Digital Forensic Investigation of IoT Devices: Tools and Methods

Tina Wu

Kellogg College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Trinity 2020

Acknowledgements

A DPhil is not just an academic achievement but also a personal journey of self discovery filled with many low roads, high roads and also surprising detours. This journey was made possible by the Centre for Doctoral Training in Cyber Security, funded by the EPSRC, Department of Computer Science and Kellogg College with their generous additional funding for conferences and equipment. This thesis was only completed with the support of several individuals, I am forever indebted to them for their kind and patient guidance along my journey.

First and foremost, I would like to thank my supervisor Prof Andrew Martin. I am grateful for the freedom and support you have provided throughout my DPhil, which has allowed me to pursue and research a range of different topics. I am also grateful that during my DPhil I was able to collaborate with amazing researchers from several different institutions. I would especially like to thank my co-authors in particular Frank Breitinger for all the challenging and open discussions during our collaborations and helping me to develop my ideas into published papers.

I would like to express my sincere gratitude to my intermediate assessors Andrew Simpson and Prof Niki Trigoni for providing valuable and challenging feedback. Similarly, I thank Jun Zhao and Harjinder Lallie for acting as assessors for my DPhil, both of whom provided invaluable feedback.

I am grateful to have been part of a CDT that undertakes excellent research. I would like to thank my colleagues in RHB 101, not just for the good times that made it a great place to be but also the encouragement, support and positive influence you have provided. I wish you all well in your future careers.

Thank you also to David Hobbs and Maureen York for always being on hand to resolve problems, with any aspect of university life in and outside of the CDT.

I was fortunate to have been member of the archery team and active member of the MCR during my time at Oxford and these activities and the friends I made proved to be a welcome distraction.

Finally, this challenging journey would not have be made without the continued support of those outside of University life. My parents and my siblings, Winnie and Henry for your support even before I started my DPhil have made this possible. I am thankful to my partner John for having your continuous moral support and believing in me, even during the late nights and hardest days of writing this thesis.

Abstract

Consumer IoT devices are becoming omnipresent in our homes, traces generated by their continuous interactions creating a rich source of evidence, useful to forensic investigations in various types of offences. However, existing research in IoT forensics has mainly focused on theoretical frameworks, as current tools and methods do not support newer IoT devices. In this thesis, we focused on developing practical generic solutions to improve the IoT forensic investigative process.

As a first step, we undertook an online survey of the digital forensic community ($n = 70$) on their interpretation of a definition for IoT forensics and a roadmap for future research. We developed a unified understanding of the various terms used and drew a definite conclusion about the IoT technology and its subdomains in IoT forensics. Participants highlighted that research should focus on IoT forensic tools and data acquisition. As a result, we developed a generic acquisition method for smart healthcare devices and leveraged the use of Bluetooth Low Energy (BLE). We demonstrated the feasibility of this method by acquiring health-related traces from smart healthcare devices and the usefulness of these traces for forensic investigations e.g., establishing a historical profile of the user's health.

We dissected the lack of IoT forensic tool development further, through a systematic literature review of ~ 800 articles on the shortcomings of research-based digital forensic tools. We found the requirement for an IoT device identification approach. Consequently, we proposed a generic identification approach that used machine learning and unique features extracted from the network packet. This created a unique "fingerprint", so forensic investigators can uniquely identify the device-type on the network without relying on traditional identifiers.

Drawing upon findings from our previous studies, we developed and evaluated an automated IoT network analysis tool. Through a study of 32 IoT consumer devices, we analysed the network traffic metadata and showed its usefulness for forensic investigations. The results showed many devices used encryption, but smart cameras and smart healthcare devices were prone to sending unencrypted content. Finally, we found most IoT traffic was being sent to the U.S.

Overall, our research has novelty, in that we developed an identification approach, acquisition method and a network analysis tool that can obtain traces from multiple consumer IoT devices. As the IoT expands, it is crucial we develop tools that can support the new devices coming on the market.

Contents

List of Figures

List of Tables

List of Abbreviations

AM . . . . . Approximate Matching

API . . . . . Application Programming Interface

BLE . . . . . Bluetooth Low Energy

BPM . . . . . Blood Pressure Monitor

CSP . . . . . Cloud Service Provider

DFIR . . . . . Digital Forensics and Incident Response

DFRWS . . . . Digital Forensic Research Workshop

DNS . . . . . Domain Name System

ECG . . . . . Electrocardiogram

eMMC . . . . Embedded Multi Media Card

GUI . . . . . Graphical User Interface

HTTPS . . . . Hyper Text Transfer Protocol Secure

IMD . . . . . Implantable Medical Device

ISP . . . . . Internet Service Provider

IoT . . . . . Internet of Things

JTAG . . . . . Joint Test Action Group

kNN . . . . . k-Nearest Neighbours

LR . . . . . Logistic Regression

mDNS . . . . . Multicast DNS

MAC . . . . . Media Access Control

MiTM . . . . . Man In-The-Middle

NIST . . . . . National Institute for Standards and Technology

NRGD . . . . Netherlands Register of Court Experts

NTP . . . . . Network Time Protocol

OS . . . . . Operating System

PCB . . . . . Printed Circuit Board

PII . . . . . Personal Identifiable Information

RF . . . . . Random Forest

RFECV . . . . Recursive Feature Elimination Cross Validation

RFID . . . . . Radio-Frequency Identification

RPi . . . . . Raspberry Pi

SFS . . . . . Sequential Forward Selection

SSH . . . . . Secure Shell

SVM . . . . . Support Vector Machine

UART . . . . . Universal Synchronous Receiver-Transmitter

UPnP . . . . . Universal Plug and Play

VA . . . . . Virtual Assistant

WAP . . . . . Wireless Access Point

XML . . . . . Extensible Markup Language

Any action of an individual, and obviously, the violent action constituting a crime, cannot occur without leaving a trace.

— Edmond Locard (1877-1966)

1 Introduction

Contents

With over 21 billion connected devices around the world serving several sectors, IoT has expanded rapidly over the last decade. This expansion is set to continue with the market predicting a remarkable threefold growth by 2025 [1]. Smart home technology has contributed to the growth over the last decade and is the main area of expansion. According to a YouGov survey from 2018, nearly a quarter of UK households have at least one smart device [2]. This technology promises significant enhancements to the lives of users, one of the main benefits is safety in the home. Popular smart home devices including security cameras, alarm systems and smart locks can quickly and easily heighten home security [3]. Virtual Assistants (VA) such as Amazon Echo and Google Home, dominates the smart device market with 67% of consumers having at least two of these devices [2]. The demand for this

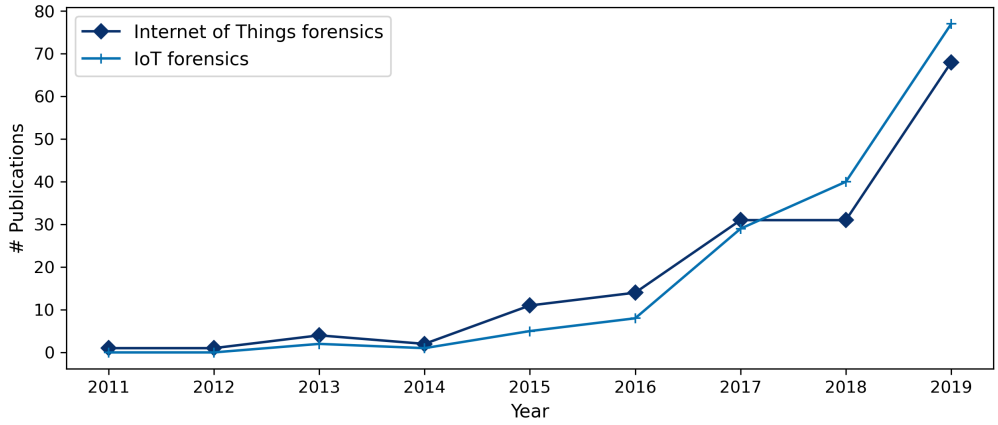


Figure 1.1: Conducting a search over time through academic publications of the term IoT forensics and Internet of Things forensics, shows the number of publications by year (search parameters were on publication's abstract and title) [1]

technology has led housebuilders to offer smart devices already pre-installed [4].

With the increasing numbers and usage of these devices, it is of paramount importance when they are involved in criminal/civil investigations to understand how to carry a forensic investigation. A digital forensic investigation involves the uncovering and interpretation of electronic data and must adhere to a digital forensic process. This commonly consists of four stages: identification, acquisition analysis, and documentation. *Identification* of the potential sources of evidence relevant to the incident. Once identified, evidence is *acquired (acquisition)* while preserving the integrity of the sources. Next, the evidence source is then *analysed* to determine the sequence of events. The final stage involves each stage of the process documented *(documentation)* to present to the court of law [5].

Digital forensics is constantly evolving and new subdomains have emerged, such as mobile forensics, which itself is a fast-growing and evolving discipline and has been defined as *"the science of recovering digital evidence from a mobile device under forensically sound conditions using accepted methods"* [6]. The proliferation of mobile devices (e.g., smartphones, tablets) on the consumer market has caused a greater demand for their forensic examination, which brings unique challenges [7]. These include altering the state of the device by powering down the device before

[1] `https://app.dimensions.ai/discover/publication`

data can be acquired and the wide range of devices available with different technical and physical characteristics (e.g., size, processor speed and memory capacity) requiring multiple acquisition tools [8]. Additionally, the amount of data acquired is dependent on the device manufacturer, model, OS and capabilities of the acquisition tools [9]. Over time, commercial tools were developed, allowing forensic investigators to recover mobile device content with minimal disruption to the device. Popular mobile forensic toolkits are available on the market including Oxygen forensics and Cellebrite's Universal Forensic Extraction Device (UFED) [10]. These types of tools can aid the extraction of certain data, such as call records, social media data, communication data.

The use of IoT devices and their rising prevalence create a unique opportunity to recover evidence in both physical (e.g., arson, assault) and virtual (e.g., DDoS attacks, identity theft) crimes. These devices are constantly in use with autonomously generated data potentially capturing useful evidence, making them an ideal digital witness for forensic investigators. However, as these devices become ever more intertwined in our daily lives, many technical issues in the forensic process have yet to be resolved. It is imperative that forensic investigators can rely on data and maintain its integrity throughout the course of an investigation, therefore validated tools and techniques are required. IoT forensics is an emerging subdomain of digital forensics and is a relatively new and unexplored area. Although academic research in this area has been expanding rapidly over recent years (see Figure 1.1), research remains "siloed" to individual devices and does not incorporate the reality of the "melting pot" that smart environments inhabit. In this thesis we address this by proposing generic methods and tools to improve the IoT forensic investigation process.

Smart home devices are an emerging technology, only increasing in popularity over the last few years. However, there are clear privacy concerns regarding IoT devices namely when they can exhibit unexpected behaviour or do not use encryption. For instance, with VA devices where reports by consumers suggest they *"often record accidentally in response to non-keywords"* when they should only respond to the programmed "wake" word [11]. While this may be a violation of privacy,

it could be useful in the forensic context, providing a treasure trove of evidence for forensic investigators. There have already been several criminal cases requiring forensic investigation of IoT devices. In particular, the US has seen several cases involving the Amazon Alexa, in 2019 police investigating a murder believed that Alexa accidentally activated, recording key moments of the incident [12]. This was not the first time Amazon had been asked to handover evidence, in 2015, data from Alexa was seized in a murder investigation. However, Amazon only handed over these recordings with the consent of the accused. In a similar case that involved a Fitbit device, police used the victim's recorded heart rate measurements to determine the exact time of the murder [13].

With the ever-increasing integration of new technology in our lives, such as autonomous vehicles, home automation and health care monitoring, investigations involving IoT devices are certain to increase. This poses new challenges for investigators as liability is certain to be disputed among manufacturers, providers, and users. Additionally, as retailer giants such as Amazon do not release consumer's user data without a "*valid and binding legal demand*" [14]. Authorities will seek other avenues to directly access hardware such as smart speakers, thermostats, and alarm systems to search for evidence in timestamps, audio files or other personal data.

1.1 Motivation

IoT forensics research is in its early stages and existing efforts have already highlighted the challenges investigators face when carrying out an IoT forensic investigation such as the lack of technical capability [15], diversity of devices/protocols [16, 17] and distributed data. However, what remains unclear are which challenges should be focused on first. Furthermore, there are many definitions of IoT forensics, but they are inconsistent and not formed from the input of the digital forensic community [18, 19]. To address these limitations, this thesis first gathers the opinions of the digital forensic community on concepts such as the definition of IoT forensics and what current and future research should focus on. This thesis developed a roadmap for current research, where we found the

focus should be on IoT forensic tool development, acquisition and generic tools / solutions compatible with multiple manufacturers. The responses to the survey question on the definition of IoT forensics helped us formulate a concise definition by unifying the diverse terms used in previous definitions and feedback from the digital forensic community. Our definition provides a clearer understanding of the type of devices investigated, skills required, sub-domains involved and finally the purpose of an investigation (Chapter 3).

Having identified that one of the main research gaps was the lack of IoT forensic tool development, this thesis then investigates the current landscape of published research-based forensic tools to establish what IoT forensic tools/solutions are required. Research-based forensic tools are the main output of digital forensic research, such as [20] tool was developed by analysing the propriety protocol (Chapter 4). As a result of this survey, we find that there is a requirement for an IoT network traffic analysis tool and approach to identify the type (model and manufacturer) of IoT devices on the network.

Similar to conventional digital forensics, IoT forensics mainly consists of the following four main stages: identification, acquisition, analysis and documentation. Holistically, we have considered each of these stages to improve the digital forensic investigation of IoT devices.

Identification. In this part of the thesis, we investigate the identification stage of the digital forensic process, which in traditional digital forensics is the physical "search and seizure" of forensic evidence [17]. As the number of IoT devices increases and their size decreases, physically locating IoT devices at a crime scene poses ever greater challenges for forensic investigators [16]. Traditionally, IoT devices can be identified using device identifiers such as the MAC address but this only reveals the manufacturer [21] and can be easily spoofed by malicious devices. However, in this thesis, we aim to identify the device-type, which is defined in this work as a combination of manufacturer, model and software version of a particular IoT device. For instance, Samsung, Samsung SmartThings Hub and v2.

Recent research has demonstrated that IoT devices can be identified by using implicit identifiers, which includes extracting information from the packet headers and computing the statistics (e.g., the minimum size of IP payload) [22, 23] or behavioural patterns of the IoT device (e.g., the total number of download and upload bytes), which are traffic characteristics obtained from network traffic flow [24]. However, it is unclear how these approaches can be applied in the forensic context, they are inefficient requiring large training datasets and depend on many features to train the classifier. To address these limitations, in our proposed machine learning approach we train the classifier using both packet header and behavioural features, benefiting from the packet header for speed and behaviour features for accuracy. Although behavioural features take longer to generate, they are unique to that device, whereas individual packet header features on their own are not, but they are quick to generate. Consequently, when these features are combined, they create a distinct "fingerprint" of the device. Additionally, we reduce the number of features required by using feature selection methods. We demonstrate our approach can be applied to multiple IoT devices by testing it on devices from various manufacturers (Chapter 5).

Acquisition. In the second part, this thesis focuses on the acquisition stage encompassing the extraction and analysis of traces directly obtained from the IoT devices. This is an important stage and it is essential that traces are accurately and completely obtained [25]. However, it is a complex process to acquire evidence from different IoT devices without a common interface, internal storage or standard protocols [26]. Prior work has developed acquisition frameworks limited to specific IoT devices, which only extract data from that ecosystem, such as the Amazon Alexa acquisition framework [27]. While [28] acquired data by developing tools/plugins for individual IoT devices, which is a time-consuming process. Instead, this thesis adopts a novel approach, developing a generic acquisition method that is compatible with a range of IoT device manufacturers. We do this by leveraging the use of a ubiquitous protocol, Bluetooth Low Energy (BLE), to acquire sensitive health-related traces from smart healthcare devices (Chapter 6).

BLE makes an attractive acquisition method, as it is integrated into many consumer IoT devices, including smartwatches, fitness trackers [29] and smart blood pressure monitors [30]. This means this method of acquisition is compatible with a wide range of BLE-enabled consumers IoT devices. In addition, BLE has limited security implementations such as the lack of authentication or secure pairing to protect data [29, 31, 32]. This enables forensic investigators to easily extract traces such as blood pressure readings and timestamps which can be used as evidence or to develop a historical profile of the users' health.

Analysis. The final part of this thesis focuses on the analysis stage of the digital forensic process, which can provide a clearer picture of events that have occurred. On an IoT device there are many different types of traces that can be analysed for forensic purposes, such as memory, internal storage [33] etc. However, this thesis focuses on the network layer, which is of greater significance as it provides an insight into ways investigators can obtain information from the IoT device e.g., enabling remote services such as SSH or telnet. Although recent work has demonstrated the benefits of analysing IoT network traffic for forensic traces, including obtaining a list of API calls to retrieve user data from the cloud [27] and sensitive data such as login credentials [28, 34]. These have been limited in the number of IoT devices they studied. Therefore, in this work, to understand what "metadata" can be obtained from the network traffic for forensic purposes we conduct a large-scale in-depth study of the network traffic from a wide range of popular consumer IoT devices. This study includes examining the IP payload for cleartext content for user credentials, and the destination IP address to reveal the final location of the data. Additionally, during this study, we found that manually analysing IoT network traffic requires a considerable amount of time. Consequently, this thesis introduces the first tool capable of automating the analysis of IoT network traffic from a multitude of IoT devices (Chapter 7).

To solve these open issues, in the next section we identify the research questions that motivate this thesis.

1.2 Research Questions

This thesis aims to improve the overall IoT forensic investigation lifecycle and we do so by addressing each stage of the digital forensic process. Furthermore, we are motivated by the lack of generic solutions that are compatible with more than one IoT device. The research questions which motivate this work are as follows:

RQ1 What are the digital forensic community's interpretation and view of IoT forensics and what should be the focus of current and future research?

RQ2 What are the capabilities and shortcomings of existing digital forensic research tools and what are the gaps in the existing offerings?

The remaining research questions are motivated by the open problems from the findings in RQ1 (Chapter 3) and RQ2 (Chapter 4).

RQ3 Can we accurately identify IoT consumer devices on the network and what methods can improve the efficiency and accuracy of identification?

RQ4 What traces can be acquired through Bluetooth Low Energy (BLE) and the accompanying mobile app? (Smart healthcare devices predominately use BLE for data transmission)

RQ5 What network traffic metadata can be utilised from the consumer IoT devices to benefit a forensic investigation?

1.3 Contributions of our Research

This section describes the contributions made by our research and outlines the publications that provide the foundation of this thesis. In Figure 1.2, we present the map of research questions and their outcomes. The aim of this thesis is to improve the overall IoT forensic investigation lifecycle and we do so by addressing each stage of the digital forensic process. Furthermore, we are motivated by the lack of solutions compatible with more than one IoT device. To do so, we first

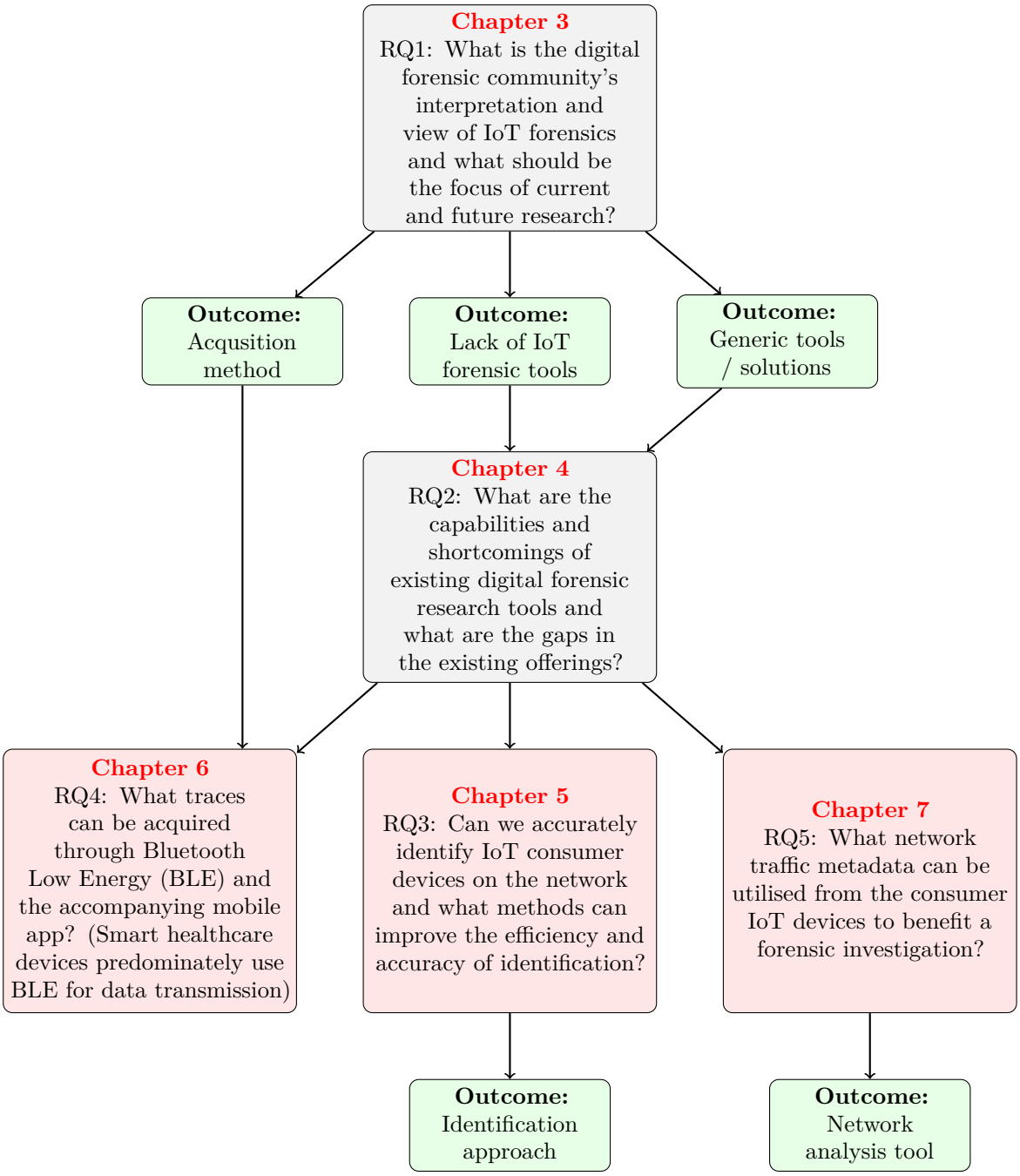


Figure 1.2: Chapters and the outcomes in which we have addressed the research questions

introduce the relevant background and related work (Chapter 2), this thesis then investigates **RQ1: What is the digital forensic community's interpretation and view of IoT forensics and what should be the focus of current and future research?** (Chapter 3). Specifically, through a substantial online survey (n= 70) from the digital forensic community. We present their interpretation and

views on concepts such as definitions, capability in the analysis of IoT data/devices and current / future challenges in IoT forensics. Although several authors propose various definitions of IoT forensics, they have so far been inconsistent and have not been thoroughly developed with the input of the digital forensic community. Based on our survey results, we present a new definition of IoT forensics by unifying terms used in previous definitions and the input of the digital forensic community. In doing so, we create a concise definition of IoT forensics and a clearer understanding of the topic. Furthermore, the survey highlighted research topics of interest. Specifically, that current research should focus on the acquisition and the development of IoT forensic tools, this is the motivation for the work in the following chapters.

Having found that there was a lack in the development of IoT forensic tools, the next natural step was to investigate **RQ2: What are the capabilities and shortcomings of existing digital forensic research tools and what are the gaps in the existing offerings?** To answer this research question, we conduct a comprehensive survey of $\sim$ 800 academic publications from various digital forensics platforms between 2014 and 2019, filtering out those presenting tools. To the best of our knowledge, this is the first study of the shortcomings in existing research-based tools. To further the state-of-art we analyse whether the tools are freely available, well-maintained and documented. In doing so, our survey brings us to an analysis of the challenges with published tools, including recommendations on how to enhance the status quo and identify the gaps in research of current IoT forensic tools. Additionally, to further the state-of-art we present an updated and extended digital forensic tool taxonomy. As a result, forensic practitioners can easily identify appropriate tools for a given task with the new subfields. This work forms Chapter 4 with the results of this study published at the *2020 Forensic Science International: Digital Investigation (FSIDI) Journal*.

Identification Motivated by the observations in Chapter 4, where we found that there is a requirement for a method to identify the type of IoT devices on the network, this thesis investigates **RQ3: Can we accurately identify IoT**

consumer devices on the network and what methods can improve the efficiency and accuracy of identification? In Chapter 5 we propose a machine learning approach to identify the type of IoT based on features extracted from the packet header and behavioural patterns of the devices. Specifically, this thesis introduces the first steps in IoT forensics to identify IoT device-types on the network without relying on traditional identifiers such as the MAC address or the hostname. We present our machine learning approach to identify the IoT device-type on the network. In contrast to earlier research, our approach uses packet header and behavioural features combined with a feature selection method to reduce the feature set from 22 to 9, and significantly improve the runtime of identification by around 80%. Overall, the k-NN classifier perform the best using 9 features with an overall identification F_1*-score* of 98.2%. We demonstrate that our identification approach is capable of identifying a multitude of IoT devices from various manufacturers and groups. Additionally, to determine the scalability of our approach we evaluated the efficiency of our approach on 3 different types of hardware. The results show that overall it took the high-end desktop 2.32 seconds(s), while the RPi 3 took on average 4.29s, nearly twice as long, but acceptable especially if we to run our approach on a mobile system.

Acquisition Influenced by the findings in Chapter 3, where we found that current research should focus on developing a generic acquisition method, we investigate **RQ4: What traces can be acquired through Bluetooth Low Energy (BLE) and the accompanying mobile app? (Smart healthcare devices predominately use BLE for data transmission)**. In doing so, this thesis introduces BLE as a novel method to acquire traces from smart healthcare devices (Chapter 6). We demonstrated the feasibility of using this method to acquire traces such as blood pressure measurements from 4 Blood Pressure Monitors (BPM), which are of forensic significance as they can aid an investigator in creating a historical profile of the users' state of health. It is further shown that this method is compatible with a range of IoT device manufacturers. As an auxiliary contribution, our study

provides an insight into the availability of the user data from accompanying mobile apps of the smart healthcare devices and detailed documentation of our findings. Furthermore, we conduct an analysis on 8 smart healthcare devices to determine the confidentiality of the data. This further demonstrates that our proposed method can be of used to an investigator as data can be easily accessible on BLE enabled devices. This work was published in *2018 TrustCom/BigDataSE* [35].

Analysis One of the gaps identified in this thesis (Chapter 4) is the requirement for a tool to analyse IoT network traffic. We do so by exploring the research question, **RQ5: What network traffic metadata can be utilised from consumer IoT devices to aid a forensic investigation?** We address this in Chapter 7 where we conduct a large-scale study on the network analysis from a forensic perspective of the communication channels of 32 IoT consumer devices. We provided insights into the use of the metadata for forensic purposes and type of traces obtainable from IoT traffic. We facilitated this, by investigating if ports used by IoT devices are exposed, if encryption is used by devices and mobile apps and examine any unencrypted traffic for cleartext content. We then identified the final destination the IoT devices communicate/establish connections with. We demonstrate the Shannon entropy test is a useful pre-test in identifying unencrypted content within the payload of the packet. Additionally, we find 4 other randomness tests that are equally suitable at identifying encrypted/unencrypted IoT traffic. Although many devices encrypt their data, we find in particular smart cameras would send data in cleartext when they detect motion or during updates. Using the destination IP address, we find the majority of data transverses to the U.S. with data stored on Amazon servers. Lastly, we find many of the devices contacted multiple destinations, which could be an issue for future investigations involving multi-jurisdictions.

We further the state-of-art by introducing our tool `IoT Network Analyzer`, which aids forensic investigators to analyse IoT traffic and can be used to compare a list of ports against a list of pre-defined secure ports and identify the location to which the data transverses. Furthermore, no tool that has all the features that our

tool offers, which can automatically calculate the entropy of the IP payload. Finally, we evaluate the tool running experimental studies. The findings from this study were published at the *2021 Digital Forensic Research Workshop (DFRWS) APAC*.

1.4 Scope of this Thesis

We have scoped this research to consumers IoT devices, which encompasses technology such as wearables, home automation, healthcare monitoring devices etc. Although there are many other IoT domains, most individuals are unlikely to interact with industrial devices from domains such as factory control systems. Furthermore, these technologies are more expensive, rarer and not readily available for use in experiments. Whereas consumer IoT devices are more likely to be owned by individuals and are openly available. Within this thesis we have used the terms IoT device and consumer IoT device interchangeably.

It is not feasible to study every consumer IoT devices, therefore we have selected to focus on smart homes and the smart health care sector. We based this on the responses to the question in our survey from Chapter 3, where we asked participants what they considered to be an IoT device, smart homes was the highest response. Additionally, smart homes and smart healthcare are closely related as they are primarily controlled by the consumer. Subsequently, we considered smart healthcare devices such as Blood Pressure Monitors (BPM), glucose meters, Electrocardiogram monitors (ECG), as part of the consumers "Daily Healthcare" in the smart home, as opposed to being confined to clinical environments.

1.5 Publications

This section gives an overview of publications and submissions currently under review.

[36] **Tina Wu**, Frank Breitinger, and Ibrahim Baggili. *IoT Ignorance is Digital Forensics Research Bliss: A Survey to Understand IoT Forensics Definitions, Challenges and Future Research Directions.* In: Proceedings of the 14th

International Conference on Availability, Reliability and Security (ARES '19), 2019.

[37] **Tina Wu**, Frank Breitinger, Stephen O'Shaughnessy. *Digital Forensic Tools: recent advances and enhancing the status quo.* In: Forensic Science International: Digital Investigation (FSIDI) Journal, 2020.

[35] **Tina Wu** and Andrew Martin. *Bluetooth Low Energy Used for Memory Acquisition from Smart Health Care Devices.* In: 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), 2018.

[38] **Tina Wu**, Frank Breitinger, Stephen Niemann. *IoT network traffic analysis: opportunities and challenges for forensic investigators?* In: Digital Forensic Research Workshop (DFRWS) APAC 2021, 2021.

1.6 Work Done in Collaboration

The research outputs presented in this thesis have been conducted in collaboration with other researchers, whose specific contributions we now acknowledge. Apart from the exceptions listed below, all other contributions and research efforts presented in this thesis are my own work, completed under the guidance of my supervisor, Prof Andrew Martin.

Chapter 3 is formed from our paper [36]. I am entirely responsible for the idea and design of the online survey, analysis of results, including the new definition of IoT forensics, as well as the identification of open research challenges. Whereas Frank Breitinger provided constructive feedback during the analysis of the results.

Chapter 4 is formed from our paper [37]. I am responsible for surveying the digital forensic tools and gathering information on the tools. I also categorised the digital forensic tools into various subfields of the taxonomy and the remaining analysis of the research. Frank Breitinger and Stephen O'Shaughnessy contributed with valuable discussions and suggestions related to the idea of the digital forensic tool taxonomy.

The work in Chapter 7 is related to our paper [38]. I am responsible for the design of the experimental setup, analysis of the collected network traffic and the development of the Python scripts for the initial entropy calculations. I am entirely responsible for the idea of the tools' features including entropy calculation, cleartext extraction, location of data, usage of secure ports and the evaluation of the tool. I began the development of the tool to identify the location of the data, cleartext extraction and identify a quicker method to process the PCAP files, while Stephen Niemann aided in the development of the tools remaining features.

1.7 Thesis Outline

This thesis is structured as follows:

- Chapter 2 introduces the reader to background knowledge in IoT and digital forensics. Then continues with a brief overview of the data sources available on an IoT ecosystem and the challenges in IoT forensics. We then review existing research in IoT forensics, particularly smart home and smart healthcare forensics.

- Chapter 3 provides an overview of the results from the online survey conducted on digital forensic practitioners. We present their interpretations and views on various issues such as definitions, experience/capability and a roadmap for future research in IoT forensics and a clearer understanding of the term.

- Chapter 4 overviews the quality and availability of research-based tools from prominent digital forensic conferences/journals, highlighting their shortcomings. Lastly, we discuss the gaps in IoT forensic research tools.

- Chapter 5 proposes an approach to identify IoT devices on the network to speed up the identification process in a digital forensic investigation. We also demonstrate the use of feature selection methods to improve the identification speed.

- Chapter 6 presents a method that can acquire traces from smart healthcare devices using BLE. Next, we discuss the types of traces found on the Android mobile device. Additionally, we perform a security analysis of the data stored on the smart healthcare device.

- Chapter 7 analyses the network traffic of 32 consumer IoT devices from the forensic perspective. We investigate whether the devices had any open ports, use encryption, examine whether the mobile app to cloud communication can be exploited using a proxy server and lastly, the destination data terminates Furthermore, to automate this analysis we developed and evaluated our prototype tool.

- Chapter 8 finally conclude this thesis by summarizing the results and discussing future work that is required to progress the field of IoT forensics.

1.8 Ethical Considerations

We acknowledge that this work raises three main ethical concerns. Firstly, the data collected from the online survey was subject to ethical review and approval from Oxford's Central University Research Ethics Committee (CUREC). This is particularly important when handling personally identifiable information or conducting studies with human participants. Approval was granted under reference number CS_C1A_18_009.

The second concern is the collection of network traffic is conducted on physical consumer IoT devices which could contain sensitive and private user information. However, we limited our data collection to a laboratory setting where we had control over the environment with limited human interactions. Additionally, we only tested with devices we owned using our personal data.

The last concern regards the tools we used and developed in our experiments, we have ensured that we have not provided tools that attackers could misuse or exploit. Whilst we consider publishing this work an important part of the process;

we have taken care to not reveal details which would enable threats to carry out attacks who otherwise would not have been able to do.

...when you have eliminated the impossible, whatever remains, however improbable, must be the truth.

— Sir Arthur Conan Doyle

2 Background

Contents

This chapter begins with a literature review methodology. Then covers the necessary background on IoT and digital forensics to contextualise the topics covered in this thesis. We then overview existing research that focuses on various aspects of IoT forensics. Specifically, to get a clearer understanding of the topic we discuss

the various definitions of IoT and the data sources available in an IoT ecosystem. We then discuss the challenges in IoT forensics and briefly overview investigative frameworks. Next, as we scoped our research to smart home and smart healthcare forensics, we provide a detailed review of the existing research in these domains.

2.1 Literature Review Methodology

A systematic literature review was conducted, throughout which a comprehensive and reproducible method of data collection was followed:

1. Planned the review: The planning phase focused on defining research questions to scope the literature search (as shown in Section 1.2).

2. Searched for literature: In the search phase, the relevant research databases and the keywords to be used during these searches were determined. Data for the study were available in the databases, which included Google Scholar, ScienceDirect, IEEEXplore and ResearchGate. Guided by our research focus on IoT forensics and related subdomains, we chose the following keywords for our literature search: “IoT forensics”,“smart home forensics” and “smart healthcare forensics”.

3. Extracted data: After a broad initial literature search of the keyword “IoT forensics”, we found 9180 results. We then set an exclusion criteria to exclude papers that did not have the keyword “IoT forensics” in the full publication. This was also carried out using the other keywords identified. We then set an inclusion criteria to select only peer-review articles between 2013-2019. The reasoning for the year criteria, is that during our initial search we found no articles on IoT forensics prior to 2013. Our final list consisted of a total of 118 peer-reviewed articles that covered areas ranging from IoT, digital forensics and IoT forensics. We then analysed each publication found and conducted the following: identified the research question/problem the author sought to address, strength and weakness of the research, key findings, results and conclusions of the study.

4. Reported and disseminated: Lastly, we reported our findings from the literature review starting from Section 2.2.

2.2 Internet of Things (IoT)

The term IoT was originally coined in 1999 by Kevin Ashton to promote the new RFID technology, however, the term did not come into popular use until 2011, and by 2014 it was in widespread use in the market [39]. It is important to have a clear definition of IoT, although it has become a common buzzword, no comprehensive definition has been accepted. An ITU internet report from 2005 defined IoT as *"anytime, anywhere and any media"* [40]. Although this definition captures the vision of IoT and works as a marketing strapline, it does not provide an accurate description. Another definition by McKinsey has interpreted IoT as:

> *"Sensors and actuators embedded in physical objects are linked through wired and wireless networks, often using the same Internet Protocol (IP) that connects the Internet."* [41]

While this definition is technically accurate and may provide a significant basis, it fails to encompass the full scale and ethos of IoT.

Consumer IoT These devices are built for domestic use, they range from large smart home appliances such as fridges to wearable health monitors and smart bulbs. While there are many IoT applications smart homes and healthcare are the major domains. Therefore, as the focus of our study, we have selected these sectors, which currently hold a combined global value of 67 billion USD [42].

Smart Homes Smart home is referred to as "*a convenient home setup where appliances and devices can be automatically controlled remotely from anywhere with an internet connection using a mobile or other networked device. Devices in a smart home are interconnected through the internet, allowing the user to control functions such as security access to the home, temperature, lighting, and a home*

theatre remotely" [43]. Although this definition encompasses what is expected of a smart home, it is verbose.

There is no established definition of what an smart device is, however, [44] has defined it as "*a context-aware electronic device capable of performing autonomous computing and connecting to other devices wired or wirelessly for data exchange.*". Smart homes provide additional functionality and security through their connectivity to the internet, allowing users to remotely control and monitor their homes. This type of technology has been available since 2000, initially it was slow to be accepted but over the years with the technology becoming more affordable and accessible it has gradually become widely adopted by consumers [45].

Smart Healthcare With the increasing life expectancy and ageing population, smart healthcare has emerged as an important sector. Yin et al. [46] defined smart healthcare as " *besides clinical usage, it also utilizes Implantable and Wearable Medical Devices (IWMDs) to gather, store, and process various types of physiological data during daily activities*". Smart healthcare aims to provide care in the home for the elderly, outpatients, people with disabilities and aids healthy individuals in monitoring their health and well-being. Wearable medical devices such as BPMs, glucose meters, etc., are used in conjunction with a mobile device, enabling a patient to receive continuous monitoring and treatment in the comfort of their homes [47, 48].

2.3 Digital Forensics

2.3.1 Definitions

Digital forensics focuses on the recovery and investigation of data to obtain digital evidence, to uncover and interpret electronic data in litigation or criminal investigations [49]. Forensic investigators are required to identify, collect, recover, analyse and preserve digital evidence from any type of storage media. This could be anything from a hard drive or mobile device to small devices such as SD cards and USB sticks [50]. For the most part, traditional digital forensics has progressed well with the creation of commercial tools, forensically sound principles,

and methodologies. Although the recent rapid growth of new technology has left the digital forensic industry and practitioners struggling to keep up in developing new tools and methods [15].

Over the years several definitions of digital forensics have been proposed, however, there were two which were prominent. The first was presented by the Digital Forensic Research Workshop (DFRWS) [51] from 2001 and defined as follows:

> *"The use of scientifically derived and proven methods toward the preservation, collection, validation, identification, analysis, interpretation, documentation, and preservation of digital evidence derived from digital sources for the purpose of facilitating or furthering the reconstruction of events found to be criminal, or helping to anticipate unauthorized actions shown to be disruptive to planned operations."*

The core of this definition was the emphasis on event reconstruction and reaction to security threats. Whereas the definition offered by the NIST is:

> *"Considered the application of science to the identification, collection, examination, and analysis of data while preserving the integrity of the information and maintaining a strict chain of custody for the data. Data refers to distinct pieces of digital information that have been formatted in a specific way."* [52]

This definition was focused on the legal aspect but also emphasise that forensic procedures should be adhered to.

2.3.2 Digital Forensic Investigative Frameworks

One of the most prominent topics in early research of digital forensics was to define a framework to make the investigative process consistent and standardised. Many general digital forensic frameworks have been proposed, such as the Digital Forensic Research Workshop investigative framework (DFRWS), which has six main steps: identification, preservation, collection, examination, analysis and presentation [53].

Over the years new frameworks were presented and refined with additional steps, with each step more precisely defined to meet the ever-changing technology. For instance, refining an existing framework to improve a specific step of the investigation such as the Systematic Digital Forensic Investigation Model (SRDFIM) [54]. This

model aids forensic practitioners in setting up appropriate policies and procedures in a systematic manner. Whereas more recent frameworks were developed for specific issues. For example, dealing with a specific category of cases such as network forensic analysis [55].

Numerous digital forensic investigative frameworks have been proposed in the literature to date without one being universally accepted as the best practice. Instead, we follow a commonly used digital forensic framework defined by the National Institute of Standards and Technology (NIST). This is a high-level digital forensic process framework and consists of the following stages: *identification* of the potential sources of evidence relevant to the incident. Once identified, evidence is *acquired (acquisition)* while preserving the integrity of the sources. Next, the evidence source is then *analysed* to determine the sequence of events. The final stage involves each stage of the process being documented *(documentation)* to present to the court of law [5].

Now that we have established a context of the term digital forensics and what the investigative process encompasses, in the next subsection, we overview existing research in IoT forensics. Next, we review existing research in smart home forensics and smart healthcare, as they provide the foundation and background for the research in later chapters.

2.4 IoT Forensics

IoT forensics is still in its infancy and very limited research has been published. This is evident when searching with Google for "definition of IoT Forensics" (in quotes) revealed only 9 results, where some pointed to the same article on different platforms (e.g., [56] can be found on ieeexplore, dl.acm.org and researchgate). To get a clearer understanding of the topic, we first look at existing definitions of IoT forensics. We then present the various data sources found on an IoT ecosystem, then discuss the current research challenges. We then overview existing research on theoretical investigative frameworks. Next, we present prior research on smart

Author	Definition	Discussion
[57]	Has broken down IoT forensics into domains such as cloud services, visualization, mobile devices, fixed computing, sensor and RFID technologies and Artificial Intelligence (AI)	The focus has been on the subdivision of IoT forensics rather than focus on the digital forensic process. The following terms were also used to emphasise the possible technology that was involved using terms such as AI, sensors, fixed computing and RFID technologies.
[56]	A separate branch of digital forensics and identify it as three different fields: cloud, network and device level forensics. The identification, collection, organisation and presentation processes deal with IoT infrastructures to establish facts about incidents	The focus was on the subdivisions of IoT forensics. Unlike [57], they also included the main steps of an IoT forensic investigation.
[19]	Separated IoT forensics as a combination of different technology zones: IoT, network and cloud zones	The focus was on the subdivision of IoT forensics into various zones. This was similar to [56, 57] definitions, but did not include details on the main processes of a digital forensic investigation in an IoT environment.

Table 2.1: Discussion on the similarities and differences of the three definitions of IoT forensics identified in previous literature.

home forensics, dividing it based on the various data sources in an IoT environment. Finally, we briefly discuss existing work in smart healthcare forensics.

2.4.1 Definitions of IoT Forensics

At the time of writing this thesis, we identified three definitions of IoT forensics in our literature review from the following authors [19, 56, 57]. Although there may be related definitions regarding digital forensics, the three proposals from [19, 56, 57] were the only specific definitions for IoT forensics. In the following, we provide detailed analysis of these three definitions, highlighting their similarities and differences. These lay the foundations for our proposal of a synthesised IoT forensic definition.

In defining IoT forensics, Oriwoh et al. [57] has broken the definition into domains such as cloud services, visualisation, mobile devices, fixed computing, sensor, RFID technology and artificial intelligence. The authors focused on the subdivision of

IoT forensics rather than on the digital forensic process. The following terms were also used to emphasise the possible technology that was involved using terms such as *"AI, sensors, fixed computing and RFID technologies"*. Similarly, Zawoad et al. [56] have defined IoT forensics as a separate branch of digital forensics and identified it as three different fields: cloud, network and device level forensics. The authors have also included the terms: *"The identification, collection, organisation and presentation processes deal with IoT infrastructures to establish facts about incidents"*, which were included to emphasise the main processes of a digital forensic investigation. While Alabdulsalam et al. [19], separated IoT forensics in to a combination of different technology zones: IoT, network and cloud zones. Again, the focus was on the subdivision of IoT forensics into various zones. This was similar to [56, 57] definitions, but did not include any details on the main processes of a digital forensic investigation in an IoT environment.

Although these preliminary definitions exist, they are inconsistent, as they cover different aspects of IoT forensics, and none is sufficient on its own. Furthermore, they were not developed through consultation within the digital forensic community, thus there is currently no universally accepted definition. Table 2.1 shows an overview of the various definitions and discussion on the similarities and differences in the definitions of IoT forensics.

2.4.2 Data Sources in IoT Ecosystems

Regardless of the type of IoT application, the ecosystem has 4 main components where data resides and is illustrated in Figure 2.1. (1) Cloud: The Cloud Service Provider (CSP) provides the interface for local and remote monitoring and control of the IoT devices. (2) Device: IoT hardware, for example in a smart home this would be the central device such as the smart hub and the connected sensors and (3) Mobile device/app: the mobile app is used to configure and manage the IoT devices and (4) Network: devices are either connected through wireless or wired through Ethernet. However, different IoT environments may also use other communication protocols, for example, BLE, Zigbee, Z-Wave etc.

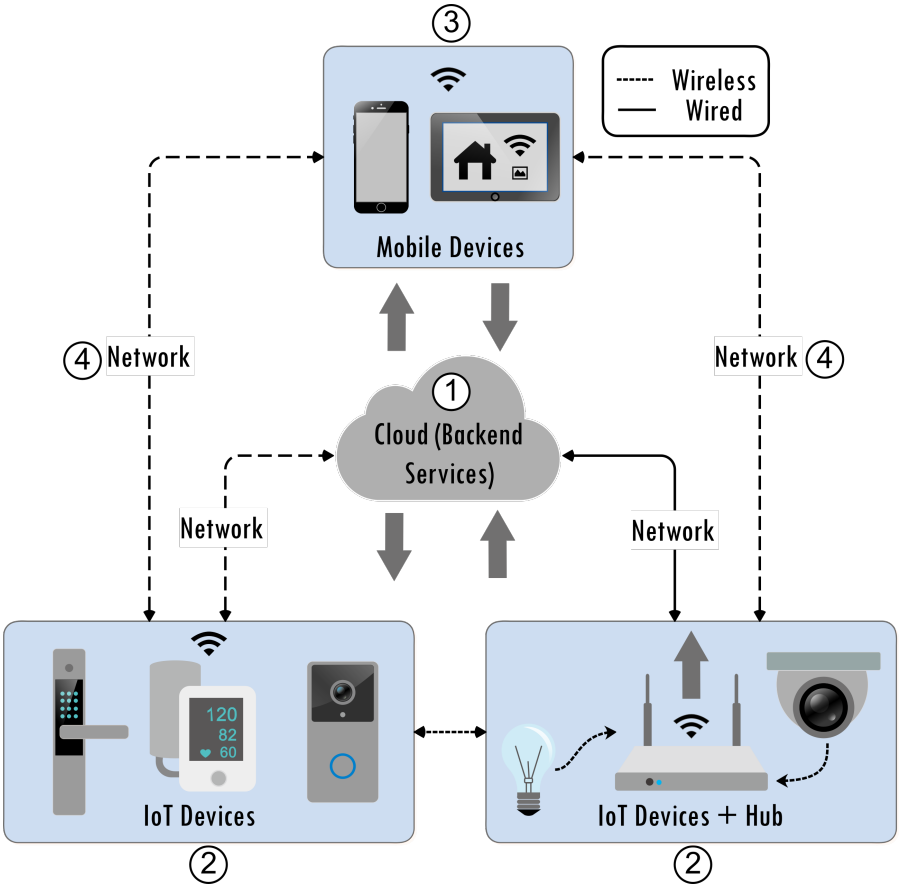
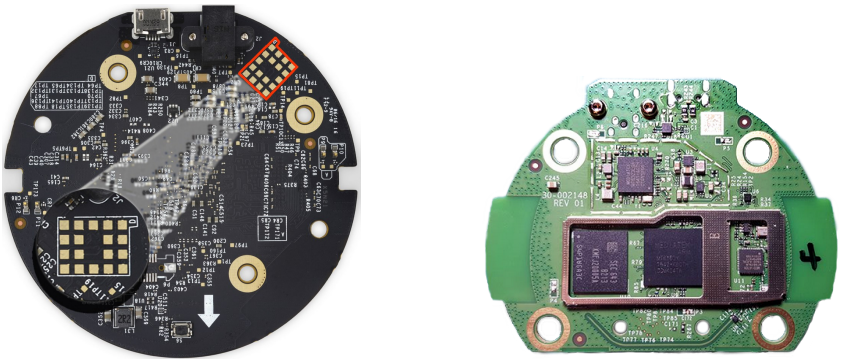


Figure 2.1: Simplified IoT ecosystem. The backend cloud serves as a bridge between the IoT devices and the companion apps use the web service APIs to provide the required interface for data exchange and management services

2.4.3 IoT Forensic Challenges

In this section, we overview the main challenges in IoT forensics that stem from related work.

Lack of Technical Capability Garfinkel [15] identified that many tools and techniques that once worked with traditional forensics will become obsolete due to the progress of technology. A recent paper by Luciano et al. [60] laid out digital forensics research for the next five years. They found there was a lack of funding towards research, training and forensic tools in IoT. Although these should be the most important research opportunities in the future.



(a) The PCB of a 1^{st} generation Amazon Echo Dot released in April 2016, highlighted in red is the location of the debug ports [58].

(b) The PCB of a 2^{nd} generation Amazon Echo Dot released in August 2019, with the debug ports removed [59].

Figure 2.2: PCB from two generations of the Amazon Echo Dots. Figure 2.2a is the Amazon Echo Dot 1^{st} generation and Figure 2.2b the Amazon Echo Dot 2^{nd} generation

Diversity of Devices and Protocols The diverse types of IoT devices has made it difficult for forensic investigators to acquire and analyse data using traditional digital forensic methods. This is demonstrated in Table **??**. Many IoT systems have varying functions and customized Operating Systems(OS)/file structures. There are continuous releases of different versions of IoT devices that have additional functions, even different versions of the same device will potentially have different components. For instance, in Figure 2.2 the Printed Circuit Board (PCB) of two different versions of the Amazon Echo Dot are shown, the 1^{st} generation device (see Figure 2.2a) has a debug port highlighted in red which is not present in the 2^{nd} generation (see Figure 2.2b). There are limited tools and methods to acquire data from these devices [16, 17]. Furthermore, IoT devices use protocols such as Zigbee, Z-Wave, BLE etc. and analysis of these is required to determine their forensic potential. Although this may require complex equipment and extended expertise.

Distributed Data Data is often separated and either stored locally or remotely in the cloud that is out of the user's control. Therefore, identifying the location of the data can be a challenge for investigators. In addition, data may also be stored in different countries, with different regulations and limitations on investigators having physical access [56].

Data Volatility Garfinkel [15] identified that many IoT devices use proprietary file systems and often require reverse engineering. Furthermore, many of these devices create large amounts of data, which makes it time-consuming for an investigator to identify the crucial pieces of evidence. Alabdulsalam et al. [19] noted that there is limited storage on IoT devices, as a result the data has a short lifespan as it will quickly be overwritten.

2.4.4 Investigative Frameworks

Many investigative process frameworks exist, which describe the steps and processes for investigators to follow during an investigation. Over time many digital forensic frameworks were proposed, evolving to meet the changing cybercrime attack landscape and technology. Early frameworks were developed as generic investigative frameworks such as the DFRWS and the Integrated Digital Investigation Process (IDIP) [65]. While later frameworks were developed either to improve existing frameworks or for specific purposes such as investigating cloud computing systems [66]. However, many of these were developed before the IoT paradigm was introduced. Although the field of IoT forensics is relatively new, there are already many frameworks proposed for forensic investigators to conduct investigations in an IoT environment.

One of the earliest IoT forensic investigative framework was developed by Oriwoh and Sant [57] systematically identified sources (i.e., where to look for evidence) using 3 zones. Zone 1 focused on the internal network e.g., hardware, software and networks. Zone 2 targeted the peripheral devices (between the internal and external network e.g., IDS/IPS, Gateway and Firewall). Lastly, Zone 3 "covered all hardware and software that is outside of the network in question" such as the cloud or Internet Service Providers (ISP). [57] presented Forensics Edge Management System (FEMS), "an autonomous forensic service and uses a layering approach". The network layer collects data from the sensors. This is then managed by the perception layer and the application layer is used to interface with the end-users. Collected data is only stored if it exceeds the predefined threshold.

Digital forensic triage helps investigations quickly identify devices that are likely to contain evidential data [67]. Using this concept Oriwoh et al. [57] presented the Next-Best-Thing (NBT) triage framework, to help determine the devices originally connected to the device of interest. Therefore, making it easier to identify those relevant to the investigation. Similarly, Harbawi et al. [68] proposed the Last-on-Scene(LoS) approach that stated the device representing the last node in the communication should be the first one to be investigated.

Besides providing a definition (see Section 2.4.1), Zawoad et al. [56] also proposed a Forensic Aware IoT (FAIoT) framework. Their concept was based on a centralized repository so evidence can easily be collected and analysed. In their approach they constantly monitored registered IoT devices and access to the evidence is provided to law enforcement through an API service.

Kebande et al. [69] presented the Digital Forensic Investigation framework for IoT (DFIF-IoT), that provided "the proactive process before an incident and the reactive process which occurred after". The benefit of their framework is that it complies with ISO/IEC 27043:2015 which is a standard for incident investigation. Extending their work, they proposed the Integrated Digital Forensic Investigation framework (IDFIF-IoT). This is comprised of nine components to assist pre-incident detection and the analysis of potential digital evidence from the IoT ecosystem [70].

Physically locating IoT devices at a crime scene is difficult due to their size or if it is concealed. To address this [71] proposed a process to help an investigator rapidly track devices. First, the investigator studied the environment, using a localization technique, then used frequency information to recognise the device and lastly the various interconnections are studied.

Zia et al. [72] proposed the Application Specific Digital Forensic framework which unlike previous frameworks focused on specific domains of IoT, such as smart homes, smart cities etc. In their work, they argued that the forensic processes were similar, but the extraction methods may differ depending on the IoT application. It was therefore beneficial as the collection of evidence was specific to the type of IoT application.

Extracting evidence without violating users' privacy can be challenging in the IoT context. To overcome this [73, 74] proposed the Privacy-Aware IoT Forensics (PRoFIT) framework, which took into consideration the privacy regulations ISO/IEC 29100:2011. They implemented privacy into their framework by promoting the voluntary collaboration of personal and non-personal IoT devices into the investigation process.

Overall, these approaches are theoretical frameworks that are either holistic such as [57, 69], or developed for specific use cases, such as protecting the privacy of the evidence [73].

2.5 Smart Home Forensics

In this section, we focused on the recovery of forensic artifacts from consumer IoT devices, specifically smart home devices. This was because we found this to be the focus of recent research in IoT forensics such as, [27, 28, 75]. Furthermore, consumer IoT devices are more easily obtainable than devices from other IoT domains.

As research progressed, efforts have focused on the recovery of forensic artifacts from consumer IoT devices, specifically smart home devices. These have been categorised based on the data sources in an IoT ecosystem as discussed in Section 2.4.2 and we conclude with holistic approaches.

2.5.1 Device

The investigator can use two methods to access the physical ports on an PCB to acquire the firmware and file systems. The first method is non-destructive, and several techniques can be used, such as Joint Test Action Group (JTAG) which uses the ports reserved for debugging the firmware. Alternatively, data could also be accessed via a serial connection called Universal Synchronous Receiver-Transmitter (UART). The second method, "Chip-Off" analysis requires the physical removal of the flash memory chips from the PCB, and a dedicated chip reader to read the data [76]. Although this method is only used when the first method is not viable. In addition, physical analysis is technically challenging, time-consuming

and is destructive for the device and traces stored, especially if an investigator has limited knowledge of the field.

VAs are rising in popularity, as of 2019 3.23 billion VAs were being used worldwide [77]. For this reason, the Amazon Echo has been the first device targeted for study in digital forensic approaches. The earliest research was by Ike [33] where the authors carried out a physical extraction from the Amazon Echo device by soldering a USB connection to the UART port. But unfortunately, they did not go into detail on the type of data stored and instead focused on the methods used to extract the data. Along similar lines, Servida et al. [28] used the serial connection to acquire the memory dump of three IoT devices (Note: the authors specified they selected devices based on their popularity in Switzerland, hence they may be unavailable elsewhere). Only two of the devices had traces, they found the WiFi password on the Arlo camera. Whereas on the Wink Hub, they gained access through SSH to acquire the full filesystem and found records of the devices connected and system logs.

Hyde et al. [78] extracted data from three different versions of Amazon Alexas' (Echo, Echo Dot and Echo Show). The authors used the "Chip-off" method and extracted WiFi connection information from the device logs and registration information. Using similar techniques [75] examined the Korean based VA speaker (NAVER Clova) and retrieved a list of previous Bluetooth connections, user location information and historical information of device usage.

2.5.2 Cloud

Smart home devices store the majority of their data within the CSP that can then be accessed using APIs, with each provider using their API structure and authentication method. One of the earliest works using APIs to extract data from the cloud was published by [79]. The authors used official APIs to recover files and revisions from four CSPs: Google Docs, Dropbox, Box and Microsoft OneDrive.

Subsequently, this led Chung et al. [27] to apply the same concept to the Amazon Alexa ecosystem and demonstrated the use of unofficial APIs to obtain

data from the cloud. However, [79] argued that unofficial APIs can be unstable and unreliable, because some API calls could be missed during capture. To acquire a list of Amazon Alexa APIs, a proxy server was setup to intercept the traffic between the cloud and the device. To automate the process, they proposed a tool called Cloud-Based IoT Forensic Toolkit (CIFT), to obtain data such as the account details, activities, skills list, location information etc. However, their tool is only compatible with devices from the Amazon Alexa ecosystem. Similarly, [80] developed a tool capable of extracting data from Amazon Alexa, Google Home and Samsung SmartThings platforms. However, there were limitations: the requirement for user credentials, some CSPs do not store historical user and API owners may also change their APIs without notice. For instance, Nest has recently removed the use of their official APIs [81].

2.5.3 Mobile Device

Smart home devices have an accompanying Android or iOS mobile app for the user to configure/control the device locally and remotely. Therefore, the mobile device can store potential evidence such as commands sent by the user, voice commands related to the VA etc. Manual analysis of the traces from a mobile app is normally required, as mobile forensic software cannot be scaled to cope with many different mobile apps [82]. The typical manner to acquire the data is through logical acquisition, which captures the files and folders but not deleted data.

The authors [83] recovered data from the Android mobile apps used to control the Wink hub, Samsung SmartCam, Amazon Alexa and Nest devices. They found user account information in the Android cache files and video files in the disk cache. Besides recovering data from the Amazon Alexa cloud, [27] examined the traces from the accompanying Android and iOS mobile app. Overall, on both systems, they found few traces. However, in an SQLite database, they found data of the currently logged in user, this included files of to-do and shopping lists. Unexpectedly, this database was deleted when the user logged out of the Android app.

Dorai et al. [84] demonstrated they can extract data from the iOS mobile apps of the Google Home Mini and Nest devices (a thermostat and 2 cameras). In their work, they examined the iPhone backup files and found data related to events. These included timestamps of the physical interactions and voice commands from the Google Home Mini to the Nest thermostat. They developed a tool called the Forensic Evidence Acquisition and Analysis System (FEAAS), which generated a report of the user events created by the voice commands. In a similar study Awasthi et al. [85] examined both the iOS and Android mobile apps of the Almond smart home hub. They found historical records of previously connected devices and information on the state changes. Although the iOS and Android databases contain the same information they were structured differently.

2.5.4 Network

In this section, we provide an overview of existing research related to the network analysis of IoT devices.This includes research in the following areas: port scanning, use of encryption on the device, accessibility of the data using an HTTP proxy and the final destination of the data.

Port scanning. An investigator can scan for open ports on a consumer IoT device to gain access and exploit the device. For instance, Awasthi et al. [85] discovered that SSH on port 22 was open on the Almond smart home hub. They extracted a list of entries containing user interactions with the smart devices and detailed history of data uploaded to the cloud. Similarly, Servida et al. [28] gained access to the Wink hub using SSH to extract the filesystem.

Most research on the port scanning of IoT devices was from the security and privacy perspective. Such as the work by [86], where they collected a dataset from an internet-wide scan by the consumer security software company Avast. Analysing the dataset, they found the most common ports open on IoT devices were device discovery (Universal Plug and Play (UPnP), Multicast DNS (mDNS)) and device administration (HTTP and HTTPS) ports. Additionally, they found

that UPnP ports were prevalent, being used by nearly half of the devices. In a similar study, Alrawi et al. [87] manually scanned IoT devices on their network and found 84 open ports running various customised services, SSH, UPnP, HTTP, DNS, Telnet and RTSO. A number of these open ports were found to be vulnerable, these included ports HTTPS/443, SSH/22. Specifically, devices running UPnP had no inbuilt security and required no authentication, this allowed anyone on the same network to control the devices.

Utilization of Encryption. An essential step when analysing network traffic is to identify whether the communication is encrypted or unencrypted. A straightforward approach is considering the port of the communication. However, ports are often abused or randomly chosen, therefore this can only be an indicator. Consequently, researchers also analysed the payload looking for less random sequences. For instance, it became common to use the Shannon entropy [88] or the chi-square test [89].

Similar to our work, Loi et al. [90] examined the network traffic of 20 consumer smart home devices using the Shannon entropy test to identify if the network traffic was in cleartext, encoded or encrypted: having a high entropy is a strong indicator for encrypted payload. During their test, they utilised the Shannon entropy as a pre-step that allowed them to ignore encrypted communication. They identified encoded[1] payload, which would have been missed if a manual method had been used.

Wood et al. [89] focused on the network traffic from 4 consumer IoT medical devices, to identify cleartext between the device-to-cloud communication. In their experiments they first filtered for HTTP traffic, then identified unencrypted traffic using the chi-squared method. Finally, they used three dictionaries to search for potential sensitive medical metadata. Although all the devices used encryption, they found the devices leaked metadata within HTTP GET requests, packet headers and the device conversation IP tables. Overall, they found cleartext data showing when the device was being used.

[1]Encoded data is considered different to cleartext data. For instance, utilizing a Base64 encoding vs. sending ASCII characters.

In another study, Valente et al. [91] examined the communication between the device-to-cloud of smart toys where they found a vulnerability in one device: The "Dino", although it encrypted its network traffic it used a weak encryption scheme and the same hard-coded keys to encrypt/decrypt traffic. This allowed the researchers to obtain cleartext information and retrieved the username, cryptography algorithm and encryption key. The newest study found was by Servida et al. [28], which investigated six smart home devices and manually examined each device for cleartext traffic. As a result, they found encrypted and cleartext passwords on two devices. They also discovered two devices that communicated between the mobile app-to-device in cleartext, revealing authentication details. On another device, they found communication logs in cleartext. This contained information on events triggered, commands sent to the hub and requests made to the cloud.

HTTP Proxy. While there has been considerable research focused on unencrypted data, there has been limited research on examining the content of encrypted traffic. A study by Kayode et al. [34] intercepted the traffic of 6 smart home mobile apps as they communicated with the cloud. They were able to obtain the user ID, MAC address, home address, username and password from many of these devices. In a recent study Chung et al. [27], used a proxy server to intercept encrypted traffic from the Amazon Alexa ecosystem. They obtained a list of unofficial APIs calls to extract forensic artefacts from the cloud. However, this was under the condition where the user credentials were available.

Destination of Data. The destination IP addresses obtained from the network traffic of the IoT device is used to determine the final location of where the data terminates. This is achieved by querying an IP geolocation database such as APIgurus[2], IP2Location[3], Google Maps Geolocation API[4] [92].

The main focus of Ren et al. [93] study explored the privacy implications of IoT traffic. This was based on whether the device's location would have an

[2]https://apis.guru/
[3]https://www.ip2location.com/
[4]https://developers.google.com/maps/documentation/geolocation/overview

impact on the type of Personal Identifiable Information (PII) exposed. Additionally, they explored the different third parties the devices communicated with such as tracking and advertising services. The authors Apthorpe et al. [94] presented IoT Inspector, which allowed a consumer on the same IP network to visualise the quantity and destination of IoT traffic. However, this required the consumer to have pre-installed their tool on the network.

2.5.5 Holistic Approaches

The majority of previous work has focused on a particular forensic framework for example the Amazon Alexa ecosystem, on the other hand, several have developed generic frameworks. Meffert et al. [26] presented the Forensic State Acquisition from the Internet of Things (FSAIoT) a generic framework that gathered the state changes on IoT devices in real-time from logs. In another study, Babun et al. [95] proposed IoTDots, a concept based on data provenance. Their system provided a complete history of device actions and events and evaluated their system on the Samsung SmartThings platform. However, as both of these approaches needed to be pre-configured. They would not be useful in a post-event forensic investigation.

2.6 Smart Healthcare Forensics

After reviewing existing research on the forensic aspect of the smart home ecosystem, we now present research on smart healthcare forensics. Given the limited research in this area, we also focus on the security and privacy aspects where our research falls broadly in this category.

Smart Healthcare Security and Privacy. The security and privacy risks associated with medical devices have been highlighted continuously in previous work. Camara et al. [96] proposed the need for security solutions in new designs of Implantable Medical Devices (IMD), while Marin et al. [97] examined the security issues in IMD for instance eavesdropping on pacemakers can cause physical harm to the patient. Research by [98] Li et al. examined glucose monitoring and insulin

delivery systems and found patient information was transmitted in cleartext. Similar work by Ellouze et al. [99] developed a theoretical investigation framework for IMD devices and developed secure storage for IMD logs. Wood et al. [89] focused on the privacy aspect of consumer IoT medical devices including smart scales and blood pressure monitors. The authors found that although the network traffic was encrypted the devices would still leak sensitive medical information.

Grispos et al. [100] asserted that there was a possibility that limited data would be stored on the medical devices. However, as Glisson et al. [101] stated further research would be required to fully understand the traces left on medical devices.

Smart healthcare devices predominately use BLE to transmit data to the mobile device and many security and privacy issues have already been highlighted with BLE. In particular, Ryan [32] studied the BLE pairing protocol and demonstrated the potential to eavesdrop during the key exchange process and then using this information to bypass encryption. In another study, Cyr et al. [31] examined the security of a Fitbit device and found that during the pairing process, credentials were exposed in cleartext.

In other research Lotfy et al. [102], studied the pairing process of three wearable devices. Although the manufacturers claimed to use a secure pairing process, they found vulnerabilities existed. These included eavesdropping and Man-in-the-Middle (MiTM) attacks.

Mobile Device Forensics. The forensic analysis of the user data from an Android/iOS device is a widely studied topic. Early work relevant to our study was on the forensic analysis of wearable devices and smartwatches. Baggili et al. [103] demonstrated that user data stored on a database and Extensible Markup Language (XML) can be directly recovered from the smartwatch. In a similar study Do et al. [104] examined a smartwatch and found user data such as contact information and messages. In a further study, Kang et al. [105] focused on 2 fitness trackers and recovered user data, including the number of steps taken, heart rate and sleep cycle.

In Plachkinova et al. [106] research they focused on the security and privacy risks of user's sensitive medical data. They intended to develop taxonomy based on the security and privacy issues identified in 38 Android and iOS health/fitness apps. From the forensic perspective, [107] studied 40 popular Android health/fitness applications and recovered user credentials, geolocation data and user information. In a more recent study [108] examined 13 Android health/fitness applications and recovered user credentials and health fitness data. Additionally, they found there were privacy concerns regarding the user's workout routes being accessible to the public.

Moving away from the analysis of health/fitness mobile apps, Grispos et al. [109] analysed a consumer ECG device and recovered ECG readings and patient metadata. However, they only examined a single device and different manufacturers may vary in the type and quantity of the data stored.

2.7 Summary

In this chapter, we reviewed background literature relevant to this thesis on IoT, digital forensics, and IoT forensics. At the time of this research it was clear from the examination of related work, there were three definitions of IoT forensics. These had both similarities and differences in the terms they used. Although they all agree that the definition should include the various domains of IoT forensics, such as mobile and cloud forensics. There was no consensus regarding the type of IoT technology involved or the main processes of an IoT forensic investigation. Moreover, there was no explanation of how these definitions had been developed.

The related work highlighted the key challenges in the forensic investigation of IoT devices. For instance, the lack of technical capability and the limited tools and methods available to acquire data. However, these have yet to be explored further to determine which is of greater importance to the real-world digital forensic community (Chapter 3).

We found most research in this area proposed approaches to facilitate IoT forensics, using frameworks in which traces can be proactively collected from the device. Although these methods are valid, they lack a practical element that can

be used by forensic practitioners. The limited practical solutions that do exist were developed with a lack of scalability, in that they are only compatible with a limited number of IoT devices (Chapter 5 and 7).

Additionally, research on smart healthcare forensics had only focused on the extraction of traces from the devices' accompanying mobile application. However, there was a strong probability that traces were stored on the device, but extracting traces directly from the device for forensic investigation had rarely been explored. (Chapter 6).

Overall, this chapter has identified the lack of consistency in the existing definitions of IoT forensics. We found they covered different aspects of IoT forensics and had not been developed with the consensus of the digital forensic community. It was also unclear which IoT forensic challenges should be focused on first. Subsequently, in Chapter 3, we carry out an online survey to identify a unified definition of IoT forensics and a clearer understanding of what it encompasses. Furthermore, our survey produces a clear roadmap for future research and helps identify research opportunities for subsequent chapters.

I do not fear computers. I fear the lack of them.

— Isaac Asimov

3

IoT Forensics: Definitions, Challenges & Research Directions

Contents

In Chapter 2, we discussed the many challenges in IoT forensics such as the difficulty in acquiring and analysing IoT data/devices and the lack IoT forensic tools. There are many concepts in IoT forensics that have yet to be explored, such

as a clear definition of IoT forensics and a roadmap for current/future research. To address these challenges, in this chapter we have conducted an online survey of digital forensic practitioners for their opinions regarding these concepts. We outline the design and methods used to disseminate the survey. This is followed by a detailed analysis on the results and finally we discuss the key issues raised by the survey.

3.1 Introduction

The Internet of Things (IoT) has enabled the creation of smart devices for everyday objects e.g., smart fridges, locks and home assistants, that connect to online services. The number of human interactions with these systems creates a new paradigm of evidential data. However, the tools and technologies in digital forensics that are meant for conventional computing are not fully capable of supporting the IoT infrastructure [56]. Forensic practitioners now face challenges such as the lack of tools and methodologies for analysing IoT devices, exponential increase in data volume, variety of non-standardized IoT devices [16, 74] and datasets for training [110]. Additionally, a comprehensive survey effort has not been conducted to understand the reality of these challenges.

There are many challenges in IoT and IoT forensics that have yet to be explored. For instance, it has yet to be determined which devices are considered IoT devices, what information an investigator can get (hope to get) from these devices, or how to acquire forensically relevant data [111]. On the other hand, there are many unexplored areas in IoT forensics such as the type of resources required in education, training or the specialized skills forensic investigators need to analyse IoT devices. Lastly, there is no clear understanding of the term IoT forensics.

In this chapter we present the feedback of 70 respondents to our 20-question online survey with the intention to gain a better understanding of the key issues in IoT and IoT forensics. We aim to understand concepts such as its definition as well as current and future research directions. In detail, this chapter provides the following contributions:

- It outlines the community's interpretation and views on concepts such as: IoT in general, IoT devices, and IoT forensics including types of evidence obtainable from devices (Section 3.3.2).
- From the results of the survey and the existing definitions, we present a thorough and unified definition of IoT forensics (Section 3.5.1).
- It identifies and prioritizes current and future research challenges so that efforts can be concentrated on these issues (Section 3.3.5).

3.2 Survey Methodology and Design

3.2.1 Survey Creation and Design

The following methodology was used to create and disseminate the survey:

1. A literature review was conducted, where we found that many previous studies suggested that there were many challenges in IoT forensics. We therefore investigate which of these challenges were a priority. Furthermore, we found there were many interpretations in the definitions of IoT and IoT forensics (as highlighted in Chapter 2), however, it is important to have a standard definition. Lastly, we found that there were different types of IoT evidence (i.e., logs and sensor data). Moreover, we queried participants experience in analysing IoT devices and data to determine which type of evidence was relevant.
2. The questionnaire was iteratively refined from reviewing existing literature and based on discussions/feedback from forensic practitioners. We then performed a small pilot test through discussions with cyber security/digital forensic academics and forensic practitioners for feedback, before the final survey was undertaken.
3. We then obtained research ethics permission from the Computer Science Departmental Research Ethics Committee (DREC). This restricted us from collecting any identifiable information and disclaiming that it posed any risk or harm to subjects.

4. The survey was structured to gather demographic information about the cyber forensic respondents. It then introduced the participant to IoT and gradually led to specific questions relating to IoT forensics. This included their opinions on the definitions of IoT forensics, experience in the analysis of IoT devices or data and current and future research priorities.
5. The survey was created using the Online surveys[1] platform and distributed via various digital forensic mailing lists, groups on LinkedIn, forensic groups/forums online and private contacts. Overall, the final survey consisted of 20 questions, and is shown in Appendix A. The 20 questions included the following:
    - 10 Multiple choice
    - 5 Likert scale
    - 3 Free response
    - 1 Multiple selection (check box)
    - 1 Ranking
6. All data was exported in XLSX/CSV and downloaded for offline processing.
7. The data was then analysed and the descriptive statistics, as well as some cross-comparisons were summarized in the results section.

3.2.2 Survey Overview

The research field was relatively new, so to gather enough responses the survey was available for eight months. In total, $n = 70$ participants submitted responses. While we aspired to obtain a higher response rate, taking into consideration that the survey was conducted over a period of eight months, we deemed that the survey was distributed correctly.

Participants were provided with either nominal or ordinal options. When possible, the survey responses were logically arranged in a meaningful order to increase the quality of the results. The arrangement of ordinal data allowed trends to be identified in the responses.

[1] `https://www.onlinesurveys.ac.uk/`

The purposive sampling technique was used to recruit participants as the main goal of this method is to focus on a particular characteristic of a population that is of interest. In this case, the main target groups were individuals with a cyber forensics background, academia and practitioners with a diverse range of experiences and training. It was also of interest to examine if the perspectives of practitioners and non-practitioners were different in relation to the priorities of current and future research challenges. To ascertain what impact experience had on their views concerning priorities for current and future research. We examined whether perspectives were different between practitioners who had undertaken an investigation and non-practitioners.

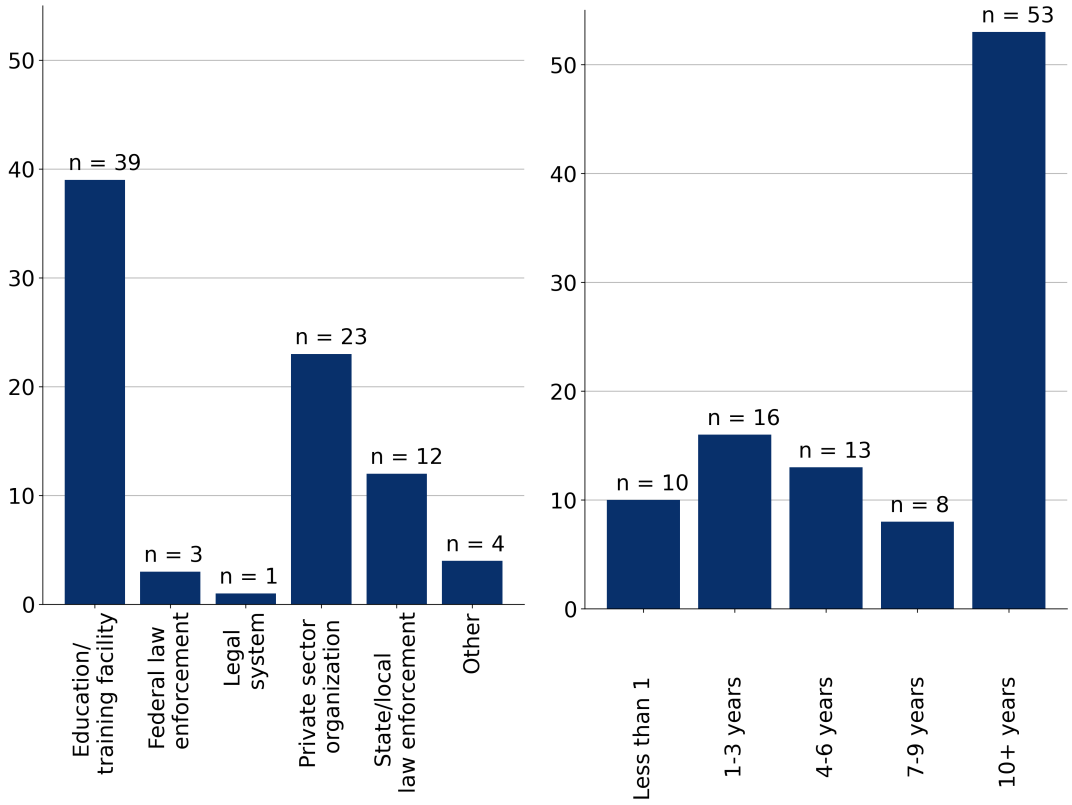
While some of the basic analysis was carried out on the survey platform, most of the analysis was accomplished in Microsoft Excel and R. To analyse free response questions, we read all responses, then identified categories based on frequently used terms, and lastly, we grouped them according to the category.

3.3 Results

The results of survey were divided into six sections: demographics, definitions, IoT forensic investigations, evidence, current and future challenges in IoT forensics which are reflected in the upcoming subsections.

3.3.1 Demographics

Q1-Q6 were in regard to demographics, these included questions on the country, age, gender, year of experience in cyber forensics, occupation category and primary occupation. The results of the demographics questions are presented in Table 3.1, where we found the majority of our ($n = 70$) participants were either American or European (80%). The largest age group with 32% respondents were 35-44 years old. Most of the respondents worked in education or a training facility (Figure 3.1b). 53% of the respondents had 10 years or more of experience (as shown in Figure 3.1a).



(a) Respondents by occupation category **(b)** Respondents by years of experience

Figure 3.1: IoT forensic survey demographics, by years of experience and occupation

The respondents from an education/training facility, were a mix of researchers, students, professors and industry instructors, however, 52% were professors. Practitioners accounted for 39%, working in the private sector organizations or state/local law enforcement. Thus, the input was provided by a mix of experience, academic researchers and practitioners.

The survey focused on participants experience and capabilities, which we considered the most important demographics. Therefore, we believe the gender ratio imbalance of more male respondents will not have an impact on the results.

3.3.2 Definitions

This section discusses participants' definitions of IoT, IoT devices and IoT forensics. Definitions are important as they enable us to form an accepted and common understanding of a word or subject. Thereby allowing for a mutual understanding of a topic under discussion.

Country	Percentage
Africa	4
America	41
Asia	3
Europe	39
Middle East	3
North America	6
Oceania	4
Age	
18-24	4
25-34	13
35-44	33
45-54	26
55-64	24
65+	4
Gender	
Female	13
Males	87
Years of Experience in Cyber Forensics	
Less than 1	10
1-3 years	16
4-6 years	13
7-9 years	8
10+ years	53
Occupation Category	
Education/training facility	39
Federal law enforcement	3
Legal system	1
Private sector organization	23
State/local law enforcement	12
Other	4
Primary occupation	
Industry instructor	4
Law enforcement practitioner	16
Non-law enforcement practitioner	23
Professor	23
Researcher	20
Student	7
Other	7

Table 3.1: These percentages account for 70 of the participants

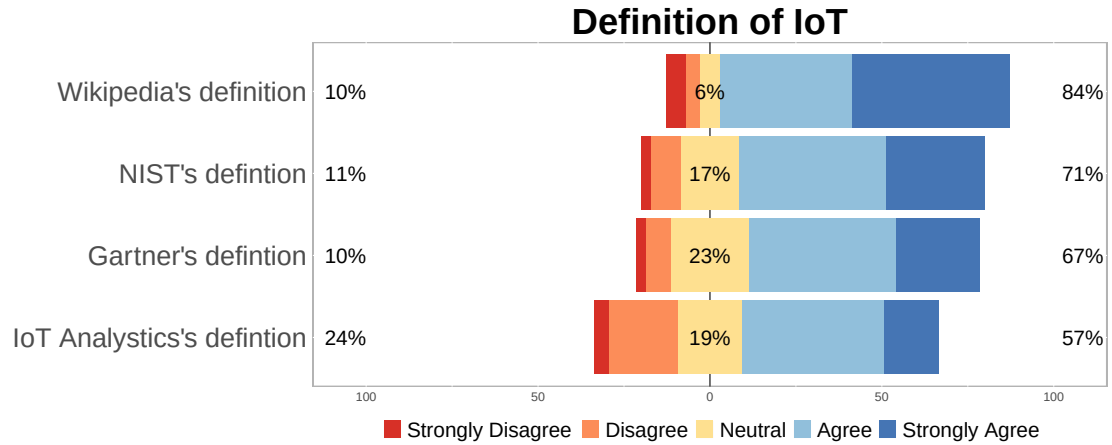


Figure 3.2: This Likert scale uses a 5 point scale ("strongly agree" to "strongly disagree"), participants were asked to rank which definitions of IoT they agree and disagree with the most.

Definition of IoT. IoT is still evolving, and many interpretations exist; therefore, the definition remains fuzzy. A sound definition can provide a better understanding of the subject, lead to further research and advance the understanding of this emerging concept. Therefore, at the beginning of the survey we asked participants in Q8: Please rate how much you agree/disagree with the following definitions of what IoT is. Four definitions from *Wikipedia, NIST, Gartner* and *IoT Analytics* were shown, and participants were asked to choose between strongly agree to strongly disagree for each. The definitions are as follows:

Wikipedia definition [112]:

> *"The Internet of things is the inter-networking of physical devices, vehicles, buildings and other items embedded with electronics, software, sensors, actuators, and network connectivity which enable these objects to collect and exchange data."*

NIST definition [113]:

> *"Jeffrey Voas from NIST defines IoT using a model he calls the Network of things that relies on 4 key components to function, sensings, computing, communication and actuation. The network of things model relies on sensors, a communication channel, an aggregator, and the cloud in order to function. Network of Things Model."*

Gartner definition [114]:

Type of IoT Device	Percentage
Self-Driving Cars	70
Smart Phones	43
Drones	61
Hub based Internet devices (ex. Internet connected light bulbs)	89
Internet connected sensors (ex. sensors monitoring power plants and factory machines.)	86
Bluetooth peripherals	44
Wearables (ex. Smart Watches)	74
Home Assistants (Google Home, Amazon Alexa, etc)	89
Smart Home Appliances	97

Table 3.2: This multiple selection checkbox question allowed participants to select as many choices they considered applied, participants were asked to select options that they considered to be an IoT device

> *"The Internet of Things is sensors and actuators embedded in physical objects that are linked through wired and wireless networks."*

Analystics definition [115]:

> *"The Internet of Things (IoT) is the network of physical objects that contain embedded technology to communicate and sense or interact with their internal states or the external environment."*

All 70 participants answered this question, and the results are shown in Figure 3.2. Most participants (84%) agreed or strongly agreed with the definition found on Wikipedia, followed by the definition given by Jeffrey Voas from NIST (71%). However, there was no clear favourite and although the definitions were different, participants found all definitions valid.

IoT Devices. The next set of questions focused on IoT devices where participants were asked Q9: what do you consider to be an IoT device? From the options offered *Smart Home Appliances* received the highest responses (97% agreed) closely followed by *Hub based internet devices* (89%), *Home Assistant* (89%) and *Internet connected sensors* (86%). Detailed results are summarized in Table 3.2.

Next, we asked participants Q10: Does an IoT device need to be constantly connected to the Internet? Out of the 69 answers, the majority (84%) responded with '*no*'. However, this showed that there is still uncertainty/disagreement as to what exactly constituted as an IoT device.

Definition of IoT Forensics. We first asked participants Q7: Do you consider IoT forensics as a subdomain of cyber forensics? Where 90% answered with '*yes*'. This showed that it is widely accepted as a subdomain and similar to other subdomains such as mobile forensics, network forensics etc., where the name of the domain defined the focus area. However, the ubiquitous nature of IoT devices makes it difficult to draw a definite conclusion to what domains are included. The 10% that said '*no*' to this question were primarily from the private sector.

Next, in a free response question, we asked participants Q12: In your own words, what is IoT forensics? 64 participants responded and we received a variety of answers.

We analysed the free response questions using the methodology by [116] and carried out the following:

1. First we gained familiarity with the 64 responses, we then excluded 23 of the responses as they were either too short (e.g., *"digital forensic in the context of IoT"*), were not specific to IoT forensics or were unclear (e.g., *"internet connected devices, either through Bluetooth or sensors or TCP/IP"*).

2. We then organised the remaining 41 responses (which often included similar terms). Next, we extracted keywords and identified categories (with direct quotes to support those categories shown in Table 3.3)

3. Finally, we placed the responses under a general category as shown below:

    - Collection, preservation, analysis and presentation of IoT devices and data (17)[2]
    - Embedded devices (7)

[2]The parenthetical numbers indicate the number of times a general idea was mentioned.

Categories	Example responses from the participants
Collection, preservation, analysis and presentation of IoT devices and data	P16: The *identification, preservation, collection, analysis and presentation* of data from any 'smart' device.
	P19: IoT forensics the *smart forensic methodology for investigate IoT based crimes* into more deeper level.
Embedded Devices	P45: *Embedded device* in electronic log and data investigations
	P30: Forensics in IoT devices Like *embedded devices* which have (in)direct connection over the internet to a central resource for example for Management purposes or control of the devices states.
Non-Traditional Computing	P3: Examination of *IP connected devices which are not regular computers, PCs, smart phones, tablets etc.*
	P29: Forensic of the equipment which have internet connection data and which is not classified as computer or *mobile forensic*
	P59: IoT forensics is the analysis of *host and network based artifacts of non-traditional devices* that have little computers and the ability to communicate over the internet or within an intra-net.
IoT Network Traffic	P36: Acquisition and analysis of evidence from IoT devices. This can be evidence stored on device itself, or communicated elsewhere, and may include more abstract sources such as *network patterns*, etc.
	P54: Imaging (capturing) and analysing volatile and non-volatile storage and *network traffic related to an IoT device*
Cloud	P2: IoT is a branch of digital forensics which deals with IoT related crimes and includes investigation of the connected devices, the sensors as well as the *cloud* of data.
	P53: IoT forensics is the acquisition and analysis of at-rest device and *cloud-resident* data from a sensor or single-function device. It is also the *reverse-engineering* and analysis of data in-transit between the IoT device and its ultimate destination.
Others	P62: The biggest difference between IoT forensics and other digital forensics is that IoT forensics is largely *reverse engineering.* IoT devices are so diverse that one can never be an "IoT expert" and it is really more about the forensic analysis of unknown digital devices.

Table 3.3: The identified categories to the responses on the definition of IoT forensics and example responses

- Non-traditional computers (6)
- IoT network traffic (4)

- Cloud (3)
- Others (4)

While the above points mostly focused on network and devices, one respondent addressed necessary skills, commented that, *"IoT forensics is largely reverse engineering and is more like forensic analysis of unknown devices"* (P53). One individual wrote that *"IoT forensics is no different from other IT and remains parts of digital forensics"* (P40). While another commented that *"IoT forensics is largely a buzzword for internet connected devices"* (P9).

In addition to the above points, one individual referred to how the evidence can be used: *"Dealing with evidence that relates to the IoT devices themselves (e.g., using them for illegal purposes, evidence of hacking attacks, etc.) and also evidence that has been gathered by IoT devices but relates to crimes that are otherwise unrelated (e.g., home security system knowing that a person was home at a particular time, cameras capturing events, GPS information from wearables, etc.)"* (P37).

Interestingly, the responses focused more on the IoT devices rather than the process, outcome of an investigation or a combination of traditional forensic steps. A detailed list of the results is shown in Appendix B.

3.3.3 IoT Forensic Investigations

The next two subsections focus on investigative experiences and capabilities.

Investigative Experience. To gain a better understanding of participants' experience with IoT devices, we asked Q11: Have you ever been involved in an investigation where you have had to analyse IoT data? From the 70 responses, 62% (43) answered no, 11% (8) answered not myself but I know a colleague and 27% (19) marked yes.

In further analysis, we split the data into practitioners and non-practitioners as shown in Table 3.4. This was carried out by splitting the 70 respondents, by their response to Q11, whether they were directly involved with investigating and analysing IoT data/devices which was either a Yes (19) or No (51) (Note. the

	Investigating IoT data / devices		
Occupation	**Yes (%)**	**No (%)**	**Total (%)**
Practitioner	16	23	38
Non-practitioner	7	47	54
Other	4	3	7
Total (%)	27	73	100

Table 3.4: Cross-correlating participants occupation vs whether they have been involved in an IoT investigation

responses where participants answered not myself, but I know a colleague, indicates that they had not been involved in an investigation and therefore we counted these responses as a no). The remaining 19 participants that had been involved in an investigation, were split into practitioners (8) and non-practitioners (11).

The respondents that stated they had either been involved in an investigation with IoT data or knew a colleague who had, were asked to include brief case details. Three respondents did not put any case details as it was either confidential or sub judice. The following responses were received along with their count:

(a) Home virtual assistant: Amazon Alexa (4), Google home

(b) Smart wearable devices e.g., Fitbit (3)

(c) Mobile device that was exposed to IoT devices (2)

(d) CCTV and DVRs (2)

(e) Raspberry Pi for research purposes

(f) Industrial IoT device

(g) IP camera that was found to have been installed without authorization

(h) Home automation system where the contractors had been accused by the homeowner of illegally accessing the home system

(i) Smart refrigerator

(j) Implanted health device

(k) Smart TV: to recover data to show there was interaction at a particular time

(l) Infotainment systems from vehicles to show GPS information

From these case details provided, it was interesting to see that there is an association with the answers provided in Section 3.3.2 (Table 3.2) on "IoT Devices" where *Smart Home Appliances* were ranked highest. Besides those highlighted by our participants, there have been several public cases that involved some of the points listed previously. For instance, in 2015 a man was charged in a murder investigation based on data extracted from an Amazon Echo [117]. In another case, data was taken from a Fitbit device and used in a murder investigation to challenge a suspect's version of the events [118].

Investigative Capability. Subsequently, we asked participants Q13: Would you consider yourself in a position to analyze an IoT device including its communications channels and network traffic? This was a free response question with 62 answers which can be summarized as follows:

- 47% (29) said they were in a position,
- 31% (19) stated they were not
- 22% (14) were unsure

As this was a free response question the 31% (19) stated that they had *"insufficient training"*, or felt they were *"not fully qualified to do so"*. Others also stated they were *"close, but encryption would obfuscate it"*. The participants that stated they were unsure mentioned that it *"depended on the type of IoT device e.g., physical interface, OS, protocols"*. Some participants thought they were *"not quite ready but working towards it"*.

For the 47% (29) who said yes, they felt capable of analysing IoT devices, we correlated them with the question *whether they had been involved in an investigation analysing IoT devices*. This showed that 11 had been involved in an investigation.

Of these 11, over half (6) were practitioners and the majority (10) had ten or more years of experience. As expected, respondents with 10+ years of experience were more confident in their capabilities.

3.3.4 Evidence on IoT Devices

Again, we used a free response question about evidence which had two parts. First, Q15: What evidence do you think we can acquire from IoT devices? Second, we asked Why is this evidence important? As with other free response questions, we summarized/clustered the results. The first part of the question received the following responses, with the general themes shown below:

1. User behaviour (6)
2. GPS data (6)
3. Sensor data (5)
4. Connection history e.g., IP addresses (5)
5. Logs (5)
6. Data on IoT device states (3)
7. Data stored on the cloud (2)
8. Telemetry data (2)
9. Data from Intrusion Detection Systems (IDS)
10. Configuration data
11. Photos, videos, audio
12. Records of conversations
13. Network related information
14. Heart rate data
15. Modified firmware
16. Events records by the IoT device

The second part of the question was to *explore the importance of the evidence* which received 21 responses. From those, we first extracted the comments that were related to why the evidence was important. The responses were then grouped based on the most frequently used terms, this also included individual direct quotations:

- Trace behavioural patterns of users, then use the data to recreate criminal activity or draw meaningful conclusions about a suspect's modus operandi (11)
- Timeline of events to trace incidence in question (2)
- To correlate data
- Determine unexpected behaviour in IoT networks
- Metadata gives clues to what happened
- Trace unauthorized interactions and data sharing
- It can tell us something that no human may have observed
- What the state of an environment is at the time an incident occurs
- That a person has been in a place at a certain time or that a person has done a certain action
- Aid investigations such as intelligence, criminal, accident, wrongful death, injury, corporate cases for manufacturing equipment failure/damage, and larger scope such as electronic voting

To provide a better insight into whether the respondents were speaking from experience, we cross-correlated the results from this question (Q15) with the responses from Q11 (see Section 3.3.3). Table 3.5 shows the 19 participants that had previously been involved in an investigation with IoT data, but there was no observable pattern that experience was relevant.

3.3.5 Current Challenges

This section identifies the important challenges currently faced by investigators in the IoT environment with regards to legislation, cloud, training, software and acquisition.

Current Issues. To learn about the current issues, we asked participants Q14: How would you rank the following issues IoT forensics is facing today (to rank nine topics with 1 being the most important and 10 being the least)? Topics and results are shown in Figure 3.3.

Participant	Response
1	Logs
2	Logs
3	Sensor data
4	Connection History
5	Telemetry data
6	Configuration data
7	Network related information
8	Heart rate data
9	Logs
10	Logs
11	Events records by the IoT device
12	Sensor data
13	Sensor data
14	Logs
15	Data on IoT states
16	Data stored on the cloud
17	Data on IoT states
18	Sensor data
19	Connection History

Table 3.5: Cross correlating the participants experience vs their previous experience of being involved in an investigation

The results indicated that the important issues were technical training (82%), software (80%) and education (78%) whereas funding, legal issues and cloud data storage currently had less relevance.

Technical training held the highest priority and this reinforces why over half of the participants felt they were unable to undertake and analyse an IoT device.

Subsequently, we asked the participants to Q18: Please rate how much you agree or disagree with the following statements. These statements were regarding legislation, data encryption, cloud and the security of IoT devices. The exact statements and the results are presented in Figure 3.4, which showed that 83% of the respondents thought IoT devices currently have weak security.

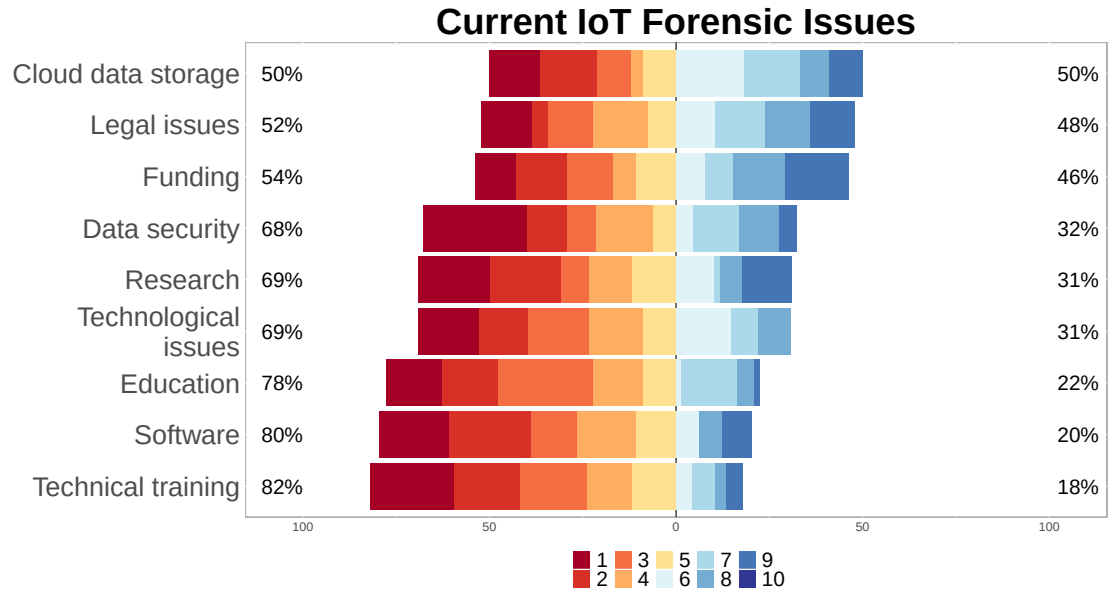


Figure 3.3: On a scale of 1 to 10 (1 being the most and 10 being the least important) participants had to rank the various challenges listed on the left.

Similar to our findings, a study by Hossain et al. [119] pointed out several other security challenges in the IoT landscape. These included constraints on memory and energy on IoT devices, which make it hard to port computationally expensive cryptographic algorithms. With the growing number of IoT devices, current security schemes lack scalability meaning these schemes would not be suitable for IoT devices. These challenges could be contributing factors as to why IoT devices have weak security.

The findings that showed IoT devices have weak security is expected, as there have been many documented security flaws found on consumer IoT devices such as baby monitors, home security cameras or smart thermostats [120]. However, legislation is being introduced to combat weak security on IoT devices, for example, in California a law is being introduced to ban default passwords [121]. In a similar effort, the UK government has released a "Code of Practice for Consumer IoT Security" document which sets out guidelines to ensure the secure design of IoT consumer products [122].

The majority of respondents strongly disagreed with the statement that *"Legislation regarding IoT forensics is currently up to date"*. This was not surprising,

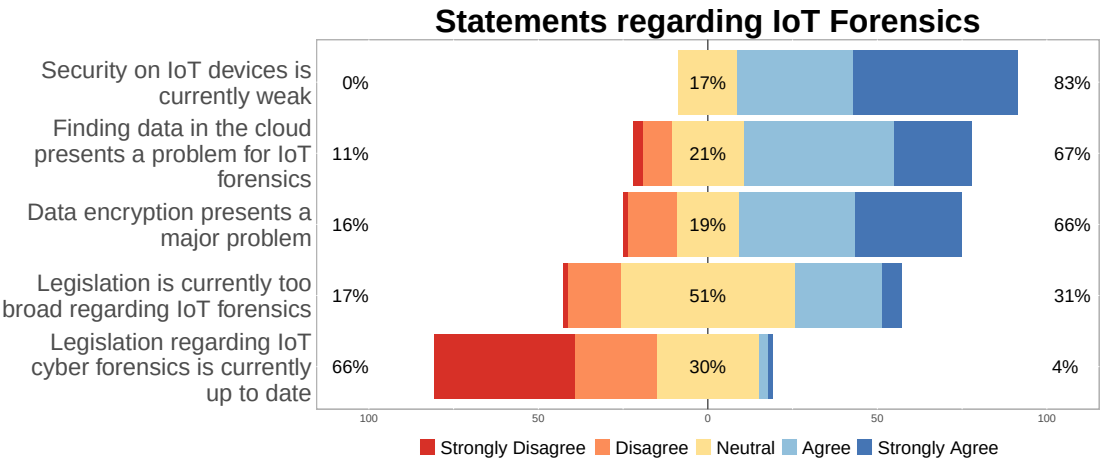


Figure 3.4: This Likert scale used a 5-point scale and asked participants if they agree or disagree with each of the statements listed on the left.

as legislation is always behind technology and is one of the main challenges in digital forensics [123]. Current legislation that governs the way data is gathered, studied, analysed and stored are aimed at traditional computing technologies [124, 125]. Legislation will need to be reviewed to include IoT devices to, standardise processes, training programs and tools

Research Directions The participants were asked Q19: Where should research be focused in order to combat current issues? The majority of participants thought that research should focus on IoT forensic tools (57%), followed by, preserving volatile data (41%) and cloud data forensics (40%).

We then cross-correlated these results with the practitioners and non practitioners. This showed that both groups thought IoT forensic tools and preserving volatile data were the most important areas. However, practitioners rated the importance of breaking encryption on IoT devices more highly than non-practitioners, who ranked cloud data forensics higher as shown in Table 3.4.

Improvements to Tools. Next, we focused on tools, where we asked participants Q16: What are the areas that are most in need of improvements to forensic and software tools used on IoT devices? The results are shown in Table 3.6 and show

Areas in need of improvement to forensic and software tools	Percentage
Data preservation	33
Data acquisition [Imaging]	73
Data encryption	30
Memory acquisition	43
Device disassembly and forensic process	63
Other	6

Table 3.6: The multiple selection checkbox question allowed participants to select as many areas in forensic and software tools which need improvements.

the respondents thought data acquisition/imaging (73%) and device disassembly forensic process (63%) were the most in need of improvement.

We continued to study whether the results would differ from the perspectives of practitioners and non-practitioners but found no difference. We also cross-correlated this against whether they had been involved in an investigation, and again found no difference.

Challenges in Acquisition. To gain a better understanding of acquisition challenges, we asked participants Q17: Which is the most challenging area to acquire IoT evidence from? Out of the 68 responses, 38% considered cloud storage as the most challenging area followed by on-device memory (28%), on-device storage (19%) and network (15%). When looking at this together with the question *areas of research to focus on to combat current issues*, one of the main areas highlighted was cloud data forensics; this was also highlighted as being the most challenging area to find (Section 3.3.5) and acquire data from. This is a clear indication that finding easier ways to identify and acquire data from the cloud should be a priority research area in cloud data forensics.

We investigated whether there was any difference of opinions between practitioners and non-practitioners which is summarized in Table 3.7. We found practitioners thought on-device storage and on-device memory as the most challenging area to acquire evidence. Whereas non-practitioners considered cloud storage and on-device memory as the most challenging.

Challenges in acquisition	Occupation		
	Practitioners (%)	Non-practitioners (%)	Total (%)
Cloud storage	7	30	37
On-device memory	11	16	27
On-device storage	13	6	19
Network	4	10	14
No answer	3	0	3
Total (%)	39	61	100

Table 3.7: Cross correlating what participants considered challenging in acquisition vs their primary occupation

3.3.6 The Future Challenges

The last question was a Likert scale question and focused on future challenges. We asked participants Q20: Please rate how much you agree or disagree that the following present a challenge to the future of IoT forensics. The results are shown in Figure 3.5. This indicated that data encryption, lack of forensic tools and trail obfuscation were the most challenging issues that needed to be focused on in the future of IoT forensics.

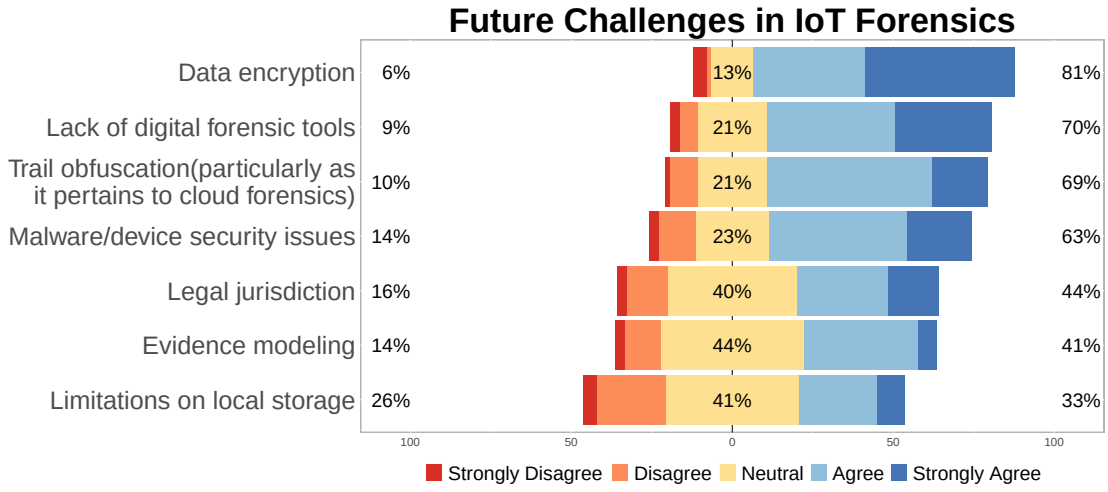


Figure 3.5: This Likert scale uses a 5-point scale ("strongly agree" to "strongly disagree"), participants were asked to rank which of the statements would be challenging for the future of IoT forensics by agree and disagree with the most.

3.4 Limitations

There were several limitations to this survey. First, this survey only had 70 participants which was a small number and answers may vary with a larger group participants. Secondly, the demographical distribution was not even, as the majority came from America or Europe, were primarily males, had 10+ years of experience and were occupied in the education/training facility sector. Thirdly, every care was taken, there was the possibility for human error while analysing the results especially the free-response questions which cannot be automated.

The survey was aimed at professionals with a cyber forensics background and distributed on forums and social media. We did not use snowballing sampling technique [126], as we assumed most professionals of any gender with this background would be participants on these platforms. The gender imbalance in the survey is representative of the gender imbalance in the industry [127, 128].

3.5 Discussion

In this section we reflect on a number of key findings from our survey and how they may influence IoT forensics research.

3.5.1 Towards a definition for IoT Forensics

In this section, we propose our definition of IoT forensics, where we identified categories, common terms used in existing definitions (Chapter 2) and responses from our survey (Section 3.3.2), thereby providing a universally accepted definition.

Reviewing existing research, we found three definitions on IoT forensics proposed by [19, 56, 57]. The issues with these definitions are that they are inconsistent and have not been developed through consultation within the digital forensic community. The terms we have used from the existing definitions have been highlighted in Table 3.8 (this was also discussed in Chapter 2).

We then analysed the results from the online survey, where we identified various categories from the responses, these included: *"collection, preservation, analysis and*

Author	Existing definitions
[57]	Has broken down IoT forensics into domains such as *cloud services*, visualization, *mobile devices*, fixed computing, sensor and RFID technologies and Artificial Intelligence (AI)
[56]	A separate branch of digital forensics and identify it as three different fields: *cloud, network and device level forensics*. The *identification, collection, organisation and presentation processes deal with IoT infrastructures to establish facts about incidents*
[19]	Separated IoT forensics as a combination of different technology zones: IoT, *network and cloud zones*

Table 3.8: Discussion on the similarities and differences between the existing definitions of IoT forensics

presentation of IoT devices and data"(17), *"embedded devices"*(7), *"non-traditional computers"*(6), *"IoT network traffic"*(4) and *"cloud"*(3).

Overall, we found there were overlaps in the terms used in the existing definitions of IoT forensics and the responses from the participants. This helped create our proposed definition of IoT forensics, which is defined as:

> *"Internet of Things (IoT) forensics is a subdomain of digital forensics and involves the collection, preservation, analysis and presentation of data obtained from IoT devices. It consists of various domains/skills including networking, embedded device, cloud, mobile, and reverse engineering."*

As a follow-up to the survey, we posted our definition of IoT forensics to various mailing lists and forums and received the following feedback. A respondent wrote that embedded devices should be changed to *"embedded systems"* as this covered traditional embedded devices. Another respondent felt that the collection of data was not only from the IoT device but also any system(s) connected to the IoT device. Based on these recommendations we updated the definition and concluded the following, with the modifications highlighted:

> *"Internet of Things (IoT) forensics is a subdomain of digital forensics and involves the collection, preservation, analysis and presentation of data obtained from or connected to the IoT device(s). It consists of various domains / skills including network, embedded systems, cloud, mobile and reverse engineering."*

Origin and motivation for the terms used in our definition: In this section we explain the origins of the terms used in our definition. Within our definition the statement *"Internet of Things (IoT) forensics is a subdomain of digital forensics and involves the collection, preservation, analysis and presentation of data obtained from or connected to the IoT device(s). It consists of various domains/skills including..."* these terms were extracted from the participants responses. We included the terms *"collection, preservation, analysis and presentation"*, as these are the main steps of the digital forensic process, and it is important to include them. In research by [19, 56, 57], they focused on dividing IoT forensics into subdomains and used the terms *"cloud, network and mobile"*. These overlapped with responses in the survey and they highlighted the various interconnected components of the IoT environment. Whereas the terms *"embedded systems"* and *"reverse engineering"* were taken solely from the responses in the survey and in contrast to [19, 56, 57] definitions, we included these terms in our definition. The term *"embedded systems"*, encompasses the type of devices that will be investigated. The term *"reverse engineering"* highlighted the importance of the specialised skills and techniques required to carry a forensic analysis of devices so diverse, which often requires the analysis and examination of previously unexplored devices. Lastly, we have excluded the terms *"AI, RFID, fixed computing, visualisation"* proposed by Oriwoh et al. [57], because they were too broad in relation to IoT forensics.

3.5.2 Readiness for IoT

In terms of their capability in analysing IoT devices, 53% of the participants felt they were either not able or unsure they were in the position to analyse IoT devices (Note. the practitioners and non-practitioners were not equally sampled). The participants that said they were not or unsure stated that this was due to *"insufficient training"* or felt they were *"not fully qualified"*. On the other hand, this was an objective question and therefore we do not know what skills the 47% (answering yes) have. In order to counteract this in the short term, we may need more training and education facilities for IoT forensics to prepare law enforcement. To facilitate this,

a discussion among experts is required on the skills needed by investigators. Maybe it is insufficient to only operate tools and we need highly technical individuals (e.g., reverse engineering experts). This is further supported by the responses from the participants when they were asked to rank the issues IoT forensics faces today, with the most important issues being technical training, software and education.

Issues regarding the lack of training had previously been highlighted in research in the field of cyber forensics. [129] found that the cyber forensic field required more Education, Training and Certification (ETC) facilities, newer programs, revision of the current curriculum and more funding in training materials should be the highest priorities.

3.5.3 Lack of IoT Forensic Tools

The findings from this survey found the majority of participants thought that research should specifically focus on the development of tools in IoT forensics (57%) (Section 3.3.5). This coincided with a study published by [60] who found that there was a lack of forensics tools in general which included IoT forensic tools. They also stated that current forensic tools were highly impractical and did not adapt rapidly to industry changes.

Although results showed 73% participants thought improvement to tools should be focused on data acquisition/imaging (Section 3.3.5). Currently only a few IoT forensics tools have been developed for specific tasks such as acquisition and analysis of cloud native artifacts. However, these tools are only compatible with specific IoT devices. For instance [27] developed and tested their tool on the Amazon Alexa ecosystem. In another [28] developed several plugins and standalone tools to acquire and analyse from specific IoT devices. Therefore, further research should be focused on developing IoT forensic tools that work reliably across a range of devices.

3.5.4 Challenges

The highest percentage of respondents at 38% thought the cloud was one of the key areas of current research, more specifically identifying and acquiring data from

the cloud. Additionally, when looking at this question combined with current issues in IoT forensics, one of the main issues found was cloud data forensics (40%) (Section 3.3.5). There are several possible reasons why the cloud is one of the main challenges; the cloud infrastructure is normally under the control of the IoT provider, the actual physical location of the data could be hidden and be distributed across cloud computing platforms [130]. A further reason could be that access to data within the cloud can only be achieved with user credentials. Although there is a potential to obtain these during the investigation, the lack of certainty can prevent progress in an investigation [57].

The findings have also highlighted that data encryption (81%) and trail obfuscation (cloud) (70%) were key areas for future research to focus on (Section 3.3.6). Both data encryption and trail obfuscation are anti-forensic techniques already known as obstacles in digital forensics. Future research should be concentrated on initiatives and strategies to address these growing problems. Data encryption presents a major challenge and has already been highlighted in previous literature as a continuous problem [49]. The results showed that participants were also worried about data encryption in the IoT domain. The focus may need to be on more non-traditional approaches. For example, [131] identified the potential of electromagnetic side channel as a method to analyse encrypted devices.

Trail obfuscation in the cloud was highlighted as one of the future challenges in IoT forensics. This could be a problem in the future when data is deleted from the cloud making it difficult to gain remote access. Data may be recoverable; however, it is challenging to recover to identify ownership [132].

Jurisdiction and legislation were regarded as a relatively low priority for current research and future challenges. However, as there was only one representative from the legal profession in the demographic, we think a follow-up survey targeting the judiciary viewpoint would be beneficial and help identify any legal issues such as privacy.

3.6 Summary

In this chapter, we presented and discussed the 70 responses to our 20-question online survey. Our results showed while participants agreed on some aspects of IoT forensics (e.g., that IoT forensics is a subdomain of digital forensics), there is also disagreement on IoT (Section 3.3.2), IoT devices (Section 3.3.2), IoT forensics (e.g., which devices are considered IoT devices and whether IoT devices are constantly connected).

Participants agreed that IoT forensics included a wide range of domains which made it difficult to draw a definite conclusion as to what was included (e.g., cloud, network and reverse engineering). When asked to define IoT forensics, respondents were mostly focused on the different types of IoT devices instead of an actual definition. Thus, to come up with the definition for IoT forensics, we extracted terms used in the existing definitions and used the responses from the survey. We then identified categories and placed them under a general category. This created a unified definition of IoT forensics, which provided a clearer and mutual understanding of the topic (Section 3.5.1).

Additionally, our results showed that participants ($\sim$ 30%) had experience in examining IoT devices/data although over 50% did not feel prepared. As a consequence, there is a need to increase education programs/technical training to prepare the workforce to deal the exponential increase in the usage of IoT devices (Section 3.3.5). Results following this survey indicated that there are many IoT forensic challenges ahead, and as technology develops these challenges will increase. We found that a high priority should be the research on the development of IoT forensic tools, specifically to develop tools that are compatible with a range of IoT devices (Section 3.5.3). In addition, current research should be aimed at data acquisition/imaging (Section 3.3.5). On the other hand, future research should focus on developing initiatives and strategies to overcome data encryption and trail obfuscation in the cloud.

The more I study, the more insatiable do I feel my genius for it to be.

— Ada Lovelace

4

The Digital Forensic Tool Landscape

Contents

After discussing existing IoT forensics research in Chapter 2 and establishing a roadmap for current and future research in Chapter 3. Our findings showed that research should focus on the development of IoT forensic tools. Consequently, we needed to establish the current state of existing digital forensic tools. In this chapter, we search for research-based tools from academic articles in pertinent digital forensic venues between 2014 and 2019. For each tool found, we establish if it has been released, its current availability and assess its usability based on various parameters such as documentation, licensing etc. Finally, we discuss the challenges in forensic tool development to establish what is missing in respect to the tools themselves and what tools are missing in IoT forensics.

4.1 Introduction

Compared to other domains, the digital forensics community has a very applied focus meaning that we are not solving problems in theory but in practice. Consequently, research endeavours frequently come with prototype implementations. For instance, [133] analysed the DJI Phantom III Drone. Their paper presented findings on the proprietary encrypted file format and presented an open-source tool that automated the parsing of the encrypted proprietary files. Additionally, tool development is encouraged by creating *Digital Forensic Competitions*, a prominent example is the Digital Forensics Research Workshop (DFRWS) Challenges: From 2005 onwards the DFRWS-conferences challenge researchers and practitioners with the goal of pushing the state-of-art in forensic tool development. Each year, following a vote for the most popular topic, a new challenge is set with a scenario, guidelines and dataset being provided. The results from the DFRWS 2014–2019 challenges have been released and are summarized in Appendix D.1. While these tools are essential to validate the proposed research, they can also provide benefits later for investigators if they encounter a similar problem. However, there have been no studies concerning the diversity, availability or quality of the published tools; nor has there been a

discussion regarding challenges and problems. Consequently, this led us to the following research sub-questions (these questions are derived from RQ2 in Section 1):

RQ2.1 What digital forensic research-based tools have been published? (categorization)

RQ2.2 Are the digital forensic research-based tools publicly freely available and/or maintained? (license, downloadable)

RQ2.3 Are the digital forensic research-based tools applicable and usable? (assessment with respect to programming style, interfaces, code quality, tested, and documentation)

Knowing what has been released and the various aspects of existing tools, this led us to a final question which will lead the discussion section:

RQ2.4 What are the current challenges in the area of digital forensic research-based tool development?

This last question focuses on aspects that we think are currently missing with respect to the tools themselves, but also on broader thoughts which we believe makes tool development more durable.

To answer these questions, we reviewed 799 research publications from various digital forensics journals and conferences between 2014 to 2019 and filtered those presenting tools. For each tool-related paper, we answered (where possible) the aforementioned questions RQ2.[1-4]. In summary, our work provided the following three contributions:

- An updated version of the subfields in digital forensics based on prior work from [134] to better cluster existing tools (Section4.6).
- An overview of recently published (in peer-reviewed venues) tools that can be used for various purposes (Section 4.6.1).
- A discussion of the challenges with published tools including recommendations on how to enhance the status quo (Section 4.7).

One may argue that technology (software and hardware) is changing rapidly and that tools are already outdated by the time they are published. Although this may be true in some cases, practitioners still encounter old hardware as well as software. For instance, Steven Watson (VTO Labs) stated during his keynote at DFRWS EU 2019, *"that they still find old file systems on recently purchased devices such as drones"*.

4.2 Related Work

The usefulness and power of open-source tools were discussed by [135] in their book "Digital forensics with open-source tools". The authors discuss software that can be helpful during the forensic process including the benefits of using open-source software. However, the book was released in 2011 and does not include recently published research. Manson et al. [136] underlined the usefulness of open-source products. In their work, they compared Sleuth Kit (software) to the commercial products EnCase and FTK. The results indicated that the tools provided the same results with varying degrees of intricacy.

Carrier [137] addressed digital forensic analysis tools and their use in a legal setting, stating that to enter scientific evidence into court, a tool must be reliable and relevant. Reliability is tested by the Daubert standard, taken from a rule of evidence regarding the admissibility of expert witness testimony [138]. This standard utilises four guiding questions to assess tools[1]:

- Testing: Can and has the procedure been tested?
- Error Rate: Is there a known error rate of the procedure?
- Publication: Has the procedure been published and subject to peer review?
- Acceptance: Is the procedure generally accepted in the relevant scientific community?

Using the guidelines of the Daubert tests, Carrier demonstrated that tools may *"meet the guideline requirements than closed source commercial tools"* by publishing

[1] Questions/itemization are copied from [137].

source code and data, which would allow the digital forensic community to examine and validate the procedures used to produce digital evidence from each tool.

Technological advancements have led to new subdomains in digital forensics. For instance [139] provided a cyber forensics classification, where they divided tools based on 2 criteria: hardware, such as large and small digital storage devices and software, such as proprietary and open-source tools. In their research, they found many tools focused only on computer forensics and mobile devices, as these were becoming more prominent in forensic investigations.

Previous work in categorising digital forensic tools has not identified groupings of research-based tools from articles or provided extensive analysis of the availability of these tools. Existing research in the taxonomy of digital forensic tools such as [140] work, presented a list of widely adopted commercial and open-source digital forensic tools (e.g., Encase, SleuthKit). Other research focused on specific subdomains including work by [141] where they presented a taxonomy of cloud forensic tools, to extract data from endpoint devices and CSP. Whereas a survey by [142] focused on anti-forensic tools (e.g., Timestomp, Transmogrify) and evaluated the effectiveness of these tools against traditional digital forensic tools. Similarly, [143] categorised anti-forensic tools and provided a comprehensive taxonomy.

The aforementioned authors have classified and categorised commercial/open-source digital forensic tools or focused on categorising tools in specific digital forensic subdomains. Additionally, their level of granularity did not cover details such as the availability of the tool or whether it had been recently maintained. Currently, there are no digital forensic tool taxonomies of recently published research-based tools in the growing field of digital forensics such as IoT and drones.

Regarding available online digital forensic tool repositories, there are two platforms that provided lists of tools: `ForensicsWiki.org/wiki/Tools` and `toolcatalog.nist.gov`. These platforms provide a combination of commercial and open-source tools. The former is a catalogue that is categorised by its functions and then on the operating platform, vendor and version. The latter is part of the National Institute for Standards and Technology (NIST) Computer Forensic Tool

Testing (CFTT) project. This allows searching by functionalities and then further refined by operating platform and various other technical parameters depending on the forensic tool. More details about these platforms are discussed in Section 4.5.1

4.3 Limitations and Scope

Although we conducted several runs, as our work was based on a manual analysis of hundreds of research articles, there was the possibility for human errors. Given the sheer number of publications, we had to limit the scope and only considered the years 2014 to 2019 from seven different venues. Lastly, when validating the availability of a tool, it may have been moved/renamed which did not allow us to find it. However, we believe that this chapter provides a solid overview of what tools are available and where to find them.

The usability of digital forensic tools can also be studied by obtaining feedback from the end-user's experience from interacting with the tools, however, this is outside of the scope of this research. Our intention was to examine the tools programming style and code quality, as this could be useful for developers who want to improve or maintain the tools.

4.4 Methodology

While these aforementioned platforms are valuable to the community, they often focused on complete software solutions (e.g., from commercial vendors). In contrast, the goal of this paper is to identify software that has primarily been developed for research purposes and to analyse any follow-up work together with other aspects of these tools.

Definition of tool (software): To analyse the availability of the research-based tools we first defined a tool (Section 4.4.1).

Tool search: The tools identified in this work were collected through extensive and exhaustive web searches as well as the searching of open-source repositories (Section 4.4.2).

Tool information: In this step, we identified variables with the tool e.g., availability of the tool, maintainability (Section 4.4.3).

Online tool searches: We searched for digital forensic tools repositories available online (Section 4.4.4).

Results: The results for the availability of the tools and online tool searches are presented (Section 4.5).

Extended taxonomy creation: Our proposed extended taxonomy was conceived based on previous work, as well as through the categorisation of each of the research-based tools identified in the articles (Section 4.6).

Types of research-based tools: We presented the various tools discovered (Section 4.6.1).

4.4.1 Definition of Tool (Software)

As a first step and before reviewing the literature, this upcoming section defines what a tool is. According to Lexico [144], a tool is defined as a device or implementation used to carry out a particular function. Sammons [145], on the other hand, has described tools as not only designed for a specific purpose but also for broader functionality.

For our research, a tool was a self-contained and provided a certain level of automation, i.e., user interaction was minimized, reduced and abstracted. For instance, a user should not have to manually find sector numbers for the tool to access the disk or translate virtual to physical addresses. Valid tools, for example, but were not limited to scripts that could be used to parse files, a code snippet that can reassemble (carve) files, or a program that helped to visualise information. Tools can be written in any programming language and were often developed by individuals or research groups. Lastly, a tool may use another program if it is automated. For instance, Autopsy is a tool supporting plugins which we also considered as tools (they add additional functionality to Autopsy).

In contrast, we did not consider comprehensive software as a tool if it was a fully-featured application such as EnCase or Cellebrite. This work also ignored articles that only explained procedures or models without any reference to implementation, some examples of which are: forensic process models, frameworks, methods, schemes and approaches. We also excluded algorithms as a tool unless it has been implemented and tested.

4.4.2 Tool search: collecting and analysing articles

Our first step was to identify tools from academic publications we focused on collecting articles from digital forensics conference proceedings and journal publications spanning the last six years (from 2014 to 2019), using the following methodology:

1. To identify the digital forensic platforms (venues are listed in Section 4.5) we used a similar selection of venues to [110]. However, we only focused on digital forensic platforms and ignored more traditional cybersecurity centred venues as our research focused purely on digital forensic tools.

2. Next, we removed all articles that were not defined as a tool based on our definition in Section 4.4.1.

3. For the remaining tool-centred articles, we grouped them based on the subfields in digital forensics (for further details see Section 4.6) and sought information about: licensing, possible updates and availability. Lastly, we reviewed the program source code, as discussed in Section 4.4.3.

4. Additionally, to verify the tools we found listed in the articles, we carried out an automated search on the PDF files using a Python script that searched for the keyword “tool/tools”.

4.4.3 Tool information

For each available tool found from Section 4.4.2, we determined the following by identifying parameters (in some cases, the data was simply unavailable).

- Code maintainability (yes/no)
- Programming language of development
- Documentation
- License
- Code review (this is discussed further in the next section)

Lastly, we reviewed the program source code, as discussed in Section 4.4.3.

Code Review The code review process started with a brief manual review to see the code was commented (we assumed that commented code had a better quality and allowed others to make modifications). Additionally, the programs' source code was analysed using an automated code review tool (Codacy[2]; see next paragraph). This is an important process in software development as it helps improve the quality of the software. Usually, automated code review tools analyse various aspects to check the quality [146], these included:

- Code complexity: Highly complex methods and classes that should be refactored
- Compatibility: Used mainly for front-end code, detected compatibility problems on different browser versions
- Errors: Code that may hide bugs and language keywords that should be used with caution.
- Security: Common security issues
- Code style: Code formatting and syntax problems. For example, variable name style, enforced use of brackets and quotation marks
- Documentation: Detected methods and classes that did not have the correct comment annotations

[2] `www.codacy.com`

- Performance: Code that could have performance problems
- Unused code: Unused variables and methods

There are a diverse number of automated code analysis tools available, however the only one that met the aforementioned requirements was Codacy. This is an open-source tool that is integrated into Github and benefits from being transparent on the type of tools used during the code review process. For example, when finding security issues in Python code, it informed the user that it used Bandit. An overall grade that ranged from A to F was given (with A being the highest), which made it easier to assess the quality of each tool. In total, there were 33 tools but the source code for one tool was unavailable. The remaining 32 tools available on Github, Bitbucket and Gitlab were cloned or forked onto a new Github account. Tools available on private websites were manually uploaded to the new Github account. Codacy was then used to analyse each repository.

4.4.4 Online tool searches

While there were many academic articles on digital forensic tool taxonomies (as shown in Section 4.2), there were also digital forensic tools available in online tool repositories. To search for these, we used the following keywords: "digital forensic tool catalogue" and "digital forensic tool repository". We focused our analysis on the first 50 results for each query. Once a forensic tool repository was identified, where possible we collected the following information: author(s), the number of tools, how the tools were categorized, whether the repository was being regularly updated and if it included any additional links to other websites related to digital forensics. The findings are presented in Section 4.5.1.

4.5 Results overview and availability of tools

This section examines the availability of the tools and whether they are maintained (see RQ2.2). Creating new tools can potentially be of great value to the digital forensic community. As tools become established, they encourage the growth of

close-knit communities of developers, providing consistently updated, patched or maintained code, bug-fixes and comprehensive documentation. We categorised the results from our tool search under the attributes of availability, code maintainability, documentation and licenses.

For this article, we reviewed 799 research publications where 62 (7%) included tools according to our definition. Almost 25% of all reviewed articles were from the Digital Forensics Research Workshops (US & EU) where 27 out of the 199 articles included tools; followed by Digital Investigation[3] (journal) where 20 tools were found out of 212 articles. Results from other venues included: (a) International Conference on Digital Forensics & Cyber Crime (ICDF2C[4]) had 3 tools out of 44 articles; (b) Association of Digital Forensics, Security & Law (ADFSL Conference) contained 3 tools out of 84 articles; (c) IFIP Working Group 11.9 on Digital Forensics had 5 tools out of 111 articles; (d) International Workshop on Digital Forensics (WSDF) contained 1 tool out of 29 articles; and (e) Journal of Digital Forensics, Security and Law (JDFSL) had 3 tools out of 120 articles. An overview of all identified tools can be found here: `https://tinyurl.com/ToolsTable`

Availability of Tools. On further examination of the 62 tools, we found that currently, 53.2% (33/62) were available online (Note, we did not contact the authors to ask if they were willing to share the tool). The 33 available tools are listed in Appendix C.1 and C.2. 78.7% (26/33) of the available tools were on public repositories (Github/25 and Bitbucket/1). Five of the tools were available on private/personal websites (e.g., downloadable from research group websites). The two remaining locations were Source Forge and Dropbox.

By implication, this meant that 29 tools were not available, the most common (28) reason for this was an unknown source. We found one article with a link to an empty Github which appeared to have never been uploaded.

[3]Volume 30 was the last volume considered for this research.

[4]We only considered articles between 2014 and 2018 as there is / was no ICDF2C 2019.

Code Review. Figure 4.1 presents the results from using Codacy to analyse the 32 tools. We found no issues in relation to the code complexity, compatibility, documentation or unused code. Overall, all the tools had issues in their source code in regards to coding style, however, this is subjective and parameters/rules can be adjusted. 24 of the tools had security issues, 14 had errors and 4 had performance issues. If major security issues were found in the code, this could affect on the integrity of the tool. For example, Codacy highlighted one tool that used an insecure MD5 and SHA-1 hash function, instead it was recommended that SHA-512 be used [147].

To get a better understanding of the quality of the code, the tools were then scored from A to F, where 6 tools scored an A; 20 tools scored a B; 5 scored a C and lastly, 1 tool scored a D. The tool with fewest issues was developed in C++ and had only 4 issues related to security and coding style. The tool with the most issues (and scored a D) was also developed in C++, and had 1044 issues, in relation to coding style, errors, security and compatibility.

Summary: Overall, 26 out of the 32 tools scored A or B. All tools had issues with coding style, but no issues were found relating to code complexity, compatibility, documentation or unused code. The tool with the most issues was developed in C++ and had issues concerning coding style, errors, security and compatibility.

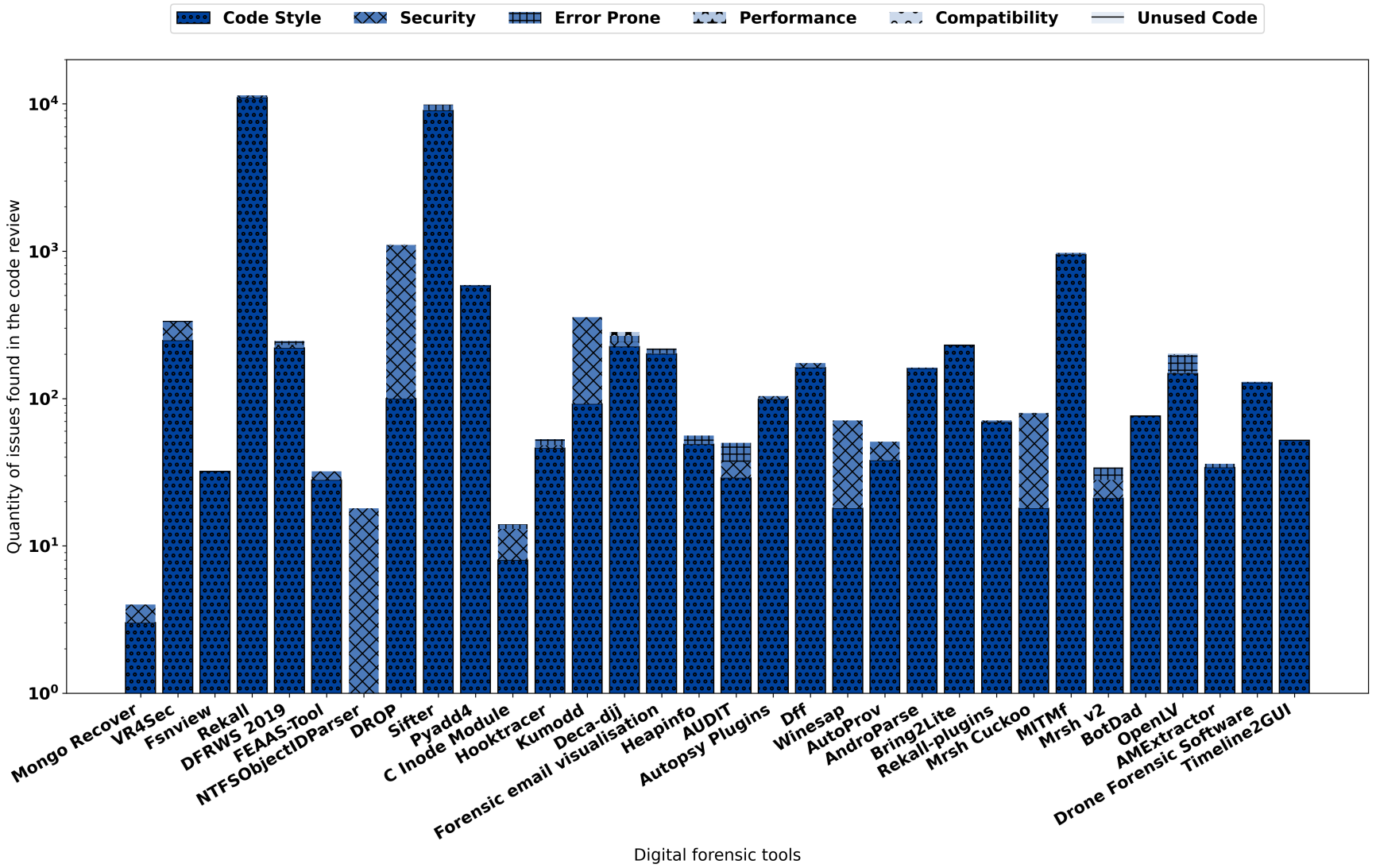


Figure 4.1: Results from using the automated code review tool Codacy to analyse the 32 digital forensic tools

Code Maintainability. We examined the 22 tools developed between 2014-2018 to determine whether they were being maintained after development/last modified, the 11 tools from 2019 were excluded as they had only been recently developed. 72.7% (16/22) tools had not been maintained after development; for 2 we were unable to identify the date they were last modified; only 4 had recent updates.

Programming Language. Tools were developed in various programming languages, ranked in descending order of popularity; Python, C++, Java, C, Golang and JavaScript (note that some tools were developed to be compatible with more than one programming language). Python was by far the most popular programming language and was frequently used for plugin-development for existing tools such as Volatility[5] or Autopsy[6].

Documentation. 29 of 33 tools had some comments in their source code, which makes maintaining the code after publishing easier. However, the quality of those comments varied widely. Furthermore, 24 of the 33 tools provided documentation, the majority(20) included a description of the tool, 13 provided additional information including usage commands, 5 included installation instructions (dependencies and requirements), 2 had configuration information, 1 had details of execution of the tool and 1 had results from the output of the tool. There was no standard format for developers to follow when documenting their tools, however, several sources have provided a recommended format. For example, in Github, a Readme file should be included with the following information: project name, description, table of contents, installation, usage, contributing, credits and license [148].

License Online repositories such as Github allow the licensing of source code so that they are open source [149]. However, only 11 out of 33 tools had licenses, this meant that to redistribute the code for the tools without licenses, permission from the copyright holder would be required and this is a cumbersome process.

[5] `https://www.volatilityfoundation.org/`
[6] `https://www.autopsy.com/`

4.5.1 Tools from Online Searches

To identify existing digital forensic tool taxonomies and online tool repositories, we performed an online search with the keywords "digital forensic tool catalogue" and "digital forensic tool repository". Overall, we identified five sources that provided repositories of forensic tools: Four were websites that provided links to a variety of tools from different categories and one source focused only on acquisition and analysis tools. While these tool taxonomies included commercial and open-source tools, they did not include research-based tools or recently developed tools.

Computer Forensic Tools & Techniques Catalogue was a comprehensive list of forensic tools. As stated on the website, "*the primary goal of the tool catalogue is to provide an easily searchable catalogue of forensic tools and techniques*". It contained 37 different categories of forensic tools across all disciplines, e.g., disk imaging, live response, drone forensics and infotainment & vehicle forensics.

Forensic Wiki. contained about 110 commercial and open-source tools that were grouped into 10 main categories based on their functionalities, e.g., disk imaging tools, memory imaging, memory analysis and network forensic. Some categories had subcategories, e.g., disk imaging tools were split based on Operating System (OS). Note, subcategories were different for each main category, e.g., file analysis had, image analysis, software forensics, open source and file analysis tools as subdomains; there appeared to be no universal method of categorizing the tools. In addition, this website also included links to websites hosting digital forensic challenges, lists of various topics and a mixed list of datasets.

Awesome-Forensics. was a Github repository created by Jonas Plum that provided a comprehensive list of forensic tools. This repository was created on March 29^{th}, 2016 and as stated on the website provided a "*curated list of awesome free (mostly open-source) forensic analysis tools and resources*". This

resource contained about 60 tools in 14 categories and was last updated on April 29^{th}, 2019. We found the repository also has a list of tools for various other domains, including penetration testing, malware analysis, hacking and honeypots.

DFIR Training. listed 27 main categories with four of those categories being loosely categorised under "Forensic Utilities" and by OS (Linux, Mac, Misc. and Windows) where Misc. and Windows have further subcategories. We could not find any information on how often this repository is updated or when last updated: `https://www.dfir.training/dfirtools/advanced-search`

DF Tools Catalogue. (`dftoolscatalogue.eu`) was developed by the Institute of Legal Information Theory and Technique of the National Research Council of Italy. This repository differentiates digital forensic tools for acquisition and those for analysis where it listed 464 acquisition tools and 1045 analysis tools that were clustered by forensic domains, e.g., network, mobile and malware. We could not find information how often this repository is updated or when last updated.

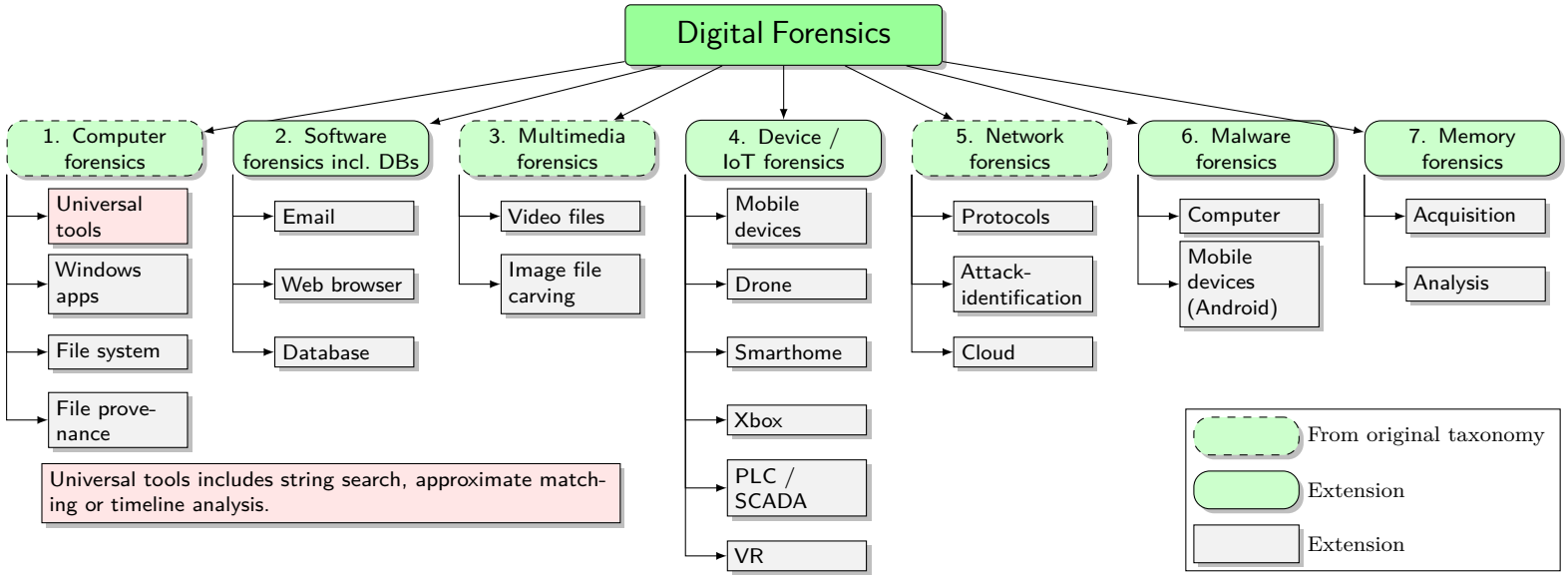


Figure 4.2: An overview of Digital Forensics subfields which expands on the recommendations made by [134]. Compared to the original version, we combined Software and Database forensics, renamed Device forensics to Device/IoT forensics and added two new subfields: Malware forensics and Memory forensics.

4.6 Extended Taxonomy of Research-Based Tools

To answer RQ2.1, this section focuses on published tools to allow practitioners to easily identify an appropriate tool for a given task. In order to cluster them, it is important to follow a digital forensics taxonomy. For example, [150] introduced the notion of abstraction layers for digital forensics at multiple levels, identifying layers for physical media, media management, file system analysis, application analysis, network analysis and memory analysis. He proposed that *"the purpose of digital forensic analysis tools is to accurately present all data at a layer of abstraction and format that can be effectively used by an investigator to identify evidence"*.

Previous attempts at digital forensic tool taxonomies such as [140] presented a list of popular commercial and open-source digital forensic tools often used in traditional forensic investigations. While other taxonomies have focused on specific domains in digital forensic, which included anti-forensic tools [142, 143] and cloud tools [141]. These past attempts were useful in classifying and categorising digital forensic tools but are now outdated and did not include new areas of digital forensics (e.g, device/IoT forensics, malware forensics). Instead, we have designed an extended version of a widely accepted digital forensic tool taxonomy, devised by [134]. They identified six subfields of digital forensics: (1) Computer, (2) Software (Note. where software is defined as *"uncovering potential evidence through examining software"*, (3) Database, (4) Multimedia, (5) Device and (6) Network forensics.

The NRGD [134] taxonomy is more intuitive and easier to navigate[7] and provided a pillar to our extended taxonomy. However, it is outdated and does not include recent digital forensic domains, for instance device/IoT forensics, malware forensics. As the digital forensic field continues to expand and new subfields are introduced, a new taxonomy of the domain must reflect this trend and provide an evolving structure. Therefore, we presented a modified and extended version of the NRGD taxonomy as shown in Figure 4.2 to include recently published research-based tools. Additionally, we go into a detailed granularity of the tools such as availability and whether the tools have been recently maintained.

[7]Note: this is a personal preference and we do not say that one is better than the other.

In our updated classification taxonomy, due to the lack of available database forensics tools we placed databases under the software category. We also extended the taxonomy to include malware and memory forensics categories, as they both constitute distinct and significant realms within the scope of digital forensics. *Universal tools* (shown in red) are classified under computer forensics as we did not consider them to be a direct subfield of digital forensics. Additionally, the origin of these tools often started in traditional computer forensics (e.g., string search, timeline analysis). The subfields for each category (shown in grey) correspond to paragraphs in the upcoming subsections. Note, despite the updated version, there are some tools that fit in several subfields. In these cases, we decided to place the work *only* in the subfield we identified as the most appropriate.

It may be argued that Devices/IoT forensics could be placed into other sections as they typically contained well-known OSes and operating environments (e.g., Linux, BSD). However, we chose to have a separate subfield as, (a) they also frequently used proprietary file formats, (b) they were often in their own ecosystem involving Smartphone Apps, on-device information, cloud data and network communication that needed to be correlated and (c) they required less common procedures like chip-off or JTAG forensics.

4.6.1 Types of Research-based Tools

In the following subsections we discuss the various tools we discovered.

4.6.2 Computer Forensics

This section discusses tools relating to the various OS as well as universal tools that can be used independently of the OSes.

Windows Apps In Windows 8, Microsoft introduced lightweight sand-boxed applications called “apps” to provide a full range of functionality on touch-enabled displays. Murtuza et al. [151] created a tool called MetroExtractor to gather static and volatile forensic data from Windows apps but did not consider inactive hibernation files.

File Provenance Tools were available to extract the provenance of a file, but these require prior installation before provenance events are generated. [152] presented the tool AutoProv that used temporal artifacts to rebuild the file provenance of a Windows forensic image. However, the limitation in the tool was that there was the possibility a user could alter the provenance to incriminate another party.

File System The Linux Ext4 file system saves metadata in the superblock or the group descriptor table. Previously this information was required to understand the correct structure of the file system, to recover the data. Dewald et al. [153] created the tool as a module within the Sleuthkit framework that did not require this information. Instead file carving method and metadata analysis is used to reconstruct the file.

Allocated files that have an OID (Note, these were created when a file was opened and then saved by an application) can be used to reconstruct user activity, with multiple entries often created. The tool NTFSObjIDParser was developed to automatically parse these entries, however it is possible the OID could be manipulated using anti-forensic tools such as fsutil[8][154].

In the following we summarize tools that we classified as *universal tools*, as they were able to be used for various OSes.

Triage Computer evidence can be quickly examined to assess the likelihood of acquiring artefacts using triage tools. Vidas et al. [155] produced a tool called OpenLV to quickly identify relevant evidence, however, the tools require administrator permissions. A second triage tool called Forensic2020 runs within a bootable environment and provides the ability to view results while the evidence is processed. They evaluated the tool and found slight changes were made to the filesystem [156]. The next tool was designed to allow an investigator to quickly view the devices that had recently synced with the seized devices. Hargreaves et al. [157] have proposed a plugin called Sync Triage which extracted details and provenance of recently

[8] `https://commandwindows.com/fsutil.htm`

connected devices. Although they only tested this on a selected number of apps, it was compatible with several OSes, including Windows, iOS and Android. To support investigators with minimal technical knowledge in searching the disk for evidence in files, emails and documents, Karabiyik et al. [158] developed the tool Advanced Automated Disk Investigation Toolkit (AUDIT) integrated with commonly used command-line tools onto a Graphical User Interface (GUI).

Non-relevant data When a bitwise copy of a system is taken, non-relevant or sensitive data must be securely deleted from the image. To address this Zoubek et al. [159] proposed a selective deletion method and implemented this as a plugin within the Digital Forensic Framework (DFF). They only evaluated their tool on the NTFS file system The tool required manual intervention by the investigator, hence the authors stated that not all irrelevant data can be identified.

Timelines Timelines are used to reconstruct events and are an important step in an investigation. This can be aided using the tool Log2Timeline, a command-line tool that create a timeline by combining several log files and events from a system. To extend Log2Timeline features, Debinski et al. [160] developed Timeline2GUI which had a GUI that provided a more user friendly interface for investigators.

Approximate Matching (AM) The similarity between different files and file fragments can be measured using Approximate Matching (AM) algorithms. [161] Breitinger et al. presented mrsh-v2, an AM tool for detecting similarities at both the file and file fragment levels. Mrsh-v2 was an improvement on the original mrsh algorithm by Roussev et al. [162]. Mrsh-net by Breitinger [163] is the network implementation of mrsh-v2, which had a single Bloom filter for the signature. Further work by Gupta et al.[164] created the tool mrsh-cf using a Cuckoo filter, which provided an improvement on previous AM tools in terms of processing speed, memory footprint, compression rate and false positive rate.

Litigation During the legal process, all documents related to the lawsuit including those stored in the cloud must be preserved. However, there is a possibility that documents could be manipulated by the parties involved. To mitigate this the authors [165] developed the LItigation hold eNabled Cloud Storage (LINCS) prototype, to enable auditors to verify whether the evidence had been manipulated.

String Search Search hit ranking algorithms can help investigators to search through retrieved evidence. Using this method, Beebe et al. [166] developed the tool Sifter that ranked the forensic importance of strings searches. It was capable of searching both allocated and unallocated space but only validated their algorithms on a single synthetic case.

Evidence Container Traditional evidence storage use vendor specific formats and lacked formal specifications. Schatz [167] proposed an open specified logical evidence container based on the original AFF4 evidence format [168]. It had the benefit of efficient storage of file content and could be easily extended.

Summary: In computer forensics we found tools under 3 main headings and included tools relating to various OSes. The categorised tools included: Windows apps (1), file provenance (1) and file system (2). Tools that were categorised as universal tools, as they were able to be used on various OSes were categorised under 7 main headings. These included: triage (4), non-relevant data (1), timelines (1), AM (4), litigation (1), string search (11) and evidence container (2).

4.6.3 Software Forensics including Database Forensics

As mentioned previously, "*Software forensics covers for example OS forensics, application software forensics and digital forensic analysis tools*" [134].

Email In total, three tools specific to the analysis of emails were found. (1) [169] was implemented into the Autopsy framework as a plugin called Privacy-Preserving Email Forensics (PPEF). The plugin encrypted the suspect's email account to protect their privacy. A keyword search was then performed and only emails that

matched the keyword are decrypted, but the plugin only worked if an exact keyword matched was provided. (2) The authors developed the tool InVEST to discover evidence and information in large email datasets. This was a visual analytic tool that assisted investigators in finding emails related to their case when the exact nature was unclear [170]. (3) The last tool, used interactive graphical visualization and statistical information to identify patterns in emails [171].

Web Browser Web storage is a client-side data storage containing web browser artefacts. Mendoza et al. [172] presented BrowStEx, a tool that parsed both SQLite files and XML files. This provided a timeline for an investigator to search and presented the web storage data based on a date/time frame. This prototype was evaluated only on the 64bit Windows OS and modifications will be required for this tool to be compatible with other OSes.

Database Forensics Research in this area has centred around relational databases. In all, we found five tools for recovering data from databases. Kim et al. [173] tool recovered deleted records and tables from the Extensible Storage Engine (ESE) database but did not recover deleted data if the record header was damaged. Database Image Content Explorer (DICE) recovered unallocated data from ten different Relational Database Management System (RDBMS) e.g., MySQL, Oracle [174]. Another recovery tool by Yoon et al. [175] focused on the MongoDB. The tool used metadata information, namespace file and the signature of deleted records to recover data. Meng et al. [176] developed Bring2lite, a parser that processed deleted SQLite records. On evaluating the parser, they found it had slightly higher recovery rates than competing tools, although it did not work on encrypted databases. The last tool by Wagner et al. [177] recovered data from multiple DBMSs then stored the data in the Database Forensic File Format. The DF-Toolkit was then used to view and search the extracted data, although, investigators may struggle to interpret the data as some artefacts in plain-text could be meaningless.

Summary: Software forensics including database forensics were categorised under 3 main headings, covering OS forensics, application software and digital forensic analysis tools. The tools categorised included emails (3), web browser (1) and database forensic tools (4).

4.6.4 Multimedia Forensics

This section deals with tools used to recover and analyse video and images.

Video File Formats Gloe et al. [178] developed parsers to extract file format structures and metadata from various cameras and mobile devices. They focused on the internal file structure of AVI and MP4 multimedia containers. Their tests did not comprise of multiple devices per camera model, so they were unable to verify that the outcome would be the same for other devices of the same model.

Image File Carving Due to the increasing capacity of storage disks, file carving would be an ideal triage solution. The Decision Theoretic Carving (DECA) tool was developed to specifically carve JPEG files and used decision theoretic analysis to consider the likely locations that the data could be stored [179].

Summary: In multimedia forensics, we found tools under 2 main headers, which included video file formats (1) and image file carving (1).

4.6.5 Device/IoT Forensics.

This subfield of device forensics is a broad topic, which gathers evidence from a range of small-scale devices such as mobile devices and smart home devices to larger-scale devices such as drones and games consoles.

Mobile Devices WhatsApp is the most widely used mobile messaging app. Capturing messages between the WhatsApp client and server can prove a useful part of an investigation, as they can contain important forensic artefacts. The tool ConvertPDML was developed to decrypt and view the messages exchanged between

the client and server [180]. However, the tool could only function on a rooted device with the password associated with the WhatsApp account.

The Forensic Evidence Acquisition and Analysis System (FEAAS) tool was used to recover data from the iPhone backup files such as date, time and changes created by multiple smart home devices. The results were then presented in a report that can be used to establish whether the data was produced through voice command or the mobile app. This study focused on data acquired logically, this meant the tool only worked if the data had not been deleted before examination [84].

Drone The rise in popularity of Unmanned Aerial Vehicles (UAV)s also known as drones, has led to increasing incidents of drones flying in restricted airspace and they have also been involved in criminal and terrorist activities e.g., being using to transport drugs and illegal surveillance [181]. Therefore they are a potential source of evidence in an investigation. This motivated the authors [133] to develop their tool DRone open-source Parser (DROP), which was designed to parse DAT files from the DJI Phantom III drone and then correlated the DAT files to TXT files. Using this tool, data could be correlated and linked to the user of a specific device based on the extracted metadata. This tool was limited to proprietary DAT and TXT files.

Renduchintala et al. [182] presented their tool to analyse the log parameters of the Yuneec Typhoon H and DJI Phantom 4 drones. The tool is capable of extracting, examining flight information and converting files for 3D flight trajectory visualization. However, it was ineffective on drones that had no logging capability or where the data was only stored on the mobile app.

Smart Home The increasing number of IoT devices in smart homes creates new opportunities for obtaining evidence. We identified two tools used to extract user behaviour artefacts from the cloud, that were specifically developed to analyse the VA ecosystem. The first tool was developed for the Amazon Alexa ecosystem and is called CIFT: Cloud-based IoT Forensic Toolkit, which used the unofficial cloud APIs as a method of extraction [27]. With the possibility of the cloud API being deprecated, this method of extraction could be seen as being unreliable. For

example, Nest [9] have stopped new users from accessing their API. The second tool focused on four VA speaker ecosystems from different South Korea manufacturers. Jo et al. [75] produced a tool to retrieve artefacts from the VA (NAVER Clova), that identified commands the user had previously executed. The application could only retrieve this information with a unique access token, which could only be obtained by intercepting the network traffic between the VA speaker and the cloud. The number of commands displayed is restricted to 20, which required the investigator to continuously scroll to see further commands.

In a study of multiple smart home devices such as cameras, hubs and an alarm system, evidence was extracted from multiple data sources such as the mobile device and the cloud. Servida et al. [28] developed three Autopsy plugins and one standalone tool to automate the extraction process. Two of these plugins can only be used to extract cloud credentials. The third, was used to extract cloud credentials, events and user actions. The standalone tool was used to download debug logs and parse events. These tools were the first to demonstrate that open-source forensic tools can be customised for use in IoT forensics.

XBox. In a study of the external storage of the Microsoft Xbox 360, we found one tool, the File System N-View (FSNView) that was developed to document metadata, using multiple storage system parsers on the same disk. Each parser reports its metadata in Digital Forensic XML, a storage language used to carry out differential analysis [183].

Programmable Logic Controller (PLC)/Supervisory Control and Data Acquisition (SCADA) The PLC is one of the most important components of a SCADA system, used in various industries such as water, electrical etc., to automate many production processes. Given their broad deployment in critical systems, they are a common target for cyber-attacks, with the most prominent being Stuxnet. In this study, we found three tools used to acquire and analyse data from individual PLCs manufacturers. Yau [184] presented the tool Control

[9] `https://www.home-assistant.io/blog/2019/05/08/nest-data-bye-bye/`

Program Logic Change Detector (CPLCD) and could be used to detect two types of attacks on the Siemens S7-1200 PLC; reprogramming of the PLC and alteration of the memory variables. A further tool by Denton et al. [20] was developed for the GE-SRTP network protocol PLC, that provided direct communication with the PLC to read the ladder logic program. The third tool, called Cutter, by Senthivel et al. [185] was compatible with PLCs using the Programmable Controller Communication Commands (PCCC) protocol. The tool was capable of extracting updates, ladder logic program and configuration information.

Virtual Reality (VR) The rising popularity of VR devices on the consumer market makes them a valuable source of digital evidence. To analyse the memory of a VR device, [186] have produced a Volatility plugin to automate the extraction of memory artefacts such as location, state and class of devices.

Summary: In device/IoT forensics we have categorised tools under 6 mains headings. As this is a broad topic and this included tools to gather data from a range of small-scale devices. The categorised tools included: mobile devices (2), drones (3), smarthome (2), XBox (1), PLC / SCADA (3) and VR (1).

4.6.6 Network Forensics

This section summarises a variety of different network forensic tools which includes protocols, wireless/wired security and cloud forensics.

In addition to network forensic tools, one publication by Vondráček et al. [187] presented an offensive security tool to raise awareness. Their tool Wifimitm "provides functionality for the automation of Man-in-the-Middle (MitM) attacks in the wireless environment". The package combined several existing tools and attack strategies to bypass wireless security mechanisms, such as WEP, WPA, and WPS.

Protocols Three tools related to the network analysis of IP protocols were found. The tool Hviz, was built for visualising, structuring, aggregating and correlating HTTP events between workstations [188]. However, it was identified that the network traffic from just one day, required many hours work to extract the HTTP

requests and responses. The authors [189] presented TLS Key EXtractor (TLSkex), used to detect and analyse threats from adversaries using TLS encryption to hide their attacks. This tool was used to extract the master key from a TLS connection to decrypt the TLS session. IPv6 is the most recent version of the Internet Protocol (IP), introduced to replace IPv4 which has vulnerabilities that can be exploited to create covert channels. A prototype tool was developed by Hansen et al. [190] to demonstrate that IPv6 was susceptible to covert channels attacks. However, the study was only conducted on synthetic datasets.

Attack-Identification In all, we found three tools to analyse network attacks. The authors present the Simple Set Comparison tool, that allowed investigators to identify the target of an attack [191]. The tool BotDAD used DNS fingerprinting to identify bot-infected machines on a network. Although the tool could not detect a bot outside of the investigation time or if the IP address was changed [192]. IoT botnets are rapidly increasing, hence [193] developed a Wireshark dissector to identify IoT Distributed Denial of Service (DDoS) Command and Control.

Cloud In this study, we found two tools for the recovery of evidence from consumer cloud storage services. The first tool called Synchronization Service Evidence Retrieval Tool (SSERT) remotely recovered evidence from the open-source cloud service Syncthing and provided an audit log of actions. The limitation of this tool was that it relied on the recovery of the public/private RSA key pairs and Device IDs from the suspects device [194]. Roussev et al. [195] presented a proof-of-concept acquisition tool, Kumodd that can acquire historical data of document revisions from four major cloud storage providers. A limitation to this tool was that it required the account holder's username and password.

Summary: In network forensics, we categorised tools under 3 main headings. This included network forensic tools for protocols, wireless / wired security and cloud forensics. The categorised tools included: protocols (3), attack-identification (3) and cloud (2).

4.6.7 Malware Forensics

This section includes tools used to analyse malware on computers and mobile devices (Android).

Computer Our study found two tools, both used binary analysis to analyse malware. Complier provenance information can be used to understand how malware binary was produced. [196] built BinComp that automatically recovered compiler provenance of program's binaries using the syntax, structure and semantic features to capture compiler behaviour. Using the concept of binary analysis to detect malware, [197] designed the tool BinGold, which automatically recovered semantics of a binary. However, the tool required a compiler or the installation of additional tools and the results were unlikely to be accurate if the user had packed the binary or used obfuscation techniques.

Mobile Devices (Android) Android is the most popular OS for mobile devices and malware is becoming more prevalent on these devices. We found three tools that focused on the acquisition and analysis of Android malware. AMExtractor [198] was created to acquire and analyse the volatile memory to detect malware e.g., rootkits, but this tool did not work on Android devices running an ARM CPU. The next tool AndroParse [199], contained a database of Android application (APK) features, to speed up the analysis process. Lastly, Wei et al. [200] presented the tool Fordroid, used to analyse APKs, to automatically identify what, where and how sensitive information was locally stored.

Summary: In malware forensics, we categorised tools under 2 main headings. These were tools used to perform the analysis of malware on computers and tools for Android mobile devices. The categorised tools included: computer (2) and mobile devices (Android) (3).

4.6.8 Memory Forensics

Acquisition of the memory is used to gather a snapshot of the system in its current state, this is then analysed with the appropriate tools. The acquisition and analysis tools are further divided based on the OS for which it has been developed.

Acquisition In total, we found two tools that can be used to acquire memory. The first tool could acquire memory from the Windows OS and was implemented as two plugins for Volatility and Rekall. It can extract ACPI tables from a memory image and scan for potential rootkits [201]. The next tool, Layout Expert, acquired memory from a Linux system, however, knowledge of the memory layout was required [202]. To overcome this complication, the tool was used to predict the memory layout. This is implemented in Python due to having better accessibility to parsing libraries, but this limited the speed.

Analysis We found three tools capable of analysing memory from a Linux OS: (1) System swap files are a back-up for the OS's virtual memory system. Both Linux and Mac OS compressed the RAM to reduce the size of the system swap files. Richard et al. [203] developed four plug-ins that worked within Volatility to analyse compressed RAM. (2) The authors presented the tool sdkernel, which was used to identify the version of the OS kernel [204]. This method does not require heavy reverse engineering, however, it can only fingerprint known kernel versions. Lastly, (3) Block et al. [205] produced six Rekall plugins that automated the analysis of heap memory. Four of the plugins found the required information in the heap then placed it into separate files. The remaining two plugins can analyse and extract command history for various applications. To fully analyse the heap memory, the swap files must also be analysed, however, this was not featured in the tool.

In this study, we found four tools to analyse memory for the Windows OS: (1) Pool tag scanning is a process used in memory analysis to locate kernel objects, that are often hidden from the OS. The authors [206] created a plugin to scan the memory pool tags. (2) Uroz et al. [207] designed a Volatility plugin called Winesap,

used to analyse the Windows Auto Start Extendibility Points (ASEPs) for evidence of persistent malware. However, the tool was limited to registry ASEPs and did not check the original ASEPs that triggered the execution of some programs. (3) The next tool focused on malware code injected into executable pages that can hide its existence and manipulate other processes. To detect these Block et al. [208] created a Recall plugin called Ptenum. However, it reported all executable pages whether malicious or not, producing a large amount of unwanted data. The final tool (4) called Hooktracer, was a Volatility plugin allowing inexperienced investigators to analyse API hooks and quickly identify suspicious APIs [209].

Summary: In memory forensics, we categorised tools under 2 main headings and these were further divided based on the OS for which it had been developed and included tools to acquire the memory of a system in its current state. The categorised tools included: acquisition (Linux (1) and Windows OS (1)) and analysis (Linux (3) and Windows OS (4))

4.7 Discussion of Challenges and Opportunities

Looking closely at the last two research questions - RQ2.3 and RQ2.4 - we realised that both are closely related and we address these in the upcoming sections. We provide a clearer separation of the two questions in the Summary.

While freely available software/tools have many advantages, there are also challenges. As discussed by [210] or [211], there are several risks of using free tools/software such as the *"lack of support, documentation and updates"* or *"safety of the software"*. Indeed, when analysing the papers and tools, we realized that tools were often poorly commented or had little or no accompanying documentation. Another aspect pointed out by [210] were advertising banners which we did not encounter. The last two reasons mentioned in this blog as *"quality of the user interface"* as well as *"developer loses interest"* where especially the latter seems to be common: we noticed that although tools have been published, most of them were only used in their referenced articles. In other words, (often) neither the authors themselves continued working the tools nor have other researchers/practitioners

Category	Subcategory	Ref	Year	Tool Name	Evaluation	Availability
Computer Forensics	Windows Apps	2015	[151]	Linux compressed swap Linux duno maps	✗	✓
	File Provenance	2017	[152]	Automated Provenance (AutoProv)	✓	✓
	File System	2017	[153]	Heapinfo, Heapdump, Heapsearch, Heaprefs	✓	✓
	-	2019	[154]	NTFSObjIDParser	✓	✓
	-	2015	[151]	Linux compressed swap Linux duno maps	✗	✓
Universal Tools	Triage	2014	[155]	OpenLV	✓	✓
	-	2019	[156]	Forensic2020	✓	✓
	-	2019	[157]	Sync triage	✓	✗
	-	2019	[158]	Advanced Automated Disk Investigation Toolkit (AUDIT)	✓	✓
	Non-relevant data	2017	[159]	Digital Forensics Framework (DFF)	✓	✗
	Timelines	2019	[160]	Timeline2GUI	✓	✓
	Approximate Matching(AM)	2014	[161]	Mrsh-v2	✓	✗
	-	2015	[164]	Mrsh-cf	✓	✓
	Litigation	2015	[165]	Litigation eNabled Cloud Storage (LINCS)	✓	✗
	String Search	2014	[166]	-	✗	✗
	Evidence Container	2019	[167]	AFF4	✓	✓

Table 4.1: Overview of digital forensic research-based tools from conferences/journals between 2014-2019

utilised the tools. Of course, there are exceptions. The upcoming subsections present some ideas on how the community may address these issues.

4.7.1 Reliability of Tools

Much like crime scene forensics, digital forensics must follow a clear process for collecting, analysing and reporting such that it can be deemed admissible in a court of law. Therefore, the tools used to collect digital evidence must be subject to stringent testing to verify their reliability to produce “forensically sound" digital evidence. Despite the increasing reliance on digital forensics in criminal cases and

Category	Subcategory	Ref	Year	Tool Name	Evaluation	Availability
Software Forensics incl. (DBs)	Email	2015	[169]	Privacy Preserving Email Forensics(PPEF)	✓	✗
	-	2016	[170]	Intelligent Visual Email Search and Triage(InVEST)	✗	✓
	-	2017	[171]		✗	✗
	Web Browser	2015	[172]	Browser Storage Extractor (BrowStEx)	✗	✗
	Database Forensics	2016	[173]	EDBForensic.py	✗	✗
	-	2018	[175]	MongoDB Deleted Data Recovery Tool	✓	✓
	-	2019	[176]	Bring2lite	✓	✓
	-	2019	[177]	Database Forensic File Format (DB3F) Database Forensic Toolkit (DF-Toolkit)	✓	✗
MFs*	Video File Formats	2014	[178]	-	✓	✗
	Image File Carving	2017	[179]	Decision-Theoretic Carving(DECA)	✓	✓
Device/IoT Forensics	Mobile Devices	2015	[180]	ConvertPDML	✗	✓
	-	2018	[84]	Forensic Evidence Acquisition and Analysis System (FEAAS)	✓	✓
	Drone	2017	[133]	DRone Open source Parser (DROP)	✓	✓
	-	2019	[182]	Digital Drone Forensic application	✗	✓
	Smart Home	2019	[75]	NAVER Clova	✗	✗
	-	2019	[28]	Plugins/Autopsy plugins	✓	✓
	Xbox	2014	[183]	FSNView	✗	✓
	PLC/SCADA	2015	[184]	Control Program Logic Change Detector (CPLCD)	✗	✗
	-	2017	[20]	DRone Open source Parser (DROP)	✓	✓
	-	2017	[185]	Cutter	✗	✗
	VR	2019	[186]	Vivedump	✗	✗

* Multimedia Forensics

Table 4.2: Overview of digital forensic research-based tools from conferences/journals between 2014-2019 (continued)

Category	Subcategory	Ref	Year	Tool Name	Evaluation	Availability
Network Forensics	Protocols	2018	[187]	Wifimitm	✓	✓
	-	2015	[188]	Hviz	✗	✗
	-	2016	[189]	TLSkex	✗	✗
	-	2016	[190]	Covert6	✗	✗
	Attack-Identification	2015	[191]	Simple Set Comparison Tool	✓	✗
	-	2019	[192]	BotDAD	✓	✓
	-	2017	[193]	-	✗	✗
	Cloud	2015	[194]	Synchronisation Service Evidence Retrieval Tool (SSERT)	✓	✗
	-	2016	[195]	Kumodd	✓	✓
MalwareForensics	Computer	2015	[196]	BinComp	✗	✗
	-	2016	[197]	BinGold	✓	✗
	Mobile Devices(Android)	2016	[198]	AMExtractor	✓	✓
	-	2018	[199]	AndroParse	✓	✓
	-	2018	[200]	Fordriod	✓	✗
Memory Forensics	Acquisition	2015	[201]	-	✓	✗
	-	2016	[202]	Layout Expert	✓	✗
	Analysis	2014	[203]	-	✓	✗
	-	2014	[204]	Sdkernel	✓	✗
	-	2017	[205]	Heapinfo, Heapdump, Heapsearch, Heaprefs	✓	✓
	-	2016	[206]	-	✓	✗
	-	2019	[207]	Winesap	✓	✓
	-	2019	[208]	Winpmem, Ptenum	✓	✓
	-	2019	[209]	Hooktracer	✓	✓

Table 4.3: Overview of digital forensic research-based tools from conferences/journals between 2014-2019 (continued)

hence the need for reliable tools, the only sustained effort towards standard tool testing is NIST's Computer Forensics Tool Testing Program[10]. Additionally, no specific international standard exists for digital forensic tools. Some countries have adopted the ISO 17025 standard, which was drafted for testing and calibration laboratories but can be applied to digital forensics. In the UK, for example, the Forensic Science Regulator's code of practice stipulates that all digital forensic

[10]`https://www.nist.gov/itl/ssd/software-quality-group/computer-forensics-tool-testing-program-cftt`

service providers must be ISO 17025 compliant.

Given the obvious need for digital forensics tools to produce accurate, repeatable and reproducible results, the question arises: Do research-based tools have a place in a digital forensics laboratory that is ISO 17025 accredited? While tools are not validated individually under ISO 17025, the standard sets out requirements for equipment and validation of methods and results.

Equipment Section 6.4 of the ISO 17025 sets out the requirements for equipment and states that a laboratory shall have access to equipment including software "that is required for the correct performance of laboratory activities and that can influence the results." To meet the requirements, the laboratory must:

- Verify equipment conforms to specified requirements before being used (section 6.4.4)
- Ensure the equipment can achieve accurate results (section 6.4.5)
- Establish a calibration programme for equipment that is subject to review (section 6.4.7)
- Ensure that any equipment giving questionable results or is defective be taken out of service (section 6.4.9)

Validation of Methods and Results Section 7.2.2 of ISO 17025 requires that the laboratory validate non-standard methods to the extent necessary to meet the needs of the application or field of application. Validation requires that the laboratory implement robust testing methods through a variation of controlled parameters, comparison of results achieved with other validated tools and inter-laboratory comparisons. Furthermore ISO 17025 section 7.7 requires that laboratories have procedures for ensuring and monitoring the validity of their results.

We can conclude that a laboratory may use research-based tools as long as they can ensure tools meet the requirements set out in the standard as outlined above for equipment and validation of methods and results.

Whilst the Daubert process, discussed in Section 4.2 is not an international standard, it may still establish the reliability of a research tool to produce results that are admissible as legal evidence. To satisfy the requirements of the Daubert process, the tools from peer-reviewed articles would need to be, well documented, maintained and their development and testing processes, including code and data, clearly documented. While maintenance is difficult to enforce, it could ensure documentation and testing. Out of the 62 tools reviewed, 46 had been evaluated within the articles and inconsistencies exist in these evaluations. Some tools tested provided a detailed evaluation on accuracy and reliability using multiple datasets/test sets e.g., the accuracy in detecting malware hiding techniques using a real-world malware sample. Whereas others tested on the performance and efficiency of the tool e.g., the time taken to run on a memory dump and process artefacts. As stated by Garfinkel et al. [212], *"techniques developed and tested by one set of researchers cannot be verified by others since the different research groups use different data sets to test and evaluate their techniques"*.

With this in mind, verifying the reliability of the tools we reviewed is a non-trivial task. As a starting point, we searched the Computer Forensics Tool Testing catalogue database[11], but none of the tools found in the peer-reviewed academic publications shown in Section 4.6.1 were found, yet this fact does not negate the tools suitability for use in digital forensic investigations.

Examining the tool catalog reveals it is somewhat limited, with just 198 tool reports published since the year 2000, which falls considerably short of the number of digital forensic tools in existence and in regular use by the digital forensic community. Furthermore, as [213] pointed out, NIST's tool testing criteria is "narrowly defined" in areas such as limited test data, use of specific tool versions and confinement to single image formats. It would appear that the lack of testing methods for tool reliability not only applies to tools developed for research, but also extends to established tools. Globally accepted standards, overseen by a

[11] `https://toolcatalog.nist.gov/`

centralised governing body, are evidently required to regulate the reliability of evidence produced by digital forensic tools.

4.7.2 Usability of Tools in Real World Settings

When discussing challenges, we must also address the usability of research-based tools, specifically, their use in real-world digital forensics scenarios. We must be cognisant of the fact that the digital forensics landscape is ever changing, where advancements in technology lead to new or previously unseen digital artefacts encountered during an investigation. For this reason, no single digital forensics tool, can accomplish every task and so researchers often develop tools out of a necessity to bridge the gaps in missing functionalities or capabilities of such existing tools. Prime examples are Carvey's RegRipper [214] or Garfinkel's Bulk Extractor [215] tools. These tools were borne from the researchers need to address the capability limitations of existing tools. Tools such as these have become almost "de facto" in the toolkit of digital forensics investigators. [216] points out that the impact of research on real-world digital investigations is difficult to quantify as the current factors for evaluating research impact are primarily academic-focused, where there is a trade-off between the competing interests of the academic industry and that of the field in which any work is aimed. Evidently, a tools's worth in real-world digital investigations can only be realised if it is accepted by the digital forensic community. But how do we bridge this gap? As pointed out by [217], the digital forensic discipline serves several different communities, i.e., military, law enforcement, private sector, public sector and academia, all of which have operated in silos with little or no sharing of information or ideas. There is a definite need for a mindset of cooperation and openness, perhaps through initiatives where research-based tools are disseminated to industry actors who otherwise historically would not interact with academia. Some efforts have been made in this regard, an example of which is the Digital Forensics and Incident Response (DFIR) Review, which seeks to serve as a focal point for up-to-date community-reviewed applied research and testing in digital forensics and incident response [218]. Researchers can

make submissions to the DFIR Review, where they are reviewed by practitioners, academics and graduate students. The view of the DFIR Review is that the faster new knowledge can be produced, reviewed, and shared among the DFIR community, the better able the digital forensics industry can cope with advances in device technology, digital artefacts and criminal activities. Despite such efforts, we are still a long way from bridging the gap between academia and industry.

4.7.3 Centralized Forensic Tool Repository

When doing our research, we found many tools/resources, but these were spread across the Internet on various platforms such as Github, personal websites or repositories (as listed in Section 4.5.1). Especially in the latter case, these repositories mostly contain commercial and open-source tools, but did not include tools from peer-reviewed academic papers or those developed for the DFRWS challenges. Additionally, 'forensic catalogues' had different structures (subfields) and thus identical tools were found in different locations. On the other hand, publishers allow/ask for tools during submission. For instance, the IEEE-Dataport[12] "*is a valuable and easily accessible data platform that enables users to store, search, access and manage data*". Elsevier also encourages sharing research data[13] but positions itself broader, defining research data as: raw or processed data files, software, code, models, algorithms, protocols, or methods. While we agree that this is necessary, tools will be further spread across platforms, making it harder to find them. To counteract, the community needs to agree on a standardised taxonomy that makes clustering tools easier. Furthermore, publishers and other platforms should agree on a single platform or a way to exchange information. Lastly, there should be a discussion of whether it should be mandatory to upload source code.

4.7.4 Increasing of Reusability/Maintainability

Most tools developed for research are done opportunistically. The tool is built to fit a specific purpose in a language that is chosen by the developer, which may not prove

[12] `https://ieee-dataport.org/about-ieee-dataport`
[13] `https://www.elsevier.com/authors/author-resources/research-data`

to be the best solution. The emphasis is not on robustness or maintainability, so the tool does not undergo sufficient testing or meet the necessary standards of coding and documentation. Lack of good coding practices can lead to many issues, such as:

- Security: inconsistent, badly written code can introduce security and logic flaws that threaten the integrity of the tool.
- Reliability: If tools are not coded correctly, it can lead to inconsistencies in results from the tool. Known inconsistencies in tool outputs could cause evidence to be ruled inadmissible in a criminal case.
- Extensible: More often than not, tools are written for a standalone, single-purpose, making updates or extensions inherently complex.
- Scalability: tools developed in confined conditions using constrained resources and data may not scale well

We have demonstrated this from our results of analysing the tools we identified from academic papers, shown in Section 4.5, where out of the 32 tools found, most of the tools had security issues.

If open-source tools are to be accepted as reliable by both the digital forensic community and courts of law, they must be developed using a clear methodology that is consistent and adheres to good coding practice. However, imposing good coding practices and standards globally may prove difficult in the realm of digital forensic tool development, especially with tools built for academic research. There are solutions currently available to tool developers that will enable consistency in code structure, syntax and documentation and thus help support acceptance of the tool and ultimately its reliability. Tools such as linters and static code analysers can greatly enhance code quality. Linters help identify general programming errors such as syntactic and stylistic errors as well as known coding bugs. Static code analysers can identify similar errors besides to searching for known vulnerabilities in the code, such as identifying where code is susceptible to SQL injection or cross-site scripting attacks. These tools are, of course, up to the discretion of the tool developer, so may never be considered.

One solution to increasing the maintainability of a tool is to develop it as a plugin for an already established tool. Generally, such software has excellent community support, so a useful plugin could easily achieve widespread acceptance. For example, Sleuthkit Autopsy is a widely accepted digital forensic tool that enables plugin additions and has a vibrant community repository on GitHub [219]. They have developed plugins for forensic tasks such as viewing forensic data, data extraction, approximate matching and reporting.

4.7.5 Development of IoT Forensic Tools

Our study also showed that the development of IoT forensic tools is still in its infancy; it has highlighted the gaps in IoT forensic tool development. While we identified multiple standalone tools and plugins that worked with Autopsy, these were limited to specific manufacturers of smarthome devices [28]. There is yet to be a generic tool compatible with multiple manufacturers or a comprehensive framework. Given the sheer amount of IoT devices, it is unclear if commercial tools alone can cover the market or if practitioners will require/depend on the research community to provide tools.

In particular, there is a requirement for more tools to extract traces left by IoT applications on mobile devices. Another challenging aspect is tools to acquire and analyse the memory of IoT devices. Lastly, we require an expansion of network forensics tools: IoT devices often use other protocols such as Bluetooth, Zigbee and Z-Wave and thus more development/research is required to process these protocols. It is also important to identify and triage IoT devices on the network, although there has yet to be a tool to automate this process. Analysing the communication protocols of an IoT devices network traffic can determine whether information is encrypted or in cleartext. This information can help identify different entry points and ways to obtain information from the devices.

4.8 Summary

In this chapter we reviewed almost 800 articles to study the digital forensic tools produced through research since 2014 from various venues. We analysed all tools identified regarding various criteria which allowed us to answer four research questions:

RQ2.1: What tools have been published? (categorisation) As stated in the introduction, digital forensics is a discipline that often involves tool development. Those tools highlighted in Section 4.6.1 where we identified 62 different tools, which we categorised according to digital forensics subfields. These subfields were proposed by [134] and were extended to cover the growing areas of digital forensics, such as device/IoT forensics.

RQ2.2: Are tools freely available and/or maintained? (license, downloadable) Out of the 62 identified tools, unfortunately only 33 were found (still) to be publicly available. The majority ($\sim 80\%$) of these tools were released on Github with the potential to be used by the digital forensic community as viable tools. We closely examined whether these tools were maintained after development and found many were not. Regarding the quality of the comments within the source code, we found 29/33 made comments, although the quality of these comments varied in how comprehensive they were. While 24 out of the 33 tools provided documentation, e.g., description of the tool, installation instructions. The level of detail in the documentation for each tool varied, as there is no standard format for developers to follow and present this information.

RQ2.3: Are tools applicable and usable? (assessment with respect to programming style, interfaces, code quality, testing, and documentation) We found that, of the tools available, a high percentage had not been maintained since their published date, nor did they have sufficient documentation. The problem

lays with the fact that, in most cases, tool development has not been the focus of the research, but a by-product. With a combined lack of coding standards, limited testing, disparate repository locations and poor documentation, it is unlikely the digital forensic community will widely adopt these tools. The results from our analyse of the 32 tools support this, where we found 24 had security issues in the source code (as shown in Figure 4.1). Documentation for 29 out of the 33 tools had some comments in their source code, the quality of the comments varied as they followed no standard format. The repository locations varied with the majority stored on public repositories (Section 4.5). Commercial or established open-source software remains the de facto "go to" tools for forensic investigators and thus has become accepted as reliable sources of evidence in criminal trials. However, with a distinct lack of proper tool testing standards across the board, perhaps that the tools identified in our research may still have the potential to produce reliable evidence. In order for these tools to be recognised, we need to develop and adopt a set of guidelines and standards for developing and testing tools for quality, maintainability and ultimately reliability.

RQ2.4: What are the current challenges in the area of forensic tool development? As suggested, there is a requirement for a centralized repository specifically for tested tools, if tools are to be exposed to the wider community and thus achieve widespread acceptance. We hope that tool researchers (developers) will spend more time on code documentation and preferably develop plugins instead of stand-alone tools, thus increase re-usability.

Lastly, we found that there was an underdevelopment of tools in IoT forensics. Despite being a growing field, there are still many gaps in tool development to automate the identification and triage of IoT devices. We found the need for network forensic tools to study the various communication channels used by IoT devices, as knowing whether data is encrypted or in cleartext will help establish ways to extract information from the devices.

To me programming is more than an important practical art. It is also a gigantic undertaking in the foundations of knowledge.

— Grace Brewster Murray Hopper

5

Identifying IoT Devices on the Network

Contents

Previously in Chapter 4, we discussed the shortcomings of existing digital forensic research tools and the gaps in IoT forensic tool development. More precisely, we found a requirement for an identification approach for IoT devices on the network. To fulfil this requirement, we propose a method that uses machine learning together

with features extracted from IoT traffic to identify the type of IoT devices on the network. First, we overview existing research in IoT identification, then discuss the datasets we use to evaluate our approach and describe the features we use for identification. Furthermore, we describe three commonly used feature selection methods to reduce the feature set. Finally, we assess the computational performance on multiple hardware to determine the portability of our approach.

5.1 Introduction

The first and most essential part of any forensic investigation is the search for evidence [17]. However, identification in the IoT context is especially difficult, the investigator does not know where the data is physically stored. Furthermore, in contrast to traditional forensics where the objects of interest are normally limited to different types of computer systems or mobile devices, IoT devices are heterogeneous and small, so that it is not always possible to physically identify all the devices at a crime scene [71]. An identification approach would speed the identification stage of the forensic process and ensure that all evidence from the devices is gathered. Additionally, collecting evidence from IoT devices is time-critical, as they have limited local storage, therefore the longer the device is running the higher the risk of data being overwritten [19]. To this end finding a method to accurately, efficiently and uniquely identify the type of IoT device is invaluable to forensic investigators.

As stated by [71], the identification stage of IoT forensics has been unexplored, requiring further investigation to locate "hidden devices". Although IoT devices can be identified on the network using the devices' MAC address, this only provides information on the manufacturer of the device and not the specific model. Instead, we aim to identify the device-type which in this work is defined as a combination of manufacturer, model and software version, creating a unique "fingerprint" of the device.

Recent research has focused on identifying IoT devices by using implicit identifiers in the network packets. Prior work by [22, 23] extracted information from the packet headers and computed the statistics (e.g. minimum size of IP payload).

Meanwhile [24] observed identifiable patterns in IoT devices traffic flow, thus extracted behavioural patterns from the device (e.g., total number of downloaded, uploaded bytes). However, these approaches were inefficient as they required a substantial number of training datasets and depended on many features to train the classifier. In this chapter, we propose our machine learning approach that can identify the of IoT devices device-type. Our approach used packet header features for speed and behaviour features for accuracy combined with feature selection improving accuracy, we adopted a feature selection method to select the best performing features, improving accuracy and training the algorithm faster as there was less data. Whereas prior approaches [22, 23, 220] depended on a large feature set to train the classifier. In our approach we used a feature selection method Sequential Forward Selection (SFS): which reduce our feature set from 22 to 9 thus allowing for quicker identification. In doing so, we were able to reduce the identification runtime, retaining high accuracy but minimising the time required for identification, thereby risking less time for data on the IoT device to be overwritten. As a result, of these experiments, forensic investigators can uniquely identify each IoT device on the network by the manufacturer, model and software version instead of relying on traditional identifiers such as the MAC address or hostname. Furthermore, we demonstrated our approach was sufficiently generic to work on multiple IoT devices by testing it on devices from various manufacturers.

To address the lack of an identification method to establish the type of IoT devices on the network, this chapter furthers the state-of-the-art by:

1. Proposes a machine learning-based approach that combines both packet headers and behavioural features to identify the type of IoT devices on the network, intending to improve identification rate (Section 5.3.3).

2. Validates our approach using data collected from 22 IoT devices over a 2-hour period and identifies the use of the feature selection method Sequential Forward Selection (SFS) to reduce the feature set from 22 to 9, improving

identification runtime by approximately 80% and demonstrate an average identification accuracy of 98.2% (Section 5.4).

3. Demonstrates the portability of our approach by assessing the computational performance on 3 different types of hardware (Section 5.4.2).

In summary, we first overview existing research in IoT identification, we then discuss the datasets we use to evaluate our approach and describe the features we use for identification. Furthermore, we describe 3 commonly used feature selection methods for reducing the features set. Finally, we assess the computational performance of our approach on multiple hardware to determine its portability.

5.2 Fundamentals of Device Identification

5.2.1 Device Identification in Digital Forensics

In conventional digital forensics when computer systems or mobile devices are the source of evidence at the crime scene, it is easy to establish the number of devices to seize. An identification approach to identify the type of end-users' mobile device is already in use for mobile forensics. Typically, mobile devices are identified using the International Mobile Equipment Identity (IMEI) number, this is designed to be unique to the device, however, it can be manipulated and is considered insecure. Therefore, [221] identified devices by "fingerprints" based on Radio Frequency (RF) hardware inaccuracies. They proposed to create a unique fingerprint for each device by detecting the physical properties of a mobile phone's RF hardware. Their approach would allow forensic investigators to identify mobile devices on the Global System for Mobile Communications (GSM) network without relying on traditional identifiers such as IMEI.

Although identification has not been fully explored in the context of IoT forensics [71], the challenges were highlighted by [16] who found that forensic investigators had difficulty in locating and identifying IoT devices due to their diminutive nature.

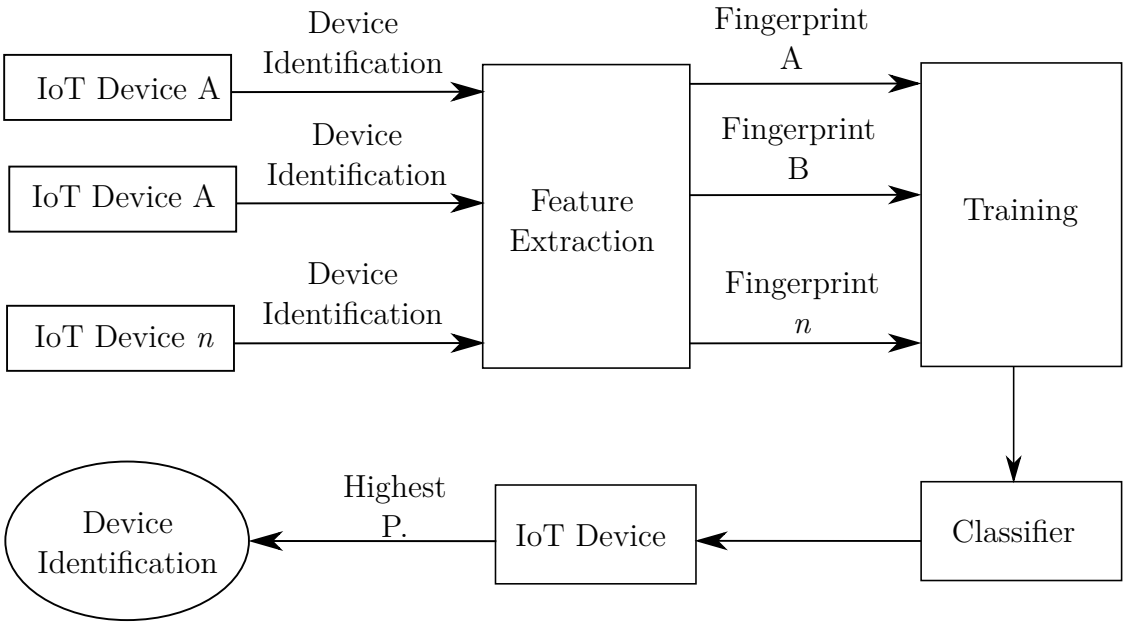


Figure 5.1: Typical workflow of using machine learning to identify IoT devices on the network [23]

5.2.2 IoT Identification

On a network, IoT devices identify each other using the MAC address, but this can only provide the device manufacturer. A recent approach is the use of implicit identifiers called features are extracted from the network traffic (or packets), which a device uses for its communication over the network.

This type of approach requires the use of machine learning and features extracted from the network traffic to create a unique "fingerprint" of the device. Figure 5.1 shows a typical flow of this, including: *training phase*, where features are extracted, a fingerprint which is a collection of features are that represent the device is generated for each device. Next, the *classification phase*, where a classifier is trained using machine learning algorithms. Next, when an unknown device needs to be identified in the *prediction phases*, a fingerprint is generated with the same set of features. The generated fingerprinting is subjected to the classifier derived earlier with a list of probabilities to be predicted (prediction phase) [23].

The earliest work using a packet header as features for IoT device identification was published by Miettinen et al. [220], where they observed that network traffic

communicated in a certain pattern during IoT device setup. Based on this finding, during the IoT device setup process they extracted 23 features from the packet header, and present IoT Sentinel framework, which achieved 50-100% accuracy across all devices. To improve on [220] method, Helsinki et al. [23] developed a sequence-based identification method, that extracted a set of packets ordered by timestamps. They extracted 90 features from the packet header (Ethernet, IP, TCP/UDP) and then calculated the statistics of each packet and improved [220] method with an average increase in the identification rate of 14%. Although [23] and [220] approaches required only two minutes of network traffic this is collected during the setup phase, which is not available post forensic incident. Additionally, a feature set as large as 90 features negatively impacts on the identification time.

Another method of IoT device identification used the behaviour patterns of the device as features. In a 6-months study, [24] Sivanathan et al., captured the network traffic from 28 IoT devices and observed that individual IoT devices exhibited identifiable patterns in their traffic flows such as activity cycles and volume patterns. Using this knowledge, they extracted 15 flow features such as the duration of a flow, the interval between flows and volume etc., then used a multi-classification process to achieve 99% accuracy. However, a multi-stage classification method is complex, and using a large training dataset and feature set can slow identification. This would hinder a forensic investigation that is time-critical especially if volatile data is stored on the device.

In related work, Bezawada et al. [222] demonstrated it was possible to use 3 main features for classification: entropy of the payload, TCP payload length and TCP window size. The authors captured network traffic from 14 IoT devices during the device setup, idle and interactions phases and observed that the length of the TCP payload was device specific, even on encrypted packets, achieved an identification accuracy of 86-99%.

More recently, Marchal et al. [223] demonstrated that it was possible to identify the type of IoT device by using the periodic background flows from IoT traffic. The authors extracted 33 features and achieved an identification accuracy of 93-99%

Tools	Description	Utilisation
Raspberry Pi 3	Wireless access point (WAP)	Capture the wireless network traffic to and from the IoT devices
TCPdump	Network traffic collection	Automate capturing network traffic
Wireshark	Packet analyser	Capture live network traffic
Argus (Audit Record Generation and Utilisation System)	Packet flow monitor and analyser	Extract flow features
Scapy	Network packet manipulation tool	Extract the packet header features
Scikit-learn	Machine learning library	

Table 5.1: Apparatus utilised in the experiments

across all device-types. However, this approach did not identify individual devices, only only the device-type based on manufacturer.

5.3 Methodology

The apparatus used in the various experiments are presented in Table 5.1. For our experiments, the following stepwise procedure was used:

IoT device selection: We selected 22 IoT devices that represented a wide range of different categories (Section 5.3.1).

Datasets: We used two different datasets, which contained network traffic obtained from 22 IoT devices. One of the datasets contained data collected by us from 11 devices (`Dataset-1`), and the other dataset contained data from an existing source (`Dataset-2`) [24]. Features were extracted from these datasets for use in our machine learning approach (Section 5.3.2).

Feature Extraction: We used various tools to extract packet header and behavioural features identified in Section 5.3.3.

Data Pre-processing: We carried out data pre-processing by replacing any missing values and normalising the dataset (Section 5.3.4).

Feature Selection: We used various feature selection methods to select the best feature set and remove any redundant (Section 5.3.5).

Classification and Evaluation Metrics: We evaluated the performance of various classifiers to identify which performed the best, using various metrics (Section 5.3.6).

Computational Performance: We evaluated the efficiency of our approach between each phase of our approach (Section 5.3.7).

Results: Finally, the results are presented, where we evaluated various feature selection methods and classifiers (Section 5.4).

No.	Category	Device Model	Non-Volatile RAM	Volatile RAM	External (SD Card)	Mobile Apps	Wireless	Ethernet
1	SH	Samsung SmartThings hub (v2)*	4GB	N\A	N\A	✓	○	◉
2		Phillips Hue bridge*	32MB	N\A	N\A	✓	○	◉
3		Vera plus hub*	256MB	N\A	N\A	✓	○	◉
4	HA	Amazon smart plug*	N\A	N\A	N\A	✓	◉	○
5		TP-Link plug(HS110)*	N\A	N\A	N\A	✓	◉	○
6		TP-Link lightbulb(LB100)*	N\A	N\A	N\A	✓	◉	○
7		Wemo plug*	N\A	N\A	N\A	✓	◉	○
8		LiFX lightbulb†	N\A	N\A	N\A	✓	◉	○
9		iHome†	N\A	N\A	N\A	✓	◉	○
10		Nest protect smoke alarm†	N\A	N\A	N\A	✓	◉	○
11	SC	TP-Link camera(KC100)*	N\A	N\A	N\A	✓	◉	○
12		D-Link camera(DCS-932LB)*	N\A	N\A	Micro 32GB	✓	○	◉
13		Netatmo welcome camera†	N\A	N\A	N\A	✓	◉	○
14		TP-Link day night cloud camera†	N\A	N\A	Micro 32GB	✓	◉	○
15		Samsung smart camera†	N\A	N\A	N\A	✓	◉	○
16		Nest dropcam camera†	N\A	N\A	N\A	✓	◉	○
17		Insteon camera†	N\A	N\A	Micro 32GB	✓	○	◉
18	VA	Amazon Echo (2nd)*	256MB	4GB	N\A	✓	◉	○
19		Google home mini*	256MB	4GB	N\A	✓	◉	○
20	M	Netatmo weather station†	N\A	N\A	N\A	✓	◉	○
21		Triby speaker†	N\A	N\A	N\A	✓	◉	○
22		PIX-STAR photo-frame†	8GB	N\A	N\A	✓	◉	○

[1] Smart Hubs (SH), Home Automation (HA), Smart Cameras (SC), Voice Assistants (VA) and Miscellaneous (M)
[*] IoT devices from `Dataset-1`
[†] IoT devices from `Dataset-2`

Table 5.2: Detailed specification of the 22 devices used in the experiments. The symbol * indicates IoT devices from `Dataset-1` and symbol † indicates to the IoT devices from `Dataset-2`.

5.3.1 IoT Device Selection

As previously stated, we selected 22 IoT devices which were either wired or had wireless connectivity. We also connected non-IP devices to the smart hubs that supported other communication protocols such as Zigbee, Bluetooth and Z-wave.

Table 5.2 lists all specifications of the devices utilised, these included information on the: device model, internal storage capacity (non-volatile and volatile RAM), external storage capacity, mobile apps and type of network connectivity. These are types of features useful to a forensic investigator examining the device. The devices were then setup using the manufacturers apps where possible. If a device was compatible with Amazon Alexa, then it was configured to connect to the service, this was done in order to emulate a consumers smart home configuration for their IoT devices, such as the Phillips Hue bridge hub and Wemo plug. We categorised the devices into 5 main categories: Smart Hubs, Home Automation, Cameras, Voice Assistants and Miscellaneous. We had two major selection criteria when we chose the devices:

Variety of families and manufacturers: The devices selected had to belong to different device manufacturers. Consequently, our selection included hubs, cameras, switches and smart speakers. This ensured that we had a represented sample of the devices available on the market.

Popularity: For each category of device we searched popular retail outlets, e.g., Amazon and selected based on price, popularity, customer rating and reviews. This ensured that we selected devices that were most likely to be used by the consumer.

5.3.2 Datasets

In the experiments we used network traffic from two sources. The first dataset (labelled `Dataset-1`) was network traffic collected from 11 IoT devices connected to our testbed, these are shown in Table 5.2. To capture the network traffic, we used the

Feature	Dataset Description
No. of IoT devices	Dataset-1- 11, Dataset-2- 11
Features	22
Behavioural feature extraction	We used the tool Argus to extract the behavioural features
Packet based feature extraction	We developed a Python script using the Scapy and Panda libraries

Table 5.3: Summary of key features of both datasets

tool TCPdump running on a Raspberry Pi 3, over a period of 10 days. To mimic a real-world investigation scenario, where human interactions with the devices would be limited as we expect the devices to be in an idle state during identification. Although some devices ran automated rules at a certain time to change the state of the lights.

We excluded devices such as that did not frequently create network traffic unless in use, for instance, the Withings scales, leaving 11 IoT devices from Dataset-2.

The second dataset (labelled Dataset-2) was created by the authors [224] and consisted of 26 weeks of network traffic from 28 IoT devices. This dataset included devices from various categories such as cameras, switches, hubs etc. We excluded devices mobile devices, laptops and devices that did not continuously create network traffic unless in use, for instance, the Withings scales. We excluded these devices and focused on classifying the device type of IoT devices only, this left 11 IoT devices from Dataset-2. A full list of the devices are shown in Table 5.2, for our experiments we randomly selected PCAP files dated $24^{th}/25^{th}/26^{th}$ September 2016 and $4^{th}/5^{th}$ October 2016. Overall, the dataset consisted of 22 IoT devices (Dataset-1 and 2). A detailed description of the datasets used in our experiments are shown in Table 5.3, and included the following information: number of IoT devices from each dataset, the methods used to extract the behavioural and packet-based features and finally a list of these features.

Sensitive data in the network traffic could reveal information about the network architecture, user identity or user information. Therefore, anonymisation techniques can be used to protect privacy by sanitising the network traffic captures so that the objects or person are no longer identifiable. Dataset-2 [24] is in the public domain, therefore data anonymisation is not required. As we collected Dataset-1,

Category	Feature (f) No.	Feature name	Description
Behavioural	1	Total bytes	Total number of bytes downloaded and uploaded
	2	TotBytes	Duration of a network flow
	3	Afr	Average network flow rate
	4	NTP interval	Period in which the device synchronises their time with the NTP servers
	5	DNS interval	Period in which the device queries a domain name
Packet header	6	Minimum size of IP payload	Smallest size of the IP payload
	7	Median size of IP payload	Middle number of the IP payload
	8	Maximum size of IP payload	Largest size of the IP payload
	9	Mean size of IP payload	Average of the IP payload
	10	1st quartile size of IP payload	Median of the lower half of the IP payload
	11	3rd quartile size of IP payload	Median of the upper half of the IP payload size
	12	Variance size of IP payload	Distribution size of the IP payload
	13	Minimum size of Ethernet packets	Size of Ethernet packets
	14	Median size of Ethernet packets	Middle number of the Ethernet packets
	15	Maximum size of Ethernet packets	Highest value of the Ethernet packets
	16	Mean size of Ethernet packets	Average of the Ethernet packets
	17	1st quartile size of Ethernet packets	Median of the lower half of the Ethernet packets
	18	3rd quartile size of Ethernet packets	Median of the upper half of the Ethernet packets
	19	Variance size of Ethernet packets	Distribution size of the Ethernet packets
	20	Minimum size of TCP window	Smallest size of the TCP window
	21	Maximum size of TCP window	Largest size of the TCP window
	22	Mean size of TCP window	Average of the TCP window size

Table 5.4: 22 features (2 categories) used for all the 22 IoT devices for device identification

a data anonymisation technique needed to be considered. However, we feel that it is not required, as our dataset was collected on a restricted testbed where no personal data was collected. Additionally, we filtered the PCAP files based on the MAC address of each device. Therefore each files contains only the network traffic for that individual device.

5.3.3 Feature Extraction

To create a set of features for IoT device identification we extracted features from the network traffic . We selected features based on the aforementioned work in the use of packet header features (e.g., mean, minimum and maximum size of the TCP window) [23, 222] and behavioural features (e.g., average network flow rate) [21]. Although previous studies in IoT identification have used these types of features separately, we combined them, because behavioural features required less training

data and together with packet features and allowed for higher accuracy.

Behavioural Features. Sivanathan et al. [24] observed that the behaviour of an IoT device created a unique pattern. This is unlike packet header features which are not unique to the device. However, to observe a pattern a longer duration was required. Overall, we selected 5 behavioural features: 3 in relation to the network flows, which were the sequence of packets between the source and the destination, and the remaining 2 features in relation to the IoT devices' traffic pattern. The 5 features were: (1) flow volume is the total number of download and upload bytes; (2) flow duration is the first and last packet of the flow; (3) average flow rate is the flow volume divided by the flow duration; (4) DNS interval is the periodic manner in which the IoT device queries a domain name. For example, the `Amazon Echo` requests the domain name example.org every $30s$. (5) NTP interval, as IoT operations are time critical; they therefore synchronise their time with publicly available NTP servers. The interval is the periodic way it synchronises with the NTP server (shown in Table 5.4). There were other behavioural features such as the type of cipher suites and the frequency of the ports. However, these are generated infrequently and are time-consuming to generate.

Packet Header Features. We extracted packet header features from each packet to gain the size of TCP window, Ethernet header and IP payload. Then for each packet we computed the following statistics: minimum, maximum, mean, median, variance and percentiles (25th and 75th). In total, this created 17 features, as shown in Table 5.4. Indistinguishable features of the device, such as source/destination IP address and port number were not selected.

Extraction. We combined the behavioural and packet header features, which created a total of 22 features. To extract these features from the network traffic, we used the MAC address to parse the devices network traffic from the PCAP files. To extract the 5 behavioural features we used the tool `Argus`[1] and then merged each

[1] `https://qosient.com/argus/`

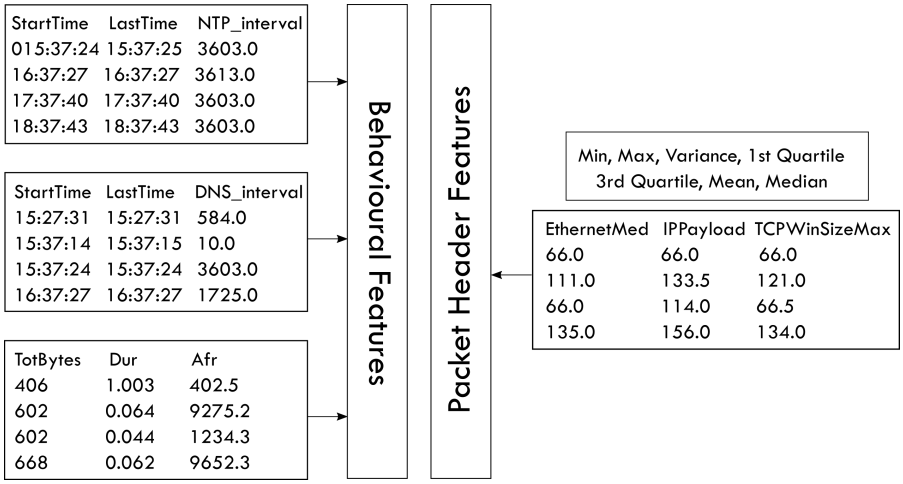


Figure 5.2: Generating packet header and behavioural features for classifier training

devices network traffic into one PCAP file and reordered the flows based on the timestamp. To extract the 17 remainting packet-header features we used a Python script developed using `Scapy`[2] and the `Panda` [225] library. The generation of the behavioural and packet header features is depicted in Figure 5.2.

5.3.4 Data Pre-processing

To determine the amount of network traffic required for identification we tested network traffic captures with different durations, which varied from 5 days, 1 day, 120, 90, 60, 30 minutes. We found that a duration of 120 minutes was optimal, as shorter times were not long enough to detect sufficient behavioural features for some devices. For example, the feature NTP interval for the `D-Link camera` took around 16 minutes to generate 1 instance.

In total, we analysed 52,245 samples of network traffic obtained from 22 IoT devices. The quantity of traffic for each IoT device varied depending on the type of device e.g., smart cameras typically created more than other devices as they were active for longer periods. Additionally, the feature lengths differed as behavioural features were generated using flows which were variable in length, while the packet header features were generated from each packet, so these feature lengths were

[2] `https://scapy.net/`

constant. This created an imbalance in the dataset, therefore any missing instances were replaced with the average value.

Next, as some classifiers performed better on numerical values, we assigned each unique device name with an integer value (Note, commonly known as one-hot encoding). For instance, `Samsung Smartcam` was assigned 1 and `TP-Link cam` 2 [226]. Additionally, some classifiers performed better on normalised data such as the k-NN classifier where the accuracy can severely degraded if the feature scales were inconsistent. Therefore, we normalised the features using `Scikitlearn StardardScale` library to have zero means and unit variances in order that each feature contributed equally.

5.3.5 Feature Selection

Feature selection is a process used to identify the best feature set and remove those that are redundant, thereby improving identification time and accuracy. In the past, feature selection has proved to be a good method of reducing the number of features, improving the classification and greatly improving the computational performance. There are two types of feature selection methods: filter and wrapper. Filter methods use statistical measures to score the correlation or dependency between the features. Whereas wrapper methods are tuned to the specific classifier to find the optimal features [227, 228]. To find discriminate features for our IoT device identification approach we evaluated the performance of the following 3 commonly used feature selection methods:

Sequential Forward Selection (SFS): This method starts from an empty subset of features and sequentially selects one at a time until no further improvement is achieved [229, 230].

ReliefF: This is a feature weight technique that estimates the feature quality by drawing instances at random, then computing their nearest neighbours. Higher weights are given to features that discriminate an instance from neighbours of different classes [231]. This method does not have to exhaustively check

every pairwise interaction, thus takes significantly less time than SFS and RFECV [232].

Recursive Feature Elimination Cross-Validation (RFECV): This feature selection method removes the weakest feature(s) until the previously selected number of features is reached [233].

5.3.6 Classification and Evaluation Metrics

We compared the performance of 4 classifiers and used cross-validation with a 10-fold stratified k-fold over 10 iterations to reduce the probability of over-fitting and ensured that each fold preserved the proportion of the devices' class. To determine the optimal parameters for the classifiers we performed a grid search, different parameters were used depending on the classifier. The parameters selected for each classifier are briefly described as follows:

k-Nearest Neighbours (k-NN): computes the distance of each test instance for the nearest k neighbours [234]. We tested the K value between 1-20 with the best results achieved with *k=5* and Euclidean distance.

Random Forest (RF): is combining the predictions from several trees, each of which is trained in isolation [235]. The main parameter for RF is the number of trees, therefore a range of [100,250,500,1000] trees were assessed, with the best parameter found at a setting of 100.

Support Vector Machine (SVM): this attempts to find a boundary that divides the data into negative and positive points so classification error can be minimized [236]. We varied the soft margin [3] constant C between 1-50 and found $C = 3$ achieved the best results.

Logistics Regression (LR): is used to predict the outcome of 2 values such as 0 or 1, yes or no, win or lose etc [237]. We achieved the best results using the following parameters *penalty=l2*, *C=1.0*.

[3]Note: The lower the C parameter the softer the margin

	High-end Desktop	Laptop	Raspberry Pi 3
CPU	Intel(R) Core(TM) CPU i7-4670K	Intel i5-1035G1 processor	64-bit 1.4GHz quad core processor
RAM	32 GB	8GB	1 GB
OS	Windows 10	Ubuntu 16.8	Raspberry Pi OS

Table 5.5: Evaluation of the computational efficiency of our approach on various hardware specifications including high-end desktop, laptop and RPi 3

Evaluation Metrics

We used 3 common classifier performance measures to evaluate our identification approach: recall, precision, and *F_1-score*. The recall is the ability of the classifier to find all positive instances and is defined as [238]:

$$Recall = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

The precision is the measurement of the probability that the classifier does not label an instance positive that is negative, and is defined as [238]:

$$Precision = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

Finally, the *F_1-score* was used to measure the effectiveness of retrieval and is used to measure the accuracy of predicting the device, which is the harmonic mean of recall and precision [239]. We used the *F_1-score* as the performance metric to measure the accuracy of prediction for each device-type, the *F_1-score* is defined as:

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

5.3.7 Computation Performance

In these experiments we evaluated the computational efficiency of our approach by measuring how much time was required between each phase of our approach, which consisted of the following 3 stages: (1) Pre-processing time: the average pre-processing time required to extract and pre-processed the packet and behavioural features. (2) Training time: the average time required to train the classifier and lastly

Feature Selection Method	Classifier	Metrics			No. of f	Average Runtime (secs)
		F_1-score Score (%)	Precision (%)	Recall (%)		
All features	RF	91.8	89.1	87.9	22	18
	SVM	70.6	77.2	73.8	22	6571
	KNN	92.4	94.1	94.2	22	57
	LR	91.2	92.5	92.8	22	3833
SFS	RF	93.0	93.5	94.1	9	1200
	SVM	77.5	76.5	74.5	10	1800
	KNN	98.2	97.1	98.1	9	1711
	LR	90.2	91.2	89.2	12	2832
RFECV	RF	93.2	92.1	93.0	20	2270
	SVM	92.7	91.2	90.2	21	4419
	KNN †	-	-	-	-	-
	LR	90.2	91.3	89.2	20	5401
ReliefF	RF	93.2	92.3	94.1	9	7
	SVM	38.1	38.3	44.0	9	2840
	KNN	94.5	96.0	94.8	9	14
	LR	87.3	88.0	89.4	9	258

† RFECV does not support the k-NN classifier

Table 5.6: Comparison of the classification performance of RF, SVM, KNN and LR using different feature selection methods

(3) the identification time: the time to carry out identification. The dataset was first divided, where 80% of the instances were for training and 20% for testing. We then conducted experiments on three different hardware systems: high-end desktop, laptop and Raspberry Pi 3 (RPi), detailed specifications are shown in Table 5.5.

5.4 Results

5.4.1 Classification and Feature Selection

The overall results from evaluating the feature selection methods and classification are shown in Table 5.6 and Figure 5.3, this was divided into 4 categories: all features, SFS, RFECV and ReliefF and tested each on the 4 classifiers (as described in Sec. 5.3.6). We decided to use F_1-score as the main performance metric with precision and recall as supporting metrics as described in Section 5.3.6. The SFS method combined with the k-NN classifier and 9 features (as shown in Figure 5.4)

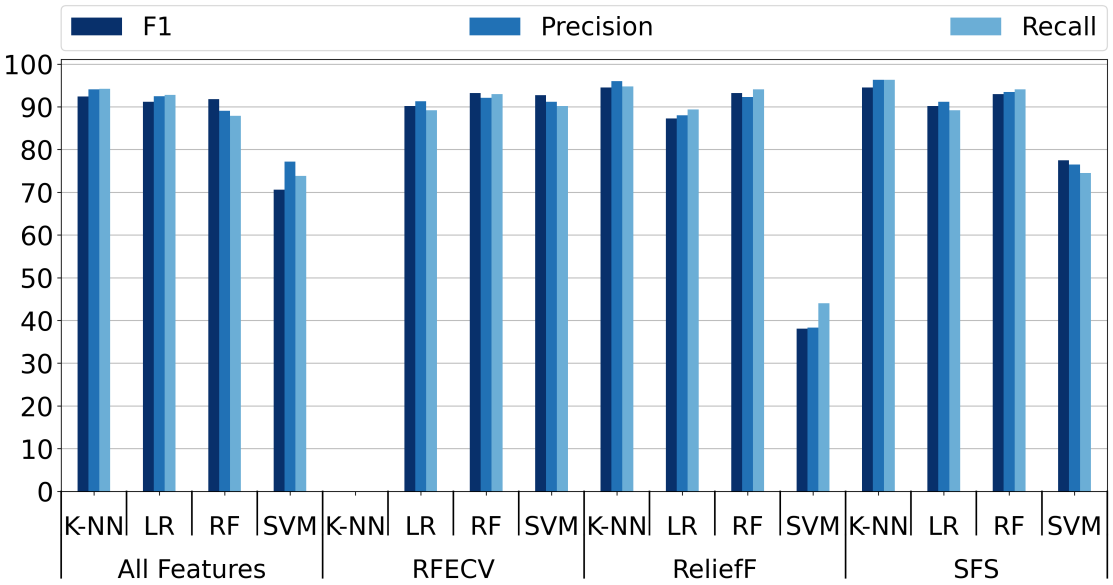


Figure 5.3: Results from comparing the four classifiers and the three feature selection methods

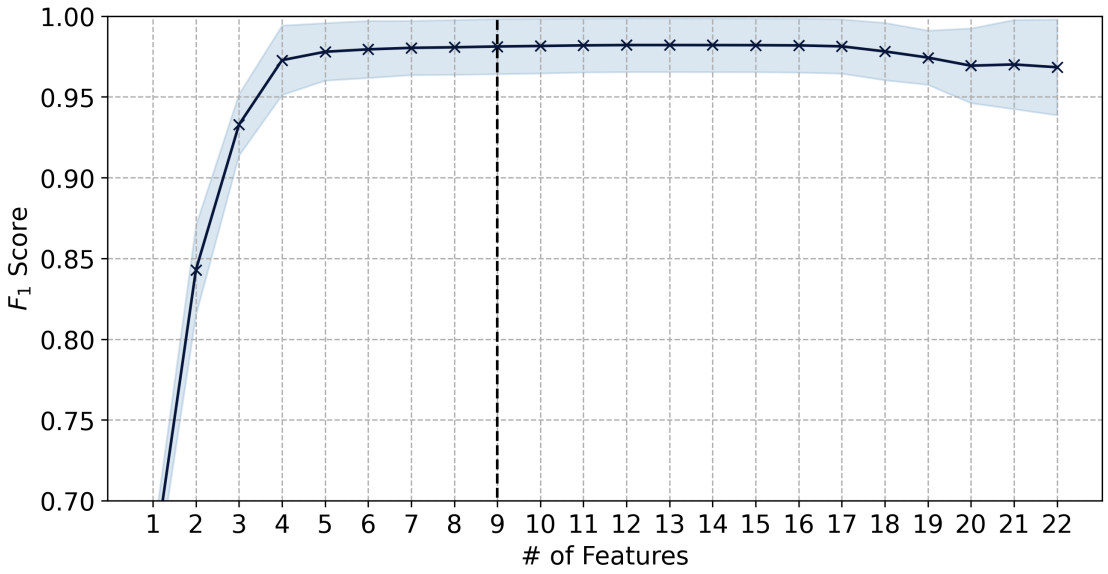


Figure 5.4: The performance of using SFS fitted to the k-NN classifier, and 9 features to get a F_1-*score* score of 98.2%

provided the highest prediction with an F_1-*score* of 98.2%, compared to using all the features where the F_1-*score* was 92.4%.

The 9 features that were selected using SFS included: Total bytes, Ethernet header (max,mean, 3rd quartile), IP payload (max) and TCP window size (max, min, mean). Although using these features only increased the identification rate by

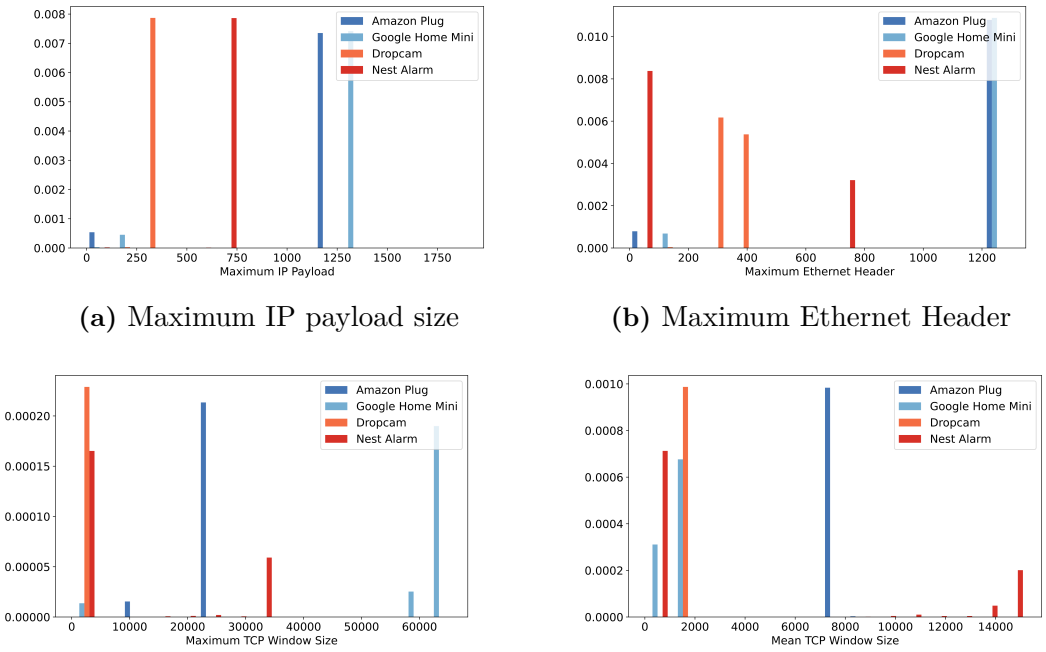
Category	Feature (*f*) No.	Description	Importance
Behavioural	1	Dur:First and last packet of flow	0.001
	2	TotBytes:total no. of downloads and upload bytes	0.004
	3	Afr:Average flow rate	0.000
	4	NTP interval	0.005
	5	DNS interval	0.004
Packet header	6	Minimum size of IP payload	0.028
	7	Median size of IP payload	0.050
	8	Maximum size of IP payload	0.057
	9	Mean size of IP payload	0.035
	10	1st quartile size of IP payload	0.037
	11	3rd quartile size of IP payload	0.034
	12	Variance size of IP payload	0.040
	13	Minimum size of Ethernet packets	0.028
	14	Median size of Ethernet packets	0.050
	15	Maximum size of Ethernet packets	0.057
	16	Mean size of Ethernet packets	0.035
	17	1st quartile size of	0.037
	18	3rd quartile size of Ethernet packets	0.035
	19	Variance size of Ethernet packets	0.041
	20	Minimum size of TCP window	0.042
	21	Maximum size of TCP window	0.125
	22	Mean size of TCP window	0.148

Table 5.7: 22 features (2 categories) used for device identification, the highlighted scores correspond to the most important features computed using the ReliefF feature selection algorithm

6%, the identification time decreased from 57 seconds (s) to 11s an improvement of 80.7% compared to using all the features. Overall, it took SFS 29 minutes to identify the best feature set, on the other hand, RFECV took considerably longer and the reduction of features were minimal.

The performance of identification was lowered by the following 5 features: the duration of flow (*f2*), average flow rate (*f3*), DNS interval (*f5*), variance IP payload (*f12*) and variance Ethernet packet (*f19*). The features *f2 f3* and *f5* decreased the performance of classification as they required a longer duration to generate these types of features. For example, for DNS interval it took on average 1201s to generate 1 instance, meant not enough instances were generated. These low features also corresponded with the results from the ReliefF algorithm where the importance was close to zero. The remaining 13 features did not improve the F_1-*score*, although they may be valuable in later stages when scaling the classifier with additional IoT devices.

The ReliefF algorithm was used to compute the importance scores for each feature



(a) Maximum IP payload size

(b) Maximum Ethernet Header

(c) Maximum TCP window size

(d) Mean TCP Window size

Figure 5.5: Distribution of the top 4 features using the ReliefF feature selection algorithm from 4 IoT devices

and are presented in Table 5.7. The most relevant features had a higher score than zero, whereas the least relevant features were zero or negative. Alternatively, if the threshold was set too low, there would an increase in the number of irrelevant features. As with any threshold, this was somewhat arbitrary, therefore since our importance scores ranged from zero to 0.148, we selected a threshold of 0.040 to avoid missing relevant features. In total, 9 features had the highest scores, and these were all from the packet header. Figure 5.5 demonstrates the distribution of the top 4 features using the ReliefF algorithm and shows a clear distribution among the features.

The TCP window size (min, max and mean) features were highlighted as the most important features as they were present in all the feature selection results. This was because TCP window sizes were dependent on the memory capacity of an IoT device and devices with fewer features tended to have smaller window sizes. For instance, the average size of the TCP window for the `Samsung Smartcam` was between 5850 - 28960, on the other hand the windows size for the `TP-Link switch` is only between 87-5816, this showed this feature separability over others features.

	Feature extraction (s)	Training (s)			Identification (s)			Overall (s)		
		Min	Max	Mean	Min	Max	Mean	Min	Max	Mean
High-end desktop	0.25	0.25	0.54	0.27	1.48	2.51	1.80	1.98	3.30	2.32
Laptop	0.42	0.57	1.60	0.50	1.81	2.90	1.60	2.80	4.92	2.52
RPi 3	0.66	0.70	1.90	0.83	2.23	3.20	2.71	3.59	5.76	4.29

Table 5.8: Computation performance of the k-NN classifiers on 3 types of hardware. These results were averaged over 10 iterations for feature extraction and 100 iterations for classification to generalise the results

5.4.2 Computation Performance

In addition to achieving high identification accuracy, it was fundamental that the approach was also fast during identification. From a performance perspective, we evaluated our approach based on the feature extraction/pre-processing time, training time and identification. To compute the time required for pre-processing we randomly selected 3 PCAP files that varied in size: 200KB, 10,000KB and 40,000KB. The average time it took to pre-process these PCAP files was 3.3s, 1250.69s and 4685.70s, respectively. The time required to pre-process correlated with the size of the PCAP file.

As the k-NN classifier performed best, for brevity we only discussed the runtime for this classifier. We looked at the computation performance which was the time required to complete each stage of our approach. Table 5.8 reports the time taken for device identification on 3 different hardware systems. The results showed that on the high-end desktop, on average using our technique we extracted, pre-processed the features, trained and classified the device in under 2.32s. Whereas the overall duration for all stages on the RPi 3 took on average 4.29s, nearly twice that of using the desktop or laptop. However, as RPi 3 technology improves, time for identification will reduce. On the high-end desktop, the best identification time was 1.48s and at worse 2.51s, while feature extraction and training required only 2.07s.

Furthermore, we found no relevant literature regarding the evaluation of a similar approach on devices such as the RPi, therefore we were not able to compare our results with previous work.

5.5 Discussion

This section discusses how our approach pushes the state-of-art and the limitations of our approach.

5.5.1 Comparison to the State-of-Art.

Current IoT identification approaches were not designed for forensic purposes. More precisely, they often required a substantial amount of training data and a large feature set that was resource-intensive. In our results, we demonstrate that using feature selection methods improves the identification accuracy by approximately 6%, and identification time by 80.7%. In addition, we used feature selection methods to reduce our feature set from 22 to 9 features and thus improved the identification time.

Table 5.9 shows a comparison of the state-of-art in IoT device identification and the various parameters. Many previous studies did not release or released a limited version of their dataset and code to the public, therefore comparing our approach to previous studies was challenging. The closest related work to ours was [223] and we both achieved the same accuracy of 98.2%. However, using a combination of the packet header and behavioural features we were able to reduce our feature set and still achieved a high accuracy. This was achieved with less dataset and only 9 features, when compared to their 33, we also suspect our identification runtime was reduced.

5.5.2 Limitations.

In Section 5.3.1 we stated that we excluded certain IoT devices, due to their infrequent network traffic as they require user interaction, devices such as smart scales and smart blood pressure monitors. Consequently, our approach was only possible if the device was on at the time of capturing the network traffic. Although we only investigated network traffic from Ethernet and Wi-Fi, IoT devices also use Zigbee, Z-Wave, BLE etc., but this was outside the scope of this study. However, we expect our approach can also be applied in a similar manner to other communication protocols.

	Miettinen et al.[22]	Helsinki et al.[23]	Bezawada et al.[222]	Sivanathan et al.[21]	Pinheiro et al.[240]	Marchal et al.[223]	Our solution
Feature type: Behaviour (B)/ Header(H)	H	H	B	H	B	B	B/H
No. of features	23	90	20	12	3	33	9
Identify device type (T) / Unique device (U)*	T	T	T	U	U	T	U
Feature selection	✗	GI[4]	✗	✗	✗	ReliefF	SFS
Average identification rate (%)	50-100	91.2	86-99	99	90	98.2	98.2
No. of IoT devices	27	27	14	28	15	23	22
Size of training data	-	-	✗	26 weeks	26 weeks	2.5hours	2hours
Computational performance	✗	✗	✗	✗	✓	✓	✓
Available online							
Dataset	✗ [1]	✗ [1]	✗	✓ [2]	✗	✗ [3]	✓ [5]
Code	✓	✓	✗	✗	✗	✗	✓

[1] Although the dataset is available this only included features that had been extracted from the PCAP files
[2] Out of the 26 weeks of IoT traffic, only 20 days of the PCAP files are publicly available[4]
[3] The authors of this research stated that the dataset will be released. However, at the time of this research, it had not been released.
[4] Gini importance
[5] The datasets and code used in our experiments are accessible through the following link: `https://tinyurl.com/DatasetCode`.
* Unique device-type refers to the manufacturer, model number and software version

Table 5.9: Comparison of the performance of the the state-of-the-art IoT device identification methods against our solution

5.6 Summary

In this chapter we propose the use of machine learning to identify the IoT device-type on the network, essentially creating a unique "fingerprint" of the device. We collected network traffic from 22 IoT devices, extracting 22 features from the packet header and behavioural features (Section 5.3.3) and demonstrated that we could achieve an identification F_1-*score* of 98.2% (Section 5.4.1). In addition, we evaluated different feature selection methods to reduce the number of features required for training and found that SFS was the best feature selection method: reducing our feature set from 22 to 9 and improving the identification runtime by around 80%.

Overall, we found the k-NN classifier performed the best with 9 features and had an overall identification F_1-*score* of 98.2% (Section 5.4)

The most important features were in the packet header category, these were the TCP header min, max and mean. The size of the TCP header varied significantly from IoT to IoT and was typically smaller on IoT devices, thus making it a good distinguishing feature. We found 3 features, the total number of bytes, average flow rate and DNS interval are the least important as they do not create enough instances.

Additionally, we compared the computational performance of our approach using 3 different types of hardware. On the RPi 3 it took on average $2.09s$ for identification, this is acceptable as our approach may have to be run on a mobile system (Section 5.4.2)

Overall, in this chapter, we have presented a machine learning approach capable of identifying IoT devices uniquely on the network using distinct features extracted from the network traffic combined with a feature selection method to reduce the identification runtime. Furthermore, we have demonstrated that our approach is generic enough to be compatible with a range of IoT devices from various categories (e.g., smart hubs, VAs) and manufacturers.

Nothing in life is to be feared, it is only to be understood. Now is the time to understand more, so that we may fear less.

— Marie Curie

6

Acquisition Method for Smart Healthcare Devices

Contents

Our findings in Chapter 3 demonstrate that acquisition should be the focus of current research. Motivated by this finding we now propose a method that uses BLE to acquire traces from smart healthcare devices. First, we briefly overview the background on BLE workflow, how data is accessed and the pairing process. We then outline our method, where we discuss how we use the BLE protocol to extract user data from BPM and their accompanying mobile apps. Furthermore,

we describe the method used to assess the security of the data on the BLE devices. Finally, we discuss the key findings from the results of our experiments which demonstrate that our method is feasible.

6.1 Introduction

The acquisition stage of the digital forensic process is important, as it encompasses the extraction and analysis of traces from consumer IoT devices. This is essential if a clear picture of events that have occurred or a historical profile of the user is to be established. However, acquisition from IoT devices is difficult, as manufacturers have not developed devices to a common standard, using a different OS, firmware and hardware, making acquiring data from these devices a complex process. Meanwhile prior work has developed acquisition frameworks limited to specific consumer IoT devices, such as the Amazon Alexa acquisition framework that only extracts data from that ecosystem [27]. More recently [28] acquired data by developing tools/plugins for individual IoT devices, which is a time-consuming process.

Therefore, in this chapter, we adopt a novel method in developing a generic acquisition method that is compatible with a range of IoT device manufacturers. We do so, by leveraging the use of a ubiquitous protocol BLE to acquire sensitive health-related traces from smart healthcare devices. Bluetooth Low Energy (BLE) operates as classic Bluetooth technology. This is used by many smart healthcare devices for short-range transmission and by the consumer to transfer data from the device to the mobile app.

BLE makes an attractive acquisition method, as it is integrated into many consumer IoT devices, including smartwatches, fitness trackers [29] and smart blood pressure monitors [30]. This means this method of acquisition is compatible with a wide range of BLE enabled consumer IoT devices. In addition, BLE lacks authentication and secure pairing to protect data, with many devices using the least secure pairing protocol "Just Works" or weak encryption, which allows credentials to be easily intercepted [31] [29, 32]. Using this method forensic investigators can

easily extract traces such as blood pressure readings and timestamps. These can be used as evidence or to develop a historical profile of the user's health.

Smart healthcare devices such as blood glucose, ECG and BPM etc., are becoming increasingly popular, as they allow patients outside of the clinical environment to share their recorded data with healthcare professionals. The amount of data generated by these devices and the accompanying mobile apps can provide a wealth of evidence, such as allowing an forensic investigator to infer a potential medical scenario. There are already criminal and civil cases where medical devices are prominent in forensic investigations. For example, medical data from the user's pacemaker was used as evidence in an arson case [241]. As the use of these devices increases and investigators become more familiar with them these cases are set to rise. Smart healthcare devices can provide invaluable data for forensic investigations but unfortunately, they are manufactured by various companies that customise their hardware and software. This results in a lack of a standard interface making acquiring data from them a challenging task. One common feature that they share is the use of BLE [242].

Another source of evidence is the mobile device, which is used to manage the smart healthcare device. The authors [100] examined the traces from the mobile app of an ECG monitoring device and found medical information, including heart rate and blood pressure readings. As each smart healthcare device manufacturer develops their own mobile app, how the data can be managed (e.g., the file format, name) is different. Since the authors only examined a single device, further research would be required to establish if manufacturers store data differently.

In this chapter, we demonstrate how BLE can be used to acquire traces from smart healthcare devices. Our contributions in this chapter are threefold:

- We present a novel acquisition method that examiners can use to extract traces from a multitude of BLE-enabled devices to acquire sensitive health-related traces from smart healthcare devices (Section 6.3.2).

- We examine the smart healthcare devices' accompanying mobile device (mobile app) for the availability and the type of traces and provide detailed documentation of our findings (Section 6.4.2).

- To assess the security of the data on the smart healthcare devices we conduct a security analysis to ascertain security measures (Section 6.4.3).

6.2 BLE Background

There are several key concepts and terms that the reader should become familiar with. In the following we provide a summary of BLE 4.0 (Note, that although BLE 4.2 is available, we found the majority of smart healthcare devices use BLE 4.0). The important procedures within the BLE workflow, include data access on a BLE device and pairing.

Overview BLE is a popular protocol for interconnecting IoT devices, offering a low energy interface for low-data-rate devices (e.g., temperature monitor, door locks).

Discrete data known as attributes are stored and processed within the Generic ATTribute Profile (GATT), which controls the connections, advertisements and how two devices interact. The data is constructed in a hierarchy within sections called *services*, which group related user data into characteristics. At the top level is the *profile* and is composed of a series of services. For example, a heart rate profile has both the heart rate service and device information service. The *services* are collections of characteristics that represent the behaviour of a device. A service can have one or more characteristics, and each one distinguishes itself from other services using a Universal Unique Identifier (UUID). *Characteristics* are the lowest level concept in GATT transactions and are containers for user data. Each characteristic has several other attributes, including the following:

- Handles: these are connection points where data can be read and written.

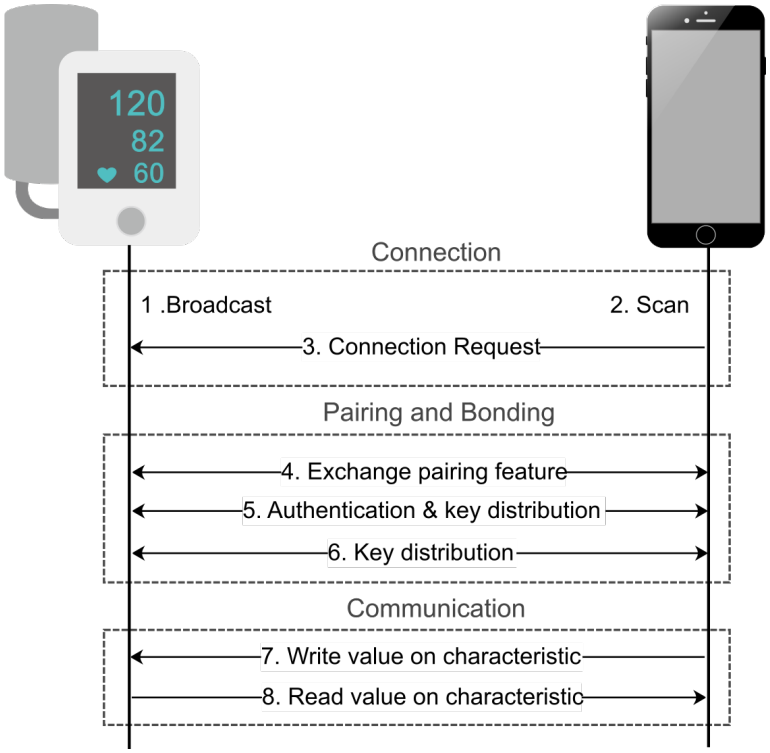


Figure 6.1: BLE workflow that shows the client on the right and the server on the left.

- UUID: each characteristic is a union of user data with metadata (descriptive information). UUID is a 128-bit (16 bytes) number that is globally unique. e.g., `0x2800`.

- Properties: these are access permissions to determine what data can be read and/or written. Each characteristic can be accessed based on the following permissions: read, write, notify and indication.

- Value: this is the value of the attribute and is described by the attribute type (UUID) [32, 243].

Connection Figure 6.1 provides an overview of the process within a BLE workflow. When one BLE device wants to access data on another BLE device, a connection is initiated by the client (e.g., mobile device) to the server (e.g., lock, thermostat and blood pressure monitor).

Pairing BLE security involves a pairing process in which two BLE devices exchange information to establish a secure link. The pairing process starts by exchanging their pairing features (e.g., input and output capabilities such as keyboard or display, see step 4 on Figure 6.1), which helps the devices to decide which pairing method to use. There are 3 main pairing methods for BLE 4.0: Just Works, passkey entry and Out of Band (OOB)

In Just Works, the Temporary Key (TK) is 0 by default, and there is no method to verify the devices taking part in the connection. In the passkey method the TK is 6 digital number and is passed between the devices. The way the number is transferred can vary; a typical example is to have the device (server) generates a random 6-digit number and this is sent to the mobile device (client) and displayed on the LCD. The client then enters this into the other device using a keypad. OOB requires that the client and server have an interface (i.e., NFC) to exchange the TK. This method is more secure as a larger TK of up to 128-bits can be used. However, when compared to other the two methods this one is rarely used [32, 243, 244].

It is important to understand the BLE pairing process and the security. Although we have not studied this in-depth as this is outside the scope of our studies. In particular our work does not consider how an investigator could bypass the security on a BLE device.

Communication In steps 7 and 8 of Figure 6.1 the characteristics can be read/written depending on the type of permissions: Read- the characteristics can be read by the client; Write - the attribute can be written by the client; Notify - the client can request a notification for a particular characteristic from the server. Once the client enables the notification, the server will send the value to the client whenever it becomes available; Indicate - is similar to notification but is more reliable, as it receives a confirmation message back from the server [32, 243, 244].

6.3 Methodology

Here we describe how we acquired useful forensics traces from smart healthcare devices through BLE protocols and these devices' mobile apps, and our experimentation with the security of the data on BLE devices. The apparatus used in this research is presented in Table 6.1 and the methodology included the following steps.

Smart Healthcare Device Selection: 6 smart healthcare devices were selected including 4 Blood Pressure Monitors (BPMs) and 2 smart scales. We intentionally chose different manufacturers so that we could compare the way different device manufacturers saved data (Section 6.3.1).

Communication Protocol - BLE We used the BLE protocol to extract data from 4 BPMs (Section6.3.2).

Mobile Device: We performed a logical and physical acquisition from the mobile apps of 6 smart healthcare devices (using 2 Android mobile devices) (Section 6.3.3).

Security Assessment on BLE devices: Tests were conducted on 8 BLE-enabled devices and consisted of: 4 BPMs, 2 scales and 2 smartwatches (Section 6.3.4).

Results: The results are then presented (Section 6.4).

6.3.1 Smart Healthcare Device Selection

As shown in Table 6.1, overall we selected 4 BPMs and 2 smart scales for BLE data extraction experiment. We first searched for the most popular categories of smart healthcare devices in retail outlets e.g., Amazon, eBay. The most popular categories were BPMs and smart scales. We then selected devices based on their reviews and ratings. We found two devices that are Food And Drug Administration (FDA)[1] approved: Koogeek KSBP2 and iHealth Track KN-550BT. However, this does not mean it has been approved for use in the UK.

[1]FDA regulates the sale of medical device products in the U.S. and monitors the safety of all regulated medical products.

Category	Device
BPM	Koogeek KSBP2*†
-	iHealth Track KN-550BT*†
-	Philips DL8760*†
-	Qardio Arm*†
Scale	Kamtron Smart Body Fat Scales*†
-	Xiaomi Mi Smart Scale 2*†
Smartwatch	Garmin Vivoactive 3†
-	Xiaomi Mi Mi Fit band 3†

* Devices used for the BLE data extraction experiment
† Devices used for analysing data confidentiality experiment

Table 6.1: The 6 devices utilised in the experiment to extract data using BLE and the additional 8 devices (additional 2 devices) utilised for analysis of data confidentiality experiments

We limited the scope of this study to IoT consumer BPMs with BLE as they were the most common devices featured in online retail outlets. Ideally, we would test several different devices for the BLE experiments. However, our aim was to demonstrate that it was feasible to use BLE to acquire and analyse traces.

6.3.2 Communication protocol:BLE

In this section, we show how we used BLE to extract data from the four BPMs and then manually examined each device using the following stepwise process:

① **Obtained the MAC address of the device:** We obtained the MAC address of each BPM monitor using the tool `hcitool` and scanned for local BLE devices using the command *hsciscan lescan.*

② **Examined available services and characteristics:** The tool `gatttool` was used to obtain the GATT profile, which provided a list of services and characteristics of each device. The services and characteristics of the Koogeek, iHealth and Phillips BPMs are shown in Tables 6.2, 6.3 and 6.4 respectively. The characteristics that can be customised by the manufacturer were labelled as “Unknown service”. By reading the “measurement” characteristic we found that the blood pressure measurements were stored under this service.

③ **Intercepted BLE packets:** To understand the communication and the values sent between the BPM and mobile device, we captured the network traffic using `Wireshark` and a BLE sniffing device (Bluefruit LE[2])

④ **Examined BLE packets:** We examined the BLE packets abd we found continuous write requests to handle `0x31` with the opcode field `0x20` (which referred to indications). We used the `gatttool` and wrote the following command in the format of char-write-req <handle> <value>. For instance, on the Koogeek BPM, we wrote `char-write-req 030 020` which returned the values `0e 73 00 46 00 00 00 01 00 f3 43 01 1b 0d 49 00 00`. To understand what the values represented, we took actual blood pressure measurements. We then uploaded them to the mobile app and repeated this procedure while capturing the BLE packets.

⑤ **Understood the values:** We compared the value field of the BLE packets and found parts that remained constant and others that varied. For example, on the Koogeek device, the bytes that repeatedly changed are highlighted:

`0e 73 00 46 00 00 00 01 00 04 03 0c 17 2e 49 00 00`

These values were converted from hexadecimal bytes to base 10, which corresponded to the blood pressure readings 115/70/73 and timestamp of 15th April, 12:23:46.

6.3.3 Mobile Device

For these experiments, we used 6 smart healthcare mobile apps, which included the 4 BPMs and 2 scales: Koogeek, iHealth, Xiaomi (Mi Fit), Phillips (PHS Health), Qardio and Kamtron (Feelfit). These were installed on the Android mobile device. New user accounts were created for each smart health care device and then data was generated through daily usage. The results may vary on different Android devices and OS versions, therefore we selected two mobile devices (Nexus 5x and Doogee X5)

[2] `https://learn.adafruit.com/introducing-the-adafruit-bluefruit-le-sniffer/using-with-sniffer-v1`

Service	Characteristic	Handle	Type of Attribute (UUID)	Permissions	Attribute Value
Generic Access	-	0x0001	0x2800	Primary Service	-
	Device name	0x0003	0x2a00	Read	Koogeek_Health
	Appearance	0x0005	0x2a01	Read	[0] Unknown
	Peripheral Privacy Flag	0x0007	0x2a02	Read, Write	Privacy is disabled on this device
	Re-connection Address	0x0009	0x2a03	Write	-
Generic Attribute	System ID	0x000e	0x2a23	Read	(0x) AA-E3-28-42-4B-48-8B-33-50
	Model Number String	0x0010	0x2a24	Read	0717
	Serial Number String	0x0012	0x2a25	Read	23033087
	Firmware Revision String	0x0014	0x2800	Read	17041401
	Hardware Revision String	0x0016	0x2a27	Read	P1332110A
	Software Revision String	0x0018	0x2a28	Read	546T0140
	Manufacturer Name String	0x001a	0x2a29	Read	Bough Tech
	IEEE 11073-20601 Regulatory	0x001c	0x2a2a	Read (0x)FE-00-65-78-70-65-72-69-6F-65-6E-74-61-6C	
	PnP ID	0x001e	0x2800	Read	Bluetooth SIG Company-Reserved ID <0xBB00>, Product Id:4611, Product Version:512
Unknown Service	-	0x0021	0xfff1	Read,Write	-
	System Settings	0x0022	0x2901	-	-
	-	0x0023	0x2803	-	-
	-	0x0027	0xfff3	Read,Write Notify, Write no response	17041401
	-	0x0028	0x2902	-	-
	Function Con	0x0029	0x2901	Read	-
	-	0x002a	0x2803	-	-
	-	0x002c	0x2800	Read	-
	Cuff Pressure	0x002d	0x2901	-	-
	-	0x002e	0x2901	-	-
	-	0x002d	0xfff5	Indicate	-
	-	0x0030	0x2902	-	-
	Measurement	0x0031	0x2901	-	-
	-	0x0001	0x2800	Read	17041401
	OADS1609	0x0033	0xfff6	Read	-
	OAD Firmware	0x0001	0x2901	Read	-
	-	0x0036	0xfff7	Read,Write	-
	Date and Time	0x0037	0x2901	Read	-
Battery	Battery Level	0x003b	0x2a19	Read, Notify	-
	-	0x003c	0x2902	-	-

Table 6.2: The GATT profile of Koogeek BPM and its 4 primary services

with different OS versions (Oreo 8.0 and Nougat 7.0). These versions of Android OS were selected as they were the most popular at the time of this study [245]. After the accounts and data were created, we used two approaches for acquisition. First, the

Service	Characteristic	Handle	Type of Attribute (UUID)	Permissions	Attribute Value
Unknown Service	-	0x0021	0x2800	Read,Write	-
	System Settings	0x0022	0x2901	-	-
	-	0x0010	0x2803	-	-
	Measurement	0x0027	0xfff3	Read,Write Notify, Write no response	17041401
	-	0x0028	0x2902	-	-

Table 6.3: The GATT profile of iHealth BPM and its unknown services

Service	Characteristic	Handle	Type of Attribute (UUID)	Permissions	Attribute Value
Unknown Service	-	0x0009	0x2800	-	-
	-	0x000a	0x2803	-	-
	-	0x000b	0x8a91	Indicate	-
	Function Con	0x0029	0x2901	-	-
	-	0x000c	0x2902	-	-
	-	0x000d	0x2803	Read	-
	System Settings	0x000e	0x8a90	Read	-
	-	0x000f	0x2803	-	-
	-	0x0010	0x8a92	Notify	-
	-	0x0011	0x2800	-	4233444
	-	0x0012	0xfff7	-	-
	-	0x0013	0x8a81	Write	-
	-	0x0014	0x2803	-	-
	Measurement	0x0015	0x8a82	Indicate	-
	-	0x0016	0x2902	-	-
	Date and Time	0x0037	0x2901	-	-

Table 6.4: The GATT profile of Phillips BPM and its unknown services

physical image was extracted using `dd`[3] and then for the logical image we used the Android Debug Bridge (ADB) with the following ADB command, `adb pull` [246].

Additionally, we carried out further experiments to determine whether traces would be deleted if the user logged out. We logged out of each mobile app and examined the traces. We then logged in again using a different account and re-examined the mobile device for traces.

6.3.4 Analysis of Data Confidentiality

In this section, we show how we examined BLE-enabled devices to establish the level of confidentiality of the stored data and examine the read and write properties of the device. Confidentiality is a basic security service for data protection and all

[3]Note. dd is a command-line tool used to create disk images, copy files, etc. of hard drives

sensitive data should be encrypted to ensure data confidentiality. If no encryption is used it would be possible for a forensic investigator to easily access this data.

We conducted tests on 8 BLE-enabled devices: Koogeek KSBP2, iHealth KN-550BT, Phillips DL8760, Qardio Arm, Kamtron Smart Body Fat scales, Xiaomi Body Composition scale, Garmin Vivoactive 3 and Mi Fit band 3. These are shown in Table 6.1, and they included the six devices from the BLE experiments and two additional devices: Garmin Vivoactive 3 and Mi Fit band 3 smartwatches. We chose to add these devices to include the additional category of smartwatches to broadened the scope of our research.

Using the nRF Connect[4] Android mobile app, we manually connected to each device, to read/write data. The characteristics of each device were then assigned to one of three levels depending on the security implemented: Low - no encryption or pairing and the attribute is accessible in cleartext, Medium (Unauthenticated encryption) - the connection is encrypted to access this attribute, although it is not a requirement for the encryption keys to be authenticated using the JustWorks pairing process, and High (Authenticated encryption) - the connection is encrypted, and pairing is required using passkey entry or OOB.

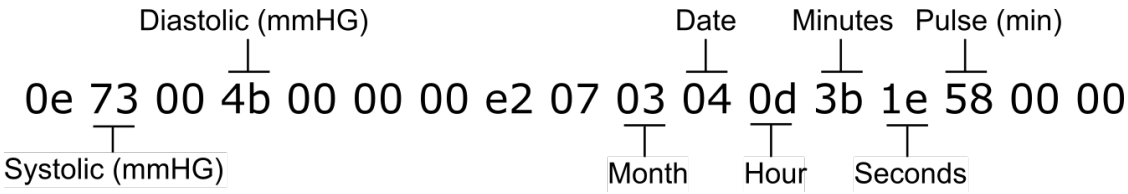
6.4 Results

In this section we discuss the key findings, and these are broken down into 3 different areas of interest: the BLE communication protocol, mobile forensic analysis of the smart healthcare apps and results from assessing the confidentiality of the data stored on the BLE-enabled devices.

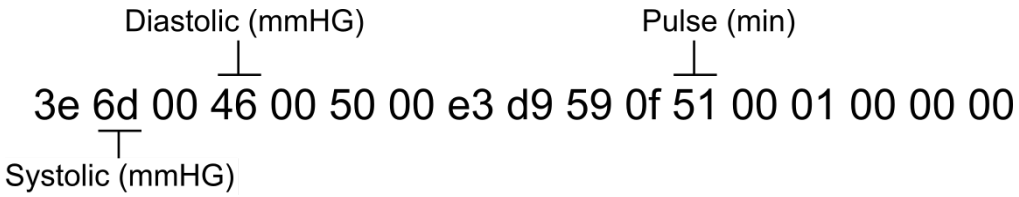
6.4.1 Communication protocol: BLE

We examined the 4 BPMs and found only the Quardio Arm BPM did not have an LCD screen or buttons to start/stop the device, and therefore required the mobile app to take measurements. We found all the BPM used the official BLE

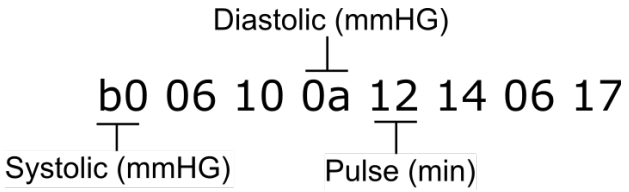
[4] https://www.nordicsemi.com/Products/Development-tools/nrf-connect-for-mobile



(a) Structure of the blood pressure measurements taken from the Koogeek BPM



(b) Structure of the blood pressure measurements taken from the Phillips BPM



(c) Structure of the blood pressure measurements taken from the iHealth BPM

Figure 6.2: Structure of the blood pressure readings taken from 3 BPMs

specifications for 3 of the 4 services: generic access, generic attribute and device information, with one of the services as "unknown service".

Each device had a different handle and when writing a request command would return the blood pressure measurements. For each device, the handle differed: Koogeek `0x30`, iHealth `0x13` and the Phillips `0x16`. The values for the measurements of the Koogeek device are shown in Figure 6.2a, and we found the measurement values consisted of the 2^{nd}, 4^{th} and 15^{th} bytes which corresponded to the systolic, diastolic and pulse readings. Additionally, within the values, the timestamp was at the 10^{th}-14^{th} bytes, in the following format `MM:DD:HH:MM:SS`. Figure 6.2b shows the values of the blood measurements from the Phillip BPM, consisting of the 2^{nd}, 4^{th} and 12^{th} bytes.

The Quardio Arm BPM had enabled the BLE feature called LE privacy, which when enabled would change the MAC address at regular intervals [243]. The device also relied on the mobile app to take measurements and only stored data until

	HSD	Movei	lineA	bpOfflineDataID	bpOfflineDia	ineHu	fflineI	flineLastChange	Offline	fflineL	Offline	OfflineMeasuredD	ieMea	fflineN	fflineI	oteCh	bpOfflinePulse	bpOfflineSys
				Filter	Filter			Filter				Filter					Filter	Filter
1	0	1	0	1427F5AFF1D...	148.0	0	0	1558552096	0.0	1	0.0	1558551960	NULL	0		0	109	169.0
2	0	1	0	1893D73979B...	102.0	0	0	1519556164	0.0	2	0.0	1519513560	NULL	0		0	70	113.0
3	0	0	0	E0691FD268F...	98.0	0	0	1583351183	0.0	3	0.0	1583350980	E06...	0		0	87	121.0
4	0	1	0	1893D73979B...	94.0	0	0	1518800193	0.0	3	0.0	1518799920	NULL	0		0	76	123.0
5	0	1	0	1893D73979B...	94.0	0	0	1519556164	0.0	3	0.0	1519515180	NULL	0		0	74	114.0

(a) iHealth BPM Blood Pressure Measurement in the SQLite database file on the mobile device

	_id	user_id	sync_status	revision	puls	dia	sys	irregular_heart_beat	neasure_dat	note	device_id	timezone	latitude	longitude
	1	2	2	2	70	81	23	107	8551960	23	123	0	0	0
53		NULL	0	0	NULL	NULL	NULL	NULL	NULL	NULL	NULL		NULL	NULL
54		NULL	0	0	NULL	NULL	NULL	NULL	NULL	NULL	NULL		NULL	NULL
55		NULL	0	0	NULL	NULL	NULL	NULL	NULL	NULL	NULL		NULL	NULL
56		NULL	0	0	NULL	NULL	NULL	NULL	NULL	NULL	NULL		NULL	NULL

(b) Qardio BPM in the SQLite database file on the mobile device

	rid	account	serial_id	uuid	sys	diy	pluse	anArterialPressu	locktime	create_time	modify_time	platform	unit
		Filter	Filter	Filter	Fi...	...	...	Filter	Filter	Filter	Filter	Filter	Filter
1	1		071723033087	3fc6423c-fde8...	105.0	81.0	91	0.0	1	1583393490834	1583393603378	0	1
2	2		071723033087	52e3e2fa-dbf...	116.0	78.0	85	0.0	1	1583393491029	1583393603364	0	1
3	3	99283	071723033087	5ce8d0dc-169...	113.0	78.0	93	0.0	1583393603370	1583393592000	1583393692224	1	1
4	4		071723033087	1aa28bdb-a1...	126.0	90.0	98	0.0	1583393603370	1583395167000	1583395209589	0	1
5	5	99283	071723033087	760483ff-c64...	124.0	83.0	98	0.0	1583395280445	1583395276000	1583395283513	1	1

(c) Koogeek BPM Blood Pressure Measurement in the SQLite database file on the mobile device

	SERVER_ID	LE_N	NAL_	USER_ID	WEIGHT	BODYFAT	SUBFAT	VISFAT	WATER	BMR	ODYA	MUSCLE	BMI	BONE	SCORE
	Filter			Filter	Filter	Filter	Fil...	Filter	Filter	Fi...		Filter	Fil...	Filter	Filter
1	12696547679...	QN...	0196	12696496287...	69.0999...	30.7999...	27.399...	9	47.5	1402.0	31	40.299999237...	27.0	2.8699...	72.6999...
2	12696551190...	QN...	0196	12696496287...	68.5500...	0.0	0.0	0	0.0	0.0	0	0.0	26.799...	0.0	NULL
3	12696554328...	QN...	0196	12696496287...	68.6500...	0.0	0.0	0	0.0	0.0	0	0.0	26.799...	0.0	NULL
4	12696556466...	QN...	0196	12696496287...	69.4499...	31.1000...	27.600...	9	47.2999...	1403.0	32	40.200000762...	27.100...	2.8699...	72.3000...
5	12696559042...	QN...	0196	12696496287...	69.4499...	31.1000...	27.600...	9	47.2999...	1403.0	32	40.200000762...	27.100...	2.8699...	72.3000...

(d) Kamtron scales in the SQLite database file on the mobile device

Figure 6.3: SQLite files of forensic significance extracted from the Android mobile app of iHealth Fig. 6.3a, Qardio Fig. 6.3b, Koogeek Fig. 6.3c and Kamtron scales Fig. 6.3d

it was connected to the mobile app. We discovered many IoT consumer medical devices and smartwatches have an LCD. This meant without this feature it was difficult to take readings and determine whether the device was on or off.

We obtained the blood pressure measurements, however, other useful information can be read from the device, such as the firmware version, model number and device name.

6.4.2 Mobile Device

We examined the traces from the 6 mobile apps, and a detailed summary of our findings is shown in Table 6.5. We found traces stored in various subfolders within the `/data/data` folder. The traces stored in database files (SQLite) and configuration

files (Extensible Markup Language(XML)) were analysed using DB Browser[5] and Notepad++[6]. Personal data including information such as email, age, gender, date of birth, height, weight etc. were found stored within XML and SQLite database files.

Overall, we recovered traces such as blood pressure measurements and body composition data from 5 out of the 6 mobile apps. In relation to the BPMs recovered from all (4) of the BPM mobile apps, these included traces such as blood pressure measurements, timestamp, diastolic, systolic and pulse readings.

Figure 6.3a shows the example traces recovered from the iHealth app. This contained a SQLite database file with a table of 59 entries on information about the user such as gender, weight and the timestamp of when the readings were taken. These traces were similarly found on both the Qardio app (fragments of the 33 entries shown in Figure 6.3b) and on the Koogeek app (fragments of the 13 entries shown Figure 6.3c).

On examining the 2 accompanying mobile apps for the scales, only the Kamtron scale stored data of the user's body composition (as shown in Figure 6.3d). On the Xiaomi scales we did not find any traces concerning body metrics. Lastly, we found when we logged out of the mobile apps the traces remained, even after logging in as a new user.

Traces such as blood pressure measurements are of forensic significance as it can aid investigators to create a historical profile of the user's state of health. For example, this type of information was proven useful, in a case in the U.S. where data from the pacemaker was used as evidence to charge a suspect for various crimes, including arson [247].

Personal information such as the weight, body fat, Body Mass Index (BMI) can provide an investigator with a detailed physical profile of the user. Smart healthcare devices including wearable devices, such as Fitbit will continue to increase in popularity, and the trend of them being used as evidential data will surely increase as well. Similarly, fitness watch data was used as vital evidence to support an investigation involving a serious crime. The exact time of death was

[5] `https://sqlitebrowser.org/`
[6] `https://notepad-plus-plus.org/`

Category	Device / Mobile App	Evidence	Recovered	Description
BPM	iHealth	Blood pressure	✓	Timestamp, Birthday, Height, Last time, User ID,
		Timestamp	✓	Weight, Height, Gender (User information), Blood
		Personal data	✓	pressure measurements(dia,pulse,sys)
	Koogeek	Blood pressure	✓	Wifi password, SSID, Email, Device ID, Country,
		Timestamp	✓	UUID, Account ID, Timestamp, Blood
		Personal data	✓	pressure measurements (dia, sys, pulse)
	Phillips (PHS Health)	Blood pressure	✓	Identifier, Authentication token, User information
		Timestamp	✓	Blood pressure measurements
		Personal data	✓	
	Qardio	Blood pressure	✓	Blood glucose measurements, Measurements,
		Timestamp	✓	Weight measurements, Measurement history,
		Personal data	✓	Visitor
Scales	Xiaomi (Mi Fit)	Weight	✗	Device ID, Access Token, UUID, Last location details
		BMI	✗	(longitude, latitude, City info, External IP address)
		Body water	✗	
		Body fat	✗	
		Timestamp	✗	
		Protein	✗	
		Bone mass	✗	
	Kamtron (Feelfit)	Weight	✓	MAC address, Server ID, Scale name, User ID,
		BMI	✓	Weight, Bodyfat, Subfat, Visfat, Water, BMR,
		Body water	✓	Bodyage, Muscle, BMI, Bone, Score, Gender,
		Body fat	✓	Birthday, Height, Resistance, Protein, FFM, Skeleton
		Timestamp	✓	Timestamp, Timezone, Email, Account name
		Protein	✓	
		Bone mass	✓	

Table 6.5: Summary of results from forensic analysis of the mobile devices

determined by examining the recorded heart rate measurements, demonstrating the usefulness of the data we extracted [13].

6.4.3 Data Confidentiality

We assessed the confidentiality of the data stored on 8 BLE-enabled devices, by establishing whether the data stored can be read or written, with the results shown in Figure 6.4. The devices that used the passkey entry method (highest security) were the Garmin Vivoactive 3 and Mi Fit band 3. The remaining devices used the

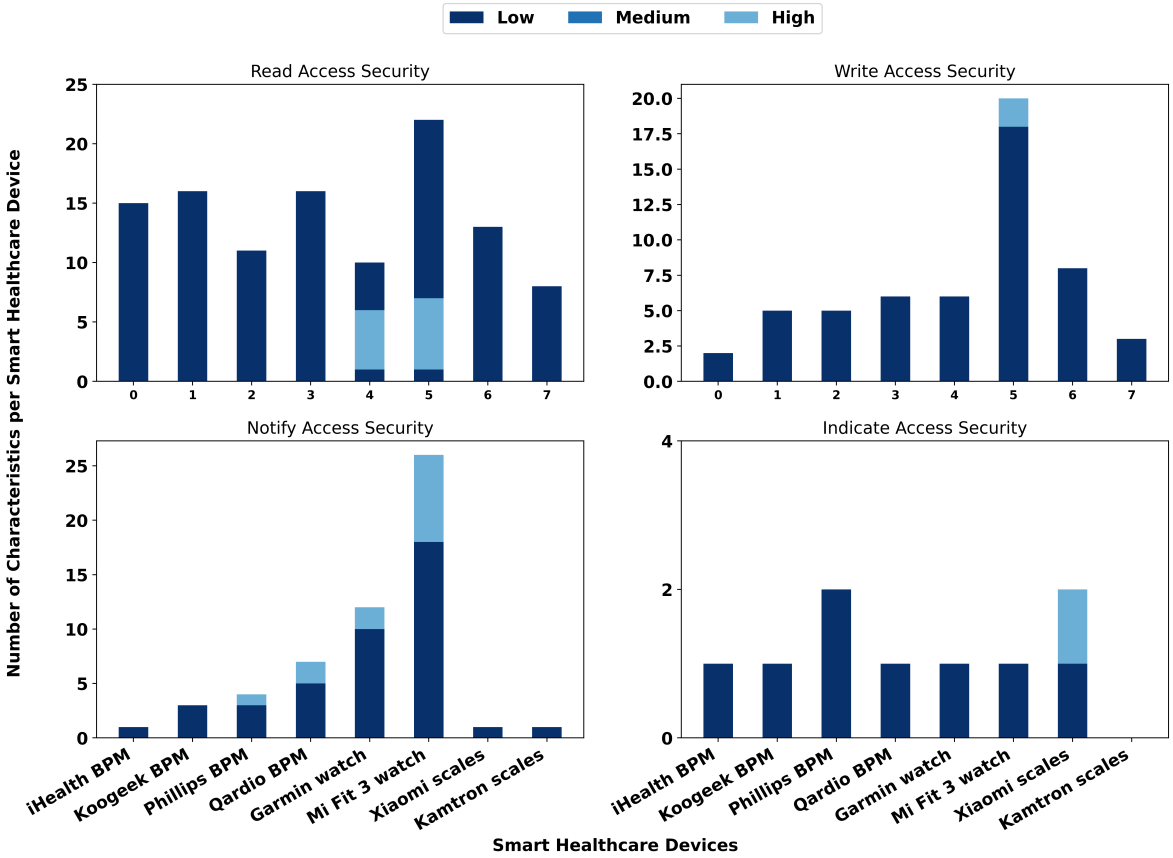


Figure 6.4: Security assessment of the characteristics on various smart healthcare devices

least secure pairing method Just Works, 5 out of the 8 devices allowed all their characteristics to be read and 7 of the devices allowed all characteristics to be written. These results were consistent with [248] findings, where they examined several IoT devices including smart watches, smart locks, home automation devices etc. and found only 2 out of 10 used security features. These result shows that the use of BLE protocol to acquire data was a feasible solution, as many devices allowed characteristics to be read without any security implementation.

6.5 Discussion

Usefulness of the Data Instead of focusing on the extraction of data from the smart healthcare devices. The majority of recent work focused on acquiring traces from the accompanying mobile apps, such as [108, 109]. This may be because there are well established methods to extract traces from mobile devices. However, we

found not all mobile apps left traces on the mobile device (as shown in Section 6.4.2).

This suggested that the traces such as blood pressure readings, timestamps and other user information found stored on the smart healthcare could be vital source of evidence and recovery is important. Previously, this type of information has proven to be helpful, such as the case in the US where pacemaker data was used as evidence to charge a suspect for various of crimes, including arson [247]. Similarly, fitness watch data was used as vital evidence to support an investigation involving a serious crime where the exact time of death was determined by examining the recorded heart rate measurements [13]. The use of these devices are becoming more widespread with real-time continuous monitoring of users outside of clinical environment, it is important forensic procedures are able to capture this evidence. Recent criminal cases in the US indicate that it is only a matter of time before medical data from these devices becomes pertinent evidence in civil and criminal cases.

Limitations We found that each smart healthcare device manufacturer had a slightly different GATT profile structure, such as the number of characteristics, making this solution more difficult to generalise. However, we found many of the devices used 4 services, where 3 followed the BLE specification and only 1 service was customised by the manufacturer, which limited the variations between devices. With the increasing numbers of IoT devices with BLE and the introduction of Bluetooth 5 [7], this would be a good start in developing an acquisition method.

Lack of security on BLE devices The findings from our analysis suggested that many smart healthcare devices did not use any security or pairing procedure. These results concur with [249] research on BLE security, where they found 12 out of 15 BLE smart locks could be easily accessed. Similarly, researchers found many BLE smart toys did not use passwords, pins or other authentication to gain access to the toy. The lack of security means that BLE as an acquisition method could be extended to other consumer IoT devices such as smart toys and smart watches [250].

[7] `https://www.nordicsemi.com/eng/Products/Bluetooth-5`

Mobile app forensics Research on mobile application forensics on smart healthcare devices was conducted by [109] on a single ECG device. Likewise, in our work, we found user data in relation to body composition and blood pressure readings from mobile app data traces. Additionally, we found the amount of user data varied between mobile apps, in particular on the Xiaomi scales we found limited user data. Therefore, we cannot rely solely on the traces from the mobile app, and the traces extracted using BLE could be vital. The research by Chung et al. [27] on mobile app forensics demonstrated that when a user logged out the database was deleted. However, we found that all the mobile apps retained data after the user logged out.

6.6 Summary

In this chapter, we presented a methodology that used BLE to extract records from smart healthcare devices (Section 6.3.2). This allowed the recovery of blood pressure readings and timestamps from 3 out of the 4 BPMs. Moreover, the traces recovered provided data useful for forensic investigations which established a historical profile of the user's health. We found we could not extract blood pressure measurements from the BPMs as the device had no LCD connecting directly to the mobile device to take readings, however we found most devices had a display for ease of use (Section 6.4.1).

The smart healthcare devices accompanying mobile apps left user data on the mobile device, which remained even when a user logged out. However, the quantity of traces varied depending on the mobile app, for example we found the Mi Fit app left only limited user data (Section 6.4.2).

Finally in this chapter we assessed the confidentiality of the data stored on the BLE enabled devices, we found that many smart healthcare devices did not require pairing or use security mechanisms. This demonstrated that data can easily be obtained using BLE, in contrast to extracting data from the cloud where credentials are required (Section 6.4.3).

I think it's very important to get more women into computing. My slogan is: Computing is too important to be left to men.

— Karen Spärck Jones

7

Network Traffic Analysis of Consumer IoT Devices

Contents

The previous two chapters investigated the first stages of the digital forensic process: identification and acquisition. However, it is also important to investigate how to analyse the traces acquired. Additionally, one of the key findings from the survey in Chapter 3 was that research should focus on the development of

IoT forensic tools. Following this in Chapter 4 we reviewed existing IoT forensic tools to establish the type of tools required, where we found the need for a tool to analyse network traffic from IoT devices. In this chapter, we fulfil this requirement by introducing a tool to analyse IoT network traffic.

Many different traces can be analysed on an IoT device, such as the memory and internal storage [28, 33]. Instead, this chapter focused on the significant traces obtained from the network layer. We examined the network traffic between the devices and systems they communicated with (e.g., cloud, mobile app), looking for unencrypted information containing potentially useful traces. Furthermore, we provided insights into ways investigators can obtain information from an IoT device e.g., enable remote services such as SSH or telnet.

Although recent work was limited in the number of IoT devices they studied. They demonstrated the potential benefits of analysing IoT network traffic for forensic traces, including obtaining a list of API calls to retrieve user data from the cloud [27] and sensitive data such as login credentials [28, 34]. As a result, this chapter aims to understand what "metadata" can be obtained from the network traffic for forensic purposes. To this end, we conducted a large-scale and in-depth study of the network traffic from a wide range of popular consumer IoT devices. This included examining the IP payload for cleartext content for credentials, and the destination IP address to reveal the final location of the data. Additionally, during this study, we found that manually analysing IoT network traffic required a considerable amount of time. Consequently, this thesis introduced the first tool capable of automating the analysis of IoT network traffic called `IoT Network Analyzer`. We focused on the following four sub-research questions derived from **RQ5** introduced in Chapter 1:

RQ5.1 Do IoT devices expose ports that allow an investigator to connect / access a device?

Motivation: Previous research has shown that many IoT devices expose their remote ports (i.e., SSH port 22) which would provide a forensic investigator access to acquire the filesystem and obtain evidence in an investigation [28,

85]. We set out to study if remote access is widely available or if this is limited to a subset of IoT devices.

Results: From the 17 devices, we found only 1 device with remote access available (port 22). Additionally, we found 2 smart cameras that exposed 21 TCP/UDP ports open and 2 smart cameras that had all their ports closed.

RQ5.2 Do IoT devices utilize encryption when sending/receiving information from the cloud and corresponding App?

Motivation: It is often stated in previous work [28] that the majority of IoT devices encrypt their communication channels, thus making it difficult to find any useful forensic traces. We set out to investigate whether this assertion was correct and if not, which devices were prone to send data in cleartext.

Results: Most devices encrypted their network traffic. However, smart cameras and smart healthcare devices sent data in cleartext. Additionally, smart healthcare devices exposed personal data potentially useful for an investigator to identify features of a person of interest.

RQ5.3 Do IoT device applications (Apps) utilize encryption when sending / receiving information?

Motivation: Being able to observe the content transmitted between the mobile apps and the cloud can be of significant forensic interest. Even when encryption is used, it can be compromised by intercepting the traffic using a proxy server to decrypt the HTTPS traffic. We set out to investigate what content was sent between the IoT devices' mobile app and the cloud.

Results: We found 7 of the 13 mobile apps allowed proxy connections, the remaining 6 mobile apps used certificate pinning. An expected result from the Kasa mobile app for the TP-Link camera was that when opened: it took a snapshot that included a timestamp and URL link to the JPEG. This function was not something the consumer could control/disable.

RQ5.4 To which countries do the IoT devices and Apps communicate/establish connections (which is an indicator where data resides)?

Motivation: It is often highlighted that data is spread across many different countries. However, one would assume that data from IoT devices are stored locally in the same country or within the EU. One of the forensic challenges discussed in existing work is the storage of IoT data in multiple locations which then leads to a different jurisdiction [16, 251].

Results: Our results show that many IoT devices sent data to multiple countries. Moreover, most devices sent their data to the US despite our testbed being based in the UK and `Dataset-2` collected in Australia.

To answer these questions, we examined the network traffic of 32 consumer IoT devices (we purchased 17 devices; the remaining 15 were part of an existing dataset). Our methodology consists of four main parts: (1) we conducted port scans to determine if these can be exploited to gain remote access. (2) we looked at whether devices used encryption and if not, the type of content exposed. (3) we examined the communication between the mobile app and the cloud to determine if it could be easily exploited using a proxy server and finally (4) we used the network traffics' 'metadata' to identify the destination the data terminated. In summary, this chapter has made the following contributions:

- It establishes a methodology to conduct an in-depth network traffic analysis from the forensic perspective of the communication channels of 32 IoT consumer devices (Section 7.1).
- It demonstrates that remote access on the majority of IoT devices is unavailable (Section 7.1.3).
- It provides an insight into the types of traces obtainable from IoT traffic and the use of the metadata for forensic purposes. We facilitate this by investigating if IoT devices expose ports, whether devices and mobile apps use encryption and whether useful cleartext content was available. We show there are still many problems with IoT devices, e.g., several do not use encryption and therefore traffic can be intercepted (Section 7.2.1).

Tools	Description	Utilisation
Raspberry Pi 3	Wireless Access Point (WAP)	Capture the wireless network traffic to and from the IoT devices
TCPdump	Network traffic collection	Automate capturing network traffic
Wireshark	Packet analyser	Capture live network traffic
Huawei (ANE-LX1, Android 9)	Mobile device	Control/setup the IoT devices
PCAP Remote	Android app network sniffer for non-rooted mobile device	Capture network traffic from mobile apps
Fiddler	Proxy server to decrypts HTTPS traffic	Observe encrypted traffic in cleartext
Jadx-gui	Analysis APK files	Decompile the IoT mobile application
Network Miner	PCAP analyser	Search for cleartext within PCAP files
Ent	Entropy test	Calculates entropy value

Table 7.1: Apparatus utilised in network traffic analysis experiments.

- It identifies an easier method to identify unencrypted traffic, including 4 tests to test the randomness of data streams (Section 7.2.1).
- It shows that data from IoT devices may be distributed around the world making it difficult to seize evidence due to jurisdictional challenges (Section 7.3.3).
- It introduces our novel tool `IoT Network Analyzer` to aid forensic investigators in analysing IoT traffic, to determine the entropy, data destination, usage of ports and extraction of cleartext (Section 7.4).
- It evaluates the performance and capabilities of the tool through experimental studies (Section 7.4.3).

In summary this chapter is organised as follows: first, we outline the methodology and the process to conduct the experiments in Section 7.1, followed by the results from the various experiments in Section 7.3. Section 7.4 briefly summarises the tool we developed. Finally, we conclude with a summary of the chapter in Section 7.5.

7.1 Methodology

The apparatus used for the various experiments are presented in Table 7.1. Figure 7.1 depicts the various experiments undertaken and the following stepwise procedure was used:

IoT device selection: 17 IoT devices were selected that represented a wide range of different categories with details provided in Section 7.1.1.

Datasets: For our experiments, we used network traffic obtained from 2 sources. The first source was data we collected from 17 IoT devices connected to our testbed, the second source was from an existing dataset. Given that IoT devices often come with supporting apps, e.g., for configuration, capturing all communication channels required various setups. These are explained in Section 7.1.2. The datasets were primarily used for the "utilising encryption" (see Section 7.2.1) and "network traffic metadata"(see Section 7.2.3) experiments.

Port scanning: After setting up all devices, we carried out port scans, as shown in Figure 7.1 (1) port scans were completed to determine which ports were open on the IoT devices. We used `Nmap` in quick scan mode for open TCP and UDP ports, using the following command `nmap -sS -sU -p 0-65535 [deviceIP]` (all ports). This step served as an active approach to analyse the devices. Once the port scan was completed, we tried to connect to open ports using appropriate software, e.g., a browser for port 80 / 443, an SSH-client for port 22, and so on.

Network traffic analysis: Figure 7.1 depicts the analysis of the network traffic, which consisted of (2) utilising encryption, (3) HTTP proxy and network traffic metadata. Details are provided in Section 7.2.

Results: The results are then presented in Section 7.3.

Tool creation: In parallel to analysing the data, a tool was created that helped us to handle the data more effectively. The tool is shared and will enable examiners to automatically extract cleartext data and identify where the data terminated (Section 7.4).

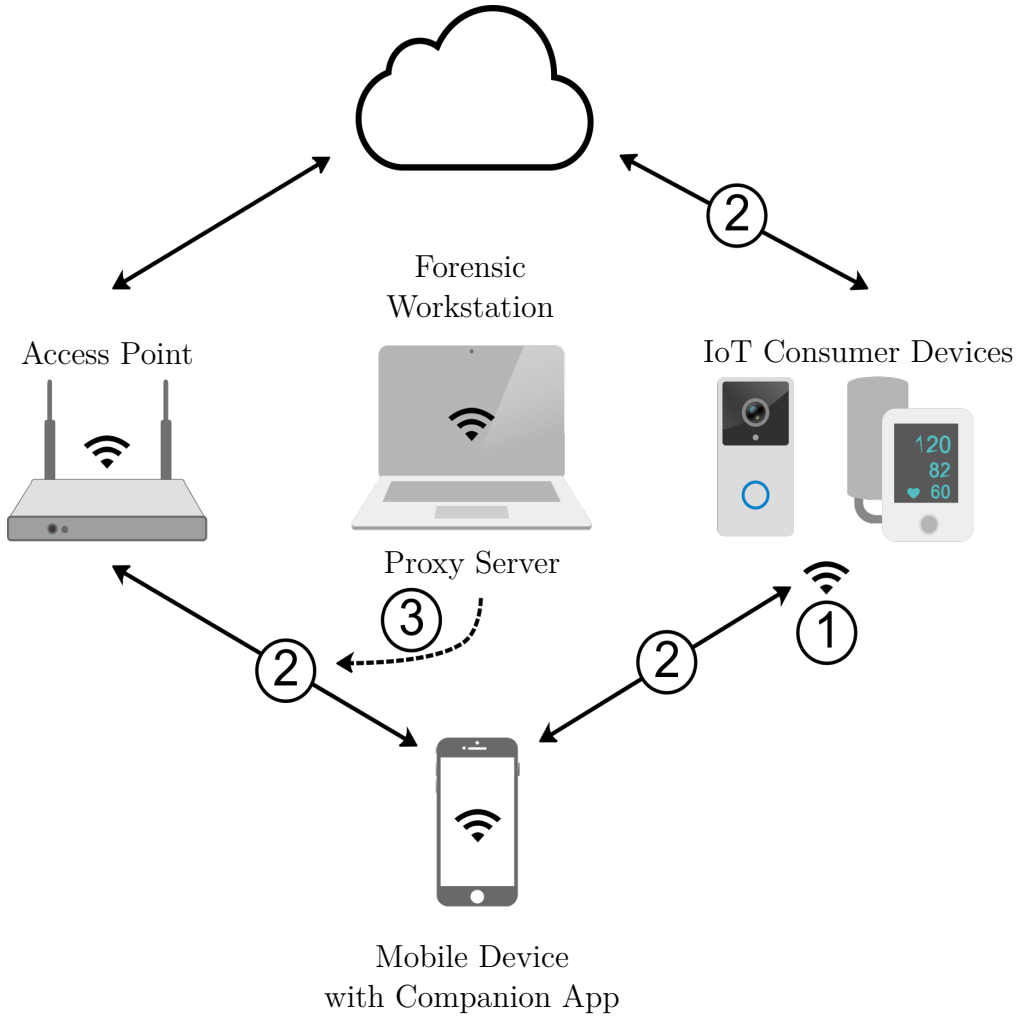


Figure 7.1: Overview of the network traffic analysis experiments

7.1.1 IoT Device Selection

As previously mentioned, 17 IoT devices were selected, these had either wired or had wireless connectivity. In detail, we had: 3 smart hubs, 5 home automation devices (3 smart plugs, 2 smart bulbs), 6 smart cameras and 3 voice assistants. We also connected non-IP devices to the smart hubs that supported other communication protocols such as Zigbee, Bluetooth and Z-wave devices. We had three major criteria when selecting a device:

Variety of families: The devices had to belong to different device families. Con-

sequently, our selection included hubs, cameras, switches and smart speakers from different device manufacturers. This ensured that we had a represented sample of the devices available on the market.

Popularity: For each category of device we searched popular retail outlets, e.g., Amazon, and selected devices based on price, popularity, average customer rating and reviews. This ensured that we selected devices that were most likely to be used by the consumer.

Compatibility with virtual assistants: When looking into the popularity of devices, we found that users favoured devices compatible with Amazon Alexa or Google Home Mini. Thus, if we had a choice between two devices, we chose the one that was compatible. This ensured that our experiential testbed was as close to a consumer setup as much as possible.

Note, during our research we included two devices with previous security concerns, these were the Victure and Wansview camera. The Victure camera had over 9000 customer reviews[1] and was said to have malware installed on the device. While others were concerned regarding the sharing of personal data [252]. Similarly, a reviewer of the Wansview camera collected network traffic and found their data was sent to China [253].

The devices were set up using the manufacturer's mobile apps. Additionally, if the IoT device supported Amazon Alexa or Google Home Mini, this was also set up. We argue that this was a setup found in most households (except maybe for individuals with a computer science/cybersecurity background).

7.1.2 Datasets

In these experiments we used network traffic from two datasets, with a combined total of 32 IoT devices:

[1]We understand that not all feedback is from genuine reviewers.

`Dataset-1` This contains network traffic data collected from the 17 IoT devices connected to our testbed and active for a duration of 7 days, which was separated into two phases:

Idle-phase: For most of the 7 days, the devices were in idle-phase, that is, the devices were turned on, but we did not perform any intentional interactions. The network traffic was collected on a Raspberry Pi 3 running `TCPdump`.

Interaction-phase: This phase started after having the devices in idle-phase for roughly 60 minutes and included intensive interactions with all devices to increase the amount of traffic produced. The exact number of interactions varied depending on the type of device, e.g., duration of voice command, powering on and off the device in short intervals. The type of interactions were separated into the following three categories:

1. The mobile app and IoT device were on the same network,
2. The mobile app and IoT devices were connected to separate networks (forces utilization of the cloud infrastructure), and
3. Voice commands were used to trigger Amazon Echo voice assistant.

To capture traffic, we used different mechanisms depending on the setup:

Mobile app-to-device/Mobile app-to-cloud: During this experiment, we used the Android app: PCAP Remote [2] , which allowed the network traffic to be saved in a PCAP file. These were then transferred to a workstation for further analysis.

Device-to-cloud: To intercept the network traffic between the device and cloud, we set up a Raspberry Pi 3 as a Wireless Access Point (WAP) and then connected to a switch that allowed port mirroring. All traffic was captured running `Wireshark` on a connected Windows 10 workstation.

[2] `https://play.google.com/store/apps/details?id=com.egorovandreyrm.pcapremote&hl=en_GB`

Dataset-2 This is an existing dataset created by Sivaraman et al. [224]. Originally, the authors used the dataset to classify IoT devices traffic using machine learning, e.g., whether it was a light bulb or a home assistant. They stated their experiments ran between 1^{st} October 2016 and 13^{th} April 2017 and collected traffic from 28 IoT devices (cameras, switches, hubs etc.). However, the only datasets publicly available were between 23^{rd} September 2016 and 12^{th} October 2016[3]. Given the sheer amount of data, we randomly selected 7 days from the dataset (24^{th}/25^{th}/26^{th}/28^{th}/30^{th} September 2016 and 4^{th}/5^{th}/6^{th} October 2016) and excluded devices that overlapped with our experiments. This left us with 15 new devices. The dataset was not labelled, we were therefore unable to determine the idle and interactive states. Remark: this was the only dataset that included complete PCAP files. Other datasets from captures were reduced down to features and thus they are primarily relevant for machine learning. The devices from `Dataset-1` and `2` were merged to create the 32 IoT devices we used for the experiments.

[3] `https://iotanalytics.unsw.edu.au/iottraces`

Category	Device Model	Open TCP ports	Open UDP ports	Vulnerable ports
SH	Samsung SmartThings hub (v2)	8889,8890,39500	123,1900,5353	-
	Phillips Hue Bridge	80,443,8080	1900,5353	80 (HTTP)
	Vera hub	22,53,80,3480,49451	Closed/Filtered	22 (SSH),80 (HTTP)
HA	iBlockcube smart plug	6668	49154	-
	Amazon smart plug	Closed	Filtered	-
	TP-Link switch (HS110)	9999	Filtered	-
	TP-Link bulb (LB100)	9999	Filtered	-
	LE LampUX	6668	49154	-
C	TP-Link cam (KC100)	9999,10443,18443.19443	514	-
	D-Link cam (DCS-932LB)	80,443,8323	Closed/Filtered	80 (HTTP)
	Xiaomi cam	Closed	5353	-
	Yi cam	Closed	Closed	-
	Wansview cam (Q5)	8080, 554	3702, 16680, 17077, 28683, 19332, 19482, 19504, 23004, 33249, 332249, 41081, 41446, 58640	-
	Victure cam (PC530)	8080,554	65000, 67, 782, 1064, 2148, 4672, 6970, 6971, 19047, 20411, 20679,21131, 21354, 23176, 33744, 40724, 46836, 49199, 53037	65000
VA	Amazon Echo (2nd gen)	4070	5353	-
	Amazon Echo (3rd gen)	4070,4071,55442,55443	Filtered	-
	Google Home Mini	8008,8009,8012,443,9000,10001	68,5000,5353	-

Table 7.2: Open TCP and UDP ports identified on the IoT devices.

7.1.3 Port Scanning

To answer RQ5.1, we carried out port scans to identify open ports on the IoT devices, which provide users access to services on the device. The results from the port scans are shown in Table 7.2 and these found that most devices used well known and proprietary ports (port range from 0-to 49151). Furthermore, we found 8 of the 17 devices used 'upper' TCP/UDP ports which ranged 49152-to 65535, 3 devices used TLS/SSL (443), 2 had port 80 open that is typically used to run an HTTP and 1 allowed SSH (22, the `Vera Plus hub`). The root password to access SSH for this device was written on the hub, so it was easy to gain root shell access. The `Victure cam` exposed a large number of TCP (2) and UDP (19) ports, in contrast to the 2 smart cameras `Xiaomi and Yi` where all ports were closed. This is beneficial from the security perspective but prevents an investigator from gaining remote access to the device and acquiring the filesystem using traditional forensic tools. Often proprietary ports were used to communicate with the app, for instance, the `TP-Link devices` had port 9999 open in order to control the device using the mobile app.

We found the `Victure cam` had UDP port 65000 open, a common port used by a specific trojan (Devil v1.3) [254]. Also, both `Victure and Wansview` cameras had port 554 open. This port is used for Real-Time Streaming Protocol (RTSP) and can be potentially exploited by sending specially crafted RTSP packets [255]. Overall, we found most of the devices did not have remote ports open. This contrasted with [90] study where they found that access through Telnet and SSH was prevalent; [256] reported a leaked list containing over half a million credentials that allowed accessing IoT devices via telnet dated October-November 2019 (the age of the devices was unknown).

7.2 Network Traffic Analysis

The analysis was divided into 3 parts: utilising encryption, HTTP proxy and network traffic metadata.

7.2.1 Utilising Encryption.

As an initial step, we manually analysed all captured network traffic using `NetworkMiner`[4] and `Wireshark`. In `NetworkMiner` we used the cleartext dictionary file to carry out a customised search, in `Wireshark` we carried out a string search (in various encoding) on the network traffic of each device. We searched for device identifiers (e.g., MAC address, serial numbers) and personal information created during setup (e.g., names, email address, passwords, usernames).

To establish an easier method to identify if the traffic was unencrypted, we tested 4 different schemes commonly used in testing the randomness of data streams: Shannon Entropy test, Monte Carlo, Average test and Serial Correlation Coefficient. We conducted tests to find which performed best at identifying unencrypted/encrypted traffic. To provide ground truth we randomly selected 6 PCAP files (captured from 6 different IoT devices) that had previously been manually analysed, 3 with cleartext and 3 with encrypted traffic. Next, we extracted the IP payload from the header of each HTTP/TCP packets using a Python script developed using `Scapy`, the tests were calculated using the tool `Ent`[5]. In the next section we briefly discuss the 4 schemes:

Shannon Entropy test. This measures the unpredictability of the payload, with the values of the test between 0-8. A high entropy value (closer to 8) indicates there is randomness in the payload and is most likely encrypted, while cleartext will exhibit low entropy (closer to zero), however, there is no definitive threshold, Entropy can be expressed as [257]:

$$H(X) = -\sum_{i=1}^{n} P_(xi)\, log\, P(x_i)$$

Entropy is calculated from the payload where X is a random variable that takes on possible values $x_1 x_2, ..., x_n.p(x_i)$ is the probability that $X = x_i$.

[4] `https://www.netresec.com/?page=NetworkMiner`
[5] `https://github.com/rsmith-nl/ent`

PCAP file	Entropy	Arithmetic Mean	Monte Carlo	Serial Correlation Coefficient
Encrypted PCAP 1	7.97	124.08	3.18	0.06
Encrypted PCAP 2	7.99	126.86	3.15	0.11
Encrypted PCAP 3	7.87	113.11	3.32	0.18
Unencrypted PCAP 4	4.78	60.93	3.99	0.58
Unencrypted PCAP 5	6.90	89.13	3.51	0.36
Unencrypted PCAP 6	5.69	77.14	3.99	0.37

Table 7.3: Results from experiments using the 4 randomness tests

Monte Carlo value π test. This method computes an approximation to the number π using the bit values of the traffic. The closer to π, the more likely the payload is encrypted [258].

Average test. This test computes the summation of the values of the bytes, the closer the value to 127.5 the more randomness (encrypted).

Serial Correlation Coefficient. This test computes the inner correlations that can reveal periodic behaviour. The randomness is checked by calculating the auto-correlation of the values, if the value is close to zero this indicates it is encrypted [259].

Details. Figure 7.3 presents the results from evaluating the 4 schemes. The results showed that the schemes performed as they should. However, we selected to use the Shannon Entropy test, as it is the most commonly used and accepted method of identifying encrypted packets in data streams [89, 260]. We set the entropy test threshold to 7 to avoid missing unencrypted traffic.

7.2.2 HTTP Proxy

We used an HTTP proxy to intercept TLS/SSL connections that decrypted the cleartext contents of encrypted traffic between the mobile app-to-cloud. A proxy server `Fiddler`[6]was set up and the Fiddler Certificate Authority (CA) certificate was installed on the mobile device (Huawei P20 Lite) so that the HTTPS traffic

[6]`https://www.telerik.com/fiddler`

could be decrypted. Additionally, we ran intensive user interactions during data collection, such as login/logon, on/off commands, etc. The collected data was then manually analysed in `Fiddler`. Overall we examined 13 mobile apps as some can be used to control more than one device, e.g., the Withings app is used to control the scales, blood pressure monitor and sleep sensor.

7.2.3 Network Traffic Metadata

In addition to the payload, the metadata was utilised where we primarily focused on the location of the connected cloud services. Therefore, a Python script was developed to extract the destination IP address, host field and the number of bytes sent to that server from each IoT device, as each IoT device can contact many different servers/destinations. We used the destination IP address to identify the location using the GeoIP[7] database and the host address using WHOIS [8] data to identify the IoT cloud infrastructure.

Where possible, we identified the type of cloud provider for an IoT device (e.g., Amazon AWS, Azure). Lastly, we examined the existing dataset whose testbed was based in Australia to identify the final destination of the data.

[7] `https://dev.maxmind.com/geoip/geoip2/geolite2/`
[8] `https://whois.domaintools.com/`

		Device-to-cloud			Mobile app-to-cloud			Mobile app-to-device		
Category	**Device Model**	**Entropy**	**Cleartext**	**Protocol**	**Entropy**	**Cleartext**	**Protocol**	**Entropy**	**Cleartext**	**Protocol**
SH [1]	Samsung SmartThings hub (v2)(wired)*	7.78	✗	TLSv1.2	-	✗	TLSv1.2	-	✗	TLS1.2
	Phillips Hue Bridge(wired)* [3,4]	7.72	✓	HTTP/ TLSv1.2	7.99	✗	TLSv1.2	7.81	✗	TLS1.2
	Vera plus hub(wired)*[2]	7.87	✗	TLSv1.2	6.24	✗	HTTP/TLS	6.54	✓	HTTP
HA [1]	iBlockcube smart plug*	7.74	✗	TLSv1.2	7.22	✗	TLSv1.2	7.45	✗	IPDC
	Amazon smart plug*	7.80	✗	TLSv1.2	-	✗	TLSv1.2	7.54	✗	-
	TP-Link plug(HS110)*	7.74	✗	TLSv1.2	7.74	✗	TLSv1.2	7.56	✗	TLSv1.2
	TP-Link bulb(LB100)*	7.72	✗	TLSv1.2	7.20	✗	TLSv1.2	7.23	✗	TLSv1.2
	LE LampUX*	7.87	✗	TLSv1.2	7.28	✗	TLSv1.2	7.91	✗	IPDC
	LiFX lightbulbs†	6.12	✓	TLSv1	-	✗	-	-	✗	-
	iHome†	7.59	✗	TLSv1.2	-	✗	-	-	✗	-
	Nest Protect smoke alarm†	7.67	✗	TLSv1.2	-	✗	-	-	✗	-
VA [1]	Amazon Echo(2nd gen)*	7.99	✗	TLSv1.2	-	✗	TLSv1.2	-	✗	-
	Amazon Echo(3rd gen)*	7.97	✗	TLSv1.2	-	✗	TLSv1.2	-	✗	-
	Google Home Mini*	7.99	✗	TLSv1.3	7.72	✗	TLSv1.2	7.91	✗	TLSv1.2

[1] Smart Hubs (SH), Home Automation (HA), Smart Cameras (SC), Voice Assistants (VA), Smart healthcare and Miscellaneous (M)
[2] On the Vera hub we connected the Aeotec Door/Window sensor Gen5 (ZW120-C), this device uses Z-wave. In order to generate data as none of the other devices were compatible with this hub
[3] On the Phillip Hue Bridge we connected a Phillips lightstrip that uses Zigbee
* Devices from `Dataset-1`
† Devices from `Dataset-2`

Table 7.4: The 32 IoT devices used in the experiments, where `Dataset-1` and `Dataset-2` were merged.

		Device-to-cloud			Mobile app-to-cloud			Mobile app-to-device		
Category	**Device Model**	**Entropy**	**Cleartext**	**Protocol**	**Entropy**	**Cleartext**	**Protocol**	**Entropy**	**Cleartext**	**Protocol**
VA [1]	Amazon Echo(2nd gen)*	7.99	✗	TLSv1.2	-	✗	TLSv1.2	-	✗	-
	Amazon Echo(3rd gen)*	7.97	✗	TLSv1.2	-	✗	TLSv1.2	-	✗	-
	Google Home Mini*	7.99	✗	TLSv1.3	7.72	✗	TLSv1.2	7.91	✗	TLSv1.2
SC [1]	TP-Link cam(KC100)*	7.99	✗	TLSv1.2	7.97	✗	TLSv1.2	7.81	✗	TLSv1.2
	D-Link cam(DCS-932LB)(wired)*	7.82	✗	TLSv1	7.78	✗	TLSv1.2	6.40	✓	HTTP
	Xiaomi cam*	7.91	✓	HTTP	7.91	✗	TLSv1.2	-	✗	-
	Yi cam*	7.92	✗	TLSv1.2	7.59	✗	TLSv1.2	-	✗	-
	Netatmo Welcome camera†	7.20	✗	TLSv1.2	-	✗	-	-	✗	-
	TP-Link Day Night Cloud camera†	7.41	✗	TLSv1.2	-	✗	-	-	✗	-
	Samsung SmartCam†	6.05	✓	HTTP/ TLSv1.2	-	✗	-	-	✗	-
	Nest Dropcam†	7.78	✗	TLSv1.2	-	✗	-	-	✗	-
	Insteon Camera(wired)†	6.57	✓	HTTP/ TLSv1.2	-	✗	-	-	✗	-
	Victure cam (PC530)*	6.06	✗	-	6.45	✗	HTTP	5.74	✗	HTTP
	Wansview cam (Q5)*	7.86	✓	TLSv1.2	5.86	✗	HTTP/TLSv	6.68	✗	HTTP
M [1]	Netatmo weather station†	7.40	✗	-	-	✗	-	-	✗	-
	Triby Speaker†	7.59	✗	TLSv1.2	-	✗	-	-	✗	-
	PIX-STAR Photo-frame†	7.48	✗	TLSv1.2	-	✗	-	-	✗	-

[1] Smart Hubs (SH), Home Automation (HA), Smart Cameras (SC), Voice Assistants (VA), Smart healthcare and Miscellaneous (M)
[2] On the Vera hub we connected the Aeotec Door/Window sensor Gen5 (ZW120-C), this device uses Z-wave. In order to generate data as none of the other devices were compatible with this hub
[3] On the Phillip Hue Bridge we connected a Phillips lightstrip that uses Zigbee
* Devices from `Dataset-1`
† Devices from `Dataset-2`

Table 7.5: The 32 IoT devices used in the experiments, where `Dataset-1` and `Dataset-2` were merged (continued).

7.3 Results

In this section, we present the results from the analysis of whether the device utilises encryption, HTTP proxy and lastly the network traffic metadata.

7.3.1 Utilising Encryption

This section addresses RQ5.2, where we examined 32 IoT devices, with the results shown in Tables 7.4 and 7.5. We found the majority had secure communication channels, especially when the mobile app communicated with the cloud.

We examined the unencrypted network traffic for evidence potentially useful to an investigation. Although the majority of the devices encrypted their content, unexpectedly, some devices during updates would send in cleartext video with unique identifiable data of the detected motion. Note, these devices mostly used encrypted channels but performed some actions using unencrypted channels.

Details Many of the devices used secure protocols TLS/SSL. However, we found 9 devices transmitted to the cloud or mobile app with no encryption (HTTP) or partially encrypted (TLS/SSL and HTTP). 7 devices used no encryption between the device-to-cloud and 3 devices used no encryption between the mobile app-to-device. We found the `LE LampUX lightbulb` and the `iBlockcube plug` used Internet Protocol Device Control (IPDC) communication protocol, which is unusual as this is typically used for Voice Over IP (VoIP). A reason for this might be that VoIP protocols are always prioritised by the router reducing latency in the device [261].

To identify unencrypted communication, we examined with the connections that had an average entropy score of 7 or below these were: `Vera plus hub`, `LIFX bulb` [9], `D-Link camera`, `Samsung camera`†, `Insteon camera`†, `Victure cam`, `Wansview cam`, `Withings scale`†, `Withings monitor`† and `Withings sleep sensor`†, the results of the entropy test are shown in Table 7.4 and 7.5. While the entropy test correctly detected the devices that used unencrypted traffic, the `LiFX lightbulb`† used encrypted protocols (TLS/SSL). It was discovered in previous research that the

[9]† Marked IoT devices are from the expanded test scenario.

```
1  PUT /ipc009-video-storage/2019/12/17/6272883219/283973403\
       _195222622.mp4\?
2  Expires=1576585342000&
3  GalaxyAccessKeyId=5371742894246&
4  Signature=GLNOHtIhIRqzzFAWRNTOs7rAjZk= HTTP/1.1
5  Host: awsde0.fds.api.xiaomi.com
```

Figure 7.2: Snippet taken from the unencrypted HTTP PUT request from the Xiaomi camera.

reason for the low entropy of the `LiFX lightbulb`† was because it was encoded and not in a human-readable format [90]. We analysed the remaining traffic and identified the `Xiaomi` and `D-Link cameras` that did not using encryption but had higher entropy scores: `Xiaomi camera` communicated to the cloud using no encryption and the `D-Link camera` communicated to the mobile app in cleartext. Both of these devices showed high entropy scores as they use video compression [260]. We found that when the `Xiaomi camera` detected motion, the unencrypted video, MAC address and timestamp were sent in cleartext through an HTTP PUT request packet, a snippet of this is shown in Figure 7.2. Also present in the header were the access key ID and signature, which can provide investigator access to the Amazon Web Services (AWS) account. We also found that when the mobile app for the `D-Link camera` was activated, cleartext was present during live streaming between the device and the mobile app and partial JPEG images were present in the HTTP header.

The 4 devices that communicated with the mobile app in cleartext included; `Vera hub`, `D-Link cam`, `Victure cam` and `Wansview cam`. The only interesting finding was from the `Victure cam`: HTTP POST requests that included the API access token and key.

Next, we examined the 7 devices that communicated with the cloud in cleartext, 2 of which were smart cameras. The first was the `Samsung camera`† that sent unencrypted HTTP POST requests to the cloud, exposing unique identifiers including, MAC address, username, serial number, timestamp and user-specific device name (e.g., 'smarthomeunsw'). The second camera an `Insteon`† also displayed cleartext information such as port numbers, MAC address, public IP address and a unique ID. The remaining 3 devices were smart healthcare devices that required

```
1         "id":11020336, #Numeric ID of user
2         "sn":"SMA", #Username of user
3         "wt":51.701, #Weight(kgs)
4         "ht":1.68, #Height(meters)
5         "agt":30.1, #Age in years
6         "sx":0, #Sex of user(0=male, 1=female)
7         "cr":1472696125, #Unix timestamp when account created
8         "lg":"fr_FR", #Language French
9         "utc":1474829944 #Unix timestamp of last measurement (Unix
             )
```

Figure 7.3: Snippet taken from the cleartext HTTP POST request of the Withings smart scale.

personal data such as height, weight, postcode/zip code etc. All this data is sensitive and helpful not just in identifying the user but also their physical characteristics. The 3 devices manufactured by `Withings`† (Sleep, baby monitor and scales) all sent cleartext data through HTTP POST requests. Although the `Withings baby monitor` and sleep† did not contain any sensitive cleartext information, the `Withings smart scales`† displayed a considerable amount of user information, e.g., weight, height, as shown in Figure 7.3. Furthermore, the API for the scales had depreciated, to understand the data we used the following source `https://blog.chris007.de/hacking-the-withings-wifi-body-scale-2/`.

We unexpectedly found that when the `Triby speaker`† communicated with the cloud during an update, the HTTP GET request was displayed in cleartext and included information such as MAC address, username and serial number as shown in Figure 7.4. We did not find any sensitive cleartext data on the `Phillip Hue Bridge` or `Vera Plus hub`. However, these provide a central gateway to connect other devices so when new sensors/devices are connected in the future, we expect cleartext data to be identified. This is especially true for the `Phillips Hue Bridge` that only used partial encryption on the device-to-cloud communication channel.

Previous research by Loi et al. [90] studied 20 IoT devices and found 5 that communicated in cleartext, 3 of these devices were smart cameras. This corresponded with our findings where we found 6 of the 8 devices that sent cleartext were smart cameras. A possible reason for this is that smart cameras have more features and

```
1 GET /update/triby/update.pup?mac=18:B7:9E:02:20:44&machine=
    triby&board_name=TRIBY_V0&revision=5&sub_revision=4271&
    version=triby-10.48.1&version_build=52.3a.e0&up=691259&
    hwcap=1:0&feature_tag=&board_flags=c:1,ci:0,vl:0,cv:0,p
    :1 HTTP/1.1
2 Host: developer.invoxia.com
```

Figure 7.4: Snippet taken from the unencrypted HTTP GET request during an update of the Triby speaker.

services compared to other devices such as smart plugs. For instance, `TP-Link plug` had 1 TCP port open while the `TP-Link camera` had 4 TCP ports open.

7.3.2 HTTP Proxy

Mobile apps to cloud. In this section, we used a proxy server to examine the encrypted contents of the network traffic between the mobile apps and the cloud (see RQ5.4). We found several of the mobile apps allowed a proxy connection while the remaining implemented certificate pinning. We found a case where a device would unexpectedly take snapshots and there was no setting to control this behaviour. On the same mobile app, the username and password were sent encoded in Base64. As this mobile app controls several devices, this meant we were able to gain access to them all.

Details We observed the decrypted communication between the mobile app-to-cloud and found 7 of the 13 apps allowed proxy connections. The remaining apps did not complete a connection with their servers as they had certificate pinning implemented. This involved the coupling of a host's trusted credentials to its identity (e.g. an X.509 certificate or public key) [18]. We decompiled the apps and then searched for keywords such as 'x509' and 'checkServerTrusted' and found further evidence of certificate pinning.

The apps iLiving-iBlock (iBlockcube smart plug device) and LampUX (LE LampUX device) allowed a proxy connection and normal device use, e.g., turn on/off, although we did not find any sensitive data transmitted by these devices. Additionally, when we opened the YI cam app, it would send a list of URLs that

```
1        POST /data/LINKIE.json HTTP/1.1
2        Content-Type: application/x-www-form-urlencoded
3        Authorization: Basic XXXXXXXXXXXXXXXXXXXXXXXXXX
4        XXXXXXXXXXXXXXXXXXXXXXX
5        User-Agent: Dalvik/2.1.0 (Linux; U; Android 9; ANE-LX1 Build
             /HUAWEIANE-L21)
6        Host: 192.168.1.151:10443
```

Figure 7.5: Snippet taken from the intercepted encrypted HTTPS POST request of the TP-link cam. The values have been obfuscated for confidentiality reasons.

contained the motion-captured, username, user ID and API key. The Kasa app controlled the `TP-Link devices (lightbulb, switch and camera)`, but only the camera routed useful data through the proxy. There was an unusual activity of the `TP-Link camera` when the app was opened: it took a snapshot that included a timestamp and URL link to the JPEG snapshot (note we were not able to control/disable this functionality). From the same mobile app, we captured HTTP GET request packets that exposed the basic authentication field that contained the username and password to login for the device. This was encoded in Base64 and is shown in Figure 7.5. Obtaining the password for one account could allow an investigator to access other IoT device accounts. Especially as [262] found that 52% of users reused their passwords for many online services.

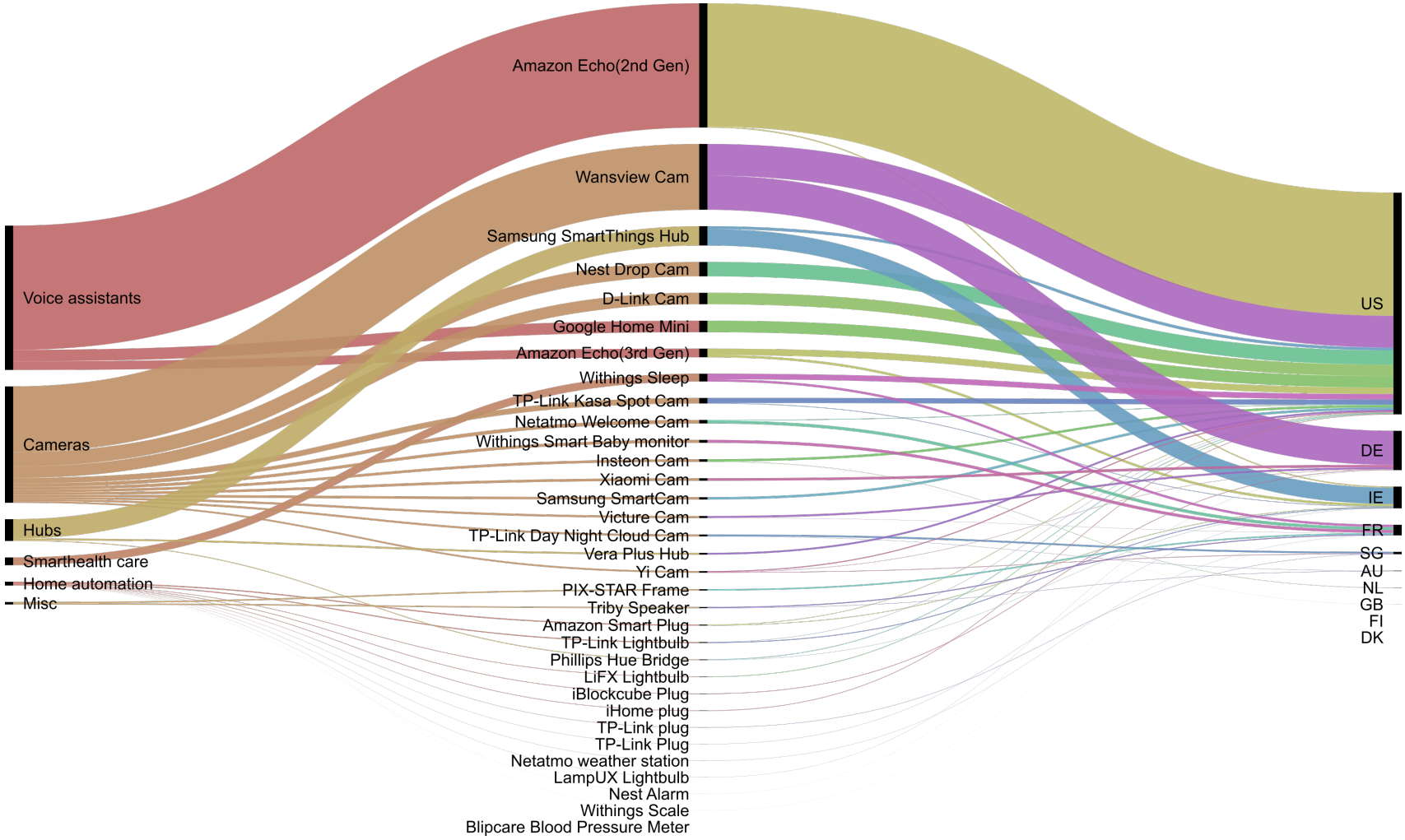


Figure 7.6: The top 10 data destinations grouped by the number of bytes transmitted by each device to their final destination.

Country	Number of Devices
United States	26
France	14
Germany	14
United Kingdom	13
Ireland	11
Netherlands	8
Australia	7
Singapore	5
Denmark	5
Finland	4
India	4
Japan	3
Mexico	3
Romania	3
China	3
Brazil	2
Russia	2

Table 7.7: The number of devices connected to each country

7.3.3 Network Traffic Metadata

In the following section, we focused on RQ5.3 and identified the destinations that the data transversed. Figure 7.6 shows the flow of traffic to the top 10 countries with the height of the bands corresponding to the number of bytes sent by each IoT device. Overall, the data for 26 of the 32 (79%) devices terminated in the US as shown in Table 7.7. This was unexpected as the closest Amazon data centre to our testbed (UK) was in Ireland [263]. While trying to establish a reason for our data being sent to the US instead of Ireland, we found that unlike, Google and Facebook that provide approximate locations for their data centres, Amazon did not advertise this freely.

	Countries																																			
Devices	AT	AU	BG	BR	BY	CA	CH	CN	DE	DK	FI	FR	GB	HK	HU	ID	IE	IN	IS	JP	KH	KR	LU	MX	NL	NO	NZ	PL	RO	RU	SG	SK	TW	UA	US	XA	Total
Amazon Echo (3rd)	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	◉	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	2
Amazon Echo(2nd)	○	○	○	○	○	○	○	○	○	○	○	○	◉	◉	○	○	◉	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	◉	○	4
Amazon Smart Plug	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	◉	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	◉	○	2
D-Link camera	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	◉	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	◉	○	2
Google Home Mini	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	◉	○	○	○	○	○	○	○	○	○	○	◉	○	2
Insteon Cam	○	○	○	○	○	○	○	◉	○	○	○	○	◉	○	○	○	◉	○	○	◉	○	○	○	○	◉	○	○	○	○	○	◉	○	○	○	◉	○	7
LiFX light-bulb	◉	◉	◉	◉	◉	○	◉	○	◉	◉	○	◉	◉	○	◉	◉	◉	○	◉	◉	○	○	○	◉	◉	◉	◉	◉	◉	◉	◉	◉	◉	◉	◉	◉	28
Phillips Hue Bridge	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	◉	○	○	○	○	○	○	◉	○	○	○	○	○	○	○	○	○	○	◉	○	3
Samsung Smart Camera	○	◉	○	○	○	○	○	○	◉	○	○	○	◉	○	○	○	○	○	○	○	◉	○	◉	○	○	○	○	○	○	○	○	○	○	○	◉	○	6
Samsung SmartThings Hub	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	◉	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	◉	○	2
TP-Link Bulb	○	○	○	○	○	○	○	○	◉	◉	◉	◉	◉	○	○	○	◉	◉	○	○	○	○	○	○	◉	○	○	○	○	○	○	○	○	○	◉	○	9

Table 7.8: The IoT devices that contacted multiple countries

	Countries																																			
Devices	AT	AU	BG	BR	BY	CA	CH	CN	DE	DK	FI	FR	GB	HK	HU	ID	IE	IN	IS	JP	KH	KR	LU	MX	NL	NO	NZ	PL	RO	RU	SG	SK	TW	UA	US	XA	Total
TP-Link Day Night Cloud camera	○	◉	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	◉	○	○	○	◉	○	3
TP-Link Kasa Spot Cam	○	○	○	○	○	○	○	○	○	○	○	◉	◉	○	○	○	◉	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	◉	○	4
TP-Link plug	○	○	○	○	○	○	○	○	◉	○	○	◉	◉	○	○	○	○	○	○	○	○	○	○	○	◉	○	○	○	○	○	◉	○	○	○	◉	○	6
TP-Link Switch	○	○	○	○	○	○	○	○	◉	◉	◉	◉	◉	○	○	○	◉	◉	○	○	○	○	○	○	◉	○	○	○	○	○	○	○	○	○	◉	○	9
Triby Speaker	○	◉	○	○	○	○	○	○	○	○	◉	◉	◉	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	◉	○	○	○	○	○	◉	○	6
Vera Plus Hub	○	○	○	○	○	○	○	○	◉	◉	◉	◉	◉	○	○	○	○	◉	○	○	○	○	○	○	◉	○	○	○	◉	○	○	○	○	○	◉	○	9
Withings Sleep	○	○	○	○	○	○	○	○	◉	○	○	◉	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	◉	○	3
Withings Smart Baby monitor	○	◉	○	○	○	◉	○	○	○	○	○	◉	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	3
Xiaomi Cam	○	○	○	○	○	○	○	◉	◉	○	○	○	○	○	○	○	○	○	○	○	○	◉	○	○	◉	○	○	○	○	○	○	○	○	○	◉	○	5
Yi Cam	○	○	○	○	○	○	○	○	◉	○	○	○	◉	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	◉	○	○	○	◉	○	4
Wansview Cam	○	○	○	◉	○	○	◉	◉	◉	○	○	○	◉	○	○	◉	○	○	○	○	◉	○	○	○	○	○	○	◉	○	○	○	○	◉	○	◉	○	10
Victure Cam	○	○	○	○	○	○	○	○	◉	○	○	◉	◉	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	◉	○	4

Table 7.9: The IoT devices that contacted multiple countries (continued)

Tilley [264] reported that data from their Ring Doorbell was routed to servers in China. We found only 3 devices; `Insteon, Wansview and Xiaomi cameras` that sent their data to Alibaba servers based in China. This was confirmed in our findings, where in Section 7.1.1 a consumer had raised the concern that their data from the `Wansview camera` was being sent to China.

Furthermore, we found 75% (24) of the IoT devices sent data to multiple destinations, while the remaining devices sent data to a single destination. This is an interesting from an investigative viewpoint, as multiple destinations and jurisdictions, will potentially cause delays in gaining access and securing the data. Also it will require communication between countries with different legal systems.

Next, we examined the devices that contacted the most destinations the results are shown in Tables 7.8 and 7.9, these were the `LiFX lightbulb` (28) followed by the `TP-Link Bulb, TP-Link plug, Vera Plus Hub` (all had 9) and the `Insteon cam` (7). When compared to other devices, such as smart hubs and cameras lightbulbs, they have limited features, yet surprisingly they were the devices that contacted the most destinations.

To gain a better understanding of the type of cloud infrastructure data was sent to, we focused on the most frequently used. We found 21 of the 32 (66%) devices contacted a server belonging to Amazon. The reliance on Amazon servers to store data could be an issue for investigators. This was demonstrated in a recent case involving Amazon Alexa where Amazon refused to hand over evidence relating to a criminal investigation and denied the request in the absence of a valid or binding legal demand. This could also be an issue for future cases when dealing with Amazon [265].

7.4 Tool Creation and Evaluation: `IoT Network Analyzer`

In Chapter 3, we undertook a survey to establish a clearer roadmap for research. One of the key findings was that research should focus on the development of IoT forensic tools. Following this, in Chapter 4 we reviewed existing IoT forensic tools to establish the type of tools required. We found the requirement for tools to

analyse network traffic captured from IoT devices. In this section, we fulfil this requirement by introducing the tool `IoT Network Analyzer`, which is constructed in Python to automate the process.

To the best of our knowledge there is no tool that has all the features that our tool offers. Although our results can be found using separate open-source tools, it would require an investigator a considerable amount of time to manually extract the data. Our tool has made the following contributions to the state-of-art:

- Entropy calculation: It can automatically calculate the entropy-based on the payload, whereas the tool Ent requires manual filtering of the IP address and extraction.
- Usage of secure ports: It can compare the ports used against a pre-defined list of ports that are known to be secure. The list of secure ports can also be modified to include other port numbers.
- Geolocation: In our tool, we have used the IPStack geolocation database, whereas in Wireshark the only option is to use GeoLite2(free) or GeoIP2(commercial). Since our tool is open-source the source code can be easily modified to add other geolocation databases.

As explained in Chapter 4, we found only 53% (33/62 tools) were publicly available. Therefore, to ensure our tool is available to the public we uploaded our tool to the a public repository; Github. **The tool is available at:** `https://github.com/Dyvels/IoTAnalyzer`

7.4.1 Tool Development

To develop the tool, we followed a Software Development Life Cycle (SDLC) to plan, manage and control the development process. Specifically, we decided to use Waterfall (shown in Figure 7.7), a commonly used SDLC model. This is a sequential software development process that comprises of five stages that flow downwards and must be executed in order [266]. When developing the tool, we

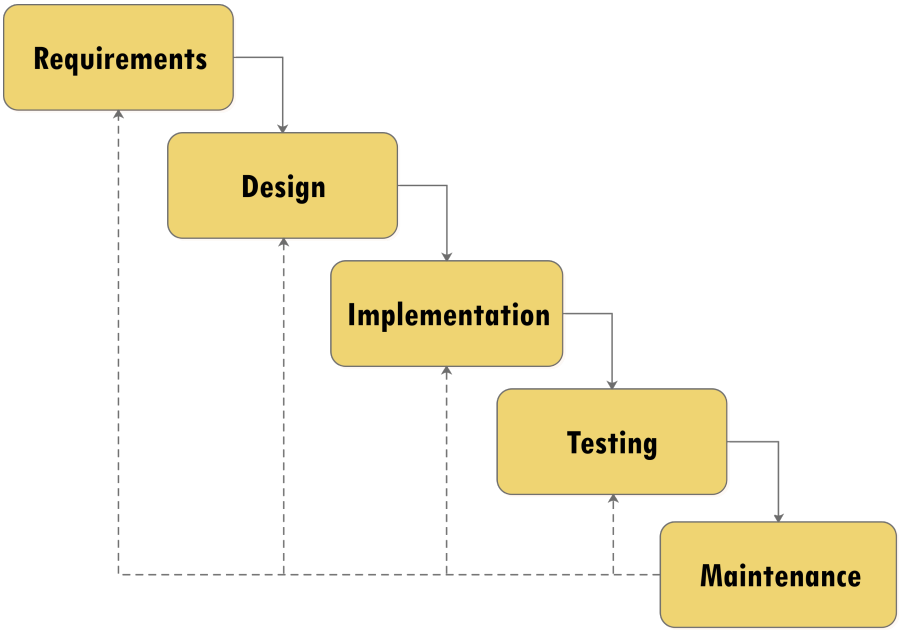


Figure 7.7: Illustrating the Waterfall software development model [266]

took into consideration the findings from Chapter 4, by including a code review, documentation, licensing and evaluation of the tool.

Requirement identification: This is where the description of the functional and non-functional requirements of the software was developed. The features of the tool were developed from our large-scale study where we found a method was required to easily identify encrypted/cleartext packets and the location of the data. In addition, we added other features to aid an investigator in analysing IoT network traffic. The following are the features we identified:

- Entropy Calculation: To identify sessions that are in cleartext, the Shannon entropy is utilised and calculated over all packets within a session. This calculation was only carried out on the data portion of the segment; header fields such as ports, the IP addresses are ignored.

- Location of Data: The source and destination IP addresses are extracted to make an assumption of their geolocation and is accomplished using IP Stack[10] that is a "powerful, real-time IP to geolocation API".

[10] `https://ipstack.com/product`

- Usage of Secure Ports: A list of ports with the total number of occurrences for each port was created and then checked against the pre-defined list of 22 secure ports (Appendix E.1), which can be modified if required. This information is then mapped with the corresponding source and destination IP addresses.
- Cleartext Extraction: Our tool attempted to extract cleartext.
- IP addresses and Connections Overview: This feature provided a list of the source and destination IP address and their respective number of packets sent.
- Packets Overview: This feature provided a structure of the different packets and outputs a list of the number of packets, which included: the total, raw layer, UTF8, byte, ARP, DHCP, DNS, and finally the total amount of packets processed.
- Command Line Interface (CLI): The tool has an interactive CLI based on the cmd2[11] library. This provided the interactive usage of the tool, data extraction and output capabilities.

Design stage: This was the planning and problem-solving process. When designing the tool, we applied the knowledge gained from Chapter 4, where we found several issues with current research-based tools. For instance, a lack of documentation and licensing. To overcome these when developing our tool we planned the following: code review, programming language, documentation and licensing.

Implementation: At this stage, we used the requirements developed during the design stage and implemented the requirements and design specifications into a physical prototype tool. We carried out the following:

- Code review: We adopted the approach explained in Chapter 4, where we carried out a code review of the tools using Codacy. The test revealed 98

[11] `https://github.com/python-cmd2/cmd2`

issues, these concerned coding style (95) and security (3) (Note. No issues were found in relation to error-prone, performance, compatibility and unused code). Since coding style is subjective and has no bearing on the functionality of the tool we chose not to address these issues. Instead, we focused on fixing the 3 security issues, these were related to the hardcoded binding to all network interfaces. This meant all incoming connections were accepted from anywhere [12], which can potentially open a service to traffic on unintended interfaces, that may not be properly documented or secured. To overcome this, we modified the hardcoded binding to only accept connections from machines on the local network.

- Programming language: As previously discussed we found many of the tools were developed in Python (Chapter 4). Therefore, we opted to use Python, as it is also good for rapid prototyping [267] and the Python libraries are available for the features we required in the tool e.g., Scapy, cmd2.

- Documentation: As there was no formal format for documentation, we used the Github readme file to include the following information: project name, description, installation, usage, contributions, credits and license.

- License: We used Github licensing of the source code to allow easy redistribution. We decided on the General Public License (GNU) v3.0 [13] as it is the most widely used free software license and allows *"the software to be used for any purpose"*.

Testing: This was to verify and validate that the tool met the requirements and specifications. In Chapter 4, we found 74% (46 out of the 62 tools) had been evaluated but there were inconsistencies. Therefore, we elected to test `IoT Network Analyzer` to verify the accuracy, performance and efficiency of the tools'

[12] `https://bandit.readthedocs.io/en/latest/plugins/b104_hardcoded_bind_all_interfaces.html`

[13] `https://www.gnu.org/licenses/quick-guide-gplv3.html`

four main features. We evaluated it against similar tools, which is discussed further in Section 7.4.3.

Maintenance: This was the process of modifying the tool after deployment to improve performance and quality. This is an ongoing process where additional features and improvements can be made, such as the CLI to provide additional output capabilities.

7.4.2 Example usage of IoT Network Analzyer

Following the development of the tool, we now demonstrate the use of `iotnetworkanalyzer`. The tool can be called in command-line as follows:

```
iotnetworkanalyzer.py -f filename [options]
```

In this example, we invoke the Python script of the tool using the Python interpreter. The tool accepts an input PCAP file as stated by the `-f <filename>` parameter, the user can then select various options:
The `-p` parameter prints a list of ports
`-c` list of connections
`-cn` list a list of connections
`-cn` list of connections between source and destination
`-i` displays a list of IP addresses
`-g` list of IP addresses and the geolocation data
`-ct` shows the cleartext
`-e` displays a list of the calculated Entropy values

In this example, the parameter `-g`, outputs the following results that included the IP addresses and the country the data originated.

```
Overview of IPs and their geolocation: '224.0.0.22':
'country': None, '54.194.162.84': 'country': 'Ireland',
```

```
Welcome to the IoT Network Analyzer
iotA>> load -f withings_scale.pcap
start loading 'withings_scale.pcap', this may take a while depending on the size of
the file
loading finished after:  4.945235499999853
start building connection set
10 % is done
20 % is done
30 % is done
40 % is done
50 % is done
60 % is done
70 % is done
80 % is done
90 % is done
100 % is done
233 packets are not analyzed because they are DNS, NTP, ICMP or DHCP packets or didn
´t have a IP Layer
finished building connectionSet and portDicionary
Overview of IPs and their geolocation will now be created
Done
Overview of IPs their send utf-8 cleartext and the destination IP will now be create
d
start building cleartext dictionary
10 % is done
20 % is done
```

Figure 7.8: Snippets of the IoT Network Analyzer tool first loading the PCAP file to analyse

```
'192.168.1.179':  'location':  'localIPaddress',
'54.194.162.98':  'country':  'Ireland', '239.255.255.250':
'country':  None, '54.194.162.25':  'country':  'Ireland',
'192.168.1.94':  'location':  'localIPaddress', '192.168.1.42':
'location':  'localIPaddress', '192.168.1.2':
'location':  'localIPaddress'
```

To demonstrate the use of the tool, Figure 7.8 shows how to load a PCAP file. Figure 7.9 shows a snippet of how the tool can be used to overview all the network connections. Lastly, Figure 7.10 demonstrates the tool's capability to show all IP addresses and geolocation information.

7.4.3 Tool Assessment

For the evaluation we tested the tool's four main features and the performance of the tool, utilising 5 PCAP files that we knew had cleartext data and another 5 PCAP files with no cleartext data.

```
iotA>> analyze -cn
Overview of all established connections:
oirigin IP address: 192.168.1.238  destination IP address: 89.30.121.150
 callAmount: 770  used whitelisted ports: 0  used not whitelisted ports:
770
None
oirigin IP address: 89.30.121.150  destination IP address: 192.168.1.238
 callAmount: 725  used whitelisted ports: 0  used not whitelisted ports:
725
None
```

Figure 7.9: Snippet of using IoT Network Analyzer tool with an overview of all connections in PCAP file

```
iotA>> analyze -g
all IPs and their geolocation informations
{'89.30.121.150': {'country': 'France', 'region_name': 'Île-de-France', '
longitude': 2.222100019454956, 'latitude': 48.870140075683594, 'continent
': 'EU', 'city': 'Suresnes'}, '192.168.1.238': {'location': 'local IP add
ress'}}
```

Figure 7.10: Snippets of using IoT Network Analyzer tool with all the IPs and geolocation information

Entropy Calculation. To evaluate the entropy part, we compared our results to a similar tool called `Ent` and both provided identical results. Our tool has the benefit of automatically calculating the entropy based on the payload, whereas `Ent` requires manual filtering of the IP addresses and extraction of the payloads. This allows an investigator using our tool to calculate the entropy on the various communication channels at the same time.

Location of Data. As we used the external IPStack service for detecting the location, we briefly compared these results with two similar services: IP2Location-Lite[14] and Geolite2 (MaxMind)[15]. These have been suggested by Gharaibeh et al. [268] as they have an accuracy of 80% at country-level. The results are shown in Table 7.10 and show little difference between the databases, meaning any of these databases would be equally suitable. Note: a method to assess the accuracy of IPStack would be to compare it against a "ground truth" database that has true geographical location for each IP address. However, such a database does not exist [268].

[14] `https://www.ip2location.com/demo`
[15] `https://dev.maxmind.com/geoip/geoip2/geolite2/`

Devices	IPStack (IoT Network Analyzer)	GeoIP	IP2location
Withings smart scale	France	France	France
Samsung smartcam	Australia, New Zealand, US	Australia, Cambodia, Germany, Luxembourg, UK, US	Australia, Japan, New Zealand, UK, US
Vera hub	US	US	US
DLink cam	Ireland	Ireland	Ireland
Insteon cam(wired)	Australia, China, Ireland, Japan, Netherlands, UK,US,	Germany, Ireland, Singapore, UK, US	China, France, Germany, Taiwan, UK, US
TPlink cam	Germany, Ireland, Singapore, UK, US	France, Germany, Ireland, Singapore, US	China, France, Germany, UK, US
Amazon Echo	Ireland, UK, US	Ireland, UK, US,	Ireland, UK, US
Google Mini	Mexico, US	Mexico, US	UK, US
Phillips Hue	Germany, Ireland, Singapore, Spain, UK, US	Germany, Ireland, Singapore, Spain, UK, US	Germany, Ireland, UK, US
Xiaomi cam	China, France, Germany, Netherlands, Singapore, Taiwan, UK	China, France, Germany, Ireland, Japan, Netherlands, Singapore, South Korea, Taiwan, UK, US	China, France, Germany, Netherlands, Singapore, Taiwan, UK

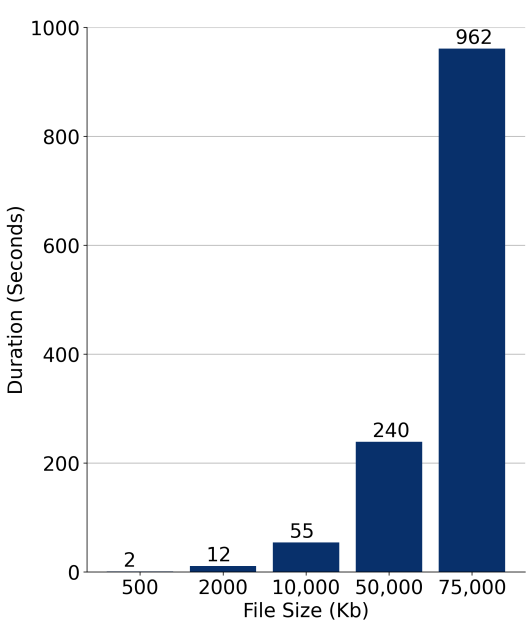
Table 7.10: Countries identified using `IoT Network Analyzer` and results from two other geolocation databases

Usage of Secure Ports. To test this feature, we analysed the 10 PCAP files using a similar tool, `T-shark` to filter for the ports and compared these results to those from our tool, which showed them to be identical.

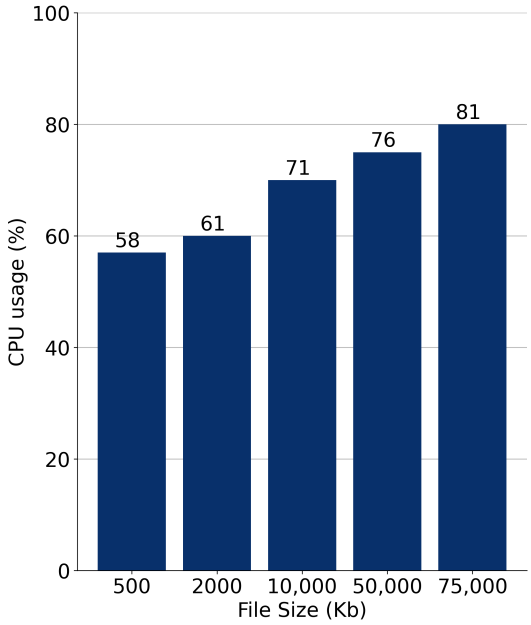
Cleartext Extraction. To identify cleartext traffic, we manually examined each PCAP file in `NetworkMiner`[16] and compared these to the results we obtained using our tool and the results were identical.

Performance Evaluation. The performance test (with respect to processing speed, Central Processing Unit (CPU) and memory usage) was conducted on a system running Windows 10 on a Intel(R) Core(TM) CPU i5-4670K with 32GB RAM. The results from processing the PCAP files of sizes from 500 - 75,000 kilobytes (kbs) are shown in Figure 7.11a, and show that it took around $1s$ to 6 minutes.

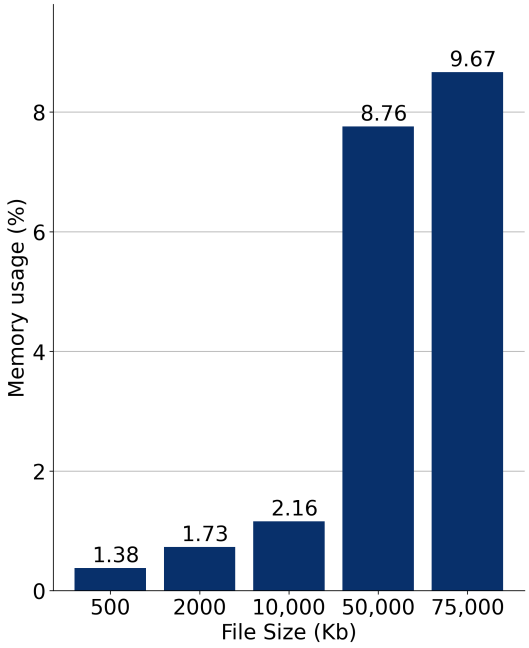
[16] `https://www.netresec.com/?page=NetworkMiner`



(a) Different sizes of PCAP files and the time required to process



(b) Different sizes of PCAP files and the amount of CPU usage



(c) Different sizes of PCAP files and amount of memory usage

Figure 7.11: Different sizes of PCAP files time required to process, amount of CPU and memory used

Figure 7.11b and Figure 7.11c showed that the tool was not resource intensive as it utilised around 60% of CPU capacity and has a small memory footprint.

7.5 Summary

In this chapter, we analysed the network traffic of 32 different IoT devices in respect to their network settings, to better understand the implications for forensic practitioners. An interesting result from this study was that the majority of devices did not have remote ports open (Section 7.1.3). This contrasts with Loi et al. [90] study where they found that access through Telnet and SSH was prevalent; [256] reported a leaked list dated October-November 2019 (the age of the devices is unknown) containing over half a million credentials that allowed access to IoT devices via telnet.

With the number of IoT devices increasing, investigators will become overwhelmed with devices to analyse, therefore the entropy test and other statistical tests such as a Monte Carlo test will be useful tools in triaging devices that have unencrypted traffic (Section 7.2.1). Regarding devices that sent data in cleartext, we found smart cameras and smart healthcare devices were more prevalent in this aspect. More specifically, smart healthcare devices exposed personal data that could be useful to an investigator in identifying features of a person of interest. We found some devices displayed unusual behaviour when sending cleartext data, such as when the Xiaomi camera detected motion, it would send an unencrypted packet containing not only the captured video but also the credentials to an AWS server (Section 7.3.1).

We examined the encrypted contents of the devices and found a device that exhibited unusual behaviour that could not be controlled by the consumer: the Kasa mobile app for the TP-link devices would take snapshots. Also on the same mobile app, we found the username and password used a weak HTTP based authentication that could be easily decoded. Several other mobile apps accepted proxy connections; however, these were smart plugs and did not contain any sensitive data (Section 7.3.2).

One of the forensic challenges discussed in existing work is the storage of IoT data in multiple locations which then leads to different jurisdictions [16, 251]. In our findings, most data were sent to the US, however, there was also a range of other countries (see Section 7.3.3). Note, this was despite our testbed being based

in the UK and `Dataset-2` having been collected in Australia. This finding was in line with previous work by [269], studied where internet traffic terminated, they also found that data mainly transversed to the US. There would be less legal complexity if the data were stored in data centres within the country of origin. These legal difficulties were highlighted in a case involving Microsoft, where they refused to comply with a US search warrant because the data was stored in the EU [270]. Additionally, we found 77% of the devices' sent data to multiple destinations and 64% of the IoT devices contacted a server belonging to Amazon.

Finally, this chapter presented a tool `IoT Network Analyzer` (Section 7.4), which can automate the analyse of IoT network traffic. The tools' main features, beyond the current state-of-art is that it automatically identifies encrypted network packets by calculating the entropy from the IP payload, extract cleartext in the IP payload and lastly identifies the location of the data using the destination IP address.

Computer Science is no more about computers than astronomy is about telescopes

— EW Dijkstra

8 Conclusion and Future Work

Contents

In this final chapter, we review and conclude the contributions we have made in this thesis and identify the directions for future avenues of research. At a high level, we developed a concise definition of IoT forensics and a research roadmap to guide future research efforts. Next, we proposed an updated taxonomy of digital forensic research-based tools and identified gaps in the IoT forensic tool/solution development. We then developed an approach to identify the type of IoT devices on the network and proposed our machine learning approach that was accurate and efficient. We proposed a data acquisition method using BLE to acquire traces from smart healthcare devices and finally, we developed a prototype tool to aid an investigator in analysing IoT network traffic. Holistically, we improved the whole forensic process at the various stages of the IoT forensic investigation process and developed generic solutions that were compatible with IoT devices from different manufacturers.

8.1 Summary of Contributions

Based on the findings in this thesis, we summarise the contributions made in this thesis:

Definition of IoT Forensics and roadmap for current and future challenges In the first part of this thesis, we carried out a systematic literature review and identified three specific definitions of IoT forensics: these were [19, 56, 57]. These definitions, although different, contained similarities in the terms used. However, none had been developed with input from the digital forensic community. Furthermore, forensic investigators face a significant number of challenges during an investigation, such as a lack of technical capabilities and data volatility. However, it was unclear which of these are real-world problems.

In Chapter 3 we addressed these limitations of existing research and provided three critical contributions. First, we presented the community's interpretation of concepts such as the definitions of IoT forensics, including the type of evidence obtainable from devices, and explored practitioners' investigative experiences and capabilities. As a result, we found participants agreed on some aspects of IoT forensics, such as IoT forensics is a subdomain of IoT forensics but disagreed on which devices they considered IoT devices. We also found that, although participants had experience in examining IoT devices/data, they did not feel prepared to do so. Second, we developed a unified and concise definition for the digital forensic community. Our proposed definition of IoT forensics addressed the limitations of [19, 56, 57] definitions. In our definition, we extracted the keys terms from the existing definitions and from the feedback from the digital forensic community. As a result, this provided the digital forensic community with a clearer understanding of the main digital forensic steps, the type of devices investigated, specified the skills required, and the various subdomains involved in IoT forensics. In our third contribution, we highlighted open research problems and identified and prioritised which of these to overcome first. A key finding from this study was that participants

agreed that research should focus on the development of IoT forensic tools and data acquisition/imaging. This motivated the research for Chapter 6.

Capabilities and shortcomings of research-based tools. Having identified in Chapter 3 that research should focus on IoT forensic tools, it was essential in this chapter to investigate the current landscape of digital forensic tools. These are essential to forensic practitioners to aid them in acquiring and analysing evidence. The main output from digital forensic research has been the development of prototype tools. The existing taxonomy presented by [141, 142] have categorised commercial/open-source digital forensics tools specific to digital forensic domains (e.g., cloud, anti-forensic tools). Similarly, [134] taxonomy was outdated and did not include recently published digital forensic tools. Furthermore, they did not study the diversity, availability, or quality of these published tools. Therefore, the primary aims of Chapter 4 was to propose an extended taxonomy for research-based digital forensic tools and identify the tools that were missing in IoT forensics.

To achieve these aims, in Chapter 4 we conducted a systematic analysis of ~ 800 academic publications from a 6-year period from various digital forensic venues, filtering those presenting tools. We examined each tool found for programming style, code quality, and its documentation. In doing so, this study offered three fundamental contributions. First, we contributed an extended and updated taxonomy of research-based digital forensic tools; we extended it to cover the growing areas of digital forensics, such as device/IoT forensics. Second, we provided a list of 62 recently published research-based tools that were categorised according to our extended taxonomy. Furthermore, we advanced the state-of-art by analysing the availability and quality of the tools and found 33 of these tools were publicly available but not maintained after development. As a third contribution, this study discussed the challenges with the published tools, including the recommendation of a centralised repository specifically for tested tools. In addition, tool researchers (developers) should spend more time on code documentation and evaluation. The findings from this study showed there were gaps in current IoT forensic tools. In

particular, a method to identify IoT devices and network forensic tools to study the various communication channels used by IoT devices. The gaps found in this chapter motivated the research for Chapters 5 and 7.

Identification Approach When we explored the shortcomings in digital forensic research-based tools in Chapter 4, we found a lack of studies focused on the identification stage of the IoT forensic process. Similarly, related research in identification was from the security perspective, such as using machine learning classifiers and unique features extracted from the network traffic proposed by [24, 222]. However, it was unclear how these approaches applied in the forensic context; they were inefficient and required substantial training datasets and features. Therefore, the primary aim of Chapter 5 was to provide solutions to the limitations of [24, 222] and improve its capabilities in the forensic context.

This study offered three significant contributions. First, we proposed an improved machine learning approach that can identify the device-type on the network that combined both packet-headers and behavioural features. Furthermore, we showed our identification approach can identify a multitude of IoT devices from various manufacturers. Second, we validated our approach by using data collected from 22 IoT devices over 2 hours. We showed it outperformed the state-of-the-art by significantly improving the runtime identification approximately 80% using SFS and retained an identification accuracy of 98.2%. Third, we evaluated the scalability of our approach and evaluated the efficiency on three types of hardware The results showed that overall it took the high-end desktop, 2.32 seconds, while the RPi 3 took on average 4.29s, which although twice as long, is acceptable if we were to run our approach on a mobile system.

Acquisition Method In Chapter 2 we highlighted that existing work had focused on developing acquisition frameworks for specific IoT devices, such as the Amazon Alexa ecosystem [27] or they were limited only to a selection of IoT devices [28]. These approaches are not scalable, as they require a tool/plugin to be developed

for each device. In addition, one of the key findings from Chapter 3 suggested that research should focus on developing an acquisition method.

Consequently, the primary focus in Chapter 6 was to develop a generic acquisition method to facilitate the acquisition of traces from a multitude of IoT devices and their connected accompanying mobile device (mobile apps). This study has made the following three substantial contributions. First, we presented a novel method that used BLE to provide a digital forensic investigator with a method to acquire traces from BLE-enabled devices. In the proposed BLE method, we addressed the current limitations of existing acquisition frameworks/solutions, such as the Amazon Alexa ecosystem; our method is not device-specific and applies to a variety of BLE-enabled devices. We demonstrated the feasibility of using this method by acquiring health-related traces and timestamps from 4 BPM. Second, our study provided an insight into the availability of the user data from the mobile apps that accompany smart healthcare devices and a detailed documentation of our findings. We showed that many of the mobile apps left user data on the mobile device. However, the quantity of traces varied depending on the mobile app and not all user data remained on the mobile device after a user logged out. Third, we analysed multiple BLE-enabled devices to determine the confidentiality of the data stored. We showed that most of the smart healthcare devices did not require pairing or use any security mechanisms.

Analysis Tool The limited research attempts in current IoT forensic tool development found in Chapter 4, motivated the work in this chapter. Specifically, we found the requirement for a tool to analyse IoT network traffic. Network traffic from IoT devices is an important source of evidence even when encryption is used. Recent work has shown the potential of analysing IoT network traffic for forensic traces, for instance, [28, 34] intercepted encrypted data to obtain sensitive data such as username and passwords. However, these previous studies are limited in the number of devices they studied and required the manual analysis of IoT network traffic, which is time-consuming. Therefore, the focus of Chapter 7 was to propose

a method on how to conduct a detailed analysis of network traffic from a variety of consumer IoT devices. Combined with a novel network forensic tool to automate the process to aid forensic investigators in analysing IoT traffic.

To address the limitations of previous studies in Chapter 7, we proposed a methodology from the forensic perspective on how to conduct an in-depth, large-scale study of network traffic analysis from IoT consumer devices. We provided insights into the type of traces obtainable from the IoT traffic and the use of metadata for forensic purposes. Overall, we found, most IoT devices did not have a remote port open, limiting remote access for forensic investigators. Smart cameras and smart healthcare devices were more prevalent in sending data in cleartext. This could be useful to an investigator in identifying features of a person of interest. We could easily obtain user credentials from mobile apps that used weak authentication. Finally, most data sent from IoT devices went to the U.S, which could cause legal complexities when the data is stored outside its country of origin.

We proposed a novel network forensic analysis tool, `IoT Network Analyzer`, which aids forensic investigators to automate the analysis of IoT traffic. It addressed the limitations of existing research by automatically analysing IoT traffic from a multitude of IoT devices. The tool has four key features: (1) It is a triage tool that the Shannon Entropy test to easily identify devices using unencrypted traffic. (2) It can easily extract cleartext data from the IP payload, so that credentials can easily be obtained. (3) It can compare ports to determine whether the IoT device used a secure port. (4) Using the destination IP address, it can identify the final location of the data.

To develop `IoT Network Analyzer`, we followed a clear process and detailed the steps taken during tool development using a software development methodology. In addition, we considered the findings from Chapter 4, where documentation and code review were lacking. We addressed these issues when we carried out a code review and fixed the security issues identified. We provided detailed documentation of the tool and included a licence so that the tools' code can be freely redistributed. The tool is available on Github in a public repository, so that other developers can

easily use, maintain, and improve the tool. The tools we identified in Chapter 4, were inconsistent in their evaluation. Therefore to overcome this in our tool, we evaluated the principal features, performance, and efficiency. We compared these results against results from similar tools and found them to be identical and showed our tool was not resource intensive.

8.2 Future Work

Informed by our findings there are several opportunities for future work and these have been divided into three themes: educational and training platforms, usability study on forensic tools and user feedback for tool

Educational program In Chapter 3 we focused on identifying what research challenges should focus on and developing a clearer definition of IoT forensics. The findings from our survey also showed the need for additional educational and training in IoT forensics, especially when participants stated they felt they were not *"fully qualified to do so"* and felt they had *"insufficient training"*.

To address this issue, educational programs/technical training in IoT forensics would need to be developed to prepare the workforce to deal with the exponential increase in the usage of IoT devices, but also to aid in the development of IoT forensic tools. Future work could involve a follow-up survey aimed at digital forensic educators and practitioners to find out what knowledge/skills are missing to develop an IoT forensic curriculum. The results from the survey could then lead to a pilot of the course with higher education students to gather feedback and revise the curriculum.

Datasets are important in IoT forensic education and research. They are used in the classroom and in online courses to increase a forensic practitioners' familiarity with traces extracted from IoT devices and support tool development and testing. However, of the several datasets available to academia, only a few contain IoT devices and these were collected in a lab setting [28, 224]. Therefore, collecting real-world datasets for training and research purposes could be beneficial in furthering

the field of IoT forensics. One way to achieve this, is to explore the development of a tool to automate the remote collection of data from consumer smart home networks. This would produce a more complex and realistic dataset for training.

Usability studies The observations of Chapter 4 could be extended along several directions. We examined the digital forensic research-based tools on the usability, from various perspectives such as the programming style/quality which is also important to the impact of the end-user's experience. The usefulness of the tool can also be robustly measured by conducting user studies with digital forensic practitioners to improve the usability, functionality and identify any shortcomings of the tools such as those provided in [271] could be utilised. We also found the level/complexity of evaluation of digital forensic tools varied, where some tools provided detailed evaluation on accuracy, reliability and used multiple datasets, whereas others evaluated the performance and efficiency. While there are existing tool evaluation metrics and frameworks, these are not typically followed because they are either too complex or do not cover all aspects of the tools. In general, determining a standardised and simplifier framework for testing digital forensic tools is an open problem.

In Chapter 7, we could extend the research in many ways. Our research demonstrated the usefulness in forensic investigations of metadata extracted from the network traffic. Future work could involve investigations from other perspectives, such as investigating the amount of network traffic exchanged when the device is initially powered on. We could then examine the devices' network traffic metadata to determine what information is exchanged during device start-up to and if data is overwritten [16, 19]. We presented our tool `IoT Network Analzyer` that can aid an investigator in analysing IoT traffic. The usefulness of this tool can be robustly measured by conducting a pilot usability study for forensic practitioners to test the application. This will improve the functionality and identify any of the tools' shortcomings.

Improvement to identification method In Chapter 6 we identified IoT devices based on their device-type and developed an identification method using machine learning to extract the packet header and behavioural features. We then used the feature selection method SFS to select the best feature set. However, the limitation of SFS, is that it evaluates all possible combinations of features and therefore takes longer to compute. There are more efficient feature selection methods that could be utilised, such as Least Absolute Shrinkage and Selection Operator (LASSO) [267]. It will be of interest to investigate the effects of these methods on the performance runtime of our proposed identification method. Additionally, before we started our experiments, we updated the IoT devices with the most recent software and no updates were required during our experiments. Therefore, it would be interesting to determine if future software updates will result in changes to the extracted features, as this may change the "fingerprint" of the device.

8.3 Final Remarks

The increasing number and variety of consumer IoT devices provide many benefits to the user, but with constant monitoring of their environment raise many privacy and security concerns. However, it is this continuous monitoring that can prove to be a rich source of evidence for forensic investigators.

Although there are many technical challenges in obtaining evidence from these devices some of which we have sort to address in this thesis. Academic research should not overshadow the educational and training requirements, participants in our survey commented there was "*insufficient training*" or felt they were "*not fully qualified*". Focusing on only developing tools and methods will be problematic not only for investigators but also for future research, as this requires constructive evaluation and feedback from informed practitioners.

In this work, we have demonstrated that there are still many gaps in IoT forensic research, currently, there is no comprehensive framework or any generic tools capable of analysing devices from multiple manufactures. It can be argued that technology (software and hardware) is changing so rapidly that tools are outdated by the time

they are published. This together with the sheer volume of IoT devices, makes it unclear if commercial tools alone can cover the market or if practitioners will require the research community to provide tools.

Appendices

A

IoT Forensic Survey Questions

The following is the complete survey questions:

1. What is your country of residence? Drop down list provided
2. Which age group do you belong to?
   (a) 18-24
   (b) 25-34
   (c) 35-44
   (d) 45-54
   (e) 55-64
   (f) 65+
3. What is your gender?
   (a) Male
   (b) Female
   (c) Other
4. How many years have you worked in cyber forensics?

(a) Less than 1

(b) 1-3 years

(c) 4-6 years

(d) 7-9 years

(e) 10+ years

5. What is your primary occupation in cyber forensics?

(a) Law enforcement practitioner

(b) Non-law enforcement practitioner

(c) Industry instructor

(d) Professor

(e) Student

(f) Researcher

(g) Other

6. Which category does your occupation most fit into?

(a) Federal law enforcement

(b) State/local law enforcement

(c) Military

(d) Legal system

(e) Education/training facility

(f) Private sector organization

(g) Other

7. Do you consider IoT forensics as a sub-domain of cyber forensics?

(a) Yes

(b) No

8. Please rate how much you agree/disagree with the following definitions of what IoT is:

   (a) The Internet of things is the internetworking of physical devices,vehicles,buildings and other items embedded with electronics, software,sensors, actuators,and network connectivity which enable these objects to collect and exchange data.

   (b) Jeffrey Voas from NIST defines IoT using a model he calls the Network of things that relies on 4 key components to function, sensings, computing, communication and actuation. The network of things model relies on sensors, a communication channel, an aggregator, and the cloud in order to function

   (c) The Internet of Things(IoT) is the network of physical objects that contain embedded technology to communicate and sense or interact with their internal states or the external environment.

   (d) The Internet of Things is sensors and actuators embedded in physical objects that are linked through wired and wireless networks

9. What do you consider to be an IoT device? (Check all that apply)

   (a) Smart Home Appliances

   (b) Self Driving Cars

   (c) Smart Phones

   (d) Drones

   (e) Hub based Internet devices (ex. Internet connected light bulbs)

   (f) Internet connected sensors (ex. sensors monitoring power plants and factory machines.)

   (g) Bluetooth peripherals

   (h) Wearables (ex. Smart Watches)

   (i) Home Assistants (Google Home, Amazon Alexa, etc)

10. Does an IoT device need to be constantly connected to the Internet?
    (a) Yes
    (b) No
11. Have you ever been involved with an investigation where you have to analyze IoT data?
    (a) Yes
    (b) No
    (c) Not myself but I know of a colleague who has.
12. In your own words, what is IoT forensics?
    (a) Text submission accepted
13. Would you consider yourself in a position to analyze an IoT device including its communications channels and network traffic?
    (a) Text submission accepted
14. How would you rank the following issues IoT forensics is facing today (1 being the highest)?
    (a) Software tools
    (b) Funding
    (c) Education
    (d) Technical Training
    (e) Legal Issues
    (f) Technological Issues
    (g) Data Security
    (h) Cloud Data Storage
    (i) Research

15. What evidence do you think we can acquire from IoT devices? Why is this evidence important?

    (a) Text submission accepted

16. What are the areas that are most in need of improvements to forensic and software tools used on IoT devices?

    (a) Data preservation

    (b) Data Acquisition[Imaging]

    (c) Cloud forensics

    (d) Data Encryption

    (e) Memory acquisition

    (f) Device disassembly and forensic process

    (g) Other

17. Which is the most challenging area to acquire IoT evidence from?

    (a) Cloud storage

    (b) On-device memory

    (c) On-device storage

    (d) Network

18. Please rate how much you agree or disagree with the following statements:

    (a) Legislation regarding IoT cyber forensics is currently up to date

    (b) Legislation is currently too broad regarding IoT forensics

    (c) Data encryption presents a major problem

    (d) Security on IoT devices is currently weak

    (e) Finding data in the cloud presents a problem for IoT forensics

19. Where should research be focused in order to combat current issues? (select up to 2)

    (a) Legislation updates

    (b) Jurisdictional issues

    (c) Cloud data forensics

    (d) Preserving volatile data

    (e) IoT Forensic Tools

    (f) Breaking Data Encryption on IoT Devices

    (g) Memory Retrieval

20. Please rate how much you agree or disagree that the following present a challenge to the future of IoT forensics:

    (a) Data encryption

    (b) Trail obfuscation(particularly as it

    (c) Pertains to cloud forensics)

    (d) Evidence modeling

    (e) Limitations on local storage

    (f) Legal jurisdiction

    (g) Malware/device security issues

    (h) Lack of digital forensic tools

B

Responses from participants' interpretation of IoT forensics

The tables shown below represents 41 responses from participants' interpretation on IoT forensics. A detailed discussion of these are in Section 3.3.2

Category	Response
Collection, preservation, analysis and presentation of IoT devices and data	P5: Investigation/analysis of real-time and prior crimes committed using IoT; the collection of evidence/data artefacts (footprints) from the IoT sensors/actuators/nodes/devices that would aid law enforcement in resolving cases.
	P11: IoT forensics is the application of the digital forensics process to handling potential evidence that might be contained in an IoT device.
	P16: The identification, preservation, collection, analysis and presentation of data from any 'smart' device.
	P19: IoT forensics the smart forensic methodology for investigate IoT based crimes into more deeper level.
	P22: The necessity of analysing data from objects we wouldn't in the past have expected to contain personal information, for example refrigerators and light bulbs.
	P27: IoT forensics is the practice of collecting, analysing, and reporting on digital data acquired from physical devices that have the ability to connect, communicate and exchange data over wired or wireless networks.
	P31: IoT forensics is a forensic sciences branch which deals with the development of scientifically proven methods to acquire evidence from IoT devices and to analyse this evidence.
	P32: The use of forensically methods and tools to investigate IoT Devices (e.g. Smart Meters) to get digital evidences which are proof for a court of law.
	P33: The analysis of data created by IoT devices in order to get insights for a forensic case
	P34: Examination of IoT devices and IoT data with the goal to find evidence for actions taken or omitted.
	P41: Gathering evidence and presenting the analysis of the evidence in a legally admissible way.
	P42: Investigation process of an incident that involves internet enabled device, which includes the identification of the source of instantiation of connection.
	P45: Digital forensic examination and analysis of IoT devices and data
	P46: The recovery and interpretation of data from IoT systems in order to create understanding of the events which led to the data being deposited.
	P49: The location, collection and analysis of cyber forensic data from components of the IoT, whether purely local or at aggregation and transmission points. It also include analysis of the analyses of IoT data.
	P51: Analysis of sensors and controls with communications. Std Who what when where and how
	P52: IoT forensics is the systematic processing of an IoT devices that is mostly likely a portion of an investigation.

Category	Response
Embedded Devices	P10: Recovery and analysis of data relevant to investigation from any specialized (not general purpose computing) internet connected device
	P17: The retrieval and analysis of artefacts retained on and sent by embedded devices to local or remote instances including the cloud and social media
	P20: The acquisition and processing of data collected from embedded devices that have an internet component that is part of its function.
	P25: The process of extracting (collecting) relevant data from networked physical devices and embedded electronic items for analysis purposes.
	P30: Forensics in IoT devices Like embedded devices which have (in)direct connection over the internet to a central resource for example for Management purposes or control of the devices states.
	P45: Embedded device in electronic log and data investigations
	P56: I would consider IoT forensics, the analysis of data generated by special purpose embedded systems. Smart watches are becoming general purpose, as tablets and phones already are, and there are forensic tools for them. There are relatively few such tools for things like nest thermostat, ring doorbell, video surveillance, industrial control, vehicle control, healthcare diagnostics, electronic voting. The tools tend to be platform specific, and because embedded software platforms may be shared between these, it may be helpful to consider how that affects the practice. Also consider that, from a forensics perspective, we are likely primarily interested in the data from the devices, which means either devices that retain data, or cloud forensics of data uploaded. Do we draw a line between cloud acquisition, and acquisition from devices themselves? Probably not. Look at cell phone forensics, where the tools for imaging phones are taking on features to use credentials from the phone to download data from associated cloud services.
Non-Traditional Computing	P3: Examination of IP connected devices which are not regular computers, PCs, smart phones, tablets etc
	P13: Forensics on small-resource network nodes
	P14: Analysis of non-PC electronics
	P18: IoT forensics is the act of recovering evidence from network enabled devices that are not traditional computers.
	P29: Forensic of the equipment which have internet connection data and which is not classified as computer or mobile forensic
	P59: IoT forensics is the analysis of 'host' and network based artifacts of non-traditional devices that have little computers and the ability to communicate over the internet or within an intra-net.

Category	Response
IoT Network Traffic	P21: Studying the underlying hidden data/communications being sent or stored by IoT type devices. This could include traditional networking appliances as well.
	P35: IoT forensics consists of systems and methods to investigate potential incidents in IoT networks for creating, collecting, analysing and evaluating data as evidence. Compared to computer forensics IoT forensics can be seen as a specialization taking care of highly distributed, loosely coupled, unreliable networks and resource constrained devices.
	P36: Acquisition and analysis of evidence from IoT devices. This can be evidence stored on device itself, or communicated elsewhere, and may include more abstract sources such as network patterns, etc.
	P54: Imaging (capturing) and analysing volatile and non-volatile storage and network traffic related to an IoT device
Cloud	P2: IoT is a branch of digital forensics which deals with IoT related crimes and includes investigation of the connected devices, the sensors as well as the cloud of data.
	P39: Getting information from IoT devices which presented in court can be held as evidence. This can be in the cloud, in the network, in the device (chip-off) etc.
	P53: IoT forensics is the acquisition and analysis of at-rest device and cloud-resident data from a sensor or single-function device. It is also the reverse-engineering and analysis of data in-transit between the IoT device and its ultimate destination.
Others	P9: I think IoT is largely a buzzword for Internet connected devices that are "specific purpose computing" as opposed to "general purpose computing". Forensics is the usual definition of processes/tools to collect/analyse digital evidence.
	P37: Dealing with evidence that relates to the IoT devices themselves (e.g. using them for illegal purposes, evidence of hacking attacks, etc.) and also evidence that has been gathered by IoT devices but relates to crimes that are otherwise unrelated (e.g. home security system knowing that a person was home at a particular time, cameras capturing events, GPS information from wearables, etc.).
	P40: Why IoT forensics! It is all Forensic Research, some components are connected through any kind of protocol is that IoT? Of course IoT differs from other IT, but it remains part of Digital Forensics.
	P62: The biggest difference between IoT forensics and other digital forensics is that IoT forensics is largely reverse engineering. IoT devices are so diverse that one can never be an "IoT expert" and it is really more about the forensic analysis of unknown digital devices.

C

Available Digital Forensic Tools: 2014-2019

Year	Ref	Category	Tool Name	Source	Last Modified*	Programming Language	Licence
2014	[183]	Device incl. IoT forensics	FSNView	Github	5 years	Python	✗
2014	[164]	Software forensics incl. DBs	Mrsh_net	Private/personal website	6 years	C/C++	✗
2014		Software forensics incl. DBs	Sifter	Github	5 years	Python	✗
2014	[155]	Software forensics incl. DBs	OpenLV	Github	5 years	Java	GNU General Public License v2.0
2015	[153]	Software forensics incl. DBs	Advanced forensic Ext4 inode carving (ADEIC)	Private/personal website	1 year	C++	✗
2015	[201]	Memory forensics	Winpmem Pmem	Github	N/A	Python	Apache License 2.0
2015	[180]	Device incl. IoT forensics	ConvertPDML	Dropbox	4 years	Python (2.7)	✗
2015	[161]	Software forensics incl. DBs	Mrsh-cf	Private/personal website	4 years	C/C++	✗
2016	[195]	Network forensics	Kumodd	Github	4 years	Python	GNU General Public License v2.0
2016	[158]	Computer forensics	Advanced Automated Disk Investigation Toolkit (AUDIT)	Source Forge	3 years	Java	No
2016	[198]	Malware forensics	AMExtractor	Github	4 years	C	✗

* This research was undertaken in September 2019

Table C.1: Available digital forensic tools from 2014-2016

Year	Ref	Category	Tool Name	Source	Last Modified*	Programming Language	Licence
2017	[171]	Software forensics incl. DBs	A Forensic Email Analysis Tool Using Dynamic Visualization	Private/personal website	4 years	Python, JavaScript	✗
2017	[133]	Device incl. IoT forensics	DRone Open source Parser (DROP)	Github	3 years	Python	MIT License
2017	[159]	Software forensics incl. DBs	Digital Forensics Framework (DFF) plugin	Github	2 years	Python/C++	✗
2017	[185]	Device incl. IoT forensics	GE-FANUC controller	Private/personal website	7 months	C++	✗
2017	[179]	Multimedia forensics	DECA: a decision-theoretic carving program	Bitbucket	1 year	C++	✗
2017	[205]	Memory forensics	Heapinfo Heapdump Heapsearch Heaprefs	Github	1 year	Python	✗
2017	[152]	Computer forensics	AutoProv	Github	1 year	Python	Apache License 2.0
2018	[199]	Device incl. IoT forensics	AndroParse	Github	1 year	Golang	GNU General Public License v3.0
2018	[175]	Software forensics incl. DBs	MongoDB Deleted Data Recovery Tool	Github	-	Python (3.5)	✗
2018	[84]	Device incl. IoT forensics	Forensic Evidence Acquisition and Analysis System (FEAAS)	Github	-	Python	✗
2018	[187]	Network forensics	Wifimit	Github	-	Python	GNU General Public License v3.0

* This research was undertaken in September 2019

Table C.2: Available digital forensic tools from 2017-2018

Year	Ref	Category	Tool Name	Source	Last Modified*	Programming Language	Licence
2019	[160]	Software forensics incl. DBs	Timeline2GUI	Github	-	Python	✗
2019	[28]	Device incl. IoT forensics	Arlo_autopsy Ismartalarm_autopsy Ismartalarm Wink_db Qbee_autopsy Crypto_dec	Github	-	Python	GNU General Public License v3.0
2019	[154]	Computer forensics	NTFSObjIDParser	Github	-	C++	GNU General Public License v3.0
2019	[207]	Memory forensics	Winesap	Github	-	Python	✗
2019	[176]	Software forensics incl. DBs	Bring2lite	Github	-		CC-BY-NC - Creative Commons license
2019	[209]	Memory forensics	Hooktracer	Github	-		✗
2019		Software forensics incl. DBs	AFF4 evidence container	Github	-	Python	Apache License 2.0
2019	[208]	Memory forensics	Ptenum	Github	-	Python	✗
2019	[186]	Memory forensics	Vivedump	Github	-	Python	✗
2019	[182]	Device incl. IoT forensics	Digital Drone Forensic application	Github	-	Java 8(JDK 1.8), JavaFX 8 libraries	✗
2019	[192]	Malware forensics	BotDAD	Github	-	Python	Apache License 2.0

* This research was undertaken in September 2019

Table C.3: Available digital forensic tools from 2019

D

DFRWS Challenges between 2014-2019

Year	Challenge topic	Developer	Tool name	Programming Language	Last modified
2014	Android Malware	[272]	Manal	N/A	2015
2015	GPU memory	[273]	Gpuhost proc maps		N/A
			Gpuhost dump map		
		[274]	Nullfinder	C++	2015
2016	Software Defined Networking (SDN)	[275]	Booze Allen Hamilton(BAH)	Python	N/A
			2 Volatility patch Files find file.patch		
			Recover filesystem.patch		
2017-2018	IoT	[276]	Google OnHub parser	Python	2018
		[277]	Google OnHub log parser	Python	2018
			JSON parser		
			SQLite database extractor		
			Timeline viewer		
		[278]	OnHub parsing	Python	2018
			SmartThings		
			Kodi		
			Googlehangouts message		
2018-2019	IoT	[279]	IsmartAlarm android		
			Android gmail plugin		
		[280]	Alexa cift parser	Python	-
			IsmartAlarm dairy parser		
			IsmartAlarm base station Server		
			Stream Parser		
			Nest video recovery		
			Wink activity parser		

Table D.1: Tools developed from DFRWS challenges between 2014-2019

E

List of secure ports

Port	Description
500	SofaWare Technologies Remote-HTTPS-Management für eingebettete Firewalls mit Check Point FireWall-1
443	TLS
22	SSH
8883	Secure MQTT
6679	IRC over SSL (Secure Internet Relay Chat)
6697	IRC over SSL (Secure Internet Relay Chat)
9090	Webwasher, Secure Web, McAfee Web Gateway – Default Proxy Port, ManageEngine Applications Manager
9091	Openfire Administration Console (SSL Secured)
19999	DNP – Secure (Distributed Network Protocol – Secure), a secure version of the protocol used in SCADA systems between communicating RTU's and IED's
1884	
8884	
5684	
1311	Dell OpenManage HTTPS
1920	IBM Tivoli Monitoring Console (https)
4712	McAfee Web Gateway 7 – Voreingestellter GUI Port HTTPS
5001	Synology HTTPS WebUI (DSM)
8243	HTTPS for Apache Synapse
8444	PCsync HTTPS
8531	WSUS HTTPS Standardport
14943	Trend Micro ServerProtect for Linux (SPLX) 3.0 web console can be accessed using HTTPS (Hypertext Transfer Protocol over SSL/TLS)
21012	AMLFilter, AMLFilter Inc. amlf-engine-01 HTTPS Standardport
21022	AMLFilter, AMLFilter Inc. amlf-engine-02 HTTPS Standardport

Table E.1: List of secure whitelisted ports

References

[1] Statista. *Forecast end-user spending on IoT solutions worldwide from 2017 to 2025*. 2020. URL: https://www.statista.com/statistics/976313/global-iot-market-size/ (visited on 02/17/2020).

[2] YouGov. *The dawn of the connected home*. Online. Apr. 2018. URL: http://campaign.yougov.com/rs/060-QFD-941/images/YouGov_UK_2018_08_smart_homes.pdf (visited on 03/20/2019).

[3] Tom Hargreaves, Charlie Wilson, and Richard Hauxwell-Baldwin. "Learning to live in a smart home". In: *Building Research & Information* 46.1 (2018), pp. 127–139. eprint: https://doi.org/10.1080/09613218.2017.1286882. URL: https://doi.org/10.1080/09613218.2017.1286882.

[4] Kier. *Kier Living has become the first UK national housebuilder to offer the full suite of Nest products and Google Home Hub to homebuyers across the country.* Online. Sept. 2019. URL: https://www.kierliving.co.uk/news/kier-living-to-offer-nest-and-google-home-smart-technology-to-new-homeowners-288 (visited on 03/20/2019).

[5] Karen Kent et al. "Guide to Integrating Forensic Techniques into Incident Response". In: *NIST Special Publication* (Jan. 2006).

[6] NIST. *Guidelines on Mobile Device Forensics*. en. Online. [Online; Accessed 03-July-2021]. Aug. 2014. URL: https://csrc.nist.gov/Projects/Hash-Functions/NIST-Policy-on-Hash-Functions (visited on 05/03/2019).

[7] C. A. Murphy. "Developing Process for Mobile Device Forensics". In: 2013.

[8] Richard P. Ayers and W. Jansen. "Guidelines on Mobile Device Forensics". In: 2014.

[9] Gary Kessler. "Are mobile device examinations practiced like 'forensics'?" In: *Digital Evidence and Electronic Signature Law Review* 12 (Nov. 2015).

[10] G. M. Jones and S. G. Winster. "Forensics Analysis On Smart Phones Using Mobile Forensics Tools". In: *International Journal of Computational Intelligence Research*. 2017.

[11] Gerald Sauer. *A Murder Case Tests Alexa's Devotion to Your Privacy*. en. Online. 2017. URL: https://www.wired.com/2017/02/murder-case-tests-alexas-devotion-privacy/ (visited on 08/04/2020).

[12] BBC News. *Florida cops hope Alexa can solve bizarre spear murder case*. Online. Nov. 2019. URL: https://www.bbc.co.uk/news/world-us-canada-50269667 (visited on 03/20/2019).

[13] Christine Hauser. *Police Use Fitbit Data to Charge 90-Year-Old Man in Stepdaughter's Killing*. Online. Oct. 2018. URL: https://www.nytimes.com/2018/10/03/us/fitbit-murder-arrest.html (visited on 03/20/2019).

[14] Alfred Ng. *Police request Echo recordings for homicide investigation*. en. Online. 2016. URL: https://www.cnet.com/news/police-request-echo-recordings-for-homicide-investigation/ (visited on 08/04/2020).

[15] Simson L Garfinkel. "Digital forensics research: The next 10 years". In: *Digital Investigation* 7 (Aug. 2010), S64–S73. URL: http://linkinghub.elsevier.com/retrieve/pii/S1742287610000368.

[16] Hegarty Robert, Lamb David J, and Attwood Andrew. "Digital Evidence Challenges in the Internet of Things". In: *INC*. 2014.

[17] Mauro Conti et al. "Internet of Things security and forensics: Challenges and opportunities". In: *Future Generation Computer Systems* 78 (Jan. 2018), pp. 544–546. URL: https://linkinghub.elsevier.com/retrieve/pii/S0167739X17316667.

[18] Lucky Onwuzurike and Emiliano De Cristofaro. "Short: Danger is my middle name - Experimenting with SSL vulnerabilities in android apps". In: *Proceedings of the 8th ACM Conference on Security and Privacy in Wireless and Mobile Networks, WiSec 2015* (2015).

[19] Saad Alabdulsalam et al. "Internet of Things Forensics - Challenges and a Case Study". In: *IFIP Int. Conf. Digital Forensics*. 2018.

[20] George Denton et al. "Leveraging the SRTP protocol for over-the-network memory acquisition of a GE Fanuc Series 90-30". In: *Digital Investigation* 22 (2017), S26–S38. URL: https://dfrws.org/conferences/dfrws-usa-2017/sessions/leveraging-srtp-protocol-over-network-memory-acquisition-ge.

[21] Arunan Sivanathan, Hassan Habibi Gharakheili, and Vijay Sivaraman. "Can We Classify an {IoT} Device Using {TCP} Port Scan?" In: *2018 IEEE International Conference on Information and Automation for Sustainability (ICIAfS) (ICIAfS 2018)* (2018).

[22] Markus Miettinen, Samuel Marchal, and N Asokan. "I O T S ENTINEL : Automated Device-Type Identification for Security Enforcement in IoT". In: *Distributed Computing Systems IOTSENTINEL,IEEE* (2017), pp. 2177–2184.

[23] Nishadh Aluthge Helsinki and Nishadh Aluthge. *IoT device fingerprinting with sequence-based features*. Tech. rep. 2017. URL: https://helda.helsinki.fi/bitstream/handle/10138/234247/Master%20Thesis%20-%20IoT%20device%20fingerprinting.pdf?sequence=2%7B%5C&%7DisAllowed=y.

[24] Arunan Sivanathan et al. "Characterizing and classifying IoT traffic in smart cities and campuses". In: *2017 IEEE Conference on Computer Communications Workshops, INFOCOM WKSHPS 2017*. IEEE, May 2017, pp. 559–564. URL: http://ieeexplore.ieee.org/document/8116438/.

[25] H. Chi, T. Aderibigbe, and B. C. Granville. "A Framework for IoT Data Acquisition and Forensics Analysis". In: *2018 IEEE International Conference on Big Data (Big Data)*. 2018, pp. 5142–5146.

[26] Christopher Meffert et al. “Forensic State Acquisition from Internet of Things (FSAIoT)”. In: *Proceedings of the 12th International Conference on Availability, Reliability and Security - ARES ’17*. 2017, pp. 1–11. URL: https://doi.org/10.1145/3098954.3104053%20http://dl.acm.org/citation.cfm?doid=3098954.3104053.

[27] Hyunji Chung, Jungheum Park, and Sangjin Lee. “Digital forensic approaches for Amazon Alexa ecosystem”. In: *Digital Investigation* 22 (Aug. 2017), S15–S25. URL: http://linkinghub.elsevier.com/retrieve/pii/S1742287617301974.

[28] Francesco Servida and Eoghan Casey. “IoT forensic challenges and opportunities for digital traces”. In: *Digital Investigation* (2019). URL: https://doi.org/10.1016/j.diin.2019.01.012.

[29] Qiaoyang Zhang and Zhiyao Liang. “Security analysis of bluetooth low energy based smart wristbands”. In: Apr. 2017, pp. 421–425.

[30] Alf Omre. “Bluetooth Low Energy: Wireless Connectivity for Medical Monitoring”. In: *Journal of diabetes science and technology* 4 (Mar. 2010), pp. 457–63.

[31] Britt Cyr et al. “Security analysis of wearable fitness devices (fitbit)”. In: *Massachusetts Institute of Technology* 1 (2014).

[32] Mike Ryan. “Bluetooth: With Low Energy Comes Low Security”. In: *7th USENIX Workshop on Offensive Technologies (WOOT 13)*. Washington, D.C.: USENIX Association, Aug. 2013. URL: https://www.usenix.org/conference/woot13/workshop-program/presentation/ryan.

[33] Clinton Ike. “A Survey of Various Methods for Analyzing the Amazon Echo”. In: 2016.

[34] Olumide Kayode and Ali Saman Tosun. “Analysis of IoT Traffic using HTTP Proxy”. In: *IEEE International Conference on Communications* 2019-May (2019), pp. 1–7.

[35] T. Wu and A. Martin. “Bluetooth Low Energy Used for Memory Acquisition from Smart Health Care Devices”. In: *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*. 2018, pp. 1256–1261.

[36] Tina Wu, Frank Breitinger, and Ibrahim Baggili. “IoT Ignorance is Digital Forensics Research Bliss: A Survey to Understand IoT Forensics Definitions, Challenges and Future Research Directions”. In: *Proceedings of the 14th International Conference on Availability, Reliability and Security*. 2019, pp. 1–15.

[37] Tina Wu, Frank Breitinger, and Stephen O’Shaughnessy. “Digital forensic tools: Recent advances and enhancing the status quo”. In: *Forensic Science International: Digital Investigation* 34 (2020), p. 300999. URL: http://www.sciencedirect.com/science/article/pii/S2666281720301864.

[38] Tina Wu, Frank Breitinger, and Stephen Niemann. “IoT network traffic analysis: opportunities and challenges for forensic investigators?” In: 2020.

[39] Kevin Ashton et al. “That ‘internet of things’ thing”. In: *RFID journal* 22.7 (2009), pp. 97–114.

[40] Ismael Pena-Lopez et al. “ITU Internet report 2005: the internet of things”. In: (2005).

[41] Chui, Michael and Loffler, Markus and Roberts, Roger. *The Internet of Things*. Online. Mar. 2010. URL: https://www.mckinsey.com/industries/technology-media-and-telecommunications/our-insights/the-internet-of-things# (visited on 03/20/2019).

[42] Mordor Intelligence. *Smart Homes Market - Growth, Trends, and Forecast (2020 - 2025)*. en. Online. 2019. URL: https://www.mordorintelligence.com/industry-reports/global-smart-homes-market-industry?fbclid=IwAR0Sx_sU1NpcBGgeOtUlJf8ulT1Dils7JKlQFjdcksiMgCi0-Ti8SD0x0vw (visited on 07/27/2020).

[43] James Chen. *Smart Home*. https://www.investopedia.com/terms/s/smart-home.asp. Feb. 2020. URL: https://www.investopedia.com/terms/s/smart-home.asp (visited on 10/17/2019).

[44] Manuel Silverio, Suresh Renukappa, and Subashini Suresh. “What is a smart device? - a conceptualisation within the paradigm of the internet of things”. In: *Visualization in Engineering* 6 (May 2018).

[45] Aleksandar Georgiev and Stephan Schlögl. “Smart Home Technology: An Exploration of End User Perceptions”. In: Feb. 2018.

[46] Hongxu Yin et al. “Smart healthcare”. In: (2018).

[47] Jorge Gomez, Byron Oviedo, and Emilio Zhuma. “Patient Monitoring System Based on Internet of Things”. In: *Procedia Computer Science* 83 (2016). The 7th International Conference on Ambient Systems, Networks and Technologies (ANT 2016) / The 6th International Conference on Sustainable Energy Information Technology (SEIT-2016) / Affiliated Workshops, pp. 90–97. URL: http://www.sciencedirect.com/science/article/pii/S1877050916301260.

[48] Kazi Abu Zilani et al. “R3HMS, An IoT Based Approach for Patient Health Monitoring”. In: *2018 International Conference on Computer, Communication, Chemical, Material and Electronic Engineering (IC4ME2)* (2018), pp. 1–4.

[49] Eoghan Casey et al. “The growing impact of full disk encryption on digital forensics”. In: *Digital Investigation* 8.2 (Nov. 2011), pp. 129–134. URL: https://www.sciencedirect.com/science/article/pii/S1742287611000727.

[50] Angus McKenzie Marshall. *Digital forensics: digital evidence in criminal investigations*. John Wiley & Sons, 2009.

[51] Gary Palmer and Mitre Corporation. *A Road Map for Digital Forensic Research*. Tech. rep. DFRWS, Nov. 2001. URL: http://isis.poly.edu/kulesh/forensics/docs/DFRWS%5C_RM%5C_Final.pdf.

[52] Timothy Grance et al. “Guide to Integrating Forensic Techniques into Incident Response | NIST”. en. In: *Special Publication (NIST SP) - 800-86* (Aug. 2006). URL: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-86.pdf (visited on 05/03/2019).

[53] The Digital Forensic Research Conference DFRWS. *A Road Map for Digital Forensic Research*. Tech. rep. The Digital Forensic Research Conference DFRWS, 2001.

[54] Ankit Agarwal et al. "Systematic Digital Forensic Investigation Model". In: *Gupta International Journal of Computer Science and Security* (Jan. 2011), pp. 2011–118.

[55] Emmanuel S. Pilli, Ramesh C. Joshi, and Rajdeep Niyogi. "A Framework for Network Forensic Analysis". In: *Information and Communication Technologies*. Ed. by Vinu V. Das and R. Vijaykumar. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 142–147.

[56] Shams Zawoad and Ragib Hasan. "FAIoT: Towards Building a Forensics Aware Eco System for the Internet of Things". In: *Proceedings - 2015 IEEE International Conference on Services Computing, SCC 2015* (2015), pp. 279–284.

[57] Edewede Oriwoh and Paul Sant. "The forensics edge management system: A concept and design". In: *Proceedings - IEEE 10th International Conference on Ubiquitous Intelligence and Computing, UIC 2013 and IEEE 10th International Conference on Autonomic and Trusted Computing, ATC 2013*. Dec. 2013, pp. 544–550.

[58] IFIXIT. *Amazon Echo Dot Teardown (1st)*. en. Online. 2016. URL: `https://www.ifixit.com/Teardown/Amazon+Echo+Dot+Teardown/61304` (visited on 07/30/2020).

[59] Maker.IO. *Amazon Echo Dot Teardown (2nd)*. en. Online. 2016. URL: `https://www.digikey.com/en/maker/blogs/2017/amazon-echo-dot-teardown` (visited on 07/30/2020).

[60] Laoise Luciano et al. "Digital Forensics in the Next Five Years". In: *Proceedings of the 13th International Conference on Availability, Reliability and Security - ARES 2018*. New York, New York, USA: ACM Press, 2018, pp. 1–14. URL: `http://dl.acm.org/citation.cfm?doid=3230833.3232813`.

[61] E. Fernandes, J. Jung, and A. Prakash. "Security Analysis of Emerging Smart Home Applications". In: *2016 IEEE Symposium on Security and Privacy (SP)*. 2016, pp. 636–654.

[62] Vaibhavkumar Yadav et al. "Smart home automation using virtue of IoT". In: Apr. 2017, pp. 313–317.

[63] Frank Bentley et al. "Understanding the Long-Term Use of Smart Speaker Assistants". In: *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2.3 (Sept. 2018). URL: `https://doi.org/10.1145/3264901`.

[64] J. Bugeja, D. Jonsson, and A. Jacobsson. "An Investigation of Vulnerabilities in Smart Connected Cameras". In: *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. 2018, pp. 537–542.

[65] Michael Kohn, Mm Eloff, and Jan Eloff. "Integrated digital forensic process model". In: *Computers and Security* 38 (Oct. 2013), pp. 103–115.

[66] Saibharath S and Geethakumari G. "Cloud forensics: Evidence collection and preliminary analysis". In: *2015 IEEE International Advance Computing Conference (IACC)*. 2015, pp. 464–467.

[67] David McClelland and Fabio Marturana. "A Digital Forensics Triage Methodology based on Feature Manipulation Techniques". In: June 2014.

[68] Malek Harbawi and Asaf Varol. "An improved digital evidence acquisition model for the Internet of Things forensic I: A theoretical framework". In: *2017 5th International Symposium on Digital Forensic and Security, ISDFS 2017*. IEEE, Apr. 2017, pp. 1–6. URL: http://ieeexplore.ieee.org/document/7916508/.

[69] Victor R. Kebande and Indrakshi Ray. "A Generic Digital Forensic Investigation Framework for Internet of Things (IoT)". In: *2016 IEEE 4th International Conference on Future Internet of Things and Cloud (FiCloud)* (2016), pp. 356–362. URL: http://ieeexplore.ieee.org/document/7575885/.

[70] Victor Kebande et al. "Towards an Integrated Digital Forensic Investigation Framework for an IoT-Based Ecosystem". In: Aug. 2018.

[71] Francois Bouchaud, Gilles Grimaud, and Thomas Vantroys. "IoT Forensic: identification and classification of evidence in criminal investigations". In: Aug. 2018, pp. 1–9.

[72] Tanveer Zia, Peng Liu, and Wei Han. "Application-Specific Digital Forensics Investigative Model in Internet of Things (IoT)". In: Aug. 2017, pp. 1–7.

[73] A. Nieto, R. Rios, and J. Lopez. "A Methodology for Privacy-Aware IoT-Forensics". In: *2017 IEEE Trustcom/BigDataSE/ICESS*. 2017, pp. 626–633.

[74] Ana Nieto, Ruben Rios, and Javier Lopez. "IoT-Forensics Meets Privacy: Towards Cooperative Digital Investigations". In: *Sensors* 18.2 (Feb. 2018), p. 492. URL: http://www.mdpi.com/1424-8220/18/2/492.

[75] Wooyeon Jo et al. "Digital Forensic Practices and Methodologies for AI Speaker Ecosystems". In: *Digital Investigation* 29 (2019), S80–S93. URL: https://doi.org/10.1016/j.diin.2019.04.013.

[76] Aya Fukami et al. "Improving the reliability of chip-off forensic analysis of NAND flash memory devices". In: *Digital Investigation* 20 (2017). DFRWS 2017 Europe, S1–S11. URL: http://www.sciencedirect.com/science/article/pii/S1742287617300415.

[77] Statista Research Department. *Number of digital voice assistants in use worldwide from 2019 to 2023 (in billions)*. en. Online. Feb. 2020. URL: https://www.statista.com/statistics/973815/worldwide-digital-voice-assistant-in-use/ (visited on 07/27/2020).

[78] Jessica Hyde and Brian Moran. *"Alexa, are you Skynet?"* en. Oct. 2017. URL: http://www.osdfcon.org/presentations/2017/Moran_Hyde-Alexa-are-you-skynet.pdf (visited on 10/17/2017).

[79] Vassil Roussev, Andres Barreto, and Irfan Ahmed. "Forensic Acquisition of Cloud Drives". In: *CoRR* abs/1603.06542 (2016). arXiv: 1603.06542. URL: http://arxiv.org/abs/1603.06542.

[80] Yohannes Yemane Brhan. "API-Based Cloud Data Acquisition and Analysis from Smart Home IoT Environments". In: 2019.

[81] Google Nest. *[Update: new users only] Nest to turn off their API*. Online. May 2019. URL: https://www.home-assistant.io/blog/2019/05/08/nest-data-bye-bye/ (visited on 10/17/2019).

[82] Xiaodong Lin et al. "Automated forensic analysis of mobile applications on Android devices". In: *Digital Investigation* 26 (2018), S59–S66. URL: http://www.sciencedirect.com/science/article/pii/S1742287618301889.

[83] Rajewski Jon. Online. Oct. 20. URL: https://www.jonrajewski.com/data/Presentations/EnFuse2016/Enfuse_2016_Internet_of%20Things_Rajewski.pdf (visited on 10/17/2019).

[84] Gokila Dorai, Ibrahim Baggili, and West Haven. "I Know What You Did Last Summer : Your Smart Home Internet of Things and Your iPhone Forensically Ratting You Out". In: *Proceedings of the 13th International Conference on Availability, Reliability and Security - ARES 2018*. August. 2018, pp. 1–10. URL: http://dl.acm.org/citation.cfm?doid=3230833.3232814.

[85] Akshay Awasthi et al. "Welcome pwn: Almond smart home hub forensics". In: *Digital Investigation* 26 (July 2018), S38–S46. URL: https://linkinghub.elsevier.com/retrieve/pii/S1742287618301907.

[86] Deepak Kumar et al. "All Things Considered: An Analysis of IoT Devices on Home Networks". In: *Proceedings of the 28th USENIX Conference on Security Symposium*. SEC'19. Santa Clara, CA, USA: USENIX Association, 2019, pp. 1169–1185.

[87] Omar Alrawi et al. "SoK: Security Evaluation of Home-Based IoT Deployments". In: *Proceedings - IEEE Symposium on Security and Privacy* 2019-May (2019), pp. 1362–1380.

[88] C. E. Shannon. "Communication theory of secrecy systems". In: *The Bell System Technical Journal* 28.4 (Oct. 1949), pp. 656–715.

[89] Daniel Wood, Noah Apthorpe, and Nick Feamster. "Cleartext data transmissions in consumer IoT medical devices". English (US). In: *IoT S and P 2017 - Proceedings of the 2017 Workshop on Internet of Things Security and Privacy, co-located with CCS 2017*. IoT S and P 2017 - Proceedings of the 2017 Workshop on Internet of Things Security and Privacy, co-located with CCS 2017. Association for Computing Machinery, Inc, Nov. 2017, pp. 7–12.

[90] Franco Loi et al. "Systematically evaluating security and privacy for consumer IoT devices". In: *IoT S and P 2017 - Proceedings of the 2017 Workshop on Internet of Things Security and Privacy, co-located with CCS 2017* (2017), pp. 1–6.

[91] Junia Valente and Alvaro A. Cardenas. "Security & privacy in smart toys". In: *IoT S and P 2017 - Proceedings of the 2017 Workshop on Internet of Things Security and Privacy, co-located with CCS 2017* Ii (2017), pp. 19–24.

[92] Venkata Padmanabhan and Lakshminarayanan Subramanian. "An Investigation of Geographic Mapping Techniques for Internet Hosts". In: *ACM SIGCOMM Computer Communication Review* 31 (Aug. 2001).

[93] Jingjing Ren et al. "Information Exposure From Consumer IoT Devices: A Multidimensional, Network-Informed Measurement Approach". In: *Proceedings of the Internet Measurement Conference*. IMC '19. Amsterdam, Netherlands: Association for Computing Machinery, 2019, pp. 267–279. URL: https://doi.org/10.1145/3355369.3355577.

[94] Noah Apthorpe et al. "IoT Inspector: Crowdsourcing Labeled Network Traffic from Smart Home Devices at Scale". In: (Sept. 2019).

[95] Leonardo Babun et al. "IoTDots: A Digital Forensics Framework for Smart Environments". In: *ArXiv* abs/1809.00745 (2018).

[96] Carmen Camara, Pedro Peris-Lopez, and Juan E. Tapiador. "Security and privacy issues in implantable medical devices: A comprehensive survey". In: *Journal of Biomedical Informatics* 55 (2015), pp. 272–289. URL: http://www.sciencedirect.com/science/article/pii/S153204641500074X.

[97] Eduard Marin et al. "On the (in)Security of the Latest Generation Implantable Cardiac Defibrillators and How to Secure Them". In: *Proceedings of the 32nd Annual Conference on Computer Security Applications*. ACSAC '16. Los Angeles, California, USA: Association for Computing Machinery, 2016, pp. 226–236. URL: https://doi.org/10.1145/2991079.2991094.

[98] Chunxiao Li et al. "Attacking and Defending a Diabetes Therapy System". In: *Security and Privacy for Implantable Medical Devices*. Ed. by Wayne Burleson and Sandro Carrara. New York, NY: Springer New York, 2014, pp. 175–193. URL: https://doi.org/10.1007/978-1-4614-1674-6_8.

[99] Nourhene Ellouze et al. "Cardiac Implantable Medical Devices forensics: Postmortem analysis of lethal attacks scenarios". In: *Digital Investigation* 21 (2017), pp. 11–30. URL: http://www.sciencedirect.com/science/article/pii/S1742287616301426.

[100] George Grispos, William Bradley Glisson, and Tim Storer. "Recovering Residual Forensic Data from Smartphone Interactions with Cloud Storage Providers". In: *CoRR* abs/1506.02268 (2015). arXiv: 1506.02268. URL: http://arxiv.org/abs/1506.02268.

[101] William Glisson et al. "Compromising a Medical Mannequin". In: (Aug. 2015).

[102] K. Lotfy and M. L. Hale. "Assessing Pairing and Data Exchange Mechanism Security in the Wearable Internet of Things". In: *2016 IEEE International Conference on Mobile Services (MS)*. 2016, pp. 25–32.

[103] Ibrahim Baggili et al. "Watch What You Wear: Preliminary Forensic Analysis of Smart Watches". In: Aug. 2015.

[104] Quang Do, Ben Martini, and Kim-Kwang Raymond Choo. "Is the data on your wearable device secure? An Android Wear smartwatch case study: Is the Data on Your Wearable Device Secure?" In: *Software: Practice and Experience* 47 (Jan. 2016).

[105] Serim Kang, Sunyong Kim, and Jongsung Kim. "Forensic analysis for IoT fitness trackers and its application". In: *Peer-to-Peer Networking and Applications* (2020), pp. 1–10.

[106] M. Plachkinova, S. Andres, and S. Chatterjee. "A Taxonomy of mHealth Apps – Security and Privacy Concerns". In: *2015 48th Hawaii International Conference on System Sciences*. 2015, pp. 3187–3196.

[107] Abdullah Azfar, Kim-Kwang Raymond Choo, and Lin Liu. "Forensic Taxonomy of Popular Android mHealth Apps". In: *CoRR* abs/1505.02905 (2015). arXiv: 1505.02905. URL: http://arxiv.org/abs/1505.02905.

[108] Courtney Hassenfeldt et al. "Map My Murder: A Digital Forensic Study of Mobile Health and Fitness Applications". In: Aug. 2019, pp. 1–12.

[109] George Grispos, William Glisson, and Peter Cooper. "A Bleeding Digital Heart: Identifying Residual Data Generation from Smartphone Applications Interacting with Medical Devices". In: Jan. 2019.

[110] Cinthya Grajeda, Frank Breitinger, and Ibrahim Baggili. "Availability of datasets for digital forensics – And what is missing". In: *Digital Investigation* 22 (Aug. 2017), S94–S105. URL: https://www.sciencedirect.com/science/article/pii/S1742287617301913.

[111] Steve Watson and Ali Dehghantanha. "Digital forensics: the missing piece of the Internet of Things promise". In: *Computer Fraud & Security* 2016.6 (June 2016), pp. 5–8. URL: https://www.sciencedirect.com/science/article/pii/S1361372315300452.

[112] Wikipedia. *Internet of things - Wikipedia*. 2018. URL: https://en.wikipedia.org/wiki/Internet_of_things (visited on 10/25/2018).

[113] Jeffrey Voas. "Demystifying the Internet of Things". en. In: *Computer* 49.6 (June 2016), pp. 80–83. URL: http://ieeexplore.ieee.org/document/7490326/ (visited on 09/13/2018).

[114] Gartner. *Internet of Things Defined - Tech Definitions by Gartner*. 2017. URL: https://www.gartner.com/it-glossary/internet-of-things/.

[115] IoT Analystics. *What is Internet of Things Analytics (IoT Analytics)? - Definition from Techopedia*. en. 2018. URL: https://www.techopedia.com/definition/31460/internet-of-things-analytics-iot-analytics (visited on 09/13/2018).

[116] Louis Grenier. *How to analyze open-ended questions in 5 steps [template included]*. Online. June 2021. URL: http://campaign.yougov.com/rs/060-QFD-941/images/YouGov_UK_2018_08_smart_homes.pdf (visited on 03/20/2019).

[117] Jimmy McCloskey. *CSI Alexa: The smart home has become the new crime scene witness*. en. Jan. 2018. URL: https://www.the-ambient.com/features/smart-home-crime-scene-264 (visited on 09/30/2018).

[118] Michelle Taylor. *Murdered Woman's Fitbit Log Used to Charge Husband*. Website. Apr. 2017. URL: https://www.forensicmag.com/news/2017/04/murdered-womans-fitbit-log-used-charge-husband (visited on 05/12/2018).

[119] Md. Mahmud Hossain, Maziar Fotouhi, and Ragib Hasan. "Towards an Analysis of Security Issues, Challenges, and Open Problems in the Internet of Things". In: *2015 IEEE World Congress on Services*. IEEE, June 2015, pp. 21–28. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7196499.

[120] Omer Shwartz et al. "Opening Pandora's Box: Effective Techniques for Reverse Engineering IoT Devices". In: *Smart Card Research and Advanced Applications*. Ed. by Thomas Eisenbarth and Yannick Teglia. Cham: Springer International Publishing, 2018, pp. 1–21.

[121] Zack, Whittaker. *California passes law that bans default passwords in connected devices*. Website. Oct. 2018. (Visited on 03/20/2019).

[122] Department for Digital, Culture Media and Sport. *Secure by Design: Improving the cyber security of consumer Internet of Things Report*. Tech. rep. Department for Digital, Culture Media & Sport, 2017. URL: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/686089/Secure_by_Design_Report_.pdf (visited on 10/24/2018).

[123] James Ami-Narh and Patricia Williams. "Digital Forensics and the Legal System: A Dilemma of our Times." In: *ECU Publications* (Dec. 2008).

[124] Timothy Grance et al. "Guide to Integrating Forensic Techniques into Incident Response | NIST". In: *Special Publication (NIST SP) - 800-86* (Sept. 2006). URL: https://www.nist.gov/publications/guide-integrating-forensic-techniques-incident-response.

[125] Association of Chief Police Officers. *ACPO Good Practice Guide for Digital Evidence*. Tech. rep. Association of Chief Police Officers, 2012.

[126] Patrick Biernacki and Dan Waldorf. "Snowball sampling: Problems and techniques of chain referral sampling". In: *Sociological methods & research* 10.2 (1981), pp. 141–163.

[127] Elena Gorbacheva et al. "Directions for research on gender imbalance in the IT profession". In: *European Journal of Information Systems* 28 (Sept. 2018), pp. 1–25.

[128] D. Peacock and A. Irons. "Gender Inequality in Cybersecurity: Exploring the Gender Gap in Opportunities and Progression". In: *International Journal of Gender, Science, and Technology* 9 (2017), pp. 25–44.

[129] Vikram S. Harichandran et al. "A cyber forensics needs analysis survey: Revisiting the domain's needs a decade later". In: *Computers and Security* 57 (2016), pp. 1–13. URL: http://www.sciencedirect.com/science/article/pii/S0167404815001595.

[130] Ameer Pichan, Mihai Lazarescu, and Sie Teng Soh. "Cloud forensics: Technical challenges, solutions and comparative analysis". In: *Digital Investigation* 13 (2015), pp. 38–57. URL: http://www.sciencedirect.com/science/article/pii/S1742287615000407.

[131] Asanka Sayakkara, Nhien-An Le-Khac, and Mark Scanlon. "Electromagnetic Side-Channel Attacks: Potential for Progressing Hindered Digital Forensic Analysis". In: *Proceedings of the International Workshop on Speculative Side Channel Analysis (WoSSCA 2018)*. Amsterdam, Netherlands: ACM, July 2018.

[132] Keyun, R., Carthy, J., Kechadi, T. "Cloud Forensics An Overview". In: *7th IFIP Conference on Digital Forensics* January (2011), pp. 35–46.

[133] Devon R Clark et al. "DROP (DRone Open source Parser) your drone: Forensic analysis of the DJI Phantom III". In: *Digital Investigation* 22 (2017), S3–S14. URL: https://dfrws.org/conferences/dfrws-usa-2017/sessions/drop-drone-open-source-parser-your-drone-forensic-analysis-dji.

[134] Netherlands Register of Court Experts (NRGD). *Standards 008.0 Digital Forensics*. Tech. rep. https://www.nrgd.nl/binaries/Standards%20Digital%20Forensics_tcm39-82994.pdf. Netherlands Register of Court Experts, Feb. 2016.

[135] Cory Altheide and Harlan A. Carvey. *Digital forensics with open source tools*. en. Burlington, MA: Syngress, 2011.

[136] Dan Manson et al. "Is the open way a better way? Digital forensics using open source tools". In: *2007 40th Annual Hawaii International Conference on System Sciences (HICSS'07)*. IEEE. 2007, 266b–266b.

[137] Brian Carrier. *Open source digital forensics tools: The legal argument*. Tech. rep. stake, 2002.

[138] Daubert vs Merrell. *Daubert v. Merrell Dow Pharmaceuticals, Inc.* 1993. URL: https://supreme.justia.com/cases/federal/us/509/579/ (visited on 09/20/2019).

[139] Ashley Brinson, Abigail Robinson, and Marcus Rogers. "A cyber forensics ontology: Creating a new approach to studying cyber forensics". In: *Digital Investigation* 3 (2006). The Proceedings of the 6th Annual Digital Forensic Research Workshop (DFRWS '06), pp. 37–43. URL: https://www.sciencedirect.com/science/article/pii/S1742287606000703.

[140] Nikita Rana et al. "Taxonomy of digital forensics: Investigation tools and challenges". In: *arXiv preprint arXiv:1709.06529* (2017).

[141] Anand Mishra, Emmanuel Pilli, and Mahesh Govil. "A Taxonomy of Cloud Endpoint Forensic Tools". In: Aug. 2018, pp. 243–261.

[142] Simson Garfinkel. "Anti-forensics: Techniques, detection and countermeasures". In: *2nd International Conference on i-Warfare and Security* (Jan. 2007).

[143] Kevin Conlan, Ibrahim Baggili, and Frank Breitinger. "Anti-forensics: Furthering digital forensic science through a new extended, granular taxonomy". In: *Digital Investigation* 18 (2016), S66–S75. URL: http://www.sciencedirect.com/science/article/pii/S1742287616300378.

[144] Lexico. *Tool | Definition of Tool by Lexico*. en. [Online; Accessed 22-Oct-2019]. Sept. 2019. URL: https://www.lexico.com/en/definition/tool (visited on 09/20/2019).

[145] John Sammons. "Chapter 3 - Labs and tools". In: *The Basics of Digital Forensics (Second Edition)*. Ed. by John Sammons. Second Edition. Boston: Syngress, 2015, pp. 31–46. URL: http://www.sciencedirect.com/science/article/pii/B9780128016350000036.

[146] Q. Ashfaq, R. Khan, and S. Farooq. "A Comparative Analysis of Static Code Analysis Tools that check Java Code Adherence to Java Coding Standards". In: *2019 2nd International Conference on Communication, Computing and Digital systems (C-CODE)*. 2019, pp. 98–103.

[147] NIST. *NIST Policy on Hash Functions*. en. Online. [Online; Accessed 22-Oct-201]. Aug. 2015. URL: https://csrc.nist.gov/Projects/Hash-Functions/NIST-Policy-on-Hash-Functions (visited on 05/03/2019).

[148] Lyczywek, Rita. *How to write a good README for your GitHub project?* en. 2018. URL: %7Bhttps://bulldogjob.com/news/449-how-to-write-a-good-readme-for-your-github-project%7D (visited on 03/20/2019).

[149] Github. *Licensing a repository*. [Online; Accessed 20-Dec-2019]. Oct. 2019. URL: %7Bhttps://help.github.com/en/articles/licensing-a-repository%7D (visited on 03/20/2019).

[150] Brian Carrier. "Defining Digital Forensic Examination and Analysis Tool Using Abstraction Layers." In: *IJDE* 1 (Jan. 2003).

[151] Shariq Murtuza et al. "A TOOL FOR EXTRACTING STATIC AND VOLATILE FORENSIC ARTIFACTS OF WINDOWS 8.x APPS". In: *Advances in Digital Forensics XI*. Ed. by Gilbert Peterson and Sujeet Shenoi. Cham: Springer International Publishing, 2015, pp. 305–320.

[152] Ryan A. Good. "AutoProv: An Automated File Provenance Collection Tool". MA thesis. United States: Air Force Institute of Technology, 2017.

[153] Andreas Dewald and Sabine Seufert. "AFEIC: Advanced forensic Ext4 inode carving". In: *Digital Investigation* 20 (2017). DFRWS 2017 Europe, S83–S91. URL: http://www.sciencedirect.com/science/article/pii/S1742287617300270.

[154] Rune Nordvik, Fergus Toolan, and Stefan Axelsson. "Using the object ID index as an investigative approach for NTFS file systems". In: *Digital Investigation* 28 (2019), S30–S39. URL: https://doi.org/10.1016/j.diin.2019.01.013.

[155] Timothy Vidas, Brian Kaplan, and Matthew Geiger. "OpenLV: Empowering investigators and first-responders in the digital forensics process". In: *Digital Investigation* 11 (May 2014), S45–S53. URL: https://www.sciencedirect.com/science/article/pii/S1742287614000115.

[156] Ibrahim Baggili, Andrew Marrington, and Yasser Jafar. "Performance of a Logical, Five- Phase, Multithreaded, Bootable Triage Tool". In: *Advances in Digital Forensics X*. Ed. by Gilbert Peterson and Sujeet Shenoi. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 279–295.

[157] Christopher Hargreaves and Angus Marshall. "SyncTriage: Using synchronisation artefacts to optimise acquisition order". In: *Digital Investigation* 28 (2019), S134–S140. URL: https://doi.org/10.1016/j.diin.2019.01.022.

[158] Umit Karabiyik and Sudhir Aggarwal. "Advanced Automated Disk Investigation Toolkit". In: *Advances in Digital Forensics XII*. Ed. by Gilbert Peterson and Sujeet Shenoi. Cham: Springer International Publishing, 2016, pp. 379–396.

[159] Christian Zoubek and Konstantin Sack. "Selective deletion of non-relevant data". In: *Digital Investigation* 20 (Mar. 2017), S92–S98. URL: https://www.sciencedirect.com/science/article/pii/S1742287617300300.

[160] Mark Debinski, Frank Breitinger, and Parvathy Mohan. "Timeline2GUI: A Log2Timeline CSV parser and training scenarios". In: *Digital Investigation* 28 (Mar. 2019), pp. 34–43. URL: https://linkinghub.elsevier.com/retrieve/pii/S1742287618303232.

[161] Frank Breitinger, Harald Baier, and Douglas White. "On the database lookup problem of approximate matching". In: *Proceedings of the Digital Forensic Research Conference, DFRWS 2014 EU* 11 (2014), S1–S9. URL: http://dx.doi.org/10.1016/j.diin.2014.03.001.

[162] Vasil Roussev, Richard G. Golden, and Marziale Lodovico. "Multi-resolution similarity hashing". In: *Digital Investigation* 4 (2007), pp. 105–113.

[163] Frank Breitinger. "File Detection On Network Traffic Using Approximate Matching." In: *JDFSL* 9.2 (2014), pp. 23–36. URL: http://ojs.jdfsl.org/index.php/jdfsl/article/view/261.

[164] Vikas Gupta and Frank Breitinger. "How cuckoo filter can improve existing approximate matching techniques". In: *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*. Vol. 157. Jan. 2015, pp. 39–52. URL: http://link.springer.com/10.1007/978-3-319-25512-5%7B%5C_%7D4.

[165] Shams Zawoad, Ragib Hasan, and John Grimes. "LINCS: Towards building a trustworthy litigation hold enabled cloud storage system". In: *Digital Investigation* 14.S1 (2015), S55–S67. URL: http://dx.doi.org/10.1016/j.diin.2015.05.014.

[166] Nicole Lang Beebe and Lishu Liu. "Ranking algorithms for digital forensic string search hits". In: *Digital Investigation* 11.SUPPL. 2 (2014), S124–S132. URL: http://dx.doi.org/10.1016/j.diin.2014.05.007.

[167] Bradley L. Schatz. "AFF4-L: A Scalable Open Logical Evidence Container". In: *Digital Investigation* 29 (2019), S143–S149.

[168] Michael Cohen, Simson Garfinkel, and Bradley Schatz. "Extending the advanced forensic format to accommodate multiple data sources, logical evidence, arbitrary information and forensic workflow". In: *Digital Investigation* 6 (2009), S57–S68.

[169] Frederik Armknecht and Andreas Dewald. "Privacy-preserving email forensics". In: *Digital Investigation* 14.S1 (2015), S127–S136. URL: http://dx.doi.org/10.1016/j.diin.2015.05.003.

[170] Jay Koven et al. "InVEST: Intelligent visual email search and triage". In: *Digital Investigation* 18 (2016), S138–S148. URL: http://dx.doi.org/10.1016/j.diin.2016.04.008.

[171] Johannes Stadlinger and Andreas Dewald. "A Forensic Email Analysis Tool Using Dynamic Visualization". In: *Journal of Digital Forensics, Security and Law* 12.1 (Mar. 2017). URL: http://commons.erau.edu/jdfsl/vol12/iss1/6/.

[172] Abner Mendoza et al. "BrowStEx: A tool to aggregate browser storage artifacts for forensic analysis". In: *Digital Investigation* 14.September 2015 (2015), pp. 63–75. URL: http://dx.doi.org/10.1016/j.diin.2015.08.001.

[173] Jeonghyeon Kim, Aran Park, and Sangjin Lee. "Recovery method of deleted records and tables from ESE database". In: *Digital Investigation* 18 (2016), S118–S124. URL: http://dx.doi.org/10.1016/j.diin.2016.04.003.

[174] James Wagner, Alexander Rasin, and Jonathan Grier. "Database image content explorer: Carving data that does not officially exist". In: *Digital Investigation* 18 (2016), S97–S107. URL: http://dx.doi.org/10.1016/j.diin.2016.04.015.

[175] Jongseong Yoon and Sangjin Lee. "A method and tool to recover data deleted from a MongoDB". In: *Digital Investigation* 24 (Mar. 2018), pp. 106–120. URL: https://www.sciencedirect.com/science/article/pii/S1742287617302347.

[176] Christian Meng and Harald Baier. "bring2lite: A Structural Concept and Tool for Forensic Data Analysis and Recovery of Deleted SQLite Records". In: *Digital Investigation* 29 (2019), S31–S41. URL: https://doi.org/10.1016/j.diin.2019.04.017.

[177] James Wagner et al. "DB3F & DF-Toolkit: The Database Forensic File Format and the Database Forensic Toolkit". In: *Digital Investigation* 29 (2019), S42–S50. URL: https://doi.org/10.1016/j.diin.2019.04.010.

[178] Thomas Gloe, André Fischer, and Matthias Kirchner. "Forensic analysis of video file formats". In: *Digital Investigation* 11.SUPPL. 1 (2014), S68–S76. URL: http://dx.doi.org/10.1016/j.diin.2014.03.009.

[179] Pavel Gladyshev and Joshua I. James. "Decision-theoretic file carving". In: *Digital Investigation* 22 (2017), pp. 46–61. URL: http://dx.doi.org/10.1016/j.diin.2017.08.001.

[180] F. Karpisek, I. Baggili, and F. Breitinger. "WhatsApp network forensics: Decrypting and understanding the WhatsApp call signaling messages". In: *Digital Investigation* 15 (2015), pp. 110–118. URL: http://dx.doi.org/10.1016/j.diin.2015.09.002.

[181] Michael Schmidt. *Pentagon Confronts a New Threat From ISIS: Exploding Drones*. en. Online. Oct. 2016. URL: https://www.nytimes.com/2016/10/12/world/middleeast/iraq-drones-isis.html?_r=0 (visited on 08/01/2020).

[182] Ankit Renduchintala et al. "A comprehensive micro unmanned aerial vehicle (UAV/Drone) forensic framework". In: *Digital Investigation* 30.2019 (2019), pp. 52–72. URL: https://doi.org/10.1016/j.diin.2019.07.002.

[183] Alex J Nelson, Erik Q Steggall, and Darrell D.E. Long. "Cooperative mode: Comparative storage metadata verification applied to the Xbox 360". In: *Digital Investigation* 11.SUPPL. 2 (2014), S46–S56. URL: http://dx.doi.org/10.1016/j.diin.2014.05.004.

[184] Ken Yau. "PLC Forensics Based on Control Program Logic Change Detection." In: *JDFSL* 10.4 (2015), pp. 59–68. URL: http://ojs.jdfsl.org/index.php/jdfsl/article/view/349.

[185] Saranyan Senthivel, Irfan Ahmed, and Vassil Roussev. "SCADA network forensics of the PCCC protocol". In: *Digital Investigation* 22 (2017), S57–S65. URL: https://dfrws.org/conferences/dfrws-usa-2017/sessions/scada-network-forensics-pccc-protocol.

[186] Peter Casey et al. “Inception: Virtual Space in Memory Space in Real Space – Memory Forensics of Immersive Virtual Reality with the HTC Vive”. In: *Digital Investigation* 29 (2019), S13–S21. URL: https://doi.org/10.1016/j.diin.2019.04.007.

[187] Martin Vondráček, Jan Pluskal, and Ondej Rysavy. “Automation of MitM Attack on Wi-Fi Networks”. In: *Digital Forensics and Cyber Crime*. Cham: Springer International Publishing, 2018, pp. 207–220.

[188] David Gugelmann et al. “Hviz: HTTP(S) traffic aggregation and visualization for network forensics”. In: *Digital Investigation* 12.S1 (2015), S1–S11. URL: http://dx.doi.org/10.1016/j.diin.2015.01.005.

[189] Benjamin Taubmann et al. “TLSkex: Harnessing virtual machine introspection for decrypting TLS communication”. In: *Digital Investigation* 16 (2016), S114–S123.

[190] Raymond Hansen, Lourdes Gino, and Dominic Savio. “Covert6: A Tool to Corroborate the Existence of IPv6 Covert Channels”. In: *Annual Conference on Digital Forensics, Security and Law* c (2016). URL: http://commons.erau.edu/adfsl/2016/tuesday/13.

[191] Jason Britt, Alan Sprague, and Gary Warner. “Phishing Intelligence Using the Simple Set Comparison Tool”. In: *Annual Conference on Digital Forensics, Security and Law* c (2015).

[192] Manmeet Singh, Maninder Singh, and Sanmeet Kaur. “Detecting bot-infected machines using DNS fingerprinting”. In: *Digital Investigation* 28 (2019), pp. 14–33.

[193] Max Gannon and Gary Warner. “An Accidental Discovery of IoT Botnets and a Method for Investigating Them With a Custom Lua Dissector BOTNETS AND A : METHOD FOR”. In: *Journal of Digital Forensics, Security and Law* c (2017).

[194] Conor Quinn et al. “Forensic analysis and remote evidence recovery from syncthing: An open source decentralised file synchronisation utility”. In: *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST* 157.OCTOBER (2015), pp. 85–99.

[195] Vassil Roussev, Andres Barreto, and Irfan Ahmed. “API-based forensic acquisition of cloud drives”. In: *IFIP Advances in Information and Communication Technology*. Vol. 484. 2016, pp. 213–235. arXiv: arXiv:1011.1669v3. URL: https://arxiv.org/pdf/1603.06542.pdf%20http://arxiv.org/abs/1603.06542.

[196] Ashkan Rahimian et al. “BinComp: A stratified approach to compiler provenance Attribution”. In: *Digital Investigation* 14.S1 (2015), S146–S155. URL: http://dx.doi.org/10.1016/j.diin.2015.05.015.

[197] Saed Alrabaee, Lingyu Wang, and Mourad Debbabi. “BinGold: Towards robust binary analysis by extracting the semantics of binary code as semantic flow graphs (SFGs)”. In: *Digital Investigation* 18 (2016), S11–S22. URL: http://dx.doi.org/10.1016/j.diin.2016.04.002.

[198] Haiyu Yang et al. “A Tool for Volatile Memory Acquisition from Android Devices”. In: *Advances in Digital Forensics XII*. Ed. by Gilbert Peterson and Sujeet Shenoi. Cham: Springer International Publishing, 2016, pp. 365–378.

[199] Robert Schmicker, Frank Breitinger, and Ibrahim Baggili. "AndroParse - An Android Feature Extraction Framework and Dataset". In: *Digital Forensics and Cyber Crime*. Ed. by Frank Breitinger and Ibrahim Baggili. Cham: Springer International Publishing, 2019, pp. 66–88.

[200] Fengguo Wei et al. "Automated forensic analysis of mobile applications on Android devices". In: *Digital Investigation* 26 (2018), S59–S66. URL: http://dx.doi.org/10.1016/j.diin.2018.04.012.

[201] Johannes Stuttgen, Stefan Vomel, and Michael Denzel. "Acquisition and analysis of compromised firmware using memory forensics". In: *Digital Investigation* 12.S1 (2015), S50–S60.

[202] Arkadiusz Socala and Michael Cohen. "Automatic profile generation for live Linux Memory analysis". In: *Digital Investigation* 16 (2016), S11–S24.

[203] Golden G. Richard and Andrew Case. "In lieu of swap: Analyzing compressed RAM in Mac OS X and Linux". In: *Digital Investigation* 11.SUPPL. 2 (2014), S3–S12. URL: http://dx.doi.org/10.1016/j.diin.2014.05.011.

[204] Vassil Roussev, Irfan Ahmed, and Thomas Sires. "Image-based kernel fingerprinting". In: *Digital Investigation* 11.SUPPL. 2 (2014).

[205] Frank Block and Andreas Dewald. "Linux memory forensics: Dissecting the user space process heap". In: *Digital Investigation* 22 (2017), S66–S75. URL: http://dx.doi.org/10.1016/j.diin.2017.06.002.

[206] Joe T. Sylve, Vico Marziale, and Golden G. Richard. "Pool tag quick scanning for windows memory analysis". In: *Digital Investigation* 16 (2016), S25–S32. URL: http://dx.doi.org/10.1016/j.diin.2016.01.005.

[207] Daniel Uroz and Ricardo J. Rodríguez. "Characteristics and detectability of Windows auto-start extensibility points in memory forensics". In: *Digital Investigation* 28 (2019), S95–S104. URL: https://doi.org/10.1016/j.diin.2019.01.026.

[208] Frank Block and Andreas Dewald. "Windows Memory Forensics: Detecting (Un)Intentionally Hidden Injected Code by Examining Page Table Entries". In: *Digital Investigation* 29 (2019), S3–S12. URL: https://doi.org/10.1016/j.diin.2019.04.008.

[209] Andrew Case et al. "HookTracer: A System for Automated and Accessible API Hooks Analysis". In: *Digital Investigation* 29 (2019), S104–S112. URL: https://doi.org/10.1016/j.diin.2019.04.011.

[210] Tips4PC. *5 Dangers of Free Software*. Aug. 2014. URL: https://techtalk.pcmatic.com/2014/08/26/dangers-free-software/ (visited on 03/20/2018).

[211] Sreenivasa R Vadalasetty. "Security concerns in using open source software for enterprise requirements". In: *SANS Institute* (2003).

[212] Simson Garfinkel et al. "Bringing science to digital forensics with standardized forensic corpora". In: *Digital Investigation* 6 (2009), S2–S11.

[213] Graeme Horsman. "Tool testing and reliability issues in the field of digital forensics". In: *Digital Investigation* 28 (2019), pp. 163–175.

[214] Harlan Carvey. *RegRipper*. 2011. URL: https://www.dfir.training/dfirtools/tools/by-developer/harlan-carvey/25-harlan-carvey-regripper (visited on 03/20/2019).

[215] Simson Garfinkel. *Bulk Extractor*. 2012. URL: %7Bhttps://github.com/simsong/bulk_extractor%7D (visited on 03/20/2019).

[216] Graeme Horsman. "Raiders of the lost artefacts: Championing the need for digital forensics research". In: *Forensic Science International: Reports* Reports 1 (2019) 100003 (2019b).

[217] Brian Carrier and Eugene H. Spafford. "Getting Physical with the Digital Investigation Process". In: *International Journal of Digital Evidence* Volume 2, Issue 2 (2003).

[218] DFIR. *Digital Forensics and Incicent Response Review*. en. Online. 2020. URL: %7Bhttps://dfrws.org/dfir-review/%7D (visited on 03/20/2019).

[219] Sleuthkit-Autopsy. *Sleuthkit Autopsy tookit plugin repository. https://github.com/candicenonsense/nullfinder*. 2019. URL: https://github.com/sleuthkit/autopsy_addon_modules (visited on 09/13/2018).

[220] Markus Miettinen et al. "IoT Sentinel: Automated Device-Type Identification for Security Enforcement in IoT". In: (2016). arXiv: 1611.04880. URL: https://arxiv.org/pdf/1611.04880.pdf%20http://arxiv.org/abs/1611.04880.

[221] Jakob Hasse, Thomas Gloe, and Martin Beck. "Forensic identification of GSM mobile phones". In: June 2013, pp. 131–140.

[222] Bruhadeshwar Bezawada et al. "IoTSense: Behavioral Fingerprinting of IoT Devices". In: (Apr. 2018). arXiv: 1804.03852. URL: http://arxiv.org/abs/1804.03852.

[223] S. Marchal et al. "AuDI: Toward Autonomous IoT Device-Type Identification Using Periodic Communication". In: *IEEE Journal on Selected Areas in Communications* 37.6 (2019), pp. 1402–1412.

[224] Vijay Sivaraman et al. "Classifying IoT Devices in Smart Environments Using Network Traffic Characteristics". In: *IEEE Transactions on Mobile Computing* (2018).

[225] Wes Mckinney. "Data Structures for Statistical Computing in Python". In: *Proceedings of the 9th Python in Science Conference* (Jan. 2010).

[226] Kedar Potdar, Taher S. Pardawala, and Chinmay D. Pai. "A Comparative Study of Categorical Variable Encoding Techniques for Neural Network Classifiers". In: *International Journal of Computer Applications* 175 (2017), pp. 7–9.

[227] Shyamala Doraisamy et al. "A Study on Feature Selection and Classification Techniques for Automatic Genre Classification of Traditional Malay Music." In: Jan. 2008, pp. 331–336.

[228] Antonio Arauzo-Azofra, José Aznarte, and José Benítez. "Empirical study of feature selection methods based on individual feature evaluation for classification problems". In: *Expert Syst. Appl.* 38 (July 2011), pp. 8170–8177.

[229] Sebastian Raschka. “MLxtend: Providing machine learning and data science utilities and extensions to Python’s scientific computing stack”. In: *The Journal of Open Source Software* 3.24 (Apr. 2018). URL: http://joss.theoj.org/papers/10.21105/joss.00638.

[230] A Marcano-Cedeno et al. “Feature selection using sequential forward selection and classification applying artificial metaplasticity neural network”. In: *IECON 2010-36th annual conference on IEEE industrial electronics society*. IEEE. 2010, pp. 2845–2850.

[231] Igor Kononenko. “Estimating attributes: Analysis and extensions of RELIEF”. In: *Machine Learning: ECML-94*. Ed. by Francesco Bergadano and Luc De Raedt. Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 171–182.

[232] Ryan J. Urbanowicz et al. “Relief-based feature selection: Introduction and review”. In: *Journal of Biomedical Informatics* 85 (2018), pp. 189–203. URL: http://www.sciencedirect.com/science/article/pii/S1532046418301400.

[233] Isabelle Guyon et al. “Gene selection for cancer classification using support vector machines”. In: *Machine learning* 46.1-3 (2002), pp. 389–422.

[234] B. Sun, J. Du, and T. Gao. “Study on the Improvement of K-Nearest-Neighbor Algorithm”. In: *2009 International Conference on Artificial Intelligence and Computational Intelligence*. Vol. 4. 2009, pp. 390–393.

[235] Misha Denil, David Matheson, and Nando De Freitas. “Narrowing the Gap: Random Forests In TheDenil, M., Matheson, D., & De Freitas, N. (2014). Narrowing the Gap: Random Forests In Theory and In Practice. Proceedings of The 31st International Conference on Machine Learning, (1998), 665–673. Retrieved from ht”. In: *Proceedings of The 31st International Conference on Machine Learning* 1998 (2014), pp. 665–673. arXiv: arXiv:1310.1415v1. URL: http://jmlr.org/proceedings/papers/v32/denil14.html.

[236] Shujun Huang et al. “Applications of Support Vector Machine (SVM) Learning in Cancer Genomics”. In: *Cancer genomics and proteomics* 15.1 (2018), pp. 41–51. URL: https://europepmc.org/articles/PMC5822181.

[237] Sandro Sperandei. “Understanding logistic regression analysis”. In: *Biochemia medica* 24 (Feb. 2014), pp. 12–8.

[238] Jesse Davis and Mark Goadrich. “The Relationship between Precision-Recall and ROC Curves”. In: *Proceedings of the 23rd International Conference on Machine Learning*. ICML ’06. Pittsburgh, Pennsylvania, USA: Association for Computing Machinery, 2006, pp. 233–240. URL: https://doi.org/10.1145/1143844.1143874.

[239] Marina Sokolova, Nathalie Japkowicz, and Stan Szpakowicz. “Beyond Accuracy, F-Score and ROC: A Family of Discriminant Measures for Performance Evaluation”. In: *AI 2006: Advances in Artificial Intelligence*. Ed. by Abdul Sattar and Byeong-ho Kang. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 1015–1021.

[240] "Antonio J. Pinheiro et al. “Identifying IoT devices and events based on packet length from encrypted traffic”. In: *Computer Communications* 144 (2019), pp. 8–17. URL: http://www.sciencedirect.com/science/article/pii/S0140366419300052.

[241] Marie-Helen Maras and Adam Scott Wandt. “State of Ohio v. Ross Compton: Internet-enabled medical device data introduced as evidence of arson and insurance fraud”. In: *The International Journal of Evidence & Proof* (2020), p. 1365712720930600.

[242] Helmut Strey et al. “Bluetooth low energy technologies for applications in health care: proximity and physiological signals monitors”. In: Oct. 2013, pp. 1–4.

[243] Carles Gomez, Joaquim Oller, and Josep Paradells. “Overview and Evaluation of Bluetooth Low Energy: An Emerging Low-Power Wireless Technology”. In: *Sensors* 12.12 (Aug. 2012), pp. 11734–11753. URL: http://www.mdpi.com/1424-8220/12/9/11734.

[244] K. Townsend, C. Cufi**********, and R. Davidson. *Getting Started with Bluetooth Low Energy: Tools and Techniques for Low-Power Networking*. EBSCOhost ebooks online. O’Reilly Media, 2014. URL: https://books.google.co.uk/books?id=24N7AwAAQBAJ.

[245] Statcounter. *Mobile and Tablet Android Version Market Share Worldwide*. en. Online. May 2020. URL: https://gs.statcounter.com/android-version-market-share/mobile-tablet/worldwide (visited on 05/20/2020).

[246] Hoog Andrew. “"Chapter 6 - Android forensic techniques"”. In: *"Android Forensics"*. Ed. by "Andrew Hoog". "Boston": "Syngress", 2011, "195–284". URL: http://www.sciencedirect.com/science/article/pii/B9781597496513100068.

[247] Lauren Pack. *Arson suspect in unique case featuring pacemaker data is back in custody*. en. Online. 2018. URL: https://www.journal-news.com/news/arson-suspect-unique-case-featuring-pacemaker-data-back-custody/dn6JyzsOemZovpayJMZLNJ/ (visited on 07/18/2020).

[248] Sławomir Jasek. “Gattacking bluetooth smart devices”. In: (2016). URL: http://gattack.io/whitepaper.pdf.

[249] Paul Wagenseil. *75 Percent of Bluetooth Smart Locks Can Be Hacked*. en. Online. Aug. 2016. URL: https://www.tomsguide.com/us/bluetooth-lock-hacks-defcon2016,news-23129.html#xenforo-comments-114962 (visited on 07/18/2020).

[250] Sharon Shasha et al. “Playing with danger: A taxonomy and evaluation of threats to smart toys”. In: *IEEE Internet of Things Journal* 6.2 (2019), pp. 2986–3002. arXiv: 1809.05556.

[251] Ibrar Yaqoob et al. “Internet of things forensics: Recent advances, taxonomy, requirements, and open challenges”. In: *Future Generation Computer Systems* 92 (2019), pp. 265–275. URL: http://www.sciencedirect.com/science/article/pii/S0167739X18315644.

[252] Amazon. *Amazon*. [Online; Accessed 22-Oct-2019]. 2018. URL: %5Curl%7Bhttps://www.amazon.co.uk/product-reviews/B07CB14GTB/ref=acr_dp_hist_1?ie=UTF8&filterByStar=one_star%5C&reviewerType=all_reviews%5C#reviews-filter-bar%7D (visited on 03/20/2019).

[253] Amazon. *Amazon*. [Online; Accessed 22-Oct-2019]. 2018. URL: %5Curl%7Bhttps://www.amazon.co.uk/product-reviews/B07QLR94S1/ref=cm_cr_getr_d_paging_btm_next_4?ie=UTF8&filterByStar=one_star&reviewerType=all_reviews&pageNumber=4#reviews-filter-bar%7D (visited on 03/20/2019).

[254] John Chirillo. *Hack Attacks Revealed: A Complete Reference with Custom Security Hacking Toolkit*. USA: John Wiley & Sons, Inc., 2001.

[255] Speedguide.net. *Port 554 Details*. 2020. URL: https://www.speedguide.net/port.php?port=554 (visited on 03/20/2019).

[256] Catalin Cimpanu. *Hacker leaks passwords for more than 500,000 servers, routers, and IoT devices*. https://www.zdnet.com/article/hacker-leaks-passwords-for-more-than-500000-servers-routers-and-iot-devices/. 2020. (Visited on 03/20/2019).

[257] C. E. Shannon. "Communication theory of secrecy systems". In: *The Bell System Technical Journal* 28.4 (1949), pp. 656–715.

[258] Andrew W. Lo and A.Craig MacKinlay. "The size and power of the variance ratio test in finite samples: A Monte Carlo investigation". In: *Journal of Econometrics* 40.2 (1989), pp. 203–238. URL: http://www.sciencedirect.com/science/article/pii/0304407689900833.

[259] James D. Knoke. "Testing for Randomness Against Autocorrelation: Alternative Tests". In: *Biometrika* 64.3 (1977), pp. 523–529. URL: http://www.jstor.org/stable/2345328.

[260] Fran Casino, Kim Kwang Raymond Choo, and Constantinos Patsakis. "HEDGE: Efficient Traffic Classification of Encrypted and Compressed Packets". In: *IEEE Transactions on Information Forensics and Security* 14.11 (2019), pp. 2916–2926. arXiv: 1905.11873.

[261] Karie Gonia. "Latency and QoS for Voiceover IP". In: *SANS Institute* (2004).

[262] Chun Wang et al. "The next domino to fall: Empirical analysis of user passwords across online services". In: *CODASPY 2018 - Proceedings of the 8th ACM Conference on Data and Application Security and Privacy* 2018-January (2018), pp. 196–203.

[263] WikiLeaks. *Map of Amazon's Data Centers*. https://wikileaks.org/amazon-atlas/map/. 2018. (Visited on 03/20/2019).

[264] Aaron Tilley. *This Smart Doorbell Was Accidentally Sending Data To China, Until People Started Freaking Out*. https://www.forbes.com/sites/aarontilley/2017/03/22/this-smart-doorbell-was-accidentally-sending-data-to-china-until-people-started-freaking-out/#612e957d5984. 2017. (Visited on 03/20/2019).

[265] Anthony Cuthbertson. *Amazon ordered to give Alexa evidence in double murder case*. https://www.independent.co.uk/life-style/gadgets-and-tech/news/amazon-echo-alexa-evidence-murder-case-a8633551.html. 2018. (Visited on 03/19/2019).

[266] W. W. Royce. “Managing the Development of Large Software Systems: Concepts and Techniques”. In: *Proceedings of the 9th International Conference on Software Engineering*. ICSE ’87. Monterey, California, USA: IEEE Computer Society Press, 1987, pp. 328–338.

[267] Krishna K. Agarwal, Achla Agarwal, and Catherine Hiller. “Rapid Applications Development in Python”. In: *J. Comput. Sci. Coll.* 28.4 (Apr. 2013), pp. 42–48.

[268] Manaf Gharaibeh et al. “A look at router geolocation in public and commercial databases”. In: *Proceedings of the ACM SIGCOMM Internet Measurement Conference, IMC* Part F131937.2 (2017), pp. 463–469.

[269] Abdul Rehman Muzammil, Sharon Goldberg, and David Choffnes. “Passport: Enabling Accurate Country-Level Router Geolocation using Inaccurate Sources”. In: (May 2019).

[270] Simon Carswell. *Microsoft warns of risks to Irish operation in US search warrant case*. 2016. URL: %7Bhttps://www.irishtimes.com/business/microsoft-warns-of-risks-to-irish-operation-in-us-search-warrant-case-1.2548718%7D (visited on 03/21/2019).

[271] H. Hibshi, T. Vidas, and L. Cranor. “Usability of Forensics Tools: A User Study”. In: *2011 Sixth International Conference on IT Security Incident Management and IT Forensics*. 2011, pp. 81–91.

[272] Seoul Tech Society. *SeoulTech/Manal*. en. [Online; Accessed 22-Oct-2019]. 2014. URL: https://github.com/SeoulTech/Manal (visited on 07/02/2020).

[273] Villani Antonio and Balzarotti Davide. *GPU Malware Research*. 2015. URL: %7Bhttp://www.cs.uno.edu/~golden/Materials/gpumalware/dfrws-challenge-cquates.zip%7D (visited on 03/20/2019).

[274] Candice Quates. *nullfinder*. 2015. URL: https://github.com/candicenonsense/nullfinder (visited on 09/13/2019).

[275] Bull Joseph et al. *SDN Forensics Challenge, 2016*. 2016. URL: https://www.cmand.org/sdn/sdnf.html (visited on 03/20/2019).

[276] David Kost et al. *dfrws2017-challenge*. https://github.com/dfrws/dfrws2017-challenge/tree/master/challenge-submissions/masaryk.university. 2017. (Visited on 10/17/2019).

[277] Jaehyung Cho et al. *dfrws2017-challenge*. https://github.com/dfrws/dfrws2017-challenge/tree/master/challenge-submissions/kookmin.university/Tools. 2017. (Visited on 10/17/2019).

[278] Gramajo Mark et al. *dfrws2017-challenge*. 2017. URL: %7Bhttps://github.com/dfrws/dfrws2017-challenge/tree/master/challenge-submissions/ssclant_dfrws_2018_challenge_submission%7D (visited on 09/20/2019).

[279] Lewis Kelly Hines and Joshua et al. *dfrws2018-challenge*. 2018. URL: %7Bhttps://github.com/dfrws/dfrws2018-challenge/tree/master/challenge-submissions/spawar-niwclant%7D (visited on 03/20/2019).

[280] Lee Gyuho et al. *dfrws2018-challenge*. 2018. URL: %7Bhttps://github.com/dfrws/dfrws2018-challenge/blob/master/challenge-submissions/tapioca.pearlo/TapiocaPearlo_Tools.zip%7D (visited on 09/20/2019).