Contents lists available at ScienceDirect

Journal of Information Security and Applications

journal homepage: www.elsevier.com/locate/jisa

Access control for secure information sharing in smart content spaces

Brian Greaves, Marijke Coetzee[*]

Academy for Computer Science and Software Engineering, University of Johannesburg, Johannesburg, South Africa

A R T I C L E I N F O

Article history:
Available online 26 January 2017

Keywords:
Access control
Mobile content sharing
Personalization
Context
Smart content space
Local policy
Global policy

A B S T R A C T

Sophisticated mobile devices are becoming more compact, powerful and cheap to produce, leading to the implementation of smart applications that enable users to create and share large amounts of data on the go. Services such as Wi-Fi Direct support device-to-device communication, enabling peer-to-peer networks called smart spaces that support the sharing of information and resources between peers. In line with current research on personalization of the security of smart spaces, this paper introduces the concept of a proximity-based local personal smart space (LPSS) that presents new security challenges such as secure content sharing. An evaluation of current research on access control for smart spaces highlights that personalized context-based access control can provide better control over shared content. A local personal smart space access control framework is proposed focusing on the very nature of local personal smart space environments, namely, the enforcement of access control using personal preferences of users that are defined using policies. A prototype is presented that implements the access control model. Finally, the paper is concluded with some insight into future improvements.

1. Introduction

Today, the sophistication of smart devices makes it possible to share information directly between two devices, and create entirely access-point-less networks of devices (Adiba et al., 2004). Generally, as users possess more than one device, they need these devices to intelligently share content between themselves and the devices of friends and colleagues with minimal intervention (Gallacher et al., 2012). Currently, cloud-based applications such as DropBox, Box, and iCloud are commonly used for sharing content between devices. Even though these solutions are very popular, users have concerns regarding the security of their data in the cloud and the upload and download costs involved when the communication medium is not free.

To address the concerns introduced by cloud-based applications, peer-to-peer mobile storage and content sharing solutions are a current focus of research. Solutions such as Haggle (Nordström et al., 2014) and Mobistore (Fleming et al., 2014) demonstrate how content can be shared automatically between devices using local connections such as WiFi or Bluetooth. Without any doubt, these solutions can offer new benefits, but also introduce new threats for users making use of their services. Users may store a variety of personal content which they may want to share selectively with others who are in range. As it is not always possible to verify someone's identity visually due to the increasing strength of radio antennae, security is important to consider when sharing mobile content or resources with others (Manaf et al., 2009).

To date, not much research on access control for peer-to-peer mobile storage and content sharing solutions has been done. The distribution of information across devices makes it difficult to control access to resources using well-known access control models, such as discretionary access control (DAC), mandatory access control (MAC), and role-based access control (RBAC) as the nature the environment dictates that access control should be dynamic in nature (Kashevnik et al., 2013). The amount of sensory and other data available on smart devices can enable the measurement of the context of interactions (Adiba et al., 2004). Devices can respond to their operational environments and change the parameters of their operation based upon their context.

Research shows that smart spaces can make people's lives easier as they provide new types of applications and capabilities. This research extends previous research (Greaves and Coetzee, 2015) to describe local personal smart spaces and their access control requirements in more detail. An access control framework is presented that uniquely addresses personal and group preferences of users who are in possession of a number of mobile devices. The research makes a contribution by demonstrating how local and global group preferences are used together.

* Corresponding author. Academy for Computer Science and Software Engineering, University of Johannesburg, Johannesburg, South Africa. Fax: +27866499344.
E-mail address: marijkec@uj.ac.za (M. Coetzee).

http://dx.doi.org/10.1016/j.jisa.2016.12.002
2214-2126/© 2017 Elsevier Ltd. All rights reserved.

The paper is structured as follows: The concept of the local personal smart space is introduced, followed by a scenario. A set of access control and other requirements are identified. The topics of context and policy are examined to determine how they can be used to protect a local personal smart space by evaluating recent literature. Finally, the paper proposes a context-aware access control model which uses two dimensions of policy, namely local and global. Then access control enforcement is described using scenario-based examples used to highlight access control policy usage. Finally, a prototype implementation is described and the paper is concluded.

2. Local personal smart spaces

Pervasive computing (Satyanarayanan, 2001) is an important research focus that has attracted much interest due to the increasing number of devices that users are confronted with in their environment such as sensors, computers and smart phones. When pervasive computing is applied to a local domain it is referred to as a smart space. A smart space is a physical environment within a specific dimension containing adaptive devices that are automatically managed (Gallacher et al., 2012). Smart space research focuses on systems for fixed smart spaces, or systems supporting mobile users. Research on fixed smart spaces such as smart homes, smart buildings and smart cities has produced intelligent applications that dynamically manage infrastructure and sensors to suit the needs of users without application pre-configuration (Gallacher et al., 2012). In order to be able to provide the user with an intelligent environment where services and resources are managed on their behalf, the personalization of the environment is required. Personalization ensures that a system behaves differently when the user or the context changes (Gallacher et al., 2010). For example, if the location of a mobile user changes, a different set of services or resources may be made available. The system thus needs to track changes and adapt its behavior as specified by user preferences for different contexts.

In this regard, the PERSIST project (Cordis.europa.eu, 2008) aimed to provide a pervasive experience through an architecture based on the concept of a personal smart space. A personal smart space (PSS) is defined as a collection of devices that can be connected in a peer-to-peer manner to bridge the gap between mobile users and fixed smart spaces (Gallacher et al., 2012). In a PSS, devices and services are owned, controlled, or administered by a single user or organization. For example, QoSDream (Naguib et al., 2001) and Sentient Computing (Newman et al., 2001) are client–server, publish–subscribe PSS applications supported by a centralized approach, where clients subscribe to a location server that regularly polls their location to send them information about resources and other clients in their environment. Even though support for mobile users is provided, the system is dependent on centralized servers and requires mobile client devices to maintain a constant connection to the Internet. In more recent times, research in smart spaces have branched out from being solely dependent on fixed spaces to addressing applications such as tourist recommendations (Varfolomeyev et al., 2015) where software agents called knowledge processors run on devices to collaboratively collect and share information via semantic information brokers. The interaction between software agents leads to the construction of a service thereby decentralizing control within the PSS environment.

Moving further from fixed spaces, a mobile PSS is defined as a PSS that provides a mobile pervasive system around the user at all times (Gallacher et al., 2012). If the range of mobile device communication is limited by connections such as Wi-Fi Direct (Alliance, 2010) or Bluetooth (Haartsen, 2000), devices of a mobile PSS need to be in proximity to be able to interact. Such a proximity-based mobile PSS limits the physical dimension of the mobile PSS to a local scope with device-to-device communication (D2D). For this research, the architecture and operation of a proximity-based mobile PSS environment is a natural fit for the peer-to-peer content sharing solution required by this research.

Based on these constraints, this research now proposes the concept of a local personal smart space (LPSS) by extending the traditional concept of the mobile PSS. The foundation of the LPSS is a mobile PSS that is defined by a set of services, available within a dynamic space of connected mobile devices, owned, controlled and administered by a single user or organization, controlled by a set of personal preferences. This research adds a local dimension by requiring that mobile devices need to be in proximity of each other as they communicate in a peer-to-peer manner using technologies such as Wi-Fi Direct or Bluetooth. A LPSS is thus a proximity-based mobile PSS that can support services such as smart content spaces.

Important features of a local personal smart space are that it is owned by a specific user or organization and moves around with the user; their preferences are maintained by a set of rules; the physical boundary of the local personal smart space is determined by the proximity of devices from each other; and the local personal smart space must be able to identify and interact with other local personal smart spaces.

Unlike Personal Area Networks (Bourgeois et al., 2001), where mobile devices connect to each other in an ad-hoc manner when they are in close proximity, a local personal smart space is a smart space that enables the creation of groups of mobile devices that are governed by rules that have been defined by the owner of the group.

The relationships between users, mobile devices and content within a local personal smart space are now further investigated by means of a scenario.

3. Motivating scenario

In order to identify functional and access control requirements of local personal smart spaces, a scenario is presented next. The formation and use of both user and organizational local personal smart spaces is illustrated by considering three members of a family as individual owners of devices, and as a family group. A local personal smart space consists of a number of mobile devices, and is identified by a group name. As shown to the left of Fig. 1, John owns three mobile devices, namely a tablet, a smartphone for work, and a privately owned smartphone. His group is depicted as *group_J*. Mary has a smartphone and tablet (*group_M*) and Peter has a smartphone (*group_P*). At a global level, John, his wife Mary, and their son Peter are members of a family (*group_H*) that possess six personal mobile devices between themselves. Mark is a member of another local personal smart space group, *group_MK*.

Even though their mobile devices are not necessarily made by the same manufacturer, they would like to share content between the devices that they own, and also between all devices that are part of the family. As John has concerns about the security of cloud-based solutions and the associated costs incurred with uploading and downloading data, he wants content to be shared directly between devices when they are at home.

First, software is installed on each mobile device. The owner of a group of devices creates a group for those devices. John forms a group for this three personal devices, similarly Mary groups her two devices. As the designated owner of the family group, John creates the family group and invites all devices that should be part of the family group. Content can now be shared between devices in a personal group and between devices in the global family group. For example, John shares the pictures he takes with his phone with his tablet to have a backup of this content. Within the family group, John, Mary and Peter share selected family pictures between their devices so that they all have access to this content.

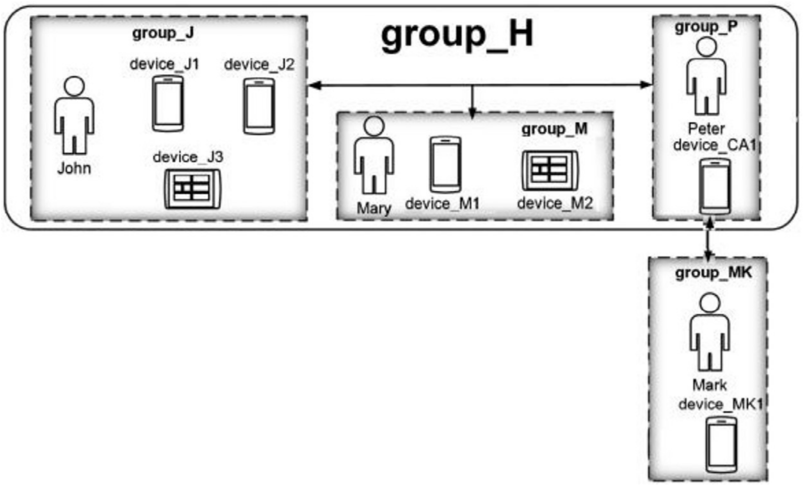


Fig. 1. Local personal smart space groups.

The devices belonging to a local personal smart space group connect to each other to share content when they are in proximity of each other. To ensure that this is done securely, a set of group rules is required to control content sharing for the family. For example, John would like to restrict his son Peter from sharing family content with friends who are not trusted. Should Mark, a trusted friend of Peter, want to view Peter's files, he can be granted access according to the family rules.

Finally, the software enabling connection, grouping, and rule definition should be easy and flexible to use, with minimal user input after the initial setup.

The scenario highlights a set of functional and access control requirements for a local personal smart space, listed next.

3.1. *Local personal smart space functional requirements*

- *A device-independent connection* — devices must be able to directly connect to each other regardless of the device manufacturer.
- *Local personal smart space group creation* — devices must be able to support the creation of groups, and group invitations.
- *Autonomous device and group management* — devices must be able to identify groups as they come into range and connect to known groups automatically.
- *Minimal user input* — after the initial setup little user input should be required.
- *Easy to use* — the system should not be confusing, nor difficult to use at any level.

3.2. *Local personal smart space access control requirements*

- *Access control enforcement* — access to resources must be enforced on each device. Policy must dictate what can and cannot be accessed.
- *Simple policy management* — access control policies should be straightforward for group and device owners to implement and maintain.
- *Policy combination* — personal and global policy rules need to be processed together, and if conflicts exist, they should be solved appropriately.
- *Context* — users, device interactions and the environment must be monitored in order to provide the basis for access control.

In order to gain a better understanding of the requirements for local personal smart space access control, a review of related research on access control is given.

4. Access control for local personal smart spaces

Access control is a security service responsible for limiting access to resources for legitimate users of a system (Sandhu and Samarati, 1994). The foundation of access control lies in the paradigm that subjects, entities capable of initiating action within the system, can perform actions on objects, representations of resources in the system (Quing-hai and Ying, 2011). The personalization of local personal smart space access control relies on the management of context so that resources and services can be accessed by a user at applicable times and manner, personalized according to the user's requirements. As each device in John's group needs to make access control decisions about the protection of resources on the device, the device is responsible for its own access control decisions. In order to be able to make valid decisions, a device needs to manage its own access control policy rules addressing user, device and environmental context.

As context is central to access control personalization, it is defined next. Thereafter a review of related work on access control and the propagation of policies between devices is considered.

4.1. *Context*

Context is anything that is used to characterize the situation of some form of entity (Abowd et al., 1999). 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, therefore, is characteristic of what the entity is, is doing, or is affected by during its operation (Feng et al., 2008). Changes in context may lead to breaches in security that need to be addressed by revising the security policy. Thus, a cyclic relationship is formed between security context, security policy, the operational environment and changes in context (Brezillion and Mostefaoui, 2004).

4.2. *Related work on access control*

Previous research and commercial solutions have addressed access control for content sharing between nodes in distributed and cloud-based systems in a very limited manner. The implementation of basic mechanisms by cloud-based content sharing solutions such as DropBox illustrates the need to have simple and intuitive interfaces. A review of literature on distributed computing environments such as mobile ad-hoc networks (MANETs) and smart spaces can provide a foundation for this research. In addition, research on access control policy management in distributed environments also needs to be explored. Next, each of these approaches is investigated in more detail.

4.2.1. MANETs

In a Mobile Ad hoc NETwork (MANET), wireless mobile nodes self-organize in random and temporary network topologies. Nodes directly communicate with other nodes within their radio ranges and via intermediate node(s) not in their range. Literature shows that trust and risk, often combined with roles suit MANET access control decision-making in emergency environments (Ayari et al., 2008; Cheng et al., 2010). Bakar and Talib (2015) show how roles are used to support access to sensitive information in disaster areas.

Due to the ever-changing nature of MANETs, they do not directly suit the group approach needed by this research. The distribution of policy in MANETs either follows a central policy server approach that distributes policies to nodes, or a distributed, hierarchical model is used (Ayari et al., 2008; Cheng et al., 2010; Alicherry et al., 2009). Because of the frequently changing network structure and its associated policy distribution mechanisms, MANETs are not ideally suited as a platform for this research.

Next, access control in smart spaces is investigated.

4.2.2. Smart spaces

Existing access control schemes that have been proposed for smart spaces are now described with the aim of highlighting features of the proposed access control solution. The development of access control models for smart spaces has been supported over the years, initially with projects such as PERSIST (Cordis.europa.eu, 2008) and its successor SOCIETIES (Ict-societies.eu, 2016), and more recently with the Smart-M3 (Honkola et al., 2010) project. Requirements identified for the Cerberus (Al-Muhtadi et al., 2003) solution specify that access control must address multilevel security based on policy, the current environment and available resources. A descriptive, well-defined, and flexible security policy language should be able to incorporate rich context information and physical security awareness. Authentication should be extended to include human users, mobile devices and applications.

Access control in smart space environments is geared to the use of a central controller to govern distributed devices. Generally, contextual information may at times be used todetermine a level of trust to determine access control decisions. White et al. (2005) define a user-centric, attribute based access control model that uses context information to define the physical space of the user as well as the context in which resources are accessed. User preferences are used to support automatic configuration of resources as long as system-wide policies are not affected. Kashevnik et al. (2013) define a decentralized smart space environment based on the open source Smart-M3 (Multidevice, Multidomain, and Multi-Vendor) platform, originally developed at Nokia Research Center in 2009. Entities are geographically dispersed and the controlling entity is fixed. The controlling entity makes access decisions via an access control service that holds all access control rules. The access control model is role and attribute-based. Roles are assigned dynamically based on the smart space participant's trust level that is calculated using the participant's context such as location, current date and device type. The personal preference of individual participants is not considered. Consec (Al-Rabiaah and Al-Muhtadi, 2012) is a context-aware access control solution for smart spaces that identifies that context information of users is sensitive, requiring that the legitimacy of sensors and receiving applications be determined. Mutual authentication with the Kerberos Protocol is supported to achieve this goal. Recently, Hosseinzadeh et al. (2016) proposed a context aware, role-based access control model for smart spaces in the health care domain that preserves the privacy of the users such as patients through their preferences. The access control scheme is supported with ontological modeling to enable representation of knowledge and support for automatic data reasoning and inferencing. A central access control service makes all decisions. There are two sets of rules, namely rules designed by the administrator, and the rules defined by the user to protect their personal data. Access control decisions take into account the denial takes precedence strategy when there is a conflict between permissions.

Smart spaces are local and dynamic in nature, making them well suited to the Internet of Things (IoT) that is characterized by ubiquitous interconnections of highly heterogeneous networked entities and networks (Korzun et al., 2013). Possible solutions to IoT smart space access control identify that access is session-based by way of join/leave operations. If two agents exist in a smart space, they each decide which information to share with the others, and which private information to keep locally accessible. An agent can combine private and global information in local reasoning. Access granted to smart space information that is contributed by a range of agents is thus controlled. Access control is context-dependent and fine-grained. The security level of a device allows it to access knowledge that has the same security level.

From this discussion it becomes clear that local personal smart space features such as the distribution of content across user devices, the ownership of content and the ability of the user to control the sharing of his content can be solved by incorporating context-awareness and user preferences in the access control model.

A relevant aspect to take note of is that user preferences are centrally stored. When user preferences need to be used in local smart space access control reasoning, decentralized policy management and policy propagation becomes important to consider, as discussed next.

4.3. Policy management and propagation

The local personal smart space that this research proposes consists of a number of mobile devices belonging to a personal group and/or a global group such as a family. Devices communicate in a peer-to-peer manner, with a designated peer acting as group leader. As each device makes its own access control decisions, there is no central control. Each peer needs to consider the access control policy of another relating to the same resource, leading to two primary concerns. The first is how a group policy should be propagated to all devices in the group, and the second is how policy changes should be managed so that consistent policies are propagated to group devices. If changes are made to a group policy from different devices, the result could be a management nightmare.

Recently, Suomalainen et al. (2010) addressed the propagation of smart space information using a publish and subscribe architecture. The solution is based on the Smart-M3 project, and defines the RIBS (RDF Information Base Solution) server, a centralized solution, that allows agents to store, retrieve, manipulate and subscribe to information such as policies. Communication between RIBS and agents is mutually authenticated and encrypted. When an agent is first introduced to smart space, it is given smart space credentials enabling it to authenticate to RIBS and other agents later on. When an agent publishes new or updated information, a broker informs consuming agents of this change in order to propagate information to all participants. When a new agent joins the smart space, it configures its security policies and sends them to the RIBS server that enforces those rules. Each agent thus carries its security policies and the RIBS does not need to know the preferences of each agent. In this environment, access control is thus based on the brokering RIBS server, which must be trustworthy. The publish/subscribe

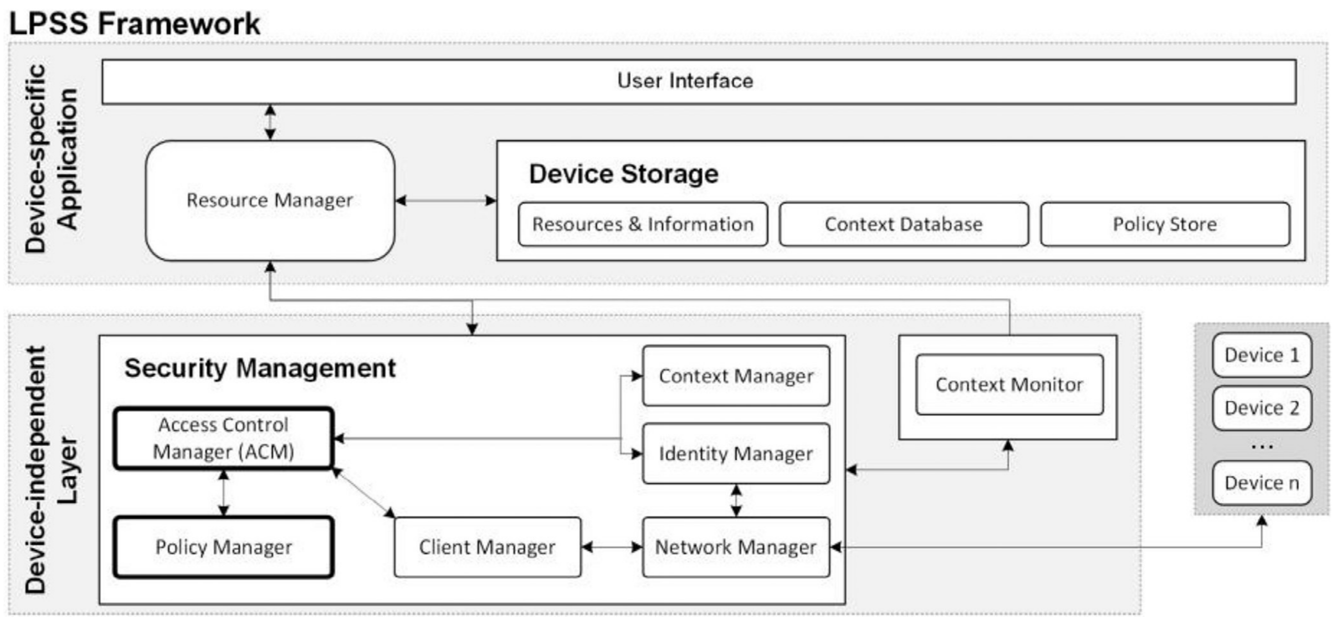


Fig. 2. Local personal smart space framework.

communication protocol is more suited to large-scale distributed applications that run across multiple administrative domains.

To maintain consistency in dynamically changing environments Díaz-López et al. (2015) propose a means for managing distributed XACML (Moses et al., 2005) policies. They identify that a security domain can be given a level of autonomy to manage their resources locally. Policies from another domain that are focused on local resources should be restricted in some way. They propose 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. For example, for a group of devices, a policy is only modified on the group management device (MPAP), but propagated to and applied by all other devices.

Next, the local personal smart space framework is proposed in line with the scenario and the findings of the state-of-the-art research.

5. Local personal smart space framework

The local personal smart space framework is installed on each device that participates in a local personal smart space group such as given in Fig. 1. The design of the framework is inspired by the PERSIST smart space framework (Gallacher et al., 2012) and enables devices to connect to each other and to share information and services within a group and between different local personal smart space groups. Fig. 2 gives a high-level overview of the layers and components of the framework which interact in order to provide necessary functionality to the local personal smart space environment. The framework is divided into two primary segments, namely the Device-independent Layer and Device-specific Application, rolled into a single application for installation.

The device-specific application, shown at the top of Fig. 2, accommodates the different types of mobile operating system platforms that are available. A user interface enables interaction with the user of the device. The Resource Manager is a middle-man in the framework that is responsible for interfacing with the device's storage for storing and retrieving various types of information.

At the next layer, the device-independent layer houses most of the functionality of the framework. The Security Manager contains all of the management components required by the framework. The Access Control Manager is responsible for making access control decisions. It primarily interacts with the other components of the Security Management layer to make decisions using identities and context as provided by the Context and Identity Managers. The Access Control Manager refers to the Policy Manager to look up access control and policy rules. The Policy Manager supports the creation and alteration of policies affecting the system. Policies can be dynamically adapted by changes in state as detected by the Context Manager. It also enables users to alter their local policy.

The Network Manager is responsible for abstracting the semantics of interfacing with lower layers such as the device's protocols and radio to pass requests and information up and down between devices. The Client Manager deals with connections, groups and passes requests to other components to be processed. The Context Monitor is an ever-present watcher that observes the context of interactions and stores them in the Context Database, via the Resource Manager.

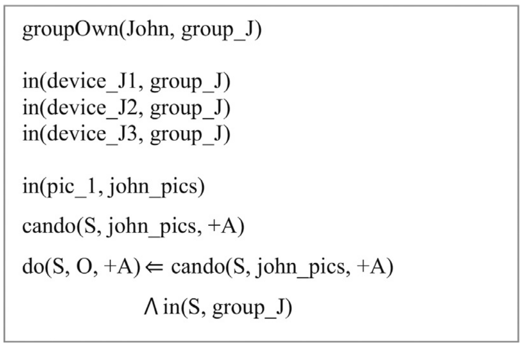
The operation of the device-independent layer is abstracted away from users, as they manage all interactions through the easy-to-use user interface. Next, the access control model supported by the Access Control Manager is defined.

6. Access control model

This research describes an access control model for local personal smart spaces that supports local and global policy rules and context constraints. In a local personal smart space, user and device identification is available. All access permissions and rules for both the personal and family groups are stored in the policy store. Access control policies of the local personal smart space are discussed next.

6.1. Access control policies

Access control policy rules are set by group owners that create groups and add resources to them. The Policy Manager supports a flexible, descriptive and well-defined policy language that can take into consideration context information. The Policy Manager makes decisions following the closed policy (Sandhu and Samarati, 1994) approach that allows access if there exists a positive authorization



groupOwn(John, group_J)

in(device_J1, group_J)
in(device_J2, group_J)
in(device_J3, group_J)

in(pic_1, john_pics)

cando(S, john_pics, +A)

do(S, O, +A) ⇐ cando(S, john_pics, +A) ∧ in(S, group_J)

Fig. 3. Local group policy for *group_J*.

for it, and denies it otherwise. Hence, actions can only be performed if there is a rule stating so. Furthermore, the Policy Manager needs to distinguish between two types of access control policies, namely:

- *Local policies* — access control rules that govern personal preferences relating to local resources of devices in a personal group of a single owner. Each device in the group has a copy of the local group policy and any changes must be propagated to each device in the group.
- *Global policies* — access control rules that are applied to all group members in a global group where devices belong to different owners. Each device stores a copy of the global group policy, and all changes made to the policy must be propagated to each device in the group.

6.2. Model definition

Access control rules define the action that can be performed by a subject on an object. If $s \in S$, $o \in O$ and $\{+a, -a | a \in A\}$, a permission can be defined as the tuple $\{s, o, \pm a\}$.

The *set of subjects S* is defined as the agents that act on behalf of the owners of devices. Each agent is uniquely identifiable.

The *set of objects O* is the *shared folders* that are used to store the content of either local or global groups. As it is very complex to share many individual files between devices, shared folders simplify the management of objects such as files. The shared folder of a local personal smart space group is a *smart content space*.

The *set of actions A* is the set of permissible actions such as read and read/write.

GRP is the set of devices that are grouped together, where a device may belong to more than one group. Each group is created independently, thus no group hierarchies exist. Each group is managed by a group management device, where access control rules are defined.

In the next section, access control rules are presented to illustrate the model of the local personal smart space access control framework.

6.3. Access control rules

This section uses example of scenarios relating to John and his family to illustrate the operation of the rules. Access control rules are defined in a logical manner, but are implemented in the framework in XML. First, the use of local policies within a local group is illustrated.

6.3.1. Local group with local policy

The first scenario deals with the creation of a *local* group, with local rules that are used to determine access, as shown in Fig. 3.

John, as shown in Fig. 1, creates a group *group_J* for his devices, *device_J1*, *device_J2* and *device_J3* before he can start sharing files and resources. He assigns himself as the group owner through the *device-specific application*. In the background, the rule to accomplish this for an owner, *Own*, and a group, *G*, is defined by:

$$\text{groupOwn}(\text{Own},\ G) \tag{1}$$

Once John has created the group, he invites his other devices to the group. Using group credentials, they are added to the group by the rule for devices, *S*, and group, *G*, as follows:

$$\text{in}(S,\ G) \tag{2}$$

With all of the group members connected, he can group his files into the group's shared folder O, or create a shared folder of his own to share. A file, F, is assigned to a shared folder O as follows:

$$\text{in}(F,\ O) \tag{3}$$

John's *device_J1*, *device_J2* and *device_J3* are the subjects of the rule and the group is his personal group, *group_J*. Once the devices are all assigned to the group John can create access rules for subjects, *S*, shared folders *O*, and actions, *A* where "−" indicates deny and "+" indicates permit as follows:

$$\text{cando}(S, O, \pm A) \tag{4}$$

This rule grants access to subjects to perform actions on object which are shared folders. As the framework uses positive authorizations for access to be granted to an object the following must evaluate to "true" — a decision is represented by "do" and an access control rule is defined by "cando".

$$\text{do}(S, O, +\ A) \Leftarrow \text{cando}(S, O, +A) \tag{5}$$

The rule needs to further limit control to member devices in the local group, *group_J*. Thus, the access control rule is augmented with a group check as follows:

$$\text{do}(S,\ O, +\ A) \Leftarrow \text{cando}(S,\ O, +A) \wedge \text{in}(S,\ \text{group\_J}) \tag{6}$$

For example, if John wants to share an object *pic_1* between his devices, he adds it to his shared folder, *john_pics* as shown in Fig. 3.

Fig. 3 lists the local access control policy of *group_J*. The devices that are part of the group are defined. A single access control rule grants access to subjects to access to John's shared folder *john_pics*, and a decision rule states that access can be granted to *john_pics* if the subject is in his group, *group_J*.

6.3.2. Local and global policies

This next scenario illustrates the use of both *local* and *global* policies on devices.

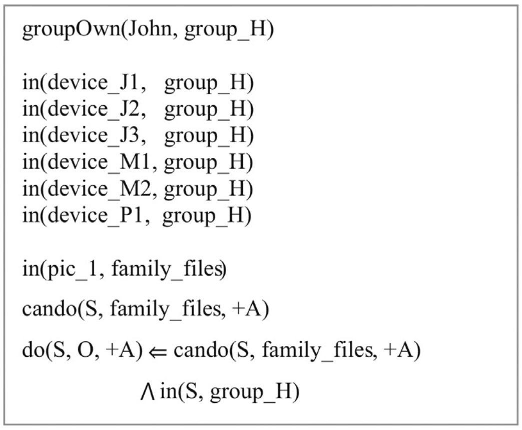
After the family members have set up their personal groups, John creates the *global* group, *group_H*, for his family shown in Fig. 4.

He sets the group owner, and invites all the family devices to the group. A shared folder *family_files* is defined. Similar to the local group, he specifies that *family_files* can only be accessed by group members.

It should be noted that a device such as *device_J1* is governed by the policies of both the local and global groups it belongs to. *device_J1* would be able to access the content of both *john_pics* and *family_files* stored in its local personal smart space application.

6.3.3. Policy using contextual information

This scenario illustrates how a policy limits access if context restrictions are not met. The access control rule is extended with user defined predicates given as literals $L_{1\dots}\ L_n$. The most common



groupOwn(John, group_H)

in(device_J1, group_H)
in(device_J2, group_H)
in(device_J3, group_H)
in(device_M1, group_H)
in(device_M2, group_H)
in(device_P1, group_H)

in(pic_1, family_files)

cando(S, family_files, +A)

do(S, O, +A) ⇐ cando(S, family_files, +A)
∧ in(S, group_H)

Fig. 4. Global group policy for *group_H*.

context dimensions include physical location, time, mobile device capabilities, and network.

$$\mathrm{dercando}(s,\ o, +a) \Leftarrow \mathrm{cando}(s,\ o, +a) \wedge \mathrm{in}(s,\ grp) \wedge L_1 \& \ldots L_n \tag{7}$$

One of John's smartphones, *device_J2*, is used for work purposes. This device is part of a global group *group_W*, set up by his manager at work to share documents. He stores documents *work_doc1* and *work_doc2 in a* shared folder *work_docs*. All devices belonging to this group are restricted to only grant access to devices of colleagues belonging to the global group *group_W*, during working hours as follows:

$$\mathrm{do}(S, O, +A) \Leftarrow \mathrm{cando}(S, \mathrm{work}_{\mathrm{docs}}, +A) \wedge \mathrm{in}(S,\ \mathrm{group}_W) \\ \wedge\ (09:00 \leq \mathrm{time\_of\_day}() \leq 17:00)$$

The *global* policy rule denies access to any file in *work_docs* if the subject is not in John's group of colleagues, *group_W*, 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.

6.3.4. Policy using trust

In this final scenario, a policy grants access using trust, irrespective of the group one belongs to. This feature is similar to trusting a device that one connects to via e.g. Bluetooth. Trust is not computationally calculated, but is defined as a binary value by the user.

Mark, shown in Fig. 1, is a trusted family friend. As he is not part of the family, or Peter's group, he cannot view their family or other pictures. To support social collaboration, Mark can be granted ad hoc access for a brief time, depending on the restrictions that are set. Access is granted for a limited time where $L_1 \ldots L_n$ are date and time literals as follows:

$$\mathrm{dercando}(s,\ o, +r) \Leftarrow \mathrm{cando}(s,\ o, +r) \wedge \mathrm{trust}(s,\ \mathrm{trusted}) \\ \wedge\ L_1\ \&\ \ldots L_n \tag{8}$$

Mark is given ad-hoc read access to the *family_files* object based on Peter's trust in him for a day. Here, a user is given the option to override the global group policy using his preferences on a specific device. An access control rule is defined to grant trusted subjects access to read the content of the object group as follows:

$$\mathrm{dercando}(s,\ o, +\ r) \Leftarrow \mathrm{cando}(s,\ o, +r) \wedge \mathrm{trust}(s,\ \mathrm{trusted}) \\ \wedge\ \mathrm{date}(12/10/2016)$$

Although Peter grants Mark ad-hoc access to the *family_files* shared folder, John as owner may not want to allow access to non-group members. The next rule denies access to any subject who is not a member of the family group, even if they were granted access by another decision. Here, the permission that Peter grants to Mark is overridden by the family group access control policy that explicitly denies access if the subject is not part of the smart group space. To ensure that only group members can access a shared folder, the following rule is set:

$$\mathrm{dercando}(s, o, -a) \Leftarrow \mathrm{cando}(s, o, +a) \wedge \ulcorner \mathrm{in}(s, grp) \tag{9}$$

Therefore any party who is not a group member is denied access to the shared folder. As both positive and negative authorizations are defined, a conflict resolution policy of denials takes precedence is applied using the closed policy as a decision policy.

$$\mathrm{do}(s,\ o, +a) \Leftarrow \mathrm{dercando}(s,\ o, +a)\ \wedge\ \ulcorner \mathrm{dercando}(s,\ o, -a) \tag{10}$$

Therefore, access is only granted if there does not exist an access control rule that denies access to the shared folder.

Next, the implementation of the local personal smart space framework is described.

7. Implementation

The local personal smart space framework was implemented as a prototype to demonstrate the operation of the Policy Manager and Access Control Manager in an Android environment using Wi-Fi Direct. The Android Studio integrated development environment using XML (Unknown, 2016d) and Java (Unknown, 2016b) was used for implementation. Database functionality was supported with SQLite (Unknown, 2016c). A Samsung smart phone and a tablet both running Android Lollipop 5.0 (Unknown, 2016a) were used as devices.

Android is structured into three layers, namely the application, middleware and kernel layers. Android applications exist in the application layer and are written in Java. Within the Android application layer are found the device-independent layer and device-specific application layer of the prototype, as was shown in Fig. 2. Within the device-independent layer lies the Access Control Manager, described next.

7.1. Access control manager in Android

The Access Control Manager in the Android application layer complements the decisions made by the stock Android reference monitor found in the Android middleware layer as shown in Fig. 5. The Access Control Manager consists of custom built policy enforcement and policy decision points. First, the policy enforcement point (PEP) intercepts application requests shown by (1). Secondly, the PEP requests an access control decision from the policy decision point (PDP) shown by (2). The PDP interrogates policy and context managers to access applicable rules related to the resource in question. The PDP uses 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 available, the PDP renders a verdict. If the PDP denies the request, the application is notified by the PEP and the operating system is not made aware of the request. If the PDP permits the request, the PEP passes the request on to the underlying operating system reference monitor shown by (3).

The reference monitor performs additional permission checks to grant or deny access to the resource. The Access Control Manager does not replace Android's stock reference monitor but rather performs an additional layer of fine-grained access control.

Android is a privilege-separated operating system where apps are isolated from each other to ensure better security. SELinux uses mandatory access control to strictly enforce a central policy that

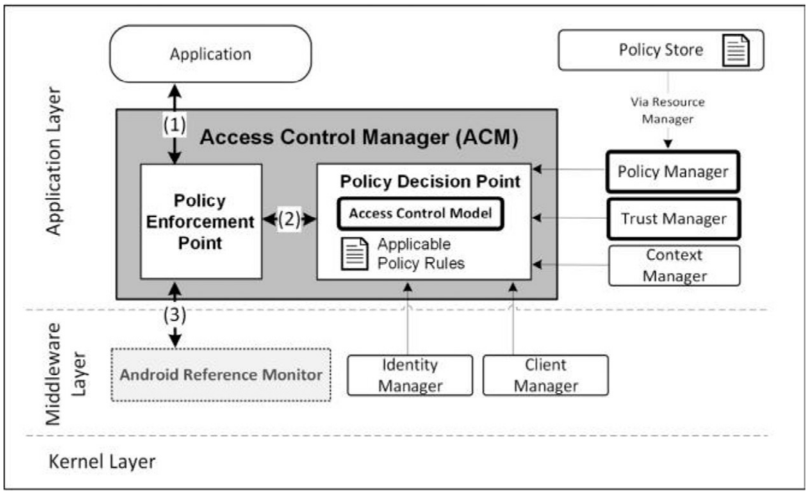


Fig. 5. Access control manager.

supports better isolation by containing installed apps within a controlled space and limiting their ability to compromise system resources (Bugiel et al., 2013).

When implementing access control in Android apps, there are two approaches that can be followed (Bugiel et al., 2013). First, middleware and/or kernel layers can be extended requiring the rooting of a device by modifying system partitions. A consequence is that unsigned kernels can contain malware that can execute using all permissions. Secondly, inline reference monitors (Jeon et al., 2012), which is the approach chosen by this research, can be used. The Access Control Manager is placed at the Android application layer to have full access to the internal state of the prototype giving access to contextual information about the requester. Unfortunately this makes the reference monitor prone to compromise as it may share the sandbox with any malicious code. The prototype is better protected by storing sensitive data in the app data folder that is only accessible to the prototype (Backes et al., 2013), and by controlling communication between apps by using e.g. the SEIntentFirewall (Mutti et al., 2015) that provides fine-grained mandatory access control for Intent objects.

7.2. Policy manager

Each device is responsible for making its own access control decisions, and needs a copy of all local and global group access control policies relating to the shared folders stored on the device. Any change made by the group management device to a group access control policy must result in the update of the policy to all members. To achieve this, the group management device acts as a MPAP (Master Policy Administration Point), requiring that policy changes are only made on this device. The remaining devices in the group serve as PAPs (Policy Administration Points) that need to have up to date policies. The problem of having different versions of an applicable policy is hereby eliminated. After authentication, devices verify each other's policy versions and if they do not match, the connection is terminated. If one of the devices is the group management device acting as MPAP, it can update the policy of the other device to the latest version.

However, the group management device is not guaranteed to always be present when policy changes occur. To prevent having outdated policies on local personal smart space devices, two configuration options are supported:

- Group policy updates may only be made on the group management device if all devices are connected to each other at the same time, to immediately propagate changes to all devices. This is suited to a small group of personal devices.
- As it would be highly unlikely that all devices in a larger group are present at the same time, changes are uploaded to the cloud, from where they are pushed to all devices. An updated group policy becomes active when all devices are successfully updated. If some devices are not updated as they are unreachable after a specific time, the rest of the group members can continue to interact, but will not be able to communicate with devices with outdated policies.

Policies maintained by the policy manager of each device are securely stored on a device, and securely transmitted between devices. The operation of the prototype is described next.

7.3. Prototype operation and user interfaces

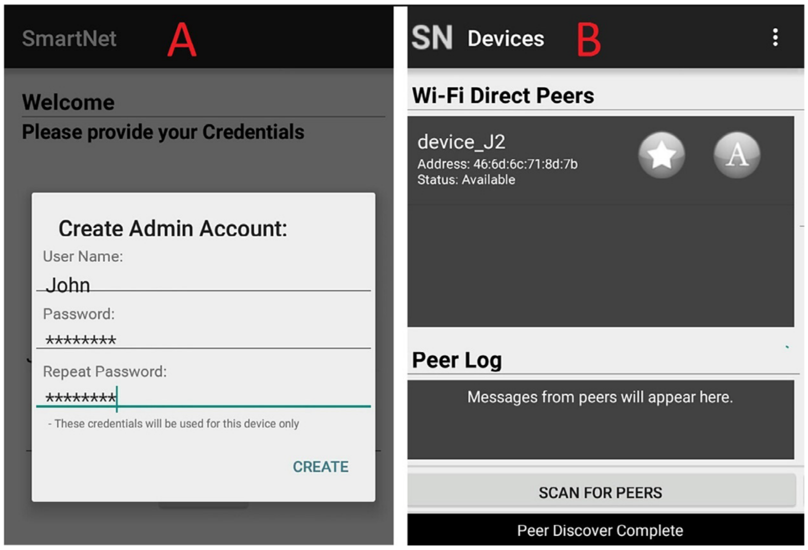
To enable ease of use when creating the prototype, a balance between complexity and control should be found by understanding user's security needs, and what they are willing to do to achieve them. The prototype should support simplicity in design when content is shared. In order to achieve this, read-only access to shared folders is provided. Generally, users understand how cloud-based content solutions work and the features that are provided. In this regard, shared folders are seen by users as separate storage spaces over which they do not have full control.

The operation of the prototype starts when two devices come in proximity of each other to initiate a local personal smart space handshake. During the handshake each device exchanges local personal smart space group and device information as well as their policy versions and group credentials. If all information is valid, devices can exchange lists of files to be added to the local personal smart space smart content space (shared folder) of the group. Missing or out of date files are identified and added to a request list that is exchanged. The Access Control Manager determines if requested files can be sent to the other device or not. Finally, permitted files are sent between devices to ensure that the local personal smart space group content is up to date on each device.

The operation of the prototype is now reviewed by describing the account creation and login procedure, group creation, shared folder creation and the setting of rules over shared folders.

7.3.1. Account and login

Upon first startup, the user is prompted to create a set of administrative credentials to control access to the application and



SmartNet

Welcome
Please provide your Credentials

Create Admin Account:
User Name:
John
Password:
****
Repeat Password:
****
- These credentials will be used for this device only
CREATE

SN Devices

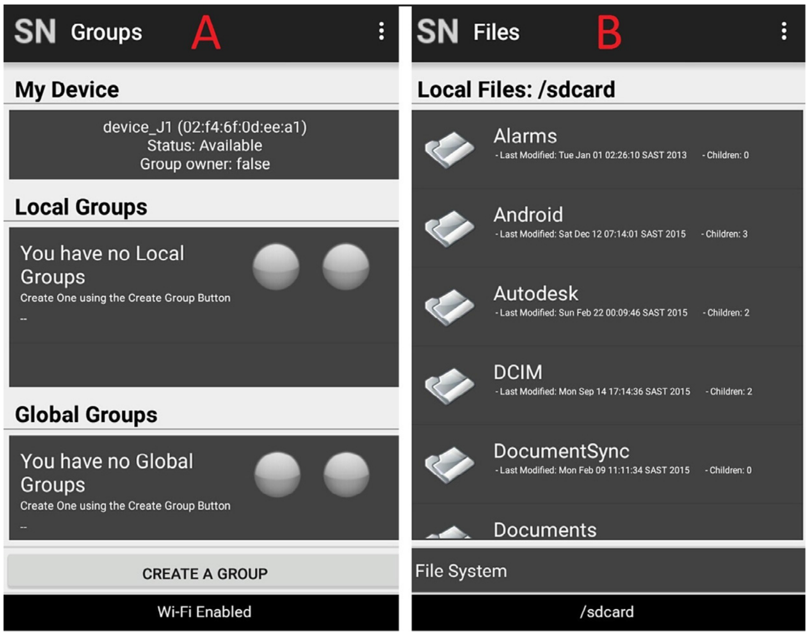
Wi-Fi Direct Peers

device_J2
Address: 46:6d:6c:71:8d:7b
Status: Available

Peer Log
Messages from peers will appear here.

SCAN FOR PEERS
Peer Discover Complete

Fig. 6. Admin creation and the devices page.



SN Groups

My Device
device_J1 (02:f4:6f:0d:ee:a1)
Status: Available
Group owner: false

Local Groups
You have no Local Groups
Create One using the Create Group Button
--

Global Groups
You have no Global Groups
Create One using the Create Group Button
--

CREATE A GROUP
Wi-Fi Enabled

SN Files

Local Files: /sdcard
Alarms
- Last Modified: Tue Jan 01 02:26:10 SAST 2013 - Children: 0
Android
- Last Modified: Sat Dec 12 07:14:01 SAST 2015 - Children: 3
Autodesk
- Last Modified: Sun Feb 22 00:09:46 SAST 2015 - Children: 2
DCIM
- Last Modified: Mon Sep 14 17:14:36 SAST 2015 - Children: 2
DocumentSync
- Last Modified: Mon Feb 09 11:11:34 SAST 2015 - Children: 0
Documents

File System
/sdcard

Fig. 7. The groups and files pages.

prevent unauthorized alterations to groups or rules. This is shown in Fig. 6 (A). Upon successful login, the user lands on the Devices page shown in Fig. 6 (B), where they can view other in-range Wi-Fi Direct devices.

7.3.2. *Groups*

From the Devices page, the user can navigate to the Groups page shown in Fig. 7 (A) to manage either Local or Global Groups. The user can create a group of their choosing as depicted in Fig. 8 (A) where a personal group is created and results in a change to the Groups page as shown in Fig. 8 (B). The user is prompted with information in order to understand whether he is creating a local or global group.

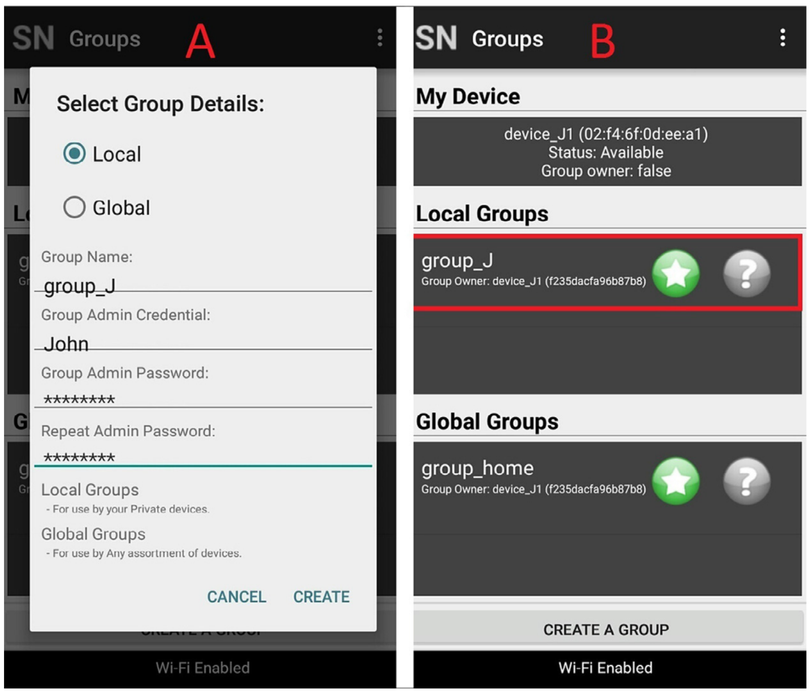
From the Devices page the user has the option to swipe to the right to access the Files page shown in Fig. 7 (B). From the files page, a user can create a shared folder (object group) to house all the files which they wish to share by selecting the option from the drop-down at the bottom of the page. The user is then presented with the dialog of Fig. 9 (A) where they may enter a name of their choosing. The user selects files to be included in the shared folder and the result is shown in Fig. 9 (B).

Likewise, a user can create groups not just for personal use, by selecting the Global option during group creation. The result of creating a group called *group_home* is shown at the bottom of Fig. 8 (B). Once a group is created, devices can be invited to the group from the Groups page.

Selecting a group from the Groups page takes the user to the Group Management page shown in Fig. 10 (A). The section shows group information and Associated Devices that are current members of the group.

7.3.3. *Access control rule creation*

Previous research points to the fact that users create permissions for a fraction of the content they share, and do not actively maintain permissions after they are initially set. When the need arises to specify stricter access controls, they create more complex



Select Group Details:

● Local

○ Global

Group Name: group_J

Group Admin Credential: John

Group Admin Password: ********

Repeat Admin Password: ********

Local Groups - For use by your Private devices.

Global Groups - For use by Any assortment of devices.

My Device

device_J1 (02:f4:6f:0d:ee:a1)
Status: Available
Group owner: false

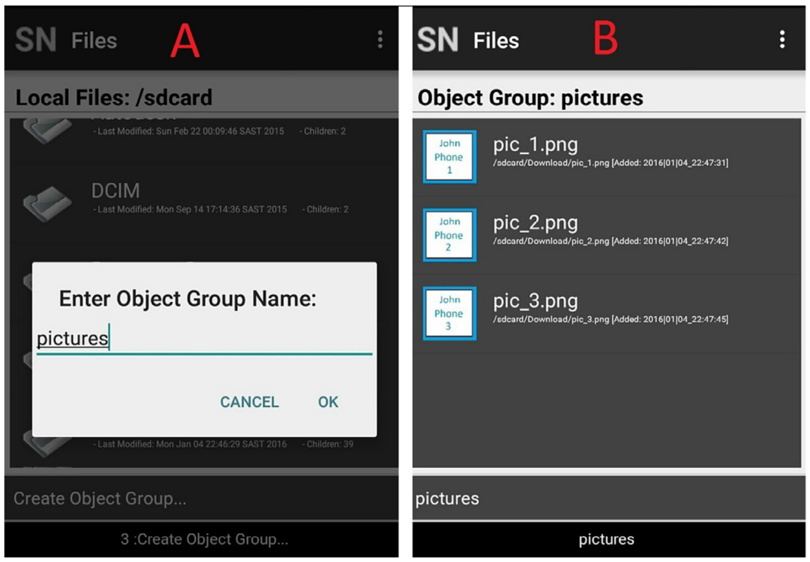
Local Groups

group_J
Group Owner: device_J1 (f235dacfa96b87b8)

Global Groups

group_home
Group Owner: device_J1 (f235dacfa96b87b8)

Fig. 8. Group creation, update groups page.



Local Files: /sdcard

- Last Modified: Sun Feb 22 00:09:46 SAST 2015 - Children: 2

DCIM
- Last Modified: Mon Sep 14 17:14:36 SAST 2015 - Children: 2

- Last Modified: Mon Jan 04 22:46:29 SAST 2016 - Children: 39

Enter Object Group Name: pictures

Object Group: pictures

pic_1.png
/sdcard/Download/pic_1.png [Added: 2016|01|04_22:47:31]

pic_2.png
/sdcard/Download/pic_2.png [Added: 2016|01|04_22:47:42]

pic_3.png
/sdcard/Download/pic_3.png [Added: 2016|01|04_22:47:45]

Fig. 9. Object group creation and adding files.

policies than administrators would, resulting in mistakes which can lead to compromises in security. To be able to implement efficient access control, users need intuitive and easy to use policy tools to assist them in this process. In mobileenvironments this is even more so, requiring tools to be pre-configured with default rules.

Users are supported in the prototype by a default rule that can be modified. Should John open *group_home*'s Group Management page and select edit group rules he will be taken to the Group Rules page shown in Fig. 10 (B).

When a user attempts to define access control for a shared folder in a group, a default rule is defined that limits access to all members of the group. The underlying rule shown by Fig. 10 (B) is stored on the device in an encrypted XML document which is not accessible. This rule is shown in Fig. 11.

If the user needs to modify the group rule, the options indicated by numbers 1–4 in Fig. 10 (B) can be used. *Family files* (1) indicates the shared folder or object the group rule applies to and cannot be changed. *Share with* (2) indicates a dropdown that enables the user to select whether or not to share the shared folder and *group_home* (3) indicates with whom. **+** (4) supports the addition of contextual constraints to the rule.

Should John wish to limit the times in which sharing within a group can take place, he could alter the rule as shown in Fig. 12 (A–C), resulting in the XML rule in Fig. 13.

7.4. Prototype evaluation

In order to test the effectiveness of the prototype, tests were performed. Two Android devices were used with the prototype in-

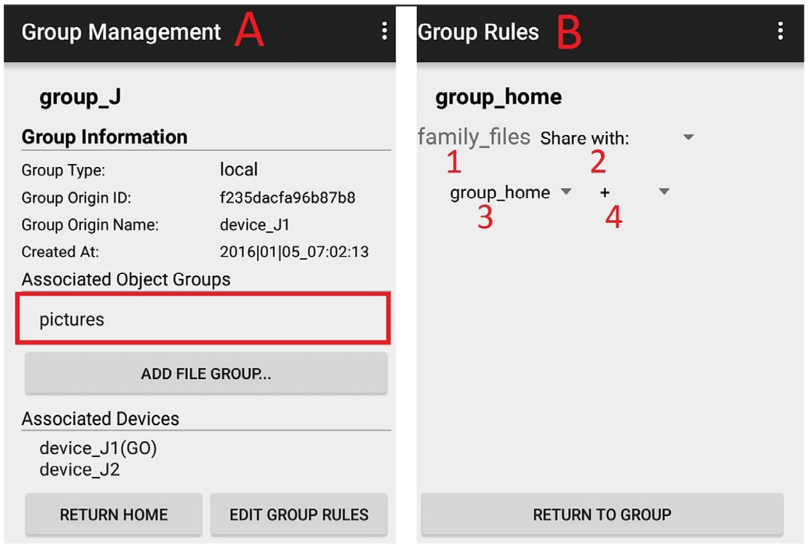
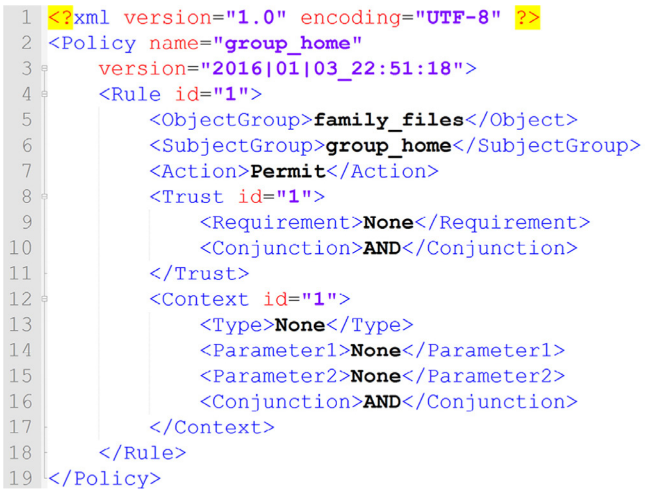


Fig. 10. Group management and group rules.



```
<?xml version="1.0" encoding="UTF-8" ?>
<Policy name="group_home"
    version="2016|01|03_22:51:18">
    <Rule id="1">
        <ObjectGroup>family_files</Object>
        <SubjectGroup>group_home</SubjectGroup>
        <Action>Permit</Action>
        <Trust id="1">
            <Requirement>None</Requirement>
            <Conjunction>AND</Conjunction>
        </Trust>
        <Context id="1">
            <Type>None</Type>
            <Parameter1>None</Parameter1>
            <Parameter2>None</Parameter2>
            <Conjunction>AND</Conjunction>
        </Context>
    </Rule>
</Policy>
```

Fig. 11. XML policy for *group_home*.

stalled. The devices used were a Samsung smart phone and a tablet both running Android Lollipop 5.0. The evaluation was limited to Wi-Fi Direct connection medium with the Android operating system Wi-Fi Peer-to-peer manager. When in proximity, the devices maintain a Wi-Fi Direct connection with each other. Smart space functionality is provided over this connection.

Once each device was set up with an administrative account and a group for their personal devices, a group was created by one device and the other was invited to the group. A shared folder for pictures and files was created. A general observation made was that devices that continuously try to discover other Wi-Fi Direct devices in a battery constrained device may be challenging to manage. Intelligent algorithms would be needed to manage trade-offs between discovery time and battery life.

A performance test was run on the Access Control Manager to determine the overhead it adds to the operation of the system. Ten performance tests were run with the Access Control Manager enabled and another ten with it disabled. The tests measured the amount of time taken from the instant that one device connects to another until the time the Access Control Manager renders its final verdict, or just before the prototype determines which files need sending or updating in the case of the disabled Access Control Manager. The average time measured for the ten runs without the Access Control Manager enabled was 929.1 ms. Once these tests were completed the Access Control Manager was re-enabled to test the complete duration of its operations, which measured an average execution time of 1316.9 ms. The additional time is due to the further checks that are made by the Access Control Manager over and above the normal Android permission checks. The individual results showed little deviation from the average which leads to the conclusion that the Access Control Manager component adds a negligible overhead of 387.8 ms to the operation of the prototype. As the number of policy rules that are processed is generally not large, it is not expected that the Access Control Manager would introduce any overhead to an application.

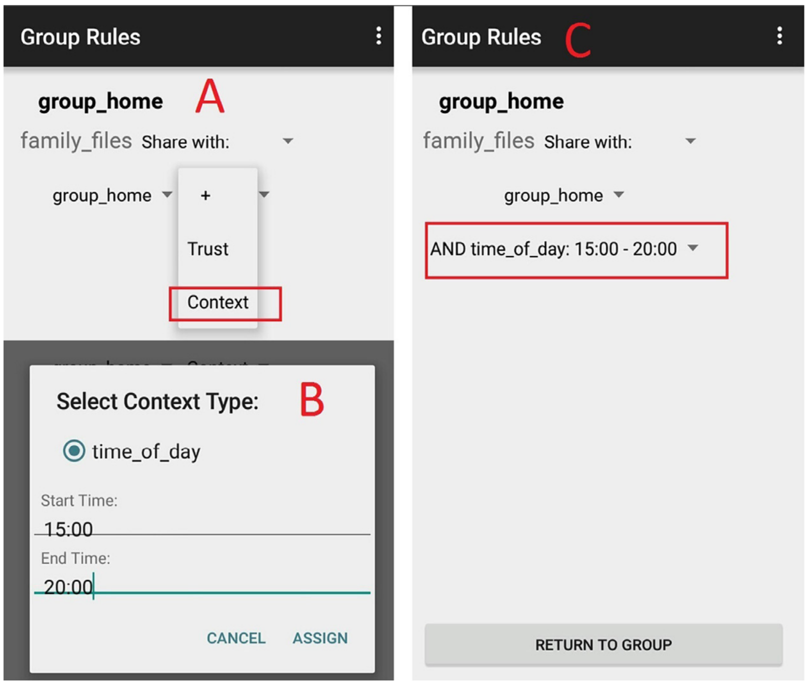
Tests were then performed to send one, five, and ten files between devices. The prototype required additional time to set up the application, create groups and invite devices, but after that no user interaction was required to send files. It was found that a 1 MB file has an average send time of 0.5 seconds and a 10 MB file has an average send time of 4.5 seconds. The main overhead introduced by the prototype would be the sending of large files, but this would also be the case if files are sent between devices in the traditional manner.

It should be noted that the ability to disable the Access Control Manager was specifically built into a special build of the prototype to perform tests. During normal operation of the full version of the prototype it is not possible to disable the Access Control Manager.

8. Conclusion and future work

This research concludes that smart space technologies can be used to share information and resources in a managed and secure manner when personal preference is considered. The foundation of the local personal smart space framework was built over existing smart space paradigms that were already tried and tested. The contribution of the local personal smart space framework is the definition of an access control model that takes personal preferences into consideration using local and global policies.

As verified in the prototype, it was possible to define rules that ensured that members of a group can set access control as long as rules do not conflict with each other. If conflict does exist, it can be resolved by following the preferences of the group owner. The local personal smart space framework was designed to be as user friendly as possible. However, smart devices have

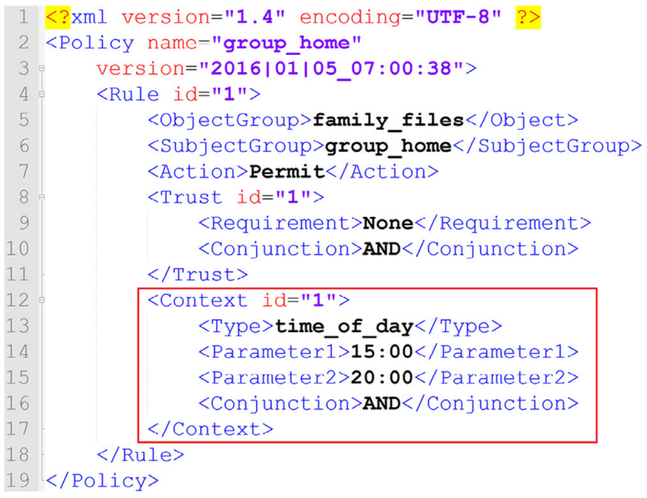


Start Time: 15:00

End Time: 20:00

AND time_of_day: 15:00 - 20:00

Fig. 12. Adding contextual constraints.



```
<?xml version="1.4" encoding="UTF-8" ?>
<Policy name="group_home"
    version="2016|01|05_07:00:38">
    <Rule id="1">
        <ObjectGroup>family_files</Object>
        <SubjectGroup>group_home</SubjectGroup>
        <Action>Permit</Action>
        <Trust id="1">
            <Requirement>None</Requirement>
            <Conjunction>AND</Conjunction>
        </Trust>
        <Context id="1">
            <Type>time_of_day</Type>
            <Parameter1>15:00</Parameter1>
            <Parameter2>20:00</Parameter2>
            <Conjunction>AND</Conjunction>
        </Context>
    </Rule>
</Policy>
```

Fig. 13. Resultant XML policy.

small screens and sub-optimal input options so compromises had to be made on the granularity of the policies defined by the policy manager. A default rule was defined whenever a shared folder was added to a group to support a user that is not familiar with access control policy specification. To address the sub-optimal inputs of smart devices, the local personal smart space 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. Even though the access control model complied with all the identified requirements, compromises had to be made which can be addressed in future work.

For example, the arbitrary trust level used in the model can be upgraded to a full-fledged, adaptive trust based access control model that would require less input from the user. Contextual information can be expanded to perform contextual profiling. The group management device for policy management can be replaced by a fully decentralized policy management suite that allows policies to be merged regardless of which devices are present. This could enable a truly decentralized 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.

References

Abowd, GD, Dey, AK, Brown, PJ, Davies, N, Smith, M, Steggles, P, 1999. Towards a better understanding of context and context-awareness In International Symposium on Handheld and Ubiquitous Computing (pp. 304–307). Springer Berlin Heidelberg.

Adiba, M, Labbe, C, Roncancio, C, Serrano-Alvarado, P, 2004. Context aware mobile transactions, mobile data management Proceedings. IEEE International Conference on, p. 167.

Al-Muhtadi, J, Ranganathan, A, Campbell, R, Mickunas, MD, 2003. Cerberus: a context-aware security scheme for smart spaces, pervasive computing and communications (PerCom 2003) Proc. of the 1st IEEE Int. Conf., pp. 489–496, 23–26 March.

Al-Rabiaah, S, Al-Muhtadi, J, 2012. ConSec: context-aware security framework for smart spaces In Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), Sixth International Conference on, pages 580{584, Palermo, IEEE.

Alicherry, M, Keromytis, A, Stavrou, A, 2009. Deny-by-default distributed security policy enforcement in mobile ad hoc networks Security and Privacy in Communication Networks. Springer Berlin Heidelberg, pp. 41–50.

Alliance, WF, 2010. Wi-Fi certified Wi-Fi direct White paper.

Ayari, M, Kamoun, F, Pujolle, G, 2008. Distributed policy management protocol for self-configuring mobile ad hoc networks Advances in Ad Hoc Networking. Springer US, pp. 73–84.

Backes, M, Gerling, S, Hammer, C, von Styp Rekowsky, P, 2013. AppGuard — enforcing user requirements on Android apps In 19th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS), Springer-Verlag.

Bakar, A, Talib, E, 2015. Formulating heterogeneous access control model for MANET emergency services. J Netw 10 (7), 407–412.

Bourgeois, M, Callaway, E, Gutierrez, J, Heile, B, Mitter, V, Naeve, M, 2001. IEEE 802.15.4: a developing standard for low-power low-cost wireless personal area networks. IEEE Netw 15 (5), 12–19.

Brezillion, P, Mostefaoui, G, 2004. Context based security policies: a new modelling approach Pervasive Computing and Communications Workshops, Proceedings of the Second IEEE Annual Conference on , pp. 154, 158, 14–17 March.

Bugiel, S, Heuser, S, Sadeghi, A, 2013. Flexible and fine-grained Mandatory Access Control on Android for diverse security and privacy policies In USENIX Security, pp. 131–146.

Cheng, Y, Ghosh, A, Chadha, R, Gary, M, Wolberg, M, Chiang, C, et al., 2010. Managing network security policies in tactical MANETs using DRAMA Military Communications Conference, IEEE, pp. 960–964.