariate Gaussian distribution.
7. Save the probability produced for each device.
8. Repeat the process for all devices to determine the maximum probability generated, which serves as our prediction data for device identification.

5.5 Experiment

The experiment's goal is to assess the proposed system for device identification using homomorphic encryption based on its accuracy performance.

5.5.1 Experiment Setup

The section offers a comprehensive examination of the experimental results connected to the suggested model, MVGD. The experimental dataset contains features discussed in Chapter 3 Section 3.3.3. The hardware of the MacBook Air and MacBook Pro served as the source of this data (memory, CPU and hard drive). Eight devices running current software were employed. This study made use of data acquired from the MacBook Air and Pro, Python programming, and Microsoft Excel. And each device has 1,000 samples for our research.

The experiment makes use of 17 features that were pre-processed, gathered from common computing devices, and then supplied to the model for training. The selected features were subsequently divided into sets to increase operational robustness. These feature sets offer more natural obfuscation, are more reliable than individual features, and generate a stronger base for applying the ICMetrics system. There are three feature sets: 8F,6F and 3F. The process of feature selection is explained in Chapter 3 (section 3.3.3).

For holdout accuracy estimations, the data were split into two sections, 80% and 20%, respectively. 20% was used for testing after 80% was used for training the model. For validating the results, k-fold cross-validation for k = 10 was employed. For k-fold cross validation, data was divided into k sections, one of which was used as test data and the remaining k-1 as train data.

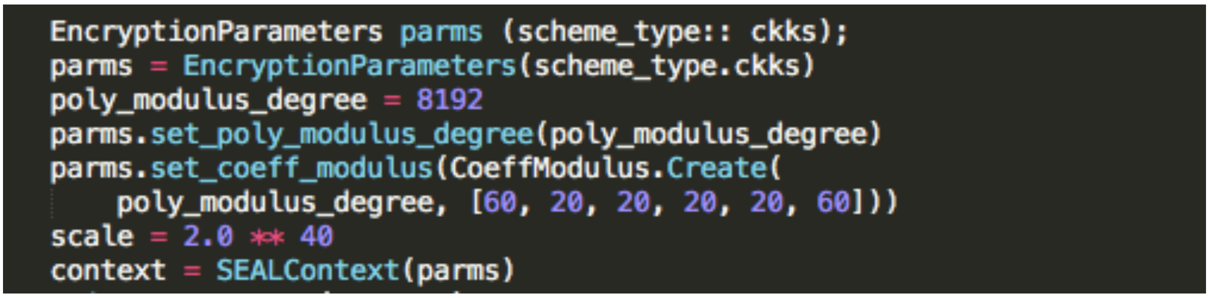
In the encryption stage of this experiment, the CKKS homomorphic encryption algorithm provided by the open-source Microsoft SEAL library [187] was utilised to encrypt the data.
Encryption parameters for CKKS are

•n: degree of polynomial modulus

•q: coefficient modulus

•scale: scaling factor for plaintext message inputs

SEAL generates all required parameters using these three parameters.



```
EncryptionParameters parms (scheme_type:: ckks);
parms = EncryptionParameters(scheme_type.ckks)
poly_modulus_degree = 8192
parms.set_poly_modulus_degree(poly_modulus_degree)
parms.set_coeff_modulus(CoeffModulus.Create(
    poly_modulus_degree, [60, 20, 20, 20, 20, 60]))
scale = 2.0 ** 40
context = SEALContext(parms)
```

Figure 5.4 Encryption Parameters

The first step in setting up the cryptosystem is to select the encryption parameters as outlined in Figure 5.4. An instance of the class Encryption Parameters contains them all. The three moduli that the encryption algorithm uses are first set: q (coefficient modulus), t (plain modulus), and Xn + 1 (polynomial modulus) – these are the three most vital criteria, and selecting them appropriately is essential for getting the best results.

5.5.2 Experimental Results

In this section, the device identification performance of the proposed system is evaluated over device feature data. In the proposed system, the encryption is performed on the feature vector. In Table 5.1 below, the time performance of the proposed system is demonstrated for polynomial modulus degree 4096 and 8192. This table captures key generation, test data encryption time, and overall computation time to uniquely identify the device for all three feature sets. It takes about 0.057s and 0.46 seconds to generate the public key and private key for 8F, and the other two feature sets have less key generation time because the other two feature sets contain fewer features than the first one. 8F test data encryption time (33.864s) and overall computation time (516.045s) are higher than the other two feature sets for polynomial modulus 4096 and 8F test data encryption time (50.698s) and overall computation time (6376.720s) are higher than the other two feature sets for polynomial modulus 8192. The conclusion is that a larger sample value means more information is preserved, which may lead to better device identification accuracy. However, the homomorphic encryption and decryption and key generation are the most time-consuming operations of the whole procedure. Therefore, a larger sample size means longer computational time, but data under homomorphic cipher text can have high privacy security.

Feature set	**Key Generation Time**	**Test Data Encryption Time**	**Overall computation time**
8F	0.057s	33.864s	516.045s
6F	0.055s	28.953s	302.858s
3F	0.053s	11.660s	70.393s

Table 5.1 Proposed system's time performance for polynomial modulus 4096 in various operations

Feature set	**Key Generation Time**	**Test Data Encryption Time**	**Overall computation time**
8F	0.463s	50.698s	6376.720s
6F	0.417s	32.934s	1502.186s
3F	0.400s	16.203s	263.586s

Table 5.2 Proposed system's time performance for polynomial modulus 8192 in various operations

The tables 5.3, 5.4, and 5.5 show results based upon device features for each feature set, respectively.

The effects of different parameters on the system's performance in terms of encryption and computational time are explored in the following tables.

Devices	Train Data Encryption Time	Computation Time
D0	44.275s	2897.953s
D1	50.160s	1085.930s
D2	40.913s	132.240s
D3	42.024s	91.805s
D4	38.562s	139.640s
D5	37.100s	1006.374s
D6	37.414s	98.050s
D7	37.039s	5831.506s

Table 5.3 8F HE based encryption and computation time for polynomial modulus 8192

Devices	Train Data Encryption Time	Computation Time
D0	28.946s	280.528s
D1	29.636s	196.820s
D2	27.341s	89.374s
D3	27.675s	72.152s
D4	26.612s	44.321s
D5	26.607s	87.765s
D6	26.553s	179.756s
D7	26.549s	297.567s

Table 5.4 6F HE based encryption and computation time for polynomial modulus 8192

Devices	Train Time	Computation Time
D0	14.079s	25.667s
D1	13.554s	20.272s
D2	13.554s	14.751s
D3	13.535s	13.807s
D4	13.601s	13.614s
D5	13.550s	22.755s
D6	13.534s	13.429s
D7	13.536s	13.412s

Table 5.5 3F HE based encryption and computation time for polynomial modulus 8192

As can be seen from Tables 5.3, 5.4, and 5.5, the average training encryption time for D0 is 44.275s, which is twice as high as 6F and three times higher than 3F because the multiplication on the cipher text requires more computation time, and in 8F, there is more data and it takes more time to execute. The computation time for D0 is 2897.953s which is higher than 6F (280.528s) and 3F (25.667s). This is the observation made when test data is verified against training data and test samples are mapped to modes. This multimodal feature process is explained in Chapter 3 (section 3.5). Three feature sets were used for our analysis, out of which FS1 (8F) train encryption and computation time are the highest as compared to the second and third feature sets. These feature sets include information about disk such as CPU performance and disk read and write operation speeds. This can be unique to different devices, resulting in the highest accuracy achieved from this feature set. Table 5.5 and 5.6 shows the HE-based percentage accuracy of 8F and 6F individual devices, respectively. The devices are individually identified according to the findings.

Devices	Accuracy
D0	91%
D1	90%
D2	90%
D3	96%
D4	90%
D5	88%
D6	89%
D7	90%

Table 5.6 8F HE based encryption-based devices accuracy

Devices	Accuracy
D0	90%
D1	88%
D2	91%
D3	90%
D4	90%
D5	92%
D6	92%
D7	94%

Table 5.7 6FHE based encryption-based devices accuracy

For 4096 polynomial modulus, tables 5.8, 5.9, and 5.10 provide the results for each feature set based on device features. The following section analysis the effects of various parameters on how well the system performs in terms of encryption and processing time.

Devices	**Train Data Encryption Time**	**Computation Time**
D0	45.204s	68.683s
D1	79.464s	231.716s
D2	241.678s	247.940s
D3	258.281s	264.952s
D4	274.792s	280.661s
D5	290.416s	364.238s
D6	374.039s	386.390s
D7	396.124s	516.044s

Table 5.8 8F HE based accuracy and computation time for polynomial modulus 4096

Devices	**Train Data Encryption Time**	**Computation Time**
D0	42.431s	98.254s
D1	106.045s	134.343s
D2	147.258s	162.582s
D3	175.875s	190.557s
D4	198.351s	205.185s
D5	212.867s	219.770s
D6	227.486s	247.497s
D7	255.253s	302.858s

Table 5.9 6F HE based accuracy and computation time for polynomial modulus 4096

Devices	**Train Time**	**Computation Time**
D0	15.939s	21.004s
D1	25.300s	27.887s
D2	32.177s	34.828s
D3	39.168s	41.709s
D4	45.994s	48.542s
D5	52.813s	57.575s
D6	61.615s	63.941s
D7	67.967s	70.393s

Table 5.10 3F HE based accuracy and computation time for polynomial modulus 4096

The average training encryption time for D0 is 45.204s, as shown in Tables 5.8, 5.9, and 5.10. This is longer than for 6F and three times longer than for 3F because the multiplication on the cipher text requires more computing time, and since there is more data and thus takes longer to execute in 8F. This finding results from comparing test data to training data and matching test samples to modes. The highest accuracy from this feature set can be attained because this is sometimes specific to distinct devices.

From the experiment results, the observation is that the proposed model using CKKS-based HE takes longer for computation depending on how many features and samples are factored, as shown in the above results. Our primary objective in this chapter is to ensure data protection during the process of analysis and device identification.

5.6 Conclusion

In this chapter, an architecture and an implementation of a device identification system in the HE domain were presented and subsequently evaluated experimentally. The system fulfils the data protection objectives. A cryptographic technique called CKKS homomorphic encryption was executed to secure the device data and to uniquely identify the device. Our proposed model has multimodal features, and for our analysis, comparison was used to identify the devices. So, our own logic was implemented to build this comparison circuit and calculated all the parameters required in our model to predict the results (in Section 5.3) which enhanced the capability of our model to be more secure and robust. By utilising HE, the security objectives of a dataset are achieved.

By using CKKS homomorphic encryption, the same accuracy results were achieved as discussed in Chapter 3 (section 3.8). The computational time and device identification accuracy are studied in this chapter. According to the experimental results, homomorphic encryption is time consuming. Efficient homomorphic encryption algorithms should be explored as future work to reduce the computational time so as to accelerate the deployment of device authentication using homomorphic encryption in some real applications.

Chapter 6

Conclusions and Future Work

6.1 Introduction

Nowadays, users are increasingly reliant on exchanging data, and they believe that their devices and data are secure from intruders. Therefore, it is the designers and manufacturers job to make sure that this is the case. Currently, a majority of IoT compatible devices are being developed and produced with minimal to no security features.
This thesis investigates using ICMetrics technology as a foundation for securing devices. The contributions that were made with the intention of revolutionising device security are summarised in this chapter.

The study's goal was to investigate and evaluate potential features for uniquely identifying the device. Many components of a conventional pattern recognition system were investigated and put into practise to achieve this goal.

Candidate features had to be found because feature selection is a crucial part of pattern recognition systems. A group of features that needed further investigation in order to produce performance measures was discovered through rigorous research. These performance indicators were then applied to determine whether each feature was suitable for deployment in an implementation. Further study was done on features as well as ways to pre-process current features to enhance their functionality. Performance measures from statistical tests were used to evaluate the acceptability of each feature on its own.

There was no set formula for feature selection. Each feature's best performing variation was compared, and the best features were ranked in accordance. The viability of a realistic implementation was then tested using a subset of these best feature vectors for device identification techniques.

The novelty of this work is to enhance device identification using the physical and behavioural characteristics of general computing devices. Our attention was on keeping track of device activity while collecting the data in a monitored environment. This helps to explain how the features behaved during the analysis. After analysis, our reflection was that feature data is multimodal in nature. The multimodal feature data was then used to a) determine the best way to precisely verify a device uniquely; b) analyse the device's stability in a variety of usage circumstances and its performance endurance; c) investigate the potential in the frequency domain; d) evaluate device authentication using homomorphic encryption.

In the next section, our research contributions are presented. Following this, the direction of future study in this field is given.
Section 6.2 describes research contributions and Section 6.3 finally discusses future works.

6.2 Research Contributions

The main objective of this research was to examine the possibility of identifying individual devices based on the features extracted from general computing devices. The secondary objective was to investigate the effectiveness of employing measured hardware features mapped into the frequency domain for device identification. And the third focus was to use homomorphic encryption for template data protection from spoofing attack.

The goals are accomplished using a method known as ICMetrics (Integrated Circuit Metrics), which is based on extracting traits and behaviours from general computing devices and using them to identify devices uniquely. The ICMetrics technology enables the construction of a device identifier by utilising the properties of the device. The device identification is then used for authentication. Because a key is only generated when needed and then deleted, ICMetrics technology acts as a deterrent to key theft. As a result, there is nothing for an enemy to steal.

In chapter 2, various security requirements are discussed. Following that, the current start of the art in authentication techniques explored. Then our focus was on background of ICMetrics, related works, extensive competitive analysis, and a summary of many different types of authentication systems was explored. Finally, discusses cybersecurity physical attacks on devices and summarized the chapter.

At the start of Chapter 3, criteria for good features are explained, and then hardware device features are used for ICMetrics device identification. Thus, the first contribution of this thesis is a thorough examination of the explicit and implicit features of a device that can be utilised to generate ICMetrics. According to a statistical examination of the feature values, each feature's data has a distinct bias because evaluation was based upon three categories of features: hard disk performance, memory, and CPU floating-point performance. A total of 38 features were collected and analysed. Analysis revealed that features are multimodal in nature. The conventional pattern recognition system has difficulty utilising unusual features. The approach converts the uncommon features into conventional forms that are simple to compute, such as the Gaussian distribution. On the original distributions, feature values are rearranged into a predetermined section. To solve multimodal features, a peak-trough algorithm approach was used to determine the number of modes in a distribution. Following that, the initial distributions were divided into distinct model features. In the multi-dimensional space, each separated distribution was considered a new feature. By addressing the multimodality of the features, the second issue was tackled, which is feature overlap. By combining the modes, the link between the modes of features (these are exclusive to each device) can be seen, which enables the differentiation of data overlap between devices for each feature.

Multi-modal features were used for device identification. Our proposed classifier and multiple conventional classifiers were benchmarked. To construct the model and conduct evaluations, the assessment results showed how employing different categories of feature data affected the performance of the identification. The results obtained for features related to writing to disk and memory-related features showed high accuracy for device identification as compared to other categories of features.

Chapter 4 of the thesis introduces a novel wavelet feature based multivariate Gaussian distribution classifier framework. The use of hardware features in the frequency domain to identify devices uniquely is the second contribution of this thesis. This chapter starts with the introduction of wavelets. For our analysis, the discrete wavelet transform was used because of the advantages it has, including the removal of noise from the data, the detection of abrupt disruptions, the reduction of large amounts of data, and most importantly, the DWT's increased computing efficiency due to the production of fewer coefficients. Then it describes a system overview, which explains the process of using hardware features to calculate the coefficients and then using these coefficients as new features for our analysis. And at the end, extensive wavelet results were produced, and compared the results based upon hardware features. At the end, conclude that wavelet-based feature results are better than raw feature-based results.

In Chapter 5, the concept of homomorphic encryption (HE) for device identification has been presented. In the HE domain, a device identification system's architecture and implementation were presented and then experimentally tested. In our research, the comparison is to identify the devices, and our proposed model contains multimodal features. In order to construct this comparison algorithm, our own logic was built and calculated all the necessary parameters needed to predict outcomes. The goal of data security is met by using HE. Hence the third contribution of this thesis is to highlight the usage of encrypted data for device identification via homomorphic encryption.

In Chapter 6, thesis was summarised and offered some recommendations for further research in this area.

6.3 Future Works

By utilising a variety of explicit and implicit device features, the thesis has shown how to create an ICMetrics. Although ICMetrics is a developed technology, there are still some areas where it might be improved upon and where additional research might be conducted. Some of the areas for further investigation are highlighted in this section. The identification of additional features that can improve the security strength of the ICMetrics system should therefore be the focus of future research.

The future work can include: -

1) Large number of devices and models. Our current constraints prevent the collection of data from a large number of devices. In the future, it will be crucial to scale the number of devices and analyse if the technology can identify a unique device in a very large population. Future system scaling will also require a more advanced data analysis solution.
2) User features need to be explored. The current set of features in the research are

hardware and software based. More user dependent features, for example, user interaction with a keyboard or touch screen, etc., can greatly raise the security level.

3) For existing multi-modal features, a more complex normalisation technique can be explored to enhance the performance of the system.
4) Explore how ICMetrics can be leveraged to enhance mobile device security, for example, Android and iOS. Can ICMetrics be used to uniquely identify mobile phones based on data obtained from the devices.

According to the study cited in this thesis, devices can be identified by using attributes and traits that are gathered from the devices themselves. The suggested system's effectiveness demonstrates that it has met the essential criteria. For a more secure environment, it is envisaged that this method will be enhanced in the future.

References

[1] Statista. (n.d.). Identity access management spending worldwide 2021. (IAM)

[2] Bartock, M., Souppaya, M., Cherfaoui, M., Xie, J. and Cleary, P. (2022). Hardware-Enabled Security: Machine Identity Management and Protection.

[3] Van Aubel, P., Bernstein, D.J. and Niederhagen, R. (2015b). Investigating SRAM PUFs in large CPUs and GPUs.

[4] CQURE Academy. (2016). Decrypting SID-protected PFX Files Without Having a Password. [online] Available at: https://cqureacademy.com/blog/windows-internals/decrypting-sid-protected-pfx-files-without-password [Accessed 26 Jun. 2021].

[5] Pegasus and surveillance spyware Policy Department for Citizens' Rights and Constitutional Affairs Directorate-General for Internal Policies PE. (2022). Available at: https://www.europarl.europa.eu/RegData/etudes/IDAN/2022/732268/IPOL_IDA(2022)732268_EN.pdf.

[6] 'Meltdown and Spectre,' Meltdownattack.com, 2013.

[7] D. Goodin, 'Plundering of crypto keys from ultrasecure SGX sends Intel scrambling again,' Ars Technica, Jun. 09, 2020.

[8] Kovalchuk, Y., McDonald-Maier, K. and Howells, G. (2011). Overview of ICMetrics Technology – Security Infrastructure for Autonomous and Intelligent Healthcare System. International Journal of u- and e- Service, Science and Technology, [online] 4, pp.49–60.

[9] Sh. Tahir and M. Afzal, 'An ICMetrics based Key Generation Scheme for Controlled Group Communication,' IISA 2014, 5th Int. Conf. Information, Intell. Syst. Appl. Chania, 2014, pp. 373–378, 2014.

[10] X. Zhai, K. Appiah, S. Ehsan, and M. Wah, 'A Self-Organising Map Based Algorithm for Analysis of ICMetrics Features,' Fourth Int. Conf. Emerg. Secur. Technol., pp. 93–97, 2013.

[11] Ratcliff, C. (2014). More than 40% of online adults are multi-device users: stats. [online] Econsultancy. Available at: https://econsultancy.com/more-than-40-of-online-adults-are-multi-device-users-stats/.

[12] Department for Digital, Culture, Media & Sport (2022). Cyber Security Breaches Survey 2022. [online] GOV.UK. Available at: https://www.gov.uk/government/statistics/cyber-security-breaches-survey-2022/cyber-security-breaches-survey-2022.

[13] Verizon (2022). 2022 Data Breach Investigations Report. [online] Verizon Business. Available at: https://www.verizon.com/business/resources/reports/dbir/.

[14] W. Stallings, Cryptography and Network Security: Principles and Practice, 5th ed. Stallings William, 2011.

[15] M. Alshahrani and H. Teymourlouei, 'Network Security: Threats and Vulnerabilities,' Int'l Conf. Secur. Manag., pp. 115–121, 2016.

[16] Helfmeier, C., Boit, C., Nedospasov, D. and Seifert, J.-P. (2013). Cloning Physically Unclonable Functions. Katzenbeisser, S., Kocabaş, Ü., Rožić, V., Sadeghi, A.-R., Verbauwhede, I. and Wachsmann, C. (2012). PUFs: Myth, Fact or Busted? A Security Evaluation of Physically Unclonable Functions (PUFs) Cast in Silicon. Cryptographic Hardware and Embedded Systems – CHES 2012, pp.283–301.

[17] Ruhrmair, U. and van Dijk, M. (2013). PUFs in Security Protocols: Attack Models and Security Evaluations. 2013 IEEE Symposium on Security and Privacy.

[18] Schuster, D. (2011). Side-Channel Analysis of Physical Unclonable Functions (PUFs).

[19] Van Aubel, P., Bernstein, D.J. and Niederhagen, R. (2015). Investigating SRAM PUFs in large CPUs and GPUs. Security, Privacy, and Applied Cryptography Engineering, pp.228–247.

[20] Anagnostopoulos, N.A., Arul, T., Rosenstihl, M., Schaller, A., Gabmeyer, S. and Katzenbeisser, S. (2018). Low-Temperature Data Remanence Attacks Against Intrinsic SRAM PUFs.

[21] Goodin, D. (2017). Millions of high-security crypto keys crippled by newly discovered flaw.

[22] Butterworth, J. and Kallenberg, C. (2013). Problems with the Static Root of Trust for Measurement.

[23] (PDF) trusted platform module – A survey - researchgate (no date). Available at: https://www.researchgate.net/publication/287984174_Trusted_Platform_Module_-_A_Survey (Accessed: November 30, 2022). 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.

[24] 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.

[25] 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.

[26] Zhao, C., Zhao, S., Zhao, M., Chen, Z., Gao, C.-Z., Li, H. and Tan, Y. (2019). Secure Multi-Party Computation: Theory, practice and applications. Information Sciences, 476, pp.357–372.

[27] Anagnostopoulos, N.A., Arul, T., Rosenstihl, M., Schaller, A., Gabmeyer, S. and Katzenbeisser, S. (2018). Low-Temperature Data Remanence Attacks Against Intrinsic SRAM PUFs.

[28] Yinghui, P. (2009). The Application of PKCS#12 Digital Certificate in User Identity Authentication System. [online] IEEE Xplore. doi:10.1109/WCSE.2009.202.

[29] Goodin, D. (2017). Millions of high-security crypto keys crippled by newly discovered flaw.

[30] 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.

[31] 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.

[32] 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.

[33] 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.

[34] 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.

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

[36] Yadav, S. and Howells, G. (2017). Analysis of ICMetrics features/technology for wearable devices IoT sensors. [online] IEEE Xplore. doi:10.1109/EST.2017.8090419.

[37] Baba, S.D., Yadav, S. and Howells, G. (2019). SortAlgo-Metrics: Identification of Cloud-Based Server Via a Simple Algorithmic Analysis. [online] IEEE Xplore. doi:10.1109/EST.2019.8806214.

[38] Yadav, S. and Howells, G. (2019). Secure Device Identification Using Multidimensional Mapping. [online] IEEE Xplore. doi:10.1109/EST.2019.8806218.

[39] Yadav, S., Khanna, P.R. and Howells, G. (2022). Device Authentication Using Wavelet Based Features. International Journal for Information Security Research, [online] 12(1), pp.1062–1072. doi:10.20533/ijisr.2042.4639.2022.0120.

[40] Yadav, S., Khanna, P.R. and Howells, G. (2021). Device Identification Using Discrete Wavelet Transform. [online] IEEE Xplore. doi:10.1109/ICEET53442.2021.9659553.

[41] Zhai, X., Appiah, K., Ehsan, S., Hu, H., Gu, D., McDonald-Maier, K., Cheung, W.M. and Howells, G. (2013). Application of ICMetrics for Embedded System Security. 2013 Fourth International Conference on Emerging Security Technologies.

[42] E. Papoutsis, 'Investigation of the Potential of Generating Encryption Keys for ICMETRICS,' _University of Kent, 2009.

[43] S. Tahir and I. Rashid, 'ICMetric-Based Secure Communication,' _in Innovative Solutions for Access Control Management, vol. 36, IGI Global, 2016, pp. 263–293.

[44] Papoutsis, E., Howells, G., Hopkins, A. and McDonald-Maier, K. (2009). Ensuring Secure Healthcare Communications via ICmetric Based Encryption on Unseen Devices. [online] IEEE Xplore. doi:10.1109/BLISS.2009.24.

[45] Zhai, X., Appiah, K., Ehsan, S., Howells, G., Hu, H., Gu, D. and McDonald-Maier, K.D. (2015b). A Method for Detecting Abnormal Program Behavior on Embedded Devices. IEEE Transactions on Information Forensics and Security, [online] 10(8), pp.1692–1704. doi:10.1109/tifs.2015.2422674.

[46] Kovalchuk, Y., Hu, H., Gu, D., McDonald-Maier, K. and Howells, G. (2012). ICMetrics for Low Resource Embedded Systems. 2012 Third International Conference on Emerging Security Technologies. [online] Available at: https://www.academia.edu/12000222/ICmetrics_for_Low_Resource_Embedded_Systems [Accessed 30 Nov. 2022].

[47] F. Diez, D. Touceda, J. M. S. Camara, and S. Zeadally, 'Toward self-authenticable wearable devices,' _IEEE Wirel. Commun., vol. 22, no. 1, pp. 36–43, Feb. 2015.

[48] Divya, J. and Shivagami, S. (2020). A study of Secure cryptographic based Hardware security module in a cloud environment. [online] IEEE Xplore. doi:10.1109/I-SMAC49090.2020.9243328

[49] Din, A. (2019). Biometric Authentication Overview, Advantages & Disadvantages [Updated 2019]. [online] Heimdal Security Blog. Available at: https://heimdalsecurity.com/blog/biometric-authentication/.

[50] ComputerWeekly.com. (n.d.). Limitations of two factor authentication (2FA) technology. [online] Available at: http://www.computerweekly.com/tip/Limitations-of-two-factor-authentication-2FA-technology [Accessed 30 Nov. 2022].

[51] Yaacoub, J.-P.A., Salman, O., Noura, H.N., Kaaniche, N., Chehab, A. and Malli, M. (2020). Cyber-physical systems security: Limitations, issues and future trends. Microprocessors and Microsystems, 77, p.103201. doi:10.1016/j.micpro.2020.103201.

[52] Singh, S., Yadav, N. and Chuarasia, P.K. (2020). A Review on Cyber Physical System Attacks: Issues and Challenges. [online] IEEE Xplore. doi:10.1109/ICCSP48568.2020.9182452.

[53] Weingart, S.H. (2000). Physical Security Devices for Computer Subsystems: A Survey of Attacks and Defenses. Cryptographic Hardware and Embedded Systems — CHES 2000, pp.302–317. doi:10.1007/3-540-44499-8_24.

[54] S. H. Weingart, ‘Physical Security Devices for Computer Subsystems: A Survey of Attacks and Defenses,’ _Springer Berlin Heidelberg, 2000, pp. 302–317.

[55] F. Koeune and F.-X. Standaert, A Tutorial on Physical Security and Side-Channel Attacks. Springer Berlin Heidelberg, 2005.

[56] V. Pasupathinathan, ‘Hardware-Based Identification and Authentication Systems,’ _Macquarie University, 2009.

[57] ISO 13491-1 (1998). Banking—Secure cryptographic devices (retail), Part 1: Concepts, requirements and evaluation methods (1st ed.).

[58] A. G. Mason and M. J. Newcomb, Cisco secure Internet security solutions. Cisco Press, 2001.

[59] C. V. Anchugam et al., ‘Classification of Network Attacks and Countermeasures of Different Attacks,’ _in Network Security Attacks and Countermeasures, vol. 50, no. 1, IGI Global, 2016, pp. 115–156.

[60] S. Yu, Distributed denial of service attack and defense. Springer-Verlag, 2014

[61] D. K. Bhattacharyya and J. K. Kalita, DDoS attacks : evolution, detection, prevention, reaction, and tolerance. CRC Press, 2016.

[62] Li, X., Wang, H., Dai, H.-N., Wang, Y. and Zhao, Q. (2016). An Analytical Study on Eavesdropping Attacks in Wireless Nets of Things. Mobile Information Systems, 2016, pp.1–10. doi:10.1155/2016/4313475.

[63] Agreement - UNECE (no date). Available at: https://unece.org/sites/default/files/2021-03/R155e.pdf

[64] El-Rewini, Z., Sadatsharan, K., Selvaraj, D.F., Plathottam, S.J. and Ranganathan, P. (2020). Cybersecurity challenges in vehicular communications. Vehicular Communications, 23, p.100214. doi:10.1016/j.vehcom.2019.100214.

[65] D. Genkin, L. Pachmanov, I. Pipman, A. Shamir, and E. Tromer, ‘Physical key extraction attacks on PCs,’ _Commun. ACM, vol. 59, no. 6, pp. 70–79, Jun. 2016.

[66] J. A. Halderman et al., ‘Lest We Remember: Cold-Boot Attacks on Encryption Keys,’ _Communications of the ACM, vol. 52, no. 5, ACM, p. 91, 01-May-2009.

[67] I. Kizhatov, ‘Physical Security of Cryptographic Algorithm Implementations,’ _Universite Du Luxembourg, 2011.

[68] A. Cullen and L. Armitage, ‘The social engineering attack spiral (SEAS),’ _in 2016 International Conference On Cyber Security And Protection Of Digital Services (Cyber Security), 2016, pp. 1–6.

[69] A. Rae and L. Wildman, ‘A Taxonomy of Attacks on Secure Devices,’ _in Proceedings of the Australia Information Warfare and Security Conference 2003, 2003, pp. 251–264.

[70] M. Gupta, J. Walp, and R. Sharman, Threats, Countermeasures and Advances in Applied Information Security. IGI Publishing, 2012.

[71] Gohwong, S.G. (2019). The State of the Art of Cryptography-Based Cyber-Attacks. [online] papers.ssrn.com. Available at: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3546334.

[72] Cangea, O. and Moise, G. (2011). A New Approach of the Cryptographic Attacks. Communications in Computer and Information Science, pp.521–534. doi:10.1007/978-3-642-21984-9_44.

[73] A. J. Mills, R. T. Watson, L. Pitt and J. Kietzmann, ‘Wearing Safe: Physical andInformational Security in the Age of the Wearable Device,’ Business Horizons,vol. 59, no. 6, pp. 615-622, 2016.

[74] K. W. Ching and M. M. Singh, ‘Wearable Technology Devices Security andPrivacy Vulnerability Analysis,’ International Journal of Network Security & ItsApplications (IJNSA), vol. 8, no. 3, pp. 19-30, 2016.

[75] VIPRE. (n.d.). 8 Security Threats Wearables Pose to Companies and Individuals. [online] Available at: https://vipre.com/blog/8-security-threats-wearables-pose-companies-individuals/ [Accessed 29 Nov. 2022].

[76] A. Muro-de-la-Herran, B. García-Zapirain, and A. Méndez-Zorrilla, ‘Gait analysis methods: An overview of wearable and non-wearable systems, highlighting clinical applications,’ _Sensors, vol. 14, no. 2, pp. 3362–3394, 2014.

[77] S. Sprager and M. B. Juric, ‘Inertial sensor-based gait recognition: A review,’ _Sensors, vol. 15, no. 9, pp. 22089–22127, 2015.

[78] J. Chauhan, H. J. Asghar, M. A. Kâafar, and A. Mahanti, ‘Gesture-based Continuous Authentication for Wearable Devices: the Google Glass Case.,’ _in 14th International conference on Applied Cryptography and Network Security, 2016, pp. 1–28.

[79] A. Bianchi and I. Oakley, ‘Wearable Authentication: Trends and Opportunities,’it - Information Technology,

vol. 58, no. 5, pp. 255-262, 2016.

[80] J. Leonard, ‘Wearable Product Security: What you need to know,’https://blog.nordicsemi.com/getconnected/wearable-product-security-what-you-need-to-know

[81] P. Chen, L. Desmet and C. Huygens, ‘A Study on Advanced Persistent Threats,’B.De Decker and A. Z´uquete (Eds.): CMS 2014, LNCS, vol. 8735, pp. 63-72,2014.

[82] S. Kaur, ‘How to Secure Our Bluetooth Insecure World!’ IETE TechnicalReview, vol. 30, no. 2, pp. 95-101, 2013.

[83] M. Rahman, B. Carbunar and M. Banik, ‘Fit and Vulnerable: Attacks andDefenses for a Health Monitoring Device,’ https://arxiv.org/abs/1304.5672

[84] M. Rahman, B. Carbunar, and M. Banik, ‘Fit and Vulnerable: Attacks and Defenses for a Health Monitoring Device,’ _2013.

[85] Security Research Report on Mercedes-Benz Cars. (n.d.). [online] Available at: https://skygo.360.net/archive/Security-Research-Report-on-Mercedes-Benz-Cars-en.pdf.

[86] 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.

[87] GOV.UK. (2022). Cyber Security Breaches Survey 2022. [online] Available at: https://www.gov.uk/government/statistics/cyber-security-breaches-survey-2022/cyber-security-breaches-survey-2022.

[88] get.hypr.com. (n.d.). State of Authentication in the Finance Industry 2022 | HYPR. [online] Available at: https://get.hypr.com/state-of-authentication-in-the-finance-industry-2022?_ga=2.110607106.683219659.1667577955-1069184861.1667577955 [Accessed 4 Nov. 2022].

[89] ISO 13491-1 (1998). Banking—Secure cryptographic devices (retail), Part 1: Concepts, requirements and evaluation methods (1st ed.).

[90] 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.

[91] 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.

[92] 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.

[93] 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.

[94] Zhai, X., Appiah, K., Ehsan, S., Hu, H., Gu, D., McDonald-Maier, K., Cheung, W.M. and Howells, G. (2013). Application of ICMetrics for Embedded System Security. 2013 Fourth International Conference on Emerging Security Technologies.

[95] Intel. (2013). Optimizing Memory Bandwidth on Stream Triad. [online] Available at: https://www.intel.com/content/www/us/en/developer/articles/technical/optimizing-memory-bandwidth-on-stream-triad.html [Accessed 4 Nov. 2022].

[96] The Block I/O Layer. (n.d.). [online] Available at: http://sylab-srv.cs.fiu.edu/lib/exe/fetch.php?media=paperclub:lkd3ch14.pdf [Accessed 4 Nov. 2022].

[97] www.sciencedirect.com. (n.d.). Floating-Point Operation - an overview | ScienceDirect Topics. [online] Available at: https://www.sciencedirect.com/topics/computer-science/floating-point-operation.

[98] Huawei Enterprise Support Community. (n.d.). What is IOPS (input/output operations per second)? [online] Available at: https://forum.huawei.com/enterprise/en/what-is-iops-input-output-operations-per-second/thread/872689-891 [Accessed 4 Nov. 2022].

[99] 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.

[100] 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.

[101] 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.

[102] 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.

[103] 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

[104]K. Harmer, G. Howells, W. Sheng, M. Fairhurst and F. Deravi,'A Peak-Trough Detection Algorithm Based on Momentum,' 2008 Congress on Image and Signal Processing, Sanya, China, 2008, pp. 454-458.

[105]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.

[106]Ye, B., Howells, G., Haciosman, M. and Wang, F. (2015). Multi-dimensional key generation of ICMetrics for cloud computing. Journal of Cloud Computing, 4(1). doi:10.1186/s13677-015-0044-6.

[107]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.

[108]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.

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

[110]Anon, (2021). Min-Max Normalization - Machine Learning Concepts. [online] Available at: https://ml-concepts.com/2021/10/08/min-max-normalization/.

[111]D. Reynolds, 'Gaussian Mixture Models,' in Encyclopedia of Biometrics, 2015, pp. 827–832.

[112]A. Singh, N. Thakur and A. Sharma, 'A review of supervised machine learning algorithms,' 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 2016, pp. 1310-1315.

[113]S. Ray, 'A Quick Review of Machine Learning Algorithms,' 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon), Faridabad, India, 2019, pp. 35-39.

[114]S. Sprager and M. B. Juric, 'Inertial sensor-based gait recognition: A review,' _Sensors, vol. 15, no. 9, pp. 22089–22127, 2015.

[115] A. Singh, N. Thakur and A. Sharma, 'A review of supervised machine learning algorithms,' 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 2016, pp. 1310-1315.

[116]S. Aljawarneh, M. Aldwairi, and M. B. Yassein, 'Anomaly-based intrusion detectio system through feature selection analysis and building hybrid efficient model,' Journal of Computational Science, vol. 25, pp. 152–160, Mar. 2018.

[117]S. Ray, 'A Quick Review of Machine Learning Algorithms,' 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon), Faridabad, India, 2019, pp. 35-39.

[118] S. Aljawarneh, M. Aldwairi, and M. B. Yassein, 'Anomaly-based intrusion detectio system through feature selection analysis and building hybrid efficient model,' Journal of Computational Science, vol. 25, pp. 152–160, Mar. 2018.

[119]X. Zhai, K. Appiah, S. Ehsan, H. Hu, D. Gu, K. McDonald-Maier, W. M. Cheung, and G. Howells, 'Application of ICMetrics for Embedded System Security,' in 2013 Fourth International Conference on Emerging Security Technologies, 2013, pp. 89–92.

[120]Y. Kovalchuk, K. McDonald-Maier, and G. Howells, 'Overview of ICMetrics Technology – Security Infrastructure for Autonomous and Intelligent Healthcare System,' Int. J. u- e- Serv. Sci. Technol., vol. 4, no. 3, pp. 49–60, 2011.

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

[122]R. Tahir, H. Tahir, and K. McDonald-Maier, 'Securing health sensing using integrated circuit metric,' Sensors (Switzerland), vol. 15, no. 10, pp. 26621–26642, 2015.

[123] Maxim Integrated Products, 'DS2411 Silicon Serial Number with VCC Input,' Sunnyvale, 2011.

[124]Sridhar, S., Rajesh Kumar, P. and Ramanaiah, K.V. (2014). Wavelet Transform Techniques for Image Compression – An Evaluation. International Journal of Image, Graphics and Signal Processing, 6(2), pp.54–67.

[125]S. B. Park, J. W. Lee and S. K. Kim, 'Content based image classification using a neural network', Pattern Recognition Letters, Vol. 25, No. 3, pp. 287-300, 2004.

[126]N. S. Manish, M. Bodruzzaman and M. J. Malkani, 'Feature Extraction using Wavelet Transform for Neural Network based Image Classification', Proc. Int. Conf. on System Theory, Morgantwon, pp. 412- 416, 1998.

[127]A. C. Gonzalez, J. H. Sossa, E. M. Felipe and O. Pogrebnyak, 'Wavelet transforms and neural networks applied to image retrieval', Proc. Int. Conf. on Pattern Recognition, Hong Kong, pp. 909-912, 2006.

[128]Kamrul Hasan Talukder & Koichi Harada, _'Haar Wavelet Based Approach for Image Compression and Quality Assessment of Compressed Image‖, IAENG International Journal of Applied Mathematics, 36:1, IJAM_36_1_9.

[129]P. Raviraj, M.Y. Sanavullah,‖ The Modified 2D-Haar Wavelet Transformation in Image Compression‖, Middle-East Journal of Scientific Research 2 (2): 73-78, 2007.

[130]Mallat, S.G.; Heil, C.; Walnut, D.F. A Theory for Multiresolution Signal Decomposition: The Wavelet Representation. IEEE Trans. Pattern Anal. Mach. Intell. 2009, 11, 674–693. [CrossRef]

[131]Addison, P.S.; Walker, J.; Guido, R.C. Time–frequency analysis of biosignals. IEEE Eng. Med. Biol. Mag. 2009,28, 14–29. [CrossRef]

[132]Subasi, A.; Ercelebi, E. Classification of EEG signals using neural network and logistic regression. Comput.Methods Programs Biomed. 2005, 78, 87–99. [CrossRef]

[133]Subasi, A. EEG signal classification using wavelet feature extraction and a mixture of expert model. Expert Syst. Appl. 2007, 32, 1084–1093. [CrossRef]

[134]Gokhale, M.Y.; Khanduja, D.K. Time Domain Signal Analysis UsingWavelet Packet Decomposition Approach. Int. J. Commun. Netw. Syst. Sci. 2010, 3, 321–329. [CrossRef]

[135]Prabhakar, S.; Mohanty, A.; Sekhar, A. Application of discrete wavelet transform for detection of ball bearing race faults. Tribol. Int. 2002, 35, 793–800. [CrossRef]

[136]Ovanesova, A.; Suarez, L. Applications of wavelet transforms to damage detection in frame structures. Eng. Struct. 2004, 26, 39–49. [CrossRef]

[137]Sharif, I.; Khare, S. Comparative Analysis of Haar and Daubechies Wavelet for Hyper Spectral Image Classification. ISPRS-Int. Arch. Photogramm. Remote. Sens. Spat. Inf. Sci. 2014, 40, 937–941. [CrossRef]

[138]Mahmoodabadi, S.Z.; Ahmadian, A.; Abolhasani, M.D. ECG feature extraction using Daubechies wavelets. In Proceedings of the 5th IASTED International conference on Visualization, Imaging and Image Processing, Benidorm, Spain, 7–9 September 2005.

[139]Clonda, D.; Lina, J.-M.; Goulard, B. Complex Daubechies wavelets: Properties and statistical image modelling.Signal Process. 2004, 84, 1–23. [CrossRef]

[140]Misiti, Y.; Misiti, M.; Oppenheim, G.; Poggi, J.-M. Wavelet Toolbox; The MathWorks Inc.: Natick, MA, USA,1996; p. 21.

[141]Cˇ erná, D.; Finek, V.; Najzar, K. On the exact values of coe_cients of coiflets. Cent. Eur. J. Math. 2008, 6, 159–169. [CrossRef]

[142]Stolojescu, C.; Railean, I.; Moga, S.; Isar, A. Comparison of wavelet families with application to WiMAXtra_c forecasting. In Proceedings of the 12th IEEE International Conference on Optimization of Electrical and Electronic Equipment, Brasov, Romania, 20–22 May 2010; pp. 932–937.

[143]Abhyankar, A.; Schuckers, S. Novel Biorthogonal Wavelet based Iris Recognition for Robust Biometric System. Int. J. Comput. Theory Eng. 2010, 2, 233–237. [CrossRef]

[144]Sindhura, S.K.; Reddy, S.N.; Kamaraj, P. Comparison of SNR Improvement for Lower Atmospheric Signals Using Wavelets. Int. J. Adv. Res. Electr. Electron. Instrum. Eng. 2015, 4, 8202–8209.

[145]Jahankhani, P.; Kodogiannis, V.; Revett, K. EEG Signal Classification Using Wavelet Feature Extraction and Neural Networks. In Proceedings of the IEEE John Vincent Atanaso_ 2006 International Symposium on Modern Computing (JVA'06), Sofia, Bulgaria, 3–6 October 2006; pp. 120–124.

[146]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.

[147]Sridhar, S., Rajesh Kumar, P. and Ramanaiah, K.V. (2014). Wavelet Transform Techniques for Image Compression – An Evaluation. International Journal of Image, Graphics and Signal Processing, 6(2), pp.54–67.

[148]al-Qerem, A., Kharbat, F., Nashwan, S., Ashraf, S. and blaou (2020). General model for best feature extraction of EEG using discrete wavelet transform wavelet family and differential evolution. International Journal of Distributed Sensor Networks, 16(3), p.155014772091100.

[149]Rafiee, J. and Tse, P.W. (2009). Use of autocorrelation of wavelet coefficients for fault diagnosis. Mechanical Systems and Signal Processing, [online] 23(5), pp.1554–1572. doi:10.1016/j.ymssp.2009.02.008.

[150]Majumdar, Swatilekha. (2013). Comparative Analysis of Coiflet and Daubechies Wavelets Using Global Threshold for Image De-Noising. International Journal of Advances in Engineering and Technology. 6. 2247-2252.

[151]Porwik P and Lisowska A. The Haar-wavelet transform in digital image processing: its status and achievements. Mach Graph Vision 2004; 13: 79–98.

[152]D. Ro ̧sca, Haar wavelets on spherical triangulations, in: N.A. Dogson, M.S. Floater, M.A. Sabin (Eds.), Advances in Multiresolution for Geometric Mod-elling, Springer, Berlin, pp. 407–419(2005).

[153]S Thakral, P Manhas, C Kumar.: Virtual Reality and M-Learning, International Journal of Electronic Engineering (2010)

[154] S. G. Narkhedkar and P. K. Patel, ‘Recipe of speech compression using coiflet wavelet,’ in 2014 International Conference on Contemporary Computing and Informatics (IC3I), 2014, pp. 1135–1139.

[155] Liu, H., Zhai, L., Gao, Y., Li, W. and Zhou, J. (2005). Image compression based on biorthogonal wavelet transform. [online] IEEE Xplore.

[156] Yadav, A.K., Roy, R., Kumar, A.P., Kumar, Ch.S. and Dhakad, S.Kr. (2015). De-noising of ultrasound image using discrete wavelet transform by symlet wavelet and filters.

[157] J. Kaur and R. Kaur, ‘Biomedical images denoising using symlet wavelet with wiener filter,’ 2013

[158] Anon, (2021). Min-Max Normalization - Machine Learning Concepts. [online] Available at: https://ml-concepts.com/2021/10/08/min-max-normalization/.

[159] D. Reynolds, ‘Gaussian Mixture Models,’ in Encyclopedia of Biometrics, 2015, pp. 827–832.

[160] A. Singh, N. Thakur and A. Sharma, ‘A review of supervised machine learning algorithms,’ 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 2016, pp. 1310-1315.

[161] S. Ray, ‘A Quick Review of Machine Learning Algorithms,’ 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon), Faridabad, India, 2019, pp. 35-39.

[162] 164, [1] Boyd, D. and Crawford, K. (2012). Critical questions for big data: Provocations for a cultural, technological, and scholarly phenomenon. Information, communication & society, 15(5):662–679.

[163] Ng, A. and Musil, S. (2017). Equifax data breach may affect nearly half the us population. Available at https://www.cnet.com/news/equifaxdata-leak-hits-nearly-half-of-the-us-population/.

[164] Newman, L. H. (2017). 6 fresh horrors from the equifax ceo’s congressional hearing. Available at https://www.wired.com/story/equifax-ceo-congresstestimony/.

[165] Zengerle, P. and Cassella, M. (2015). Estimate of americans hit by government personnel data hack skyrockets. Available at https://www.reuters.com/article/us-cybersecurity-usa/ millions-more-americans-hit-by-governmentpersonnel-data-hackidUSKCN0PJ2M420150709.

[166] Mir Sajjad Hussain Talpur, Md Zakirul Alam Bhuiyan, and Guojun Wang. 2015. Shared–node IoT network architecture with ubiquitous homomorphic encryption for healthcare monitoring. International Journal of Embedded Systems 7, 1 (2015), 43–54.

[167] Nishikant, S., Burande, Prof. Kahate, S.A. (2015). Design Model for Two Server Password Authentication Protocol, IJCSET , 11,5, November.

[168] Suveetha, K., Manju, T. (2016). Ensuring Confidentiality of Cloud Data using Homomorphic Encryption, Indian Journal of Science and Technology , 9 (8), February.

[169] Im, J. H., Choi, J., Nyang, D., Lee, M. K. (2016). Privacy-Preserving Palm Print Authentication Using Homomorphic Encryption, In : 2016 IEEE 14th Intl Conf on Dependable, Autonomic and Secure Computing , 878-881.

[170] Marta Gomez-Barrero., Emanuele Maiorana., Javier Galbally., Patrizio Campisi., Julian Fierrez., Multi-Biometric Template Protection Basedon Homomorphic Encryption, Pattern Recognition , 67, July, 149-163.

[171] Scheibner, J., Ienca, M. and Vayena, E. (2022). Health data privacy through homomorphic encryption and distributed ledger computing: an ethical-legal qualitative expert assessment study. BMC Medical Ethics, 23(1). doi:10.1186/s12910-022-00852-2.

[172] A Vijaya Kumar, Mogalapalli Sai Sujith, Kosuri Tarun Sai, Galla Rajesh, and Devulapalli Jagannadha Sriram Yashwanth. 2020. Secure Multiparty computation enabled E-Healthcare system with Homomorphic encryption. In IOP Conference Series: Materials Science and Engineering, Vol. 981. IOP Publishing, 022079.

[173] Razvan Bocu and Cosmin Costache. 2018. A homomorphic encryption-based system for securely managing personal health metrics data. IBM Journal ofResearch and Development 62, 1 (2018), 1–1.

[174] Xiaoni Wang and Zhenjiang Zhang. 2015. Data division scheme based on homomorphic encryption in WSNs for health care. Journal of medical systems 39, 12 (2015), 1–7.

[175] Mostefa Kara, Abdelkader Laouid, Mohammed Amine Yagoub, Reinhardt Euler, Saci Medileh, Mohammad Hammoudeh, Amna Eleyan, and Ahcène Bounceur. 2021. A fully homomorphic encryption based on magic number fragmentation and El-Gamal encryption: Smart healthcare use case. Expert Systems (2021), e12767.

[176] Mir Sajjad Hussain Talpur, Md Zakirul Alam Bhuiyan, and Guojun Wang. 2015. Shared–node IoT network architecture with ubiquitous homomorphic encryption for healthcare monitoring. International Journal of Embedded Systems 7, 1 (2015), 43–54.

[177] Haowen Tan, Pankoo Kim, and Ilyong Chung. 2020. Practical homomorphic authentication in cloud-assisted vanets with blockchain-based healthcare monitoring for pandemic control. Electronics 9, 10 (2020), 1683.

[178] Aitizaz Ali, Muhammad Fermi Pasha, Jehad Ali, Ong Huey Fang, Mehedi Masud, Anca Delia Jurcut, and Mohammed A Alzain. 2022. Deep Learning Based Homomorphic Secure Search-Able Encryption for Keyword Search in Blockchain Healthcare System: A Novel Approach to Cryptography. Sensors 22, 2 (2022),528.

[179] Albrecht M, Chase M, Chen H, et al. Homomorphic encryption standard[J]. Cryptology ePrint Archive, 2019.

[180]C. Gentry, ‘Fully homomorphic encryption using ideal lattices,’ in ACM Symp. Theory of Comput., 2009, pp.

[181]C. Gentry, ‘Toward basing fully homomorphic encryption on worstcase hardness,’ in Proc. CRYPTO, 2010, pp..

[182]C. Gentry, ‘Computing arbitrary functions of encrypted data,’ Commun. ACM, vol. 53, no. 3, pp. 97–105, 2010.

[183]K. Lauter, M. Naehrig, and V. Vaikuntanathan, ‘Can homomorphic encryption be practical?,’ in Proc. ACM Cloud Comput. Security Workshop, 2011.

[184]Ihsan Jabbar, Saad Najim, ‘Using fully Homomorphic encryption to secure cloud computing’, Internet of things and cloud computing, 2016, Volume 4, Issue 2, pp. 13-18, Science Publishing Group, ISSN: 2376-7731, DOI: 10.11648/j.iotcc.20160402.12.

[185]Microsoft Research. (n.d.). Microsoft SEAL: Fast and Easy-to-Use Homomorphic Encryption Library. [online] Available at: https://www.microsoft.com/en-us/research/project/microsoft-seal/.

[186]Cheon J H, Han K, Kim A, Kim M, and Song Y.: Homomorphic encryption for arithmetic of approximate numbers. In International Conference on the Theory and Application of Cryptology and Information Security, pp. 409-437.Springer (2017).

[187]J. H. Cheon, A. Kim, M. Kim, and Y. Song, ‘‘Homomorphic encryption for arithmetic of approximate numbers,’’ in Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur., 2017, pp. 409–437.

[188]F. Boemer, Y. Lao, R. Cammarota, and C. Wierzynski, ‘‘nGraph-HE: A graph compiler for deep learning on homomorphically encrypted data,’’ in Proc. 16th ACM Int. Conf. Comput. Frontiers, 2019, pp. 3–13.

[189]F. Boemer, A. Costache, R. Cammarota, and C. Wierzynski, ‘‘nGraphHE2: A high-throughput framework for neural network inference on encrypted data,’’ in Proc. 7th ACM Workshop Encrypted Comput. Appl. Homomorphic Cryptogr., 2019, pp. 45–56.

[190]X. Jiang, M. Kim, K. Lauter, and Y. Song, ‘‘Secure outsourced matrix computation and application to neural networks,’’ in Proc. ACM SIGSAC Conf. Comput. Commun. Secur., Oct. 2018, pp. 1209–1222.

[191] F. Bourse, M. Minelli, M. Minihold, and P. Paillier, ‘‘Fast homomorphic evaluation of deep discretized neural networks,’’ in Proc. Annu. Int. Cryptol. Conf., 2018, pp. 483–512.

[192] J. H. Cheon, K. Han, A. Kim, M. Kim, and Y. Song, ‘‘A full RNS variant of approximate homomorphic encryption,’’ in Proc. Int. Conf. Sel. Areas Cryptogr., 2018, pp. 347–368.

[193]Y. Lee, J.-W. Lee, Y.-S. Kim, and J.-S. No, ‘‘Near-optimal polynomial for modulus reduction using L2-norm for approximate homomorphic encryption,’’ IEEE Access, vol. 8, pp. 144321–144330, 2020.

[194]E. Lee, J.-W. Lee, J.-S. No, and Y.-S. Kim, ‘‘Minimax approximation of sign function by composite polynomial for homomorphic comparison,’’ IEEE Trans. Dependable Secure Comput., early access, Aug. 18, 2021, doi: 10.1109/TDSC.2021.3105111.

[195]J.-P. Bossuat, C. Mouchet, J. Troncoso-Pastoriza, and J.-P. Hubaux, ‘‘Efficient bootstrapping for approximate homomorphic encryption with non-sparse keys,’’ in Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn., 2021, pp. 587–617.

[196]Y. Lee, J.-W. Lee, Y.-S. Kim, H. Kang, and J.-S. No, ‘‘High-precision approximate homomorphic encryption by error variance minimization,’’ in Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn. (Eurocrypt), 2022.

[197]Microsoft Research. (n.d.). Microsoft SEAL: Fast and Easy-to-Use Homomorphic Encryption Library. [online] Available at: https://www.microsoft.com/en-us/research/project/microsoft-seal/.