Md Sakib Nizam Khan

Enhancing Privacy in IoT Devices
through Automated Handling of Ownership Change

Master’s Thesis
Espoo, June 30, 2017

Supervisor: Professor N. Asokan, Aalto University
Professor Danilo Gligoroski, NTNU
Instructor: Dr. Samuel Marchal (Postdoc), Aalto University
Considering the increasing deployment of IoT devices, their ownership is likely to change during their life cycle. Personal IoT devices used in smart home environment contain privacy sensitive user data. Ownership change of such devices can introduce threats against privacy sensitive data handled by them. To address this problem, we present a system called chownIoT for securely handling ownership change of IoT devices. chownIoT introduces a privacy enhancement protocol that leverages authentication and data encryption for protecting owner privacy. We also present an owner profile management scheme for better management of owners during the life cycle of a device. For automatic detection of ownership change, we use a simple technique which leverage the context of a device. Finally, we present a prototype that implements chownIoT including the privacy enhancement protocol and the owner profile management scheme.
Acknowledgements

It is a remarkable moment for me to submit my Master’s thesis which marks the end of my Master’s journey. First of all, I want to express my sheer gratitude to the Almighty for giving me the strength, knowledge and ability to complete it satisfactorily. I would also like to thank some special people without whom this thesis would not have been possible.

This list starts with my supervisors Professor N. Asokan of Aalto University, Professor, Danilo Gligoroski of NTNU and my advisor at Aalto University Dr. Samuel Marchal. Their support and continuous guidance worked as a true encouragement for me throughout my thesis journey. Their able supervision has guided me through this thesis with great support. I would also like to express my gratitude to Dr. Andrew Paverd of Aalto University for his continuous help, guidance and valuable comments. Without his guidance this thesis would not have been possible.

Finally, I would like to thank some very special persons for whom I have been able to survive this tough journey: my greatest parents, my beloved brother Kashif Nizam Khan, my sister-in-law Jinat Rehana, my family and all my dearest friends for their unwavering affection, love and inspiration.

Espoo, June 30, 2017

Md Sakib Nizam Khan
## Abbreviations and Acronyms

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IoT</td>
<td>Internet of Things</td>
</tr>
<tr>
<td>DDOS</td>
<td>Distributed Denial of Service</td>
</tr>
<tr>
<td>DVR</td>
<td>Digital Video Recorder</td>
</tr>
<tr>
<td>AP</td>
<td>Access Point</td>
</tr>
<tr>
<td>BLE</td>
<td>Bluetooth Low Energy</td>
</tr>
<tr>
<td>NFC</td>
<td>Near Field Communication</td>
</tr>
<tr>
<td>WLAN</td>
<td>Wireless Local Area Network</td>
</tr>
<tr>
<td>ISM</td>
<td>Industrial, Scientific &amp; Medical band</td>
</tr>
<tr>
<td>STA</td>
<td>Stations</td>
</tr>
<tr>
<td>BSS</td>
<td>Basic Service Set</td>
</tr>
<tr>
<td>SSID</td>
<td>Service Set Identifier</td>
</tr>
<tr>
<td>DS</td>
<td>Distribution System</td>
</tr>
<tr>
<td>ESS</td>
<td>Extended Service Set</td>
</tr>
<tr>
<td>IBSS</td>
<td>Independent Basic Service Set</td>
</tr>
<tr>
<td>NIC</td>
<td>Network Interface Card</td>
</tr>
<tr>
<td>SIG</td>
<td>Special Interest Group</td>
</tr>
<tr>
<td>WPAN</td>
<td>Wireless Personal Area Network</td>
</tr>
<tr>
<td>RFID</td>
<td>Radio Frequency Identification Device</td>
</tr>
<tr>
<td>STL</td>
<td>Standard Template Library</td>
</tr>
<tr>
<td>SDK</td>
<td>Software Development Kit</td>
</tr>
<tr>
<td>HMAC</td>
<td>Hashed Message Authentication Code</td>
</tr>
<tr>
<td>MAC</td>
<td>Media Access Control</td>
</tr>
<tr>
<td>AES</td>
<td>Advanced Encryption Standard</td>
</tr>
<tr>
<td>CCM</td>
<td>Counter mode with Cipher block chaining Message authentication code</td>
</tr>
<tr>
<td>UDP</td>
<td>User Datagram Protocol</td>
</tr>
<tr>
<td>AKA</td>
<td>Authentication &amp; Key Agreement</td>
</tr>
<tr>
<td>EAP</td>
<td>Extensible Authentication Protocol</td>
</tr>
<tr>
<td>PBKDF</td>
<td>Password Based Key Derivation Function</td>
</tr>
<tr>
<td>CPU</td>
<td>Central Processing Unit</td>
</tr>
</tbody>
</table>
Contents

Abstract i
Acknowledgements ii
Abbreviations and Acronyms iii
Contents iv

1 Introduction 1
  1.1 Motivation .................................................. 1
  1.2 Contributions ................................................ 2
  1.3 Organization ............................................... 3

2 Background 4
  2.1 What is Internet of Things (IoT) .......................... 4
  2.2 Home Automation: IoT Application Scenario ............... 5
  2.3 Communication Technologies ............................... 7
      2.3.1 Wi-Fi ...................................................... 7
      2.3.2 Bluetooth ............................................... 8
  2.4 Context Awareness ........................................... 9
  2.5 Taxonomy of Smart Home Devices .......................... 9
  2.6 Technical Background ....................................... 11
      2.6.1 Raspberry Pi ........................................... 11
      2.6.2 C++ ..................................................... 12
      2.6.3 Android ............................................... 12

3 Problem Statement 13
  3.1 Threat Model ............................................... 13
  3.2 Ownership Change Scenarios ............................... 15
  3.3 Requirements and Challenges .............................. 17
4 Proposed Solution
4.1 Solution Overview ........................................ 19
4.2 Design and Architecture ................................. 23

5 Implementation ............................................. 27
5.1 Implementation Choices ................................. 27
5.2 Components .............................................. 29
  5.2.1 Context ............................................. 29
  5.2.2 Trusted Device Identity ............................... 30
  5.2.3 Owner Profile ....................................... 30
5.3 Challenge Response Mechanism .......................... 31
5.4 Authentication Mechanism ............................... 33
5.5 Data Encryption Mechanism ............................. 34
5.6 Initial Configuration - Figure 4.3 (1) - (6) .............. 36
5.7 Smart Home Device ..................................... 37
  5.7.1 Ownership Change Detection - Figure 4.4 (1) .... 37
  5.7.2 Owner Authentication - Figure 4.4 (2) - (14) .... 38
5.8 Trusted Device/New Control Device ..................... 39
5.9 Communication Protocol ................................. 42

6 Evaluation ................................................. 46
6.1 Resource Constraints .................................... 46
6.2 Robustness .............................................. 48
6.3 Deployability ........................................... 51
6.4 Usability ................................................ 53

7 Related Work .............................................. 55
  7.1 Context Aware Security ................................ 55
  7.2 Smart Home Device Privacy ............................ 57
  7.3 Access Control of IoT Devices .......................... 58

8 Conclusions ................................................. 61
  8.1 Summary of Contributions .............................. 61
  8.2 Future Work .......................................... 62

Bibliography ................................................ 64
List of Figures

2.1 Smart Home IoT Scenario. (Part of the figure taken from [2]) 5
3.1 Misuse Case Diagram of Smart Home IoT Devices 15
4.1 Flow Diagram of chownloT 20
4.2 Possible Scenarios 21
4.3 Sequence Diagram of chownloT During Initial Configuration 23
4.4 Sequence Diagram of the Proposed Protocol. Due to additional space requirements the unavailability of trusted device case is elaborated in a separate diagram highlighted in green and pointed with an arrow. 25
5.1 Sequence Diagram for Shared Secret Derivation (Diffie-Hellman Key Exchange) and Challenge-Response Mechanism 31
5.2 User Interface (UI) of Control Application 40
5.3 Generic Packet Structure 42
# List of Tables

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>Elements for Each Component</td>
</tr>
<tr>
<td>5.2</td>
<td>Protocol Message Types</td>
</tr>
<tr>
<td>6.1</td>
<td>Encryption CPU Usage</td>
</tr>
<tr>
<td>6.2</td>
<td>CPU Usage for Key Derivation and Hashing</td>
</tr>
<tr>
<td>6.3</td>
<td>Specifications of IoT Boards</td>
</tr>
<tr>
<td>6.4</td>
<td>Memory Usage for Different Operations</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1 Motivation

The Internet of Things (IoT) is considered as the third wave in the information industry after computer and Internet. The paradigm has gained considerable popularity. The concept of IoT refers to inter communication between everyday devices which embody sensing capability and contextual awareness. It is getting integrated in every part of our day to day life and home automation is one such example. Home automation refers to the living environment in which systems and devices can be controlled automatically with the help of technology. IoT devices for home automation are getting widely deployed and used in real world scenarios. Furthermore, home automation systems are getting implemented as built-in feature integrated in newly built houses recently.

IoT devices produce and store sensitive information related to their sensing capabilities and contextual awareness. Similarly, they contain information related to configuration settings, credentials for network and user authentication, etc. all of which are privacy sensitive. The widespread reach of IoT devices, apart from enhancing the user comfort and user experience, also increases the security risk. Security has been one of the major concerns of the IoT paradigm in recent years primarily due to constrained resources. Attacks on IoT devices are uncomplicated and can be performed easily [26]. Several research works have already shown different attacks on IoT devices [46],[16]. Apart from those, one of the major Distributed Denial of Service (DDOS) attack was recently orchestrated using the Mirai botnet which is mostly composed of IoT devices, such as digital cameras and DVR players and it is still active [18]. The problem with the paradigm is that one compromised device can lead to compromising several other connected devices. Hence, IoT de-
CHAPTER 1. INTRODUCTION

Devices are becoming easy targets for attackers. Moreover, home automation devices such as surveillance cameras contain user privacy sensitive data and the compromise of these devices can leak the data.

The enormous growth in deployment of IoT devices in end user level has introduced the possibilities of device ownership change (e.g., selling used devices). Ownership in IoT domain refers to the ability of controlling, managing and accessing a particular device. An example scenario can be an owner who sells his used surveillance camera to someone. Selling such used devices introduces device ownership change. Due to the nature of IoT devices, the ownership change can introduce security threats on the privacy sensitive data handled by such devices. Devices may contain privacy sensitive data or may have access to user’s private cloud storage. Due to the lack of awareness, the ownership change can compromise data or access rights for both previous and new owner. Therefore, handling ownership change in a secure manner becomes necessary.

To address ownership change issue, we propose a new system for automatic ownership change detection and privacy enhancement. The proposed system detects ownership change based on contextual information gathered through sensor data and other means and eventually enhances privacy by handling it in a secure manner. The secure handling of ownership change is done by managing owner profile and introducing authentication. In case of a possible ownership change, the system requests owner to authenticate based on some predefined mechanism. Depending on the authentication result, the device either allows the user to access the sensitive data on the device or protects it by means of encryption. The system also allows a new owner to create a new profile which remains isolated from the existing profiles of the previous owners.

1.2 Contributions

The major contributions of this work are:

- **A threat model for ownership change of smart home devices:**
  - developing the threat model for identifying threats raised due to ownership change of smart home IoT devices (Section 3.1).

- **A protocol for protecting device owner’s privacy sensitive data:**
  - providing a privacy preserving protocol that can prevent privacy
CHAPTER 1. INTRODUCTION

breach due to ownership change of smart home devices (Chapter 4).

• An owner profile management scheme:
   – developing an owner profile management scheme for better handling ownership change and for recovering protected data of smart home devices (Section 5.7.2).

• A simple automatic ownership change detection technique:
   – developing a simple context based automatic ownership change detection mechanism using Wi-Fi (Section 5.7.1).

• A prototype implementation of the system called chownIoT:
   – developing a prototype for realization of the proposed system (Chapter 5).

1.3 Organization

The remainder of the work is organized as follows: Chapter 2 gives an overview of IoT and home automation. Then we provide short descriptions of communication technologies used in IoT. Finally, a technical background is inserted to introduce the reader to the used technologies. Chapter 3 provides a threat model for ownership change of smart home devices. The requirements of a potential solution to protect the threats are also discussed. Chapter 4 presents the proposed solution to protect owner privacy during ownership change and also discusses its design and architecture. Chapter 5 discusses the implementation of the proposed solution. Chapter 6 provides the evaluation of the proposed solution. Chapter 7 discusses the related work. Finally, in Chapter 8, we conclude the work.
Chapter 2

Background

2.1 What is Internet of Things (IoT)

The concept of Internet of Things (IoT) refers to the intercommunication of everyday objects. The term was first coined by Kevin Ashton in the year 1999 as a title of his presentation [9]. Later in 2001, MIT Auto-ID formally came up with a vision on IoT [15]. The paradigm connects different things or objects to the Internet either wired or wirelessly. The goal of IoT is to make every single networking capable objects in the world network connected or more specifically Internet connected. There are already predictions that the number of connected devices by the year 2020 will be 50 billion [19]. The paradigm is primarily driven by the tremendous and continuous progress in networking technologies, wireless sensors and microelectronics in the domain of ubiquitous computing. IoT has enabled instantaneous access to information about the physical world and its objects, which has opened up a new paradigm for innovative services intended for increasing efficiency, productivity and comfort. It has also been one of the most widely researched areas in recent years [32].

IoT has already started impacting several aspects of everyday life and behaviour of its users. The impact is present in both working and domestic environments. Smart home and offices, e-health, smart manufacturing (i.e. Industrial IoT) and e-learning are some of the major applications scenarios of IoT [10]. Among them, smart home is one of the most popular scenarios [6]. Smart home features such as baby monitoring, home surveillance and indoor weather control has made life more comfortable and easier. However, similar to most other technologies, IoT has also some drawbacks. The major concern of IoT is the risk of privacy breach due to the existence of all devices on the Internet. Moreover, most IoT devices are resource constrained, thus,
breaking the security of such devices is easier than for traditional systems. As a result, attackers are currently targeting IoT devices for mounting attacks (e.g. DDOS attacks) and breaking into systems [58],[18].

2.2 Home Automation: IoT Application Scenario

As discussed previously, smart home or home automation is one of the most widely adopted application scenarios of IoT. Figure 2.1 depicts a sample smart home environment equipped with different smart devices. The five
CHAPTER 2. BACKGROUND

major components in the figure are: Access Point (AP), Internet, cloud service, control device and the IoT device itself. The AP works as the entry point to the Internet and the cloud service for the IoT devices. Smartphones are mainly used as control devices for configuring, managing and controlling IoT devices. Apart from smartphones, some device manufacturers also provide support for desktop or laptop computers to be used as control device. The life-cycle of an IoT device begins with configuring the device into the deployment network. Currently, most of the available IoT devices require a vendor provided smartphone application for the initial configuration as well as for later management/control [41, 43, 44].

IoT devices use either Wi-Fi or other short range communication technologies such as, Bluetooth, Bluetooth Low Energy (BLE) or NFC to communicate with smartphone application for initial configuration and mostly Wi-Fi for later management/control. The key goal of the initial configuration is to authenticate the device to the AP and the cloud service. The user provides desired network name and access credential using the smartphone application in order to connect the smart home device with the AP. Similarly for the cloud service, the user either provides the credentials if it already has an existing account with the vendor or creates a new account using the application. Furthermore, the devices are either linked with a cloud account or with a smartphone for access control purpose. The initial configuration requires physical access to the device which improves the security regarding unauthorized access.

Some IoT devices come with a gateway device and different sensing modules. The sensing modules are connected to the Internet via gateway and do not have direct presence in the Internet. The gateway devices use either Wi-Fi or ethernet to communicate with the AP, and other short range technologies such as, Bluetooth and ZigBee to communicate with the sensing devices. For instance, in Figure 2.1 the human body sensor and the wireless switch communicate with the gateway device using ZigBee. The initial configuration is responsible for connecting the gateway to the AP as well as the sensing devices to the gateway.

Most smart home devices contain user privacy sensitive data. For instance, home surveillance cameras contain images and videos of different rooms of the house which are privacy sensitive. In most cases, IoT devices communicate with the cloud service to store and process sensed data. Devices such as, surveillance camera and thermostat store contextual data to the cloud. The smartphone application receives data from the cloud service over Internet and provides real-time update to the user in any remote location at any time. Apart from showing real-time updates, the cloud service and the Internet also enable controlling the device from a remote location.
CHAPTER 2. BACKGROUND

2.3 Communication Technologies

Most smart home IoT devices currently available in the market directly communicate either with AP, control device or gateways. They typically use different technologies such as, Wi-Fi, Bluetooth and ZigBee for these direct communications. The devices mostly use Wi-Fi or Bluetooth for communicating with the control devices for initial configuration. In the case of AP, Wi-Fi is the default communication mechanism. In the case of gateway, IoT devices typically use Bluetooth, Bluetooth Low Energy (BLE) and ZigBee. Section 2.3.1 and 2.3.2 provides a brief overview of Wi-Fi and Bluetooth communication protocols respectively in the context of IoT.

2.3.1 Wi-Fi

Wi-Fi is a Wireless Local Area Network (WLAN) technology based on IEEE 802.11 standards [28]. It enables wireless network communication over a short distance (i.e. 30 meter indoor [17]) using radio signals. Wi-Fi works in the industrial, scientific and medical (ISM) band with frequency range of 2.4 and 5 GHz. The WLAN architecture consists of several components which interact to provide a wireless LAN, supporting station mobility that is transparent to upper layers. The IEEE 802.11 standard defines two basic modes of operation for wireless LAN: ad hoc and infrastructure mode. In ad hoc mode of operation, the participating nodes establish peer-to-peer connection and communicate directly. There is no intermediate component in ad hoc mode. On the other hand, in infrastructure mode of operation there is a centralized entity called Access Point (AP) that all the nodes connect to. All communications happen through the AP. In addition, the AP also serves as a bridge to other networks such as, Internet and wired LAN. In infrastructure mode, a WLAN network composed of an AP and several associated nodes or stations (STAs) is termed as Basic Service Set (BSS). The AP broadcasts a Service Set Identifier (SSID) and STAs connect to it using the SSID. The SSID is the primary name that identifies a particular wireless network. Furthermore, in order to extend the network multiple BSSs can connect to each other using a component called Distribution System (DS). The network of multiple BSSs connected with DS is termed as Extended Service Set (ESS). By contrast, in ad hoc mode of operation such network scaling is not possible, hence the network formed between stations by peer-to-peer connection in this mode is termed as Independent Basic Service Set (IBSS).

IoT devices generally use infrastructure mode of Wi-Fi for initial config-
uration and AP communication. The difference between these two scenarios is that during the initial configuration the IoT device plays the role of an AP and control device connects to it as a station, whereas for the latter it plays the role of a station and connects to an AP. To act as an AP, the IoT device creates a hotspot which is a soft AP. The hotspot mechanism enables the wireless network interface card (NIC) of a device to function as an AP. During the initial configuration the IoT device creates a hotspot. The smartphone connects to the hotspot and provides necessary setup instructions to the device. After receiving the instructions, the IoT device breaks the connection with the smartphone and connects to the desired Internet enabled AP as a station.

2.3.2 Bluetooth

Bluetooth is a short range wireless technology based on radio signals for exchanging data over short distance. It works in the industrial, scientific and medical (ISM) band from 2.4 to 2.485 GHz. It was first invented in 1994 by telecom vendor Ericsson in Sweden [52]. Bluetooth has an open specification developed and maintained by Bluetooth Special Interest Group (SIG) [13]. It is a packet based protocol which uses a master-slave architecture. The specification defines two connectivity topologies which are, piconet and scatternet. A piconet is a wireless personal area network (WPAN) where one Bluetooth device acts as a master and one or more Bluetooth devices act as slaves. A master device can connect up to seven slaves in a piconet. Furthermore, two or more operational piconets together form a scatternet. A single Bluetooth device in a scatternet can simultaneously act as a slave in one piconet and master in another piconet, which enables flow of information outside the range of a single piconet. There can be multiple masters in a scatternet.

IoT devices generally use Bluetooth either for communicating with the control device during initial configuration or for communicating with a gateway device. In IoT environment, the interacting devices typically form a piconet. In case of control device communication, the vendor provided application discovers the IoT device and initiates the connection process. The control device acts as master and the IoT device acts as slave. After a successful connection, the application provides necessary configuration instructions (i.e. Wi-Fi access credential, cloud service credential etc.) to the IoT device over Bluetooth. In addition, in case of gateway device, the initial configuration also instructs the gateway to connect to external modules. The gateway device (i.e. master) pairs with the modules (i.e. slaves) and can establish a connection automatically as they establish a shared secret.
2.4 Context Awareness

Context awareness can be defined as the ability for a system or its components to collect information about its surrounding environment and adjust its behaviour accordingly at any time. Abowd et al. in [5] defined context as any information which can be utilized for characterizing the situation of an entity. They also defined context awareness as the capability of a system to use its context for providing relevant information and/or services to the user. Currently, majority of available devices are equipped with one or more sensors. These sensors provide real-time data about their surrounding environment or context. The devices or applications can use this data to adapt their behaviour according to the environment or context. This is known as context awareness.

With the rapid growth of IoT, the number of devices with sensing capability has also increased. Most IoT devices are equipped with multiple sensors, such as temperature, humidity, pressure, accelerometer, location, etc. The data provided by these sensors are utilized by smart applications to provide context aware services. For example, an application with location sensing capability can detect user’s current location. Whenever a user comes within a 500 meters range of his house with his smartphone and such an application, the application sends a notification to the home automation system to turn on the lights and air conditioning. Similarly whenever a user leaves the house, the application can also send a notification to turn off the lights and air conditioning based on user’s location. The key driving force behind these context aware services are the small and cost effective sensors. The inception of such low cost sensors has opened up enormous possibilities for intuitive and context aware services.

2.5 Taxonomy of Smart Home Devices

In a home automation environment there can be different types of devices in terms of mobility, Internet access capability and availability of sensor modalities. These diversified characteristics of the devices can have direct or indirect impact on the ownership change. Therefore, it becomes necessary to classify
CHAPTER 2. BACKGROUND

the devices based on their different characteristics. The classifications are discussed below:

Based on mobility we can divide the devices into three categories as follows:

- **Static Devices**: In a home automation environment, in most cases the devices are static. The static devices are mounted in a certain place during the installation process and never move after that. For example smart AC, smart fridge, smoke detectors etc.

- **Semi Static Devices**: Some devices in home automation environment are semi static and tend to move rarely. One of the reasons for the mobility of these devices are interior design change. For example, in houses we sometimes change the position of our smart TV or smart microwave due to interior design change. In addition, devices such as baby monitoring cameras can also move initially based on baby’s current location but becomes static once deployed. One of the major characteristics of these devices is that they move within a certain boundary (e.g. within a home or apartment etc.)

- **Mobile Devices**: These kind of devices change positions very frequently. The devices are either cell phones or body worn devices such as wearables which accompanies the user all time (e.g. smartphones, smartwatches, fitness trackers etc.). The major characteristic of these devices is that there is no fixed boundary of movement. They always accompany the owner and can come in and go out from the network very frequently.

Based on Internet access capability the devices can be categorized into two different categories. Those are as follows:

- **Direct Internet Accessible Devices**: In home automation environment, in most cases the devices are capable of connecting to the Internet directly. They periodically update contextual information to a cloud service or can be controlled remotely via Internet. (e.g. surveillance camera, smart AC).

- **Bridged/Indirect Internet Accessible Devices**: There are some IoT devices that are not directly Internet capable and can only communicate to other devices using technology, such as Bluetooth, BLE etc. These devices depend on other devices (e.g. bridged mode) to access or announce their presence on the Internet. Bluetooth beacons or standalone sensors are examples of this kind of devices.
CHAPTER 2. BACKGROUND

Apart from these we can also categorize IoT devices based on the availability of different sensor modalities into two categories. Those are as follows:

- **Multiple Available Sensor Modalities:** Some IoT devices are equipped with multiple sensors, such as temperature, accelerometer, humidity, gyroscope, magnetometer etc. Contextual information gathering highly depends on available sensor modalities. Multiple sensors can provide more information about the context than a single sensor.

- **Single Sensor Modality:** Some IoT devices are equipped with single sensor. Such devices can provide very little information about the deployment context. All IoT devices have at least a communication interface, usually wireless (e.g. Bluetooth, Wi-Fi, ZigBee etc.), that can provide this single sensor modality.

2.6 Technical Background

The purpose of this section is to provide a general overview of the devices and technologies used in implementation. It is primarily focused on the aspects that are strongly related to their usage in this work.

2.6.1 Raspberry Pi

Raspberry Pi is an ARM based single-board computer originally developed by Raspberry Pi foundation for promoting basic computer science education in schools [3]. However, it became more popular for several different purposes including academic research, prototype development and also robotics. There are already several different versions of Raspberry Pi launched and still available. The newest version of mainline Raspberry Pi currently available is version 3. The key advancement in version 3 are on board Wi-Fi and Bluetooth modules which were absent in previous versions.

Initially, it supported only Linux based operating systems. However, currently it can run IoT version of windows 10 operating system. Raspbian is currently the most popular operating system for Raspberry Pi [29]. It is a debian based operating system specially optimized for Pi. The latest version of Raspbian currently available is Raspbian Jessie-April 2017. The operating system has support for most modern day programming languages and other related equipment required for development work. Furthermore, the introduction of on board Wi-Fi and Bluetooth has enabled it to be used as a substitution of most IoT devices specially for research purpose.
CHAPTER 2. BACKGROUND

2.6.2 C++

C++ is an object oriented general purpose programming language designed by Bjarne Stroustrup [56]. It is a superset of C and designed to enhance the capabilities provided by C. It provides both high level features of a modern day programming language and at the same time low level features, such as memory manipulation of embedded programming languages. The key design goals of C++ programming language includes data abstraction and object oriented programming which are not available in C.

C++ is efficient for system level programming and resource constrained embedded devices. In addition, as it is a superset of C, it also supports all the features provided by C. A valid C program is also a valid C++ program. Apart from these, it has a very rich Standard Template Library (STL), which provides common classes, such as containers and associative arrays. The STL makes programming easy and at the same time C++, a powerful language.

2.6.3 Android

Android is a Linux kernel based mobile operating system developed by Google. It is currently the most popular platform for smartphones [30]. One of the main reasons behind its popularity is that it is an open source operating system. In addition, Android provides an application framework for developers.

Android is primarily designed for touchscreen based mobile devices, such as smartphones or tablets. It provides a software development kit (SDK) which includes a comprehensive set of development tools, such as debuggers, emulator, libraries and documentation, eventually making it developer friendly.
Chapter 3

Problem Statement

In Section 1.1, we discussed different privacy sensitive data handled by IoT devices. Ownership change of such devices can introduce threats against these data. The potential attackers in an ownership change scenario of smart home devices can be a malicious previous owner or a malicious new owner. We perform threat modeling to identify potential threats of ownership change of smart home IoT devices. Section 3.1 represents the threat model, Section 3.2 elaborates the potential threats using real world examples and finally, Section 3.3 talks about the requirements and challenges of a potential solution for mitigating the threats.

3.1 Threat Model

In this section, we present the threat model for ownership change of smart home IoT devices by identifying attacker goals, the attack surface and also the capabilities of different attackers. We also provide a misuse case diagram which illustrates all possible threats of smart home IoT devices during ownership change. We begin by discussing attacker goals, attack surface and attacker capabilities.

Attacker Goal: The prime goal or motivation of an attacker in an ownership change scenario is to either retrieve owner’s privacy sensitive data or gain access to owner’s network or cloud service. As discussed previously, the smart home devices contain user privacy sensitive data and also authentication keys of user’s network and cloud service which are the main targets of a potential attacker.

Attack Surface: Smart home devices mostly store data in device storage and on the cloud. Usually, the authentication credentials are stored in the
device storage and the sensed data is stored in cloud storage. Therefore, the major attack targets include the device storage and the cloud storage. Apart from this, the access right to previous owner’s network can also be considered as an attack surface as it can be exploited to gain access to previous owner’s network.

**Attacker Capabilities:** We consider two potential attackers (i.e. malicious new owner and malicious previous owner) in an ownership change scenario who possess different capabilities. The previous owner knows the cloud credential of the sold device which can be used to access the cloud storage and extract data. The cloud storage can be only accessed remotely using the cloud provider interface. In addition, the cloud credentials can also be used to remotely control the device. In contrast, the new owner has full control on the device which can be used to extract both device and cloud data. The full control capability enables the retrieval of privacy sensitive data both by using control device application (i.e. accessing cloud service or device storage) or by directly reading the physical memory location. Apart from these, a new owner can also use the device as an entry point to previous owner’s network by extracting the network authentication key from the sold device.

**Misuse Case Diagram:** Figure 3.1 depicts a misuse case diagram for ownership change of a smart home IoT device. The transparent ovals represent normal use cases that are exposed to threats whereas the black ovals depict the misuse cases which introduce threats. In addition, the arrows connect the normal use cases that are threatened by a particular misuse case. Furthermore, the transparent human shape depicts the normal actor and the black human shape depict the malicious actor. The misuse case diagram provides an overview of the possible threats that are raised due to the ownership change of a smart home IoT device. For instance, as highlighted in the figure, a device can upload data to owner’s private cloud which is a normal use case for a smart home device. However, in an attack scenario, a new owner/attacker with malicious intents can exploit this access right to retrieve previous owner’s privacy sensitive data which is a misuse case. Similarly, the figure provides other possible misuse cases which threaten normal use cases and also associates their normal and malicious actors.

As per the capabilities discussed previously, the previous owner can access the cloud storage and modify/retrieve new owner’s data using the previous credentials. In addition, the cloud credentials can also be used to control the device remotely. Thus, the previous owner can have unauthorized access to the device using the previous credentials.

On the other hand, according to new owner capabilities, he/she can retrieve previous owner’s data and extract authentication keys by accessing the
device storage. Similarly, the new owner can also access and modify/retrieve previous owner’s cloud data by misusing the access rights of the device to the previous owner’s account. Apart from this, spoofing and stealing device can also raise threats against the normal use cases.

Each of these possible misuse cases are illustrated in the misuse case diagram. The next section elaborates the misuse case diagram by providing different ownership change scenarios, their corresponding threats and real world examples.

### 3.2 Ownership Change Scenarios

Used smart home devices can be sold/lent or can even get stolen. Such scenarios introduce change of ownership, which, in turn, introduce threats
against privacy sensitive information handled by the smart home devices. The possible ownership change scenarios and their corresponding threats are discussed below.

**Scenario 1:** Selling a smart home device.

- **Description:** Selling a used device introduces change of ownership between the seller (i.e. previous owner) and the buyer (i.e. new owner). This change of ownership can introduce threats for both previous and new owner if handled without proper security measures.

- **Threats - Previous Owner’s Perspective - TP1:** A new owner with malicious intents can use the bought device to connect to previous owner’s network and perform malicious activities.

  *Example:* An owner sells his Smart Microwave to his neighbor next door. The Microwave is always in range of previous owner’s wireless network. As the device gets automatically connected to the known wireless network if in range, hence a malicious neighbor/new owner can use the Microwave as an entry point to the previous owner’s network. In addition, the new owner can extract authentication keys from the device to access personal data or cloud storage of the previous owner.

- **Threats - New Owner’s Perspective - TN1:** A previous owner with malicious intents can still access the sold device information using previous credentials and perform malicious activities.

  *Example:* A person sells his smartwatch, which is connected to a cloud service. The watch periodically updates the cloud service with user related information. The new owner only uses the smartwatch to connect and transfer information with a smartphone and is not concerned about the cloud service or it is possible to register several account/users for the same smartwatch. In such scenario, the sold watch still updates new owner’s information to the cloud service linked with previous owner’s account or previous owner still has access to the service. Hence, the previous owner can misuse the new owner’s privacy sensitive information.

**Scenario 2:** Selling a house or change of tenant in apartments with integrated smart home devices.

- **Description:** Home automation devices are implemented as a built-in requirement in new houses. Hence, selling houses equipped with home automation devices or change of tenant are common scenarios.
CHAPTER 3. PROBLEM STATEMENT

- Threats - Previous Owner’s/Tenant’s Perspective - TP2: As most home automation devices contain privacy sensitive data thus, if the ownership is not transferred with proper security measures e.g. Reset device to factory defaults, wipe the data history etc. then the new owner can retrieve and misuse previous owner’s personal data.

  Example: A home surveillance camera may contain privacy sensitive image or video data. In some cases the data is periodically uploaded to a cloud data storage. Due to ownership change, the new owner/tenant can use the surveillance camera to access the cloud storage of the previous owner/tenant and misuse the data.

- Threats - New Owner’s/Tenant’s Perspective - TN2: Opposite to previous case, if the new owner/tenant is not aware of the cloud storage feature of the surveillance camera then the previous owner/tenant can access the cloud storage with previous credentials and misuse the new owner’s/tenant’s privacy sensitive data without his/her knowledge.

**Scenario 3:** A device gets stolen.

- Description: Some smart home IoT devices can be small in their form factor (e.g. smartwatch, fitness tracker etc.) or can be deployed in a place which is easily accessible (e.g. in a garden). Hence, getting stolen is a possible scenario.

- Threats: As discussed previously in Scenario 2 that smart home devices can contain privacy sensitive information and stolen devices can leak that information.

**Scenario 4:** Lending a smart home device.

- Description: Some smart home IoT devices are wearables or body worn devices and some may have small form factors and can be lent for some days use. This kind of scenario causes temporary ownership change.

- Threats: Wearable devices such as smartwatch or fitness trackers can also contain privacy sensitive user data. Hence, the threats TP2 and TN2 discussed in Scenario 2 are also applicable to this scenario.

### 3.3 Requirements and Challenges

The primary goal of this project is to develop a solution to protect the privacy sensitive information handled by smart home devices in case of an ownership change. The intended solution should be able to automatically detect
ownership change and enhance privacy by facilitating the ownership change procedure in a secure manner. Smart home devices have some special characteristics, such as limited resources, different sensor modalities and vendor specific operating systems and application dependencies, which isolate them from traditional devices. Thus, in order to address these special characteristics the intended solution must fulfill certain requirements:

1. Resource Constraints - Ability to work on resource constrained devices.  
   Challenges: Most smart home devices are built with limited resources. However, security solutions sometimes require a large amount of resources to be completely secure. The intended solution should have the least overhead to be able to efficiently work on devices with limited resources. For practical evaluation, it should be implementable for instance on typical IoT boards available in the market.

2. Robustness - Secured against all possible threats and in all ownership change scenarios.  
   Challenges: Section 3.1 and 3.2 identified several possible threats during ownership change. The intended solution should provide robust security against all these identified threats and scenarios.

3. Deployability - Adaptable to the largest possible class of devices.  
   Challenges: There are different types of IoT devices available in the market from different vendors. These devices have different operating principles (e.g. different OS, different protocols etc.) as well as different sensor modalities. Developing different solution for each class of devices is very challenging. Therefore, the intended solution should be generic. In addition, it should be adaptable and efficient on all class of devices previously described in Section 2.5.

4. Usability - Intuitive and easy to use.  
   Challenges: Enhancing privacy/security can lead to a complex solution which eventually reduces its usability. Thus, the intended solution should not degrade user experience and at the same time should be secure. It should also have least overhead.
Chapter 4

Proposed Solution

4.1 Solution Overview

The previous chapter discussed the threats that can arise due to ownership change of smart home IoT devices. In addition, we also identified the requirements and challenges of a potential solution to securely handle ownership change. Now in this chapter, we propose a new system called chownIoT to address the identified challenges and fulfill the requirements. chownIoT enhances user privacy in case of ownership change of smart home IoT devices by handling it in a secure manner. There are mainly two parts of a complete solution: first, the automatic detection of ownership change and second, enhancing privacy by handling it securely. chownIoT primarily focuses on privacy enhancement of smart home IoT devices during an ownership change. It defines a protocol for securely handling ownership change. The system assumes that there are already techniques to automatically detect an ownership change. Nevertheless, we also propose a simple technique for automatic ownership change detection.

chownIoT assumes that based on some contextual information smart home IoT devices can infer an ownership change. The system infers that a change in context of smart home device means a possible change of ownership. The smart home device stores the information about the context in which it starts operating during the initial configuration. chownIoT terms it as known context. Thus, if current contextual information does not match with a stored known context then the smart home device infers a possible change of ownership. As discussed in Section 2.2, most smart home devices require a smartphone during initial configuration to start operating. According to ”resurrecting duckling security policy model” a device should recognize as its owner the first entity with which it participates in a ”take ownership”
ceremony [55]. Based on this theory, chownIoT considers the first device which is used to configure a smart home device as the owner device. The
system stores the identity of the smartphone during the initial configuration and terms it as the trusted device. In addition, it establishes a security association between the trusted device and the smart home device for future verification of the trusted device. chownIoT also establishes an authentication mechanism according to the user/owner preference and available resources for future interactions in case of unavailability of the trusted device during a possible ownership change.

A simple flow diagram of chownIoT after initial configuration is shown in Figure 4.1. After the initial configuration, the smart home device starts detecting a possible ownership change. Once the device infers a possible ownership change (i.e. change of context), at first it searches for the trusted device in its vicinity. If the smart home device is able to find the trusted device nearby, it sends a challenge. Upon receiving the challenge, the trusted device sends a response back to the smart home device without any user intervention. The response is generated by computing HMAC [11] of the challenge using the key established during security association. The details of response calculation is provided in Section 5.3. It is worth noting that the challenge response mechanism without any user intervention can expose the system to relay attacks [22]. However, involving users can reduce the usability of the system. Therefore, in this scenario, as a design choice we preferred usability over security. The potential for relay attack is further discussed in Section 6.2.

A valid response for the challenge creates a new known context for the current owner. This situation is depicted in Scenario 1 of Figure 4.2. We assume that the trusted device is always able to successfully authenticate if it is present. However, if the trusted device is unable to authenticate or if it is unavailable, the smart home device protects the data of the current

Figure 4.2: Possible Scenarios
CHAPTER 4. PROPOSED SOLUTION

owner profile. The term *owner profile* here refers to all necessary information related to a particular owner including his authentication information, secret keys, trusted device identity, cloud credentials, contextual information (i.e. known contexts) and user data. There can be multiple owner profiles in chownIoT. However, during device operation only one owner profile can be active and all others are protected.

Next, the smart home device looks for another device willing to act as a *new control device*. Upon identifying a new control device, the smart home device sends the list of existing profile to it and waits for the user to choose one or create a new owner profile. If the user selects an existing profile, the smart home device authenticates the selected profile according to the mechanism established by the owner during initial configuration. This situation is depicted in Scenario 2 of Figure 4.2. The differentiation between the trusted device and new control device improves both the robustness and the usability of the system. If the owner has the trusted device nearby during a possible ownership change of a particular IoT device, the verification of ownership change happens seamlessly. Thus, if all control devices act as trusted devices then it will degrade the robustness of the system. For instance, the owner may want to allow guest access on the smart home device with a control device, however it does not want the guest to move the device to another location (i.e. new context). In such cases, the differentiation between trusted device and new control device allows the owner to have more fine-grained access levels for the smart home device. Nevertheless, the ability to securely access a particular IoT device with any control device also improves the usability of the system.

If the authentication is successful then the smart home device unlocks the profile data and similar to trusted device scenario it also creates a new known context for the current profile. The owner can also add the new control device as a trusted device. Next, if the user wants to create a new profile, the smart home device responds to the request. For new profile creation, chownIoT establishes a new security association, new owner authentication mechanism and also a new known context list. In addition, it also stores the new control device identity as the trusted device for the new owner. The new owner profile is completely isolated from the previous owner’s profile. It is worth noting that the proposed system assumes all communication between the smart home device and trusted/new control device are protected using encryption at the physical layer.
4.2 Design and Architecture

The previous section provided an overview of chownIoT. Now in this section we present our privacy enhancement protocol and our design choices in detail.

The proposed privacy enhancement protocol builds the security mechanism by trusting the smartphone used during initial configuration of a smart home device. During the initial configuration the protocol establishes a security association as well as an owner authentication mechanism for future verification of the trusted device or of the owner in case of ownership change. The interactions between the smart home device and the trusted device during initial configuration are depicted in Figure 4.3. As shown in the figure, the trusted device starts the process by performing the device specific configurations of the target smart home device (1). This process is common for most of the available smart home devices, such as establishing Wi-Fi network connection by providing SSID and access credential of the desired AP and setting up desired device parameters.

![Sequence Diagram of chownIoT During Initial Configuration](image)

Figure 4.3: Sequence Diagram of chownIoT During Initial Configuration

After the configuration process, the privacy enhancement protocol initiates a security association between the trusted device and smart home device (2). The security association can be establishment of a shared secret key or it can also be sharing of public key. The process eventually establishes a
shared security attribute between the trusted device and the smart home device which is used to authenticate the trusted device during a possible ownership change scenario. Next, the protocol stores the identity of the trusted device (3). The identity of the trusted device is dependent on the technology used during initial configuration. For instance, in case of Bluetooth, the identity can be the Bluetooth device name and address pair, in case of Wi-Fi, it can be the Wi-Fi Media Access Control (MAC) address. In addition, the protocol also stores the shared security attribute established in (2) as an additional element of trusted device identity.

Next, the protocol initiates a setup owner authentication process (4). This process establishes an authentication mechanism and the required credentials between the smart home device and the owner. In addition, the user also selects a profile name using which the smart home device sets up the initial owner profile. The authentication mechanism for instance, can be password based authentication where the owner selects a password of his/her own choice. Therefore, when the proof of authenticity of the owner is required and the trusted device is not available, he/she is prompted to enter the defined password on a new control device. Both the shared security attribute and the authentication mechanism are linked with the owner profile. The only difference is that the shared security attribute is used to authenticate the owner on the trusted device without requiring any user intervention whereas the authentication mechanism is used as fallback to authenticate the owner on a new control device which require active user involvement. Finally, the protocol stores the essential owner authentication information (5) as well as the current contextual information as known context for the current profile (6).

After the initial configuration the smart home device starts operating normally. In addition, it also starts detecting possible ownership change based on contextual information. Figure 4.4 depicts the sequence diagram of the proposed privacy enhancement protocol in case of a possible ownership change. The system continuously loops to detect a possible ownership change or change to an unknown context (1) as depicted in the sequence diagram. If the smart home device detects a possible ownership change, at first it searches for the trusted device (2) whose identity is stored during the initial configuration. Next, if the smart home device is able to find the trusted device in its vicinity, it sends a challenge (3) to the device. The trusted device replies with a response (3) satisfying the challenge. Upon receiving a response, the smart home device verifies it. If the challenge is satisfied, it stores the current context as known context for the current profile (4).

If the trusted device is unable to satisfy the challenge (3) or if it is unavailable in (2) then the smart home device encrypts the profile data (5),
Figure 4.4: Sequence Diagram of the Proposed Protocol. Due to additional space requirements the unavailability of trusted device case is elaborated in a separate diagram highlighted in green and pointed with an arrow.
CHAPTER 4. PROPOSED SOLUTION

(6). Next, the smart home device starts looking for a new control device (7). Upon identifying a new control device, the smart home device sends a list of existing profiles to it and waits for the user choice response (8). At this point, the user can either choose an existing profile or can create a new profile. If the user selects an existing profile, the new control device prompts for authentication (9) according to the mechanism established during the initial configuration of the selected profile. If the authentication process is successful then the smart home device decrypts the data of the associated profile (10) and stores the current context as known context (11) for the selected profile, similarly to the case of trusted device.

By contrast, if the authentication process is unsuccessful (12) or the user chooses to create a new profile in (9) then the protocol creates a fresh profile. To create a new profile, the protocol configures the new control device (13) by establishing a security association as well as an owner authentication mechanism for the future verification of the new owner, similar to steps (2) to (6) of initial configuration process depicted in Figure 4.3. With the new profile, the new owner has full control on the device.

As we can see from the protocol description, it requires to define a challenge response mechanism (3), an owner authentication mechanism (9) and also a data encryption mechanism (5), (6), (7). In addition, for realizing the system, we need to also define a technique for automatic ownership change detection (1), even though it is not the main focus of chownIoT. There can be several different approaches for each of these requirements. However, to implement the protocol, we need to determine specific techniques for each of them.
Chapter 5
Implementation

The previous chapter provided a detailed description of the proposed solution. In this chapter, we provide an overview of the prototype implementation. We start the description by providing a brief overview of the implementation choices.

5.1 Implementation Choices

As identified in the previous chapter, we need to specify a challenge response mechanism, an owner authentication mechanism and also the data encryption technique. Apart from these, we also need platforms to implement the smart home device functionality and trusted device/new control device functionality. A brief overview of implementation choices based on the requirements are discussed below:

Components: To realize the system, we need to define three components which are: context, trusted device identity and owner profile. We have the following instantiations for these components in our implementation.

- Context: The term context in the implementation refers to the triplet <SSID, MAC address, access-credential of an AP>.

- Trusted Device Identity: It includes the Bluetooth device name and MAC address of the trusted device and the established shared security attribute between the trusted device and smart home device.

- Owner Profile: In the implementation, owner profile refers to the known context list, the trusted device identity list, salt and the profile name. The term salt here refers to a random number used for deriving the encryption key. The details are provided in Section 5.5.
CHAPTER 5. IMPLEMENTATION

**Challenge Response Mechanism:** To implement a challenge response mechanism, the smart home device and the trusted device require a security association between them, one form of which is establishment of shared secret. The shared secret establishment in turn requires a key agreement protocol. For establishing a shared secret, we perform a Diffie-Hellman \([50]\) key exchange between the smart home device and the trusted device during initial configuration. This established shared secret is later used as the basis of challenge response mechanism for verifying the trusted device in case of an ownership change.

**Owner Authentication Mechanism:** For authenticating the owner in case of unavailability of the trusted device, chownIoT requires an authentication mechanism. We chose to implement a password based authentication for authenticating the owner. During the initial configuration, the owner sets up a password of his/her own choice. Later when required, the owner proves his/her authenticity by demonstrating knowledge of the defined password.

**Data Encryption Mechanism:** chownIoT requires data encryption to preserve privacy of the previous owner in case of an ownership change. For data encryption, we implement AES-CCM authenticated encryption technique. The authenticated encryption mechanism provides authenticity and encryption at the same time.

**Initial Configuration:** During an ownership change, chownIoT needs to verify the authenticity of the owner. For this verification, it requires establishing some authentication credentials besides regular setup steps during initial configuration. We implement the initial configuration using Bluetooth communication between the smart home device and the trusted device.

**Smart Home Device:** The smart home device features of chownIoT are implemented using Raspberry Pi development board. In addition, we chose C++ as the programming language for the actual implementation of the smart home device functionality.

**Trusted Device/New Control Device:** The trusted device and new control device features of chownIoT is implemented using the Android platform.

**Communication Protocol:** To realize chownIoT, we require a communication protocol between the smart home device and trusted device/new control device. Thus, we define a packet based communication protocol. In addition, we use UDP for sending and receiving the packets of the defined protocol.
5.2 Components

In this section, we provide a detailed description of the three important components of chownIoT. As identified in the previous section these three components are: context, trusted device identity and owner profile. These components are used repeatedly in next subsequent sections. Thus, it becomes necessary to discuss them beforehand. Table 5.1 provides an overview of the elements for each of the three components.

<table>
<thead>
<tr>
<th>Context</th>
<th>Trusted Device Identity</th>
<th>Owner Profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>AP SSID</td>
<td>Bluetooth Device Name</td>
<td>Known Context List</td>
</tr>
<tr>
<td>AP MAC Address</td>
<td>Bluetooth MAC Address</td>
<td>Trusted Device Identity</td>
</tr>
<tr>
<td>AP Access Credential</td>
<td>Shared Secret</td>
<td>Salt</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Profile Name</td>
</tr>
</tbody>
</table>

Table 5.1: Elements for Each Component

5.2.1 Context

chownIoT identifies the change of ownership based on change in context. For the prototype implementation, we consider, Wi-Fi information as the contextual information of the smart home device. It means that a change in Wi-Fi connection of smart home device infers a change in context. To have internet connectivity, the smart home device needs to connect to an AP. The smart home device receives the SSID and access credential of the desired AP from the control device during initial configuration. It stores the SSID, MAC address and access credential of the connected AP during initial configuration and terms it as known context. During operation, the smart home device continuously monitors the currently connected AP information against the stored known context. If there is any mismatch, it triggers a ownership change handle procedure. The known contexts are specific to the owner, which means each owner has a separate list of known contexts. From security point of view, SSID and MAC address can be easily spoofed which in turn makes known context spoofable. However, apart from SSID and MAC, we also store the access credential of the AP which is difficult to spoof. As a result, the access credential mismatch protects the device from connecting to a spoofed known context.
5.2.2 Trusted Device Identity

cnownIoT considers the control device used during initial configuration as the trusted device. During ownership change process, at first cnownIoT looks for the trusted device. Thus, it needs to store the identity of the trusted device during initial configuration. In addition, for the verification of the trusted device cnownIoT establishes a shared secret between the trusted device and smart home device. In the prototype implementation, we use Bluetooth for the initial communication. The smart home device stores the Bluetooth device name, MAC address and the shared secret as the trusted device identity as illustrated in Table 5.1. When the smart home device needs to find the trusted device it triggers a Bluetooth scan looking for the stored Bluetooth device name and MAC address. If the Bluetooth discovery is successful, it verifies the identity using the shared secret. Similar to known context scenario, the trusted device identity can also be spoofed. However, the trusted device is verified using the shared secret and even if an adversary is able to spoof the device name and MAC address, spoofing the shared secret is not possible. Thus, authentication of trusted device based on shared secret protects against spoofing the trusted device identity.

5.2.3 Owner Profile

Owner profile is one of the main components of cnownIoT. In cnownIoT, the smart home device maintains a profile for each owner. This profile management enables the isolation of one user profile from the other which enhances user’s privacy. As discussed in Section 5.2.1 and 5.2.2, the smart home device needs to store known contexts and the trusted device identity which are specific to owner. Apart from this, cnownIoT also needs to store a salt value which is used to derive encryption key. The salt is also specific to owner and each owner has a unique salt. Each owner profile is identified using the profile name which is selected by the owner during initial configuration. Thus, in the implementation, the term owner profile refers to the combination of known context list, trusted device identity, salt value and profile name as depicted in Table 5.1. The smart home device can have multiple known contexts and trusted devices for a single owner during its life cycle. This improves the usability of the system. For instance, if an owner frequently moves his smart home device between his apartment and summer cottage, then a single known context requires the owner to authenticate during every move. However, due to multiple known contexts, the owner needs to authenticate only once for each new context. Similarly, having multiple trusted devices allows, multiple owner devices (e.g. multiple family members in a home) to have full control.
of the smart home device which also improves the user experience.

![Sequence Diagram for Shared Secret Derivation (Diffie-Hellman Key Exchange) and Challenge-Response Mechanism](image)

Figure 5.1: Sequence Diagram for Shared Secret Derivation (Diffie-Hellman Key Exchange) and Challenge-Response Mechanism

### 5.3 Challenge Response Mechanism

During a possible ownership change, in case of availability of the trusted device, chownIoT authenticates it using a challenge-response mechanism. There are already several techniques available to authenticate low-end devices using challenge response mechanism, such as challenge-response based RFID
authentication protocol [51, 54] and authentication protocol in security layer for smart tags [20]. Similar mechanisms can be used for authenticating the trusted device. The main requirement for a challenge response mechanism is that the two involved parties should have a security association between them. There are different techniques to establish a security association. For instance, the involved parties can have each others public key and possess own private key. Key agreement is an alternative mechanism for security association where the involved parties establish a shared secret key between them. The public/private key mechanism require larger key sizes than symmetric key to achieve similar security [35]. In addition, they also require more computational processing power [34]. As our system is intended for resource constrained devices, we chose shared secret to establish a security association between the smart home device and the trusted device. There are again several key agreement protocols available for establishing the shared secret. One such example is Diffie-Hellman key exchange protocol [50]. In the prototype implementation, we establish a shared secret between the smart home device and the trusted device using Diffie-Hellman key exchange protocol. For the smart home device, the OpenSSL [1] C library is used for the implementation. In addition, for trusted device the Java security libraries are used. Figure 5.1 depicts the sequence diagram of the shared secret generation and also the challenge response mechanism. The shared secret generation process is initiated by generating the parameters (1) required for the key exchange. The parameters include the prime number (p) and the generator/primitive root modulo p (g). The prime number p used in our implementation is a 2048 bit long fixed prime which is pre-computed and provided to the smart home device in advance. Next, the smart home device generates the private and public key pair (2). Once the keys are generated, the smart home device sends the parameters (i.e. p and g) along with the public key (3) to the trusted device and waits for the response. Once the trusted device receives the parameters and the key, it generates its key pair (4) using the received parameters (i.e. p and g). Upon generating the key pair, the trusted device sends its public key (5) as response to the smart home device. Finally, both the smart home device and the trusted device derive the shared secret (6) using their own private key and received public key. This process is performed for a particular owner only once. If an owner wants to add a new control device as a trusted device, the smart home device sends the initially generated shared secret to the new control device. Thus, all the trusted devices under a particular owner shares the same secret key.

\[ \text{Response} = HMAC_{SHA256}(Secret, \text{Challenge}) \]  

(5.1)
CHAPTER 5. IMPLEMENTATION

Next, when the smart home device needs to authenticate the trusted device, it generates a pseudo random number (7) and sends it to the trusted device (8) as a challenge. The challenge is generated using the pseudo random generator function `srnd()` from GNU C library\(^1\) and it is seeded with time. To calculate the response, the trusted device computes the Hashed Message Authentication Code (HMAC) [11] of the challenge using the shared secret key (9). The smart home device also generates the same response by computing the HMAC of the challenge. SHA256 hash algorithm is used for computing the HMAC. HMAC provides better security than just directly hashing the message concatenated with the key. In addition, it is significantly faster than other secure authentication tag generation mechanisms, such as CBC-MAC [12]. The response calculation is illustrated in equation 5.1. Once the response calculation is done, the trusted device sends the calculated response to the smart home device (10). Finally, the smart home device compares its own generated response with the received value (11). A match indicates a successful authentication whereas a mismatch the opposite.

5.4 Authentication Mechanism

Apart from authenticating the trusted device, chownloT needs to authenticate the new control device or more specifically the owner, in case of unavailability of the trusted device. There are several different authentication mechanisms available, such as password based authentication, public key based authentication protocol, Authentication and Key Agreement protocol (AKA) and Extensible Authentication Protocol (EAP). Moreover, designing lightweight authentication protocol for IoT environment has been among the widely researched topics in recent years [21]. Thus, based on our requirements we need to select any such authentication mechanism for authenticating the new control device or the owner in particular.

Password based authentication mechanisms are widely used as they are convenient to use and implement [64]. Moreover, in chownloT we require authenticating the owner on any control device. To do so, authentication mechanisms such as public key or AKA require transferring the authentication key to each control device, which is not convenient. Therefore, to satisfy the usability requirements, we chose to implement a password based authentication mechanism. During the initial configuration, the owner is prompted to setup a password. The password strength requirement must be enforced

using a configurable password policy. The minimum requirement for choosing a password is currently enforced using hard coded policy which is: at least 8 characters long with 1 uppercase 1 lowercase and 1 number. The policy is imposed to increase the strength of the password and reduce the possibilities of a successful password guessing attack. Thus, when the smart home device requires authenticating the owner on a new control device, it prompts the owner to enter the previously defined password. Once the smart home device receives the owner response, it verifies the password to ensure the authenticity of the owner.

5.5 Data Encryption Mechanism

crownloT needs to protect the previous owner’s privacy sensitive data once it infers that the ownership has changed. There can be different techniques for protecting the data. Access control is one such technique where the smart home device restricts the access of the new owner to previous owner’s data. However, this technique can be insecure due to the fact that the new owner has physical access to the device as discussed in Section 3.1. The new owner can misuse this physical access capability to retrieve the previous owner’s data by physically reading the particular memory location. Therefore, the protection technique needs to be stronger than just access control. Thus, in case of an ownership change, we decide to encrypt the previous owner’s data available on the smart home device.

For data encryption, we use AES-CCM [37] authenticated encryption technique in the prototype implementation. It is a technique that provides both authentication as well as encryption at the same time. In addition, we use key derived from owner provided password as the encryption key for AES-CCM. We cannot store the encryption key on the device as physical memory access can leak the encryption key. In addition, we also require to provide profile retrieval mechanism using any control device. Thus, we cannot store the key on the control devices either. To fulfill these requirements, we chose to derive the key from owner password. The main benefit of password based key derivation is that it can be instantly derived from the owner provided password without requiring to be stored and it also facilitates profile retrieval using any control device. During the initial configuration the smart home device receives the hash of the owner password and derives a key from the password for the authenticated encryption. However, passwords are known to be insecure against dictionary and brute force attacks. Thus, weakness of the passwords can make password based key derivation insecure. One way to secure the key derivation is to use trusted hardware, such as ARM
TrustZone-M [4]. The key derivation function can be implemented inside the trusted execution environment. The trusted hardware derives the key using the password and another strong key which never leaves the hardware. In addition, the number of password guessing can be controlled by rate limiting the use of strong key with monotonic counters [48]. Therefore, use of trusted hardware can protect against brute forcing the password to derive the encryption key. However, it is currently not possible to implement due to lack of full support for trusted execution in available low-end devices. In the current implementation, we use Password-Based Key Derivation Function 2 (PBKDF2) with 256 bits random salt and 4096 iterations of SHA256 hash algorithm to derive the key from the password. The key derivation function is illustrated in equation 5.2. The OpenSSL C library for PBKDF2 is used for the actual implementation.

\[
Key = H_{SHA256}(Password_{SHA256}, Salt_{256bit}, 4096\text{iterations})
\]  

(5.2)

At the time of initial configuration, the control device performs one way hash of owner password and sends it to the smart home device. It is worth noting that hashing the password at control device end does not provide any additional benefits in terms of security. However, as the hash functions provide fixed length outputs, from implementation perspective sending and receiving fixed length data is easier than arbitrary length. Next, as soon as the smart home device receives the hash of the owner password from the trusted device, it derives the key according to equation 5.2. The salt for the key derivation is also randomly generated using srand() function, similar to the challenge generation discussed in Section 5.3. Once the key derivation is completed the smart home device only stores the salt and the key, however, deletes the password hash. The salt and the key are stored in file. During a possible ownership change, if the trusted device is unavailable or it fails to authenticate, the smart home device encrypts the profile data except the salt value using AES-CCM and the derived key. Once the encryption process is completed, it also deletes the derived key. Therefore, when the owner wants to retrieve the encrypted profile, he/she is prompted to provide the owner password. Upon receiving the password hash, the smart home device again derives the key using the provided password hash and stored salt value. Once the key is derived, it performs authenticated decryption using AES-CCM and the derived key. If the process is successful the owner is authenticated and the profile data gets decrypted. It is important to note here that the profile data gets encrypted using password derived key and that the smart home device neither stores the derived key nor the password hash. As a result, once a profile gets encrypted, the trusted device has to complete the owner...
authentication process (i.e. provide the owner password) in order to retrieve
the profile. The challenge response process is not applicable in such scenario.

5.6 Initial Configuration - Figure 4.3 (1) - (6)

The initial configuration is responsible for starting the operation of smart
home devices in the deployment environment. chownIoT requires some addi-
tional steps besides traditional configuration steps during initial configu-
ration. In this section, we provide the implementation details of the initial
configuration steps of chownIoT illustrated in Figure 4.3.

For the initial configuration at first, the smart home device performs some
initialization steps (1) and eventually waits for the initial configuration re-
quest from some control device. The initialization steps (1) include turning
on the discoverable mode of Bluetooth and creating server socket which lis-
tens for packets on a particular port. The smart home device listens on port
6346. The initial configuration request is implemented using Bluetooth pair-
ing. The control device discovers the smart home device and sends a pairing
request. The smart home device responds to the pairing request and once
paired, the control device sends the SSID and access credential of the desired
AP to the smart home device. Next, the smart home device establishes the
Wi-Fi connection with the AP using wpa_supplicant [36]. Once the connec-
tion is established, the device stores the current context as known context
(6) (Section 5.2.1). It is stored the persistent storage (e.g. stored in a file).

Next, the smart home device establishes a shared secret (2) with the
control device using Diffie-Hellman key exchange (Section 5.3). It also stores
the trusted device identity (3) (Section 5.2.2). Similar to the known context,
the trusted device identity is stored in persistent storage. In addition, the
smart home device establishes a password based authentication (4) (Section
5.4). Apart from this, during the initial configuration the smart home device
also receives the hash of the owner password and a profile name from the
trusted device. As discussed in Section 5.5, once it receives the password
hash, it derives the encryption key according to equation 5.2. After the key
derivation, the smart home device stores the salt and the derived key in file
storage (5) and deletes the password hash. The received profile name is used
for uniquely identifying a particular owner profile. The smart home device
maintains a list of existing profiles using the profile names.
5.7 Smart Home Device

This section provides an overview of smart home device feature implementation of chownIoT. The smart home device is one of the major components of chownIoT. There are two main features of the smart home device in chownIoT, which are: ownership change detection and owner authentication. We start the description by discussing the implementation platform and then in Section 5.7.1 and 5.7.2 we discuss ownership change detection and owner authentication respectively in detail.

*Implementation Platform:* IoT devices are diverse in terms of their capabilities, operating principles and hardware specifications. In addition, most smart home devices available in the market are closed source and do not provide any programming capability. Device vendors enforce restrictions due to business requirements. However, to implement the proposed protocol, we require a programmable device. Raspberry Pi is a strong representative of a typical IoT device with similar capabilities and it also costs low. In addition, it has a development friendly environment, supporting modern programming languages. Due to all these aforementioned reasons, we chose Raspberry Pi for the prototype implementation. The implementation of this module is done in C++ programming language on Raspberry Pi 3<sup>2</sup> running on Raspbian Jessie<sup>3</sup> operating system.

5.7.1 Ownership Change Detection - Figure 4.4 (1)

The main challenge of implementing the smart home device features of chownIoT is automatic detection of ownership change. Although, the proposed system is not focused on automatic ownership change detection, for the prototype implementation, we develop a simple technique. According to the classification provided in Section 2.5, most smart home devices available in the market are either static or semi static in terms of mobility and also Internet reachable in terms of Internet capability. Most devices achieve Internet connectivity through wireless connection with an AP. As discussed in Section 2.3.1, the AP broadcasts the SSID and smart home devices connect to it using the SSID. The devices know the SSID or the name of the wireless network that it is currently connected to. Furthermore, in a typical smart home scenario, the SSID is the same for the whole house or apartment. Therefore, according to the characteristics of static and semi static devices, their

---

<sup>2</sup>https://www.raspberrypi.org/products/raspberry-pi-3-model-b/

CHAPTER 5. IMPLEMENTATION

connected SSID is unlikely to change until a particular device is under the same owner. However, in an ownership change scenario, a new owner needs to connect to a different AP, which also changes the SSID of the particular smart home device. Thus, the SSID can be considered as a contextual information based on which the change of ownership can be detected. The smart home device stores the context as defined in Section 5.2.1 during the initial configuration in the list of known contexts. In addition, during the operation the smart home device continuously monitors the context. If the current context does not match with any of the entries in the list of known contexts, it infers a possible ownership change. In the implementation, we only consider the state when we detect a new SSID as a possible ownership change scenario. We discard the disconnection state of the device, when there is no SSID detected. This reduces the possibility of false positives, as the smart home device can be disconnected from the AP due to different reasons. Moreover, even if there is a false detection of ownership change, the seamless verification of trusted device helps protecting the usability of the system.

5.7.2 Owner Authentication - Figure 4.4 (2) - (14)

In chownloT, during operation, if the smart home device detects an ownership change, it requires authenticating the owner. In the implementation, first the smart home device triggers a Bluetooth discovery looking for the trusted device (2). If the trusted device is discovered, it resolves the new IP address of the trusted device by parsing identity broadcast packet (i.e. discussed in Section 5.9) repeatedly broadcasted by the trusted device. Upon resolving the IP address, the smart home device sends a challenge (3) and waits for the response. During this waiting time the smart home device also calculates the expected response value of the sent challenge according to equation 5.1 described in Section 5.3. Similarly, the trusted device calculates the response and sends it back to the smart home device. Upon receiving a response, the smart home device compares the received response with the computed expected response. If the challenge is satisfied, it stores the current context as known context (4). However, if the challenge is not satisfied, the smart home device encrypts the profile data (5) according to the mechanism discussed in Section 5.5.

If the smart home device is unable to discover the trusted device in Bluetooth discovery, similar to the previous case it encrypts the profile data (6). In addition, it starts looking for a new control device (7). The discovery of new control device is implemented using broadcast packets. New control devices continuously (in 30 seconds interval) broadcast their identity in the network. The identity broadcast packet includes the Bluetooth device name,
MAC address and the IP address of the control device. When the smart home device is unable to find the trusted device, it starts looking for the broadcast packets from a new control device. Upon identifying a new control device, it sends a list of available owner profiles (8) and waits for the user choice response. The user here can either choose an existing profile or can create a new profile. If the user selects an existing profile, the smart home sends a request for authentication (9) and waits for the response. The authentication response contains the hash of the owner password of the selected profile. Upon receiving the response, the smart home device verifies it according to the mechanism discussed in Section 5.5. If it receives a satisfying response, the data gets decrypted (10) and it stores the current context as known context (11). However, if the new control device is unable to authenticate or the user chooses to create a new profile (12), the smart home device creates a new profile for the new control device by establishing a new shared secret and requesting to setup new owner password (13) as mentioned in Section 4.2. In addition, it stores the current context in a new known context list and also control device identity in a new trusted device list which is specific to the new owner profile. Apart from this, smart home device can also receive a request from a user to retrieve an existing user profile at any time. In such a case, the device first encrypts the current profile and then initiates the required retrieval process.

5.8 Trusted Device/New Control Device

This section provides the implementation overview of the trusted device and new control device features of chownIoT. Both the trusted device and the new control device features of chownIoT are implemented in a control application running on Android platform. Similar to traditional smart home device applications, the control application provides configuration interface for smart home device. Figure 5.2 illustrates the user interfaces of the application during different stages of operation starting with initial UI in snippet (1). There are mainly three functions of the control application, which are: device discovery, password and shared secret setup and authentication and profile management.

Device Discovery: The first task of the control application is smart home device configuration. To configure the device, user selects Configure Device option which triggers a Bluetooth scan to discover the smart home device as depicted in snippet (2). Upon discovering the smart home device, the application prompts the discovered device information with pairing option to the
user as shown in snippet (3). The user needs to select pair to trigger pairing process. Once the pairing process is completed, the application prompts user to connect the device to Wi-Fi. The snippet (4) in figure 5.2 depicts the UI for the Wi-Fi connection prompt. The user enters the SSID and access credential of the desired AP and taps to connect. Next, the application sends the trusted device IP address to the smart home device. This IP address is sent to enable Wi-Fi communication between the trusted device and smart home device, as prior this point none of them possess any IP address related

Figure 5.2: User Interface (UI) of Control Application
CHAPTER 5. IMPLEMENTATION

information of each other.

**Password and Shared Secret Setup:** Once the Wi-Fi connection is successful, the application prompts for setting up owner information as depicted in snippet (5). User needs to enter a profile name and the desired owner password. The password requirements should be enforced using configurable password policies in order to avoid weak passwords. In the implementation, we currently support only one hard-coded password policy according to which the password must be: at least 8 characters long with 1 uppercase 1 lowercase and 1 number. The password strength checking is implemented using regular expressions. Once the user selects the desired profile name and password satisfying the strength requirements, it sends the profile name and hash of the password to the smart home device. The password hashing is implemented using SHA256 hash algorithm [25]. Upon receiving the owner information, the smart home device initiates the key agreement process. The process is performed in the background. During the background process a configuring device progress bar is shown in the UI. Once the process is completed, the application shows user a message i.e. "Device Configuration Successful". Apart from this, in order to receive request during a possible ownership change the application continuously listens to the port 6346.

**Authentication and Profile Management:** Next, for the new control device functionality, the control application broadcasts the device identity (i.e. Bluetooth device name, MAC address and IP address) in every 30 seconds interval. Therefore, when the trusted device is not available, the smart home device identifies the new control device by parsing the broadcast packet. In all other situations the identity broadcast packets are discarded. In addition, during a possible ownership change when the application receives the list of existing profiles, it prompts user to choose either a profile from existing list or create a new profile. Once the user enters the choice, it sends the response to the smart home device. If the user selects an existing profile, it prompts the user to enter the password for the selected profile. Once the user enters the password it forwards the hash (i.e. SHA256) of the entered password to the smart home device for verification. However, if the user chooses to create a new profile, it prompts the user to enter owner information as depicted in snippet (5) of Figure 5.2. Apart from this, the user can also retrieve any existing profile at any time by selecting Manage Profile option from the initial UI shown in snippet (1) of Figure 5.2. The manage profile option requests the smart home device to send the list of existing profiles. Once the application receives the list, the rest of the process happens similarly as in the case of user choosing an existing profile during a possible ownership change discussed previously.
5.9 Communication Protocol

This section discusses about the communication protocol between the smart home device and the control application. The communication between the two parties is performed using User Datagram Protocol (UDP) sockets. UDP is faster and more efficient than protocols such as Transmission Control Protocol (TCP) due to less overhead [60]. As the solution is intended for devices with limited resources, UDP is well suited for the scenario. To facilitate the communication, we defined a simple protocol. The communication protocol is a binary protocol which works on top of UDP and contains a 12 bytes fixed header. Figure 5.3 illustrates the structure of a generic packet. The packet contains 12 bytes of fixed header and n bytes of payload. The meaning of the different fields are as follows:

**Version**: The version of the protocol. It is currently 1.

**Message Type**: The type of the message. The valid values are provided in Table 5.2.

**Reserved**: Reserved bits for future protocol modification and enhancement. It is unused and set to zero in this version.

**Payload Length**: The length of payload in bytes, header length is discarded.

**Sender Port**: The listening port number of the original sender.

**Sender IP**: The IPv4 address of the original sender. The sender IP field in the header facilitates multihop communication scenarios where two devices may communicate through other devices. In such scenario packet received from a device may not be the original sender but only a forwarder.

**Payload**: The actual message, specific to the message type.

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
<th>21</th>
<th>22</th>
<th>23</th>
<th>24</th>
<th>25</th>
<th>26</th>
<th>27</th>
<th>28</th>
<th>29</th>
<th>30</th>
<th>31</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
<td>31</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Version</th>
<th>Message Type</th>
<th>Reserved</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Payload Length</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Sender Port</td>
<td></td>
</tr>
</tbody>
</table>

Sender IP

Payload (n bytes)

...
The protocol also defines 17 different message types for different communication requirements between smart home device and trusted/new control device. Table 5.2 illustrates the different message types. The details of each message and their corresponding payloads are as follows:

1. **MSG_IP_BROADCAST**: Intended for broadcasting IP address during initial configuration. The payload of this message contains the device name of the control device used during the pairing process.

2. **MSG_IP_UNICAST**: Used by smart home device to send the response of IP broadcast packet. The payload of this message contains smart home device name.

3. **MSG_OWNER_INFO**: Contains profile name and hash of the user defined password. It is sent by the control device to the smart home device.

4. **MSG_OWNER_INFO_CONFIRMATION**: Used by smart of device to send confirmation response of OWNER_INFO message. The payload is empty for this message.

<table>
<thead>
<tr>
<th>Message Type</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSG_IP_BROADCAST</td>
<td>0x01</td>
</tr>
<tr>
<td>MSG_IP_UNICAST</td>
<td>0x02</td>
</tr>
<tr>
<td>MSG_OWNER_INFO</td>
<td>0x03</td>
</tr>
<tr>
<td>MSG_OWNER_INFO_CONFIRMATION</td>
<td>0x04</td>
</tr>
<tr>
<td>MSG_IDENTITY_BROADCAST</td>
<td>0x05</td>
</tr>
<tr>
<td>MSG_CHALLENGE</td>
<td>0x06</td>
</tr>
<tr>
<td>MSG_CHALLENGE_RESPONSE</td>
<td>0x07</td>
</tr>
<tr>
<td>MSG_AUTHENTICATION_REQUEST</td>
<td>0x08</td>
</tr>
<tr>
<td>MSG_AUTHENTICATION_RESPONSE</td>
<td>0x09</td>
</tr>
<tr>
<td>MSG_KEY_&amp;_PARAMETERS</td>
<td>0x10</td>
</tr>
<tr>
<td>MSG_KEY_&amp;_PARAMETERS_RESPONSE</td>
<td>0x11</td>
</tr>
<tr>
<td>MSG_PROFILE_LIST</td>
<td>0x12</td>
</tr>
<tr>
<td>MSG_PROFILE_LIST_RESPONSE</td>
<td>0x13</td>
</tr>
<tr>
<td>MSG_PROFILE_REQUEST</td>
<td>0x14</td>
</tr>
<tr>
<td>MSG_CREATE_PROFILE</td>
<td>0x15</td>
</tr>
<tr>
<td>MSG_SHARE_SECRET</td>
<td>0x16</td>
</tr>
<tr>
<td>MSG_SHARE_SECRET_RESPONSE</td>
<td>0x17</td>
</tr>
</tbody>
</table>

Table 5.2: Protocol Message Types
5. **MSG.IDENTITY_BROADCAST**: Sent by new control device to broadcast identity. The payload contains the Bluetooth MAC address and device name of the new control device.

6. **MSG.CHALLENGE**: Used for sending a challenge to the trusted device. The payload contains the challenge value.

7. **MSG.CHALLENGE_RESPONSE**: Intended for sending the response value of a corresponding challenge value. The payload contains the response value.

8. **MSG.AUTHENTICATION_REQUEST**: Used by the smart home device to send an authentication request to a new control device. The payload is empty for this message.

9. **MSG.AUTHENTICATION_RESPONSE**: Intended for sending authentication response from new control device to the smart home device. The payload contains the hash of the owner password.

10. **MSG.KEY_PARAMS**: Used for sending the key agreement parameters and the public key of the smart home device to the trusted device. The payload contains the prime number, the group and the public key.

11. **MSG.KEY_PARAMS_RESPONSE**: Used for sending the public key of the trusted device to the smart home device as the response of **KEY_PARAMS** message. The payload contains the public key.

12. **MSG.PROFILE_LIST**: Intended for sending list of existing profiles to the new control device. The payload contains the number of profiles followed by the profile names.

13. **MSG.PROFILE_LIST_RESPONSE**: Used to send the user choice response of selected profile to the smart home device. The payload contains selected profile name.

14. **MSG.PROFILE_REQUEST**: Sent by the trusted/new control device to request existing profile lists from the smart home device. The payload is empty for this message.

15. **MSG.CREATE_PROFILE**: Used to send a create new profile request to the smart home device from the trusted/new control device. The payload of this message is also empty.
16. \textit{MSG\_SHARE\_SECRET}: Intended for sending the shared secret from smart home device to the new control device when an owner wants to add a new control device as trusted device. The payload contains the shared secret.

17. \textit{MSG\_SHARE\_SECRET\_RESPONSE}: Used for sending the shared secret received confirmation to the smart home device from the new control device. The payload is empty.
Chapter 6

Evaluation

This chapter evaluates the proposed system based on the identified requirements and challenges in Section 3.3. In the next subsequent sections, we discuss about the resource constraints, robustness, deployability and usability of chownIoT in details.

6.1 Resource Constraints

Some smart home devices have limited resources. To work on such devices, the proposed system needs to be efficient and have least overhead. In this section, we evaluate the performance of the proposed solution.

CPU Usage: Some IoT devices, such as smart switch produce small amount of data (i.e. 10KB) whereas devices such as weather stations produce significantly more data (i.e. 10MB) than smart switches. In addition, devices such as surveillance camera can produce large amount of data (i.e. 100MB). The major resource hungry operations in chownIoT are encryption, key derivation and hashing. During encryption besides protocol data the smart device needs to encrypt the data produced by these devices. Thus, to evaluate the performance of chownIoT, we measure the CPU usage for encrypting 10KB, 10MB and 100MB data.

<table>
<thead>
<tr>
<th>Data Size</th>
<th>CPU Usage (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10KB</td>
<td>0.010</td>
</tr>
<tr>
<td>10MB</td>
<td>5.89</td>
</tr>
<tr>
<td>100MB</td>
<td>60.67</td>
</tr>
</tbody>
</table>

Table 6.1: Encryption CPU Usage
Table 6.1 illustrates the CPU usage measured in seconds for different data sizes. As we can see from the measurement data that 10KB and 10MB data requires only 0.010 second and 5.89 seconds respectively for encryption which is not a large computational overhead. Most IoT devices produce data between 10KB to 10MB. Thus, encrypting data on such devices is perfectly feasible. Apart from this, we can also see from table that 100MB data require 60.67 seconds. For instance, surveillance cameras can produce this amount of data. Such devices are also equipped with good processing capabilities. In addition, these devices do not need to be always active. Thus, for such devices spending 60 CPU seconds for encrypting data is feasible. Apart from encryption, we also measured CPU usage for key derivation and hashing. Table 6.2 illustrates the measured value. The time required for key derivation and hashing are 0.060 and 0.010 second respectively. Similar to the case of encryption, the key derivation and hashing does not introduce large computational overhead and feasible for most IoT devices. Table 6.3 illustrates the CPU specifications of three of the popular IoT boards. Arduino Tian and Intel Edison have less powerful CPUs than Raspberry Pi 3. The resource hungry operations may need twice the CPU time of Raspberry Pi 3 to execute on these boards. Thus, operations, such as encryption of large data can eventually degrade the usability of the system on these boards due to the limited processing power of the CPUs.

<table>
<thead>
<tr>
<th>Operation</th>
<th>CPU Usage (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key Derivation</td>
<td>0.060</td>
</tr>
<tr>
<td>Hashing</td>
<td>0.010</td>
</tr>
</tbody>
</table>

Table 6.2: CPU Usage for Key Derivation and Hashing

**Memory Usage:** Besides limited computational power IoT devices also have limited memory. Thus, we also measured RAM usage of each resource hungry operation. Table 6.4 illustrates the measured RAM usage. For encrypting different data sizes (i.e. 10KB, 10MB and 100MB) and also for key deviation and hashing, the RAM usage is approximately 1MB. It is constant regardless of data size and operation performed. Some of the currently avail-

<table>
<thead>
<tr>
<th>Board</th>
<th>CPU</th>
<th>RAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raspberry Pi 3</td>
<td>ARM Cortex-A53, 1.2GHz</td>
<td>1GB</td>
</tr>
<tr>
<td>Arduino Tian</td>
<td>Atheros AR9342 560MHz</td>
<td>64MB</td>
</tr>
<tr>
<td>Intel Edison</td>
<td>Dual-Core Intel Atom 500MHz</td>
<td>1GB</td>
</tr>
</tbody>
</table>

Table 6.3: Specifications of IoT Boards
Table 6.4: Memory Usage for Different Operations

<table>
<thead>
<tr>
<th>Operation</th>
<th>Ram Usage (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encryption 10KB</td>
<td>1.031</td>
</tr>
<tr>
<td>Encryption 10MB</td>
<td>1.051</td>
</tr>
<tr>
<td>Encryption 100MB</td>
<td>1.094</td>
</tr>
<tr>
<td>Key Derivation</td>
<td>1.047</td>
</tr>
<tr>
<td>Hashing</td>
<td>1.045</td>
</tr>
</tbody>
</table>

able popular IoT boards are Raspberry Pi, Arduino and Intel Edison. RAM specifications of these three boards are illustrated in Table 6.3. Raspberry Pi 3 and Intel Edison are equipped with 1GB RAM and Arduino Tian has 64MB RAM. As most resource hungry operations of chownIoT require around 1MB of RAM, thus it can be easily run on any of these boards and also most IoT devices.

6.2 Robustness

In Section 3.1 and 3.2, we identified several threats that can arise due to ownership change of smart home IoT devices. Next in Section 4.1 we introduced chownIoT that is intended to provide protection against these identified threats. In this section, we evaluate the robustness of chownIoT by analyzing how it provides protection against each of the threats. We start by discussing about different protection mechanisms of chownIoT.

1. Data Protection: The profile management system of chownIoT isolates owner profiles from one another. This isolation protects the data a particular owner from other owners of the smart home device. chownIoT encrypts the current owner’s profile data as soon as it detects the absence of the trusted device in case of an ownership change. For the implementation, we used AES encryption with 128 bit key length which is recommended by NIST [23]. The mode we use for encryption is AES-CCM. AES-CCM provides authenticated encryption. Authenticated encryption generates an authentication tag to verify the authenticity of the encrypted data which is not present in normal encryption modes. As we do not store the encryption key or password, we need to authenticate the owner by decrypting the encrypted data using the key derived from user provided password. Thus, to verify whether the decryption process is successful or unsuccessful we use the authentication tag generated during the encryption process. One problem with AES-CCM is that it becomes vulnerable if the same nonce is reused. As the resource
CHAPTER 6. EVALUATION

constrained devices have limited computational power, nonce repetition is a possible scenario. However, in chownIoT, as we do not encrypt data all time, we assume that possibilities of repeating a nonce is negligible. The system deletes both the owner password and the derived encryption key once the data gets encrypted. Thus, even if an adversary tries to retrieve data by reading physical memory location, he/she is only able to retrieve the encrypted data. Moreover, as the owner password or the encryption key is not stored, the only possibility left to retrieve the actual data is to brute force the password and derive encryption key. To prevent brute forcing the password, we rate limit the number of allowed consecutive tries for a particular profile within a specific time period. Apart from this, as discussed in Section 5.5, the use of trusted hardware to derive the encryption key can also protect against password brute force as in our system every authentication attempt requires key derivation. However, none of these techniques are effective for offline brute force attack where the adversary has the encrypted data and salt to generate the encryption key offline, which is a limitation of chownIoT. Lastly, the physical access capability on the device can also expose chownIoT to side channel attacks such as, cold boot attack on encryption key [24]. In cold boot attack, the attacker makes use of the data remanence property of DRAM and SRAM to retrieve the encryption key after a cold reboot of the device.

2. Protection against Context Manipulation/Spoofing: From the implementation point of view, the known context can be spoofed by replicating the AP SSID and MAC address. chownIoT stores both the SSID and access credential of the AP for the known context information. Thus, even if the adversary is able to spoof the SSID and MAC, however, spoofing the access credential is difficult. As a result, the credential mismatch prevents the device from connecting to a spoofed known context.

3. Protection against Trusted Device Spoofing: Depending on the communication technology, chownIoT stores the identity (e.g. Wi-Fi MAC) of the trusted device during initial configuration. Later, in case of an ownership change, it looks for the trusted device based on the stored identity. However, the trusted device can be easily spoofed by spoofing the device MAC address. To protect against such spoofing, chownIoT authenticates the trusted device using a challenge response technique which depends on the shared secret established between the trusted device and the smart home device during initial configuration. Therefore, even if an adversary spoofs the MAC address, spoofing the shared secret is not possible.

In Section 3.1, we identified different capabilities of previous and new
owner that can raise threats during ownership change. The misuse of physical access capability of new owner is protected using data encryption in chownIoT. In addition, the misuse of previous owner’s access capability to the device or cloud storage with previous credentials to retrieve new owner’s privacy sensitive data is protected by isolating profiles. Once chownIoT detects an ownership change, it creates a complete new profile for the new owner. The new profile requires setting up new trusted device and also new authentication credentials which eventually restricts previous owner from accessing new owner’s privacy sensitive information. Besides this, as chownIoT provides the capability of retrieving existing profile, a previous owner can misuse this capability to retrieve his/her owner profile while the current owner’s profile is active. We assume previous owner to be remote in such a scenario. Thus, reactivation of profile can be enforced by physical proximity provided by direct Bluetooth communication. Moreover, the new owner always has the capability of factory resetting the device which ultimately deletes all existing profile and also prevents the misuse.

**Limitation - Abnormal Power Off:** One limitation of chownIoT is that if the device gets stolen or abnormally powered off before detecting ownership change then the data remains unencrypted. As a result, if an adversary has physical access in such a scenario, then he/she can retrieve user data by reading the physical memory location. One solution to this problem can be encrypting data all the time. However, as the proposed system needs to be generic and some IoT devices can be battery powered. Thus, for such devices, it is not feasible due to the fact that encryption and decryption requires substantial energy. For this particular reason, chownIoT does not encrypt data all time. Nonetheless, for certain types of devices, encrypting data all time is feasible and certainly improves the overall security.

**Limitation - Relay Attack:** As pointed out in Section 4.1 that the challenge response based authentication with trusted device without any user intervention can introduce relay attacks [22]. During a relay attack in our implementation scenario the adversary spoofs the Bluetooth device name and address of owner’s trusted device. The adversary then takes the smart home device in a new context. As the context has changed, the smart home device identifies the spoofed device as trusted device and sends a challenge. Upon receiving the challenge, the adversary forwards the challenge to the actual trusted device via some remote communication. As there is no user intervention, the actual trusted device calculates the response and sends back to the spoofed device. The spoofed device then forwards the response to the smart home device. In such scenario, the previous owner profile remains ac-
CHAPTER 6. EVALUATION

ditive and also data remains unprotected due to fact that the authentication of trusted device is successful. Assessing the proximity of the participating devices based on their contextual data during each challenge response can be a potential solution to protect against relay attacks. Apart from this, one simple solution of this problem can be asking permission from the user each time before sending a response. However, this can reduce the usability of the system. Nonetheless, as a design choice, we preferred usability over security in this scenario. It is also worth noting that according to the current implementation during a context change the owner needs to keep the control application in the foreground to allow the trusted device respond to the challenge. Thus, it is not easy for an adversary to gain control of owners trusted device which reduces the possibility of the relay attack.

**Limitation - Malicious AP:** For identifying the spoofed context, in the current implementation, we do not explicitly check the authentication results, rather we rely on the AP. Thus, if a malicious AP accepts the authentication credential without any verification then the smart home device can connect to a spoofed known context. However, this can be mitigated by explicitly checking the authentication process which ensures that the AP is not spoofed and the actual authentication process is successful.

### 6.3 Deployability

The smart home devices are diverse in terms of their capabilities, operating systems, hardware specifications, mobility and available sensor modalities, as identified in Section 2.5. Thus, one of the main requirements for the proposed system is that it should be generic and adaptable to all possible class of devices. In this section, we evaluate chownIoT in terms of deployability.

**Operating System:** The prototype implementation of chownIoT uses C++ and Linux based operating system to implement the smart home device features of the proposed protocol. In addition, it uses packet based protocol running on top of UDP for communication. Features such as UDP and C++ are supported by most modern operating systems. Thus, realizing chownIoT in any operating system is possible.

**Hardware:** chownIoT does not depend on any specific hardware requirements. It can be implemented on any device supporting some form of communication (i.e. Wi-Fi, Bluetooth or ZigBee etc.) which is mandatory for a smart home device. As the system can be implemented using packet based protocol, thus it is not dependent on any physical communication technolo-
CHAPTER 6. EVALUATION

gies. We implemented the smart home device features on Raspberry Pi board. In table 6.3, we provided specifications of different popular IoT boards. Intel Edison and Arduino which are two popular IoT boards have similar specifications as Raspberry Pi. Thus, the smart home device features of chownloT can also be easily ported to Intel Edison or Arduino boards.

Sensor Modalities: chownloT requires sensor involvement only to detect the change of ownership. The privacy enhancement protocol of chownloT does not depend on any sensors. Moreover, the simple ownership change detection technique developed for the prototype implementation depends only on Wi-Fi communication and does not involve any other sensors. This ownership change detection technique can also be easily adapted for other communication technologies, such as Bluetooth and ZigBee, for instance by analyzing/monitoring the available nearby devices of a smart home device. Thus, it is possible to implement chownloT on devices with any number of sensor modalities as long as the device has a communication interface.

Mobility: chownloT is applicable for static, semi static and also mobile IoT devices. For static and semi static devices the context seldom changes, however, for mobile IoT devices the context can frequently change. chownloT stores all the known contexts for a particular owner. Therefore, for mobile IoT devices every new context needs to be authenticated only once. After authentication, the system automatically identifies the known context in future and does not prompt user for any authentication which makes the system applicable even for mobile IoT devices. Apart from this, some mobile IoT devices do not use Wi-Fi and rather pair with phone or other devices. For such devices, the current implementation of ownership change detection is not applicable.

Energy Consumption: Some IoT devices are battery powered, as a result energy considerations of chownloT becomes necessary. In the current implementation, the smart home device continuously loops to detect change of SSID for detecting ownership change. This require good amount of energy, as the monitoring process needs to run all time. This can be improved by implementing some call backs that will notify when there is any disconnection or state change of Wi-Fi connection rather than continuous monitoring. Apart from this, in case of control device, it needs to broadcast the identity. In the current implementation, the control device broadcasts its identity in every 30 seconds interval. This broadcast is only done when the control application is in the foreground. In addition, typically smartphones are used as control devices and currently they are equipped with powerful batteries. Thus, energy consumption due to identity broadcast does not have severe.
impact on the control devices. Nonetheless, to make the system more applicable for battery powered smart devices the ownership change detection technique needs to be improved.

### 6.4 Usability

Usability is one of the most important requirements of any solution. However, solutions for ensuring security/privacy can be complex which ultimately reduces its usability. In this section, we evaluate the usability of chownIoT.

1. **Initial Configuration** Deployment of a smart home device for the first time requires configuring the device. Typically, the initial configuration of a smart home device requires user intervention. The users need to provide network access credential as well as cloud credential which require active user participation. In chownIoT, besides this regular configuration, we additionally setup an owner authentication mechanism and a shared secret with the trusted device. The owner authentication mechanism only requires user participation. For instance in prototype implementation the user sets up a profile name and password. The shared secret generation does not require any active user involvement. Thus, the only overhead in terms of interaction is password setup, which improves usability of chownIoT.

2. **Ownership Change Management:** One major benefit of chownIoT is that it adds very little overhead for a typical user even in case of managing ownership change. During a possible ownership change, if the trusted device is available, chownIoT verifies the trusted device automatically without any user interaction. For instance, a user wants to change the location of his smart home device from his apartment to his summer cottage. The user just plugs-out the device from his apartment and plugs-in in his summer cottage. If he/she has the trusted device with him in the summer cottage, then once powered on, chownIoT detects an ownership change and finds the trusted device nearby. The device authenticates the trusted device automatically. The user does not have to do anything apart from plugging in the device in the power socket or turning it on if it is a battery powered device. Furthermore, if the user does not have the trusted device with him in such a situation then also he/she will be automatically prompted to enter his/her authentication key on a new control device which also improves the usability and intuitiveness of the overall system. Apart from this, for each new context of a particular owner the authentication is required only once. Once the owner authenticates a context, if in future he/she again changes back to previously authenticated context, the system automatically detects it as known context.
CHAPTER 6. EVALUATION

Profile Management: The profile management feature of chownIoT allows to retrieve any existing profile at any time. This feature also allows secure transfer of temporary ownership change without losing previous owner data. For instance, if an owner wants to lend his device temporarily, he/she just needs to give the device to the new owner. When the new owner powers on the device in a new context, the device identifies the change in context and encrypts the previous owner profile. The new owner creates a new profile and starts using the device. Once the new owner returns the device, the previous owner retrieves his/her profile and gets the previous data back using the profile retrieval feature without having to reset up anything. Apart from lending, this feature also allows to retrieve existing profile in case if an owner gets back his lost device. Thus, this profile management feature enhances the usability of the overall system.

Limitations: One limitation of chownIoT in terms of usability is false detection of ownership change. If the detection mechanism falsely detects an ownership change and if the trusted device is not nearby, the profile gets encrypted. To retrieve the profile, the owner needs to perform the authentication steps for each false detection which can degrade the usability of chownIoT. Apart from this, a rigorous evaluation of usability requires a comprehensive user study where random users are exposed to the system to test it and provide feedback. However, due to the time constraints we have not been able to perform the user study and hence, we do not have a proper usability evaluation.
Chapter 7

Related Work

Recently, several research works have been done to secure and ensure smooth operation of IoT devices. Our proposed solution is closely related to context aware security, smart home device privacy and access control of IoT devices. Thus, we divide the related works into these three major categories. In Section 7.1, we discuss about research works related to context aware security followed by Section 7.2 where we discuss about smart home device privacy. Finally, in Section 7.3, we discuss about the works related to access control of IoT devices.

7.1 Context Aware Security

With the emergence of ubiquitous computing, the term context aware service came into use. Leveraging context to provide better service is the main idea of context aware service. The increasing number of sensors on devices has been one of the main driving force behind it. Several security solutions [27, 59] have also started using context to provide better security. Some of the recent works on context aware security are discussed below.

Context aware access control has been an interesting research topic in recent years. Several works have been done to integrate context for providing better access control techniques. Ren et.al. [49] proposed an enhanced context-aware adaptive access control system (EasyGuard) in Android which provides adaptive access control through automatic detection of specific context according to pre-configured policies. In addition, Zhang et.al. [63] proposed a context aware access control mechanism which dynamically grants and adapts permissions for accessing resources or services to users according to current context. Apart from this, Kapsalisa et.al. [33] proposed a context aware access control architecture for e-services and Jih et.al. [31]
proposed a similar mechanism for pervasive health-care. Context in these works is mostly considered as the device location, time, system resources, network state, network security configuration, etc. Similarly, Miettinen et al. [39] introduced a context aware access control framework ConXsense for smartphones which makes use of context profiling and context classification for taking access control decisions. The authors also presented the real world application of the proposed framework to protect against device misuse and defend against sensory malware. The ConXsense framework uses a context model for extracting context features related to familiarity of the context and persons in the context. The model uses GPS and Wi-Fi for modelling familiarity of the context and Bluetooth for modelling persons in the context.

Adaptable security has been seen as a solution for finding balance between security level and overall performance. The mechanism takes context awareness as an additional parameter that can influence the security evaluation outcome under consideration. Zurek et al. [65] proposed a proposition logic based context aware security adjusting model including a formal representation. The authors discussed a context consistency analysis mechanism for investigating whether the provided contextual data is obtained and interpreted correctly. In addition, for illustrating the proposed approach the authors presented a case study of context aware security management for mobile devices.

Besides access control, in another work, Miettinen et al. [38] proposed a new approach for secure zero interaction pairing intended for IoT and wearable devices, which uses context to identify the pairing devices. The identification of the correct devices in done by measuring sustained co-presence over time in the proposed scheme. For co-presence detection, the fingerprint of the ambient context of the devices are calculated using the information provided by the commonly available sensor modalities, such as ambient noise and luminosity. The authors also introduced a robust and inexpensive approach for fingerprinting context over time.

Utilizing the randomness provided by context or physical environment for building security solutions has gained significant attention recently. The main idea is to use the randomness provided by physical quantities, such as sensor readings or wireless channel profiles and develop better security solutions. Zenger et al. [62] analyzed the security as well as the IoT suitability of correlated observations through mutual information extraction in general and also investigated corresponding real world challenges. The major contributions of the paper are in-depth security analysis of key agreement protocol based on correlated observations and, comparison of well known cryptographic primitives and physical layer approaches for key agreement in IoT perspective.
Apart from proposing new context aware security solutions, some studies also focused on finding vulnerabilities in already existing solutions. To overcome the relay attacks against challenge response authentication mechanism, distance bounding or contextual co-presence detection mechanisms have been proposed. Some proposed works demonstrate feasibility of using sensor modalities, such as audio, radio (Wi-Fi, Bluetooth and GPS) and environment sensors for contextual co-presence detection. The research works focusing contextual co-presence detection in majority cases assume that it is very hard to manipulate the contextual environment. In [53], Shrestha et al. however showed that context manipulation attacks are feasible against variety of sensor modalities by manipulating different sensor readings using low-cost, off-the-shelf equipment. The authors also showed that the ability of context manipulation provides significant advantage in defeating contextual co-presence detection. Based on machine learning techniques, they also investigate two sensor fusion approaches and show that both are vulnerable to context manipulation attack. In chownIoT, we use context to detect ownership change. The entire system relies on the correct detection of ownership change. Thus, if an adversary can manipulate the context and force a device to believe that the context has not changed when it has actually changed then the system does not trigger any protection mechanism. Thus, the proposed solution can be vulnerable to context manipulation.

All of these works leverage context either to take access control, pairing or key agreement decisions and eventually improve the security. Some works also showed that it is possible to manipulate the context and this manipulation can raise threats against context aware security solutions. In our work, we leverage context of a smart home device for detecting the ownership change. None of the works prior us have leveraged context for such purpose. The use of context for detecting ownership change provides the capability to smart home device to take necessary security measures for privacy protection automatically.

### 7.2 Smart Home Device Privacy

The enormous growth of smart home IoT devices is making consumer life more convenient. However, it is also introducing novel challenges for preserving privacy inside the smart home. Thus, studies are focusing on mitigating the privacy issues of smart home devices. Some of the works related to smart home device privacy are discussed below:

Apthorpe et.al. [8], examined different smart home IoT devices and found out that even with encrypted traffic, the network traffic rates of the devices
can reveal potentially sensitive user interactions. The authors developed a strategy for replicating passive network observer capabilities to infer consumer behaviour from rates of IoT device traffic. The strategy depends on the nature of IoT devices for mapping traffic patterns to device states. They used four commercially available smart home devices for the experiment. By observing the traffic patterns of these devices, the authors showed that it is possible to infer user’s sleeping patterns, user’s interaction with camera, when camera detects motion in its field of vision, when an appliance is turned on and off and also when a user is interacting with an intelligent personal assistant. The results indicated that there is need for technological solution to protect IoT device owner privacy. In another work [7], the same set of authors proposed mechanisms for preventing network observers from inferring consumers’ private in-home behaviors. They proposed four different strategies that the device manufacturers and third parties can take for protecting side-channel traffic rate privacy threats. The proposed strategies are: blocking traffic, concealing DNS, tunneling traffic and shaping and injecting traffic in the network.

The main difference between these and our proposed work is that these works only focus on privacy issues of smart home devices related to passive network observation. The privacy issues related to ownership of smart home IoT devices are not addressed which is the main focus of our work.

### 7.3 Access Control of IoT Devices

Due to the limited computational capabilities, IoT devices require light weight yet secure authentication and access control mechanisms. Several research works have proposed authentication protocols specifically designed for IoT environment. In addition, some of these proposed works have focused on designing authentication mechanism that is applicable throughout the life cycle of IoT devices. Apart from these, some of the works also talked about authentication requirements during ownership change of IoT devices. A few of these research works are discussed below:

With the increased demand of IoT devices, the need for stronger authentication and access control mechanisms has also increased to a great extent. The legacy mechanisms lack in meeting the requirements of IoT. IoT demands a holistic authentication mechanism throughout the device life-cycle. Yavuz [61] proposed a cryptographic authentication scheme called Efficient and Tiny Authentication (ETA) intended for resource constrained devices. The author used digital signature scheme with smaller key sizes for authenticating the devices. Mora-Afonso et.al [42] proposed a scheme for authenticating com-
munication between smart devices in domestic environment. The authors used Identity-Based Cryptography (IBC) to authenticate communications between smart wireless devices in the home. Apart from this, Miettinen et al. [40] proposed a system which is capable of automatically identifying the types of devices being connected to an IoT network and enables enforcement rules for constraining the communication access rights of vulnerable devices to minimize the damage resulting from their compromise. Neto et al. [45] presented a suite of protocols termed as Authentication of Things (AoT) which incorporates authentication and access control during the IoT device lifespan. AoT is primarily based on Identity and Attribute-Based Cryptography to cryptographically enforce Attribute-Based Access Control (ABAC). The protocol enables seamless wireless interoperability of new and guest devices. Moreover, AoT also allows transfer of device ownership in a secure manner. For ownership change, AoT reassigns the access key to the new owner and deletes the previous owner keys from the IoT device. The key management is performed using a centralized local server and cloud. The difference of AoT with our proposed system is that AoT is more focused on cryptographic aspect, such as key generation, distribution and revocation for authentication and access control of IoT devices during their life cycle. The protocol does not say anything about protecting owner data, recovery of data and profile management for handling multiple owners data.

The ubiquitous computing devices can be borrowed/lent temporarily or sold permanently, which can raise privacy concerns, as they can contain personal user data. Several works have focused on protecting privacy sensitive user information handled by these devices. Tam et al [57] and Bohn [14] proposed ownership transfer mechanisms for smart devices in their individual works for securely transferring ownership. Similarly, Pradeep et al. [47] also proposed a concept of ownership authentication transfer for securely handling the ownership transfer of a device to a new owner. In the proposed protocol, the previous owner willing to sell a device starts the ownership transfer procedure. After ownership transfer process, the previous owner is not able to use the device. According to the proposed protocol, if the ownership transfer is not carried out properly then neither of the owner is able to use the device. This feature also handles scenarios when the device has been stolen or lost and avoids impersonation attack. The main difference between ownership authentication transfer protocol and our proposed system is that in ownership authentication transfer the seller has to initiate the transfer process, there is no notion of detecting ownership change automatically. In addition, the protocol requires a central key server for key management which may not be feasible in case of smart homes. The protocol does not also mention any data protection mechanism in particular for protecting the privacy sensitive
data.

The research works related to access control of IoT devices focus either on cryptographic aspect of authentication and access control or on secure ownership transfer mechanisms for smart devices. None of these works focus on protecting privacy sensitive data of the smart devices. Our proposed system is primarily focused on protecting the user privacy sensitive data handled by the smart devices. Moreover, the works on secure ownership transfer do not use any automatic ownership change detection mechanism and mostly rely on the previous owner to initiate the ownership transfer process. The automatic ownership change detection mechanism proposed in our work allows the device to take necessary security measures without depending on any user intervention which improves the overall security of the system.
Chapter 8

Conclusions

In this work, we presented a technique to improve the privacy of smart home IoT devices during ownership change. We now summarize our contributions and draw some paths for future work.

8.1 Summary of Contributions

**Threat Model:** We present a threat model for ownership change of smart home devices. The threat model identifies the potential threats that can arise due to ownership change of such devices. In addition, it outlines the potential attackers, their capabilities and goals, and the possible attack surface. The threat model helps to understand the requirements of a potential solution for mitigating the identified threats.

**Privacy Enhancement Protocol:** We introduce a protocol to enhance privacy of smart home devices during ownership change. The protocol preserves the privacy of both new and previous owner by encrypting the profile data in case of ownership change. It does not depend on any specific hardware requirements. Thus, it can be applied for most smart home devices currently available in the market.

**Owner Profile Management:** We also present a scheme to manage owner profiles of the smart home devices. The owner profile management scheme helps to protect the owner privacy sensitive data. The profiles are password protected and only the owner with correct credentials can access or retrieve his/her profile data. The scheme also enables an easy and secure transfer of ownership which can be temporary or permanent. Specially, in case of temporary change, the profile protection and retrieval process reduces the overhead of the previous owner and also enhances the overall security.
CHAPTER 8. CONCLUSIONS

**Simple Ownership Change Detection Technique:** We present an idea of leveraging context for detecting ownership change. In addition, we also present a simple technique for automatic ownership change detection based on device context. The technique introduces the concept of using Wi-Fi connection information as context for smart home devices. It defines that the change of Wi-Fi connection information indicates a possible change in ownership.

**Prototype Implementation:** Finally, we present a prototype implementation of the proposed system. The smart home device features of the proposed system are implemented using Raspberry Pi board. The control device features are implemented using Android application running on smartphone. The implementation defines different components required for realizing the system. In addition, it outlines different possibilities and difficulties of implementing the system in reality. Apart from this, it also helps understanding the practical requirements and effectiveness of the proposed system.

8.2 Future Work

**Ownership Change Detection:** One of the main requirements for a complete solution is the automatic ownership change detection. This work primarily focused on the privacy enhancement protocol. Even though we developed a simple technique for detecting ownership change for the prototype implementation, however, it can definitely be improved. In addition, the solution is not feasible for mobile devices, as this type of devices sometimes do not require Wi-Fi communication. Thus, the automatic ownership change detection can be a subject of future work. An effective ownership detection technique should involve all available sensor modalities to detect contextual change and accurately detect the ownership change. In addition, it should also be applicable for any class of devices including mobile devices. The main challenge of an ownership change detection policy based on contextual information is reducing the number of false positive and false negatives. The sensor readings can change frequently which eventually increases the rate of incorrect guessing. Moreover, the privacy enhancement protocol completely depends on the ownership change detection, thus the rate of incorrect guessing can severely impact the usability of the overall system. Therefore, developing a robust ownership change detection mechanism involving multiple sensor modalities can be a good future direction for overall improvement of the proposed work.
**Owner Authentication:** For authenticating the owner, the proposed work uses password based authentication. In addition, the encryption key is also derived from the password. However, as passwords are vulnerable to dictionary attacks, the adversary can retrieve the encryption key using offline brute force attack. We could use trusted hardware for protecting against the password guessing which can be a potential solution. Apart from using trusted hardware, designing an authentication mechanism for authenticating an owner in such a scenario can also be a subject of future work. The main challenge for designing the authentication mechanism for such scenario is that it needs to be lightweight, intuitive and also should be completely secure. In addition, the generation of encryption key should also be more secure to protect against offline brute force attacks. Such strong and intuitive authentication mechanism will improve the overall security and usability of the proposed system. Apart from this, protecting encryption key from side channel attacks, such as cold boot attack [24] can also be considered as a subject of future extension of the current work.

**Improved Profile Management:** Currently, the proposed system keeps all existing profiles on the device storage. The current owner has physical access to the device storage. Thus, retrieving any encrypted profile and applying brute force attack is always possible. One solution of this problem can be transfer of encrypted profiles to cloud. Secure transfer, retrieval and maintenance of existing profiles in cloud also can be a subject of future work. If the profiles are managed in cloud then the current owner will not have any physical access to the encrypted data which in turn will prevent possibilities of brute force attack and improve the security of the proposed system.
Bibliography


BIBLIOGRAPHY


BIBLIOGRAPHY


