COPYRIGHT AND CITATION CONSIDERATIONS FOR THIS THESIS/ DISSERTATION

- Attribution — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

- NonCommercial — You may not use the material for commercial purposes.

- ShareAlike — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.

**How to cite this thesis**

Access Control for Local Personal Smart Spaces

by

Brian Rex Greaves

submitted in fulfilment of the requirements for the degree

Magister Scientiae

in

Information Technology

for the

Faculty of Science at the

University of Johannesburg

Supervisor: Prof. M. Coetzee
Abstract

More powerful smart devices come onto the market and into the hands of consumers every year. These devices provide less reliance on fixed mediums for information and content sharing by supporting peer-to-peer connections such as Wi-Fi Direct, Bluetooth and Near Field Communication. Technologies such as these enable resource sharing that is not hindered by fixed access point range or costly Internet connections. This new-found mobility has enabled a greater degree of freedom for smart device users to share and consume resources wherever they are. New technologies not only support unrestricted content sharing, but also introduce new threats to the security of personal and corporate information resident on smart devices.

Peer-to-peer technologies do not provide a means to enable streamlined sharing for multiple files. It quickly becomes a cumbersome task to synchronise content such as files or calendars without a third-party application such as Dropbox or iCloud, which may incur additional costs to the user. To combat this limitation, smart spaces can enable the sharing of resources on-the-go, whenever other capable devices are in range.

Smart spaces provide a managed means for users to share and protect their resources. Although current smart spaces can manage user resources in a secure way, they lack the ability to allow users to define content sharing preferences and have them considered when access control is performed. Current smart spaces thus provide a greater degree of autonomy and security, but they do not support the measure of personalisation and control needed by today’s smart device users.

In order to address the contents sharing and personalisation concerns relating to smart spaces, this dissertation proposes the concept of a Local Personal Smart space to provide user-tailored services to smart devices in close proximity. This research proposes a Local Personal Smart Space framework that is geared to provide secure resource sharing by supporting resources and access control policy management to perform access control locally on a peer device without the reliance on third parties. This dissertation defines a trust- and context-based access control model capable of catering to user preferences and the security of groups of devices and files through the use of local and global policies which are combined to consider the personal preference of the device owner and the security rules set by the group owner.

To test the Local Personal Smart Space framework and its access control model, this research developed the SmartNet prototype to verify their effectiveness at providing user-tailored services and security for content resident on the devices. The SmartNet application also serves to verify the effectiveness of this research in achieving its research objectives.
Acknowledgments

First, and foremost, I would like to thank Professor Marijke Coetzee for the guidance and mentorship provided throughout the course of this research. Without your encouragement, dedication and patience, I would not have been able to reach the end of this journey. Thank you for the many, many hours spent refining the content of papers and this document.

To my family, both present and those no longer with us, I would like to dedicate this document to you. Thank you for your unwavering faith, encouragement, guidance and provision. You are my greatest blessing and I am eternally grateful for all that you have done, and continue to do, in my life.

God bless you all.
Table of Contents

Chapter 1 ................................................................................................................................................ 1
1.1 Introduction ...................................................................................................................................... 1
1.2 Important terms ................................................................................................................................ 1
1.3 Research problem ............................................................................................................................. 3
1.4 Research objectives .......................................................................................................................... 4
1.5 Research questions ........................................................................................................................... 5
  1.5.1 Resulting research outputs ........................................................................................................ 5
1.6 Research methodology ..................................................................................................................... 5
  1.6.1 Review relevant literature ......................................................................................................... 6
  1.6.2 Design a model to address research objectives ......................................................................... 6
  1.6.3 Evaluate model through the use of a prototype ....................................................................... 6
1.7 Layout of this document ................................................................................................................... 7
  1.7.1 Introduction ............................................................................................................................... 8
  1.7.2 Part 1: Literature review ............................................................................................................ 8
  1.7.3 Part 2: Model & prototype ......................................................................................................... 9
  1.7.4 Conclusion .................................................................................................................................. 9
1.8 Conclusion ......................................................................................................................................... 9

Part 1

Chapter 2
2.1 Introduction .................................................................................................................................... 11
2.2 Definitions ....................................................................................................................................... 12
  2.2.1 Smart device ............................................................................................................................ 12
  2.2.2 Smart space .............................................................................................................................. 12
  2.2.3 Personal smart space ............................................................................................................... 14
  2.2.4 Local personal smart space ...................................................................................................... 14
  2.2.5 Definition of a local personal smart space: .............................................................................. 15
2.3 Scenario ........................................................................................................................................... 16
2.4 Local personal smart space requirements ...................................................................................... 18
  2.4.1 Local personal smart space functional requirements: ............................................................. 18
  2.4.2 Local personal smart space access control requirements: ...................................................... 19
2.5 Conclusion ....................................................................................................................................... 19
Chapter 3

3.1 Introduction ................................................................................................................. 21
3.2 Access control .................................................................................................................. 22
  3.2.1 Access control concepts ................................................................................................. 24
  3.2.2 eXtensible Access Control Markup Language ................................................................. 25
  3.2.3 Access control policy propagation .................................................................................... 27
3.3 Access control models ..................................................................................................... 28
  3.3.1 Discretionary Access Control ............................................................................................. 28
  3.3.2 Mandatory Access Control ................................................................................................. 29
  3.3.3 Role-Based Access Control ................................................................................................. 30
  3.3.4 Attribute-Based Access Control ......................................................................................... 32
  3.3.5 Context-Based Access Control ........................................................................................... 34
  3.3.6 Trust-Based Access Control ............................................................................................... 36
3.4 Evaluation of access control models .................................................................................. 38
  3.4.1 DAC .................................................................................................................................. 38
  3.4.2 MAC .................................................................................................................................. 39
  3.4.3 RBAC .................................................................................................................................. 39
  3.4.4 ABAC .................................................................................................................................. 39
  3.4.5 CBAC .................................................................................................................................. 40
  3.4.6 TBAC .................................................................................................................................. 40
3.5 Formal evaluation of access control models ...................................................................... 41
  3.5.1 Findings ............................................................................................................................... 43
3.6 Conclusion .......................................................................................................................... 44

Chapter 4

4.1 Introduction ......................................................................................................................... 45
4.2 Trust ..................................................................................................................................... 45
  4.2.1 Properties of trust ............................................................................................................... 46
  4.2.2 Recommendations ............................................................................................................. 47
4.3 Context and trust in access control models ....................................................................... 47
  4.3.1 A fuzzy approach to trust based access control in the internet of things ......................... 48
  4.3.2 Context-based access control model for smart spaces ...................................................... 50
  4.3.3 TrustAC: Trust-based access control for pervasive devices ............................................. 52
4.4 Access control model evaluation .................................................................................................... 53
4.5 Revisiting access control requirements .......................................................................................... 55
4.6 Conclusion ....................................................................................................................................... 56

Chapter 5
5.1 Introduction .................................................................................................................................... 57
5.2 Personal Smart Spaces ..................................................................................................................... 57
  5.2.1 The architecture of Personal Smart Spaces ............................................................................. 58
  5.2.2 Can a LPSS application be implemented using the PSS framework? ....................................... 59
5.3 Android ........................................................................................................................................... 60
  5.3.1 The Android system structure ................................................................................................. 60
  5.3.2 Android security model ............................................................................................................ 62
  5.3.3 Android access control examples ............................................................................................ 63
5.4 Evaluating Android access control .................................................................................................. 65
5.5 State-of-the-art research on Android access control ..................................................................... 66
  5.5.1 Android file access control system .......................................................................................... 66
  5.5.2 CRePE ....................................................................................................................................... 68
5.6 Conclusion ....................................................................................................................................... 70

Part 2
Chapter 6
6.1 Introduction .................................................................................................................................... 72
6.2 The local personal smart space framework .................................................................................... 73
  6.2.1 The Device-specific application ................................................................................................. 74
  6.2.2 The Device-independent Layer ................................................................................................. 75
  6.2.3 The Policy Manager .................................................................................................................. 76
  6.2.4 The Access Control Manager ................................................................................................... 78
6.3 Conclusion ....................................................................................................................................... 80

Chapter 7
7.1 Introduction .................................................................................................................................... 82
7.2 LPSS access control model goal ...................................................................................................... 83
7.3 LPSS access control model .............................................................................................................. 83
  7.3.1 LPSS access control model fundamental concepts ................................................................. 83
List of Figures

Figure 1.1: Dissertation layout ................................................................................................................ 7
Figure 2.1: Smart spaces interacting with each other and the external world ........................................... 13
Figure 2.2: Personal smart spaces ......................................................................................................... 14
Figure 2.3: A Local Personal Smart Space ........................................................................................... 16
Figure 2.4: John's LPSS group .............................................................................................................. 17
Figure 2.5: John and his family’s LPSS group ..................................................................................... 17
Figure 3.1: The subject-action-object relationship of access control ................................................... 22
Figure 3.2: Centralised access control enforcement ............................................................................. 23
Figure 3.3: Distributed access control enforcement ............................................................................ 24
Figure 3.4: XACML policy section ...................................................................................................... 27
Figure 3.5: A user with multiple roles .................................................................................................. 31
Figure 3.6: A role assigned to multiple users ........................................................................................ 31
Figure 3.7: Relationship between user, roles and their associated permissions ..................................... 32
Figure 3.8: Authorisation Architecture in the context of the system (Hu et al., 2014) .............................. 33
Figure 3.9: Context manager ............................................................................................................... 36
Figure 4.1: The three-layer structure of the FTBAC framework .......................................................... 49
Figure 4.2: The cyber-physical environment ...................................................................................... 50
Figure 4.3: TrustAC model .................................................................................................................. 52
Figure 5.1: PSS Architecture ............................................................................................................. 58
Figure 5.2: Android Platform Structure .............................................................................................. 61
Figure 5.3: ICC using permissions and UIDs ...................................................................................... 64
Figure 5.4: Android File Access Control System architecture ............................................................ 67
Figure 5.5: XML-based policies for Android ...................................................................................... 68
Figure 5.6: CRePE architecture ........................................................................................................... 69
Figure 6.1: Access control framework for Local Personal Smart Spaces ............................................. 74
Figure 6.2: Resource sharing with object groups ............................................................................... 75
Figure 6.3: Access control components of the LPSS framework ....................................................... 76
Figure 6.4: Local and global groups .................................................................................................. 77
Figure 6.5: Policy propagation in a group .......................................................................................... 78
Figure 6.6: Architecture of the ACM .................................................................................................. 79
Figure 7.1: Local policy for group_J .................................................................................................... 87
Figure 7.2: Global policy for group_home ........................................................................................... 88
Figure 7.3: Revised global policy for group_home using trust ............................................................. 91
Figure 8.1: Initial account setup and login ........................................................................................... 97
List of Tables

Table 3.1: Access Control Matrix......................................................................................................... 29
Table 3.2: Example Security levels of Mandatory Access Control ...................................................... 30
Table 3.3: Examples of different types of context ................................................................................ 35
Table 3.4: Example Trust Levels .......................................................................................................... 37
Table 3.5: Summary of access control models...................................................................................... 43
Table 4.1: Tabular summary of access control model comparison....................................................... 54
Chapter 1
Introduction

1.1 Introduction
Smart devices and mobile computing have been around for some time. As more smart devices find themselves in the hands of consumers, there is less reliance on fixed access points to share resources and gain access to information. Users need access to files and services when they want, wherever they are. As users move away from the use of fixed access-points to a more decentralised approach to computing, there are new concerns to be identified and addressed (Mustafee & Bessis, 2015).

The smart devices of today are not static objects that are utilised for a singular purpose. They are multi-faceted tools that can be adapted to the preferences of the user. Applications can be installed and changed to meet the requirements of the user for many different purposes. As the users of these smart devices gain more and more control over what they can do and change, they demand more flexibility in other aspects of their daily lives. Modern technologies attempt to keep up with the ever-evolving demands of the user, but networking technologies are falling behind (Maheswaran et al., 2015).

Wireless networking technologies such as Bluetooth (Bluetooth.org, 2015), NFC (Nearfieldcommunication.org, 2015) and Wi-Fi Direct (Wi-Fi Alliance, 2015) can be used to enable unhindered sharing of content such as files, information and resources between devices when in close proximity. Technologies such as these are evolving at an unprecedented rate to keep up with consumer demand, but the mechanisms which secure them are also falling behind (Ovaska & Evesti, 2011) (Mustafee & Bessis, 2015).

Users do not want to be tied-down to physical locations when they share their content and resources. That is why modern smart devices have been built to embrace decentralised communication through a vast array of networking technologies. These technologies enable smart device users to share their content on-the-go, without the need to stop and connect to a fixed medium. These technologies, however, are not usable, secure or personal enough for the modern-day user (Ratner et al., 2001).

1.2 Important terms
Before identifying the problem domain for this research, it is imperative to define a number of important terms closely related to this research.
a) Access control
Access control is a security service responsible for limiting access to resources for legitimate users of a given system (Sandhu & Samarati, 1994). Access control enables the protection of valuable resources in a system from those who would perform unwarranted actions to misuse them.

b) Access control policy combination
Access control policy combination is performed by a Policy Decision Point during access control enforcement when there are a number of access control rules relating to a common object. The policy combining algorithm is responsible for rendering a verdict when rules are in conflict with one another (Li et al., 2009).

c) Peer-to-peer
Peer-to-peer (p2p) is a networking concept where hierarchy does not exists and all nodes are equal (Pandurangan et al., 2003). Each node is responsible for managing its own interactions with other nodes. In a p2p environment a node is therefore both a server and client when needed (Pandurangan et al., 2003). The server aspect is responsible for managing connections and the receiving of information while the client aspect is responsible for information sending, both of which happen concurrently on each device.

d) Wi-Fi Direct
Wi-Fi Direct is a peer-to-peer networking technology similar to Bluetooth (Bluetooth.org, 2015), which enables devices to connect to each other using Wi-Fi technologies, without the need for a dedicated access point (Wi-Fi Alliance, 2015). Wi-Fi Direct allows peer devices to negotiate who takes on the role of the access point to manage connections.

e) Smart space
Smart Spaces are ecosystems formed by devices that connect to one another to provide and consume information and resources without the reliance on any one explicit connection medium (Ovaska & Evesti, 2011). Smart spaces are built upon a common set of rules on how content within the space is managed to provide users with up-to-date information and access to resources when needed (Cook & Das, 2007)(Oliver, 2009).

f) Local Personal Smart Spaces
Local Personal Smart Spaces (LPSS) are a type of smart space defined by this research. LPSSs are and built upon the smart space paradigm but have both a local and personal dimension to address the problem domain specified in this document.
The personal dimension of the LPSS caters to the personal preference of users. Users may wish to connect and share information between devices but require a certain degree of personal preference in the manner in which the LPSS operates and access control performed (Papadopoulou et al., 2012).

The local dimension of the LPSS defines the boundaries of the space. As the user carries their personal devices with them, the bounds of the smart space are always centred around the user. Wherever the user goes, the LPSS goes.

The next section introduces the reader to the problem domain to be addressed by this research.

1.3 Research problem

As more consumers come into the possession of smart technologies, they become more reliant on them for their everyday lives. The smart devices that users carry become repositories for their personal or corporate information and resources. These devices are usually on the person of their owner and enable immediate access to resources wherever the user is. In the modern world, users are moving away from the reliance on fixed access points which ultimately hinder their mobility and ability to access information or resources wherever they are (Ratner et al., 2001).

The abundance of inherently mobile smart devices, have brought about the need for a direct means for users to share resources between their own devices and those belonging to others. Hence, the move away from access-point-oriented communication to a peer-to-peer approach when on-the-go (Chlamtac & Faragó, 1999). Technological advancements have been made in areas that facilitate direct communication between smart devices, without the reliance on access points or third-party service providers which cost money to utilise (Lehr & McKnight, 2003). The direct control that users have over these direct means, enable users to make their own decisions about what to share and when. Thus, users are given control based on their own preference. These technologies, however, are cumbersome and not easy to use.

In order to facilitate direct communication between devices that does not require user micromanagement while still having a degree of personal preference, smart spaces (Ovaska & Evesti, 2011) were developed. Smart spaces and the Local Personal Smart Spaces defined by this research enable smart devices to form groups and act in autonomy to provide and consume resources as users require. Smart spaces are a managed way of sharing resources with other smart devices without the need to directly manage each and every inter-device transaction. Smart spaces, however, also suffer from the lagging state of security (Sharifi et al., 2009)(Gilman et al., 2013).

Due to the decentralised nature of smart spaces and LPSSs, there are no central policy stores managed by competent administrators to ensure resources are only provided to those who are legitimate users.
of the system. Furthermore, these smart spaces do not yet support personal preference to the same degree that users are accustomed to having on their smart devices.

When considering the secure sharing of resources in decentralised environments, smart spaces can provide a good foundation. However, it is unknown how they can be used to securely enable users to share content between their devices and the devices of others while enabling users to have their personal preference considered in the form of a set of rules which govern how content is shared. Therefore, the problem to be addressed by this research is defined by the lack of personal preference and policy management in peer-to-peer smart space environment.

To achieve the objectives of this research a number of goals must first be specified. The following section addresses the research objectives for this document.

1.4 Research objectives

The primary objective of this research is to investigate how to securely share information and resources in Local Personal Smart Spaces using the personal preferences of users, where smart devices are connected to each other with access-point less mediums such as WiFi-Direct.

Given the nature of these environments, and the inherent lack of third-party administrators or security providers, this research needs to investigate how access control can be applied to ensure fine-grained control over personal information and resources so that they can only be shared with authorised parties in a flexible manner.

The secondary objectives of this research are twofold. Firstly, the definition and management of an access control policy needs to be considered. This objective aims to determine how access control policies are defined for peer-to-peer smart spaces, where no third-party administrator or a centralised policy store exists. Secondly, the personal preference of the user must be addressed as users need a greater degree of control over the devices and resources that they own.

In order to achieve all of the above, this research aims to culminate in the creation of a system that is capable of supporting secure information and resource sharing between peer-to-peer smart devices while taking into consideration the personal preferences of users.

For this research to achieve its research objectives, a number of questions must be answered. The next section identifies the questions that will guide this research toward addressing the problems identified.
1.5 Research questions

Based upon the objectives stipulated in the previous section, this research aims to primarily address the following question:

*Can fine-grained access control be defined to support resource sharing between smart devices in a Local Personal Smart Space, by taking into account user preferences?*

In order to find an answer to this question, three questions are derived to be addressed throughout the remainder of this document. The questions are defined as follows:

1. What are smart space environments and how can they be used to share information and resources between peer-to-peer smart devices?
2. How can user information and resources securely be shared in Local Personal Smart Spaces so that they are not misused or compromised?
3. How can the personal preferences of users be considered when securely sharing Local Personal Smart Spaces resources, without compromising smart space security?

The three questions listed above serve to narrow the scope and guide the course of this research.

1.5.1 Resulting research outputs

During the course of this research, the findings were peer reviewed and presented in the form of the following publication:


The published paper is included on the disk accompanying this dissertation, along with its full abstract included in Appendix B of this document.

1.6 Research methodology

To address the objectives of this research, the questions posed in the section above must be further broken down over the course of this document to gain an understanding of the problem domain and provide the basis for a definitive solution to the problem presented in Section 1.3. In order to achieve this, a scientific and well-accepted approach must be followed to ensure the validity of the results found during the course of this research.

To adequately address the problem presented in Section 1.3, this research follows an approach designed by Olivier (2009). For this research to achieve the research objectives laid out in Section 1.4
and answer the resultant research questions posed in Section 1.5 the following actions are undertaken throughout the course of this document:

- Review relevant literature
- Design a model to address objectives
- Evaluate the model through the development of a prototype

Each of the above are expanded upon in the three sections that follow.

1.6.1 Review relevant literature

In the approach defined by Olivier (2009), the researcher must take a methodical approach to investigate the problem domain and all concepts closely related to it. This is done to gain a solid understanding of all concepts that contribute to every facet of the problem and provide the basis for a complete solution.

The literature review is covered by chapters 2 – 5. This research explores access control and smart spaces with evaluations done for the closely-related topics of context, trust and policy and how they contribute to access control in current smart space environments to aid in the next step.

1.6.2 Design a model to address research objectives

The second step in the approach defined by Olivier (2009) is to build upon the relevant literature by the creation of a model designed to address the research objectives.

The model defined by this research in chapters 6 – 7 addresses the functional and access control concerns which arose in the last step. The goal of the model is to address the research problem by achieving the goals of this research in the form of a LPSS framework which focuses on security, access control and policy.

1.6.3 Evaluate model through the use of a prototype

The final stage is to evaluate the designed model through the creation of a prototype, as described in chapter 8. In Olivier's (2009) approach, it is essential to focus only on the essential aspects of the system being created. As the ultimate goal of this research is to create a system that is capable of providing secure access to resources while taking into account user preference and still being useable, it is required to address topics closely related to access control and current smart space technology. Furthermore, it is imperative that these concepts are used as a solid foundation for the creation and testing of the resultant model in this prototype.

Each of the objectives of Section 1.4 were presented to the user in sequence to better guide the course of this research so that each topic builds on top of the previous naturally. Therefore, the structure of this document was designed to reflect this by addressing each of the research objectives sequentially
as the research unfolds. To achieve this, this document is divided into four distinct parts with a number of chapters in each which address the research objectives and questions. Figure 1.1 illustrates the layout of this document and the flow of concepts from one chapter to the next.

The research methodology undertaken by this research follows the approach suggested by Olivier (2009). Each part of this research addresses topics relevant to the objectives in order to culminate in a clear and well-researched outcome. The following section breaks down each of the four parts of this research which follow this research methodology.

### 1.7 Layout of this document

In order to follow the research methodology suggested by Olivier (2009) this document is structured in such a way that each section is built upon the sections that followed so that at the end the outcome and the reasoning behind it is clear. This section presents the structure of this document. This document comprises of four sections as illustrated in Figure 1.1.

![Figure 1.1: Dissertation layout](image)

This document is broken down into four sections namely; *introduction, part one, part two and conclusion*. Each of these are discussed in the following sections.
1.7.1 Introduction

This section of the dissertation introduces the reader to the nature of the problems to be addressed, the objectives required to address them and the questions that must be satisfied to achieve the objectives. This section also serves to introduce the reader to the research methodology used to structure the research and this resultant document.

1.7.2 Part 1: Literature review

Part 1 of this research focuses on the review of relevant literature to form the basis for Part 2. This section is broken down into four chapters which review current, relevant literature on topics closely related to the problem domain. An overview of each chapter is presented in the sections that follow.

a) Chapter 2

Chapter 2 introduces the reader to the concepts of a smart space and introduces to topic of a Local Personal Smart Space. This chapter presents a scenario involving the mobile devices of a family in order to provide a basis for the research to follow. The scenario also serves as a point of reference throughout this document and as a practical means to illustrate important concepts as a basis for examples.

b) Chapter 3

Chapter 3 explores the most popular and well-documented access control models currently in use and evaluates them against the scenario of Chapter 2. The evaluation serves to determine the access control traits most suitable for use in a smart space environment to ensure secure resource sharing. Furthermore, it serves to highlight the way to move forward when considering models for evaluation in the state-of-the-art.

c) Chapter 4

Chapter 4 explores three current access control models directly related to smart spaces. The state-of-the-art in smart space access control is evaluated and a number of key concepts are identified for further research in the chapters to follow. This chapter serves to validate the need for this research in the specific field of access control, required by the research objectives.

c) Chapter 5

Chapter 5 serves two purposes namely; the exploration of personal preference in smart spaces as well as the ability of smart space components to be implemented in the Android operating system. First, it explores the architecture of Personal Smart Spaces to provide some insight into the degree of personal preference that is already provided by smart space environments. The second purpose of this chapter is to explore the structure of the Android framework and the access control supported by it as a basis for the prototype of this research to implement key smart space functionality.
1.7.3 Part 2: Model & prototype

Part 2 of this research addresses the second and third points of the research methodology suggested by Olivier (2009), by presenting the culmination of this research in the form of a model which is evaluated through the use of a prototype. The following sections discuss this section's constituent chapters.

a) Chapter 6

Chapter 6 introduces the reader to the Local Personal Smart Space framework used to securely share resources between smart devices using peer-to-peer networking technologies. The model focuses on the definition of access control policies via the Policy Manager and the use of personal preference in the decisions made by the Access Control Manager.

b) Chapter 7

Chapter 7 builds upon the model of Chapter 6 by defining the access control model used by the Local Personal Smart Space framework's access control manager to provide robust and flexible access control to users of the system.

c) Chapter 8

Chapter 8 evaluates the model presented in Chapter 6 and 7 by documenting the prototype implemented in Android to provide Local Personal Smart Space functionality. This chapter also evaluates the successes, failures and future work for the prototype.

1.7.4 Conclusion

This chapter revisits the research objectives and questions and evaluates the effectiveness of this research in achieving the objectives by answering the research questions. This chapter also serves to evaluate any shortfalls encountered during the course of this research.

1.8 Conclusion

This chapter has introduced the problem domain for this research. A number of research objectives were identified and a number of research questions were extracted to facilitate the research process and provide a means for evaluating the outcome of this research. Next, the research methodology undertaken was detailed and the layout of this document was presented. The next chapter starts Part 1 of this dissertation.
Part 1
Literature Review
Chapter 2
Local Personal Smart Spaces

2.1 Introduction
With large numbers of mobile smart devices steadily on the rise, it was only a matter of time before the move was made away from communication supported by networked access points, to device-to-device-centric resource sharing. Today, the abundance of smart devices makes it necessary to share information directly between devices, using access-point-less networks (Serrano-Alvarado et al., 2004). Users need to be provided with intelligent environments where services and resources on their devices are managed with minimal intervention. Such developments can bridge the gap between fixed networks and mobile pervasive systems by providing access to resources from anywhere (Papadopoulou et al., 2012).

People generally possess more than one smart device, thereby creating an abundance of information. Information such as pictures and contact details, and services such as data connectivity may need to be shared between devices belonging to the same owner. To be able to share information, a group of smart devices can form a smart space to support interactions (Gilman et al., 2013). Unfortunately resources can then be exposed without the knowledge or consent of the owner of the smart device (Ovaska & Evesti, 2011)(Mustafee & Bessis, 2015).

Without any doubt, smart spaces offer new benefits, but also new threats to the users making use of them. Smart devices serve as repositories of confidential information which must be protected from unauthorized access. Thanks to the increasing strength of radio antennae, it is not always possible to visually verify the identity of the owner of another device. Thus, trust and security mechanisms become vital aspects to consider when sharing resources with others (Sharifi et al., 2009)(Ovaska & Evesti, 2011).

As smart devices have evolved, so too has the amount of sensory data they have access to. Smart devices have built-in sensors such as gyroscopes, Global Positioning Systems (GPSs) and cameras. Such technologies provide smart devices with the ability to measure the context of their operation and interactions, leading to the use of the term “context-awareness” (Serrano-Alvarado et al., 2004)(Nixon et al., 2004). Thanks to context-awareness, smart devices can change the parameters of their operation based on sensory data from their operational environment. Even with these advances, no context-aware and trust-based security mechanisms exist for peer-to-peer smart spaces (Sharifi et al., 2009) (Gilman et al., 2013).
This chapter introduces the concept of a smart space in Section 2.2 as a means to enable direct peer-to-peer resource sharing. The concept is expanded to address current developments in the use of this technology, leading to the introduction of the Local Personal Smart Space, which is the focus of this research. In order to motivate the need for this research, a local personal smart space scenario is presented in Section 2.3, which serves as a point of reference throughout this document. Section 2.4 identifies a set of functional and access control requirements for a local personal smart space using the scenario. Finally, the chapter is concluded with a brief look at the focuses of the chapters to follow.

**2.2 Definitions**

The term *smart device* has been around for some time and is used pervasively in society (Poslad, 2009). The term *smart space*, however, is less-known and not as pervasive. The following sections define a number of key concepts that are used throughout this research. Smart devices and smart spaces are defined along with the evolution of smart spaces into personal smart spaces. Finally, the term *local personal smart space* is defined by this research.

**2.2.1 Smart device**

A smart device is an electronic device capable of some level of autonomy and interactivity through connections such as Wi-Fi, 3G, and so forth (Weiser, 1991). Smart devices outnumber most other types of devices and are seen as an enabler for the Internet of Things (Poslad, 2009). Smart devices range from cellular phones, to tablets, to Personal Digital Assistants (PDAs) and many other forms of mobile computing technologies.

**2.2.2 Smart space**

A smart space enables entities such as smart devices to connect to one another through various connection mediums to share resources and information (Ovaska & Evesti, 2011). Smart space technology enables both networked and device-to-device communication between entities such as smart devices or computing systems. The flexibility of a smart space gives entities easy access to service discovery and information sharing (Korzun et al., 2013).

A smart space is an ecosystem formed by a number of connected entities, built upon a shared knowledge base (Korzun et al., 2013). The shared knowledge base varies from smart space to smart space, but typically supports the definition of rules on how content is managed (Poslad, 2009). Rules address topics such as workflow and connectivity semantics, which govern all members and their interactions within the smart space. More importantly, it is a store of information about the environment, its resources and which entities are, or have, connected to the smart space.

The goal of a smart space is to provide all entities within the ecosystem with up-to-date information and services as they are needed (Cook & Das, 2007)(Oliver, 2009). Information and services may
change over time, but information is always managed and provided when entities require it (Papadopoulou et al., 2012). Important features of a smart space include the ability for the administrator to maintain the preferences of the smart space in the form of a set of rules, as well as providing automated device and resource management. As the space is dynamic, it not only supports automated sharing, but also allows entities to automatically be presented with available services as they connect to each other.

Another notable feature of smart spaces are their geographically flexible nature. As smart spaces are dynamic and support a number of connection types, it does not matter where an entity connects from. The only limitation of the connection is the ability of the device to connect to the desired network (Weiser, 1991). Thus, the operational domain of smart spaces can vary from simple connections of devices in close proximity, to vast interconnected networks of equipment and electronics that span the globe (Balandin & Waris, 2009)(Liuha et al., 2009). In current research, Smirnov et al. (2013) document a smart space environment that supports connections from anywhere around the globe, where devices operating in the physical realm are directly manipulated through the cyber realm.

Smart spaces do not have to operate in isolation and can interact with both the external world and other compatible smart spaces (Smirnov et al., 2013)(Korzun et al., 2013). Figure 2.1 shows two smart spaces interacting with each other and the external world.

![Figure 2.1: Smart spaces interacting with each other and the external world](image)

Smart spaces interact with other smart spaces through Knowledge Processors (KPs), which are agents that provide a gateway into, or out of, a smart space. Thus, entities are not limited to consuming resources in their own smart space but can make use of Internet-based services or information that other smart spaces have on offer (Papadopoulou et al., 2012).
2.2.3 Personal smart space

Personal smart spaces (Crotty et al., 2008) are built upon the concept of the traditional smart space. A personal smart space is defined as a set of services within a dynamic space of connectable devices, where the services are owned, controlled, or administered by a single user (Roussaki, 2009). The goal of a personal smart space is to provide individually-tailored services to the user, regardless of his/her geographical location. Should a user move from one personal smart space to another, they carry their personalisation options with them, so that the next personal smart space can provide services tailored to their needs.

A personal smart space is a personal area network that is supported by a variety of devices such as smart devices, printers or wearable devices. A personal smart space can be mobile where its physical boundary moves with the personal smart space owner, or fixed within a static structure (Roussaki, 2009). When ad-hoc network technology is used, a personal smart space can interoperate with other personal smart spaces for information and service sharing as shown in Figure 2.2.

![Figure 2.2: Personal smart spaces](image)

2.2.4 Local personal smart space

This research refines the concept of a personal smart space by adding a further local dimension and introducing the concept of a local personal smart space. Today, more and more users own a number of smart devices that contain resources that need to be shared between smart devices. Such sharing is currently supported by cloud-based solutions with limited functionality and control, or by a manual and error-prone process of direct file sharing with Bluetooth. A current research challenge is how to enable users to share resources locally between smart devices that are in proximity of each other, while having fine-grained control over such sharing.

The personal and local dimensions of a local personal smart space are now defined for this research:
a) The personal dimension

Personalisation of smart space technology is nothing new (Papadopoulou et al., 2012). Personalisation adapts the behaviour of a system to meet the needs of the owner of the personal smart space to allow the system to behave differently for the owner, and to adapt to changes in the environment in a way that is satisfactory to the owner.

Similarly to personal smart spaces, this research defines the personal dimension of a local personal smart space as a set of resources and services that are available within a dynamic space of connected devices that is owned, controlled and administered by a single user. Thus, the local personal smart space is owned or managed by a single user who dictates which resources must be shared, how they are shared, and with whom.

b) The local dimension

A local personal smart space is created between any two devices owned by a single user in a direct, peer-to-peer manner using ad-hoc wireless network technologies such as Wi-Fi Direct or Bluetooth. The local dimension requires that the physical boundary of the local personal smart space moves with the user and his devices. The local personal smart space exists wherever the user and his devices are. Unlike Personal Area Networks (Bourgeois et al., 2001), where multiple devices connect to each other when they are in close proximity, a local personal smart space is a personal space that enables the creation of a group of devices that are governed by rules that have been defined by the owner of the group.

Similarly to personal smart spaces, a local personal smart space must be able to identify and interact with other local personal smart spaces in order to share resources not available in the user’s local personal smart space.

In contrast to personal smart spaces, the local personal smart space is limited to sharing resources within proximity. Although this may seem like a disadvantage, the peer-to-peer nature of a local personal smart space provides greater mobility and no reliance on fixed access points resulting in less data costs. Rules define more fine-grained control over the sharing of resources in and between groups of devices than when using cloud or other sharing solutions.

2.2.5 Definition of a local personal smart space:

A Local Personal Smart Space (LPSS) is a smart space consisting of a number of smart devices, wherein resources and services are governed and managed by a set of rules, defined by the owner of all devices (personal), and exists when the smart devices of the user are in proximity of each other (local).
Figure 2.3 illustrates the concept of a local personal smart space graphically. A local personal smart space enables smart devices, owned by a user, to connect to each other to share files and resources according to a set of personal preferences. The bounds of the local personal smart space move around with the user when they have at least one of their personal devices with them.

![Local Personal Smart Space](image)

*Figure 2.3: A Local Personal Smart Space*

From this point of the research, the researcher will use the abbreviation LPSS when referring to a local personal smart space. The next section proposes a scenario to identify research challenges for this research.

### 2.3 Scenario

In order to outline the security and access control concerns of LPSS environments, a scenario is now presented. Consider John, who is the owner and user of a number of smart devices such as phones and tablets that he uses for both work and personal purposes. Should John want to share information such as a calendar or a file between his devices, he would either have to send files directly using Bluetooth or other direct means. Alternatively, he could make use of a cloud-based application such as DropBox (DropBox, 2015) to synchronise or share files. Unfortunately, both ways of sharing files introduce difficulties. To directly share files is a cumbersome process, especially if there are multiple files to be shared. By using Dropbox, John has limited control over settings and rules governing his information and files. Additionally, data costs may be incurred if John does not have both devices connected to a free Internet connection.

Considering the LPSS technology introduced previously, it would be beneficial for John to be able to connect his devices and use mechanisms supported in a LPSS to automate and control his sharing needs. Figure 2.4 shows John's desired file-sharing scenario, with his devices all connected to his LPSS.
In Figure 2.4, John’s group of smart devices is depicted namely; a private phone, a work phone and a private tablet. Even though the devices are not necessarily made by the same manufacturer, John's personal LPSS must be able to manage the files that he shares. Furthermore, the LPSS needs rules and preferences to be set in a simple manner.

![Figure 2.4: John’s LPSS group](image)

John, however is not alone. He has a wife Mary and son Peter, who also use similar smart devices in their personal capacity. They too want to share their files and resources among their personal devices. Thus, they need to create their own LPSS groups much like John's in Figure 2.4.

Furthermore, John wants to connect all of his family's devices within a family LPSS so that all members of his family can share their files and resources over and above their personal groups. Figure 2.5 depicts the whole family connected to their own, and the family's, LPSS named *Home Group*.

![Figure 2.5: John and his family's LPSS group](image)

Mary, as depicted in the lower-central block, has a private phone and tablet and Peter, their son, only has a private phone. Even though Peter does not have multiple smart devices, his parents need to
control access to the resources on his phone. For instance, John may wish to prevent Peter from gaining access to financial documents of the family or making use of the data connection on his phone.

In addition to the groups of devices that are created to share information among personal devices, the family would like to have a LPSS for the family called *Home Group*, used to share files and resources among themselves. In order to save on data costs, John wants to specify that the smart devices can only connect to each other in close proximity using Wi-Fi Direct or Bluetooth.

John and Mary do not want anyone to be able to connect to the family's *Home Group* and access their resources. They prefer to grant access to sensitive resources to trustworthy people. Thus, access control rules need to be defined, together with the preferences of the private and family groups. Should Mark, one of the Peter's trusted friends who they know well, come over to visit, John would not mind to give him access to some of Peter's files and resources in accordance with Peter’s rules. The LPSS needs to be able to know who can be trusted and who cannot, based upon what the family knows about Mark.

John is also a member of a LPSS at work, where work documents and calendar information are shared. As these documents may be sensitive in nature, fine-grained access control is required to ensure that his family or other parties cannot access these resources. For instance, company policy may dictate that these documents may only be shared during working hours, and while he is at work.

Finally, John needs user-friendly software to create and maintain connections, groups, and rules with minimal user input after the initial setup.

### 2.4 Local personal smart space requirements

In light of the scenario presented, a set of requirements to create and maintain the Local Personal Smart Spaces for John and his family are now defined. These requirements enable this research to move forward with a clear goal of what needs to be explored. The requirements of the scenario are broken down into two distinct categories, namely; *functional* and *access control* requirements. These are defined as follows:

#### 2.4.1 Local personal smart space functional requirements:

- **A device-independent connection:** LPSS capable devices must be able to connect to each other regardless of the smart device manufacturer, using an access-point-less means.
- **Group creation:** The LPSS framework must facilitate the creation of groups, and allow smart devices to be invited to join them.
Device and group management: LPSS capable smart devices must be able to identify LPSS groups they come into range of and connect to known LPSS groups automatically.

Minimal user input: After initial setup, little user input should be required.

Easy to use: The system should not be confusing, nor difficult to use at any level.

2.4.2 Local personal smart space access control requirements:

Strict access control enforcement: Access to resources must be strictly enforced on each smart device. Policy must dictate what can and cannot be done so that unauthorized entities are not granted access to resources.

Simple access control policy management: Access control policies should be straightforward for group and device owners to implement and maintain.

Policy combination: Policy rules of different LPSSs need to be processed together, and if conflicts exist, they should be resolved appropriately.

Context: Users, device interactions and the environment must be monitored in order to define flexible access control rules.

Trust: Trust can be used to grant access to devices that are not members of a LPSS group, based upon the level of trust in them.

The functional requirements specified above motivate the focus of the LPSS software that needs to be created and directly influence the structure of the framework to be defined. The access control requirements directly affect the definition of an appropriate LPSS access control model.

2.5 Conclusion

In this chapter, this research described a number of fundamental concepts starting with a smart space from which the personal smart space evolved. In turn, this research uses the personal smart space as the foundation for the LPSS that supports peer-to-peer connections between devices with the aim of enabling localised resource sharing.

Next, John and his family’s LPSS scenario was introduced as a reference point to be used throughout this research. The notion of a LPSS was introduced as a means to enable sharing of information between devices without the effort of direct file sending, or the inflexibility of remote storage. After the scenario was described, a set of functional and access control requirements were extracted to further focus the course of this research.

To be able to address the identified requirements, the next chapters branch into a number of directions. Chapter 3 focuses on an evaluation of current access control models. These models are evaluated against the access control requirements of this chapter to determine which features would
best suit a LPSS access control model. Thereafter, Chapter 4 reviews current research in the field of trust and context-based access control for smart spaces to better guide this research. To address the functional requirements, Chapter 5 explores the architecture and design of personal smart spaces in greater detail and evaluates how to implement the LPSS over the Android operating system.
Chapter 3
Access Control Models

3.1 Introduction

Access control is a well-researched topic in information security (Sandhu & Samarati, 1994)(Bishop, 2003). Historically, physical access control has always been applied by entities such as gate guards who either open the gates to legitimate persons or deter those who seek to gain unwarranted entry. Logical access control performs a similar function to this physical concept, but exists in the cyber realm instead of the physical (Mo et al., 2012).

This research addresses the problem of protecting the resources of John and his family when they share files and pictures on their smart devices within their Local Personal Smart Space (LPSS). In order to support adequate protection for these resources, access control mechanisms need to be in place. A variety of access control models exist that all strive to achieve the same goal, namely to provide access control mechanisms to limit access to resources. However, access control models each address specific environmental requirements and are thus not similarly designed and implemented. What is consistent across all models is the fundamental principle that resources are of value and need to be protected from those who would wish to gain unwarranted access to them (Sandhu & Samarati, 1994).

This research aims to identify the most appropriate access control mechanisms to apply to the LPSS environment. Various access control models such as Mandatory Access Control (MAC) (Bell & LaPadula, 1975), Discretionary Access Control (DAC) (Vinter, 1988), Role-Based Access Control (RBAC) (Demuriian et al., 1993)(Sandhu et al., 1994), Attribute-Based Access Control (ABAC) (McCollum et al., 1990), Context-Based Access Control (CBAC) (Brezillon & Mostefaoui, 2004) and finally Trust-Based Access Control (TBAC) (Kagal et al., 2001) are all evaluated. Each model is summarised in-terms of its operation and mechanisms to evaluate which characteristics of each are most applicable to the requirements of a LPSS environment.

In order to provide a basis for this comparison, access control and related terminology is defined in Section 3.2. Section 3.2.2 continues with an exploration of XACML policies and the use of policy combination algorithms. In Section 3.3 a focus is placed on the exploration and evaluation of the aforementioned access control models to find the characteristics which could support an access control solution for this research. Section 3.4 evaluates the access control models and Section 3.5 tabulates the results and provides some insight into the findings.
The next section introduces access control as a security service and defines the important terminology used in this research.

### 3.2 Access control

Access control is the security service responsible for limiting access to resources for legitimate users of a system (Sandhu & Samarati, 1994). The role of access control is not to deny access to illegitimate users of the system, as that is the role of authentication. Authentication ensures that the user is legitimate and who they claim to be (Bakar & Haron, 2013). In order to fully understand the role of access control, one must understand its three cornerstones namely *subjects*, *objects* and *actions* as illustrated in Figure 3.1 below. Finally, the enforcement of access control is described.

![Figure 3.1: The subject-action-object relationship of access control](image)

**Figure 3.1: The subject-action-object relationship of access control**

**a) Objects**

Objects are representations of resources controlled by the system (Sandhu & Samarati, 1994). Objects are data stored in structures such as files within the system that require restricted access and protection. Physical protection of objects, wherever they are stored, is needed as it should not be possible to directly access them by bypassing access control.

**b) Subjects**

Subjects are entities within a system that initiate action (Sandhu & Samarati, 1994). Subjects are most often either users of the system, or programs which operate on behalf of a user. It may be the case that a subject is also an object. It may also be the case that a subject can create, or spawn, additional subjects in order to complete its action.

**c) Actions**

Actions form the bond between subjects and objects. Subjects are capable of initiating actions or operations on objects as specified in the authorizations implemented in the system. These authorisations are set in accordance with organisational security policy (Sandhu & Samarati, 1994)(Quing-hai & Ying, 2011). Policies define who (subject) is capable of doing what (action) on what (object).
d) Access control enforcement

In order to enforce the protection of a system and its objects, access control needs to ensure that every attempt to access a system or any of its resources is controlled. Thus, only legitimate users gain access to what they are allowed (Samarati & Vimercati, 2001). An access control system requires a set of rules that define the regulations under which access is granted and the implementation of system functionality to ensure that it happens. Access control is dependent on a number of system factors for its operation and is enforced in both centralised and decentralised system.

In centralised systems, as shown in Figure 3.2, a reference monitor acts as a mediator between the authenticated subject and the object which the subject is requesting access to. Once a request is formalised, the reference monitor consults an authorisation database to determine if the subject is authorised to make use of the requested resource. The database’s authorisation list is maintained by a security administrator in line with the organisation’s security policy. Auditing is done in an overarching manner to ensure all is operating as expected (Sandhu & Samarati, 1994).

In distributed systems, the reference monitor is extended to include Policy Enforcement Point(s) (PEPs) and a Policy Decision Point (PDP) that are considered to be trusted entities (Liu et al., 2009) (Ferraiolo et al., 2010). A distributed environment may have many PEPs who are responsible for intercepting requests and enforcing the decisions of the PDP. There exists a single Policy Decision Point (PDP) entity which can be physically located anywhere in the environment (Chbeir & Bouna, 2013). This is shown in Figure 3.3.
Should a subject request to perform an action on an object, the PEP intercepts the request and asks the PDP to make a decision. The PDP may ask a Policy Information Point (PIP) if it requires further information to make a decision. The PIP returns information about subjects, objects, actions and environmental attributes (Li et al., 2009). Finally, the PDP returns its decision to the PEP who enforces it.

In order to better understand how access control is enforced, the following section describes access control concepts.

### 3.2.1 Access control concepts

In order to evaluate access control models, it is imperative to first present the foundational concepts relating to access control in greater detail. These concepts are *access control policy, mechanisms, models and architectures* and are defined in the following sections.

**a) Access control policy**

An access control policy is the definition of a set of high-level rules by which access control may be enforced (Samarati & Vimercati, 2001). The word policy may also be used to refer to a particular instance of a policy or its implementation in a non-tangible means. Policies govern a set of rules relating subjects to objects through actions (Quing-hai & Ying, 2011). Policies specify the security requirements that are to be enforced in a particular system. More specifically, policies separate the specification of behaviours away from their actual implementation in the form of access control mechanisms (Lupu & Sloman, 1997).
b) Access control model
An access control model refers to the formal representation of an access control policy, or policies, and their workings (Samarati & Vimercati, 2001). Access control models can be grouped into three main classes, namely discretionary access control (DAC), mandatory access control (MAC) and role-based access control (RBAC) models. The formalisation of a model provides grounds for the proof of the security properties and abilities of the access control system being designed. It is arguable that if a model, which formalises the relationship between policies and their implementation as mechanisms, is secure then the system itself will be secure with respect to the security concern being tested.

c) Access control mechanism
An access control mechanism defines low-level hardware and software functions that implement the controls defined in the policy, and formally stated in the model (Samarati & Vimercati, 2001). For example, the discretionary access control model (DAC) can be implemented with an access control matrix as a mechanism. Mechanisms enforce the security requirements stated in access control policies which realise the requirements of the policy. That being said, should a policy be subject to change, so too should the mechanism realising it.

d) Access control architecture
An architecture describes the high-level security design of the major components of the access control system (Liu et al., 2009) as shown by the discussion of access control for either centralized or distributed systems. All components of the access control system integrate to provide functionality greater than the sum of its constituents, without losing the flexibility of a modular approach. The access control architecture is able to adapt as changes are introduced. Changes come in the form of updates to both policies and the mechanisms realising them, as the necessity arises.

In distributed environments, it becomes important to apply access control rules from different domains to the same resource. The next section investigates XACML to understand how access control rules can be processed together to make an access control decision.

3.2.2 eXtensible Access Control Markup Language
As XACML (Oasis, 2013) is considered the de facto standard for access control policy specification (Li et al., 2009), this section considers rules, policies and policy sets as well as the rule combining algorithms (RCAs) used in XACML.

a) Rules, Policies, Policy-sets
In the XACML language there are three elements that are defined: rules, policies and policy-sets.

The rule has three subcomponents which are its target, condition and its effect. The target states which subjects, objects and actions the rule applies to. The condition states any restrictions that apply
to the subjects, objects or actions in the target. Lastly, the effect defines what the prescribed outcome of the rule is, either permit or deny. Thus, targets can be permitted or denied from doing things based on the effect of the rule. Additionally, a rule can be deemed as applicable or not applicable if a request does or does not respectively satisfy both the target and condition (Li et al., 2009).

A policy contains a number of rules (Li et al., 2009) that each have four components: a target, a rule combining algorithm, a set of rules and obligations. The target decides if a request is applicable or not applicable to the policy. The rule combining algorithm dictates how decisions from the rule-set are combined to make a final ruling. Obligations are a set of functions that must be executed along-side the enforcement decision.

A policy-set also has four main components: a target, a policy combining algorithm, set of sub-policies and obligations. Both the target and conditions operate in the same manner as they do in a policy. A sub-policy can be a policy or a policy-set. The policy combining algorithm determines the final outcome when a set of sub-policies is executed to form a final outcome. Thus, a policy-set contains a number of policies or other policy-sets (Li et al., 2009).

b) Rule and policy combining algorithms

XACML supports distributed access control enforcement model where PEPs intercept requests and PDPs make access control decisions. The result of XACML rule processing is either, permit, deny, not applicable or indeterminate.

When there are many rules to consider at the same time, XACML must combine all applicable rules to gain a final verdict. For this purpose, XACML has four standard policy and rule combining algorithms namely; permit-overrides, deny-overrides, first-applicable and only-one-applicable. Additionally, there is provision made for empty policies.

Each of the combining algorithms are now briefly discussed (Li et al., 2009):

- **Permit-overrides**: Permit-overrides ensures that if any rule evaluates to permit, the combined decision is also permit.
- **Deny-overrides**: Deny-overrides ensures if any rule evaluates to deny, the combined decision is also deny.
- **First-applicable**: First-applicable merely returns the result of the first applicable rule outcome if there are no errors. If there are errors, an indeterminate result is returned.
- **Only-one-applicable**: Only-one-applicable returns a result is the rule is applicable. If more than one rule is applicable, indeterminate is returned.
- **Empty policies**: Should a policy or policy-set be empty then the outcome of a combination is always not applicable.
c) Example

The following incomplete section of a policy defines a rule to control users who need to read a web page. The policy applies to anyone taking any action on the resource, and contains a rule that applies to a specific action. The effect of the ReadRule is permit if the Action is read and if the Condition is met, which requires group membership to access the resource.

```xml
<Policy PolicyId="ExamplePolicy"
    RuleCombiningAlgId="permit-overrides">
    <Target>
        <Subjects>
            <AnySubject/>
        </Subjects>
        <Resources>
            http://abc.com/code/docs/developer-guide.html
        </Resources>
        <Actions>
            <AnyAction/>
        </Actions>
    </Target>
    <Rule RuleId="ReadRule" Effect="Permit">
        <Target>
            <Subjects>
                <AnySubject/>
            </Subjects>
            <Resources>
                <AnyResource/>
            </Resources>
            <Actions>
                DataType="http://www.w3.org/....">read</AttributeValue>
            </Actions>
        </Target>
        <Condition
            DataType="http://www.w3.org/2001/....">developers</AttributeValue>
    </Condition>
    </Rule>
</Policy>
```

**Figure 3.4: XACML policy section**

When there is no central point to host access control policies for decision-making it becomes important to consider how access control policies are propagated to all hosts, for example the devices in a LPSS group. Furthermore, if the LPSS owner has multiple devices and he/she makes different changes to an access control policy from different devices, the result could be a management nightmare. The next section describes a solution to this problem.

### 3.2.3 Access control policy propagation

In traditional centralised environments, policies are administrated by dedicated administrators in a centralised location. LPSS environments do not have this luxury as the environment is decentralised and subject to frequent change.

Diaz-Lopez et al. (2015), propose a means for managing distributed XACML (Oasis, 2013) policies in order to maintain consistency, especially for changing environments. They identify that a security
domain can have a level of autonomy to manage their resources locally without the intervention of another security domain. Therefore, any policies originating from another domain that are focused on local resources should be restricted in some way. Their solution proposes the use of a Master Policy Administration Point (MPAP) or root PAP as the only location where a policy can be altered within a domain. The remaining Policy Administration Points (PAPs) are solely responsible for ensuring that its policy in use is up-to-date.

Thus, a central entity can be used to administrate policies and all other connecting entities are responsible to ensure that their policies are up to date whenever they are able to connect to the MPAP.

Thus, a central entity can be used to administrate policies and all other connecting entities are responsible to ensure that their policies are up to date whenever they are able to connect to the MPAP.

Next, an evaluation of access control models is discussed.

### 3.3 Access control models

There are a number of existing access control models and associated mechanisms that can be used for LPSS access control. As the research community has not yet formally established an access control model for this environment, it needs to be investigated. The next paragraphs discuss DAC, MAC, RBAC, ABAC, CBAC and TBAC to evaluate their relevance for a LPSS access control model. Each model and its mechanisms are discussed. Thereafter, the LPSS scenario is used as the baseline against which all of the access control models are evaluated to determine which mechanisms are more suitable.

#### 3.3.1 Discretionary Access Control

Discretionary Access Control (DAC) (Vinter, 1988) is built on the premise that users of the system, or subjects, seek to gain access to resources, or objects, and their access is limited by who they are. Thus, the access of valid subjects to objects of the system are limited by their identity, or the identity of a particular group to which they belong. DAC primarily allows for certain users to discretionarily allocate their own access control authority to other valid users of the system (Quing-hai & Ying, 2011). In this way a user can grant access to resources, to users in an identical capacity as they already have. In order for a system such as DAC to operate, a means to keep track of the users, the resources and who can do what is a necessity. DAC can use access control matrices, access control lists or access control capability lists to implement access control decisions. Next, the access control matrix of DAC is discussed as an example.

**a) Access control matrix**

As early as 1971, Lampson proposed an access control matrix model in order to protect resources in an operating system (Lampson, 1974). Even though the model was later revised and refined, the basis remains the same today. An access control matrix uses a two-dimensional matrix, which stores every
subject, object and their corresponding access control privileges. Rows are occupied by subjects, the columns by objects and the remainder of the matrix by corresponding access control privileges.

**Table 3.1: Access Control Matrix**

<table>
<thead>
<tr>
<th></th>
<th>My_info.txt</th>
<th>Schedule.doc</th>
<th>Agenda.doc</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mary</td>
<td>R</td>
<td>R</td>
<td>Own R W</td>
</tr>
<tr>
<td>John</td>
<td>Own R W A D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>David</td>
<td></td>
<td>R W</td>
<td>R</td>
</tr>
</tbody>
</table>

Table 3.1 gives the access control restrictions of Mary, John and David. If John would like to make changes to one of his text documents "my\_info.txt", a lookup for John's access to *my\_info.txt* returns *Own, R, W, A\& D* indicating that as the owner of the file, John is able to perform actions on the file such as Read from, Write to, Add or Delete. Should the same lookup be done for Mary's access to the file *my\_info.txt*, Mary is only able to Read the file and nothing else. David, however, has no access to the file.

An access control matrix can be used to enforce flexible security while still embodying the secure strategy of DAC (Quing-hai & Ying, 2011). There are, however, some short-falls. Information transmission is not regulated resulting in no security guarantees. Once discretionary access is granted, no further steps are taken to limit the subject's operations including granting access to others. Finally, a sparsely-populated matrix leads to wasted space and decreased performance as it grows.

### 3.3.2 Mandatory Access Control

Mandatory Access Control (MAC) (Bell-La Padula 1975) is more secure than DAC (Quing-hai & Ying, 2011). As information flow within the DAC system is not strictly controlled, Bell-La Padula (1975) refined the military information flow security strategy to ensure secure information flow using both security classes and a lattice structure (Thomas, 1988).

**a) Security classes**

MAC requires that each subject and object is assigned a fixed security class. In MAC a subject cannot change its own security class nor that of any object. A dedicated administrator is responsible for assignment and alteration or security levels for each subject and object. In general, the system compares the subject's security level to that of the object in order to determine whether access is granted or not. The security levels of MAC, though variable, generally are in the form of the following (Quing-hai & Ying, 2011):
Table 3.2: Example Security levels of Mandatory Access Control

<table>
<thead>
<tr>
<th>Abbr.</th>
<th>Security Level Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>TS</td>
<td>Top Secret</td>
</tr>
<tr>
<td>S</td>
<td>Secret</td>
</tr>
<tr>
<td>C</td>
<td>Confidential</td>
</tr>
<tr>
<td>R</td>
<td>Restricted</td>
</tr>
<tr>
<td>U</td>
<td>Unrestricted</td>
</tr>
</tbody>
</table>

As illustrated by the security levels in Table 3.2, TS represents the highest level of security and U – the lowest. Thus, the level of security can be expressed, in a rudimentary form, such that: $TS > S > C > R > U$.

b) Lattice structure

MAC requires that information can only flow in one direction. Single-direction information flow is achieved by appointing each subject, object and security control unit with a security level and only allowing information to be passed from a less secure to an equally, or more, secure security level. Thus, the two primary rules of MAC’s lattice structure can be derived as (Dennings, 1976)(Jiang et al., 2004):

- **No-read-up**: Low security-level entities cannot read information that is above their security level.
- **No-write-down**: High-level entities cannot write, or "leak", information down to lower security-level entities.

In essence, the lattice of MAC ensures that information flows in one direction only. Thus, users of the system can only share information with those who are on their level or a level above. The lattice can make supervisor-employee interaction almost impossible as any work done by the supervisor cannot be written to lower levels for the employee to work on. Work can be shared if an employee who is at the Confidential level writes work to a higher security level such as Secret and the supervisor can read the Secret or lower security levels.

3.3.3 Role-Based Access Control

Role-Based Access Control (RBAC) (Sandhu et al., 1996) evolved from the multi-level security policy formalized by Bell and LaPadula (Ferraiolo et al., 2007). RBAC has its roots in the large-scale commercial world as it reduces the complexity and cost of security administration. A fundamental difference between RBAC and DAC is that users cannot pass access permissions on to others at their discretion. The Roles, rules and permissions of RBAC are now described.
a) Roles
The premise of the RBAC model is that a user’s access to an object is based upon their role within the organisation (Sandhu et al., 1996). Administrators designate roles such as manager that compartmentalise users into task competencies or authority positions. In RBAC it is possible for a user to have multiple roles. A user may have multiple roles and conversely a role may be assigned to multiple users (Ferraiolo et al., 2007). For example; John could be a programmer as well as a project manager. While both John and Mary could have the role of a programmer. This separation is depicted in Figure 3.5 and 3.6 below.

Figure 3.5: A user with multiple roles

![Figure 3.5](image)

Figure 3.6: A role assigned to multiple users

![Figure 3.6](image)

The concept of a session is a one-to-many mapping from users to roles. The user activates her or his assigned role(s) for the duration of a session. A session is thus an active subject.

b) Rules
As discussed by Ferraiolo & Kuhn (1992) there are three rules that form the basis of roles in order to ensure the correct operation of the RBAC model. They are as follows:

- **Role assignment**: A subject, or user, can only participate in a transaction if the subject is assigned to the required role.
- **Role authorisation**: A subject’s active role must be authorised for that particular subject. Thus a user can only use roles during their session that they have been authorised to.
- **Transaction authorisation**: A transaction can only be executed by a user if the user is authorised, according to the user’s active role permissions.
Therefore, a subject requires a role to execute a transaction. The transaction, however, requires the subject to have a particular role and that role can only be assigned, or chosen, if the subject is authorised to do so.

c) Permissions
The key feature of RBAC is that all accesses in the model are done using roles, where roles are essentially a collection of permissions (Ferraiolo et al., 2007). Thus, in order for a user to be granted access to a particular system or file, that user must be first be assigned to a role which in-turn has the required permissions assigned to it. This simplifies the management and review of access controls by managerial or administrative staff. If a user is assigned a role, it is not necessary to change their role as systems are updated or changed. The permissions assigned to the role are merely changed. The relationship between the user, roles and permissions can be generalised as depicted in Figure 3.7 below, as adapted from Ferraiolo et al. (2007).

Figure 3.7: Relationship between user, roles and their associated permissions

Roles do not change, even though users can be added and removed from the system and permissions can be changed. This ensures that users are limited to do only that which they are authorised to do via their assigned role.

3.3.4 Attribute-Based Access Control
Attribute Based Access Control (ABAC) allows subjects to be granted or denied access to objects based on the assigned attributes of the subject or object, environmental conditions, and a set of policies that are specified in terms of those attributes and conditions (Hu, et al., 2014). ABAC consists of two primary parts, namely the policy model which defines the policies used in ABAC and the architecture model which applies the policies to achieve actual protection of objects (Yuan & Tong, 2005). Before detailing these components, the attributes of ABAC are first defined.

a) Attributes
Unlike RBAC, ABAC defines permissions based on almost any form of characteristic, not just those applicable for roles (Hu et al., 2014). These permissions are known as attributes (Yuan & Tong, 2005). Attributes exist as name-value pairs The three primary attributes in a system are defined as follows:
- **Subject attributes**: Subjects are users, applications or processes that are capable of initiating actions on objects that have attributes associated with it such as unique identifiers, roles, job titles, organisations. For example, the attributes describing a student can be \{student-id, gender, degree, campus\}.

- **Object attributes**: An object are a resource in the form of a data structure, component of a system or a web service. Objects are defined by any number of attributes that are extracted from the resource’s metadata such as \{size, file_type, date-of-creation\}.

- **Environmental attributes**: Attributes that describe the context of the environment such as operational, situational or technical attributes such as \{current-date, time, security-level\}.

The ability to treat both the role and identity of the subject as characteristics allows ABAC to encompass the functionality of the previously discussed RBAC (Yuan & Tong, 2005). In addition, ACBAC’s ability to take into account object attributes such as the sensitivity of stored information allows for it to also support MAC (Yuan & Tong, 2005).

**b) Authorisation architecture model**

The Authorisation Architecture of ABAC is responsible for deciding whether a subject is granted or denied access to an object. A typical Authorisation Architecture consists of four main components (Yuan & Tong, 2005), depicted in Figure 3.8 below.

![Figure 3.8: Authorisation Architecture in the context of the system (Hu et al., 2014)](image)

Each of the components of the authorisation architecture are defined as follows:

- **Attribute Authorities (AA)**: AA are primarily responsible for the creation and management of attributes for subjects, resources and the environment respectively. Irrespective of where the attributes are stored, the AA is responsible for discovering,
provisioning and binding attributes to entities - be it subjects, resources or the environment.

- **Policy Enforcement Point (PEP):** The PEP is responsible for the enforcement of authorisation decisions, whether it is to allow or deny access to a resource.

- **Policy Decision Point (PDP):** The PDP reviews any applicable polices relating to the subject, resource and environment to make a decision. Should any attribute referenced in the request not be present, it is the duty of the PDP to contact the appropriate AA to attain the corresponding attribute values.

- **Policy Authority (PA):** The PA is responsible for the creation and maintenance of access control policies.

c) **ABAC policy formulation**

In the ABAC Policy Model the formulation of policies can be done in a semantically richer manner when compared to RBAC (Yuan & Tong, 2005). Due to ABAC’s ability to make use of any combination of subject, objects and environmental attributes there is a far more fine-grained means to express policies within the system. These fine-grained policies can be expressed as complex Boolean rule sets that evaluate many attributes (Hu et al., 2014).

In the most general sense, a policy rule that is used to decide whether a subject, \( S \), is able to access a object, \( O \), in the environment, \( E \), is a Boolean function enacted on \( S \), \( O \) and \( E \)’s attributes in the form of:

\[
grant\_access(S, O, E) \leftarrow f(ATTR(S), ATTR(O), ATTR(E))
\]

If the Boolean evaluation of the function returns true then access to the object, \( O \), is granted to subject, \( S \); otherwise \( S \) is denied access to \( O \).

3.3.5 **Context-Based Access Control**

Context-Based Access Control (CBAC) (Shang et al., 2008) is a natural evolution of the ABAC and RBAC models (Kapsalis et al., 2006). CBAC takes environmental attributes, native to ABAC, to address the context of various subjects, objects and the environment itself. The primary basis for CBAC is context, which is discussed first.

a) **Context**

Context is anything that is used to characterise the situation of an entity (Feng et al., 2008). An entity may range from a user of the system, to a resource being used or the very system itself. The context of an entity is characteristic of what the entity is, is doing, or is affected by during its operation. In contrast to attributes as defined in ABAC, context is application specific and may consist of any information that an access decision may depend on.
For a LPSS, the following could be examples of contexts for various entities:

**Table 3.3: Examples of different types of context**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>User</strong></td>
<td>Time logged in</td>
</tr>
<tr>
<td></td>
<td>Time since last keystroke</td>
</tr>
<tr>
<td><strong>Device</strong></td>
<td>Signal strength</td>
</tr>
<tr>
<td></td>
<td>Geographical location</td>
</tr>
<tr>
<td><strong>Environment</strong></td>
<td>Number of connected devices</td>
</tr>
<tr>
<td></td>
<td>System up-time</td>
</tr>
</tbody>
</table>

By using the context of entities such as John’s smart device, it is possible to determine its access control permissions (Sang et al., 2008). The assignment of permissions is adjusted according to changes in context information, based on the interaction between the environment, subjects and objects. Thus, authorisation can only be granted to an entity when the requirements governing access control for a particular object are met (Sang et al., 2008).

The type of contextual information used is entirely dependent on the application domain. This contextual information is called *context constraints* in CBAC (Kapsalis et al., 2006). Brezillian & Mostefaoui (2004) specify three types of contextual information as follows:

- **Simple**: Context information to be used in its raw un-processed form.
- **Interpreted**: Context information that is processed into a more meaningful form. Processing of information is done when raw context information is not meaningful enough as is.
- **Composite**: Context information collated into a set. Can contain both simple and/or interpreted context information.

Context is taken into consideration in order to cope with new situations that may arise due to the mobility and heterogeneity of devices used in ubiquitous computing environments (Brezillion & Mostefaoui, 2004). In order to control a pervasive environment, a specific configuration of access control policies is defined, based upon the initial context of the environment. Thereafter, context is continually revised as the environment changes. In order to deal with revisions in context, the access control policies governing the environment need to adapt to align with contextual changes. Changes in context may lead to breaches in access control that need to be addressed by revising the access control policy. Thus a cyclic relationship is formed. Figure 3.9 illustrates the cyclic nature of policy changes resulting from contextual changes in a CBAC system.
**Figure 3.9: Context manager**

**b) CBAC policy formulation**

As CBAC is built upon RBAC it is only natural that it uses some of the mechanisms from RBAC. CBAC augments the roles and permissions of RBAC by adding context in the form of constraints. Context constraints provide another level of granularity for CBAC policies over those of RBAC. Another advantage of this extra granularity is that it does not have to be managed by an administrator thanks to the cyclic nature of context in CBAC.

Thus, a policy rule that is used to decide whether subject, \( S \), is able to access object, \( O \), in the environment, \( E \), is evaluated in the form of:

\[
\text{grant_access}(S, O, E) \iff \text{roles}(S) \subseteq \text{required_roles}(O) \\
\land \text{permissions}(\text{roles}(S)) = \text{required_permissons}(O) \\
\land \text{constraints}(S, E) = \text{required_constraints}(O, E)
\]

Thus, a subject is granted access if they have the required role to access the object and the required role has the required permissions and the contextual constraints of the object in this particular environment are fulfilled by the subject (Kapsalis et al., 2006). The contextual constraints are imposed by context monitor and change over time so a subject may be granted access today but not tomorrow dependant on their actions in the interim.

**3.3.6 Trust-Based Access Control**

Trust-Based Access Control (TBAC) is a model that has evolved from RBAC (Lin et al., 2006), where trust is defined as the capability to rely on the ability, character, or integrity of some entity (Lin et al., 2006). In particular, trust describes a relationship between two or more parties. The relationship is usually established after interaction has occurred over time. Thus, if Mary’s smart device trusts John’s smart device, then it can predict with a measure of certainty how John’s smart device would
behave in future interactions. The relationship, once determined, influences all future interactions between the parties.

Once a trust relationship is formed, it allows parties to share and make use of services and resources of a particular level, similar to how a role is used in RBAC. Thus, if Mary’s device trusts John’s device, it allows the device to access to authorised files and services with a certain trust level in the LPSS.

a) Trust relationships
A trust-based relationship exists between two entities where the level of trust between the two is determined by taking into consideration all previous interactions. Although there are many ways to calculate trust, the next example, as adapted from Lin et al. (2006), give an indication of aspects that are considered in such a calculation.

\[ T_{p1,p2} = \frac{\sum_{i=1}^{n}[o_i(B_i) - e_i(B_i)]}{t} \]

- \( T_{p1,p2} \) is the level of trust between \( p1 \) & \( p2 \) (usually a value between 0 and 1).
- \( t \) is timeframe of their interactions.
- \( B_i \) is the set of all behaviours occurring during the \( i^{th} \) interaction.
- \( o_i \) is the value observed for the behaviours of \( B_i \) and
- \( e_i \) is the value expected for the behaviours of \( B_i \).

Thus, the trust one entity has in another is subject to the sum of the outcomes of all previous interactions over a given timeframe. The behaviours observed during the interactions can be, and are most likely, derived from the context of the interaction between subject, object and the environment.

b) Trust levels and policies
TBAC allows for fine-grained control over the amount of access granted, dependant on the level of trust that one entity has in relation to another. For example, trust can be measured on a scale of 0 to 1 where 0 represents no trust and 1 represents absolute trust, or on a trust-level system similar to the security-levels of MAC.

An example for John's Local Personal Smart Space can be constructed as follows:

<table>
<thead>
<tr>
<th>Trust Level</th>
<th>Access granted</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No access granted whatsoever</td>
</tr>
<tr>
<td>0.25</td>
<td>Can search for, and list resources and services</td>
</tr>
<tr>
<td>0.50</td>
<td>Can search for, list and read resources and services</td>
</tr>
<tr>
<td>0.75</td>
<td>Can search for, list, read, execute, add and modify resources and services</td>
</tr>
<tr>
<td>------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>1</td>
<td>Can search for, list, read, execute, add, modify, delete and grant access to resources and services</td>
</tr>
</tbody>
</table>

As TBAC is based on RBAC it is only natural that some of RBAC’s mechanisms carry through to TBAC. A subject needs a required role to gain access to an object, but roles are assigned based on trust levels. Subject roles are dynamically assigned by the TBAC model during the trust computation for the subject (Kagal et al., 2001). Thus, if an object is given a trust level of 0.75, he/she is assigned the role of trusted. The dynamic nature of the trust computation means that a subject will not always get the same role assigned, especially if they exhibit deviant behaviour.

Each of the access control models described here are now evaluated for a LPSS environment to determine how well-suited they would be.

### 3.4 Evaluation of access control models

In order to evaluate the access control models to protect a LPSS environment, the LPSS scenario posed in Chapter 2 is now referenced. In this scenario, subjects in the LPSS are the group of devices that are connected to each other. The subjects require access to a particular file (object) on another smart device. The action that needs to be performed is the sharing of the file. The scenario is defined as follows:

- John has a smart device named `device_{j_1}` and Mary has a smart device `device_{M_1}`. These are the subjects.
- Both devices belong to the LPSS of their owner, as well as the family’s LPSS.
- John needs to grant access to file `A` that is on `device_{j_1}`, to `device_{M_1}`. File `A` is the object and `device_{M_1}` is the subject requesting the file to be shared.

In each of the following cases, the access control policy is defined by a logical rule that each of the respective access control models are capable of supporting. For the best possible outcome, a rule should accommodate all of the LPSS’s access control requirements.

#### 3.4.1 DAC

`Device_{M_1}` is granted share access if the result of the access control list lookup for the file `file_A` returns a set containing at least share under `device_{M_1}`’s entry.

**DAC policy:**

\[
grant\_access\left(device_{M_1}, file_A\right) \iff \text{"share" } \in ACL\_lookup\left(file_A, device_{M_1}\right)
\]
DAC may be able to support the granting of access to all devices in the LPSS to the file in question, but it requires that the ACL be updated for each and every smart device, as well as the type of access required. For John, this would become a managerial nightmare. Furthermore, Mary can discretionarily assign her privileges to Mark, Peter’s friend, thereby invalidating the intention of the owner to protect and object. Given the access control requirements of the LPSS, DAC fails to provide a means to use contextual information such as distance and time of day, and trust levels in access control rules.

3.4.2 MAC

Device \_M \_1 can be granted access to file \_A if it has a security level greater or equal than file \_A.

MAC policy:

\[
grant\_access(device\_M\_1, file\_A) \Leftarrow security\_level(device\_M\_1) \\
>= required\_security\_level(File\_A)
\]

MAC allows Mary’s device share access to objects provided that it has at minimum the required security level of that object. MAC’s lattice, however, does not allow device \_M \_1 to write to an object that is of a security level below its own, which may be too strict for a LPSS. MAC fails to use contextual to determine information such as the time of day. Nor does it have a means to gauge how trusted a subject is. The administrator is responsible for setting access levels which means dynamic alteration of the trust level of subjects is not possible.

3.4.3 RBAC

Device \_M \_1 is granted access to file \_A if the roles assigned to file \_A are a subset of the roles assigned to device \_M \_1 and the required permissions are assigned to the roles.

RBAC policy:

\[
grant\_access(device\_M\_1, file\_A) \Leftarrow required\_roles(file\_A) \subseteq roles(device\_M\_1) \\
where: required\_roles(file\_A) = \{supervisor\} \\
&"supervisor" \in roles(device\_M\_1)
\]

RBAC provides a finer-grained level of control when compared to DAC or MAC but it still requires a dedicated roles administrator. RBAC can be extended to accommodate either temporal or spatial contexts such as time or location. However, like DAC and MAC, there is no possibility to define trust in the base model.

3.4.4 ABAC

Device \_M \_1 is granted access to file \_A if the required attributes of file \_A are a subset of device \_M \_1 ’s assigned attributes.
ABAC policy:

\[
\text{grant}\_\text{access}(device\_M_1, file\_A) \iff \text{required\_attributes(file\_A)} \subseteq \text{attributes(device\_M_1)}
\]

\[
\text{Where: required\_attributes(file\_A) = \{"wife", "tablet"\} & "wife", "tablet" \in attributes(device\_M_1)}
\]

ABAC is a further step in the right direction as it lessens the managerial burden as no roles and their permissions have to be defined. However, attributes must be assigned. ABAC focuses less on the identity of a subject and more on the properties of the subject. This increases access control flexibility. The down-side however is that access control policies require manual alteration when new trends in the environment emerge. Trust metrics can be seen as an attribute, but are not directly supported.

3.4.5 CBAC

device\_M_1 is granted access to file\_A if it is a certain time of the day, and the devices are within a certain distance from each other.

CBAC policy:

\[
\text{grant}\_\text{access}(device\_M_1, file\_A, smart\_space) \iff (time\_of\_day(smart\_space) > 9Am \land time\_of\_day(smart\_space) < 5Pm) \land distance(device\_I, device\_M_1) < 15m
\]

CBAC naturally supports LPSS access control requirements such as context, which can evolve based on changes in the environment. The model does not support the computation of trust, but it can be included as a context constraint. The major advantage is the lack of time consuming administration of roles, permissions or attributes.

3.4.6 TBAC

device\_M_1 is granted access to file\_A if the device has the same or better trust level than the trust level assigned to file\_A.

TBAC policy:

\[
\text{grant}\_\text{access}(device\_M_1, file\_A, smart\_space) \iff trust(device\_M_1, smart\_space) \geq trust(file\_A, smart\_space)
\]

The trust level of device\_M_1 is computed based on all previous interactions over a given timeframe. The behaviours observed during interactions are derived from the context of the interaction between subject, object and the environment. TBAC allows for important coverage of some of the LPSS access control requirements. However, there is the exception of dynamic changes in policy and mechanisms based upon changes in the environment.
After the evaluation of each of the access control models using a practical example, ABAC, CBAC and TBAC stand out as having the most complete set of features to address the LPSS access control requirements. Finally, a more formal evaluation of these access control models is provided.

3.5 Formal evaluation of access control models

This section serves to collate the findings of each of the addressed models and provide some insight into each of their strengths and weaknesses in a LPSS environment. In the previous chapter, a number of access control requirements were presented for LPSSs. It is imperative for an access control model to address the majority of these requirements for it to be successful in a LPSS. Below, these requirements are reiterated for comparison and evaluation:

1. **Strict access control enforcement**: Access to resources must be strictly enforced on each device so that access to them is limited. Policy must dictate what can and cannot be done so that unauthorized entities are not granted access to resources.

2. **Simple access control policy management**: Access control policies should be straightforward for group and device owners to implement and maintain.

3. **Policy combination**: Policy rules of different LPSSs need to be processed together, and if conflicts exist, it should be solved appropriately.

4. **Context**: Users, device interactions and the environment must be monitored in order to define flexible access control rules.

5. **Trust**: Trust can be used to grant access to devices that are not members of a group, based upon the level of trust in them.

These requirements, however, are very broad and cannot be directly addressed by all of the models in their current state. In order to combat this, each of these requirements are expanded upon to better address each of the models unique characteristics.

The first requirement of **Strict access control enforcement** can be further broken down into *controlled information flow, unidirectional information flow, discretionary delegation of access control privileges and mechanism-level management* (sections a – d below) to better understand what each model brings to the table in terms of its ability to enforce access control.

The second requirement of **Simple access control policy management** can be further broken down into *manual policy-level management and self-adapting characteristics* (sections e & f below). This is done to better understand the amount of policy administration required and the model’s ability to dynamically adapt its own policies without administrator intervention.
The third requirement of Local and global policy combination does not directly address a characteristic of any of the access control models above and will be addressed in future chapters of this research.

The fourth and fifth requirements of Context and Trust are expanded on in the form of use of contextual information and use of trust computation respectively (sections g & h below).

Evaluation Topics:

a) **Controlled information flow**: The ability of the access control model to enforce strict access control. Once access control is granted, actions on objects should be monitored to ensure that nothing irregular happens (Dennings, 1976).

b) **Unidirectional information flow**: The ability of the access control model to limit who can read or write to an object. This is a property only held by the MAC lattice, which prevents reading up and writing down (Dennings, 1976)(Jiang et al., 2004).

c) **Discretionary delegation of access control privileges**: The ability of the access control model to allow users to discretionarily grant access control privileges to others. This circumvents Strict access control enforcement in the DAC model and is not advantageous to include in the LPSS access control model (Quing-hai & Ying, 2011).

d) **Mechanism-level management**: The ability of the access control model to enable manual administration of access control mechanisms which realise policies.

e) **Manual policy-level management**: The ability of the access control model to support manual policy management for either the operation of access control or the initial set-up of the access control policies.

f) **Self-adapting characteristics**: The ability of the model to adapt its policies based upon automatic feedback from its operation and to thereafter adapt its mechanisms to enact the changes on a policy level (Kagal et al., 2001).

g) **Use of contextual information**: To use the context of subjects, objects and/or the environment (Kapsalis et al., 2006).

h) **Use of trust computation**: To determine a trust metric dynamically and use it in access control decision-making (Kagal et al., 2001).

Here follows a tabular summary of the information presented above when brought into context with the requirements established in the last chapter. Table 3.5 presents each of the models explored in this chapter and details which characteristics each model has. The columns presented in green are advantageous to the LPSS scenario while those in red are not. The orange column comprises of characteristics which are important but are neither advantageous nor provide a disadvantage.
Table 3.5: Summary of access control models

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>MAC</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DAC</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>RBAC</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ABAC</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>CBAC</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TBAC</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3.5.1 Findings

When reviewing Table 3.5, a number of models have disadvantages which render them incapable for use in the LPSS environment. Others are borderline as they have disadvantages, but also do not have an abundance of advantages.

Immediately, MAC and DAC can be dismissed for a LPSS environment. DAC supports the ability to discretionarily assign access control privileges to another subject and can lead to a direct breach of security in a LPSS environment. For MAC, the lattice structure only allows for uni-directional information flow which is not conducive to a peer-to-peer environment. Both models are static and require a great deal of administrative effort to operate.

Native RBAC is a borderline case as it does not provide any real disadvantages apart from the administrative effort required to maintain the roles of subjects and objects. This, however, can be a nightmare when the LPSS is set to operate on mobile devices with small screens that are not conducive to policy administration. Thus, native RBAC is also not suitable for a LPSS environment.

ABAC is an improvement over RBAC as it allows for the requirement of contextual information to be used in the form of attributes. It does however still have the disadvantage of requiring manual mechanism-level management as the ABAC model cannot dynamically adapt itself to changes. ABAC is a step in the right direction but it is still not fully suitable for a LPSS environment.

The models which have more advantages are CBAC and TBAC. In isolation they are not perfect, as TBAC does not directly support contextual information and CBAC is incapable of computing trust. If these models are combined, they could potentially cover all of the advantages required by the LPSS
environment. Trust and context mechanisms could be applied together to fulfil LPSS access control requirements. In doing so, it would make it possible for this research to move forward with a clear direction of how to address the access control solution to the problems facing John’s LPSS scenario.

3.6 Conclusion

In this chapter the topic of access control was discussed. Various concepts relating to access control terminology and policy were discussed to gain better insight into what is required of a fully-functional access control model. Various popular access control models were then presented and analysed to determine how they could be used in-line with the LPSS scenario to address the questions posed in this research. Each model was broken down into the key concepts that made it unique. Thereafter, each was addressed in terms of its ability to protect resources in the very specific scenario posed by this research.

The results of the evaluations were tabularised into a bite-sized summary of this chapter's findings. It was found that no one model fulfils all of the requirements of the LPSS scenario. As such, no one model can be used in isolation to secure a LPSS such as the one required by John. The concepts which made each of the models functional were broken down so that all of the models could be compared. It was found that CBAC and TBAC could together fulfil the access control requirements of the scenario. Even with the inclusion of the neutral property manual policy-level management they do not suffer from the limitation to constantly adjust policies as the operational environment changes as this can be done automatically without user input. Thus, the manner in which CBAC and TBAC use context or trust respectively is fundamental to fulfil the access control requirements of the LPSS scenario.

In the next chapter, the topics of context and trust are explored in existing smart space access control models to gain a better understanding of how they can be applied to a LPSS.
Chapter 4
Context and Trust in Smart Spaces

4.1 Introduction

In the previous chapter, features of access control models that would best suit a Local Personal Smart Space (LPSS) were identified. Both trust and context were recognized to be relevant to access control for LPSSs. Context was identified as a natural means to enhance access control decisions, as smart devices can measure interactions with other devices and the environment in which they operate. From context, the concept of trust evolved as a means to further refine an access control model by allowing the nature of ongoing interactions to change the way in which access control decisions are made. Thus, good behaviour can enhance a trust score and provide greater access while deviant behaviour can degrade a trust score so that less access is granted.

The focus of this chapter is to explore the use of both context and trust in current access control research for smart spaces. As the topic of context was discussed in the previous chapter, it is not covered in any further detail by this chapter. This chapter introduces trust and its properties in Section 4.2, to understand how it can be used in access control. The chapter describes three current context-and trust-based access control models according to a set of criteria presented in Section 4.3. Thereafter, Section 4.4 evaluates each of the access control models to determine their suitability for LPSS access control. In Section 4.5 the results are tabulated and the chapter is concluded.

4.2 Trust

Trust in computer systems is used in much the same way as in real life. Where a person would like some assurance of the reliability or trustworthiness of another person in a real-life transaction, entities in a computer system require the same assurances. There are a number of factors to take into consideration when trust is concerned, they are outlined as follows:

1. What does it mean to be trustworthy? More so, what is trust?
2. Is John’s evaluation of trust the same as Mary’s?
3. How does trust evolve? Is it computed once and the entity is forever trusted at a particular level or does it change based on how the entity interacts?
4. If John trusts Mark, can Mary also trust Mark?
5. How do we know if John is being honest?
Trust in computer systems is first and foremost a measure of whether or not an entity can be relied upon to act in a particular way (Luhmann, 1979)(Zucker, 1986)(Hwang & Burgers, 1997)(Golbeck, 2006). Thus, a trustworthy device can be expected to act in a way that is acceptable to the system, whereas an untrustworthy device can be expected to act in a way that is not acceptable to the system.

When one meets a stranger they do not know whether or not that person can be trusted to perform a task in a satisfactory manner. In a scenario such as this, one needs to have some initial faith in the stranger to give them the opportunity to earn trust through their actions. Trust formed in this manner takes time and requires an initial degree of faith in the other party. Alternatively, if one is already trusted by another party, they can vouch for the stranger's trustworthiness. The latter approach may seem to be a better choice as it does not require one to implicitly trust a stranger who may act in an unacceptable way. However, as trust is subjective, one person may trust a stranger implicitly whereas another may not trust them at all. This leads to conflicting reports of the trustworthiness of a stranger.

To better understand the above in the context of computers and access control, the following sections investigates properties of trust and recommendations to better understand the way in which trust is used in computer systems.

**4.2.1 Properties of trust**

Trust relationships in computer systems have properties, much like real-life relationships, which can drastically change the ways in which trust is computed and updated. The calculation of trust values may be done through a series of formulas. However, there are a number of trust properties which must be taken into account before formulas can be created and used. The universal properties of trust and how trust relationships evolve are described as follows (Hwang & Burgers, 1997)(Lin et al., 2006)(Golbeck, 2006):

- **Trust is subjective**: Trust computations are subject to an entity’s judgement levied against circumstances, which may not be the same for all. For example, John trusts Mary to drive a car but not to perform surgery on him.

- **Trust is unidirectional**: One entity may trust another, but the trust may not be reciprocated and cannot be automatically derived. For example, John trusts Mary, but Mary does not trust John.

- **Trust is situational**: Dependent on the situation of the interaction, one party may trust another, but not in all cases. For example, John trusts Mary to drive his car but not after she has had a glass of wine.

- **Trust evolves over time**: Relationships between entities can be strengthened or weakened as more interactions are undertaken by the two parties. Thus, deviant behaviour
could lower an otherwise untarnished trust score, as acceptable behaviour can strengthen a poor score.

- **Trust is easier to destroy than to repair**: A single misdemeanour can ruin a trust value, while multiple instances of acceptable behaviour may not necessarily repair it.

- **A base-level of trust can be established if an entity produces a certificate signed by a valid certification authority**: Third-party certification enables a greater amount of trust to be generated upon first contact as they can have their behaviours traced.

### 4.2.2 Recommendations

In real life, people form trust in others by collecting information through experiences with people and receiving the recommendations of others. A real-life recommendation is an opinion obtained from another party relating to a specific situation such as a service provided. For example, John gives a recommendation to Mary that Mark can be trusted to fix her car, thereby enhancing Mary's trust in Mark if she trusts John's judgement. In a system, computed trust values can be made available to others as recommendations (Luo et al., 2008)(Hasan et al., 2009), much like humans can ask for someone's opinion of another person's trustworthiness.

A system can accept a degree of trust in the form of a recommendation from another system that has an established degree of trust in the entity in question (Luo et al., 2008). Thus Mary can improve her trust value for Mark, based on John's opinion. However, to address the question of whether John can be trusted, there must be more work done by Mary. In computer systems it is very seldom that entities operate in isolation (van der Horst et al., 2005). Mary can request recommendations for Mark from other entities that have a trust relationship with him. If John is the only entity capable of providing a recommendation for Mark, she can accept it based on her degree of trust in John. Thus, if she trusts John, she trusts his judgement. However, if multiple recommendations are available for Mark then she can consider all of them collectively; both the positive and the negative, and reach a trust value for Mark based on general consensus (Luo et al., 2008). If John is the only one that trusts Mark, Mary will not be inclined to trust Mark, as the consensus is that he is not trustworthy.

In the next section, three access control models relevant to this research are explored to determine how context and trust are used.

### 4.3 Context and trust in access control models

Three recent works are now evaluated to determine how context- and trust-based access control models are applied to smart space environments. In each case, the evaluation starts with a brief introduction of the research and then moves on to quantify the work with respect to five areas of
interest. Finally, Sections 4.4 and 4.5 compare and contrast each access control model to provide insight into what is being done with respect to the access control mechanisms of context and trust.

The five areas of primary interest are:

1. **Nature of environment:** The environmental nature considers how centralised or decentralised the environment is and the distribution of devices therein. For centralised environments, a dedicated controller is present, and for decentralised environments, devices are autonomous and manage their own connections.

2. **Access control goals:** The goals the research set out to achieve, and the means they used to achieve the end goal.

3. **Context:** The way in which context was used in terms of users, devices, environment and recommendations.

4. **Trust:** The way in which trust was computed and used.

5. **Policies:** The nature of the policies used in the smart space. This addresses management, scope and granularity of each.

Three access control models are now discussed, each with different characteristics that highlight smart space functionality. The models range from small device-clusters in close proximity to globe-spanning networks. Each model was chosen based upon the services that it provides to users and the type of access control and policy management that they use. The first model evaluates smart devices in close proximity while the second evaluates a world-wide smart space network. The third is geared towards peer-to-peer environments with devices in close proximity.

### 4.3.1 A fuzzy approach to trust based access control in the internet of things

Mahalle et al. (2013) identify that devices in the Internet of Things (IoT) have nomadic tendencies to form dynamic decentralised networks when they come into contact with other devices. Devices have varying forms and capabilities, but due to the nomadic nature of some of them, traditional access control models are not suitable for use.

In order to combat the shortfalls of access control for decentralised environments, the research proposes a Fuzzy Trust-Based Access Control (FTBAC) framework which caters to smart spaces by using contextual information to determine trust in the form of a linguistic term used for access control. The research structures their FTBAC framework in three distinct layers as depicted in Figure 4.1 below.
The FTBAC framework is structured in three layers. The Device Layer comprises of devices in the IoT in close proximity that collect contextual information to form experience, knowledge and recommendation values. The Request Layer collects pre-compiled experiences, knowledge and recommendation information from all devices. Devices deliver information to the controller of this layer, which is the current group owner. Thereafter, the controller computes a trust value for each device which is translated into a Fuzzy Trust Value (FTV). The Access Control Layer maps the FTV to a list of access control permissions. This layer is ultimately responsible for enforcing access control.

a) Nature of environment
The research caters to a decentralised smart space environment. There is no central control as devices are nomadic and they form connections as they come into range. Each device is responsible for managing its own context information, trust computation and resultant access control decisions. A controlling entity is the owner of the group within the smart space that is negotiated during initial connection or when the current controlling device leaves smart space range.

b) Access control goals
The goal of the access control model is to use contextual factors of experience, knowledge and recommendation, to compute a reliable trust value, to assign to each device, to use in access control decision-making.

c) Context
The model makes use of device and environmental contextual information but user context is not considered. The values of contextual information are stored for long periods of time to ensure fair computation of trust.
d) Trust
The research leaned heavily on the work of Jianyu et al. (2008) for trust computation in trust-based access control, as similar values for experience, knowledge and recommendation are used. Experience (EX) values such as good, average, bad are computed as trust value for a device, using the context of current interactions. Knowledge (KN) is a measurement of the amount of contextual information available that could result in linguistic terms such as complete, less or insufficient depending on the quantity of information available. Recommendation (RC) is derived from the opinions of other devices. The results of each computation results in a FTV that is used to determine if access is to be granted. The resultant trust term is discarded after access decisions are made.

e) Policies
The controlling device specifies the policy that each connected device is subjected to. As the controlling entity is subject to change, so too are the policies governing the smart space. The result is a global policy which is specified by whichever device is in control of the smart space at the time. The use of FTVs makes policy specification simple and straight-forward.

4.3.2 Context-based access control model for smart spaces
Smirnov et al. (2013) propose that smart spaces are part of a physical environment as they connect physical devices that exist in the physical world, while at the same time sharing information in cyberspace. The physical environment provides the context for the operation of the smart devices, to allow them to form opinions based upon their interactions (Mohsin Saleemi et al., 2011). There are four major components to consider as shown in Figure 4.2.

![Figure 4.2: The cyber-physical environment](image)

Virtual community members are users of smart space services that exist in the physical world with their devices, and consume or produce information in the cyber realm. Acting resources control physical resources and information resources exist in the cyber realm. Both are bridged by computation resources that exist in the physical world but act upon information that exist in the cyber realm. For example, a device can measure the temperature which is existent in the physical-world and
then store information in the cyber realm with the digital thermometer being the *computation resource*.

The research motivates the importance of giving smart space service providers more security responsibilities and not leaving it to connecting entities. For example, a centralised smart space service provider can verify all applications in the smart space, thereby adding an additional degree of security for its users. The research draws a comparison to cloud computing environments where similar problems have been solved by building access control directly into the infrastructure.

**a) Nature of environment**

The access control model caters for smart spaces that can be distributed across the globe and connected to through various means such as the Internet, Bluetooth, or Wi-Fi. Control is exerted by a single monitoring and controlling entity which is not subject to negotiation. The research is built upon the Smart-M3 open-source platform (Honkola et al., 2010) and supports core smart space functionality on most mobile platforms. Information is distributed throughout the smart space via the use of imbedded information brokers making use of Uniform Resource Identifiers (URIs) (Burners-Lee et al., 2005).

**b) Access control goals**

The research has a number of defined goals, most notably:

1. Access control is provided at multiple levels as specified in predefined policies which are adapted based on the context of current interactions.
2. Predefined policies are descriptive, flexible, well-defined and simple to configure.
3. Post-access control, private information is transferred through a secure channel.
4. Authentication is done for both users and their devices.
5. Trust is computed based on the context of user, device and environmental interaction.
6. Trust values are assigned to roles in order to save resources within the system.

The combination of all of the above allow for the access control model to provide access control services to decentralised environments that are not limited by physical location.

**c) Context**

The management of context in the research deals with both the physical and virtual components of user, device and environmental context. The physical component deals with sensory information about where and what a device is, and the virtual component deals with what the device is doing. For example, a device in a trusted location may be assigned a high environmental context value, and a device in an untrustworthy location may receive a significantly lower value. The values of contextual information are stored for long periods of time to ensure fair computation of trust.
d) Trust

Trust in the research is defined by a numerical value which is computed in the range of $[0, 1]$, which is computed based on the context of the current situation of the device in the smart space. A 0 value indicates no trust and value of 1 indicates absolute trust, with values ranging in between. Based on the outcome of the trust value computation, a role is then assigned to the device such as trusted or untrusted. These roles are required by various resources for access to be granted. The trust computation determines the trust level by considering any number of contexts over a given timeframe which are weighted according to rules outlined in policies. The trust values are disregarded after access control is done as an up-to-date computation is done for each access request.

e) Policies

The research caters for a single global policy which is stipulated by the controlling entity. The policy is dictated by whomever is administrating the controlling entity and no form of personal preference for any connected entities is considered. The policies are flexible, well-defined and cater to the use of the roles used in the model.

4.3.3 TrustAC: Trust-based access control for pervasive devices

Almenarez et al. (2005) propose a model capable of dealing with peer-to-peer smart device interactions and resource sharing. The research proposes a trust-based model capable of reducing the policy administration workload of traditional centralised access control models by using dynamic trust computation and policy generation. The research aims to replace existing static access control models with a dynamic and decentralised approach to access control with trust and policy generation at its core.

![TrustAC model](image)

**Figure 4.3: TrustAC model**

In Figure 4.3, users have smart devices that connect to others to share information and resources. Each device is responsible for storing information about interactions with other devices in order to make their own access control decisions. Should devices not have any stored information about another, they can request recommendations from devices in the environment. Each device computes a RBAC-like trust role for all devices it encounters. Trust roles are mapped to permissions which dictate what
actions can be performed on resources. The model aims to break away from traditional access control which is ineffective in decentralised environments.

a) Nature of environment
The research caters to a decentralised smart environment where devices are in close proximity to one another. Devices are peers with no form of central control, as each device is responsible for its own management and access control.

b) Access control goals
The goal of the research is to provide a means for devices to be self-reliant with respect to access control decision-making.

c) Context
Context is used in the research as a form of evidence of past interactions. If the interaction was good, a positive contextual value is stored. If the interaction was bad or malicious, a negative value is stored. Thus, malicious interactions can severely reduce a device's computed trust value. The model stores contextual information about users, devices and the environment to use in every trust computation.

d) Trust
As depicted in Figure 4.3, devices are assigned a trust value, or degree, based upon the contextual information available to the device at the time. If no information is available, a recommendation can be used to provide a degree of initial trust. Recommendations are averaged and weighted by the degree of trust that the device has in the recommender. Once a trust value in the range of \([0, 1]\) is computed, it is assigned a Fuzzy Trust Value (FTV) which acts similarly to a RBAC role. The role has a number of permissions assigned. During access control requests, each device compares the permissions assigned to the FTV and determines if the requesting device should be granted access or not.

e) Policies
Users are given a degree of personal preference as they can specify the access control policies of their devices. Policies are specified using XACML to provide fine-grained flexibility over rule specification, which is augmented by the abundance of contextual information stored by each device.

4.4 Access control model evaluation
The table below considers the five main topics of Section 4.3 and summarises them into an easy-to-compare tabular format.
Table 4.1: Tabular summary of access control model comparison

<table>
<thead>
<tr>
<th>Model Functionality</th>
<th>A fuzzy approach to trust-based access control in the Internet of Things</th>
<th>Context-based access control model for Smart Space</th>
<th>TrustAC: Trust-based access control for pervasive devices</th>
</tr>
</thead>
<tbody>
<tr>
<td>Environment Type</td>
<td>Decentralised Smart Space</td>
<td>Decentralised Smart Space</td>
<td>Decentralised Smart Space</td>
</tr>
<tr>
<td>Proximity of Entities</td>
<td>Close Proximity</td>
<td>Geographically Dispersed</td>
<td>Close Proximity</td>
</tr>
<tr>
<td>Controlling Entity Selection</td>
<td>Device Negotiation</td>
<td>Fixed</td>
<td>None</td>
</tr>
<tr>
<td>Context Types Used</td>
<td>Device</td>
<td>Device</td>
<td>Device</td>
</tr>
<tr>
<td></td>
<td>Environmental</td>
<td>Environment</td>
<td>Environment</td>
</tr>
<tr>
<td></td>
<td>User</td>
<td>User</td>
<td>User</td>
</tr>
<tr>
<td>Contextual Information Stored</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Recommendation Used</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Trust Outcome Type</td>
<td>Fuzzy Linguistic</td>
<td>Numeric</td>
<td>Fuzzy Linguistic</td>
</tr>
<tr>
<td>Trust Value Stored</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Fine-grained Computation Control</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Use of Roles</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Use of Attributes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Uses Minimal Resources</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Policies</td>
<td>Global</td>
<td>Global</td>
<td>Local</td>
</tr>
</tbody>
</table>

The access control models have similarities and differences in architecture and operation. Two cater toward devices in close proximity and one allows devices to negotiate control over the environment.

Their use of context is uniform, they all store similar types of contextual information, but only one model stored the resultant trust values after using it in access control decision-making. Therefore, this
comparison supports the requirement of a LPSS scenario for a trust and context-based solution, as each of the models successfully incorporates it. Some of the differences such as the use of fixed controlling entities, and the lack of control negotiation however renders the models incapable of being used directly in the LPSS scenario posed by this research.

The most relevant aspect to take note of arises when the use of policies is compared. All except one model use a global policy that governs every device in environment without considering any personal preferences. The others are peer-to-peer oriented where the local policy on a device is used for decision making.

The LPSS scenario defined by this research requires that user preference is taken into consideration to ensure that files or services are not shared if it is against the will or knowledge of the user. From literature it was seen that user preferences are not generally supported as only one model supports this case. However, none of the models discussed in this chapter allow policy rules from different domains or users to be taken into consideration at the same time during access control decision-making. Therefore, this research now focuses on how sets of policy rules can be used together to supports the preferences and restrictions of different LPSSs.

At the end of chapter 3, four of the five access control requirements were evaluated with respect to current research. The next section reviews the outstanding access control requirement, namely policy combination, to better focus the research.

4.5 Revisiting access control requirements

Considering the findings given above with regards to the manner in which policies are used, this research now revisits the access control requirement related to policy combination as follows:

- **Policy combination**: Policies rules of different LPSSs need to be processed together, and if conflicts exists, it should be solved appropriately.

Current research is incapable of fulfilling this LPSS access control requirement as different levels of policies need to be considered when making access control decisions.

Aspects that may be important to consider in this regard are:

1. **Personal/group policies**: The personal preference of device owners need to be defined in an access control policy for the group of devices belonging to an owner such as John. Additionally, if John’s group of devices belong to another LPSS group such as the family, this group’s restrictions must also be defined in a policy.
2. **Policy combination:** Personal and group policy rules need to be processed together, and if conflicts exist, it should be solved appropriately. The personal policy should allow device owners to protect their resources and files. Likewise, group policies should protect the more general group. If they are in conflict, a resolution must be reached.

These aspects serve to not only strengthen the security requirements of the environment to which devices connect, but also enables users to personalise the protection of their own resources and files.

### 4.6 Conclusion

In this chapter, trust and its properties were discussed. Thereafter, various context and trust-based access control models from literature were evaluated to determine their suitability to the scenario posed by this research. Each model was critically evaluated using five common areas of interest. All of the models had similarities and differences, but all contributed to the understanding of the way in which trust- and context-based access control can be used in smart space environments.

The most notable finding of this chapter was a on the manner in which policies were used. Policies could either govern an entire environment or only a personal device. There was no middle ground. The controlling entity of each smart space environment was solely responsible for dictating how access control decisions were made within the environment leading to very rigid security. A LPSS is by its very nature personal, and John and his family would not use such an application if they could not apply their personal preferences when sharing information with others. Conversely, not having a global set of rules for the family would mean an additional management workload for each user. Another problem arising from this is how the rules relating to personal preference and the larger group are combined so that all parties accept the access control decision being made.

In the next chapter, the topic of Personal Smart Spaces is reviewed to understand how such applications are designed. This evaluation focuses on the access control and policy components for the purposes of this research. Thereafter, the chosen implementation platform for the prototype is introduced and evaluated with regards to its access control and policy features.
Chapter 5
Personal Smart Space Components and Implementation

5.1 Introduction
This research focuses on enabling smart devices to access and share content with others in the same group while on the move. Today, most sharing schemes rely on intermediary services such as Dropbox and sophisticated infrastructure, with no fine-grained control over resources. The Local Personal Smart Space (LPSS) approach to content sharing proposed by this research can be an attractive alternative to cloud services, to allow users to share content without having to be concerned about data costs.

In order to design a LPSS application, this chapter firstly aims to review current research on smart spaces to identify suitable LPSS components. Secondly, to implement the LPSS framework, mobile smart device platforms need to be considered of which Android (Google, 2015), iOS (Apple, 2015) and Windows Phone (Microsoft, 2015) are the most popular (IDC, 2015). For this research, Android is chosen as its 2014 market share was an estimated 81.5% of all mobile devices globally (IDC, 2015), with iOS at 14.8% and Windows Phone at a meagre 2.7%. As Android is an open platform, with an abundance of support freely available, it is the best choice for the development of the prototype for this research. To determine the suitability of Android for the LPSS framework implementation, it needs to be critically evaluated by exploring the structure of Android. As LPSS access control enforcement needs to be defined in conjunction with natively provided Android access control, a review of Android access control features is provided.

This chapter explores the general architecture of Personal Smart Spaces and their fundamental components in Section 5.2. Next, Section 5.3 discusses the structure of the Android system. Section 5.4 critically evaluates the layers of a PSS implementable by Android in its native state. Section 5.5 describes current access control research for Android. The chapter is then concluded.

5.2 Personal Smart Spaces
Personal Smart Spaces (PSS) are geared to mobile environments, making them ideally suited to be used as foundation for the LPSS components needed by this research. This section presents PSS architecture and briefly discusses its layers and components to highlight key components which need
to be implemented in the prototype of the LPSS. Thereafter, an evaluation is done to determine if a LPSS application can be implemented using the PSS framework.

The next section discusses PSS architecture and its components.

**5.2.1 The architecture of Personal Smart Spaces**

Papadopoulou et al. (2012) discuss the architecture of the PSS platform that is the basis of the PSS called the Persist project (PERSIST, 2008), as shown in Figure 5.1.

![Figure 5.1: PSS Architecture](image)

The PSS architecture consists of five distinct layers. From the bottom, starting with layer 1 up to 5, the layers are: **Devices**, **System Run-Time Environment**, **Overlay Network Management**, **Service Run-Time Environment** and the **PSS Framework**. Each of the layers are discussed in greater detail in the sections that follow.

**a) Layer 1 - Devices**

The *devices* layer is a collection of all the devices which form a specific PSS.
b) Layer 2 - System Run-Time Environment
The system run-time environment is an abstraction layer that allows higher layers of the framework to operate independently of the operating system and devices layer below, consisting of two components namely sensor management and mobility management.

c) Layer 3 - Overlay Network Management
The overlay network management layer is responsible for managing peer-to-peer connections within the PSS and between PSSs. Additionally it supports the discovery of group nodes, maintaining of connections with nodes, discovery of other PSSs, service advertising between PSSs and message routing within and between PSSs.

d) Layer 4 - Service Run-Time Environment
The Service Run-time Environment (SRE) is a PSS-wide resource manager for the Persist framework. It provides the layers above with information about the services available in the layers below. The SRE is capable of providing information management features for availability of data and addressing storage requirements as needed by the PSS.

e) Layer 5 - PSS Framework
The PSS framework layer is responsible for adaptation of the PSS environment based on information gathered through operation. The PSS framework layer is made up of intelligent components which form the heart of the framework. These components provide basic user interaction, personalisation and smart space management components. Beyond simple components, this layer provides more complex learning systems such as the learning management, proactivity and recommender components which learn from the environment and its operation to provide user-tailored services. This layer is also responsible for management of services to meet user needs and provide security through various components which operate throughout the layers of the architecture.

These components work together to ensure resources are only accessible by entities within the PSS or those from other PSSs who are adequately trusted by the framework.

5.2.2 Can a LPSS application be implemented using the PSS framework?
On a conceptual level, the vast majority of the components of the PSS lend themselves well to implement the LPSS scenario posed by this research. The architecture of a PSS is not only modular but also flexible regarding how services are provided. The most notable functions provided by the PSS architecture are the management of connections between smart devices, and the management of the preferences of the users that use them. Other important functions provided are the security mechanisms which allow for secure resource advertising, sharing, utilisation and management.
In light of the above, this research now proposes that the Local Personal Smart Space Framework is designed using a similar layered architecture. The PSS framework components to consider in more detail are; Access control, Trust Management and Policy Management. For example, policies used by the Policy Management component of the PSS framework do not take into account the local preferences of a specific device, but only considers the global policy of the PSS (Papadopoulou et al., 2012). As a result, this research aims to extend the Policy Management and Access Control components to support both local and global policies. Similarly, Access Control and Trust Management components need to be adapted to address all LPSS framework requirements. Valuable PSS functions that need to be implemented in the LPSS framework are functional components that allow users to set up their PSS in a manner that suits them personally, and have those settings and preferences move around with them wherever they are. PSSs also support autonomous operations such as the sharing of files and resources without the explicit intervention of users.

Android is now explored with a focus on security and access control, to determine how the security components of the proposed LPSS framework can be built to complement those of the Android operating system.

5.3 Android
The general system structure of the Android environment is presented in Section 5.3.1, which introduces the layers of the Android system, their functionality and access control components. Section 5.3.2 describes key concepts of Android security and how mandatory access control is used within the system to limit access to privileged resources.

5.3.1 The Android system structure
Android is a comprehensive open-source mobile operating system that supports an execution environment for applications to run on a mobile platform. Primarily, Android provides an application framework with libraries and a virtual machine which runs over a Linux kernel (Android, 2015p)(Sun et al., 2012)(Huang & Wen, 2012).

In Figure 5.2, the Android system is structured in three distinct layers (Oh et al., 2014). These layers are the application, middleware and kernel layers as discussed next:

a) Application Layer
Android applications exist in the application layer and are written in Java. Applications are deployed in the form of Android application packages (APKs) and are referred to as apps (Android, 2015t). Once installed, Android applications use the underlying Application Framework to gain access to resources and libraries to operate. Applications run on an Android device in a virtual machine and can be anything from those illustrated in Figure 5.2 to productivity, leisure or utility software.
b) Middleware Layer
The middleware layer hosts the Android application framework which provides Android libraries for graphics, multimedia, database and a plethora of other required functions. Although Android apps are written in Java, the execution of Java byte code is not directly supported by Android (Ehringer, 2010). To deal with this, applications run on the Dalvik Virtual Machine (DVM) (Oh et al., 2014), a lightweight and highly-optimised version of the Java Virtual Machine. The DVM allows for apps to each run in their own process with their own allocated secure storage space (Oh et al., 2014).

c) Kernel Layer
The Android software stack is an open platform built on top of the Linux 2.6 kernel, which handles all hardware interactions, security, memory & process management and networking capabilities for the device.

When considering these layers, it is important to note that each of the layers play an important role in how applications operate, communicate, and interact with one another and the resources of the system. Android is a privilege-separated operating system which runs each application in isolation to ensure better security (Oh et al., 2014). This can lead to problems when applications need to gain access to files or resources that are not a part of their reserved storage space. In order to provide access to files and services the Android kernel provides Inter-Process Communication (IPC) channels for standard communication with functions of the operating system available to all, and Inter-Component Communication (ICC) for access to privileged information as depicted on the right of Figure 5.2.

Figure 5.2: Android Platform Structure
To better understand the nature of Android security using IPC and ICC, the next section discusses the Android security model and its components.

5.3.2 Android security model

Android is built upon a Linux core that provides security and privacy to all aspects of the operating system (Enck et al., 2009). Due to Android’s privilege-separation, the operating system divides the system into parts with a unique identity to enable the Linux kernel to isolate applications from the rest of the system and even other applications (Android, 2015o).

In Android, each application runs in a separate sandboxed environment to isolate data and code execution from other applications (Huang & Wen, 2012)(Gong et al., 1997). During installation, each application is assigned its own sandbox using a distinctive identifier created using a User ID (UID) (Android, 2015q). Should the application store any data on the device, it is signed using the UID and is not normally accessible to other applications (Shebaro et al., 2015). There are two classes of resource requests that an application can make namely:

- Requests for resources provided by the operating system.
- Requests for access to resources that are privileged or owned by another application.

a) Inter-Process Communication

When applications require access to resources provided by the operating system, it is achieved through Inter-Process Communication (IPC). IPC is a means for the Android operating system to provide applications with access to the processes that manage resources within the operating system. Applications can communicate through basic kernel functionality to access resources such as the current clock time or battery level. There are restrictions imposed on IPC of which the two most notable are the inability for privileged resources to be accessed, or applications to communicate with one another either directly or indirectly (Oh et al., 2015). For anything more than basic communication through system services, ICC is required.

b) Inter-Component Communication

Inter-Component Communication (ICC) is required if applications need access to resources that are not a part of their own personal sandbox. To be able to access resources, an application must explicitly declare the permissions that it requires in the Android manifest file during the application’s development (Android, 2015q)(Shebaro et al., 2015). For example, if an application requires access to pictures or a Wi-Fi connection it must explicitly declare its need for the class of permission, to enable the operating system to control such restrictions (Shebaro et al., 2015)(Android, 2015q). The ICC reference monitor, pictured on the right of Figure 5.2, determines if an application had been given the required permission at install time, and allows or denies the communication based on these permissions.
ICC can also be used when one application needs to communicate with another to gain access to files or resources that does not belong to it. As each application is sandboxed, it is not possible to directly gain access to another application’s storage. ICC uses a Mandatory Access Control (MAC) reference monitor located in the middleware layer to mediate communication requests (Oh et al., 2015). This enables applications to store their information in isolation while still allowing some level of communication.

c) ICC and Intents

ICC uses intents (Android, 2015s) to enable communication between applications requesting and providing resources (Android, 2015r)(Android, 2015s). Intents are a rich inter-application message passing system used for inter-application communication and notification (Android, 2015e)(Android, 2015r)(Chin et al., 2011). Android intents can be used to start an application or service to get a result that cannot be directly accessed (Android, 2015u). In some cases a required file might require an application to open in the form of starting its main activity (Android, 2015a) to get the location of a picture or file to be returned. In other cases the intent can start, or request information from, a service (Android, 2015v) which runs in the background or can be used for broadcasts (Android, 2015s)(Android, 2015w). These different types of intents identify what type of application either delivers the intent or makes use of it.

Intents cannot be used to gain access to resources that an application does not have privileged access to. Thus, an application cannot create an intent to access device storage if it does not have the permission to do so itself. In this case, the MAC reference monitor will not allow the intent to be sent.

To better understand the use of IPC and ICC in Android, the next section presents examples.

5.3.3 Android access control examples

To illustrate the usage of IPC and ICC, this section provides examples of Android’s MAC reference monitor’s use of user IDs and application permissions to limit access to resources such as pictures. John and Mary use applications to request access to the following:

1. IPC: John’s legitimate application tries to access system clock time.
2. ICC using UIDs: John’s legitimate and malicious applications request access to pictures stored on the device.
3. ICC using permissions: John’s legitimate application tries to access local pictures and Mary’s legitimate application tries to remotely access pictures on John’s device.

Each of these are illustrated in the following examples.
a) IPC

IPC provides basic system functions to all processes that request them. When John’s legitimate application tries to access system clock time the Android IPC will provide the required information as it is not a privileged component. IPC has no real limitations on who can access what system functions as IPC only provides access to the processes managing basic system functionality.

b) ICC using UIDs

Here, John’s legitimate and malicious applications request access to pictures stored on the device. Should John have a number of pictures stored on his device, IPC cannot be used to access them. ICC is used to get access to pictures when using one of the many available picture viewers or file browsers. These applications are allowed access to the files through ICC because they are signed using John’s UID at install time and John accepted the required permissions. This is shown in segment (A) at the left of Figure 5.3.

Should a malicious application, which was installed by John under false pretences, try to gain access to the same pictures, it will be prevented if it lacks the permission to access device storage as illustrated in segment (B) in the centre of Figure 5.3. Likewise, the application will be prevented from creating an intent to access the pictures as it does not have the required permission even if the legitimate picture viewers do. Thus, it cannot make use of intents to surreptitiously gain access to files in another application's secure storage. Should the malicious application request the correct permission to access the device storage at install-time and John accepts it then the application will not be prevented access to the pictures. If users accept any type of permission, applications get privileges that they either do not need or will use maliciously (Mustafa & Sohr, 2012), leading to a very insecure environment.

![Figure 5.3: ICC using permissions and UIDs](image)

c) ICC using permissions

This scenario illustrates how pictures are transferred from one device to the next, which is relevant to how an LPSS application would be used. Firstly, Mary connects her phone to John’s through
Bluetooth, Wi-Fi Direct or some other means. John uses an arbitrary application called *File Sender*, which has the correct permissions, to send a file to Mary as pictured in segment (A) at the left of Figure 5.3. Mary however, cannot directly access John’s files from her device even if she had an application to view and request file as she has a completely different UID. This case is pictured in segment (C) at the right of Figure 5.3.

**d) Summary**

There are two main Android concepts that need to be considered namely:

1. If a user accepts a permission that an application requests then the application can do as it pleases with that permission.
2. Users do not have direct access from their device to files of others on remote devices when those devices are connected to each other. All access occurs using applications with valid permissions.

For point 1, a local user has almost free reign of their device as the user can accept any permissions that applications require. This could lead to breaches in security if applications are malicious or require permissions that they do not use. Additionally, the access that Android grants to files and resources is binary in nature. Either the application has the required permissions and UID or it does not. No further restrictions can be assigned by a user through the reference monitor or via some form of policy as the *Android Policy Manager* only supports a basic set of static rules (Shebaro et al., 2015)(Android, 2015j).

Point 2 is more complicated. If both John and Mary have the same application that can request a file, and John has access to the file then that application can send the file to Mary, provided the application has the correct permissions. Because Mary and John’s UIDs are different, an application is explicitly built to enable the sending of the file. The application is managed locally after the user has accepted its permissions and installed it. Thus, Android only supports indirect access to files on remote devices.

With an overview of Android and its fundamental security features in-hand it is now possible to evaluate the ability of Android to provide the basis for functionality required by a LPSS.

**5.4 Evaluating Android access control**

This section serves to highlight the areas where Android fails with respect to directly implementing a prototype of a LPSS application. When considering the layers of the PSS framework, over which the LPSS application is built, the majority do not relate to access control and fall out of the scope of this research. Layers 1 to 4 provide various functions to the user and enable the smart space to operate
which can all be implemented in Android. The layer of greatest importance is layer 5 where the access control functions of the PSS reside and is therefore of the greatest importance to this research.

Layer 5 of the PSS framework is responsible for managing services such as context measurement, user interaction, personalisation and a number of other integral components. The most important block of layer 5 is the **security & privacy management** block pictured in Figure 5.1. However, when Android is considered as the implementation environment for the prototype, Android fails to address two primary areas required by this block. Namely:

1. The access control component lacks a robust and adaptable implementation in Android that cannot be circumvented by users or applications with too many permissions (Huang & Wen, 2012). No fine-grained access control mechanisms exist as sandboxing, intents and permissions do not provide an adequate level of control over what can and cannot be done with resources.

2. Android’s native policy manager is not a true policy manager in the traditional sense. It can only support a fixed set of basic rules which affect the system and is not flexible (Huang & Wen, 2012)(Ongtang et al., 2012)(Android, 2015j)(Android, 2015l). As the LPSS application requires flexible and fine-grained policies, this aspect must be implemented in another form in the prototype.

The next section describes current research that attempts to address Android’s inability with respect to the two areas above.

5.5 State-of-the-art research on Android access control

This section serves to address the two shortfalls of Android when implementing the PSS framework as the basis for a LPSS prototype.

5.5.1 Android file access control system

Huang & Wen (2012) propose a method to augment Android file access security to indirectly gain access to resources that are normally isolated by the Android file system. Access control is executed in kernel space to provide the full access to resources, while limiting other application’s ability to do the same. They also address Android’s lack of flexible policy tools. The architecture of their solution is presented in Figure 5.4 below.
In Figure 5.4, the file access control system makes a clear differentiation between the *concrete file system* and *virtual file system* of the Linux kernel. The *virtual file system* is responsible for managing all requests to resources in the *concrete file system*. Android’s standard reference monitor is still in place to make initial decisions relating to UIDs and permissions but access may be further limited by the *access control* component of the *virtual file system*. Contrary to standard Android, their system leaves only a small amount of functionality in the User Space, while the remainder is moved from the middleware layer into kernel space. In User Space, *applications* are allowed to operate as normal and request access to resources through the kernel. *Policy management* allows access to the *policy file* for administrative changes to policy.

When *applications* request access to resources in the *virtual file system*, the *virtual file system* is responsible for policy decisions and access control enforcement. Their means for dealing with access control enforcement effectively circumvents the problems with exploiting UID or excessive permissions to gain unwarranted access (Huang & Wen, 2012). This allows the virtual file system to make its own decisions about access control and not be reliant on the reference monitor of standard Android. However, due to the requirement of kernel alteration, this is not an ideal solution for this research as not everyone wants to, or is capable of rooting their devices.

The other area where the author’s research makes tangible strides is that of policy. The authors created simple XML-based policy tool to enable more flexible access control. This is especially useful considering the inherent lack of policy tools in Android which provides a basis for implementation in the prototype chapters to come. A simple policy which explicitly allows certain applications, *app1*
and app2, to perform the operations of create, read, read/write and delete on files, file1 and file2, is presented in Figure 5.5 below.

```
<accessControl>
  <resources>
    <filesset id="1" property="FILE">file1, file2, ...</filesset>
  </resources>
  <rules>
    <rule id="1">
      <global>CREATE|READ|READWRITE|DELETE</global>
      <create>app1, app2, ...</create>
      <read>app1, app2, ...</read>
      <readWrite>app1, app2, ...</readWrite>
      <delete>app1, app2, ...</delete>
    </rule>
  </rules>
  <controls>
    <control fileset="1" rule="1"/>
  </controls>
</accessControl>
```

**Figure 5.5: XML-based policies for Android**

The use of an XML-based policy allows for a degree of flexibility that Android in its native state cannot provide. This improves the access control policy management functionality of Android for use in a PSS environment.

### 5.5.2 CRePE

As another means to combat access control and policy enforcement failures of Android, Conti et al. (2011), propose an Android access control and policy management framework to enable fine-grained policy enforcement on mobile smart devices. Unlike the work of Huang & Wen (2012), the model is built in the Application and Middleware layers of Android which is a great advantage as it does not require a device to be rooted to gain access to the benefits. The architecture of CRePE is presented in Figure 5.6 below.
Figure 5.6: CRePE architecture

In Figure 5.6, applications request access to resources. The CRePE Reference Monitor (CRM) intercepts the requests before they reach the underlying Android reference monitor. This has the added advantage that access control can be performed before the Android reference monitor performs its permission and UID checks.

The CRM relies on a dedicated policy manager which enables users to specify access control polices which Android is natively lacking. Additionally, the policy manager is augmented by the context interactor which detects context changes and enables a finer granularity in the policies that can be specified. Another key difference between CRePE and the previously discussed framework is that policies are stored in the policy provider which is not located in application layer. This provides an additional degree of security for the stored policies as they are more difficult to access.

The final important component of the CRePE architecture is the action performer, which is responsible for performing specific actions on behalf of the CRePE system. The most notable of which is its ability to intercept IPC calls to prevent actions such as turning off of the device or to release resources that are being locked by other applications.

The CRePE architecture may not be a completely functional solution for every failure of Android but it does provide a solid basis to address the access-control-related shortfalls. It also provides some additional insight into where policy management and storage can take place which provides added benefit when looking forward to the implementation of the LPSS prototype.
5.6 Conclusion

This chapter has explored the architecture of PSSs and the components required for their operation and management. A number of components were identified and important access control components were earmarked for the LPSS. Next, the structure of Android was presented and its security model was explored through the use of examples. It was found that Android, although well suited to the architecture of a PSS, was lacking with respect to its access control measures and policies management. Even though Android was not completely ideal for the implementation of PSS components, it is still a viable platform to use. Android’s structure illustrated its key functionality and when evaluated against those required by a LPSS, the majority were present. Android, however, needs to be extended to accommodate the access control and policy requirements of this research.

This research went on to investigate current research on Android access control and policy to determine the best way to address Android’s lack of policy tools and explore a means for better securing resources on smart devices. The work of Huang & Wen (2012) provided the basis for the implementation of basic XML policy mechanisms in Android. To augment this, the work of Conti et al. (2011) provided a more reliable basis for implementing the access control components without the need to root the devices.

In the next chapter, this research proposes a LPSS framework which is directly supported by the layered structure of the Android system. The LPSS framework hosts components that feature prominently in PSSs and provide necessary functionality for both the LPPS and the requirements of this research.
Part 2
Model & Prototype
Chapter 6
The Local Personal Smart Space Framework

6.1 Introduction
Part 1 of this research saw the exploration of various topics related to access control and policy for smart space environments. The research commenced by describing a Local Personal Smart Space (LPSS) scenario and identifying both functional and access control requirements that need to be met by this research. Next, a review of a number of access control models showed that no current access control model directly supports the access control requirements of the LPSS environment. The evaluation indicated that trust and context-based mechanisms may be best suited to the LPSS and were further analysed. In this analysis, no current smart space access control model was found to fulfil LPSS access control requirements. Current research on smart space access control discovered that the use of a controlling entity in access control decision-making, where a global policy dictates all control, is a limitation. Preferences of users are thus not considered together with global policy rules.

By their very nature LPSS applications are decentralised, where access control and policy management need to be performed locally. This lead to the realisation that policies from different LPSSs need to be processed together when many rules are found to be applicable to the same resource.

To be able to implement a LPSS application, the research proceeded to review components of Personal Smart Spaces. From this analysis, important components relating to access control, policy and trust were identified, as well as their place in a general LPSS architecture. To implement the LPSS application, Android was chosen as the prototype development platform. It was found that the native capabilities of Android do not provide fine-grained access control and policy management. Current Android access control mechanisms were investigated to determine how fine-grained access control can be implemented in conjunction with those provided by Android.

In part two of this research, the researcher sets out to design a LPSS framework and access control model to address the functional and access control requirements that were identified. The major contribution of the LPSS framework and access control model is to provide a means to share resources between devices in a secure and managed way without using a data connection where
external costs are incurred. This chapter presents the high-level LPSS framework, its components and an overview of its operation.

6.2 The local personal smart space framework

The aim of this research is not to reinvent a personal smart space, but to rather define a LPSS framework based on a solid and well-accepted framework by considering current research on PSS frameworks, together with the identified functional and access control requirements. The functional requirements identified that devices need to be able to connect to each other regardless of the device manufacturer, the creation of LPSS groups need to be supported, devices should be able to identify and connect to those who come into range, and the LPSS application should be easy to use and require minimal setup. Figure 6.1 gives a high-level overview of the layers and components of the LPSS framework which interact in order to provide necessary functionality to the LPSS environment.

The architecture of the framework is based on the PSS framework, and is designed with an Android implementation in mind. To the far left of Figure 6.1, the position of Android components namely the Android Linux kernel, application framework and Android application layers are indicated with respect to the LPSS framework. As shown, the majority of the framework operates within the Android application layer, with both the device-specific application and device-independent layer rolled into an Android application, which is to be implemented as a prototype.

A number of key functions of the LPSS application require the use of the identity manager, client manager, network manager and context monitor from the Android application framework underneath it. None of the functions of the framework operate within the kernel space as this requires the rooting of the device.

At the bottom right of the diagram are a set of connected devices which each have the LPSS application installed on. These devices connect to one another through a wireless peer-to-peer service such as Wi-Fi Direct. The LPSS application enables devices to connect to each other and share information and services.
The next section describes the device-specific application layer.

6.2.1 The Device-specific application

The device-specific application layer has three primary components supported by the Android application layer, namely the user interface, resource manager and the device storage block. This layer relies on underlying functionality such as networking capabilities and access to context sensors that are exposed by the underlying operating system.

The user interface supports interaction with the user of the device and provides a means to manage connections, groups, policies, files and resources. The resource manager is a middle-man in the framework that is responsible for interfacing with the device storage block for storing and retrieving various types of information as needed. The final component is the device storage block which is a representation of the various information storage components stored on the device’s physical storage medium. Here, user information such as pictures, documents and preferences are stored for use.

The manner in which resources are shared is described next.

a) Resource sharing

The focus of the LPSS application is the sharing of resources within a specific LPSS, and between different LPSSs. If the LPSS application supports a fine-grained level of sharing it would result in an administrative nightmare for the owners of devices. For example, if a device owner selects a few pictures to be shared between all his devices, it would be difficult to manage and keep track of. For this reason, shared folders are created where resources such as pictures and other files can be stored, from where it can be securely shared with other devices in the LPSS. A requirement is that all related
LPSS devices have a shared folder with the same name. For example, John creates a shared folder for personal documents, named *personal_docs*, to share between his personal devices in *group_J* and a shared folder *pictures* to share with his family in *group_home*. Each of the devices in the groups need to have the group's shared folders. The shared folder is a group of objects called an *object group*. In the LPSS framework, access control is performed on these object groups instead of individual objects. Thus, it is only necessary for each device to have the object groups required by the groups to which they are members. This is illustrated in Figure 6.2.

![Figure 6.2: Resource sharing with object groups](image)

The next section describes the *device-independent* layer.

### 6.2.2 The Device-independent Layer

Below the *device-specific application* is the *device-independent layer* which houses the LPSS framework’s main functionality in the form of *security management*. In this layer, the five access control requirements that were identified are addressed. Access to resources must be strictly enforced on each smart device, access control policies should be straightforward for group and device owners to implement and maintain, policy rules of different LPSSs need to be processed together, and if conflicts arise, they should be solved appropriately. Finally, context and trust should be included when access control decisions are made.

The access control components of the LPSS framework is located in the security management block of the device-independent layer, highlighted with darker borders in Figure 6.3. The primary security management components are the *access control manager*, *trust manager*, and the *policy manager* which are each discussed in greater detail in the sections that follow. The first component to be discussed is the policy manager.
6.2.3 The Policy Manager

The Policy Manager supports the specification of access control policies, enables the sharing of policies between group members, and supports propagation of policy changes to all group members. Next, the group management device, policy creation, policy types and policy propagation are discussed.

a) The group management device

The device that creates the LPSS group is called the group management device. This is the only device in a particular LPSS that can create and alter policy rules for that group. As there is no central policy store, this ensures consistency when policies are created for the group.

b) Policy creation

To create a policy, the policy manager on the group management device is used by the owner of the device to create and specify access control policy rules. A policy is defined by using an easy to use user interface exposed by the LPSS mobile application. The policy manager supports a flexible, descriptive and well-defined policy language that takes into consideration resources, groups, subjects, context information and trust values.

c) Policy types

In a LPSS environment, two types of policies are used to secure the resources exposed by the LPSS application.

- **Local policies**: Local policies define the personal preferences and restrictions of the owner of a set of devices such as John.
- **Global policies**: Global policies define the preferences and restrictions of a more general group such as a family that the device may belong to. In this case, members of the group may be devices with different owners.

![Figure 6.4: Local and global groups](image)

Figure 6.4 shows group_J, a local group of smart devices belonging to John. These devices also belong to the global family group, group_home. As shown, there are two policies that are applicable to the three smart devices in group_J (device_J1, device_J2 and device_J3) namely policy_LH and policy_GH.

Both types of policies must be kept up-to-date on each of the three member devices, discussed next.

**d) Policy propagation**

As each device is responsible for making its own access control decisions, it is imperative that each device has its own copy of all access control policies. Any changes made by the group management device to the access control policy must result in an update to the copy of each member. To achieve this, the group management device acts as a Master Policy Administration Point (MPAP). Any policy changes can only be made on the master device. The remaining slave devices in the group will serve as limited Policy Administration Points (PAPs) that can only verify if their policies are up to date. The problem of having different versions of a policy is eliminated as only one device per group can support changes. Should a policy on a slave device be out of date, the slave device will receive the latest version from the master device as depicted in Figure 6.5.
However, the master device is not guaranteed to always be present when policy changes occur. More than likely, slave devices interact with one another to share resources at some point. To prevent the presence of outdated policies on slave devices, two solutions are defined.

- A rule is set to ensure that policy updates on the master device may only be made if all slave devices are connected and present in the LPSS. Thus, all changes are propagated directly to slave devices as changes are made. This rule can easily be applied in the case of John and his group of devices, or those of his family.
- As it would be highly unlikely that all devices in a group are present at the same time, changes are uploaded to the cloud, from where they are pushed to all slave devices. In this case, devices need to be connected to access points to be able to access cloud resources. An updated policy only becomes active when all slaves have successfully been updated.

A group may choose any of these two solutions when the group is configured for the first time, depending on their specific requirements. The cloud propagation solution may be preferred as it is the most convenient one, however it lends more to a centralised policy approach and may incur additional costs so the prior option is favoured by this research.

The next section discusses how access control is enforced by the Access Control Manager.

### 6.2.4 The Access Control Manager

The Access Control Manager (ACM) is responsible for all access control decisions made for LPSS requests. The ACM is a policy enforcement and decision point that intercepts all access control requests for resources. The architecture of the LPSS access control manager, presented in Figure 6.6 below, and the components supporting its operation are briefly detailed next.
Figure 6.6: Architecture of the ACM

Normally, applications requests to access files and resources are processed by the Android reference monitor. As found by this research, this provides limited control over resources.

The LPSS framework adds another layer of protection above the native Android reference monitor. Figure 6.6 shows the Access Control Manager (ACM) present in the application layer of Android. With an added layer of protection for resources protected by the LPSS framework, there are now three steps in the process of requesting access to those resources. The steps involved are the initial request from the application (1), decision making (2) and underlying access control (3).

In the first step, depicted as (1) in Figure 6.6, an application requests access to resources as it normally would. If the resource is under LPSS framework protection, the ACM's Policy Enforcement Point (PEP) intercepts the request.

In step 2, the PEP requests an access control decision from the Policy Decision Point (PDP). The PDP relies on the information provided by the policy, trust and context managers. The policy manager, provides the PDP with all applicable rules relating the subject to the resource in question. The trust and context managers, provide the PDP with trust and contextual information respectively. The PDP also relies on Android's native identity and client managers to provide additional information about the subject if it is a remote device, such as one connected via Wi-Fi Direct or some other medium. With all of the required information in-hand, the PDP renders a verdict which is directed by the access control model.
If the PDP denies the resource request then the application is notified by the PEP and the operating system is not made aware of the request. If the PDP permits the request then the PEP passes the request on to the underlying Android reference monitor in step 3. The reference monitor performs its UID and permission checks to render a verdict as the operating system normally would. If the Android reference monitor decides that either the UID or permissions of the application are not correct it will deny access even though the ACM has permitted access. Alternatively, access is granted and the application gains access to the resource requested.

The key concept to be noted is that the ACM provides an additional layer of protection for resources under LPSS framework protection and is reliant on other components to make informed decisions, based on available information and in line with policy.

The trust and context managers and their component interactions are briefly summarised next.

**a) Trust manager**

The Trust Manager provides the ACM with trust information stored for the smart device making a request. The information is presented in the form of a trust level such as trusted or un-trusted. When group owners and members invite smart devices to join their groups, they can assign an arbitrary trust level to the smart device that can be taken into consideration by the ACM when making access control decisions. The trust manager relies on the context manager to report any deviant behaviour so that trust levels can either be automatically assigned or altered during operation.

**b) Context manager**

The Context Manager provides the ACM with up-to-date context information such as time-of-day. The context manager either gets the current information from the underlying context monitor or returns computed information from device storage.

These are the primary components of the LPSS framework which operate in unison to provide a more secure means to manage and share files over a peer-to-peer LPSS. The device-independent layer is the security hub of the framework that is responsible for access control to files, in line with group owner policy. This layer requires the underlying functionality exposed by the operating system and a means to interact with the user, which is provided by the device-specific application.

### 6.3 Conclusion

This chapter introduced the structure of the Local Personal Smart Space framework. Thereafter, the device-specific application and device-independent layer were discussed with a brief overview of each of their most notable components. The layers of the LPSS framework were plotted against the layers of the Android operating system to highlight the reliance on the existing components provided
by the underlying operating system. Finally, the access control components related to the access control manager were presented and discussed. The access control manager does not replace or alter the stock Android reference monitor in any way, but rather augments security for resources under LPSS framework protection by providing an additional layer of security. The access control manager seeks to make informed decisions based on user-defined policy and is supported by various components that provide it with up-to-date information. The LPSS framework addresses both the function and access control requirements defined in this research.

The next chapter takes the access control requirements defined throughout this research and defines the LPSS access control model.
Chapter 7

The Local Personal Smart Space  
Access Control Model

7.1 Introduction

In order to provide secure resource sharing between Local Personal Smart Space (LPSS) group members, the LPSS framework was designed with strict access control and flexible policy management in mind. Chapter 6 provided an overview of LPSS framework's Access Control Manager (ACM) and how each of its constituent components interact to provide strict access control and flexible policy management. During the course of this research there were a number of requirements identified for the access control aspects which the LPSS framework must implement to achieve the goals of this research.

This research identified that having both local and global policies to govern personal and impersonal groups was an area lacking in current smart space research. A means was thus sought to support both strict access control and personal preference by using local and global policies. The creation of local and global groups is thus a mandatory aspect of the LPSS framework to provide users with a means of sharing content between the personal devices in their local groups and any other devices in their global groups. In order to achieve this goal, and provide a research contribution, the ACM of the LPSS framework facilitates the creation, management and processing of local and global policies. Due to having multiple sets of local and global policies stored on any device, it is also required that the ACM resolve with any conflicts which should arise during the combination of local and global policies, to ensure that security is not compromised.

To reduce the amount of work required by the group owner, it was decided that the use of object groups would simplify policy creation by reducing the amount of effort required to specify which files are under the protection of the LPSS framework. Additionally, context and trust-based access control mechanisms were identified as a flexible means to define additional constraints imposed on the resources under LPSS framework protection.

This chapter defines the access control model used in the LPSS framework's ACM and details its specification in logical form. Section 7.2 of this chapter presents a number of access control model questions to guide the reader through the structure of this chapter. Section 7.3 formally defines the
access control model used in the LPSS framework and validates it through the use of scenario-based examples. The chapter is then concluded.

7.2 LPSS access control model goal
The main goal of the LPSS access control model is to ensure that only devices that have been granted permission to access resources in shared LPSS object groups are granted access to those object groups, whether they are local or global in nature. A number of questions are now identified to help guide this research as follows:

1. How are LPSS groups created?
2. How are members of LPSSs granted access to shared folders?
3. How is context-based access control defined?
4. How are conflicts solved between rules from different LPSSs policies, applicable to the same resources?
5. How is access granted to subjects who are not part of an LPSS?

The next section presents the LPSS access control model.

7.3 LPSS access control model
This research proposes an access control model for LPSS applications that is defined by local and global policy rules, context constraints and the trustworthiness of requesters and trust requirement of resources. In a LPSS, user and device identification is first performed. All access permissions and rules for both local and global groups are stored in the policy store of each device and accessed via the policy manager.

Next, a number of concepts required for the specification of the access control model used by the ACM is given and the manner in which access control rules are specified is described.

7.3.1 LPSS access control model fundamental concepts
In order to fully define the access control model there are a number of fundamental terms that must first be defined. These terms are defined as follows:

a) Set of user devices
A device owned by a user, \( U \), is denoted by \( device_i \) and is any form of electrical equipment capable of connecting to a LPSS through an installed LPSS application, facilitated by a wireless networking technology such as Wi-Fi Direct.
A user, $U$, is a human being who owns, manages, or operates any number of smart devices which are capable of connecting to a LPSS in order to share information resources such that:

\[ \text{devices}_U = \{ \text{device}_0, \text{device}_1, \ldots, \text{device}_i \} \]

-where $\text{devices}_U$ is the set of devices operated by $U$ and $i$ is an arbitrary number reflecting the number of devices operated by user $U$.

b) Local group
A set of devices owned by a user, $U$, which have been grouped such that:

\[ \text{group}_L = \{ \text{device}_{U0}, \text{device}_{U1}, \ldots, \text{device}_{Ui} \} \]

c) Global group
The group, $G$, is a set of devices, not necessarily owned by the same user, added to a group such that:

\[ \text{group}_G = \{ \text{device}_0, \text{device}_1, \ldots, \text{device}_i \} \]

d) Object group
An object group, $OG$, is a shared folder consisting of a set of objects that are under the protection of the LPSS such that:

\[ OG = \{ \text{object}_0, \text{object}_1, \ldots, \text{object}_i \} \]
-where $\text{object}_0-i$ are resources stored on a device.

e) Group member
A group member is any device, $K$, that is a member of a group, $L$, such that:

\[ \exists \text{device}_K \mid \text{device}_K \in \text{group}_L \]

f) Group management device
The group owner, $GO$, is a user who is capable of performing management of a group, $L$, through a single group management device, $device_{GO}$, such that:

\[ \exists device_{GO} \mid device_{GO} \in \text{group}_L \]

The next section uses the terms defined above to formally specify the logical structure of the access control rules used within the ACM.

7.3.2 Access control rule specification
LPSS access control rules need to be specified in a language that is flexible and expressive and supports the closed policy. For this purpose, a logical specification language, the Authorisation Specification Language (ASL) (Jajodia et al., 1997) is chosen as a foundation to specify rules. It
should be noted that access control rules are specified in ASL, but implemented in the prototype in XML (Graham & Quin, 1999) and processed in code.

Next, the set of predicates used by this research are given.

- `group_man(s, go)` Defines the management device (s) of the group (go).
- `in(s, g)` Adds a subject (s) to a group (g).
- `in(o, og)` Adds an object (o) to an object group (og).
- `trust_subject(s, t)` Defines the trust level (t) of a subject (s).
- `trust_object(og, t)` Defines the trust level (t) of an object group (og).
- `Ctx(attr_1, ..., attr_j)` Defines a context constraint with attributes where Ctx is the name of the constraint.
- `cando(og, s, ±a)` Is an access control rule to grant or deny access to a subject (s) to perform an action (a) on an object group (og).
- `dercando(og, s, ±a)` Is the access control rule derived from the system.
- `do(og, s, ±a)` Is the accesses that must be granted or denied after conflict resolution has been done.

As both a positive and negative authorization can be defined, it should be specified how conflicts should be treated when many authorization are associated with the same object group. The conflict resolution policy to be applied is that of denials take precedence. To ensure that a decision is made when conflict exists, the closed policy is used as a decision policy. The closed policy denies an access if there exists a negative authorization for it, and allows it otherwise.

**Denials take precedence in conflict-resolution with closed policy:**

\[
do(og, s, +a) \iff \neg \text{dercando}(s, work\_docs, +a) \land \text{dercando}(s, og, -a)
\]

This policy states that the subject can be granted access to the object group if there is a positive authorization, and a negative authorization cannot be derived by the system.

LPSS access control rules are described next.

**7.3.3 Access control rules**

The specification of access control rules to meet the goal of the access control model is now presented. The following examples are used to highlight the access control model’s ability to apply both local and global policies using rule combinations with context and trust constraints.

Each of the access control questions posed are now answered by specifying access control rules based in the example scenario as follows:
• Definition of a LPSS group
• Local and global access control policy rules
• Context-based access control rules
• Access control policy conflict resolution
• Trust to grant access to non-member subjects

a) Definition of a LPSS group
To be able to create a group, either local or global, a group management device is first defined. Thereafter, devices are invited to the group by the group owner and added as members.

John creates a group, \( \text{group}_J \). The device he used to create the group, \( \text{device}_J \), is automatically assigned as the group management device. The rule to accomplish this for a subject, \( s \), and a group, \( g \), is defined by:

\[
\text{group}_{\text{man}}(\text{device}_J, \text{group}_J)
\]

Once John has created the group, he can invite his other devices \( \text{device}_J \) and \( \text{device}_J \) to the group with the group management device before he can start sharing files and resources. Once John provides the group credentials to each of his devices, they are added to the group by the rule for device, \( s \), and group, \( g \), as follows:

\[
\text{in}(\text{device}_J, \text{group}_J) \\
\text{in}(\text{device}_J, \text{group}_J) \\
\text{in}(\text{device}_J, \text{group}_J)
\]

b) Local and global access control policy rules
After the LPSS group has been created, an object group such as \( \text{pictures} \) is defined on the group management device that will be used by the group to share resources. Each device has an identically named shared folder created automatically. An access control rule needs to be defined to grant all members of the group access to the shared folder.

\[
dercando(s, \text{og}, +a) \Leftarrow cando(s, \text{og}, +a) \land \text{in}(s, g)
\]

An access decision can be derived if the subject is granted permission to perform the action, and the subject is in the LPSS group.

For example, John grants access to all his devices in \( \text{group}_J \) to the object group as follows:

\[
dercando(s, \text{og}, +a) \Leftarrow cando(s, \text{pictures}, +a) \land \text{in}(s, \text{group}_J)
\]

John adds a picture \( \text{pic}_1 \) to the object group, thereby making it available to all devices in the group.

\[
\text{in}(\text{pic}_1, \text{pictures})
\]
Thus subjects which are not members of group_J are denied access to the object group. Figure 7.1, illustrates the local access control policy of group_J. The devices that are part of the group are defined, an access control rule grants access to subjects to access pic_1, and a decision rule states that access can be granted to pictures if the subject in group_J.

It should be noted that the access control policy rules for a global group are created similarly. After each of the family members have set up their personal groups, John creates a global group, group_home, for his family. One of his devices is set as the group management device. He then invites family devices to the group and defines access to the object group family_files of the family. The group contains all devices in the family, including his wife Mary’s two devices and his son Peter’s one device.

No group member may be allowed to join a group without the explicit permission of the group owner. This is ensured by having password protection on the group management application and when joining a group. Thus John, as group owner of the family group has full control over the members of this global group from the designated group management device. This means that not all members of a local group may necessarily be members of a global group.

| group_man(device_J1, group_J) |
| in(pic_1, pictures) |
| in(device_J1, group_J) |
| in(device_J2, group_J) |
| in(device_J3, group_J) |
| cando(s, pictures, +a) |
| dercando(s, og, +a) ⇐ cando(s, pictures, +a) ∧ in(s, group_J) |
| do(s, og, +a) ⇐ dercando(s, og, +a) |

Figure 7.1: Local policy for group_J

A device such as device_J1 is granted access to two object groups and their content, based on its membership of two groups namely group_J and group_home. The resultant object groups are pictures for the local group and family_files for the global group as depicted in Figure 7.2.
In the next scenario, context is added to access control decisions.

c) Context-based access control rules

The basic access control rule is now altered by the addition of a context constraint. LPSS access control should address the context of the environment to provide better control over who can access resources using environmental and other constraints. For example, John would like to restrict the family’s access to the family_files object group of the family to just a few hours a day. He adds a context constraint to the global group_home access control policy, as follows:

\[
\text{do}(s, \text{family_files}, +a) \iff \text{cando}(s, \text{family_files}, +a) \land \text{in}(s, \text{group_home}) \\
\land (15:00 \leq \text{time_of_day}() \leq 20:00)
\]

The global policy rule denies access to any file in family_files object group if the subject is not in group_home, or the request is out of hours, denoted by the environmental context returned by time_of_day(), or if there is no rule allowing access to it. Appendix A of this document shows the complete policy for group_home.

d) Access control policy conflict resolution

Local and global policies may contain conflicting authorizations. These conflicts may exist between a local and global policy, two local policies, or two global policies that are applicable to the same object groups.

For example, a global policy may deny access, and needs to override another global policy that may grant access. At work, John and his colleagues are part of a global LPSS named group_work. Here,
documents are stored that are shared between group members. For example, *work_doc1* and *work_doc2* were shared with all members at a recent meeting at work in the object group *work_docs*.

John inadvertently gives all members of *group_home* access to *work_docs* by specifying an authorization in the *group_home* policy as follows:

\[
\text{cando}(s, \text{work_docs}, + a)
\]

To protect the *work_docs* object group, the group owner, Bill, who is the manager of the department where John works, sets an additional constraint. Bill explicitly adds that if a subject is not part of the *group_work* LPSS, access should be denied.

\[
\text{dercando}(s, \text{work_docs}, -a) \iff \text{cando}(s, \text{work_docs}, + a) \land \neg \text{in}(s, \text{group_work})
\]

When access control rules are processed, the *global* policy of *group_work* explicitly denies access if the subject is not part of the *group_work*. The closed policy thus denies access to the resource.

It should be noted that a local policy rule may be set to override a global policy rule in the same manner.

e) *Trust to grant access to a non-member subject*

John would like to ensure that his son Peter is prevented from giving himself, or his friend Mark access to resources that John as group owner would not want to allow. Should Peter add his friend Mark's *device_MK* to his local group, *group_P*, Mark can view all files in his object group *peter_stuff*, Mark’s device could potentially also access all other resources that Peter can view if he inadvertently is made a member of *group_home* as Peter was given the credentials by John.

John can add a rule to the global policy of *group_home*, of which Peter is a member, to deny access to resources for anyone in Peter's group. The new global rule is defined by:

\[
\text{dercando}(s, \text{og}, -a) \iff \text{cando}(s, \text{og}, + a) \land \neg \text{in}(s, \text{group_P})
\]

Unfortunately, this rule also prohibits Peter from accessing any object groups of the family, making the rule impractical.

To be able to grant subjects access to object groups, without requiring them to be members of the LPSS group, the concept of a trust level is used. A trust level is assigned to both subjects and objects so that access can be granted.

A trust level is set for an object group belonging to the family LPSS named *dec_holidays* as follows:

\[
\text{trust_object}(\text{dec_holidays}, \text{to})
\]
where $t \in \{completely\_trusted > trusted > untrusted\}$

A trust level is assigned to Mark's $device\_{MK_1}$ by John or it is computed over time using the Trust Manager of the framework. In either case, the resultant value can be assigned to $device\_{MK_1}$ as follows:

$$trust\_subject(device\_{MK_1}, ts)$$

where $ts \in \{completely\_trusted > trusted > untrusted\}$

A global rule for the $group\_home$ LPSS is defined as follows:

$$dercando(s, og, +a) \iff cando(s, dec\_holidays, \pm a), (trust\_subject(s, ts), trust\_object(dec\_holidays, to)) \land ((ts > to) \lor (ts = to))$$

In order to grant access, the trust level of the subject must be greater than or equal to the trust level of the object. This means that Mark does not need to be a member of the $group\_home$ LPSS to be able to see these pictures. If access to him needs to be revoked, his trust level can be changed. The global policy gives John greater peace of mind about what his child and what his friends are able to do with their smart phones.

Figure 7.3 shows the revised rules for the $group\_home$ global group. It is of key importance to note that there is no rule explicitly adding Mark's device to the group. Thus, Mark's device when accessing $dec\_holidays$ is not subject to the group rule requiring group membership, only a sufficient trust level of trusted. Thus, he can't access $family\_files$ by the can access the $dec\_holidays$ object group.

Appendix A of this document contains the collated rules for each group used in these examples.
This chapter defined the access control model for use in the access control manager of the LPSS framework. The access control model was defined using a number of scenarios to highlight its operation through various interactions. The model itself, caters towards simplicity by making use of object groups for rule specification. The extra simplicity allowed by the object groups, however, does come at the expense of flexibility and fine-grained control over resources. The use of object groups allows the group owner to more easily specify which resources are under the protection of the LPSS framework and define rules that govern them. There is no way allowed for individual resources to have rules created for them. There is an advantage and a disadvantage to this. If there are no dedicated, experienced administrators it makes the job of specifying access control rules a lot easier for the group owner to manage. However, the loss of flexibility may be inexcusable to those who demand fine-grained control. This compromise was made because of the nature of mobile smart devices. They are used by a denomination of user and not just the initiated or security conscious and it is better to have some form of security than none at all because the user was incapable of using the

```prolog
    group_man(device_J1, group_home)
    in(file_a, family_files)
    in(file_a, dec_holidays)
    in(device_J1, group_home)
    in(device_J2, group_home)
    in(device_J3, group_home)
    in(device_M1, group_home)
    in(device_M2, group_home)
    in(device_P1, group_home)
    trust_object(dec_holidays, "trusted")
    trust_subject(device_MK1, "trusted")
    cando(s, family_files, +a)
    cando(s, dec_holidays, +a)
    dercando(s, og, +a) ← cando(s, family_files, +a) \ in(s, group_home)
    \ (15:00 ≤ time_of_day() ≤ 20:00)
    dercando(s, og, +a) ← cando(s, dec_holidays, ± a).
    \ (trust_subject(s, ts), trust_object(dec_holidays, to )
    \ (ts > to) ∨ (ts = to))
    do(s, og, +a) ← dercando(s, og, +a)
```

Figure 7.3: Revised global policy for group_home using trust

### 7.4 Conclusion

This chapter defined the access control model for use in the access control manager of the LPSS framework. The access control model was defined using a number of scenarios to highlight its operation through various interactions. The model itself, caters towards simplicity by making use of object groups for rule specification. The extra simplicity allowed by the object groups, however, does come at the expense of flexibility and fine-grained control over resources. The use of object groups allows the group owner to more easily specify which resources are under the protection of the LPSS framework and define rules that govern them. There is no way allowed for individual resources to have rules created for them. There is an advantage and a disadvantage to this. If there are no dedicated, experienced administrators it makes the job of specifying access control rules a lot easier for the group owner to manage. However, the loss of flexibility may be inexcusable to those who demand fine-grained control. This compromise was made because of the nature of mobile smart devices. They are used by a denomination of user and not just the initiated or security conscious and it is better to have some form of security than none at all because the user was incapable of using the
systems in place. The addition of the default rule aids in the latter to provide a solid baseline for security in any LPSS group while still providing simplicity.

The next chapter documents the SmartNet prototype, to validate this research.
Chapter 8
SmartNet: The LPSS Framework Prototype

8.1 Introduction
This chapter introduces the SmartNet prototype to showcase the completed Local Personal Smart Space (LPSS) framework. The SmartNet prototype demonstrates the operation of the LPSS framework in an Android environment using Wi-Fi Direct to make peer-to-peer connections. The SmartNet prototype serves as the culmination of this research through the embodiment of the design of the LPSS framework and access control model as described in Chapters 6 and 7. The main focus of the SmartNet prototype is to illustrate the function of the Policy Manager and Access Control Manager which enables the creation of groups of devices which can use object groups to specify how files are shared between group members.

The SmartNet prototype supports policy rules which limit who has access to object groups by defining policies which can have context or trust constraints imposed on them. For simplicity sake, the SmartNet prototype also supports having a default rule in place in case a user does not want to impose any further restrictions on their object groups or only requires a solid foundational level of security. The policies that are created are implemented in the form of local policies which reflect the personal preferences of the owner of a device, or global policies which are for impersonal groupings of devices. The SmartNet prototype finally demonstrates how access control decisions are made using policies from different groupings of devices and how conflicts are resolved.

This chapter is broken up as follows: Section 8.2 introduces the SmartNet prototype and the goals of its operation. Section 8.3 documents the initial setup of the SmartNet prototype. Section 8.4 discusses the creation of LPSS groups and object groups. Section 8.5 details the use of the SmartNet prototype for rule creation and Section 8.6 details policy processing and conflict resolution. Section 8.7 provides observations about the prototype, with an analysis of what was successful and unsuccessful and what can be improved upon in future. Finally, the chapter is concluded.

8.2 The SmartNet prototype
This section introduces the reader to the SmartNet prototype by detailing its utilised technologies, security aspects and operation in the sections to follow. The technologies used in SmartNet are discussed first.
8.2.1 Technologies
The SmartNet prototype was built using the Android Studio integrated development environment using XML (Android, 2015y) for the layout and aesthetics of pages and Java (Android, 2015x) for the background code. The Android SDK, used for the prototype, targets the Android 4.4 KitKat (Android, 2015aa) runtime to support backward compatibility should the need arise during testing.

The devices used for the testing of the prototype are a Samsung smart phone and tablet both running Android Lollipop 5.0 (Android, 2015z). All database functionality is done through SQLite (Android, 2015ab).

The SmartNet prototype was build to connect to peer devices through Wi-Fi Direct as implemented in the Android operating system by the Wi-Fi Peer-to-peer manager (Android, 2015g). Each device can maintain a Wi-Fi Direct connection with multiple devices. The SmartNet prototype is built on top of this connection to provide smart space functionality to devices with the application installed and to utilise the secure underlying connection.

The security mechanisms built into SmartNet are discussed next.

8.2.2 SmartNet security
When considering the SmartNet prototype as an application required to provide secure and managed file sharing, it is imperative to consider the security aspects relating to the application and the way in which content is protected that is stored on the device, and sent to other devices. The focus of this research is on access control which is provided by the access control manager built into the LPSS framework's security management block. However, the SmartNet prototype's access control manager does not stand alone. It is required that a high degree of security is achieved on the device to ensure that access to resources is not allowed by circumventing the LPSS framework. This section discusses the use of authentication, confidentiality and integrity of files and resources under the protection of the LPSS framework. Each of these are discussed in the sections that follow.

a) Authentication
The SmartNet prototype performs authentication at two stages of its operation. The first is when the application is started and the user is prompted to enter their credentials before they can proceed to any of the LPSS management pages. This prevents deviant behaviour should the device be lost or stolen.

The second stage where authentication takes place is when a group invite occurs. The invited party is required to provide the group password which is provided by the group owner is the same manner as a Wi-Fi access point key when friends or family come over to visit. This prevents deviants from inviting themselves to someone's groups to get access to their files if they gain access to the device
after the user has logged in to the SmartNet prototype but before they have logged out or the auto-
logout has occurred.

b) Confidentiality and integrity of stored and transmitted data
The final security concern to be discussed is the confidentiality and integrity of the files that are stored
and transmitted on the devices using the SmartNet prototype. Each Android application has the files
associated with it controlled through the use of storage encryption which ensures that the only
application which can gain access to the files stored for the SmartNet prototype is itself (Android,
2015a)(Android, 2015b). This ensure both the confidentiality and integrity of the files protected by the
LPSS.

When the sharing of files over a wireless medium is concerned the SmartNet prototype ensures the
confidentiality of files through the use of strict access control enacted by the access control manager
of the LPSS framework. This ensures that files are only sent to those devices which the group owner
has allowed through the use of local and global policies. Furthermore, the policy combination which
utilises a closed policy approach further ensures that insecure rules cannot be created by group owner
to gain access to another's files.

When considering the actual sending of files over a wireless medium, the Wi-Fi Peer-to-peer manager
(Anddroid, 2015a) built into Android which facilitates all Wi-Fi Direct communication has WPA2
encryption built into its communication mechanisms (Wi-Fi Alliance, 2003). This ensures that
confidentiality and integrity are not lost if packets are intercepted over the wireless medium.

The next section details how the SmartNet prototype is used, and this chapter's strategy to showcase
its functionality.

8.2.3 Operation of the Smartnet prototype
The discussion of the operation of the SmartNet application follows the manner in which the LPSS
access control model was presented in Chapter 7 by answering the following questions:

a) How is the SmartNet prototype installed, configured and navigated?
b) How are groups of devices and object groups created?
c) How are devices invited to groups?
d) How are objects added to object groups, and groups managed?
e) How are access control rules created and enhanced with context?
f) How are access control rules enhance with trust to allow non-group member to access object
groups?
g) How does rule processing and conflict resolution take place?
Questions (b – f) illustrate the operation of the Policy Manager, whereas question (g) describes the processes of the Access Control Manager.

Policy Questions (b - d) are answered in Section 8.4 where object groups and groups are created, managed and devices invited. Questions (e) and (f) are both answered in Section 8.5 where rules are discussed. Finally, question (g) is answered in Section 8.6 where rule processing and conflict resolution are discussed.

The next section introduces the reader to the SmartNet prototype by documenting its initial setup to answer question (a).

**8.3 Basic setup of the SmartNet prototype**

The SmartNet prototype is installed using a package manager as it does not require the device to be rooted. This provides the advantage of having the ability to reach the widest audience without having to make modifications to device's operating system software. The user, however, must first enable the installation of applications from unknown sources from the device application setting. Alternatively applications can be installed remotely by enabling USB debugging which is used for application testing in Android Studio.

Prior to operation of the SmartNet prototype, there is some initial setup required by device owners and LPSS group owners. The following sections detail what a user has to do to initially setup the application and gain access to the secure functionality. Once a user is logged in they can begin to secure resources, create groups and rules and invite devices to their groups. Each of the sections that follow build upon the steps detailed in the previous steps to illustrate the usage of the SmartNet prototype based upon the examples of Chapter 7.

**8.3.1 Initial setup and home pages**

Upon first-time start-up of the SmartNet prototype, the user is presented with a prompt to create an account for their device in order to secure access to the application. Figure 8.1 shows the initial screens that a user is presented with upon starting the SmartNet prototype.
When the SmartNet prototype starts, the user logs in with a username and password, depicted in Figure 8.1(B). If this is the first time the user uses the application, he/she needs to create an account shown in (A). Selecting the three vertical dots located at the top-right of every page of the application after logging in, sends the user to the Settings page depicted in (C) where the user can make basic changes to the way in which the SmartNet prototype operates. Functions include the ability to prevent the device from storing the SmartNet prototype username, changing the theme, setting time and date format and specifying if and when files should be automatically requested from connected devices.

Upon successful login, the user is redirected to the home page. The home page consists of three separate pages which can be navigated between by swiping side-to-side on the screen. The three home pages are depicted in Figure 8.2.
Figure 8.2: Home pages

Figure 8.2 (A) shows the Devices page, which is the initial landing page after login. Swiping to the left sends the user to the Groups page (B) and swiping right sends the user to the Files page (C).

Each of the home pages of Figure 8.2 are detailed as follows:

a) Devices page

Here, the user manages connections to all Wi-Fi Direct devices. The devices page has three important components, namely; the Wi-Fi Direct peers list, Peer log and Scan for peers button.

At the top of the devices page at (A), the user is presented with a list of Wi-Fi Direct peer devices that are in range and broadcasting. This list shows the name of the device, followed by its MAC address and current status. To the right of the text, there are two round icons that depict the device's Wi-Fi Direct group and connection status. The various possible statuses are depicted in Figure 8.3.

Figure 8.3: Device states

Should the user press and hold on one of the devices in this list, a drop-down list will appear with six options. The options are; info, connect\disconnect, invite to group, send message, send file and assign trust. These are detailed as follows:
1. **Info:** Presents the user with a new page that provides some additional details about the device and its capabilities.

2. **Connect|disconnect:** Invites the device to form a peer-to-peer connection and begin sharing files. If the device is connected, pressing this button will disconnect the device and tear down the Wi-Fi Direct connection.

3. **Invite to group:** Allows the group management device to select a local or global group to invite the other device to join.

4. **Send message:** Allows the user to send an instant message to the device. The message appears in the peer log at the bottom of the devices page.

5. **Send file:** Allows the user to manually specify a file to be sent to the device. This is used for both testing and when the user does not wish to create a group and assign files to the protection of the LPSS and invite the other device to the group just to send a single file. Note that access control is still performed on files that are under LPSS protection.

6. **Assign trust:** Allows the group management device to assign a trust level of *untrusted, low-trust, or trusted* to a device for use in trust-based access control rules.

Below the peer list is the **Peer log**, which provides the user with a log of all activity that is of importance. This includes connection information such as invites, connections and disconnects. Any messages that the device receives and the progress of file sending or receiving is also presented here.

The **Scan for peers** button is a manual way to initiate a Wi-Fi Direct peer scan. Normally the scan is done upon initial login and is then automatically done in short intervals.

**b) Groups page**

From the **devices** page, the user can swipe left to get to the **groups** page at (B). Here, the user can create and manage both local and global LPSS groups. The **groups** page has three components, namely; the **My device** section, the **Local** and **Global groups** lists and the **Create a group** button.

The **My device** section presents the user with information about their device, such as the device's name, MAC address, status and whether or not it is currently in charge of a Wi-Fi Direct connection.

The **Local** and **Global groups** lists show the user the local and global groups of which the device is a member. If there are no local or global groups the only entry is an instruction to create a group using the **create a group** button. If there are groups, each entry in the list provides the user with the group's name and the name and address of the group management device. Each group has two icons that provide additional information. If the left icon is a grey star, such as the one in Figure 8.3, it indicates the device is not the group management device for the group. Alternatively, a green star indicates it is the group management device. The right icon does not provide information, but tapping it will open the **group management page**.
The **Create a group** button presents the user with a dialog that allows them to create a local or global group and specify its name and credentials.

c) Files page
From the devices page, the user can swipe right to access the files page at (C). Here, the user can manage which files are under the protection of the LPSS framework by assigning them to object groups. The files page has two components, namely; the **file browser** and the **location selector**.

The **file browser** at the top shows the current location in the device's directory structure (*Local Files: /sdcard* by default), which allows the user to navigate to files and open them from within the application. Users can also select files by pressing and holding on a file and a dialog appears asking them which **object group** they would like to assign the file to.

The second component of the file page is the **location selector** drop-down. As given in Figure 8.2, the default setting **File System** is shown, which is the device's storage directory shown in the file browser. If the user taps this, a list containing the File System, followed by all of the current object groups appears. At the bottom of the list is the **create object group** button. Should there be no object groups then the user can select this button to create one, which is discussed in Section 8.4.1.

The next section details the use of the files page to create object groups and assign files to the protection of the LPSS.

**8.4 Object group creation, group creation and management**

After John has logged in, he needs to create an object group to assign files to. He then needs to create a LPSS group to assign the object group to. This section covers the three distinct actions of:

1. Creating an object group and assigning files to it
2. Creating a group and inviting members
3. Managing a group and assigning an object group to its protection

Each of the following sections address the actions above. The creation of object groups and the assignment of files to them is discussed next.

**8.4.1 Object group creation and file assignment**

If no object groups are present on the device, pressing the **location selector** on the files page presents the user with only two options, namely; **file system** and **create object group**. Should the user press the **create object group** button, the user is prompted to enter the desired name for the object group, shown in Figure 8.4 (A). The name entered must be unique, as object groups are used for access control.
rules. Once done, selecting the location selector shows the previous two options as well as the name of group created.

![Image](image.png)

**Figure 8.4: Create Object group, add files, review group**

Once an object group is created, it is now possible to add files to it. The user can browse through the file system and select individual files or directories by pressing and holding. A prompt appears which asks the user to select the desired object group. This is depicted in Figure 8.4 (B). For example, John selects a number of pictures to add to the *pictures* object group. By default, there are no object groups so the user needs to create them as desired. Object groups belonging to other groups also appear here.

Once the user completes the selection of the files to share using the object group, he/she reviews the files by using the *location selector* to select the desired object group. Figure 8.4 (C) shows the populated *pictures* object group. The user can press a file in the object group to open it in an external application such as a picture viewer or press and hold to remove it from the object group. To return to the file browser, the user presses the location selector and select the *file system* button.

Thus, users such as John, Mary and John's manager Bill can create any number of object groups and assign files to them. Next, groups are created.

### 8.4.2 Creating a local or global group

Once a user has created an object group they can navigate back to the *groups page* shown in Figure 8.2 (B) to create a local or global LPSS group. On the groups page, the user selects the *create a group* button. The user is then presented with a prompt to enter the details of the group. The popup is depicted in Figure 8.5 (A).
Figure 8.5: Group creation prompt, group management and object group assignment

The user selects whether the group is *local* or *global* and specifies a unique *group name* and the *administration credentials*. The popup notifies the user of the nature of local and global groups as a local group is for privately owned devices and a global group for any other LPSS-capable devices. Once the user presses *create*, the group is created and is reflected in the *local* or *global group list* on the group’s page. In this manner, John can create his local group `group_J` and the global group for his family to use, `group_home`. This is depicted in Figure 8.5 (B). Likewise, Mary, Peter and Bill can create their own groups, such as the group of John's colleagues, `work_group`. The device creating the group is now the management device for the group as indicated by the green star icon.

The next section details how a device can be invited to a group.

### 8.4.3 Inviting devices to a group

Now that the desired groups are created, it is possible to invite other devices to join the groups. The user returns to the *devices page* and presses and holds on the available, in-range device they wish to invite to a group. The user is presented with the dropdown menu where they select *invite*.

Pressing the *invite* button begins the invite process which has the following steps:

1. **Determine if the devices are connected:** If the two devices are not connected then a connection request is sent to the other device.
2. **Select group:** Once a connection is verified, a prompt appears for the user to select the group they want to invite the other device to. John can invite his personal tablet to his private group or he can invite Mary's personal phone to the family group. This is shown in section (A) on the left of Figure 8.6.
3. **Enter credentials:** The invited device will receive a notification to enter the credentials of the group. This adds an additional layer of protection if John forgets to log out of the SmartNet prototype so that others can’t invite themselves to his groups. Shown in section (B) on the left of Figure 8.6.

4. **Create Group:** Once the credentials are entered by John for his other personal device or by Mary for her personal device, the group information is created on the other device and the device reflects as having joined the group.

5. **Send group policy:** Upon successful group invitation, the management device automatically sends the group's current policy to the new member device in the background. The policy is stored locally for access control decisions. If no changes have been made to the group policy, the default rules are used in the policy.

6. **Setup object groups:** Once the policy arrives, the device can create each of the object groups that are listed in the XML policy document in the background, so that access control can take place.

7. **Begin file send/give access to file list:** For local groups, files are automatically queued to be sent to the new member device in the background. For global groups, the user has the option to select whether they want to get all the files from the group or select the ones that they want through a file browser.

Once steps (1) through (6) are complete the user of the invited device navigates to the **groups page** to verify that the new group shows under the local or global groups list to verify the presence of the new group, as shown in Figure 8.6 (C). If the user can navigate to the **group management page** they will not be able to change policy rules or add or remove object groups from the group as their device is not the group management device.
With object groups and groups created the user can assign the created object groups to the protection of the desired group as shown in the next section.

### 8.4.4 Group management and assignment of an object group to a group

Once local or global groups are created, the user can press on any of them in the group list of the groups page to go to the group management page, depicted in Figure 8.7 (A).
Figure 8.7: Group management and object group assignment

The group management page provides information about the group in question. The group management page has four main components, namely; Group information section, the Associated object groups section, the Associated devices section and the Return to home and Edit group rules buttons. Each are detailed in Sections (a) through (d) that follow.

a) **Group information**

The group information section provides the user with the following information:

1. **Group type**: Indicates if the group is a local or global group.
2. **Group origin ID**: Specifies the unique identifier for the device where the group originated. This is used to test which device is the group management device.
3. **Group origin name**: Provides the name of the device where the group originated.
4. **Created at**: Provides the date and time the group was created.
5. **Auto-download files**: This is exclusively for global groups. This checkbox allows the user to specify if they want the LPSS application to automatically send all shared files to their device. This option can be set by the user regardless of whether they are the group owner or not. This allows a user to select whether they want every file shared in the group (checked) or to be able to select which files they want (unchecked). Local groups default to sharing everything.
b) Associated object groups
The associated object groups section provides the user with a list of all object groups associated with the current group. Figure 8.7(A) shows that there are currently no associated object groups however, when listed, the groups are those specified by the group owner on the group management device. As the current device is the group management device, the add object group can be used to prompt the user to select a previously created object group to assign to the group. This is shown at the centre of Figure 8.7 (B). The right screenshot (C) shows the resultant change to the group management page, with the newly-added object group highlighted.

To remove an object group from the group, the user has to press and hold on the desired object group name and it will be removed from the list and the group without deleting any of the files or the object group itself.

c) Associated devices
The associated devices section provides the user with a list of smart devices that are currently members of the group. Figure 8.7 (A) and (C) show device_f1 and device_f2 as group members with device_f3 as the group owner.

d) Return to home and edit group rules buttons
The return to home button allows the user to navigate back to their home page where they can resume LPSS management activities.

The edit group rules button allows the user to go to group rules page. This button is only visible to the group owner on the group management device.

The next section shows how the group management device can view the group's default rule for every associated object group and make various changes to the rules to complete the group access control policy.

8.5 Group access control policy
Should John press the edit group rules button on the group management page, he is taken to the group rules page. The number of rules shown here is dependent on the number of object groups associated with the group selected. This is true for both local and global groups. For example, Figure 8.8 (A) shows the rule related to the family group, group_home. It should be noted that the user only sees the left side of the figure. The XML policy is stored in encrypted form, hidden on the device and is not user accessible, but it is shown throughout this discussion for the purposes of clarity.
The remainder of this section is divided into four sections namely: The default rule, Altering the default rule, Adding context to a rule and Adding trust to a rule. Each of these actions are discussed in the sections that follow.

8.5.1 The default rule

The Group rules page pictured in Figure 8.8 (A) shows the default rule for the family_files object group in group_home. The default rule is created when an object group is added to the group without requiring user input, to provide a basic degree of protection.

The access control rule definition is pictured in Figure 8.8 (A). The setting of a rule is straightforward and simple. Each rule has four parts that have been labelled 1-4 in red in (A), namely:

1. **Object group name** is the name of the object group to which the rule pertains. In this case the family_files object group.
2. The **share with selector** indicates that the default rule is a positive action relating (1) to (3). By default this is set to share with but can be changed to don’t share with.
3. The **group selector** is set by default to the current group which is group_home. This can be changed to another group or any.
4. The **plus (+) button** allows for additional trust or context parameters to be added to the rule. None are added by default.

![Figure 8.8: The default rule for the family_files object group in group_home](image)

The default rule states that the family_files object group is shared with group_home. Figure 8.8 (B) shows the resultant XML policy containing the default rule for the family_files object group in group_home. Labels 1-4 in red relates to the specification of the rule in (A).
Changes to the default rule are discussed next.

### 8.5.2 Altering the default rule

Figure 8.9 (A) shows how a *share with* (permit) rule can be changed to a *don't share with* (deny) rule with the resultant XML shown in (B). Section (C) shows how the target group can be changed from `group_home` to *any*, or another group (E) with the resultant XML depicted in (D) and (F) respectively.

![Figure 8.9: Altered default rule for the group_home](image)

The next section shows how the basic rule can be enhanced by a contextual constraint.

### 8.5.3 Adding context to an access control rule

Consider the example where John wants to limit the time of day that his family can access `family_files`. The default rule can be enhanced by selecting the *plus button* on the group rules page and selecting *Context* as shown in Figure 8.10 (A). He is prompted to select the type of context to be used (B). Upon selection of *time of day* he is prompted to enter the contextual parameter of *start time* and *end time*, also shown in (B). The resultant access control rule is shown in Figure 8.10 (C) and the resultant policy in (D).
Therefore, should John's family group members try to access the family_files object groups outside of the specified times then they are denied access. Anyone else that is not a member of group_home is also denied.

The next section details how trust can be added to further enhance a rule.

### 8.5.4 Adding trust to a rule

John wants to share the dec_holidays object group with family and people that he trusts, such as Mark, Peter's friend. John creates the object group and enhances the default rule by pressing the plus button where he is prompted to select between a trust and context addition to the rule as shown in the Figure 8.11 (A).

Upon selecting trust, he is prompted to select one of the trust levels, namely untrusted, low-trust, or trusted as shown in (B). He selects trusted. The resultant change is shown in (C) and the resultant XML policy for group_home is shown in (D).

John sets the trust level of Mark and each of his family members at the devices page by pressing and holding Mark, or a family member's, device and then selecting the assign trust option. The user is prompted to assign the trust level from untrusted, low-trust, or trusted similarly to Figure 8.11 (B). The default trust level of every device is low-trust which is not added to the policy to keep its size down. The resultant change to the group_home policy when device Mk1 is set to trusted is shown in Figure 8.11 (E).
Having defined each of the types of rules can be applied, the next section describes rule processing.

### 8.6 Access control decisions

This section discusses the operation of the Access Control Manager located on the device by reviewing various scenarios to illustrate access control decision-making. For each scenario, a similar process is followed.

The Policy Enforcement Point (PEP) receives a request from the SmartNet prototype. The PEP contacts the Policy Decision Point (PDP) to make a decision. The PDP retrieves all applicable rules that relate to the object group in question from the Policy Manager. The PDP processes all rules and returns a decision to the PEP. If the request is granted, the PEP passes a request to the Android Reference Monitor for the resources in question and the request is processed.

Next, four different scenarios are considered:

1. Local rule processing
2. Global rule processing using context
3. Global rule processing using trust
4. Conflict resolution in a global group

#### 8.6.1 Local rule processing between John's devices

John has two devices, `device_J1` to `device_J2` in his local group, `group_J`. He needs to share the `pictures` object group between them. Each device contains the same group policy as shown (A) and (B) in Figure 8.12. The Policy Manager on each device send the relevant policy to their PDP for a
decision. The PDP grants access to the object group on each device when the devices connect to each other.

![Diagram of policies on devices J1 and J2](image)

**Figure 8.12: Selection of applicable rules to send to the access control manager**

After the PDP grants access to the object group, and the Android Reference Monitor validates that all permissions are in place, the PEP of the ACM queues and sends all of files from one device to the other. The initial and final state of the pictures object group on device J1 is depicted in Figure 8.13 (A) and (B) respectively. The initial and final states on device J2 are depicted in (C) and (D) respectively. After processing, the object group of both devices contain the same set of files, shown by (B) and (D).

![Initial and resultant state of pictures object groups](image)

**Figure 8.13: Initial and resultant state of the pictures object groups on device J1 and device J2**

In the next section the sharing of files between members of a global group is discussed.
8.6.2 Global rule processing using context

John and Mary share the family_files object group between each other in group_home group during certain hours of the day. Both the local and global rules of each device are now reviewed by the Policy Manager. Figure 8.14 (A) shows the applicable rules in green, selected on device_J_1 and (B) device_M_1. The PDP and PEP now performs similarly to the previous scenario.

![Figure 8.14: Applicable rule selection on device_J_1 and device_M_1](image)

The next example discusses the use of trust in a global group.

8.6.3 Global rule processing using trust

Mark, who is not a member of any group of John and his family, comes over to visit. John trusts him and grants device_Mk_1 access to the dec_holidays object group without making his device a member of group_home. Whichever device device_Mk_1 connects to will not send or receive any files from group_home as it is not a group_home member. As Mark is Peter's friend, device_Mk_1 connects to device_P_1 and requests files from Peter through the family group. Figure 8.15 (A) shows the XML global rule and trust value for Mark's device selected on Peter's device_P_1 for processing. Note that
no local rules are selected on either device in the local policy sections of (A) or (B). The trust level which was manually assigned to Mark's device in Section 8.5.4, is processed with the rules to determine his trust score as required by the `dec_holidays` access control rule.

![Figure 8.15: Applicable rule selection on device_P1](image)

As `device_Mk1` is trusted sufficiently, it is allowed access to the `dec_holidays` object group and the files are sent. Figure 8.16 (A) shows the initial and resultant state of the `dec_holidays` object group on Peter's device. Peter's device receives nothing from Mark's device as he is not a member of `group_home`. Thus, the initial and resultant `dec_holidays` object group on `device_P1` remains the same. Figure 8.16 (B) and (C) show the initial and resultant state of `device_Mk1`'s `downloads folder`. Note that Mark does not have a `dec_holidays` object group on his device because he is not a member of `group_home` and has therefore not had it created. Thus, his device is merely sent the files from Peter's device to view which appear in the generic `downloads folder`.

![Figure 8.16: Initial and resultant state of object group and downloads folder](image)
The final example shows how conflicts in policy are dealt with.

### 8.6.4 Conflict resolution between access control rules of groups

The global group `group_work`, set up by John's manager Bill contains the `work_docs` object group. By error, John has shared this in his `group_home` group. This means that his family may have access to sensitive work documents that should only be visible to members of the global work group he belongs to. John's `device_{J1}` has two global policies, one for his `group_home` (A) and one for `group_work` (B) which each have an applicable rule to be sent for processing, shown in Figure 8.17.

When a request is received to access the `work_docs` object group, the PDP makes a decision according to the closed policy approach. As the rule from `group_work` states that only members of `group_work` can access `work_docs`, a decision of `deny` presides. Therefore, the result is a `deny` and no files are sent to any member of `group_home` regardless of John's erroneous rule in the policy.
Figure 8.17: Applicable rule selection on device J1

Now that all scenarios have been described, observations about the prototype are given.

8.7 Observations

The SmartNet prototype has a number of limitations that need to be addressed by future work. The limitations and areas of consideration are discussed in the following sections.

a) Usability

The setting of access control is a difficult task for any general user. It should be noted that the definition of a user-friendly interface was out of the scope of this research. As the SmartNet prototype has a number of pages that have to be navigated it may seem to be difficult to use. Even though usability is a concern that the researcher attempted to address in the design of the prototype, the user interface is not as easy to use as it could be. This can be daunting for the uninitiated, especially when functions such as the creation of object groups are done on one page and the assignment to groups on another page that is not a single button press away. Furthermore, policy management needs to be simplified to allow users to manage their own policies without a knowledgeable administrator. Future research should be done to ensure user-friendly access-control and policy management.
b) Use of trust and context
The use of trust in the access control policy of the SmartNet prototype was limited to the static and arbitrary assignment and use of basic trust values, which are more akin to how roles are used in trust-based access control models. This was done to limit the scope of this research. Future research should address dynamically changing trust values and more sophisticated trust computation algorithms.

In the examples given in this research, very simple context such as time of day was used. Future work should increase the number of context types in order to provide finer-grained control over access control.

c) Policy distribution
In the SmartNet application, in a peer-to-peer environments, no central location stores the policies of each group. This research made use of a group management device as a Master Policy Administration Point to ensure that no inconsistencies can occur between devices belonging to the same group. For future work, better, more distributed solutions to this problem should be researched.

d) Security
The researcher made the assumption that it is not plausible for every Android user to root their device to make use of the SmartNet prototype. Without rooting the device it is not possible to implement a kernel level policy enforcement point. In light of this decision, it was conceded that not every security aspect of the application could be completely achieved. The area of the prototype that suffered because of this was the protection of resources on the device from direct access.

8.8 Conclusion
In this chapter the SmartNet prototype was introduced to validate this research. The application was built to address the functional and access control requirements identified in Chapter 2 to address the questions posed in Chapter 1. The SmartNet prototype was presented along with a detailed overview of its initial setup and operation. The scenario of John and Mary was used throughout this chapter to highlight key LPSS functionality demonstrated by the SmartNet prototype during the various stages of its operation.

The most important aspect of this chapter was the demonstration of how the SmartNet prototype was used to share files between local and global group members with agreeing and conflicting rule sets. Each device was shown along with the rules that is uses to highlight how local and global rules can be combined. The SmartNet prototype provides a means for personal smart spaces to be built that can enforce both strict access control and personal preference, which is the contribution of this research.

The next chapter concludes this research.
Chapter 9
Conclusion

9.1 Introduction
This research aimed to create a Local Personal Smart Space (LPSS) framework capable of providing users with secure resource sharing capabilities while still factoring their personal preference into the decision-making process. As smart devices are becoming more and more pervasive in society and no real support exists to support personalisation when directly sharing resources, this topic presented itself as an appropriate topic to research.

This chapter serves as the culmination of research covered throughout the course of this document which set out to address the problem outlined in Chapter 1 by answering the research questions also posed in Chapter 1. Firstly, the SmartNet prototype which was developed to validate the LPSS framework and accompanying access control model is evaluated. Section 9.2 evaluates the access control model and Section 9.3 provides some critique. Following the evaluation, this chapter re-visits the research objectives and questions posed in Chapter 1 to determine if the research answered them successfully. Section 9.4 answers these questions and Section 9.5 reviews the contribution made by this research and the future work that can be done to improve it. Finally, the research draws to a close in the last section.

9.2 LPSS framework evaluation
The primary goal of this research is to investigate how to securely share information and resources in Local Personal Smart Spaces using the personal preferences of users, where smart devices are connected to each other with access-point less mediums such as WiFi-Direct. The culmination of a LPSS framework will answer the research questions posed in Section 1.5.

The objective specified in Chapter 1 was broken down in Chapter 2 where a number of functional and access control requirements were proposed to evaluate the LPSS framework and accompanying access control model. The purpose of this section is to evaluate the LPSS framework against those requirements to determine if they have been adequately met.

The next section discusses the functional requirements of the model and whether or not they were achieved. The second section addresses the access control requirements.
9.2.1 Functional requirements

The functional requirements of Chapter 2 are used to evaluate the functionality provided by the LPSS framework as verified in the SmartNet prototype. Each of the requirements are evaluated in the following sections.

a) A device-independent connection

Smart spaces, by their very nature, are geared towards interoperability. The LPSS framework was designed to connect devices of varying configurations. Due to the layered approach undertaken when designing the LPSS framework it is possible to support various connection types. Thus, connection semantics are abstracted from the layers above to ensure interoperability.

The SmartNet prototype was built on top of stock Android which is available to a large portion of the world's devices. Although the prototype only implemented Wi-Fi Direct, it is possible to allow a LPSS to be connected to via Bluetooth or any other peer-to-peer connection medium. Furthermore, any device that supports the network protocols used by the SmartNet prototype to connect and share information, should be able to connect regardless of whether they are running Android or another operating system. Thus, the LPSS framework can be implemented in other operating systems.

b) Group creation

The SmartNet prototype demonstrated the ability of the LPSS framework to create both personal and impersonal groups in the form of local and global groups. These groups could have object groups added to them and rules specified. Users could then invite other devices to their groups to share files.

c) Device and group management

The SmartNet prototype demonstrated the LPSS framework's ability to identify and connect to devices and form groups. The group management tools enabled a fair degree of control over what can be done in groups through the specification of access control rules.

d) Minimal user input

The LPSS framework was designed to be as user friendly as possible. However, smart devices have small screens and sub-optimal input options so compromises had to be made. In centralised environments there are dedicated policy stores operated by competent access control administrators. This is not the case in a decentralised environment where every device owner is responsible for their own access control rules. This raises two problems; first device inputs such as touch screens are not optimal for the input of complex access control policies. Secondly, not everyone is proficient in access control policy creation and management.

The policy manager used by the LPSS framework had to address both of these problems by compromising on the granularity of the policies that could be created. The policy manager allowed for
a simple default rule to be present whenever an object group was added to a group. This gives the user greater peace of mind if they are not familiar with access control policy specification. To address the sub-optimal inputs of smart devices, the LPSS framework opted for a less flexible policy approach that allowed users to select their desired policy options through a set of drop-down interface elements.

Minimal user input was not fully achieved as there are still a number of steps that have to be taken to set up the SmartNet prototype before it can be used securely. Although this is not ideal, the compromise between usability and the granularity of policy rules had to be made.

e) Easy to use
Ease of use is coupled with the requirement of minimal input. The Android environment used by the SmartNet prototype is not ideal for a complex smart space management suite. Thanks to the abundance of required functionality, the number of pages required was considerable. It was often not possible to group every form of related functionality in the same location as screen real-estate is a concern of mobile smart devices. Again, compromises had to be made and functionality was grouped as best as possible given the constraints.

9.2.2 Access control requirements
The access control requirements of Chapter 2 serve as the basis to evaluate the access control and policy aspects of the LPSS framework and accompanying access control model defined by this research. The next sections evaluate each of these requirements.

a) Strict access control enforcement
The SmartNet prototype demonstrated the ability of the LPSS framework to share resources in a managed way. Users could specify their access control rules or rely on the default rule. These rules ensured that files would only be sent to group members according to the set of access control rules.

The one failure in terms of access control was the inability to adequately secure the resources on the user's device. The research of Chapter 5 detailed the requirement of Android devices to be rooted to gain access to kernel functionality. As most users are either incapable of rooting their device or do not wish to undertake such actions, this research decided against the rooting of the device. The LPSS framework was built in the application layer to share resources between devices that have the SmartNet prototype installed. The access control manager of the framework is still reliant on the underlying Android reference monitor for low-level access control enforcement.

b) Simple access control policy management
Simple access control policy management was demonstrated in the SmartNet prototype where users were allowed to rely on a pre-defined access control rule should they not wish to provide a rule of
their own specification. Furthermore, the selection of a built-in selectable set of predefined policy options allows users on smart devices to more intuitively make changes to their policies.

This however comes at the expense of fine-grained control over what can be done with the rules. Fully-fledged access control policy tools could provide far more secure and fine-grained policies but smart devices are not geared to this level of involvement. Additionally, most users may not be capable of administrating such policies. Thus, this research compromised on the granularity of the policies in favour of ease of use and simplicity.

c) Policy combination
The SmartNet prototype demonstrated the LPSS framework's use of local and global policies to enable user preference. Both the local and global policies of various groups can be combined and processed when access control decisions are made. The approach chosen by this research to combine policy rules is simple and based on best practise. If there are two rules that are applicable to a resource, and one of the rules denies access, the final decision would be to deny the request. Thus, the model achieved a basic level of policy combination that supported user preference. Considering the dynamic and changing environment of the LPSS framework, better policy combination algorithms can be researched in future work.

d) Context
The LPSS framework illustrated via the SmartNet prototype that context can be successfully applied in access control decision-making. Although the LPSS framework model was designed to provide the users with access to the many forms of contextual information available on a smart device, the SmartNet prototype only made use of the time of day. This singular use of context was used to prove the concept of context in the access control model. The use of additional context would be beneficial especially in a real-world scenario.

e) Trust
The use of trust in the LPSS framework was arbitrary. Users could specify a static trust value for the device used by someone they know and then specify the required level of trust needed for a file to be sent to that device. The selection of this form of trust gives the user an additional degree of control over how they can specify their policy rules at the expense of some additional interaction being required with the application.

This approach is not ideal in large-scale smart spaces where there may be more devices that the user would want to specify trust values for. However, the LPSS framework caters toward smaller and more personal groupings of smart devices. Due to this reasoning, it was conceded that the implementation of a fully-fledged adaptive trust model could be moved to future work as it did not directly add to the contribution made by this research.
9.3 Critique

In order to test the viability of the LPSS framework and its accompanying access control model it was necessary to implement a prototype system. The SmartNet prototype allowed for the demonstration and evaluation of the LPSS framework and its access control model in a practical environment.

The SmartNet prototype was used to demonstrate the functional and access control requirements stated in Chapter 2 as a means to address the primary objective of this research. The SmartNet prototype provided a means to manage a LPSS environment on a mobile smart device and share files with group members. The prototype documentation in Chapter 8 highlighted the cornerstones of the LPSS framework in practical operation to validate the created model.

The prototype, however, was not infallible. There were a number of areas where the prototype fell short of its initial specification. Due to the restrictions of the Android operating system it was not possible to secure files and resources on the device to the degree that was initially intended. The prototype did however provide the degree of security required to validate the primary requirement of secure resource sharing.

Another area of the prototype that fell short was with regard to usability. Mobile devices have limited screen real-estate and the number of functions that the SmartNet prototype had to cater for, resulted in more pages that initially thought to be viable for a mobile application. Although some functionality could not be placed in such a way that setup and operation felt natural, the application still allowed the user to access all of the LPSS framework's functionality. The use of simple pre-configured user interface components for rule creation and alteration also aided in increasing the usability.

The prototype system, therefore helped in the evaluation of the suitability for the LPSS framework and its access control model to answer the questions that guided the course of this research. The following section re-visits the research objective and the underling research questions.

9.4 Research objectives and questions

In this section, the research objectives and questions are re-visited to gauge the outcome of this study. Chapter 1 defined the research objectives to address the problem domain of this research. Based on the research objective, a number of questions were extracted to further guide the course of this research.

Before answering the primary research question of this research, the three sub-questions of Section 1.5 are first discussed.
a) Question 1

*What are smart space environments and how can they be used to share information and resources between peer-to-peer smart devices?*

The literature review of Part 1 of this research investigated aspects relating to smart spaces and their ability to share information between peer-to-peer devices in a secure and managed way. Chapter 2 investigated the concept of a smart space and its natural evolution, the personal smart space, then went on to define the Local Personal Smart Space as a type of smart space to address the objectives of this research.

Chapter 4 considered aspects of how context and trust are used in smart spaces when the sharing of information and resources is to be done. Chapter 4 also served to highlight how other forms of smart space are structured and used to provide a vast array of services to the smart devices that connect to them.

Chapter 5 built directly on top of Chapter 4 by investigating the Personal Smart Space and detailed its structure. This chapter served to illustrate the core components of a smart space that would be required to build the model of this research. Chapter 5 also considered the Android operating system as a viable starting point for the creation of the prototype that directly addresses the objectives of this research.

To answer Question 1, the Local Personal Smart Space framework was developed to provide a means to share resources between peer-to-peer smart devices in an autonomous and managed way. The SmartNet prototype was built to verify the ability of the framework to provide smart space services in a usable and managed way.

The research carried out through this document leads to the answer that smart space environments are a viable and flexible means to facilitate the interconnection of smart devices and provide a common medium for managed information and resource sharing.

b) Question 2

*How can user information and resources securely be shared in Local Personal Smart Spaces so that they are not misused or compromised?*

As early as Chapter 3, this research investigated access control and why some of the most popular access control models are not suitable for use in LPSS environments. Chapter 4 followed on by investigating how both trust and context are used in modern smart spaces to provide flexible and robust access control.
Chapter 5 evaluated the chosen prototype implementation environment, Android, and its ability to implement the fundamental components of Personal Smart Spaces; the closest relative to the LPSS defined in this research.

To answer Question 2, this research developed the LPSS access control model for use in the LPSS framework. The access control model was built upon the findings of Chapters 3, 4 and 5 to provide a simple and flexible way for a smart device owner to protect their information and files. The model grouped files into object groups to simplify access control and used context and arbitrary trust to provide greater flexibility. The access control concepts, however, were all built upon a solid foundation provided by the access control research of this document.

c) Question 3

How can the personal preferences of users be considered when securely sharing Local Personal Smart Space resources, without compromising its security?

Question 3 was found to be closely related to Question 2 because users take into account their own personal preference when specifying which files or resources they want to share and with whom.

Chapter 4 reviewed a number of smart space environments and found that they had different approaches when considering the use of preference. Some of the models enforced strict access control but had no place for personal preference, while others considered personal preferences.

Chapter 5 built upon the notion of personal preference in access control by taking the notion of policy combination from Chapter 3 and evaluating how Personal Smart Spaces used personal preference to give users greater control over their environment.

To answer Question 3, this research found no specific way for user preference to be combined with access control policy to enable secure and personal smart spaces. In order to address this issue, this research defined the concept of local and global policy and policy combination as a part of the access control model defined in Chapter 7. As none of the explored literature was able to provide a basis for this form of combination, this research designed and built the local and global policy combination into the Local Personal Smart Space model to enable users to specify their personal preferences.

With the three sub-questions addressed, this research can move on to discuss the primary research question of this study and elaborate on how it was used to achieve the main objective of this study.

d) The primary research question

Can fine-grained access control be defined to support resource sharing between smart devices in a Local Personal Smart Space, by taking into account user preferences?
The main point to be discovered and confirmed through the content of this research is that the Local Personal Smart Space defined by this research can be used to securely share information and resources between smart devices in such a way that security is maintained even when personal preference is taken into consideration. The SmartNet prototype serves to confirm this point, however the implementation and model created to address this research question are not perfect and can be improved.

The next section addresses how the LPSS framework that was defined to answer the primary research question made a contribution and how it can be improved through future work.

### 9.5 Research contribution and future work

The research done in this document concluded that smart space technologies can be used to share information and resources in a managed and secure manner even when personal preference is considered. The majority of the components of the Local Personal Smart Space framework that were verified through the use of the SmartNet prototype were built using existing smart space paradigms that were already tried and tested.

Where the LPSS framework took a different approach was with regard to access control policy. The LPSS access control model required that both strict access control and personal preference be factored in to access control decisions. As no currently-defined method for doing this was found, this research defined the use of local and global policies to cater to personal preference and access control respectively. The access control model defined by this research was, therefore, built to accommodate the specification and combination of both of these types of policy to ensure secure and flexible access control.

As verified in the prototype, users of the SmartNet prototype were able to specify the access control rules for their groups and also provide a second set of rules for their personal groups which could be used to override the impersonal rules for their global groups. In doing so, users could effectively prevent the sharing of predefined files even if the group which they are connected to warranted the sharing of those files. Thus, a degree of personal preference was achieved and combined with access control enforcement to ensure both security and user preference.

As with all aspects of information security, there are compromises that had to be made and improvements that can be made in future work. The following are a list of considerations that can improve the LPSS framework and its access control model:

- The arbitrary trust mechanisms used in the model can be upgraded to a fully-fledged, adaptive trust based access control model that would require less input from the user.
• Contextual information can be expanded to perform contextual profiling for use in the trust-based access control model above.

• The group management device for policy management can be replaced by a fully decentralised policy management suite that allows policies to be merged regardless of which devices are present. This could enable a truly decentralised smart space management approach.

• User interaction and usability could be improved if personal preference could be set up on one device and be automatically transferred to another device when they log in.

The list of items above is not an exhaustive list of changes that could be made to the model. This list serves to illustrate some of the limitations of the model on a conceptual level that could be improved through future work as they were beyond the scope of the objective of this research. The following section concludes this research.

9.6 Conclusion

This chapter has provided a brief overview of the content of this dissertation. This research focused on enhancing the security and personal preference capabilities of smart space environments. In order to achieve this objective, various topics relating to the problem domain were explored over the duration of this document. The LPSS framework model, and the access control components that support it, were developed and practically tested to demonstrate a prototype for providing a means for users to connect their smart devices to share information and resources securely while having their personal preference taken into consideration before any access control decisions are made. To this end, this research concludes that LPSSs can be used to provide both resources sharing and personal preference without compromising too heavily on security.
Appendix A: Complete ACL Rules for Chapters 7 & 8

**Group J Rules**

\[ \text{group\_man}(device_{J1}, \text{group\_J}) \]
\[ \text{in}({\text{pic\_1}}, \text{pictures}) \]
\[ \text{in}({\text{pic\_2}}, \text{pictures}) \]
\[ \text{in}({\text{pic\_3}}, \text{pictures}) \]
\[ \text{in}({\text{pic\_tab\_1}}, \text{pictures}) \]
\[ \text{in}({\text{pic\_tab\_2}}, \text{pictures}) \]
\[ \text{in}({device\_J1}, \text{group\_J}) \]
\[ \text{in}({device\_J2}, \text{group\_J}) \]
\[ \text{in}({device\_J3}, \text{group\_J}) \]
\[ \text{cando}(s, \text{pictures}, +a) \]
\[ \text{dercando}(s, \text{og}, +a) \leftarrow \text{cando}(s, \text{pictures}, +a) \land \text{in}(s, \text{group\_J}) \]
\[ \text{do}(s, \text{og}, +a) \leftarrow \text{dercando}(s, \text{og}, +a) \]

**Group M Rules**

\[ \text{group\_man}(device_{M1}, \text{group\_M}) \]
\[ \text{in}({\text{pic\_mary\_1}}, \text{pictures}) \]
\[ \text{in}({\text{pic\_mary\_2}}, \text{pictures}) \]
\[ \text{in}({device\_M1}, \text{group\_M}) \]
\[ \text{in}({device\_M2}, \text{group\_M}) \]
\[ \text{cando}(s, \text{mary\_stuff}, +a) \]
\[ \text{dercando}(s, \text{og}, +a) \leftarrow \text{cando}(s, \text{mary\_stuff}, +a) \land \text{in}(s, \text{group\_M}) \]
\[ \text{do}(s, \text{og}, +a) \leftarrow \text{dercando}(s, \text{og}, +a) \]
**Group home Rules**

\[
\begin{align*}
group\_man & (device_{J_1}, \text{group}_\text{home}) \\
\text{in} & (\text{file}_a, \text{family}_\text{files}) \\
\text{in} & (\text{pic}_\text{peter}_1, \text{dec}_\text{holidays}) \\
\text{in} & (\text{pic}_\text{peter}_2, \text{dec}_\text{holidays}) \\
\text{in} & (device_{J_1}, \text{group}_\text{home}) \\
\text{in} & (device_{J_2}, \text{group}_\text{home}) \\
\text{in} & (device_{J_3}, \text{group}_\text{home}) \\
\text{in} & (device_{M_1}, \text{group}_\text{home}) \\
\text{in} & (device_{M_2}, \text{group}_\text{home}) \\
\text{in} & (device_P, \text{group}_\text{home}) \\
cando & (s, \text{family}_\text{files}, +a) \\
cando & (s, \text{dec}_\text{holidays}, +a) \\
cando & (s, \text{work}_\text{docs}, +a) \\
dercando & (s, \text{og}, +a) \\ & \Leftrightarrow \ cando (s, \text{family}_\text{files}, +a) \land \ \text{in} (s, \text{group}_\text{home}) \land (15:00 \leq \text{time\_of\_day} \leq 20:00) \\
dercando & (s, \text{og}, +a) \\ & \Leftrightarrow \ cando (s, \text{dec}_\text{holidays}, \pm a) \land (\text{trust\_subject}(s, ts), \text{trust\_object}(\text{dec}_\text{holidays}, to) \land ((ts > to) \lor (ts = to))) \\
dercando & (s, \text{og}, +a) \\ & \Leftrightarrow \ cando (s, \text{work}_\text{docs}, +a) \\
do & (s, \text{og}, +a) \Leftrightarrow \ dercando (s, \text{og}, +a)
\end{align*}
\]

**Group work Rules**

\[
\begin{align*}
group\_man & (device_{Bill}, \text{group}_\text{work}) \\
\text{in} & (\text{work}_\text{doc1}, \text{work}_\text{docs}) \\
\text{in} & (\text{work}_\text{doc2}, \text{work}_\text{docs}) \\
\text{in} & (device_{J_3}, \text{group}_\text{work}) \\
\ldots & \text{other group assignments} \ldots \\
cando & (s, \text{work}_\text{docs}, +a) \\
dercando & (s, \text{og}, +a) \\ & \Leftrightarrow \ cando (s, \text{work}_\text{docs}, +a) \land \ \text{in} (s, \text{group}_\text{work}) \\
do & (s, \text{og}, +a) \Leftrightarrow \ dercando (s, \text{og}, +a)
\end{align*}
\]
Appendix B: Resultant Research Output Abstract


Abstract:

As computer systems grow more compact, powerful and cheap to produce, they become more pervasive in society. Smart devices enable users to compute and share resources on the go. Services such as Wi-Fi Direct allow for the creation of device-to-device networks, of a peer-to-peer nature, deemed "smart spaces". Smart spaces are capable of providing an access-point-less means to share information and resources between their peers. Recent research points to the personalisation of smart spaces, making their management more challenging. Personalised smart spaces, advanced as they may be, introduce new security challenges such as secure resource sharing. This paper consequently evaluates a family-related scenario then a LPSS access control framework is proposed, with a focus on the specific nature of LPSS environments namely, local and global sets of rules defined in local and global policies. Finally, access control rules are presented, with respect to the motivating scenario, to illustrate the operation of access control enforcement using local and global policy rules.

Keywords: access control, context, smart space, trust, policy
References


