Context-Aware Role-based Access Control (CARBAC) for Smart Spaces Using Semantic Technologies

Master of Science in Technology Thesis
University of Turku
Department of Information Technology
Networked Systems Security
2014
Shohreh Hosseinzadeh

Supervisors:
Johan Lilius (Åbo Akademi University)
Seppo Virtanen (University of Turku)
Natalia Díaz Rodríguez (Åbo Akademi University)

UNIVERSITY OF TURKU
Department of Information Technology

SHOHREH HOSSEINZADEH: Context-Aware Role-based Access Control (CARBAC) for Smart Spaces Using Semantic Technologies

Master of Science in Technology Thesis , 98 p., 0 app. p.
Networked Systems Security
February 2014

With maturity of information technologies and growth of human interaction with computers there is a growing exigency for computer security. Smart Spaces (SS) are types of Wireless Sensor Networks (WSN) with heterogeneous embedded and wearable sensors and devices. The goal of these spaces is to boost the quality of the people's lives by making their environments more intelligent, comfortable, and energy efficient places to live. The SS technology is recently being used in various public and private sectors, including health and well being (e.g., for health monitoring and remote rehabilitation means). Undoubtedly, like any other networked systems and health-care information systems, the information sharing raises some security and privacy concerns into debate, due to the high sensitivity of the information shared and stored in these systems.

The thesis is written as a part of the Smart Space Project at Embedded Systems Laboratory, Åbo Akademi University. The project aims at developing a smart box that provides smart services (e.g., remote rehabilitation monitoring and activity recognition) at home. The main objective of this thesis is to enhance the security in the smart spaces based on the Smart-M3 platform, and maintain the privacy of its users. In this respect, we proposed a solution to enhance the security and privacy in these environments using Semantic Web technologies. The proposed solution includes: a) designing a security framework that addresses the security requirements, such as authentication, authorization and access control, and b) proposing and implementing a Context-Aware Role-based Access Control (CARBAC) scheme. We modeled our access control scheme using ontological techniques and Web Ontology Language (OWL), and implemented it via CLIPS rules (a tool based on C-language used for developing expert systems). The privacy in our system is supported by data encryption techniques and privacy rules.

In order to evaluate the overhead that the proposed access control scheme introduces to the system, the response time for four different types of requests to perform actions (supported by SSAP protocol) is measured. The requests include the request for reading, inserting, updating, and deleting a triple, for different sizes of the SIB (ranging from 1 to 100000 triples). We used the Python language in the experiment, which is one of the supported languages by KP Interface (KPI). Through the obtained results, we concluded that the proposed idea is considered as an efficient way of controlling the access in the Smart-M3 framework at triple level.

Keywords: smart space, Smart-M3, computer security, privacy, e-Health, Semantic Web

Acknowledgements

I would like to express my sincerest gratitude to Professor Johan Lilius, who gave me the opportunity to conduct my study in his research group, introduced me with the field, supported and encouraged me throughout the whole work.

My warm appreciation goes to Adjunct Professor Seppo Virtanen, who led me in the two years of my Master studies and also supervised me in my thesis with his precious comments.

I am also grateful to Natalia, my thesis advisor, who was a great colleague and friend. Natalia, your assistance was priceless and it was always joyful working with you. I would also thank to other members of the the project, Smart-M3, and other labmates in ESLAB, who helped me a lot and made the atmosphere a friendly place to work.

I would like to say a special thank to Mehdi for being exceptionally supportive in the tough days, and also my friends in Turku, who cheered me up and colored my life in the sad days far from home.

Finally, my dear family, I owe you more than thanks for your unconditional love and support. I would happily dedicate my thesis to my dear Mom and Dad.

Contents

Chapter 1

Introduction

With the advance of information technologies and growth of human interaction with computers, there is a growing need for computer security. The term security in computer industry appertains to all measures taken to retain the data safe. Computer Security, also known as Cyber Security, is concerned with preventing, detecting, and confronting with the security attacks [1]. It is defined by National Institute of Standards and Technology (NIST) [2] as:

> ”The protection afforded to an automated information system in order to attain the applicable objectives of preserving the integrity, availability, and confidentiality of information system resources (including hardware, software, firmware, information/data, and telecommunications)”

Furthermore, various computer systems cooperate with each other in order to provide better services. Information sharing is the element that makes this cooperation feasible, but this sharing itself raises privacy concerns requiring critical discussion. In this respect, education and study in information privacy has gained a lot of interest recently, and has developed rapidly in vast range of domains, such as health care, finance, politics. Privacy of the information becomes more momentous when it comes to personal information. A breach in the information system not only violates the personal privacy, but it also increases the risks of someone taking advantage of the compromised information in some malicious activity, for example identity theft. Undoubtedly, there is a coherence between security and privacy of the information on the networks. Hence, a more secure network, ascertains the privacy of the information more reliably.

Smart spaces are types of wireless sensor networks, that are composed of sensors, devices and data providers that co-operate with each other in order to make people's lives smarter and more comfortable. They also have shown to be successful in efficient energy consumption. The technology is recently being used in public and private sectors, such as smart buildings, smart lighting systems, health monitoring and remote rehabilitation systems.

This thesis has been written as a part of the Smart Space Project running in Embedded Systems Laboratory, Åbo Akademi University. The project focuses on the development of a Smart-M3-based home gateway for storage and processing of information such as various sensor inputs. An example of services the gateway provides is virtual remote rehabilitation monitoring with the help of depth sensors such as Kinect. Due to the fact that the information in any health-care system is remarkably sensitive, it is highly significant to ensure the security and privacy of the data while it is stored or being transmitted. In this thesis we study the security and privacy related issues and existing security approaches regarding smart spaces, and we propose a solution to enhance the security of these environments using the Semantic Web technologies. The proposed solution includes a security framework and an access control scheme. The security framework is composed of various components collaborating together to support different aspects of security, such as authentication, authorization and access control. The proposed access control scheme is Context-Aware Role Based Access Control (CARBAC), that controls the access of the users to the system in accordance to their role in the system and the current context information. We model our access control scheme via ontological modeling techniques (ontology is known to be one of the best ways to present knowledge), and implemented it via CLIPS rules (CLIPS is a tool based on C language used for developing expert systems). In the end, we evaluate the overhead that the proposed access control introduces to the system.

The thesis is structured in five different chapters, commencing with an introduction in chapter 1, followed by a preamble to smart spaces in chapter 2. In chapter 3, security and privacy of smart spaces, and state-of-the-art in the domain is discussed. Chapter 4 encompasses the proposed security framework and access control approach along with evaluating the approach and analyzing the result. Conclusion is discussed in chapter 5.

Chapter 2

Smart Spaces (Smart Environments)

Smart spaces (SS) enrich people's lives by creating intelligent environments that serve users by automating their daily routine activities with as little interaction with them as possible. Smart homes, smart offices, smart buildings and smart cities are examples of such environments. The space consists of a multitude of wearable and embedded computers, sensors, information appliances, electronic and tags. that obtain information about the environment in order to perform appropriate actions. The goal of the smart spaces is to bring comfort and efficiency to the their inhabitant's lives [3]. Daily routine activities of inhabitants in a smart space often follow a special pattern. By observing, analyzing and learning of the pattern of users' last activities, the intelligent environment predicts the next action and controls the devices accordingly. This way, the smart environment adapts itself to user preferences [4, 5]. Lately, smart spaces are potentially being used for various purposes including education, lighting and temperature control systems, meetings, as well as health and well-being (section 2.1) which is the main focus of this thesis. The following subsection gives a broader view on how the smart spaces work, from gathering the information to putting the information into actions.

Smart Space Architecture

As depicted in figure 2.1, the architecture of a SS is composed of four layers: Physical Layer, Communication Layer, Information Layer, and Decision Layer. In the physical

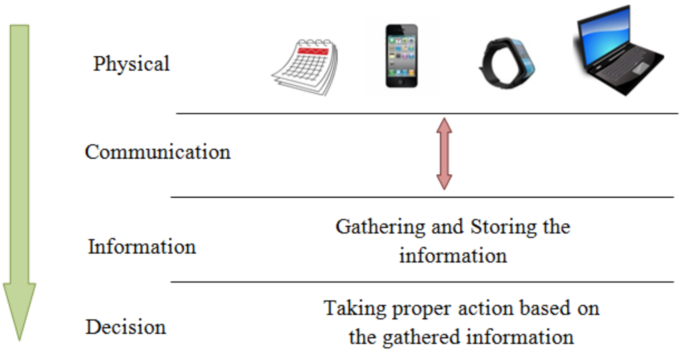

Physical
Communication
Information
Gathering and Storing the information
Decision
Taking proper action based on the gathered information

Figure 2.1: Smart Space Architecture. (On the left downwards different layers of a smart space are illustrated, and on the right it is shown how the collected data is put into actions [4, 6].)

layer, the raw data is generated by people or devices (e.g., sensors capturing environmental information) and sent over to the information layer through the communication layer. The received data is stored, managed and used to produce serviceable knowledge for decision making. In case of new information, the decision layer chooses the next action to be taken, based on the acquired data. [4, 6]

Smart Home

In a smart home, the purpose is to free inhabitants from performing some of their routine tasks and purvey a more convenient life environment to them. In order to provide better services, the smart system must be programmed particularly for the occupants of the house and be updated in accordance to the changes in their lifestyle [7]. Also the services should be customizable for different users residing in the space, to fulfill various needs of different themes (e.g., busy family, families with children and senior members). In [8], an idea to identify and track the users of a smart home by their footsteps via force-sensitive tiles on the floor (smart floor) is proposed.

The smart home developed by S. H. Park et al. [9] keeps records of the living patterns, preferences and habits of the inhabitants (lighting, sounds, shopping favorites) in different spots of the space. Later it can make the best decisions according to its users' life patterns (such as playing the favorite relaxing music when user goes to bed). In this project, the smart home entails a set of smart objects, including: a) a smart wardrobe that automatically checks the weather condition and recommends proper clothing to the user. Additionally, the clothes could be labeled for laundry function after being used. b) a gate reminder that reminds the user about the necessary things to take before leaving the house. c) an electronic paper to display e-papers that provides more convenient and pleasant reading. d) a smart plant. The DigiFlower, through its receivers, realizes whenever any of the family members is nearing the house and blossoms. e) a smart refrigerator that alerts the user about the expiration of the food items and takes care of the shopping list.

Smart Office

An example of a smart office is shown in [10]. In this space by gathering the information about the current mood of the worker in the office, the environment is adjusted in a way to create a better mood for the worker. A camera and a blood flow sensor are used to provide input data. The camera monitors the individual's face and body movement to assess the level of sleepiness and concentration, and the sensor determines the level of tiredness. Thence, according to the attained knowledge, the actuators start to function and affect the environment, in order to relief the exhaustion of the user. For instance, a motion chair generates slight moves, the speakers play suitable background music, lighting devices and aroma generator are adjusted to the user's psychological mood, and finally an automatic coffee machine serves a cup of coffee.

Smart Hyper-space or Hyper-environment

A person's daily life is in the interaction with diverse spaces and environments, such as home, office, library, gym. Even though all these environments can be assumed to be smart, they cannot serve the user as they are isolated from each other. Therefore, there should be an association and exchange of information between the spaces. This cooperation of separated smart spaces composes a high-level space termed as Smart Hyperspace or Hyper-environment. UbicKids Project is an example of a smart hyperspace which assists parents in taking care of their kids. It is known as a hyperspace because of involving more than one space a child might be in (home, yard, road, car, etc.) [11]

2.1 Smart Spaces in Health and Wellbeing (HWB)

With the aging of the population and due to the fact that elderly people feel safer at their own home, there is a growing need for the indoor assistant applications and technologies for seniors to boost the quality of their lives and give them the chance to live safer, more independent, convenient and active [3]. In this respect, a smart care-giver technology could be remarkably advantageous to support aging. The needed support ranges from helping people to accomplish their daily tasks (such as reminders to take their medications and monitoring their own health indexes) to emergency alerts in case of accidents [12]. In these applications, different aspects of the individual's health are taken into account in order to help them with a thorough wellbeing program. Three different categories of factors are studied in [3], including mental wellbeing (help people to deal better with the challenges of their quotidian activities), physical wellbeing (attempts to elevate the quality of people's diet), and social wellbeing (tries to improve quality of their social relationships with others). Pollack [13] has classified the types of assistance provided by the smart technologies into three main categories: assurance of the safety of the individual, supporting individuals with their impairments, and assessing and specifying the wellness condition of the individual. A smart assistant at home observes and indicates a model for the user's behaviors. Thence, this model is used to verify unexpected changes in behavior and identify the emergency situations [5].

The use of smart spaces for rehabilitation purposes has gained a lot of interests recently. For instance, in a smart home, by automatic monitoring of the physical behavior of the patients, care givers are able to study and evaluate the impacts of the medications, treatments and regimens. The raw data is collected via data collectors (e.g., wearable sensors) and are analyzed by machine learning techniques to model patterns for the subject's activities. The patterns are used to distinguish any anomalous situation in the individual's behavior.

Ambient Intelligent (AmI) is a novel domain in Information Technology (IT) that endeavors towards enhancing the quality of life of humans by bringing more automation to their daily activities. To fulfill this desire, different aspects of science (including Artificial Intelligence, Psychology, Software Engineering, Mathematics) are coming together to build up a multi-agent perimeter that interacts and monitors the subjects in order to provide care services. [14] There are numerous projects and groups researching and developing these environments. **Ambient Assisted Living (AAL)** [15] is a European foundation active in the field of AmI, which attempts to improve the living condition of senior citizens by applying Information and Communication Technology (ICT). In their projects they attempt to amend the safety, security, self-confidence, and sociality of the older adults. HOPE, HELP, AMICA, and JOIN IN are some instances of these projects. *HOPE* project is a smart platform helping people, especially with Alzheimer's disease, at home with the functions like monitoring that the patient does not leave home, reminding him with the ingredients while cooking, etc. Additionally, in case the system perceives the situation to be risky (e.g., the patient fainting), it talks to the patient and calls for help. *HELP* project is a drug delivery system for the patients with Parkinson's Disease. The health parameters (blood pressure) and kinetic states (gait patterns) of the subject are measured through sensors and devices to determine the appropriate dose of drug needed. *AMICA* project is designed to provide medical services for Chronic Obstructive Pulmonary Disease (COPD) patients through the psychological signals attained by the sensors and consultations at home. *JOIN IN* project aims at socializing elders via a gaming platform including multi-player games and exergames. Apart from these games, end users are able to communicate and chat with other users, share their matching fondnesses, perform group exercises, and so on.

Lately, Microsoft Kinect is being used not only for entertainment purposes, but used by researchers as a 3D-depth sensor to track and detect the user's body movements. Cameras, typically, store the captured images, which threatens the privacy of the subjects. On the contrary, 3D-depth sensors do not store the images, but construct and store a skeleton structure of the individual's body. Díaz Rodríguez et al. [16] are using Kinect to model and study the user's movements and interactions by monitoring the duration, repetition, and intensity of the workouts performed and giving feedback to them (e.g., the quality of the exercises done or improved in a period of time). The aim is to achieve remote rehabilitation and two-sided feedback for the user and for the physiotherapist.

2.2 Semantic Web

The Semantic Web (SW) refers to the web of data that represents semantics (meanings) through linking of related information. It does not aim at replacing the existing World Wide Web (WWW), but to extend it in order to improve interaction of the data. The SW is mainly designed to be used by machines or applications, while WWW is presented in a way to be used by humans. [17]

SW technologies are known to be successful in demonstrating the contexts and reasoning [16], and the same technologies are being used in smart space for storing and representing the data. Therefore, in the following comes a brief introduction to the semantic technologies.

The SW is composed of elements such as statements, ontologies, URIs, reasoners, and rule engines [17]:

Statement: The SW is built over the statement blocks. The statements enable more flexible and meaningful expressing, sharing and inferencing of the information, in a way that the SW is better understood than WWW. The SW uses Resource Description Framework (RDF) to store and represent data in a machine understandable manner. RDF, which is the triple of three components- subject, predicate, and object- is structured as a graph. Subjects and objects are the nodes of the graph, and predicates (properties) form the edges. The following example (figure 2.2) illustrates how a predicate connects the subject to the object, 'Shohreh' is the subject, 'Natalia' is the object and 'knows' is the predicate.

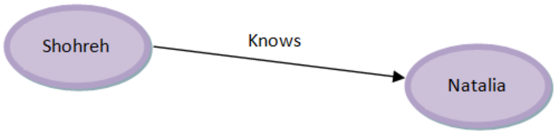


Figure 2.2: RDF Graph is a triple of three components (subject, predicate, object). In this example the triple is: (Shohreh, knows, Natalia)

URI: All items (subjects, predicates, and objects) are uniquely identified via Uniform Resource Identifiers (URIs) over the Internet.

Ontology: Ontology is made up of the statements to model concepts, their relationships, and their properties.

Reasoner: Reasoning the information helps the SW figure things out by inferring new relationships based on the ontology model, which acts as the equivalent of a schema if we compare it with a relational model. A simple example of the reasoning of data is:
Duck is a bird. Bird is an animal. Thus, *Duck is an animal.*
Reasoning is one of the advantages of the SW that makes the contradictions possible and also generates a link between related information, which enables a more flexible navigation.

Rule Engine: Rule engine does the similar function as reasoning, with the difference that, it follows rules instead of ontologies.

Due to the limitations of RDF in expressing the ontologies, OWL (Web Ontology Language), an expressive semantic mark-up language, was designed by the World Wide Web Consortium (W3C) Web Ontology Working Group. By having further vocabularies, it provides better facilities than RDF, RDFS (RDF Schema) and XML (eXtensible Markup

Language). Three sub-languages are defined for OWL: OWL Lite, OWL DL, and OWL Full. [18] As discussed in [19], an ontology language needs to be expressive, well-defined (in syntax and semantics in order to be processed better by machines), and should support reasoning.

SW attempts to present the information on the web in a machine understandable manner to provide automation. The SW forms the resources of the web as RDF in accordance to the ontologies; however, due to the fact that the web is not properly respondent to the dynamically changing environment of agents, Smart-M3 is developed to dynamically share semantic information. [20]

2.3 Smart-M3

There exists different software implementations for smart spaces, some examples are [21]: RIBS (RDF Information Base Solution) [22], Smart-M3 [23] (section 2.3), and ADK [24]. This thesis relies on Smart-M3 platform and focuses on the security and privacy aspects of it. Thus, in the following comes the history and specifications of this platform.

Smart-M3 was originally developed at Nokia Research Center at 2009. It commenced with the two main projects, SOFIA (Artemis Smart Objects For Intelligent Applications) and DIEM (Tivit Device Interoperability Ecosystems). The Smart-M3 project is now available as open source at [23]. FRUCT Association is one of the organizations actively working on this field, developing their projects based on Smart-M3. [25]

Smart-M3 is a Multi-device, Multi-domain, and Multi-vendor platform to share the semantic-based information. It is an inter-operable approach based on the principles of the SW [23].

There are several benefits known for Smart-M3 that motivates projects to take advantages of it. The most remarkable feature is the publish/subscribe paradigm implemented on top of the RDF store which is missing in majority of semantic RDF stores. With this feature, the applications, sensors and devices using Smart-M3 interface can subscribe to changes in the underlying RDF store. In this way, subscribers are being notified automatically any time a change happens to the subscribed resource. Subscription mechanism avoids constant launching of queries to evaluate if a certain condition is satisfied. It also avoids bottlenecks to happen. The other advantage of Smart-M3 is the complementary support to semantic web standards, i.e., the original SSAP protocol. The protocol allows multiple operations including insert, update and remove triples, subscribe and unsuscribe to a triple pattern, join and leave a concrete SS (the descriptions of the operations are discussed later in detail).

A. Kashevnik [26] claims that smart spaces based on Smart-M3 could be considered as an evolution of cloud computing, with some differences. In Table 2.1, cloud computing is compared to smart spaces [27].

Table 2.1: Smart Space vs. Cloud Computing

Smart Space	**Cloud Computing**
Personal and user specific	Not personal and vendor specific
Control by user	Control by supplier
Continuous connection to the network	No need for continuous connection
Memory and computational resources are limited to the capacity of the devices in the space	Unlimited computational resources and memory
Applications decided by user	Applications decided by service provider
Concerns: Data ownership, data sharing	Concerns: Data privacy and ownership

Smart-M3 platform is composed of two types of components: *Semantic Information Brokers (SIB)* and *Knowledge Processors (KP)* (figure 2.2). SIB is a hardware/software entity that acts as the back bone to the space and shares the information between KPs. The SIB supports the SSAP protocol and provides all the related services to the KPs. A SIB is formed of five layers, working together handling interoperable information exchange: Transport Layer, Operation Handling Layer, Graph Operation Layer, Triple Operation Layer, and Persistent Storage Layer [20].

KPs are agents in the smart space including devices, sensors, people, and other data providers that can affect, produce and consume the information on the space [28, 29]. The space may consist of more than one SIB; they are connected to each other and make information synchronized so that regardless of which SIB the KPs connect to, they see the same information. KPs have loose coupling communications, i.e., they communicate merely by inserting and consuming the information on the SIB. Thus, there is no direct connection between any pair of KPs, but they connect indirectly via a SIB. [20]

A smart space may be composed of several smart spaces joining together with a mediator KP to exchange knowledge between spaces [30]. Figure 2.3 shows a functional architecture of a Smart-M3 based space.

Smart Space Access protocol (SSAP) is the protocol used to support communication between SIB and KP. It allows the KP to query, join and leave, insert and remove, subscribe and unsubscribe to the smart space [30]. The following are the operations KPs can perform in a smart space:

- Join: A KP sends its credentials to the SIB and attempts to build a communication session. If the status was OK, the SIB sends its own credentials back to the KP.
- Leave: By terminating the session, the KP leaves the space and is not able to consume or insert information any longer.
- Insert: A KP after joining to the space can generate new information on the space.
- Remove: A KP in the space is able to remove the existing information from the space.
- Query: KPs can query and consume the information existing on the space. The result is either a list of triples, or a string containing ”Yes” or ”No”.

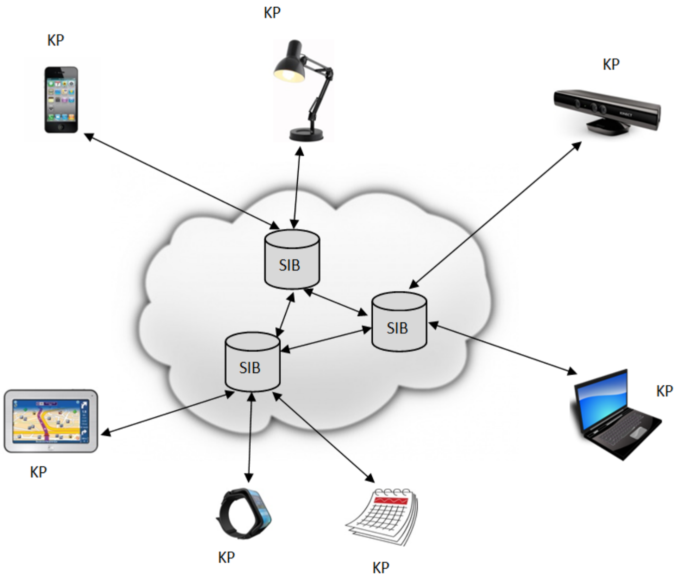


Figure 2.3: Smart Space (The space may consist of several SIBs that are connected to each other and synchronize the stored information. KPs access to the SIBs in order to share information or connect to other KPs. KPs may be in the form of gadgets, sensors, devices, people that consume and provide data. The two-sided arrows illustrate the bilateral flaw of data between the SIBs and KPs)

- Subscribe: A KP can subscribe to a SIB. The subscription process is a persistent query operation stored in the SIB and, by any new changes in the content of the smart space (about the information added or removed), it automatically re-evaluates the query and reports the result to the subscriber KP. In this way the KP is notified about the new changes and synchronizes its local information with the shared information on the space.

- Unsubscribe: Subscription query can be revoked by the Unsubscribe operation, so that the KP will no longer receive any update.

Smart spaces use several technologies, such as XML (eXtensible Markup Language) to encode data, RDF to represent the information, and ontology languages (e.g., OWL) to describe concepts and relationships [21](discussed more in Section 2.1. Semantic Web).

Chapter 3

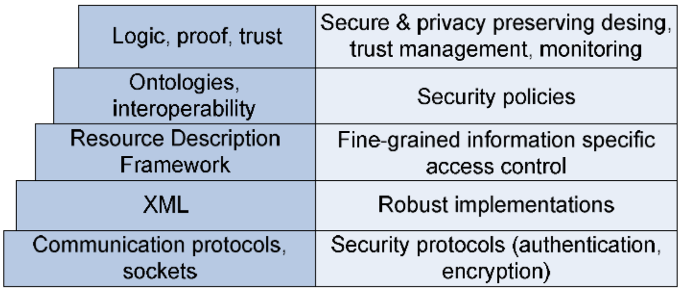
Security and Privacy in Smart Spaces

Security in smart spaces is considered to be challenging because of the nature and characteristics of these environments. Some of these features are the following [29]:

- *Dynamism*: Devices leave and join the space continuously and therefore change the environment. Hence, it is not feasible to anticipate all upcoming situations beforehand and define all security policies at design-time.
- *Heterogeneity*: There are numbers of devices in the space using different technologies to interact and have various security characteristics.
- *Distinct Security requirements*: In various situations, different security objectives and requirements are followed. Sometimes integrity is the highest priority, while in some other situation, preserving the privacy is the biggest concern.
- *Openness* and free use of information in smart environments brings more security challenges in addition to the traditional security risks.

According to the context and security requirements of the smart spaces, different levels of security are required. For instance, in a confidential meeting higher security is needed in comparison to a personal space [31]. And it should be taken into account that, the same security mechanism is not suitable for securing all smart spaces [29].

There are diverse security technologies developed that smart spaces can benefit from, such as secure communication offered by cryptography, robust implementation techniques, authorization and controlling access over resources, and key management techniques. Figure 3.1, known as Security Cake, shows the existing security mechanisms that could be supplied to SW. On the left there are different layers of SW [32], and on right there are the security elements necessary for smart spaces. [22]



Logic, proof, trust	Secure & privacy preserving desing, trust management, monitoring
Ontologies, interoperability	Security policies
Resource Description Framework	Fine-grained information specific access control
XML	Robust implementations
Communication protocols, sockets	Security protocols (authentication, encryption)

Figure 3.1: Security Cake, quotation from [22] (different layers of SW are listed on the left, on the right there are the security mechanisms that could be applied to a SS.)

In this literature review, miscellaneous security perspectives concerning the smart spaces, applying existing security technologies to these environments, and the existing approaches in the field of security and privacy in smart spaces are studied. This chapter summarizes the perused security aspects (including authentication, authorization and access control, communication security, privacy, vulnerabilities and attacks) and the state-of-the-art in the domain.

3.1 Authentication

With the emergence of technology, the need for personal identification to enhance the security has been increased. In the process of authentication, the validity of the claimed identity of the user is surveyed. [33]

There are different methods developed and used, such as password-based, token-based and biometrics-based, to authenticate individuals [33]. In the following we describe each method and the security challenges faced by them.

1) Password-Based Authentication

A Password based authentication system works by comparing the password presented by the user at the time of login with the previously stored one, for that particular user ID. The role of the ID is to authorize the user and in some cases, determine the rights a user may have due to its status (e.g., the superuser). [33]

In fact, employing passwords for the means of authentication has advantages such as being fast and easy to use; on the other hand, it brings plenty of disadvantages and deficiencies. Passwords are prone to be forgotten and subject to attacks. Here are some examples of the threats to passwords as stated in [33]:

- *User Mistakes:* 1) Users may desire to write down their passwords, not to forget them. The written passwords could then be read by an adversary. 2) Users also may give their passwords intentionally to others, e.g., a colleague or a family member. 3) Users may be the target of social engineering tricks and disclose their passwords.

- *Dictionary Attack:* An Attacker might be successful in discovering the password by finding a match via comparing the hash value of the stored passwords with the hash value of the popular common passwords.

- *Popular Password attack:* People usually prefer to take passwords that are simply recalled, which makes them effortless for the attackers to guess.

- *Monitoring:* Passwords are at risk of an eavesdropping attack in remote logging systems.

- *Multiple Use of Passwords:* A single password that is used in multiple networks makes the attack more feasible.

- *Password Guessing:* By gaining knowledge about the user and the policies of the system (e.g., required length of the password), attackers may try to guess the password.

2) Token-Based Authentication

Tokens are the objects at disposal of the individual which are used for authentication purposes. Literally, in this method the token is authenticated, rather than the user, and the user authenticates itself to the token, by means of a PIN (Personal Identification Number). Smart card is the most significant type in token based authenticators. It contains an embedded chip that carries the data. The chip is read by a reader in a challenge-response process. An adversary may get physical access to the card, or access to the data on the chip through cryptographic attacks. [33]

3) Biometric Authentication

Among authentication mechanisms biometric recognition systems have drawn a lot of attention over the past century, because of having advantages over other methods. They are secure, fast, and convenient to use. The identifiers usually use i) What the user carries, e.g., keys and cards, which have the risk to be stolen or to be lost. ii) What the user is, e.g., usernames and passwords that might be forgotten. iii) What the user knows, e.g., face, fingerprint, voice. The last group is known as the most secure and reliable authentication method, the hardest to forget and is not something to be forgotten, nor stolen. This is what is called Biometric Authentication. [34]

Biometric Authentication is the automatic recognition of users using their biometric characteristic. Any behavioral or biological characteristic that is *universal* (everyone has it), *distinctive* (any two persons are disparate by having this characteristic), *permanent* (invariant over time), and *collectable* (quantitatively measurable), could be taken as biometric recognizer. [35]

Biometrics, due to the considerable advantages they have brought for security, are widely being used these days. There are three main groups of applications, taking benefit from biometric authentication technique [35]: 1) Commercial applications, such as computer network logins, Internet Access, ATM, cellular phone, PDA, medical records management, and distance learning. 2) Governmental applications: such as national ID card, correctional facility, driving license, social security, welfare disbursement, border control, and passport control. 3) Forensic: such as corpse identification, criminal investigation, terrorist identification, parenthood determination, and missing children, relies on experts to compare biometric features.

In general, biometric features are captured and stored in a database, the first time the user enrolls in the system (figure 3.2.a), and later on, they are used to provide secure personal recognition for *identification* or *verification* means. Identification is a one-to-many comparison that searches templates of all users in database for a match, to check whether the identity of the user who is attempting to log in to the system exists in the database. This also prevents a single person from using multiple identities (figure 3.2.b). The verification process compares the captured data of the user with the stored data in the database to check whether the user is really who he/she claims to be and also prevents multiple people from using the same identity (figure 3.2.c). [34]

The architecture of a biometric authenticator system is constituted of four main modules [35]:

- *Sensor* in a biometric-based authentication system is used for taking biometric samples. The sensor could be in the form of a scanner, a camera, and an audio device.

- *Feature Extractor* converts a raw biometric sample acquired by the sensor to a digital representation for further processing. In this step, a quality test should be done to reject unsuitable images, the unwanted noise should be removed from the sample by a filter, and the signals should be normalized.

- *Matcher* compares the extracted sample with the stored template in the storage through pattern recognition techniques. Then the result, reported by a numeric

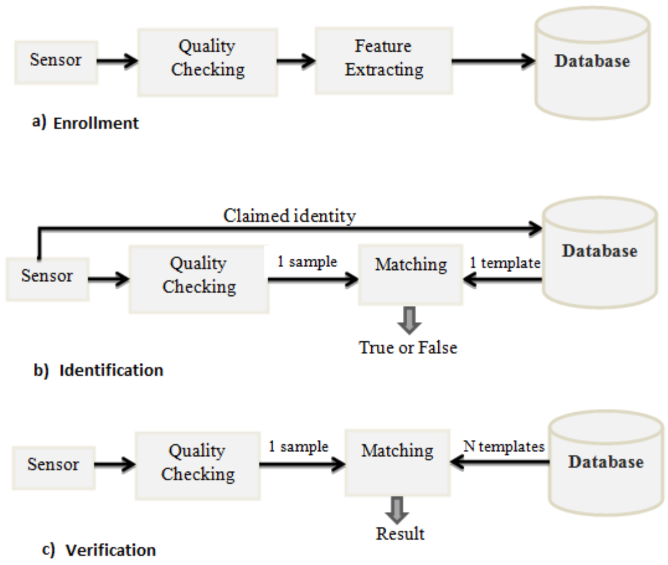


Figure 3.2: Architecture of a Biometric-based Authentication System (a) enrollment: a sample of the user's biometric feature is captured and stored in the database, when the user is enrolling in the system for the first time. b) identification: the system searches the whole database to find a match with the presented sample. c) verification: the presented sample is compared to the template stored in the database, when the user logs in to the system, to check whether the user is really who he/she claims to be.)

score, shows how similar they are. If it is greater than the pre-defined threshold it is accepted, and if not, it means that the sample does not match.

- *Storage* is where all the templates and information are stored. In the enrolling process the database should make sure that this user does not exist already.

With the high accuracy the biometric authentication brings, there still are some possibilities for errors that can affect the accuracy of the authenticator and may result in false match or false non-match. For instance, the two samples taken in different times may not be exactly the same because of imaging conditions (e.g., sensor noise and dry fingers), changes in the user's physiological or behavioral characteristics (e.g., cuts and bruises on the finger), and ambient conditions (e.g., temperature and humidity), and user's interaction with the sensor. Therefore, there is a factor called matching score introduced to represent the similarity between the input sample and the previously stored template. The higher the score, the more certain it is that the two samples are from one person. [33]

Various biometric features

Biometric characteristics that fulfill the requirements mentioned above, could be used for the means of personal authentication. Among all biometric characteristics known, iris is the most secure, and fingerprint and facial image are the most commonly used ones.

- *fingerprint* is a pattern of the rigid and valleys on fingertip. It has several advantages that make it a favorable method to use, such as the high accuracy (even the fingerprints of identical twins are different), and the inexpensiveness of fingerprint reading machines. On the other hand, there are some drawbacks: for a portion of the population this method is not usable, because of aging, genetic factors, occupational reasons (the cuts and bruises on manual workers' hands make the fingerprints change). The systems using fingerprint for authentication, especially when it is used for identification, require too much computational resources. Additionally, it requires a direct connection between the sensor and the user's body, which might not be pleasant to some users. [35]

- *Facial Image* is the most common biometric feature. The face recognition applications could either be based on the overall analysis of the facial image or analysis of the facial attributes such as the shape and location of lips, eyes, eyebrows etc. Although it is known as an adequate method it is vulnerable to impersonating attack. This attack happens by playing back the video of a person without his/her presence. To prevent such attacks, liveness testing (e.g., lip motion and body temperature) could be helpful. In addition to the vulnerabilities this method has, an image of a person's face could simply be taken in public without his/her consent. [35]

- *Iris* is the annular region of the eye bounded by the pupil and the sclera (white texture) on either sides. It is formed in the first two years of life. Among the personal identifiers iris is the most secure one because of its stable reliable behaviors (it is unique, not identical even in two eyes of a same person). Its texture is not mutable over time, nor does it shrink or stretch by light. The image is acquired optically without any physical contact which is more desirable for the users and makes registration easy. Nevertheless, like any other biometric identifiers, it has plenty of downsides, such as the expensive equipment needed, the social acceptance is rated to be low, a lot of memory for saving the data is required, and since iris is a visible organ, images of it could be taken without the consent of a user which propounds privacy issues. [36]

Attacks on biometric systems

The goal of the biometric system is to identify and verify the authorized persons at the time of entering the system, whereas the False Acceptance (FA) threatens the most systems when an unauthorized user succeeds to access the system. False Rejection (FR) occurs when a legitimate user fails to access the system. In addition to the errors raised from the inaccuracy of the system, there are numerous attacks threatening different units of a biometric system [34, 37]:

1) *Attacks on sensor*: Most of the serious attacks happen in this unit and because it is the entrypoint of the system, security mechanisms such as digital signature and encryption are not helpful. The following are some instances of possible attacks on a sensor: a) *Impersonating*: changing the appearance in a way that the image matches a stored template. b) *Coercive attack* arises when the legitimate user provides data to an attacker for example because the user is threatened or forced. A security camera could be useful in this case. c) *Replay attack* happens when the biometric data of a user is recorded and played again without user's presence. Liveness testing is helpful to assure that the legitimate user is representing the biometric trait. d) *Local storage of data* by some sensors may make the

stored data the aim of an attack. The stored data could be retained and used maliciously. Clearing the sensor's local memory and buffer , when not needed, prevents these type of attacks. e) *Fake physical biometric device*: the latent biometric data such as finger and hand prints could easily be found any where without the owner's consent, and might be used to make a fake biometric identifier by using cheap materials and limited knowledge. Many biometric authentication systems based on identifiers, such as fingerprint, hand print and iris, are attacked this way. Liveness testing (e.g, continuous movements in iris-based recognition, pulse detection for fingerprint-based recognition) may help making sure that a live person has presented the sample. f) *Latent print on sensors*: these prints may be copied and misused later like the example for fingerprint spoofing.

2) *On feature extractor*: In feature extractor false data may cause a DoS (Denial of Service) attack, disable the system, and harm the software/firmware of the system.

3) *On matcher*: Attacks on this part make the matcher produce a false score. A well-known biometric attack in this section is ”hill climbing” that works with a man-in-the-middle attack in which the attacker changes the image in a way that increases the score. The attacker can then use that image as input to the security system to which the original template belongs. Any biometric system offering a score is susceptible to this kind of attack.

4) *On database*: Examples of attacks on database are re-enrollment with a new name, modification in the content database, and privacy attacks. Privacy attacks occur when an attacker gain access to the information on the database and use it for malicious purposes. This information may consist of personal information of the users of the attacked database.

5) *On channel*: the attack could happen: a) on the channel between the sensor and the feature extractor. Replay attack is an example of such attacks that may happen at this point. b) on the channel between the feature extractor and the matcher. c) on the channel between the database and the matcher to change the presented sample to the matcher. d)

on matcher's output to override the result. With the help of cryptography the templates could be protected during transmission.

6) Other significant attack types: Cutting the power to paralyze the system, cryptanalysis of the encrypted data, piggy-back attack, swamping attack.

3.1.1 Data Provenance

Cryptographic systems are categorized depending on [33]:

1) How the input plaintext is converted into cryptotext: In this manner, there are two main cipher methods, Block Cipher and Stream Cipher. In *Block cipher* the plaintext is split into blocks and processed consecutively, to generate blocks of output; while in *Stream cipher*, the input text is processed continuously to produce the output element as one single element.

2) Key distribution method: in this respect, we have symmetric and asymmetric cryptosystems. In an *symmetric cryptosystem* (denominated also as single key and secret key), both sides of the communication essentially use a single common key for encrypting and decrypting the messages. Data Encryption Standard (DES) [38] is the most widely used symmetric algorithm known. In a *asymmetric cryptosystem* (denominated also as public key and two-key), encryption and decryption is done using a key pair consisting of a public key and a private key. The public key can be openly distributed, while the private key is confidential to its owner. Data encrypted using the public key can only be decrypted using the private key. RSA [39], Diffie-Hellman [40], DSS [41], and Elliptic Curve Cryptography (ECC) [42], are asymmetric encryption algorithms. RSA is the first public key algorithm introduced by Rivest, Shamir, and Adleman in 1977. Although RSA is commonly accepted and widely used for encryption and digital signature, in order to have a secure RSA algorithm a longer bit size is required, which introduces larger overhead. For this reason Elliptic Curve Cryptography, might be an alternative to RSA that provides equal security while using smaller bit length; however, ECC has not been shown to be

as secure as RSA. Diffie-Hellman is a key-exchange protocol in which two parties are attempting to agree on a shared secret. *Digital Signature Standard (DSS)* was developed in 1991 by NIST and is used to provide digital signature. [33]

3.2 Authorization and Access Control

Authorization is allotting permits to the network entities that have requested to access the network resources.

Access Control pertains to inhibition of unauthorized access to the network and the resources on the network, and guarantees the secrecy, availability and integrity of the system [33]. The following are some paradigms of existing access control methods which could be exploited in smart spaces:

1) Access Control List (ACL) ACL, as is indicated from its name, is basically a list attached to the objects of the system. Object is any entity in the network to which the access should be controlled). The stored lists, contain the names of subjects (any network entity that attempts to access the network resources) associated with the names of the objects, they are allowed to access. When a request for accessing an object is received, access manager tries to find the subject's name on the object's ACL. If a match from the requester to the specific resource is found, access is granted and otherwise it is denied.

This method requires supplementary memory in comparison to other access controlling methods in order to store the ACLs. Moreover, there is a need of continuous maintenance, since for a slight change, several ACLs need to be modified. An alternative could be a central access control list for all resources. Nonetheless, the central ACL is considered to be inconvenient to the authorizer because of the need for specifying access rights individually to each user. [43]

2) Role-based Access Control (RBAC)

RBAC model, proposed by Sandhu [44] in 1996, relies on a central administration, i.e., the administrator defines roles according to the jobs and positions existing in the organization and grants essential permissions properly to the roles allowing them to perform their tasks. Thence, users are assigned role(s) due to their responsibilities and qualifications. In this manner, users acquire permissions and are authorized to do actions. RBAC is considered suitable to satisfy the Separation of Duty requirement which supports the security. The term, Separation of Duty, in computer security refers to the principles that provide control for the cases that singles task have to be performed by several people. The goal of these principles is distributing the tasks and responsibilities between the people to prevent possible frauds. RBAC is considered as a beneficial solution for spreading the responsibilities to the individuals through their roles in the system [45].

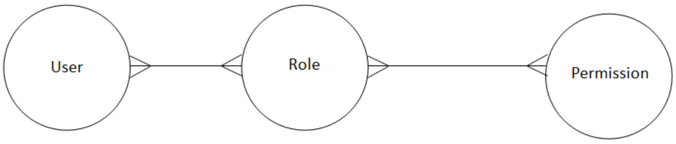


Figure 3.3: Role Based Access Control (RBAC) [44]: users are assigned some role(s) according to their responsibilities in the system. Access rights are granted to the roles, and consequently, users get permissions for performing a task.

Simply, permissions could be revoked from a role and new ones be granted, any time needed. In other words, privileges of the users could be updated merely by modifying the rights of the corresponding role. This feature is one of the advantages of RBAC from administrative point of view.

As it is seen in figure 3.3, User and Role have many-to-many relation; meaning that a user may have several roles and a role might be assigned to more than one user. For instance, a user can have the role of Patient in a health care system, and also the role of a Doctor medicating patients; and on the other hand, the role of Doctor/Patient can be assigned to more than one user. Certainly, the user cannot activate both roles at the same time. Similarly, there is a many-to-many relation between role and permission, which points out that a role is entitled to do different actions, and permission of performing of an action may be given to more than one user. To cite an instance, a user with the role of Doctor has the permission to update the medical history of a patient along with reading it. On the other hand, the permission to read the medical history of a user could be allocated to multiple roles, such as Nurse and Doctor.

As the access control used in the thesis is based on RBAC, here we explain the terms used in RBAC:

User: A human being interacting with the system [46]

. *Role:* A job or position within an organization. Each role has specific authorities and restrictions for accessing the resources [44].

Administrator: A user in charge of establishing security policies, defining and modifying users, roles and permissions [46].

Subject: Active entities, such as user, role and device trying to access resources or causing the data flow [46].

Object: Passive entities, such as resources and information [46].

Access Control: Restricting the access to the resources of the system only to the legitimate parties [47].

Authorization: granting access to a user or program.

Permission: A rightful access of the subject to an object in the system [46].

Role Hierarchies (RH) is the way of organizing roles to express the authorities and responsibilities within a system [44]. RH is beneficial for designing generic access rules as an alternative to applying the rules to each role, because with a hierarchical structure, roles inherit permissions of their predecessor [48]. Assume role x $\geq$ role y, then members of x are implicitly members of y and x inherits permissions of y. [49] In this example, x is a senior role and y is a junior role. Senior roles have more power than junior roles, for instance, a Dermatologist has some other rights in addition to the rights inherited from Doctor's rights. As shown in figure 3.4, senior roles come on top of the diagram, while junior roles, as they have less power, come towards the bottom. [44]

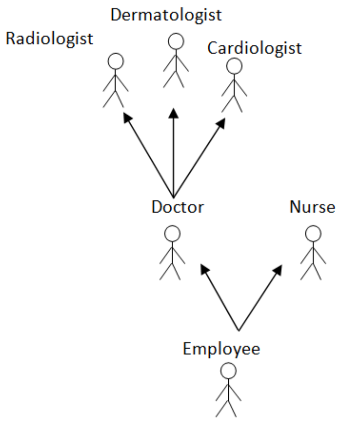


Figure 3.4: Role Hierarchy (RH). RH is a way of organizing roles, in which they inherit permissions of their predecessor. The roles coming on top of the diagram are senior roles and have more power than the junior roles which come toward the bottom of the hierarchy diagram. In this example Radiologist is a senior role that has additional rights to the junior role, Employee.

Although RBAC has some advantages over ACL such as mapping individuals to the access control list, it is criticized because of being insufficient for dynamic environments [48]. The insufficiency refers to its static design and its disability to respond to the changes in the environment, i.e., the context information about the environment does not affect the way roles are allocated, or on decision makings.

3) Attribute-based Access Control (ABAC)

In ABAC, instead of applying permissions directly from subjects to objects, authorization is based on their attributes (figure 3.5). The attributes are either static, e.g., name and role of the subject, or dynamic, e.g., age and location. [50]

As an example, in a company access could be granted to the employees in working hours and denied after that. In this example, the ”working hours” is the attribute. ABAC rules can simply be designed and quickly adapted to changes; however, at run-time, a large number of rules must be executed [48]. ABAC is flexible and thus more suitable for open, dynamic, sharable environments. In such environments there exists a large number of

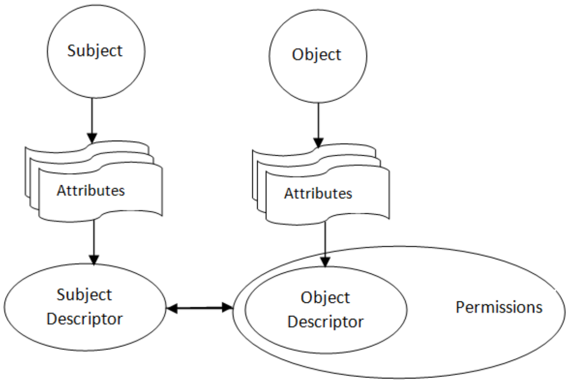


Figure 3.5: Attribute-based Access Control, quotation from [51]: authorization is based on the attributes that the subject and the object have.

users of whom some are not known in advance, and also roles are not statically allocated in the beginning [50]; however, the flexibility of ABAC makes the policy management and authorization complicated and error prone [51].

4) Context-aware Access Control

Context refers to the information describing the current situation of an entity (e.g., a person, device and an application interacting with the user). The notion of context aware computing was stated by Schilit et al., for the first time in 1994 [52]. A. Kashevnik et al. in [26], categorize the participant's context in a smart space into:

- Physical components (time, geographical location and device type)
- Virtual components (the software, public key and digital signature used by user to access the smart space)
- Social components (position of the user in the organization, relationships that a user has such as family and friends with whom different level of information is shared).

This information is gathered and saved in smart space devices and used later for access control decision making and for enabling granting of access permissions.

By using the context information of a participant in a smart space, intelligent applications can perform actions and adapt themselves towards user's intentions and preferences without involving any human interaction [53]. For instance, based on the information collected from the user's calendar, GPS, Facebook profile, and a restaurant's website the smart phone can figure out that the user is in his free time and is located near his/her favorite restaurant and can inform the user of the special meal of the day.

According to [54], the context information in a health care system is categorized in three different groups: a) environmental information (time and coordinates), b) personal information (age and medical history of the patient- that is a record of past treatments, present illnesses and family history that may be relevant to the present state of health of the patient [55]), and c) medical data (blood pressure, heart rate and stress level).

Context awareness is being sensitive to the captured information by sensors and devices of the space. A context-aware application is ”an application that automatically adapts to discovered context, by changing its behavior” [56]. On that account, due to the fact that sensors sense and report the context information (and sometimes sensitive information, such as identity of the users), they are considered to play an important role in smart spaces and hence, susceptible to attacks [57].

Three types of threats for context information are cited in [57]: a) Attack over context provider (when an untrusted sensor is added to the space or sensors of the space are replaced by fake ones) b) Intercept the communication between context provider and receiver in order to alter it or benefit from it. c) Attack on context receiver (when the context information is received by an unauthorized application). Therefore, for better protection, it is necessary that the context providers (sensors) as well as context receivers (applications) authenticate themselves to the system in order to be able to communicate.

3.3 Communication Security

In order to guarantee communication security in the network, a wide range of security mechanisms is exploited. Cryptographic techniques are types of these mechanisms that are being utilized to establish a secure channel and protect data from unauthorized access. As listed in [58], the goals that a secure communication channel tries to achieve are:

- **Authentication:** Receiver should be sure that the message is coming from a legitimate sender.
- **Authorization:** Valid users should be able to gain access only to the resources allowed for them.
- **Accountability:** Users should be unambiguously identified.
- **Integrity:** Receiver should be sure that the message is not modified during the transmission.
- **Confidentiality:** The message should be encrypted before being transmitted, so that only the intended entity is able to decrypt and read it.
- **Reliability:** The service provider (host) should be resistant to attacks which lead to hampering its quality or availability.
- **Non-repudiation:** Due to the fact that all communication parties should be confident about the security of the transaction, it must be ensured that none of the parties denies the participation in the communication. To preserve this mutual trust, the parties may exert digital signatures to verify the authenticity of the other party.
- **Privacy:** Identity of the entities or users in a network should be protected from being revealed.
- **Consistency:** The sides of a communication should maintain a consistent view of each other.

In a smart space, there might be multiple SIBs serving the KPs. In order to have data securely exchanged, there must be a trust relation between communication parties. Ergo, before starting the communication, SIB and KP authenticate themselves to each other. Additionally, to ensure the integrity of the information during the transmission, it should be encrypted in a way that only the intended receiver can gains access. There exist several security mechanisms to guarantee the communication security over the networks that SS can take advantage of. KPs are free to choose any security mechanism, as long as they are sufficiently secure and supported by the SIB [22].

Security in RIBS (RDF Information Base Solution) is taken care of by the *TLS* (Transport Layer Security) protocol and the *X.509* certificate. The first time a KP tries to join the space, it is given credentials issued by a trusted third party, Certificate Authority (CA). The credential, which could be an X.509 certificate or a username-password combination, contains pairing and policy information. The KP utilizes the certificate to authenticate itself to the RIBS and also to open a communication session. TLS protocol is used below SSAP, and it addresses inter-device security (security between the RIBS and KPs). [22]

TLS is a protocol that secures communication over the Internet, and prevents security breaches (e.g., eavesdropping and message falsification). Its purpose is to establish a secure interoperable connection between parties in an efficient way. In this protocol, before starting the transmission, the client and the server authenticate themselves and negotiate to each other through a handshake protocol. Negotiation is done to agree on the cryptographic keys and the encryption algorithm they are going to use to send data. [59]

To make sure that a malicious party cannot forge the public key and pretend to be the key owner, a public key certificate is used. The public key certificate is issued and signed by a trusted CA. The certificate includes the public key and the user ID of the key owner, with the whole block signed by the CA (encryption of the other fields with the private key of the CA). Users securely provide their public key (for example in a hashed format) to the CA to gain a certificate. Thence, users make their certificates public, to be used by others to communicate with the certificate holder. The X.509 standard defines a widely accepted scheme for formatting public key certificate used in many network security applications.

[33]

Securing the network communication in smart spaces is rather challenging comparing to other networks; since the devices participating in the space mainly have low capacity in memory, battery, CPU capability, and computational power. Due to these limitations, most of the devices in a smart space are not able to support high-scale security deployments. [60]

Host Identity Protocol (HIP)

Internet Protocol (IP) addresses are currently being used to both identify and locate hosts. Nevertheless, the IP address of a portable host changes every time it shifts from a network to another. To maintain communication, the host is required to re-identify itself to the peers it was communicating with, and to confirm that this is the same host. Thus, it is difficult to securely implement mobility and multihoming over the existing Internet. Moreover, IP has some weak points, such as a) being susceptible to DoS and impersonation attacks (IP addresses are simply counterfeited), b) IP security (IPsec) is not easy for the users to configure. Therefore, the transmissions of data in the network layer happens mostly in plaintext, if IPsec is not used, and c) there still are some challenges in interoperating the IPv4 applications working with new IPv6 applications. These shortcomings motivated the Internet Engineering Task Force (IETF) community to consider new approaches. HIP (Host Identity Protocol) was firstly proposed by Moskowitz in 1999, and continued being developed by IETF later on. The advantages of HIP over IP are: mobility and multihoming, security, IPv4 and IPv6 interoperability, and identifier/locator split. In IP, hosts are identified and also located via IP addresses. In HIP, identification and locating the hosts are seperated from each other. Hosts are similar to IP identified by IP addresses, but located using Host Identity (HI). The HI is a pair of public and private keys, used to identify the host to other parties. The identifier/locator split characterisitc of HIP allows the mobile hosts have constant HI (preserves its identity) even if the IP address (the location) changes upon moving. [61]

I. Nikolaevskiy et al. are proposing to use HIP [62] for Smart-M3 platform [60]. This protocol extends the current Internet architecture with an additional layer between the network layer and the transport layer (figure 3.6.b). In the new architecture, identification is done through cryptographic identifiers (a pair of public and private keys). [63] HIP depends on HIP Base Exchange (BEX) to exchange shared keys. However, due to the fact that HIP BEX is not efficient for these environments, a lighter version of it is used. It is called HIP Diet Exchange (DEX), and it consumes lower computational resources [64]. HIP DEX is based on Elliptic Curve Cryptography (ECC) to share secrets between the two parties (SIB and KP) in communication. Both sides authenticate themselves to each other; KP is the initiator, commencing the HIP DEX, and the SIB is the responder [60].

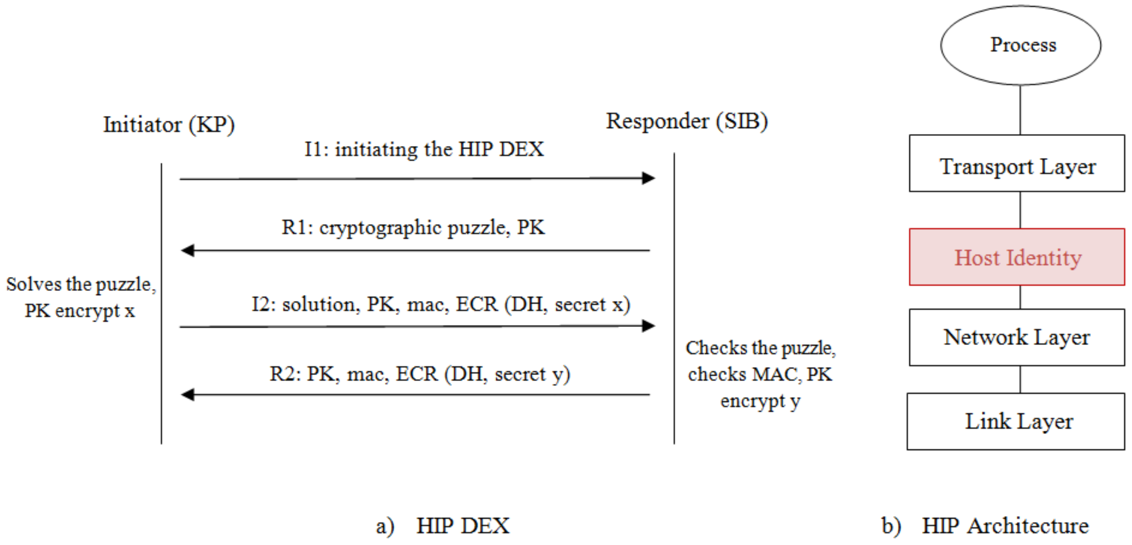

Initiator (KP)
Responder (SIB)
I1: initiating the HIP DEX
R1: cryptographic puzzle, PK
Solves the puzzle, PK encrypt x
I2: solution, PK, mac, ECR (DH, secret x)
R2: PK, mac, ECR (DH, secret y)
Checks the puzzle, checks MAC, PK encrypt y
Process
Transport Layer
Host Identity
Network Layer
Link Layer
a) HIP DEX
b) HIP Architecture

Figure 3.6: a) HIP DEX Scheme: four packets of the size of 530 bytes are exchanged for initial authentication. In case of successful authentication, the actual data flow starts over. b) HIP Architecture: the current Internet architecture is extended by adding an additional layer between between the network layer and the transport layer

Figure 3.6.a) depicts initial packet exchange between Initiator (I) and Responder (R) for authentication. In the whole process, 4 packets with the size of 530 bytes are traded [64]. The actual data flow happens after these four packets in case of successful authentication. In the second and third packets (R1 and I2 packets), Diffie-Hellman (DH) keys are exchanged for encrypting the initiator's HI. The signature of the responder in R1 assures the initiator that the keys are created by a rightful responder and also prevents downgrade attacks (attacks that occur by altering the I/R packets, to change the result). In the third and fourth packets, the authentication of the two entities happens. [65]

Furthermore, when the initiator incepts the authentication process, it receives a cryptographic puzzle from the responder. The idea of using this cryptographic puzzle is to protect the responder from DoS attacks (the puzzle could be adjusted to control the performance level [60]). While the initiator is solving the puzzle, the responder goes to stateless mode, until it gets the I2 packet. Then the solved packet is validated by the responder to check whether the initiator is sincere. [65]

3.4 Privacy

The terms "Security" and "Privacy" though may be used interchangeably, have different meanings [66]. As defined in Merriam Webster Online Dictionary, security is:

> "The quality or state of being secure: as a) freedom from danger: safety, b) freedom from fear and anxiety. And security protection are the measures taken to guard against espionage or sabotage, crime, attack, or escape."

And privacy is:

> "The quality or state of being apart from company or observation: Seclusion, freedom from unauthorized intrusion."

In any information system, both privacy and security have to be satisfied in order to achieve the goal of protecting the information. However, collection of personal information has always raised individuals' concerns on privacy loss, about the confidentiality and security of the database where the personal data are stored, the proportionality of the data gathered (if the data is really used in relevance to the purpose it was collected), to whom the data would be handed over, the accuracy of the system (False Acceptance in case of similarity, False Rejection in case of having unreadable data), if the data is used with the consent of the owner, and if the data is protected from the unauthorized access. Some people believe that, ”if you have nothing to hide, you have nothing to fear”; however, others are on the opinion that, ”just because you value your own privacy, it does not mean that it is about hiding criminality.”

On this basis, governmental organizations have issued regulations to support an individual’s privacy by protecting the privacy of their personal information. These regulations *notify* the user about the gathering of their information and how it is going to be used, giving the users the right to express their *consent*, or withdrawal of it, on revealing their sensitive information, giving them the right to *access* their own information and modify it, authorize them to *enforce action* if their information is not used legally, certify the users that the use of their information will be *consistent* with what they agreed on, and ensure the individuals about the *security* (confidentiality and integrity) of the collected and stored information. [33]

Privacy in a computer system is split into four main domains, including *Anonymity, Pseudonymity, Unlinkability, and Unobservability*. *Anonymity* helps the user to interact with the system without being identified and discovered (without his identity disclosed) by other users. In *Pseudonymity* users are given an alias, so that the system identifies them with this alias, and their real identity is not divulged to other users. Users may utilize different resources. *Unlinkability* ensures that the user’s identity is not disclosed, by linking the data on different sources together. *Unobservability* allows users to use resources and benefit from the services, assuring that the resource/service is not observed by intruders. [33]

With the maturity of technology, the ubiquitous computing era is evolving and becoming more widespread. Ubiquitous computing refers to computers joining together, obtaining and processing data, in order to provide required services to users without their interference [67]. Ubiquitous environments and smart spaces, despite of offering comfort to the users, raise various concerns about potential violations of privacy. A breach in privacy may happen because of the software (e.g., trojan code in a software application) or hardware used in a SS, originating from an untrusted party [68].

In [68], there are several privacy threats mentioned for Personal Smart Spaces (PSS).

- Collecting the personal data without the user's consent and notice.
- Disproportional data collection: only the type of data from users that is related to the specific purpose must be collected.
- Illegitimate access to the personal information of the users.
- Identity theft (using identity of a person for an unlawful activity without his permission)

Nowadays most of new devices have GPS support, and this makes Location Based Services (LBS) possible and readily available to most users. LBSs are the services that rely on physical position of the user; however, they release some private information (location) to provide services, which might not be pleasant for users considering their privacy. Moreover, users of these devices might be put in danger of being possibly tracked [69]. Smart spaces may utilize LBSs to collect locational information of the users. The collected information, together with other context information, is used for reasoning and intelligent decision making. Therefore, SS is prone to the security threats known for the LBSs. To overcome these security related issues, there are several approaches proposed for privacy protection such as access control and k-anonymity (section 3.4.1).

Access control determines who can gain access to particular information, under specific conditions. Privacy could be protected using access control policies; however, it is not feasible for the designer to foresee all the probable undesired data disclosures at design time to prevent them in policies. A solution for the unwanted data leakage is strict policies, which again is not appropriate for smart space, because some of the components and services of the system might become disabled. [68]

As mentioned before, one of the aspects of privacy protection is keeping users anonymous in the system. Health-care information systems are one of the information systems where anonymity has great significance, not only to preserve the privacy, but also to prevent abuse of the personal information. The following section overviews the concept of anonymity and existing approaches to preserve anonymity of the users in the system.

3.4.1 Anonymity

Anonymity and privacy protection go hand in hand, i.e., actions taken to help end users stay anonymous in the system and also to protect their privacy. Basically, users provide their information to service providers aiming to receive services and still not expose their identity. Several approaches are proposed in this field to preserve anonymity:

K-anonymity: There has always been attempts to secure sharing of the privately held information, in any information system, in a way that the identities of the people is protected. Authentication of the users and controlling the access only prevents the direct disclosure of the stored information on a database, while there are always possibilities of data leakage that occurs if an adversary performs an inference attack. Linking together information the user has shared in different sources is an example of this issue: the attacker may be able to uniquely identify some individuals. [70]

L. Sweeney [70] has shown that in USA, by linking the medical information (including date of birth, nationality, gender and zip code) together with the information about the voters gained from the registration list, (including date of birth, gender and zip code), some of the individuals are identified. As an example, there are six people on the voting list that have the care data available with the same birth date. Three of them are men and only one of them has a 5-digit zip code. This information has helped attackers to identify an individual uniquely (see figure 3.7).

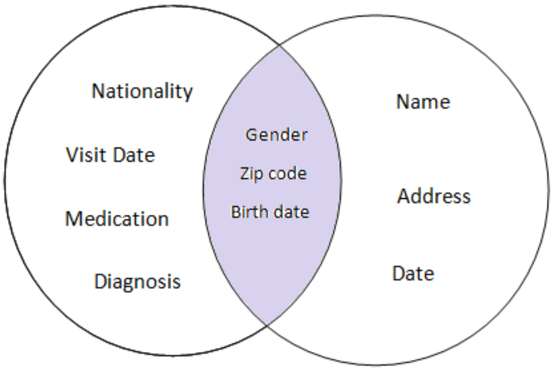
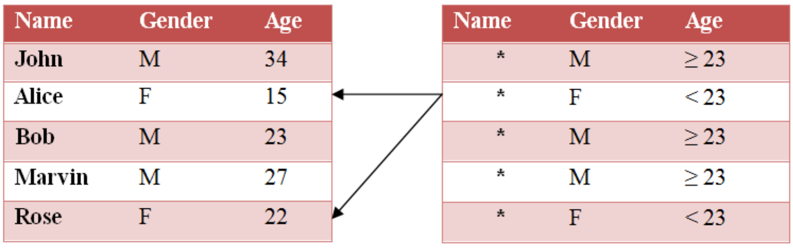


Figure 3.7: Linking the data from different sources. In this example, in the left circle there are some medical information gathered from the users, in the right circle there are the voters' information gained from the registration list in an election, and the dark area in the middle highlights the common information between the two resources. By linking together the information from these different sources, some of the individuals are uniquely identified.

K-anonymity model [70] protects user's identity by generalizing quasi-identifiers (QIs), which are the attributes that with linking to other sources, could be used to identify a user [71]. By generalizing, it would not be easy to find a specific match. In figure 3.8, on the left there is the raw data set, and on the right is the k-anonimyzed data set (k=2). The generalized record could be related to k individuals. The data is generalized into two age groups, people with the age lower than 23, and people with the age of 23 or greater. Thence, in this example, we will have two female individuals below the age of 23 and it is not possible to identify to whom each record belongs. [70] Nevertheless this method has several limitations: a) It reduces the precision and consequently the performance. b) By submitting several queries to the smart space attack is possible. c) Applying K-anonymity in smart spaces is complicated because of generation of various data over time. [71]

C. Patrikakis et. al state that k-anonymity techniques are incomplete for pervasive computing systems [72]; since by monitoring user's behavior and the responded services, it is possible to identify the user that the service is intended for. A pervasive gym il-



Name	Gender	Age
John	M	34
Alice	F	15
Bob	M	23
Marvin	M	27
Rose	F	22

Name	Gender	Age
*	M	≥ 23
*	F	< 23
*	M	≥ 23
*	M	≥ 23
*	F	< 23

Figure 3.8: K-Anonymization: on the left table there is a raw data set, and on the right there is the k-anonimyzed model of it (k=2). People are categorized into two different groups based on their age: a) people younger than 23, b) people with the age of 23 or older. With this generalization, we have two records of women under age of 23 and it is not possible to identify them uniquely.

lustrates a simple example of this issue. There are k anonymous participants getting services, e.g., suggested exercises, by providing their physical condition. By observing users' activities and recommended exercises for a period of time, a single user who starts exercising with the suggested machine is recognizable.

Identity Generalization: In a request from a user, some attributes uniquely point out an exact user. This data should be removed from the request. This could be done by substituting the unique identifier with a pseudo-id; however, it would be impossible to customize services for the users; because there would be no reference to their information. L. Pareschi et al. generalize the user identity using stereotypes for authentication and session management, while the user stays anonymous [73]. The disadvantage of generalization is that it affects the quality of the services, i.e. to acquire the appropriate level of anonymity, the context data becomes too generalized to be served at a proper quality level.

Using Privacy Policies: To avoid the aforementioned drawback for the generalization method, anonymity can be supported in the privacy policies that users define to generalize a context [73].

Policy 1: If *activity==shopping* Then *anonymity-level:=low*

Policy 2: If *activity==walking* Then *accurate-location*

Policy 3: If *activity==working* Then *anonymity-level:=high*

In policy 1 and 2, the level of anonymity is set to low, and the accurate information about the user is provided.

3.5 Vulnerabilities and Possible Attacks

The weaknesses and vulnerabilities in a system make it prone to security threats, and consequently, security attacks. A resource or a component in a system is vulnerable, if it is: *corrupted* (on account of some unintended changes, the resources do not function as they should), *leaky* (improper access to the resources), or *unavailable* (the system is disable or too slow to render services). By taking advantage of the vulnerabilities in a network, attackers could violate the system and cause security harms to the assets. [33]

In a computer system there exists a wide variety of security threats that could cause malfunctioning of the system, unavailability of resources and unintended disclosure of information to unauthorized entities. W. Stallings et al. [33] pointed out some possible threats and the consequences of each threat (see table 3.1). As smart space is considered a networked system, it could be receptive to the same threats and attacks.

Table 3.1: Security threats and consequences.

Security Threat	Consequence
Exposure: direct revealing of information to an illegitimate user; for instance, system errors or intentional data disclosure by an insider.	**Unauthorized Disclosure:** An entity obtaining access to the data which it does not have the right.

Continued on the next page

Table 3.1 – continued from the previous page

Security Threat	Consequence
Interception: adversary gaining access to the data while transmitting between the sender and receiver. **Inference:** indirect access of unauthorized party to the data, by reasoning of information collected from communication, e.g., traffic analysis. **Intrusion:** gaining access by bypassing the security safeguards, e.g., breaking the access control system.	
Falsification: modifying or replacing the data to deceive the authorized receiver. **Masquerade:** an adversary pretends to be an authentic user. **Repudiation:** a user does not take the responsibility of an action it has done, e.g., denying the sending or receiving of a piece of data.	**Deception:** Authentic party receives the false data without being aware.
Misappropriation: unauthorized consumption of the system resources, e.g., Denial of Service attack. **Misuse:** making the system take destructive actions.	**Usurpation:** an attacker taking control of the system.

Continued on the next page

Table 3.1 – continued from the previous page

Security Threat	Consequence
Corruption: (attack on integrity of the system), malicious behavior that causes the system to operate in an unintended way. **Obstruction:** hampering the system from providing services, for instance by disrupting the communication or overloading the system. **Incapacitation:** (attack on integrity of the system), debilitating system components, in order to stop it from functioning, e.g., physical damages and viruses.	**Disruption:** creating a disturbance that causes the system to stop functioning, or making the system malfunction.

A threat that has been accomplished is termed an attack [33]. A successful attack jeopardizes the security and privacy of the information on the network. Attacks are categorized into active and passive types. In a passive attack, the intruder utilizes the information it obtained; but does not make any alteration, which makes it more difficult to be detected. On the other hand, in an active attack, the attacker has more malicious purposes, such as modifying the intercepted message, affecting network resources and system functionality. [33]

Common Attacks on Networked Systems

Denial of Service Attacks (DoS): In this type of attacks the attacker's aim is to hamper the users of a service to access the network and its resources, by targeting the service provider. The most common form of DoS attack is *flooding* the network with useless traffic. Every sent request requires bandwidth, memory, CPU and other network resources. In a DoS attack, the malicious requests occupy the network resources, so that they become unavailable and are not able to respond to the legitimate requests. Moreover, a DoS attack may target the physical components of the network, trying to wreck them or attempting to change the network configuration. Since DoS attacks have been observed to be highly destructive, precautionary measures (such as using anti-virus software and firewalls), could lessen the probability of attacks to occur. [74] As pointed out in [64], the HIP protocol exploits a cryptographic puzzle to protect the system from DoS attacks.

Eavesdropping Attack: An unauthorized entity intercepts the communication and passively listens to the communication [74].

Man-in-the-middle Attack: As it is indicated by the name of the attack, it pertains to an active intruder between the two communicating parties. The attacker not only captures the information but also controls the communication. For instance, it might alter the messages, without letting the users know. [74]

3.6 State-of-the-Art On Security and Privacy in Smart Spaces

In the previous section, the concepts related to security and privacy of networked systems including smart spaces, were studied. In this section, a review of various existing approaches proposed to enhance security in smart spaces is presented.

Cerberus is a context-aware security scheme introduced by J. Al Muhtadi et al. [31], for the Gaia project. It should be mentioned here that Cerberus is different from Kerberos [75]. Cerberus is a security framework proposed for an SS, while Kerberos is a computer network authentication protocol. In the Cerberus framework, principals (called also agents in the article) use different authentication devices and methods to identify themselves to the system. Each of these devices has a level of confidence (trust level) based on how reliable and secure they are. In addition, for performing a particular action, at least a specific minimum confidence value is required. Access rules decide whether a requester is allowed to access a certain resource. This model is considered to be flexible and appropriate for smart spaces and dynamic systems. Though, in this approach, anyone satisfying the trust level necessary for highly confidential data, can succeed to gain access. However, in the project discussed in this thesis it may be necessary to deal with personal and medical information, so the Cerberus approach is not appropriate for the project. More restricted access control is demanded to specify who is allowed to access the sensitive data.

A. Keshavenik et al. [26] proposed a context-aware access control for Smart-M3. Context refers to the information that describes the situation of an element. This proposal is a combination of RBAC and ABAC, meaning that roles are assigned to the individuals dynamically, according to their context. Context information of the participants including physical, virtual and social components is used to define three sets of rules. The first set assigns a numeric value representing the level of trust. This trust level is derived from the user's context, e.g., identification attributes, location, time. The second set is used at authentication time. In compliance with the trust values, the roles are allocated to the users (ABAC). The last set determines access control policies expressing if a user having a particular role is permitted to access a specific resource (RBAC).

K. Yudenok et al. [64] present the idea of mapping the RDF graph into Virtual File System (VFS), for access controlling means. The triples are sorted in a tree structure. The operations in RDF are presumed equivalent to operations in VFS (e.g., query command resembles the 'read' right).

Consec [57] is another context-aware security framework developed for smart spaces. S. Al-Rabiaah et al. claim that due to the fact that sensors play an important role in smart spaces to generate context, and furthermore, context information may carry sensitive data about users, e.g., GPS location information, there is an essential need to ensure the legitimacy of both the context provider (CP) (sensor) and the context receiver (CR) (application). To achieve this goal, they have designed a security framework, in which it is assured that the information is obtained from a trusted CP, expended by a trusted CR, and secured while being delivered, meaning that all sensors and applications, in order to be able to communicate with the system, must authenticate themselves. Authentication is based on Kerberos Protocol [75]. In this protocol, the nodes in the network authenticate themselves to each other based on tickets. The integrity of the communication is guaranteed using hash functions and digital signature, and the confidentiality of the communication is taken care of via symmetric key cryptography.

We model our access control scheme using ontological modeling techniques because they are successful in representing knowledge and useful for data reasoning and inferencing. For this purpose we have studied the existing access control ontologies proposed for context-aware, dynamic and distributed systems. By comparing them, we concluded that none of these ontologies tackles all the requirements of our system. Thus, in chapter 4 we will propose an access control ontology. Table 3.2 is a summary of some of the existing access control ontologies and the proposed ontology in this thesis. In the table different approaches are compared with respect to the criteria including, a) the access control model they represent, b) context-awareness, c) their capability to support the rules, d) the domains they are proposed for, e) their capability to control the privacy of the users, and f) their capability to control the access at triple level. According to the table, we can claim that our ontology is unique by being context-aware, rule-based, proposed for smart spaces and being able to control the users' privacy and access at triple level. The other factor that differentiates our work from others is the capability of implementing alerts using the publish/subscribe mechanism supported by Smart-M3 (e.g., if a critical situation in the patient's health occurs, and if an access to the data that user has labeled as highly sensitive occurs).

Table 3.2: Access control ontologies

Ref.	**Access Control Model**	**Context Aware**	**Rule-based**	**Domain**	**Privacy Control**	**Triple level Control**
[76]	Context-based	✓	✓	Pervasive Computing Environments	×	×
[77]	Privacy-centric	×	✓	Heterogeneous administrative medical domains	✓	✓
CoBrA [78]	No access control ontologies	✓	✓	Context-aware systems and SS	✓	×
OWL-S [79]	Ontology-based	×	✓	Semantic Web services	✓	×
OPO [80]	Access Control List (ACL)	×	✓	Linked Data	✓	✓
[81]	User Behavior and Capability Based Control Access	✓	✓	SS	✓	×
[82]	Credential-based	✓	✓	XACML and SAML[1]-based systems.	✓	×

[1] XACML: eXtensible Access Control Markup Language, provides policy based language that provides

Ref.	**Access Control Model**	**Context Aware**	**Rule-based**	**Domain**	**Privacy Control**	**Triple level Control**
SitBAC [83]	SitBAC	✓	✓	SS	✓	×
Proteus [84]	Context-centric	✓	✓	Pervasive Environments	×	×
This work	CARBAC	✓	✓	SS used in health and well-being	✓	✓

In [76] any resource has an owner. The owner specifies a set of context conditions (e.g., time, location, entity's activity). If the requester satisfies those conditions, then it is permitted to access the resource.

In [77] patients consent to the use of their information, and in accordance to the patients' consents (preferences), the access control rules are defined.

CoBrA [78] is a set of ontologies that build a context-aware system together. Privacy protection ontology is one of the ontologies in CoBrA. The privacy of the users is protected via the rules defined by them.

In [79] various ontologies are designed to sketch different security related aspects (e.g., the ontologies that represent credentials, security mechanisms, privacy). Access to the resources is controlled via pre-defined privacy policies. Any of the web services has its own privacy policies attached to it that specify allow and deny access right.

OPO [80] is a lightweight vocabulary generated according to the users' privacy preferences to control access over RDF data (any user has its own Access Control Lists (ACL)

access controlling. It is implemented in XML and primarily proposed for Attribute Based Access Control (ABAC) systems [82]. SAML: Security Assertion Markup Language is an standard for exchanging the data used for authentication and authorization means based on XML. [82]

that defines the access privileges to the user's data). A requester, in order to gain access to an object, is required to satisfy a set of attributes that specify who is granted access.

In [81] the proposed access control model is based on the users' behavior and capabilities, context information and historical data (i.e., a pattern of past actions the user has taken) and aims at assisting the dependent people. The behavior of the users is tracked (for instance by a 3D sensor) to uniquely identify the users. In this way users are identified more discretely, which makes it possible to provide more customizable services to them with a higher security.

In [82] all users are issued a set of credentials by a central issuer. When the user requests to access a resource, it receives a set of policies. The received policies specify the requirements that the user must satisfy in order to be granted the access. In this model the privacy is supported via anonymous credentials (allowing users to fulfill the required conditions for accessing a resource without revealing their identity).

SitBAC knowledge framework [83] is based on Situation Based Access Control. The access control is proposed to control the access over Electronic Health Records (EHRs). In this model the access is controlled according to circumstances that match a pre-defined pattern. The access control policies are specified in accordance with the possible situations. Privacy in this system is protected via access control scheme.

Proteus [84] is a context-centric policy model based on semantic technologies (ontologies and rules). In this approach instead of associating the access control rules with the subjects, they are associated with the context, i.e., context is considered as the central element for specifying the rules. Context refers to the information that describes the state of an entity or an activity. The rules define the possible actions on the resources. An entity is only able to perform a requested action, if the current context of the environment matches the required context of the requester.

Our access control model is a Context Aware Role Based Access Control (CARBAC) scheme. In this model, roles are assigned to the users by a central administrator when they register in the system. At run-time, the security rules are executed according to the requester's role and context information, to grant/deny access. Privacy is protected via privacy rules. The modeled ontology aggregates all the aspects that were missing in the existing access control ontologies in the literature, as a way to handle the security and privacy issued related to the smart spaces, all together at once.

Chapter 4

A Semantic Security Framework for Smart Spaces

4.1 Requirements for a SS Security Framework

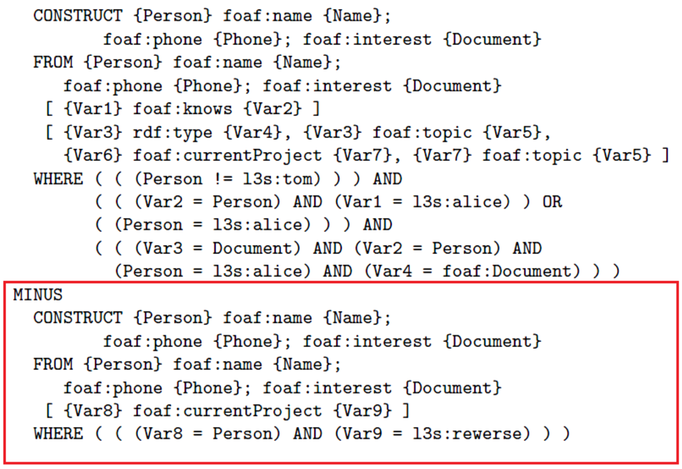
Development of the Internet technologies has made it possible for many industries to yield their business and services over the Internet, and so does the health care industry. Basically, in health care, service providers (e.g., hospitals, laboratories, medical institutes, and insurance companies), have their own Electronic Medical Records (EMR) and databases, and a patient may be a client to multiple service providers [85]. Certainly, an integrated health care system can serve clients more efficiently [86]; however, information sharing always raises security risks and privacy concerns.

This thesis aims at improving the security of smart spaces, with the main focus on health care and well being. Thus it is highly significant to protect the privacy and confidentiality of patients' medical and personal data from unauthorized access while in the repository and while being transported. It is even more crucial and difficult to administrate the information and physical security in ubiquitous environments with numbers of participants continuously joining and leaving the space. Hereupon, we are proposing a security framework that takes the following security objectives into account, which will be discussed in detail in this section: Authentication, Authorization and Access Control, Communication Security, Privacy, Data Provenance, Anonymity.

4.2 Specification of the Security Framework

Currently, most of the security proposals for smart spaces are limited and coarse-grained. They control the access to the whole data repository, e.g., in the existing RDF repositories (such as Sesame [87] and Jena [88]). However, we need to have fine-grained access control at triple level. Although a fine-grained access control comes with costs and overhead (more computation and higher response time), it is considered to be acceptable when dealing with highly sensitive information. There are two ways to authorize a request: post-filtering and pre-filtering. Post-filtering is retrieving the requested triples and filtering out the returned result afterwards, which is not an efficient solution. Since it requires memory and computational power, it increases the response time to return some unwanted triples. On the contrary, in pre-filtering, the incoming query could be expanded in a way that only allowed triples are queried and returned. This could be done by removing disallowed triples from the allowed ones in the query using the MINUS operand [89] (see figure 4.1).

The proposed security framework (see figure 4.2) comes on top of an RDF repository to check the authenticity and authority of the requester, along with managing its access before getting into the data repository. By evaluating the available access control rules for the corresponding requester, the request is modified and constrained by the matching privileges. Thence, the modified request is forwarded to the repository to query for the intended resources. As a result, the output is only the RDF triples that are accessible for the requester [89]. This framework is inspired from ideas proposed in different works to gain the best benefit out of them. The framework works as follows:



```
  CONSTRUCT {Person} foaf:name {Name};
         foaf:phone {Phone}; foaf:interest {Document}
  FROM {Person} foaf:name {Name};
     foaf:phone {Phone}; foaf:interest {Document}
   [ {Var1} foaf:knows {Var2} ]
   [ {Var3} rdf:type {Var4}, {Var3} foaf:topic {Var5},
     {Var6} foaf:currentProject {Var7}, {Var7} foaf:topic {Var5} ]
  WHERE ( ( (Person != l3s:tom) ) ) AND
        ( ( (Var2 = Person) AND (Var1 = l3s:alice) ) OR
        ( (Person = l3s:alice) ) ) AND
        ( ( (Var3 = Document) AND (Var2 = Person) AND
          (Person = l3s:alice) AND (Var4 = foaf:Document) ) )
MINUS
  CONSTRUCT {Person} foaf:name {Name};
         foaf:phone {Phone}; foaf:interest {Document}
  FROM {Person} foaf:name {Name};
     foaf:phone {Phone}; foaf:interest {Document}
   [ {Var8} foaf:currentProject {Var9} ]
  WHERE ( ( (Var8 = Person) AND (Var9 = l3s:rewerse) ) )
```

Figure 4.1: Pre-filtering a query, quotation from [89]. The received query could be expanded with MINUS operand so that some specific unwanted triples are not returned. Therefore, the result would be merely wanted triples.

1. The query is received from an agent (including users and data providers). The query is an attempt to read/insert/update/delete the data of the space.

2. The Authentication of the requester is checked, to ascertain whether it is a legitimate user, sending the query. It is done by comparing the credentials provided by the user at login time, with the previously stored data (e.g., passwords and biometric templates). If a match is confirmed, the user is authenticated and allowed to proceed with the process, and if not, the process stops here.

3. If the authentication was successful, then the query, a with the requested role, is forwarded to the Access Control Engine.

4. In the Access Control Engine first the context information determines if the situation is considered critical or normal. If it is critical, no control over the access is required. Thus any authentic user could gain access over the medical data and the medical history of the patient. In normal situations, the normal process is followed: a) It is checked whether the user has the requested role; if yes, the role is allocated.

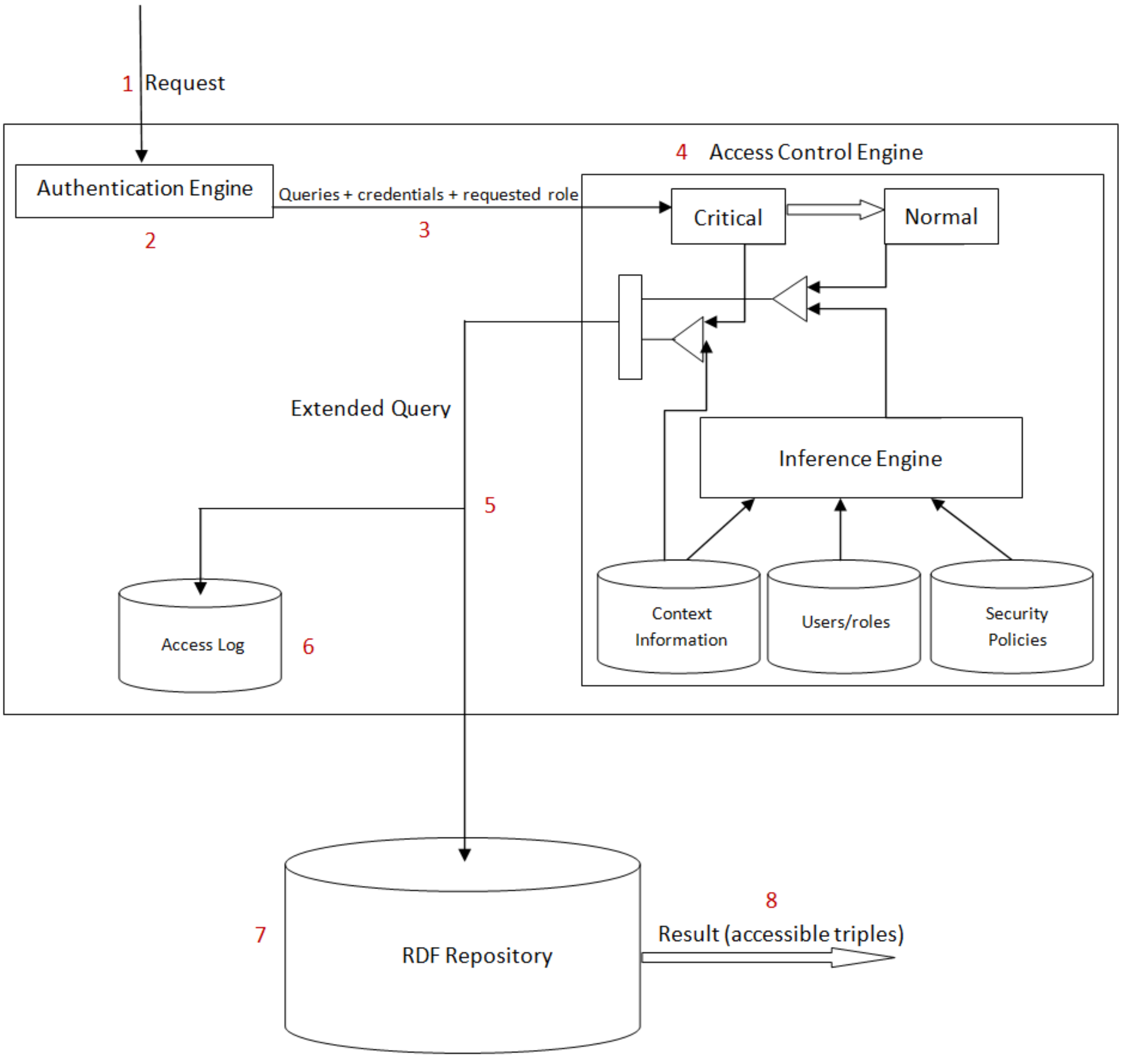


Figure 4.2: Security Framework. The framework comes on top of the RDF repository to check the authenticity and access of the requester. The Authentication Engine in (2), assures that the requester is an authentic user. The Access Control Engine in (4), checks whether the requester has the right to perform the requested action. Access Log in (6), keeps a record of the recent accesses. Finally in (8), the result (the triple (s) that are accessible by the requester) is retrieved from the repository and forwarded back to the user.

b) The required information for decision making is gathered from three different units including the Context Information, Roles, and Security Policies units. Inference Engine is an engine in charge of making decisions about the permission/prohibition of the requested access. According to the requester's role and context information, it executes the rules, and the makes proper decisions (according to the security policies, the received query is extended in a way that accessible triples are extracted from inaccessible ones. Thence, the result would be only the query to accessible triples).

5. When access to the requested information is approved, the extended query is forwarded to the RDF repository, to retrieve the triples. Also, a new record for this access is added to the Access Log.

6. In the Access Log, records of the recent accesses to the RDF repository are stored, which could be helpful in case of security breach. This way it is possible to track who got access and when, and to what information.

7. The RDF repository is where the information is stored, in form of triples (s, p, o) (subject, predicate, object).

8. The result is sent back to the requester.

4.3 Access Control Scheme Specifications for the Security Framework

There are different mechanisms used to control the access over resources. Comparing to others, Role Based Access Control (RBAC) is known as the most applicable one for health care information systems. The proposed access control scheme is a Context-Aware Role Based Access Control. This access control scheme is fairly similar to [26] in the way that both are combinations of RBAC and ABAC. What makes them different is that in [26], the roles are conceded to individuals dynamically in accordance with the contexts; however, in our system, roles are allocated statically to the users at the time of registering in the system by the central administrator, and assigned dynamically at the time of logging into the system. For more flexibility, users may have several roles and activate one of them when logging in the system.

In general, the proposed access control scheme is called Context-Aware RBAC, because it has the characteristics of both the RBAC and context awareness. It is an RBAC scheme, since users get permissions/prohibitions for accessing a resource, according to their roles in the system. It is context aware, since in three different points context information affects how the rules are executed. These point are: a) when the context information defines that the situation is critical, b) when the context information affects what role is assigned to a user, and c) when the context information affects how the rules are executed. We discuss about the effect of context information below in detail.

- The first case is where context information reports the patient's medical status as critical, which refers to the cases that some pre-defined emergency conditions are satisfied (e.g., the user's heart rate drops to zero or blood pressure rises unexpectedly). To define the critical situation, different information is collected from different sources. For instance, when the user's sensor shows a high blood pressure situation, the space should check from the user's calendar whether he/she is in a gym, in a meeting with boss, or resting at home. Then it can decide whether the high blood pressure has happened because of a normal activity or it is a critical situation. If it is a critical situation, no control over access to the data is checked. This eliminates the required time for authorization and access control for caregivers assisting the patient.

- The second case pertains to the influence of the context information on allocating the suitable role to the users. At this point, together with the role of the users in the organization, some other run-time parameters take part in decision making. For instance, The context of location and time may be taken into account in deciding which role and, consequently, which restrictions the user may have. For example:

(triple (Shohreh, requestsRole, Nurse))

(triple (Shohreh, hasData, LocationShohreh))

(triple (LocationShohreh, hasValue, TYKSHospital))

⇒

(assert(triple (Shohreh, hasRole, TYKSNurse)))

- The third case refers to the impact of the context information in rule execution and decision making. The time and location information may make the rules execute differently. For instance, a doctor with a different location than the hospital, or after working hours, has limited rights in comparison to the time during which he is working in the hospital.

  As an example: in office hours, a doctor in the hospital can update the medical history of the patients:

  (triple (Shohreh, hasRole, Doctor))

  (triple (Shohreh, hasData, LocationShohreh))

  (triple (LocationShohreh, hasValue, TYKSHospital))

  ⇒

  (assert (triple (Shohreh, roleHasReadPermissionOverData, ?MedicalHistory)))

  (assert (triple (Shohreh, roleHasWritePermissionOverData, ?MedicalHistory)))

  (assert (triple (Shohreh, roleHasUpdatePermissionOverData, ?MedicalHistory)))

  (assert (triple (Shohreh, roleHasDeletePermissionOverData, ?MedicalHistory)))

  However, a doctor is restricted to only reading the medical history of the patients after office time or outside the hospital:

  (triple (Shohreh, hasRole, Doctor))

  (triple (Shohreh, hasData, LocationShohreh))

  (triple (LocationShohreh, hasValue, TrainStation))

  ⇒

  (assert (triple (Shohreh, roleHasReadPermissionOverData, ?MedicalHistory)))

The other attribute that differentiates the proposed scheme from traditional RBAC is that, for privacy protection purposes, users of the system have the authority to specify some access control rules according to their preferences. The user defined rules are added to the rules defined by the administrator at the design time (participants have the right to request for restrictions on their information). Users may determine particular individuals to have the right to access, and similarly, to ban others from having specific permissions on their data, partially or completely. On that account, individually assigned access permissions and prohibitions would be advantageous. These access rules come on top of the administrative rules and have higher priority. This means that if an access to a resource is normally permitted for a role, but prohibited for a special user, then the permissive rule is overridden by the prohibition rule. In the same way, a permission determined by the user is preferred to a prohibition for a role assigned by admin.

For instance, a user named Jack with the role of Doctor, by default, has all access permissions over users' medical history including Read, Write, Update and Delete. Another user, Maria, with the role of Patient, declares that Jack should only be permitted to Read her medical history and be banned from Write, Update and Delete actions. As a result of the patient's preferences, Jack only is eligible to Read the medical history of Maria:

(triple (Jack, hasRole, Doctor))
(triple (Maria, hasRole, Patient))
(triple (Maria, hasMedicalHistory, ?h))
$\Rightarrow$
(assert (triple (Jack, roleHasReadPermissionOverData, ?h)))
(assert (triple (Jack, roleHasWritePermissionOverData, ?h)))
(assert (triple (Jack, roleHasUpdatePermissionOverData, ?h)))
(assert (triple (Jack, roleHasDeletePermissionOverData, ?h)))

On the other hand Maria's rule permits only Read action for Jack, and prohibits for three other actions:

(assert (triple (Jack, userHasReadPermissionOverData, ?h)))

(assert (triple (Jack, userHasUpdateProhibitionOverData, ?h)))

(assert (triple (Jack, userHasDeleteProhibitionOverData, ?h)))

(assert (triple (Jack, userHasWriteProhibitionOverData, ?h)))

Therefore, as a result of the right side of the rule, Jack is allowed to perform the Read operation.

4.3.1 Access Control Ontology

In order to represent and classify knowledge, there exists two different approaches, data-driven and knowledge-driven techniques. Data-driven methods rely more on machine learning and statistical approaches. Although they have shown to be accurate in many domains, they are not appropriate in dynamically changing environments, such as smart spaces. For such environments, knowledge-driven techniques are preferably used; for instance, rule-based systems and ontological approaches. [90]

In terms of modeling context information, several approaches are available to wield, e.g., object oriented models, graphical models, logic based models, key-value models and ontology-based models [91]. In the proposed architecture, an ontology based model is chosen due to its advantages. Firstly, it is flexible, expressive and generic; secondly, ontologies are the most favorable methods to model context information; thirdly, we can take advantage of ontology reasoning and automatic code generation. [53]

Figure 4.3 depicts the sketched access control ontology proposed in this thesis. Users are free to choose one of the credentials that is acceptable by the system in order to sign themselves into the system. For each of the authentication methods there is a trust level defined that shows how secure the authentication method is.

Users in the system may have several roles, but can activate only one of them at a time. By activating roles, users get rights [1] to access the data and perform actions.

[1] as mentioned before, rights might be assigned to the users instead of the roles

The rights are permissions and prohibitions to perform four different actions, including reading, updating and deleting the existing data on the space or writing new data to the space. As mentioned before, in order to support the privacy of the users, they are able to define some access control rules to have more restrictions on their own information. Therefore, as it is shown in figure 4.4, there are some relations between the classes, User and Data to define if a subject (user) is permitted or prohibited to perform an action on a specific object (data). These relations are similar to the relations we had between the Role and the Data entities.

The data is provided either by data providers (devices and sensors) or users. We have categorized the stored data into two different subcategories: a) personal data that refers to the personal information of the users provided by them, e.g., gender, address, social security number, and medical history. b) medical data that refers to the information provided by the data providers, e.g., sensors and devices.

To model our OWL ontology we have used Protégé 4.2 editor [92], a free open source platform. Figure 4.4 displays variants of the entities in our access control ontology.

An ontology, in Protégé is composed of various components, including *Classes*, *Object Properties*, *Data Properties* and *Individuals*. Individuals (represented by diamonds in an ontology) are the objects we are interested in. Classes (represented by circles or ovals in an ontology), are the abstract concept used to group individuals of certain type. Individuals, are connected to each other through properties, which are classified as object properties, mapping two individuals together, and data properties, mapping an individual to a data value (e.g., int, float, boolean, datetime). Several characteristics are defined for properties; next, we go briefly through them, with some examples regarding our ontology. [93]

a) Functional Property: there is only one single individual corresponding to the given property. For instance, in *(LocationShohreh hasDataProvider GPSShohreh), hasDataProvider* is a functional property because there is only one value (an individual, *GPSShohreh*)

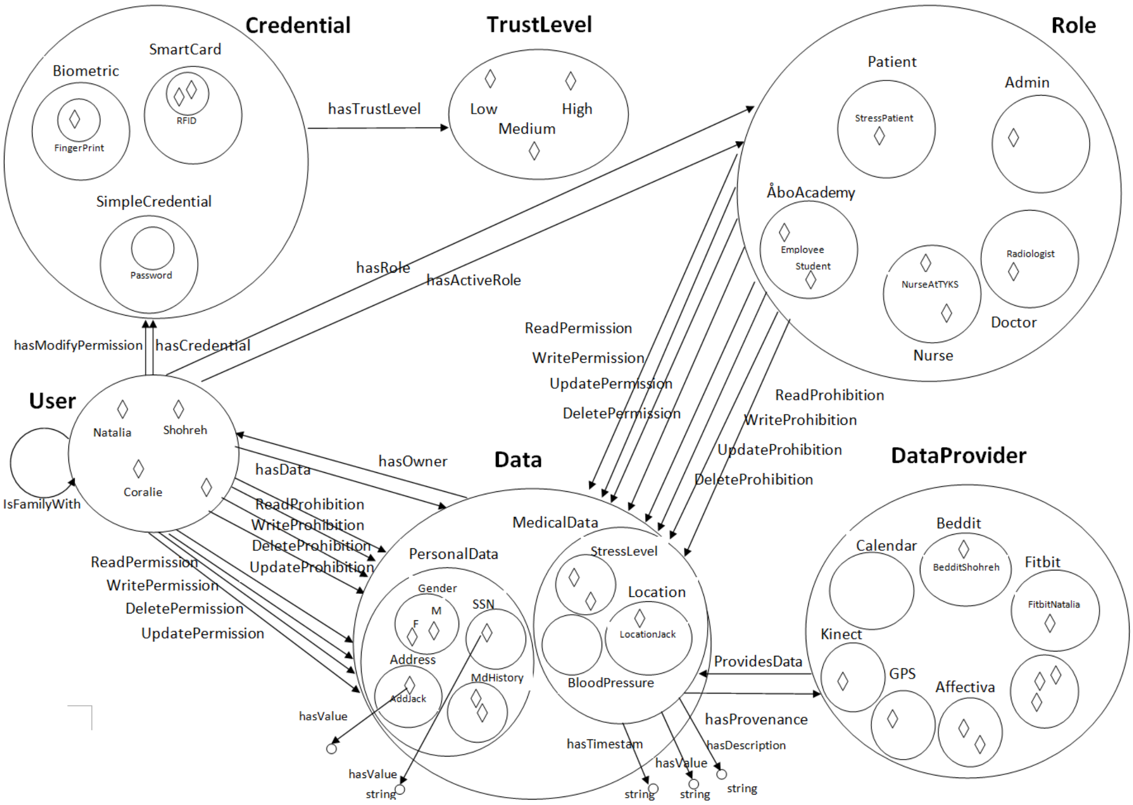
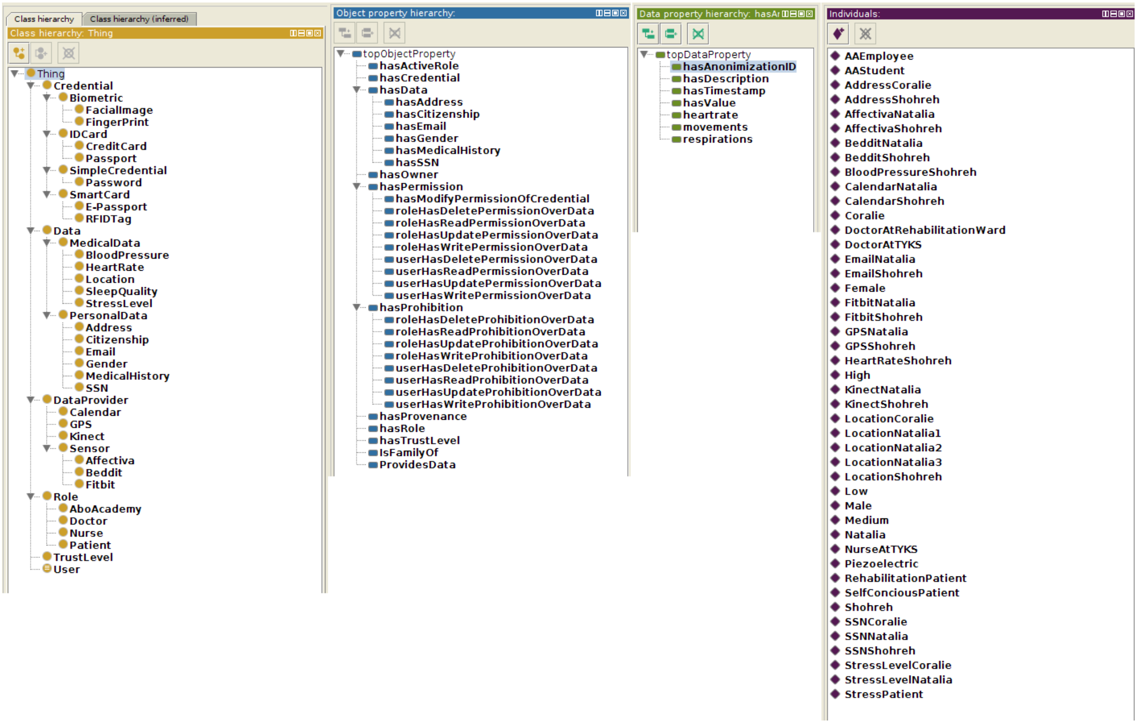


Figure 4.3: Access Control Ontology. Users choose credentials (with different trust levels) to login to the system. After logging in, they activate a role in order to get rights for accessing data. The possible rights are permission/prohibition for four different actions including read, write, update and delete. In some cases (in cases that users define some rules to have more restricted access on their information) the rights also might be given to individuals. The existing data is provided by data providers and belongs to the users of the space.

to be related to the individual *LocationShohreh*.

b) Transitive Property: If an individual I1 is associated with individual I2 and I2 is associated with individual I3, we can deduce that I1 and I3 are linked via property P. For instance: If *(Shohreh hasRole StressPatient)* and *(StressPatient hasReadPermission stressLevel-Shohreh)* then we can deduce that *(Shohreh hasReadPermission stressLevelShohreh)*, where I1 = *Shohreh*, I2 = *StressPatient*, I3 = *stressLevelShohreh*, and P = *hasReadPermis-*



Class hierarchy: Thing

```
Thing
  Credential
    Biometric
      FacialImage
      FingerPrint
    IDCard
      CreditCard
      Passport
    SimpleCredential
      Password
    SmartCard
      E-Passport
      RFIDTag
  Data
    MedicalData
      BloodPressure
      HeartRate
      Location
      SleepQuality
      StressLevel
    PersonalData
      Address
      Citizenship
      Email
      Gender
      MedicalHistory
      SSN
  DataProvider
    Calendar
    GPS
    Kinect
    Sensor
      Affectiva
      Beddit
      Fitbit
  Role
    AboAcademy
    Doctor
    Nurse
    Patient
  TrustLevel
  User
```

Object property hierarchy:

```
topObjectProperty
  hasActiveRole
  hasCredential
  hasData
    hasAddress
    hasCitizenship
    hasEmail
    hasGender
    hasMedicalHistory
    hasSSN
  hasOwner
  hasPermission
    hasModifyPermissionOfCredential
    roleHasDeletePermissionOverData
    roleHasReadPermissionOverData
    roleHasUpdatePermissionOverData
    roleHasWritePermissionOverData
    userHasDeletePermissionOverData
    userHasReadPermissionOverData
    userHasUpdatePermissionOverData
    userHasWritePermissionOverData
  hasProhibition
    roleHasDeleteProhibitionOverData
    roleHasReadProhibitionOverData
    roleHasUpdateProhibitionOverData
    roleHasWriteProhibitionOverData
    userHasDeleteProhibitionOverData
    userHasReadProhibitionOverData
    userHasUpdateProhibitionOverData
    userHasWriteProhibitionOverData
  hasProvenance
  hasRole
  hasTrustLevel
  IsFamilyOf
  ProvidesData
```

Data property hierarchy: hasAı

```
topDataProperty
  hasAnonimizationID
  hasDescription
  hasTimestamp
  hasValue
  heartrate
  movements
  respirations
```

Individuals:

```
AAEmployee
AAStudent
AddressCoralie
AddressShohreh
AffectivaNatalia
AffectivaShohreh
BedditNatalia
BedditShohreh
BloodPressureShohreh
CalendarNatalia
CalendarShohreh
Coralie
DoctorAtRehabilitationWard
DoctorAtTYKS
EmailNatalia
EmailShohreh
Female
FitbitNatalia
FitbitShohreh
GPSNatalia
GPSShohreh
HeartRateShohreh
High
KinectNatalia
KinectShohreh
LocationCoralie
LocationNatalia1
LocationNatalia2
LocationNatalia3
LocationShohreh
Low
Male
Medium
Natalia
NurseAtTYKS
Piezoelectric
RehabilitationPatient
SelfConciousPatient
Shohreh
SSNCoralie
SSNNatalia
SSNShohreh
StressLevelCoralie
StressLevelNatalia
StressPatient
```

Figure 4.4: Protégé Entities of Access Control Ontology

sion.

c) Symmetric Property: If an individual I1 is linked to individual I2 via a symmetric property P, then we can claim that individual I2 is linked to individual I1 via the same property P. As an example, if *(Coralie isFamilyWith Shohreh)*, then *(Shohreh isFamilyWith Coralie)*, where I1 = *Coralie*, I2 = *Shohreh*, and P = *isFamilyWith*.

d) Anti-Symmetric Property: If an individual I1 is linked to individual I2 via an anti-symmetric property P, then we cannot say that individual I2 is linked to individual I1 via the same property P. For example, if *(Natalia hasCredential fingerprintNatalia)*, then we cannot say that *(fingerprintNatalia hasCredential Natalia)*, where I1 = *Natalia*, I2 = *fingerprintNatalia*, and P = *hasCredential*.

e) Reflexive Property: A reflexive property links an individual I to itself. As an example, in the triple *(Shohreh knows Shohreh)*, the property P = *knows* is a reflexive property where I = *Shohreh*.

4.3.2 Access Control Rules

In our system, the access control policies are expressed via rules. The possible actions are inserting new data to the space, reading, updating/modifying (combination of the two actions, delete and insert) and deleting the existing data. The rules (including allow and deny) are separated into two groups: the rules designed by the administrator at design time, and the rules defined by the end user. At run-time, the rules are executed in order to decide whether an agent is permitted or prohibited to perform an action. The second group that reflects user's preferences has higher priority. Moreover, there are cases, termed as "conflict", that occur when a triple comes in the scope of both positive and negative permissions. Therefore, there must be some legislation to decide which one is to be considered. Figure 4.5 clarifies how a group how a group outweighs the other, in compliance with the administrative rules [94]. P stands for positive rules, and N for negative rules. In a) and c), in case of any conflicts, positive rules are preferred to negative ones; while in b) and d), negative rules are preferred. In a) and b), for the cases not defined, decisions are assumed to be positive by default; while in c) and d), decisions are assumed to be negative by default. In all four cases, the colored regions display the accessible domains. Negative-negative cases, i.e., d) are known tosecure situations because they have the most restrictive permissive area. [94]

We have designated the negative-negative status in our access control rules. In addition to allow and deny rules, we have some other rules defined to resolve conflicts. In our case a negative rule has higher priority and it overrules the positive rules. Therefore, we will have prohibition of an action in case a conflict occurs. There is an exception to this strategy, which is the case that the negative rule is defined by the administrator and the positive rule by a user. In this situation, the user's rule (positive one) overrules the administrator's rule (positive one) due to its higher priority. We also selected the negative default for the non-defined cases. For instance, if the access for a particular subject to a specific object was not defined, the access would be prohibited.

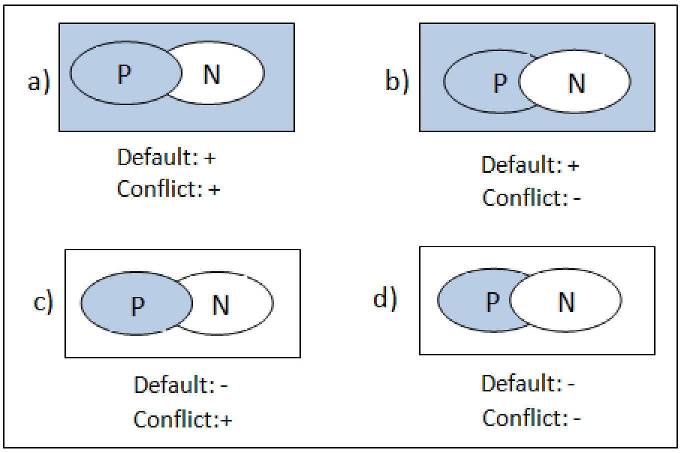


Figure 4.5: Policy Semantics, quotation from [94]: the dark areas in the figures show the positive access rules and the light areas show the negative rules. There are some cases in the rules that a triple comes in the scope of both positive and negative rules. There are different policies for solving the conflict and decide about non-defined cases as it is shown in the figure: a) Positive rules are selected as default and in case of conflict. b) Positive rules are selected as default and negative in case of conflict. c) Negative rules are selected as default and Positive in case of conflict. d) Negative rules are selected as default and in case of conflict.

For writing the access control rules, we are using **C Language Integrated Production System (CLIPS)** version 6.24. CLIPS is a software tool for developing expert systems. It was firstly developed in 1984 in NASA's Johnson Space Center to amend the failures that LIPS-based expert systems had. LISP is the language that was being used in most of the expert systems at its time; however, it had three main failures (low availability, high cost and poor integration with other languages) that hampered NASA from using LISP-based expert systems and induced it to use a conventional language such as C; but the problems remained the same. Therefore, they started developing a C-based expert system tool called CLIPS. Later in 1986 its version 3.0 was released for use outside NASA. CLIPS itself is developed in C language, and it has been a great help in improvement of expert system technology because of its particular characteristics including: its low-cost, portability, extensibility and capabilities. These remarkable features have made this language to be widely used in both public (e.g., in governmental and military fields, and in all sites of NASA) and private sectors (e.g., in many universities for academic and training purposes, and many companies). Originally, CLIPS was merely a rule language based on the RETE algorithm [95]. The RETE algorithm is a method used to compare a large set of patterns with a large set of objects to find all possible matches. The intention in using RETE is to improve the speed in rule-based systems, by providing faster computation.

After years of further development in CLIPS language, it supports numerous new features such as procedural programming, Object Oriented Programming), integration between rule-based and OOP programming. [96]

Our access control rules are as follows:

(deftemplate MAIN::triple (slot s) (slot p) (slot o))

1) Administrator has *modify* permission on the users' credentials.

(defrule AdminModifyCredential (triple (s ?s1)(p hasRole)(o Admin))

(triple (s ?s2)(p hasCredential)(o ?c))

(triple (s ?c)(p hasValue)(o ?v))

⇒

(assert(triple (s ?s1)(p hasModifyPermission)(o ?c))))

2) Users have *modify* permission on their credentials.

(defrule UserModifyCredential

(triple (s ?s1)(p hasCredential)(o ?c))

(triple (s ?c)(p hasValue)(o ?v))

⇒

(assert(triple (s ?s1)(p hasModifyPermission)(o ?c))))

3) Users have *read* permission on their own Medical Data provided by data providers.

(defrule userHasReadPermissionOverMedicalData

(triple(s ?s1)(p hasData)(o ?MedicalData))

(triple(s ?MedicalData)(p hasProvenance)(o ?DataProvider))

⇒

(assert(triple(s ?s1)(p userHasReadPermissionOverData)(o ?MedicalData))))

4) Users can do any action on their own Personal Information.

(defrule userHasAllPermissionsOverPersonalData

(triple(s ?s1)(p hasEmail)(o ?email))

(triple(s ?email)(p hasValue)(o ?v1))

(triple(s ?s1)(p hasSSN)(o ?ssn))

(triple(s ?ssn)(p hasValue)(o ?v2))

(triple(s ?s1)(p hasAddress)(o ?add))

(triple(s ?add)(p hasValue)(o ?v3))

(triple(s ?s1)(p hasCitizenship)(o ?cz))

(triple(s ?cz)(p hasValue)(o ?v4))

(triple(s ?s1)(p hasGender)(o ?gen))

(triple(s ?s1)(p hasMedicalHistory)(o ?his))

⇒

(assert(triple(s ?s1)(p userHasReadPermissionOverData)(o ?email)))

(assert(triple(s ?s1)(p userHasReadPermissionOverData)(o ?add)))

(assert(triple(s ?s1)(p userHasReadPermissionOverData)(o ?cz)))

(assert(triple(s ?s1)(p userHasReadPermissionOverData)(o ?gen)))

(assert(triple(s ?s1)(p userHasReadPermissionOverData)(o ?his)))

(assert(triple(s ?s1)(p userHasReadPermissionOverData)(o ?ssn)))

(assert(triple(s ?s1)(p userHasWritePermissionOverData)(o ?email)))

```
(assert(triple(s ?s1)(p userHasWritePermissionOverData)(o ?add)))
(assert(triple(s ?s1)(p userHasWritePermissionOverData)(o ?cz)))
(assert(triple(s ?s1)(p userHasWritePermissionOverData)(o ?gen)))
(assert(triple(s ?s1)(p userHasWritePermissionOverData)(o ?his)))
(assert(triple(s ?s1)(p userHasWritePermissionOverData)(o ?ssn)))

(assert(triple(s ?s1)(p userHasUpdatePermissionOverData)(o ?email)))
(assert(triple(s ?s1)(p userHasUpdatePermissionOverData)(o ?add)))
(assert(triple(s ?s1)(p userHasUpdatePermissionOverData)(o ?cz)))
(assert(triple(s ?s1)(p userHasUpdatePermissionOverData)(o ?gen)))
(assert(triple(s ?s1)(p userHasUpdatePermissionOverData)(o ?his)))
(assert(triple(s ?s1)(p userHasUpdatePermissionOverData)(o ?ssn)))

(assert(triple(s ?s1)(p userHasDeletePermissionOverData)(o ?email)))
(assert(triple(s ?s1)(p userHasDeletePermissionOverData)(o ?add)))
(assert(triple(s ?s1)(p userHasDeletePermissionOverData)(o ?cz)))
(assert(triple(s ?s1)(p userHasDeletePermissionOverData)(o ?gen)))
(assert(triple(s ?s1)(p userHasDeletePermissionOverData)(o ?his)))
(assert(triple(s ?s1)(p userHasDeletePermissionOverData)(o ?ssn))))
```

5) Doctors have *read* permission on both Medical Data and Personal Data of the patients.

```
(defrule DoctorPermission
(triple(s ?s1)(p hasRole)(o Doctor))
(triple(s ?s2)(p hasRole)(o Patient))
(triple(s ?s2)(p hasFamilyDoctor)(o ?s1))
(triple(s ?s2)(p hasEmail)(o ?email))
(triple(s ?email)(p hasValue)(o ?v1))
```

(triple(s ?s2)(p hasSSN)(o ?ssn))

(triple(s ?ssn)(p hasValue)(o ?v2))

(triple(s ?s2)(p hasAddress)(o ?add))

(triple(s ?s2)(p hasCitizenship)(o ?cz))

(triple(s ?cz)(p hasValue)(o ?v4))

(triple(s ?s2)(p hasGender)(o ?gen))

(triple(s ?s2)(p hasMedicalHistory)(o ?his))

(triple(s ?his)(p hasValue)(o ?v5))

(triple(s ?s2)(p hasData)(o ?MedicalData))

(triple(s ?MedicalData)(p hasProvenance)(o ?DataProvider))

$\Rightarrow$

(assert(triple(s ?s1)(p roleHasReadPermissionOverData)(o ?MedicalData)))

(assert(triple(s ?s1)(p roleHasReadPermissionOverData)(o ?his)))

(assert(triple(s ?s1)(p roleHasReadPermissionOverData)(o ?email)))

(assert(triple(s ?s1)(p roleHasReadPermissionOverData)(o ?ssn)))

(assert(triple(s ?s1)(p roleHasReadPermissionOverData)(o ?add)))

(assert(triple(s ?s1)(p roleHasReadPermissionOverData)(o ?gen)))

(assert(triple(s ?s1)(p roleHasReadPermissionOverData)(o ?cz))))

6) Family Members introduced by the user have the *read* permission on Medical Data and Personal Data of the user.

(defrule FamilyAccessPermissions

(triple(s ?s1)(p hasEmail)(o ?email))

(triple(s ?email)(p hasValue)(o ?v1))

(triple(s ?s1)(p hasSSN)(o ?ssn))

(triple(s ?ssn)(p hasValue)(o ?v2))

(triple(s ?s1)(p hasAddress)(o ?add))

```
(triple(s ?add)(p hasValue)(o ?v3))
(triple(s ?s1)(p hasCitizenship)(o ?cz))
(triple(s ?cz)(p hasValue)(o ?v4))
(triple(s ?s1)(p hasGender)(o ?gen))
(triple(s ?s1)(p hasMedicalHistory)(o ?his))
(triple(s ?his)(p hasValue)(o ?v5))
(triple(s ?s1)(p hasData)(o ?MedicalData))
(triple(s ?MedicalData)(p hasProvenance)(o ?DataProvider))
(triple (s ?f) (p isFamilyOf) (o ?s1))
(test(neq ?s1 ?f))
⇒
(assert(triple(s ?f)(p roleHasReadPermissionOverData)(o ?MedicalData)))
(assert(triple(s ?f)(p roleHasReadPermissionOverData)(o ?his)))
(assert(triple(s ?f)(p roleHasReadPermissionOverData)(o ?email)))
(assert(triple(s ?f)(p roleHasReadPermissionOverData)(o ?ssn)))
(assert(triple(s ?f)(p roleHasReadPermissionOverData)(o ?add)))
(assert(triple(s ?f)(p roleHasReadPermissionOverData)(o ?gen)))
(assert(triple(s ?f)(p roleHasReadPermissionOverData)(o ?cz))))
```

7) Conflict-Solving rule for *read* permissions:

```
(defrule ReadPreference
(triple(s ?s)(p userHasReadProhibitionOverData)(o ?o))
(triple(s ?s)(p roleHasReadPermissionOverData)(o ?o))
?f1 ⇐ (triple(s ?s)(p roleHasReadPermissionOverData)(o ?o))
⇒
(retract ?f1))
```

8) Conflict-Solving rule for *update* permissions:

(defrule UpdatePreference

(triple(s ?s)(p userHasUpdateProhibitionOverData)(o ?o))

(triple(s ?s)(p roleHasUpdatePermissionOverData)(o ?o))

?f1 ⇐ *(triple(s ?s)(p roleHasUpdatePermissionOverData)(o ?o))*

⇒

(retract ?f2))

9) Conflict-Solving rule for *insert* permissions:

(defrule WritePreference

(triple(s ?s)(p userHasWriteProhibitionOverData)(o ?o))

(triple(s ?s)(p roleHasWritePermissionOverData)(o ?o))

?f3 ⇐ *(triple(s ?s)(p roleHasWritePermissionOverData)(o ?o))*

⇒

(retract ?f3))

10) Conflict-Solving rule for *delete* permissions:

(defrule DeletePreference

(triple(s ?s)(p userHasDeleteProhibitionOverData)(o ?o))

(triple(s ?s)(p roleHasDeletePermissionOverData)(o ?o))

?f4 ⇐ *(triple(s ?s)(p roleHasDeletePermissionOverData)(o ?o))*

⇒

(retract ?f4))

11) When a request for the action *read* is received, it should be checked if the requester has the permission to read the specific data. If the user is permitted, then the action is performed and the requested data is printed out.

(defrule ControllingRead

?f1 ⇐ (triple(s ?s1)(p requestsForRead)(o ?o1))

(triple(s ?o1)(p hasValue)(o ?v1))

(or (triple(s ?s1)(p roleHasReadPermissionOverData)(o ?o1)) (triple(s ?s1)(p userHasReadPermissionOverData)(o ?o1))) : This checks for the existence of any of these two triples (having either role or individual permission for reading)

⇒

(retract ?f1))

(printout t ?v1 crlf) : This prints out the requested value if the permission is approved.

12) When a request for the action *insert* is received, it should be checked if the requester has the permission to insert new data on the specific object. If the user is permitted, then the action is performed and the new data is inserted.

(defrule ControllingInsert

?f1 ⇐ (triple(s ?s1)(p requestsForInsertOn)(o ?o1))

?f2 ⇐ (triple(s ?s1)(p insertNewValue)(o ?v2)) : the new value user is trying to insert

?f3 ⇐ (triple(s ?o1)(p hasValue)(o ?v1)) : the current value that the object has

(or (triple(s ?s1)(p roleHasWritePermissionOverData)(o ?o1)) (triple(s ?s1)(p userHasWritePermissionOverData)(o ?o1))) : This line checks if the user is permitted to write, either individually or by the role it has got.

⇒

(retract ?f1 ?f2 ?f3))

(assert(triple(s ?o1)(p hasValue)(o (str-cat ?v1 ?v2)))) : The request is approved, so this line concats the old value with the new one.

13) When a request for the action *delete* is received, it should be checked if the requester has the permission to delete the specific data. If the user is permitted, then the action is

performed and the data is removed.

```
(defrule ControllingDelete
?f1 ⇐ (triple(s ?s1)(p requestsForDeleteOn)(o ?o1))
(triple(s ?o1)(p hasValue)(o ?v1))
(or (triple(s ?s1)(p roleHasDeletePermissionOverData)(o ?o1)) (triple(s ?s1)(p userHasDeletePermissionOverData)(o ?o1)))
⇒
(assert(triple(s ?o1)(p hasValue)(o ””)))
(retract ?f1))
```

14) When a request for the action *update* is received, it should be checked if the requester has the permission to update the specific data. If the user is permitted, then the action is performed and the data is updated.

```
(defrule ControllingUpdate
?f1 ⇐ (triple(s ?s1)(p requestsForUpdateOn)(o ?o1))
?f2 ⇐ (triple(s ?s1)(p UpdatingNewValue)(o ?v2))
?f3 ⇐ (triple(s ?o1)(p hasValue)(o ?v1))
(or (triple(s ?s1)(p roleHasUpdatePermissionOverData)(o ?o1))(triple(s ?s1)(p roleHasUpdatePermissionOverData)(o ?o1)))
⇒
(assert(triple(s ?o1)(p hasValue)(o ?v2)))
(retract ?f1 ?f2 ?f3))
```

4.4 Evaluation of the Access Control Scheme

The case study focuses on a health care information system, in which there exists a range of personal and medical information about the inhabitants of the space, stored on the

SIB, in triple format. Users take the available roles, i.e., administrator, patient, doctor, and patient's family member. In accordance with their roles, they are given privileges to access the data on the SIB. In the experiment, we measure the required time to check the access for four different types of requests, including insert new data to the SIB, read, delete and update (combination of deleting and inserting) the existing data for different data set sizes. The size of the SIB grows by increasing the number of the triples (from 1 to 100000). From the result of the response times, we can have an estimation on the overhead that our access control scheme brings to the system.

The experiment is done on an Intel Core 2 Duo CPU E8500 @ 3.16GHz, on a virtual machine with a RAM of 1.5 GB, and Ubuntu 12.04 LTS. The Python language was used because it is one of the supported languages by the KP interface (KPI). We had the SIB and test KP on the same machine. For more information on running Smart-M3, we refer the reader to [97].

4.4.1 Experiment Setup

Figure 4.6 illustrates the sequence diagram of how our access control works.

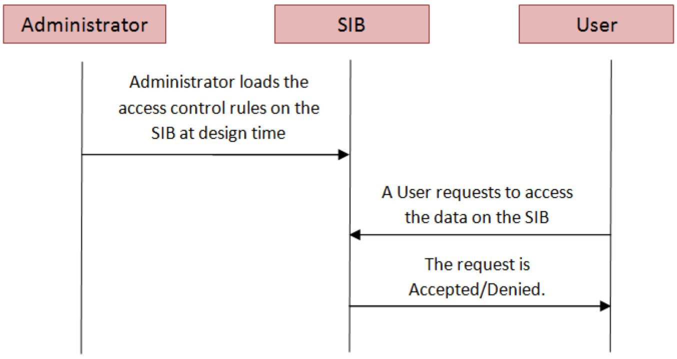

Administrator
SIB
User
Administrator loads the access control rules on the SIB at design time
A User requests to access the data on the SIB
The request is Accepted/Denied.

Figure 4.6: Access Control Sequence Diagram. Administrator sets up security rules on the SIB at design time. When a request to access the data on the SIB is received, it is checked whether the user has the permission to access or not.

The administrator defines the access control rules in CLIPS and adds them to the

SIB at design time. The listing 4.1 loads the access control rules (described in detail in section 4.3.2) on the SIB. The rules are added as a file ”AccessControlRules.clp” into the ”RedSibCLIPS.c” file where the CLIPS environment is Initialized.

```
 /* Loading Access Control Rules in the SIB.*/
char *a = "AccessControlRules.clp";
Load(a);
```

Listing 4.1: Loading Access Control Rules in the SIB

Users register in the system and the collected information from the users is added to the SIB, in the triple format. The users are given some role(s). When logging in the system, they activate one of their roles (e.g., admin, doctor, patient, family member). In order to have some data in the SIB to run the experiments, we filled the SIB with triples of some test users. Listing 4.2 adds some test users and triples to the SIB. First a KP joins the SIB, adds some triples and then leaves the SIB. Here we have used our computer as the KP and the IP address is the IP address of our computer.

```
from smart_m3.m3_kp import *
import uuid
import time
from time import time
import subprocess
import platform
NS = "http://www.abo.fi/SSOntology.owl#"
class ProducerKP(KP):
    def __init__(self, server_ip, server_port):
        KP.__init__(self, str(uuid.uuid4())+"_ProducerKP")
        self.ss_handle = ("X",(TCPConnector,(server_ip,server_port)))
    def join_sib(self):
        self.join(self.ss_handle)
    def leave_sib(self):
        self.leave(self.ss_handle)
    def insert(self, triples):
        ins = self.CreateInsertTransaction(self.ss_handle)
        ins.send(triples)
        self.CloseInsertTransaction(ins)
    def remove(self, triples):
        rem = self.CreateRemoveTransaction(self.ss_handle)
        rem.remove(triples)
        self.CloseRemoveTransaction(rem)
    def update(self, i_trip, r_trip):
        # Update = Insert + Remove
        upd = self.CreateUpdateTransaction(self.ss_handle)
        upd.update(i_trip, "RDF-M3", r_trip, "RDF-M3")
        self.CloseUpdateTransaction(upd)
pd = ProducerKP("10.0.2.15", 10010)
pd.join_sib()
while True:
#--------------------------------Admins--------------------------------
    pd.insert([Triple(URI("Tom"),URI("hasRole"),Literal("Admin"))])
#--------------------------------Doctors--------------------------------
    pd.insert([Triple(URI("Robert"),URI("hasRole"),Literal("Doctor"))])
    pd.insert([Triple(URI("Maria"),URI("hasRole"),Literal("Doctor"))])
#--------------------------------Patient 1--------------------------------
    pd.insert([Triple(URI("Sim"),URI("hasRole"),URI("Patient"))])
    pd.insert([Triple(URI("Sim"),URI("hasCredential"),Literal("
    fingerPrintSim.png"))])
    pd.insert([Triple(URI("Sim"),URI("hasData"),URI("locationSim"))])
    pd.insert([Triple(URI("locationSim"),URI("hasProvenance"),URI("
    GPSSim"))])
    pd.insert([Triple(URI("Sim"),URI("hasCitizenship"),URI("Canada"))])
    pd.insert([Triple(URI("SimCitizenship"),URI("hasValue"),Literal("'
    SimCitizenship'"))])
    pd.insert([Triple(URI("Sim"),URI("hasMedicalHistory"),URI("
    SimMedicalHistory"))])
    pd.insert([Triple(URI("SimMedicalHistory"),URI("hasValue"),Literal(
    ”'Migrane'”))])
    pd.insert([Triple(URI(”Sim”),URI(”hasGender”),Literal(”female”))])
    pd.insert([Triple(URI(”Sim”),URI(”hasAddress”),Literal(”Hameenkatu
    5”))])
    pd.insert([Triple(URI(”Sim”),URI(”hasEmail”),Literal(”Sim2012@yahoo
    .com”))])
    pd.insert([Triple(URI(”Sim”),URI(”hasSSN”),Literal(”C8906”))])
    pd.insert([Triple(URI(”Sim”),URI(”hasFamilyDoctor”),Literal(”Robert
    ”))])
    pd.insert([Triple(URI(”Toumas”),URI(”isFamilyOf”),Literal(”Sim”))])
pd.leave_sib()
```

Listing 4.2: Filling the SIB with some test triples

Any time that a user attempts to access the data on the SIB (by sending a request), the security rules on the SIB are executed in order to check that the requester has the permission to gain access on the requested resource or not. For execution of the rules, the requester's role and the context information are taken into account. With the code in listing 4.3, we measure the time that it takes to check the access for a request to read a triple from the SIB. In this example, a doctor requests to read the medical history of a patient. By adding this triple rule number 11 is activated. If the patient is under the doctor's medication, the access is allowed and the requested data is sent back to him/her.

```
from smart_m3.m3_kp import *
import uuid
import time
from time import time
import subprocess
import platform
NS = ”http://sm3-tut/Ontology.owl#”
class ProducerKP(KP):
  def __init__(self, server_ip, server_port):
    KP.__init__(self, str(uuid.uuid4())+”_ProducerKP”)
    self.ss_handle = (”X” ,(TCPConnector ,( server_ip , server_port )))
  def join_sib(self):
    self.join(self.ss_handle)
  def leave_sib(self):
    self.leave(self.ss_handle)
  def insert(self, triples):
    ins = self.CreateInsertTransaction(self.ss_handle)
    ins.send(triples)
    self.CloseInsertTransaction(ins)
  def remove(self, triples):
    rem = self.CreateRemoveTransaction(self.ss_handle)
    rem.remove(triples)
    self.CloseRemoveTransaction(rem)
  def update(self, i_trip, r_trip):
    # Update = Insert + Remove
    upd = self.CreateUpdateTransaction(self.ss_handle)
    upd.update(i_trip, ”RDF-M3”, r_trip, ”RDF-M3”)
    self.CloseUpdateTransaction(upd)
pd = ProducerKP(”10.0.2.15”, 10010)
pd.join_sib()
while True:
  test_triple = [Triple(URI(”Robert”),URI(”requestsForRead”),URI(”
    SimMedicalHistory”))]
  str_time = time()
#by inserting this triple, rule number 11 in the access control rules
    will be fired.
  pd.insert(test_triple)
#we do not need this triple, so we remove it from CLIPS.
  pd.remove(test_triple)
  end_time = time()
  print ”Execution time: ” + str(end_time-str_time)
pd.leave_sib()
```

Listing 4.3: Measuring the response time to check a read request

With the code in listing 4.4, we measure the time that it takes to check the access for a request to insert a triple to the SIB. To avoid the redundancy, here we only write the code after joining the SIB (from the point that the insert request is received).

```
pd.join_sib()
while True:
    triple1 = [Triple(URI("Robert"),URI("requestsForInsertOn"),URI("SimMedicalHistory"))]

    triple2 = [Triple(URI("Robert"),URI("InsertingNewValue"),Literal("'Some pills are prescribed. (15.01.2014)'"))]
    str_time = time()
#by inserting these two triples, rule number 12 in the access control rules will be fired.
    pd.insert(triple1)
    pd.insert(triple2)
#we do not need these two triples, so we remove them from CLIPS.
    pd.remove(triple1)
    pd.remove(triple2)
    end_time = time()
    print "Execution time: " + str(end_time-str_time)
pd.leave_sib()
```

Listing 4.4: Measuring the response time to check a insert request

With the code in listing 4.5, we measure the time that it takes to check the access for a request to delete a triple from the SIB.

```
pd.join_sib()
while True:
    test_triple = [Triple(URI("Robert"),URI("requestsForDeleteOn"),URI("SimMedicalHistory"))]
    str_time = time()
#by inserting this triple, rule number 13 in the access control rules will be fired.
    pd.insert(test_triple)
#we do not need this triple, so we remove it from CLIPS.
    pd.remove(test_triple)
    end_time = time()
    print ”Execution time: ” + str(end_time-str_time)
pd.leave_sib()
```

Listing 4.5: Measuring the response time to check a delete request

With the code in listing 4.6, we measure the time that it takes to check the access for a request to update a triple on the SIB.

```
pd.join_sib()
while True:
    triple1 = [Triple(URI(”Robert”),URI(”requestsForUpdateOn”),URI(”SimMedicalHistory”))]
    triple2 = [Triple(URI(”Robert”),URI(”UpdatingNewValue”),Literal(”’Sim is under medication of Robert from 18.09.2013 for his Migrane’”))]
    str_time = time()
# by inserting these two triples, rule number 14 in the access control rules will be fired.
    pd.insert(triple1)
    pd.insert(triple2)
#we do not need these two triples, so we remove them from CLIPS.
    pd.remove(triple1)
    pd.remove(triple2)
    end_time = time()
    print ”Execution time: ” + str(end_time-str_time)
pd.leave_sib()
```

Listing 4.6: Measuring the response time to check an update request

4.5 Analysis of the Results

We evaluate the overhead of the access control scheme by measuring the time it takes to check the access of the requester to the requested resource. The experiment is done for four different requests supported by the SSAP protocol, including the request to read, insert, delete and update a triple. We repeat the experiment for different data sets with different numbers of triples, ranging from 1 to 100000. The result of the experiment is presented in Table 4.1.

Table 4.1: Access Controlling Time for Different Requests and Different Sizes of SIB

No. of Triples	Read Request Time (s)	Insert Request Time (s)	Delete Request Time (s)	Update Request Time (s)
1	0.00471	0.00621	0.00422	0.00738
10	0.00407	0.00582	0.00439	0.00607
100	0.00387	0.00750	0.00408	0.00680
1000	0.00539	0.00905	0.00496	0.00909
10000	0.01908	0.03672	0.02008	0.03145
100000	0.10668	0.20454	0.10594	0.19553

The result of the experiment is illustrated as a plot in figure 4.7. The response time to check the access for four different actions is tested and the results are displayed in four distinct colors.

As seen in figure 4.6, with the enlargement in the size of the SIB, the execution time grows gradually for each plot. Throughout the whole experiment, the graphs related to the response time of the *Insert Request* and *Update Request* have approximately the same

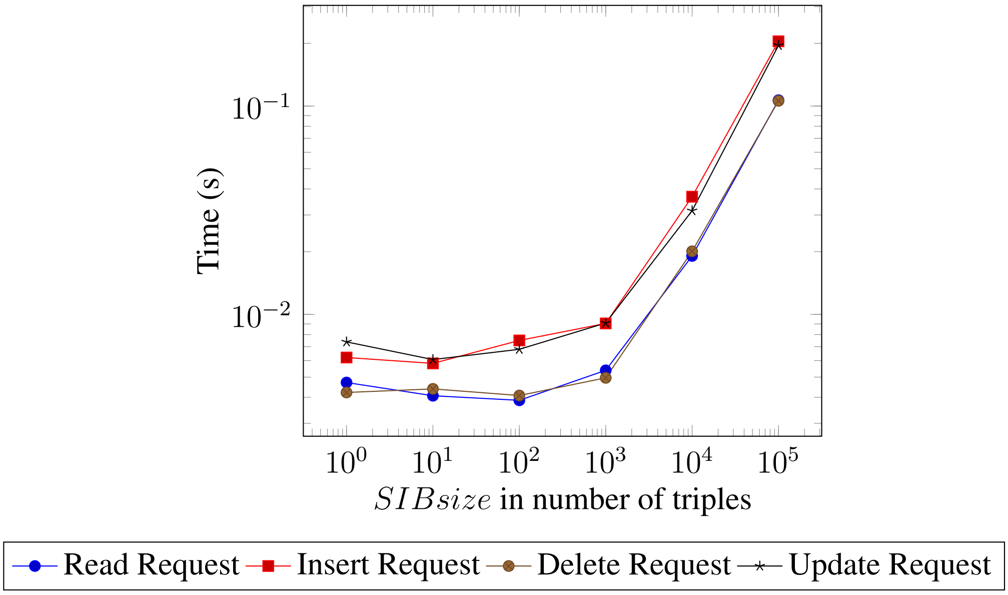


Figure 4.7: Access controlling average time for different requests and different sizes of the SIB

scheme. Their similar functions at triple level explains this behavior. Moreover, this two requests hold higher response times than the other two ones. The *Delete Request* (a request to delete a triple from SIB appears) appears to be the fastest request to be checked in execution time because in this operation no value is returned.

As a conclusion drawn from the experimental results we can claim that despite this access control method's overhead, it is still efficient to be used. As mentioned before, some overhead is acceptable in enhancing the security and privacy levels of the system especially when dealing with medical or personal information.

4.6 Privacy Solution

As mentioned before, this thesis is concerned with health-care information security and privacy. Undoubtedly, by the migration of health data from traditional paper-based records to electronic systems, accessing the information has become faster and the dissemination of the information has become more convenient. The purpose in the e-health systems is to share information without putting confidentiality, integrity, and availability of the assets at risk [66]. The Health Insurance Probability and Accountability Act (HIPAA) [98] defines the privacy in health-care systems as:

> ”The regulations which address the protection of patient information in any format and by any user. Privacy necessitates providing an individual’s health related information and disclosure of how and where that information being used.”

The ”Privacy Rule” was issued for the first time by U.S. Department of Health and Human Services (HHS) to protect the Individually Identifiable Health Information (the information the disclosure of which may result in identification of the that person). This information includes the past, present and future health condition of the individual, and also personal information that comes in the form of name, birth date, address, and social security number. The Privacy Rule addresses a balance between preserving the privacy and revealing the health information for proper use by limiting the data disclosure only to necessary circumstances. [99]

In our system, the privacy of the participants is protected in three ways:

- Data is encrypted during transmission and while stored in the data repository. Therefore, the system would be resistant against security attacks such as physical attacks, spoofing attacks and man-in-the-middle attacks. For instance, even if an attacker gets access to the data repository or spoofs the data while transmitting, he cannot read the data because he does not have the correct decryption key.

- Users declare some positive or negative access control rules. In this way, they can choose who can access what information and generally control the restriction level over their data. This is done by individually assigned rules (see section 4.3.2. Access Control Rules).

- Users have the option to label some of their information as highly sensitive. Any time this highly sensitive information is accessed, the user is informed by an email about the detail of the access (who accessed and when). Certainly, only limited information (e.g., social security number) should be grouped as highly sensitive to avoid network overhead.

Chapter 5

Conclusion

The goal of the smart spaces is to boost the quality of people's lives by making their environments better places to live. For this purpose, different devices, data and service providers come together to share information. On this basis, security and privacy of the data, while stored or transmitted, are crucial factors to address in any information system.

Recently, smart spaces have drawn a lot of attention and interest, in various domains including eHealth, which is the main emphasis of our case study. The objective of this Master's thesis was to design a security framework that enhances the security in Smart-M3 project, and to implement an access control scheme in order to prevent unauthorized access. The framework is composed of different components and modules to support different aspects of security such as authentication, authorization, and access control, and also to preserve the privacy of the participants of the system. The proposed access control is a Context-Aware Role Based Access Control (CARBAC) scheme. It is called RBAC because the users get roles regarding their responsibilities in the system, and it is called Context Aware (CA) because of the impact of the context information on the access control scheme. Context information refers to the information related to the environment of the subjects/objects in a space. In dynamic environments such as smart spaces, these information can be changeable. the context information plays an important role in an SS, since with the help of this information the space adapts itself with the users' preferences and provides an intelligent environment. The context information in an SS that is used for the means of health and well being can be classified as: a) environmental information (e.g., time and location), b) personal information (e.g., age and medical history), and c) medical data (e.g., blood pressure, stress level, and heart rate). A context aware application is an application that is sensitive to the context information and adapts itself accordingly. Context information in our access control scheme is considered in three different points: a) Context information decides if a situation is critical or normal. In critical situations, no control over the access is required. b) Context information may affect which role to assign to the user. c) Context information influences the privileges a user gets and and the way the rules are executed (described in detail in section 4.3). In this thesis, we modeled our access control scheme using ontological modeling techniques and OWL language, and implemented it via CLIPS rules. CLIPS is a tool developed via C language and is used for developing expert systems. Originally, it was merely a rule-based language based on the RETE algorithm (a method used for comparing sets of patterns and objects in order to find all possible matches, in rule-based systems, in an efficient way).

Privacy in this thesis is supported via data encryption techniques and the privacy rules (the access rights defined by the users to have tighter restriction on their own data). Another proposed idea is to inform users about the access to the information that they have labeled as highly sensitive. This could be implemented using Smart-M3 publish/subscribe mechanism that is supported by Smart-M3. We evaluated our access control scheme by measuring the response times, required to control the access of an incoming request for four different types of requests, including the requests to insert new data, read, update and delete the existing data on the SIB, for different sizes of the SIB, ranging from 1 to 100000 triples. We used the Python language in the experiment which is one of the supported languages by KP Interface (KPI).

The attained results showed that the response times related to the two requests for insert and update had approximately the same value in the whole experiment. We explained this behavior because of their similar functionality at triple level. While these two requests had the highest values of response times, delete was the fastest request to check the access for.

As a result, we concluded that the proposed access control scheme produces only a small overhead to the system, i.e., the highest response time was 0.2 seconds (to check the access for an insert and also the same for update request). This small overhead is acceptable due to the significance of the security in any information system.

Future work will include implementation of the security alerts via publish/subscribe mechanism supported by Smart-M3. The security alerts will be implemented for two different situations, including a) the critical situation in the user's health status, and b) access to the data that user has labeled as highly sensitive.

References

[1] United States Computer Emergency Readiness Team: http://www.us-cert.gov. Verified 2014-02-11.

[2] National Institute of Standards and Technology: http://www.nist.gov/. Verified 2014-02-11.

[3] EIT ICT Labs. http://www.eitictlabs.eu/. Verified 2014-01-30.

[4] A. D'Elia. *Semantic Service Architectures for Smart Environments*. PhD thesis, University Of Bologna, Italy, 2009.

[5] D. J. Cook and S. K. Das. How smart are our environments? An updated look at the state of the art. *Pervasive and mobile computing*, 3(2):53–73, 2007.

[6] D. J. Cook, M. Youngblood, III Heierman, E. O., K. Gopalratnam, S. Rao, A. Litvin, and F. Khawaja. MavHome: an agent-based smart home. In *Pervasive Computing and Communications, 2003. (PerCom 2003). Proceedings of the First IEEE International Conference on*, pages 521–524, Fort Worth, TX, 2003.

[7] R. Harper. *Inside the smart home*. Springer, Bristol, United Kingdom, 2003.

[8] C. D. Kidd, R. Orr, G. D. Abowd, C. G. Atkeson, I. A. Essa, B. MacIntyre, E. Mynatt, T. E. Starner, and W. Newstetter. The Aware Home: A Living Laboratory for Ubiquitous Computing Research. volume 1670 of *Lecture Notes in Computer Science*, pages 191–198. Springer Berlin Heidelberg, 1999.

[9] S. H. Park, S. H. Won, J. B. Lee, and S. W. Kim. Smart home – digitally engineered domestic life. *Personal and Ubiquitous Computing*, 7(3-4):189–196, 2003.

[10] H. Shigeta, J. Nakase, Y. Tsunematsu, K. Kiyokawa, M. Hatanaka, K. Hosoda, M. Okada, Y. Ishihara, F. Ooshita, H. Kakugawa, S. Kurihara, and K. Moriyama. Implementation of a Smart Office System in An Ambient Environment. In *Virtual Reality Short Papers and Posters (VRW), 2012 IEEE*, pages 1–2, Costa Mesa, California, 2012.

[11] J. Ma, L. T. Yang, B. O. Apduhan, R. Huang, L. Barolli, M. Takizawa, and T. K. Shih. A walkthrough from smart spaces to smart hyperspaces towards a smart world with ubiquitous intelligence. In *Parallel and Distributed Systems, 2005. Proceedings. 11th International Conference on*, volume 1, pages 370–376 Vol. 1, Fukuoka, Japan, 2005.

[12] J. A. Kientz, S. N. Patel, B Jones, E. Price, E. D. Mynatt, and G. D. Abowd. The Georgia Tech Aware Home. In *CHI '08 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '08, pages 3675–3680, New York, NY, USA, 2008. ACM.

[13] M. E. Pollack. Intelligent technology for an aging population: The use of AI to assist elders with cognitive impairment. *AI magazine*, 26(2):9, 2005.

[14] R. Costa, D. Carneiro, P. Novais, L. Lima, J. Machado, A. Marques, and J. Neves. Ambient Assisted Living. In J. M. Corchado, D. I. Tapia, and J. Bravo, editors, *3rd Symposium of Ubiquitous Computing and Ambient Intelligence 2008*, volume 51 of *Advances in Soft Computing*, pages 86–94. Springer Berlin Heidelberg, 2009.

[15] Ambient Assisted Living (AAL):http://www.aal-europe.eu/. Verified 2013-10-15.

[16] N. D. Rodríguez, R. Wikström, J. Lilius, M. Pegalajar Cuéllar, and M. Delgado Calvo-Flores. Understanding Movement and Interaction: an Ontology for Kinect-based 3D Depth Sensors. Lecture Notes in Computer Science 8276:1-8, 2013.

[17] J. Hebeler, M. Fisher, R. Blace, and A. Perez-Lopez. *Semantic web programming*. Wiley, J. & Sons, Indianapolis, Indiana, 2011.

[18] OWL Web Ontology Language: http://www.w3.org/tr/owl-features/. Verified 2014-02-11.

[19] G. Antoniou and F. Van Harmelen. Web Ontology Language: OWL. In *Handbook on ontologies*, pages 67–92. Springer Berlin Heidelberg, 2004.

[20] J. Honkola, H. Laine, R. Brown, and O. Tyrkko. Smart-M3 information sharing platform. In *Computers and Communications (ISCC), 2010 IEEE Symposium on*, pages 1041–1046, Riccione, Italy, 2010.

[21] J. Suomalainen and P. Hyttinen. Security Solutions for Smart Spaces. In *Applications and the Internet (SAINT), 2011 IEEE/IPSJ 11th International Symposium on*, pages 297–302, Munich, Bavaria, 2011.

[22] J. Suomalainen, P. Hyttinen, and P. Tarvainen. Secure information sharing between heterogeneous embedded devices. In *Proceedings of the Fourth European Conference on Software Architecture: Companion Volume*, pages 205–212, Copenhagen, Denmark, 2010. ACM.

[23] Smart-M3 Project: http://sourceforge.net/projects/smart-m3/. Verified 2014-02-11.

[24] J. F. Gomez-Pimpollo and R. Otaolea. Smart Objects for Intelligent Applications - ADK. In *Visual Languages and Human-Centric Computing (VL/HCC), 2010 IEEE Symposium on*, pages 267–268, Leganes, 2010.

[25] Finnish-Russian University Cooperation in Telecommunications: http://www.fruct.org/. Verified 2014-02-11.

[26] A. Kashevnik and N. Teslya. Context-Aware Access Control Model for Smart-M3 Platform. 2013.

[27] Clouds, Spaces and Information Sharing A Future for the Semantic Web. In *FRUCT*, St.Petersburg, Russia, 2009.

[28] J. Suomalainen. Flexible security deployment in smart spaces. In *Lecture Notes in Computer Science*, volume 7096, pages 34–43. Springer, 2012.

[29] A. Evesti, J. Suomalainen, and E. Ovaska. Architecture and Knowledge-Driven Self-Adaptive Security in Smart Space. *Computers*, 2(1):34–66, 2013.

[30] D. G. Korzun, S. I. Balandin, and A. V. Gurtov. Deployment of Smart Spaces in Internet of Things: Overview of the Design Challenges. In *Lecture Notes in Computer Science 8121: 48–59, 2013.*

[31] J. Al-Muhtadi, A. Ranganathan, R. Campbell, and M.D. Mickunas. Cerberus: a context-aware security scheme for smart spaces. In *Pervasive Computing and Communications, 2003. (PerCom 2003). Proceedings of the First IEEE International Conference on*, Fort Worth, TX, 2003.

[32] T. Berners-Lee, J. Hendler, O. Lassila, et al. The semantic web. *Scientific american*, 284(5):28–37, 2001.

[33] W. Stallings and L. V. Brown. *Computer security*. Prentice-Hall, Upper Saddle River,New Jersey, United States, 2008.

[34] A. K. Jain, A. Ross, and S. Pankanti. Biometrics: a tool for information security. *Information Forensics and Security, IEEE Transactions on*, 1(2):125–143, 2006.

[35] A. K. Jain, A. Ross, and S. Prabhakar. An introduction to biometric recognition. *Circuits and Systems for Video Technology, IEEE Transactions on*, 14(1):4–20, 2004.

[36] R. P. Wildes. Iris recognition: an emerging biometric technology. *Proceedings of the IEEE*, 85(9):1348–1363, 1997.

[37] I. R. Buhan and P. H. Hartel. The state of the art in abuse of biometrics. 2005.

[38] M. E. Smid and D. K. Branstad. Data encryption standard: past and future. *Proceedings of the IEEE*, 76(5):550–559, May 1988.

[39] J.-J. Quisquater and C. Couvreur. Fast decipherment algorithm for rsa public-key cryptosystem. *Electronics Letters*, 18:905–907(2), October 1982.

[40] E. Rescorla. Diffie-hellman key agreement method, 1999.

[41] Rosario Gennaro, Stanisaw Jarecki, Hugo Krawczyk, and Tal Rabin. Robust threshold dss signatures. In Ueli Maurer, editor, *Advances in Cryptology EUROCRYPT 96*, volume 1070 of *Lecture Notes in Computer Science*, pages 354–371. Springer Berlin Heidelberg, 1996.

[42] W. J. Caelli, E. P. Dawson, and S. A. Rea. Pki, elliptic curve cryptography, and digital signatures. *Computers & Security*, 18(1):47 – 66, 1999.

[43] M. Abadi, A. C. Goldstein, and B. W. Lampson. Compound principals in access control lists, 1994. US Patent 5,315,657.

[44] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman. Role-based access control models. *Computer*, 29(2):38–47, 1996.

[45] R. T. Simon and M. E. Zurko. Separation of duty in role-based environments. In *Computer Security Foundations Workshop, 1997. Proceedings., 10th*, pages 183–194, Rockport, MA, 1997. IEEE.

[46] US Department of Defense. *Department of Defense Trusted Computer System Evaluation Criteria*, 1985. DOD 5200.28-STD (supersedes CSC-STD-001-83).

[47] US Department of Defense. *Department of Defense National Computer Security Center, Glossary of Computer Security Terms*, 1988. NCSC-TG-004-88,FT Meade.

[48] D. R. Kuhn, E. J. Coyne, and T. R. Weil. Adding Attributes to Role-Based Access Control. *Computer*, 43(6):79–81, 2010.

[49] L. Zhang, G. J. Ahn, and B. T. Chu. A role-based delegation framework for healthcare information systems. In *Proceedings of the seventh ACM symposium on Access control models and technologies*, SACMAT ’02, pages 125–134, New York, NY, USA, 2002. ACM.

[50] S. Verma, S. Kumar, and M. Singh. Comparative analysis of Role Base and Attribute Base Access Control Model in Semantic Web. *International Journal of Computer Applications*, 46(18), 2012.

[51] T. Priebe, W. Dobmeier, and N. Kamprath. Supporting attribute-based access control with ontologies. In *Availability, Reliability and Security, 2006. ARES 2006. The First International Conference on*, Vienna, Austria, 2006. IEEE.

[52] B. Schilit, N. Adams, and R. Want. Context-aware computing applications. In *Mobile Computing Systems and Applications, 1994. WMCSA 1994. First Workshop on*, pages 85–90, Santa Cruz, CA, 1994. IEEE.

[53] M. Mohsin Saleemi, N. D. Rodríguez, J. Lilius, and I. Porres. A Framework for Context-Aware Applications for Smart Spaces. In *Applications and the Internet (SAINT), 2011 IEEE/IPSJ 11th International Symposium on*, Munich, Bavaria.

[54] O. Garcia-Morchon and K. Wehrle. Efficient and context-aware access control for pervasive medical sensor networks. In *Pervasive Computing and Communications*

Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on, pages 322–327, Mannheim, 2010.

[55] MediLexicon International Ltd. Medical Dictionaries, Drugs & Medical Searches: http://www.medilexicon.com/. Verified on 2013-08-25.

[56] G. Chen and D. Kotz. A survey of context-aware mobile computing research. Technical report, Hanover, NH, USA, 2000.

[57] S. Al-Rabiaah and J. Al-Muhtadi. ConSec: Context-Aware Security Framework for Smart Spaces. In *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2012 Sixth International Conference on*, pages 580–584, Palermo, 2012. IEEE.

[58] A. Gurtov. *Host Identity Protocol (HIP): Towards the Secure Mobile Internet*, volume 21. John Wiley & Sons, Southern Gate, Chichester, West Sussex, United Kingdom, 2008.

[59] E. Rescorla. The Transport Layer Security (TLS) Protocol version 1.1. *Transport*, 2006.

[60] I. Nikolaevskiy, A. Gurtov, and D. Korzun. Securing Interactions of Smart Objects in Smart-M3 Spaces. 2012.

[61] A. Gurtov, M. Komu, and R. Moskowitz. Host Identity Protocol: Identifier/Locator Split for Host Mobility and Multihoming. *Internet Protocol Journal*, 12(1):27–32, 2009.

[62] R. Moskowitz, P. Nikander, P. Jokela, and T. Henderson. Host Identity Protocol. *RFC5201*, 2008.

[63] K. Yudenok and K. Krinkin. Distributed service environment (smart spaces) security model development. In *12th FRUCT Conference of Open Innovations Framework Program FRUCT*, Oulu, Finland, 2012.

[64] K. Yudenok and I. Nikolaevskiy. Smart-M3 Security: Authentification and Authorization Mechanisms. 2013.

[65] R. Moskowitz, T. Heer, P. Jokela, and T. Henderson. Host Identity Protocol Version 2 (HIPv2): http://tools.ietf.org/html/draft-ietf-hip-rfc5201-bis-14.html, draft-ietf-hip-rfc5201-bis-14.

[66] P. Deshmukh and D. Croasdell. HIPAA: Privacy and security in health care networks. *Peace, AC,. & Freeman, L.(2005). Information ethics: Privacy and intellectual property. Hershey: Information Science Publishing*, pages 219–237, 2005.

[67] M. Weiser. The computer for the 21st century. *IEEE pervasive computing*, 1(1):19–25, 2002.

[68] N. Liampotis, I. Roussaki, E. Papadopoulou, Y. Abu-Shaaban, M. H. Williams, N. K. Taylor, S. M. McBurney, and K. Dolinar. A Privacy Framework for Personal Self-Improving Smart Spaces. In *Computational Science and Engineering, 2009. CSE '09. International Conference on*, volume 3, pages 444–449, Vancouver, BC, 2009.

[69] M. Werner. Privacy-protected communication for location-based services. *Security and Communication Networks*, 2011.

[70] L. Sweeney. K-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002.

[71] J. C. Chau and T. D. C. Little. Challenges in Retaining Privacy in Smart Spaces. *Procedia Computer Science*, 19(0):556 – 564, 2013. The 4th International Conference on Ambient Systems, Networks and Technologies (ANT 2013), the 3rd International Conference on Sustainable Energy Information Technology (SEIT-2013).

[72] C. Patrikakis, P. Karamolegkos, and A. Voulodimos. Security and Privacy in Pervasive Computing. *Pervasive Computing, IEEE*, 6(4):73–75, 2007.

[73] L. Pareschi, D. Riboni, S. Mascetti, and C. Bettini. Towards Privacy Protection in a Middleware for Context-awareness. *Context Awareness and Trust 2007 (CAT07)*, page 36, 2007.

[74] Microsoft TechNet: http://technet.microsoft.com/en-us/. Verified 2014-02-11.

[75] S. P. Miller, B. C. Neuman, J. I. Schiller, and Saltzer J. H. Kerberos authentication and authorization system. In *In Project Athena Technical Plan*, Cambridge, United States, 1987. Massachusetts Institute of Technology (MIT).

[76] A. Toninelli, R. Montanari, L. Kagal, and O. Lassila. A semantic context-aware access control framework for secure collaborations in pervasive computing environments. In I. Cruz, S. Decker, D. Allemang, C. Preist, D. Schwabe, P. Mika, M. Uschold, and L. Aroyo, editors, *The Semantic Web - ISWC 2006*, volume 4273 of *Lecture Notes in Computer Science*, pages 473–486. Springer Berlin Heidelberg, 2006.

[77] A. Khan and I. McKillop. Privacy-centric access control for distributed heterogeneous medical information systems. In *Healthcare Informatics (ICHI), 2013 IEEE International Conference on*, pages 297–306, Philadelphia, PA, 2013. IEEE.

[78] H. Chen, T. Finin, and A. Joshi. An ontology for context-aware pervasive computing environments. *The Knowledge Engineering Review*, 18(03):197–207, 2003.

[79] OWL for Services: http://www.ai.sri.com/daml/services/owl-s/security.html. Verified 2014-02-11.

[80] O. Sacco and A. Passant. A privacy preference ontology (ppo) for linked data. In *in Linked Data on the Web Workshop at the World Wide Web Conference*. Citeseer, 2011.

[81] A. Mhamed, M. Zerkouk, A. Husseini, B. Messabih, and B. Hassan. Towards a context aware modeling of trust and access control based on the user behavior and capabilities. In J. Biswas, H. Kobayashi, L. Wong, B. Abdulrazak, and M. Mokhtari, editors, *Inclusive Society: Health and Wellbeing in the Community, and Care at Home*, volume 7910 of *Lecture Notes in Computer Science*, pages 69–76. Springer Berlin Heidelberg, 2013.

[82] C. A. Ardagna, S. De Capitani di Vimercati, G. Neven, S. Paraboschi, F.-S. Preiss, P. Samarati, and M. Verdicchio. Enabling privacy-preserving credential-based access control with xacml and saml. In *Computer and Information Technology (CIT), 2010 IEEE 10th International Conference on*, pages 1090–1095, Bradford, United Kingdom, 2010.

[83] M. Peleg, D. Beimel, D. Dori, and Y. Denekamp. Situation-based access control: Privacy management via modeling of patient data access scenarios. *Journal of Biomedical Informatics*, 41(6):1028 – 1040, 2008.

[84] A. Toninelli, R. Montanari, L. Kagal, and O. Lassila. Proteus: A semantic context-aware adaptive policy model. In *Policies for Distributed Systems and Networks, 2007. POLICY '07. Eighth IEEE International Workshop on*, pages 129–140, Bologna, Italy, 2007.

[85] R. Zhang and L. Liu. Security Models and Requirements for Healthcare Application Clouds. In *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*, pages 268–275, Miami, FL, 2010.

[86] J. Hu and A. C. Weaver. A dynamic, context-aware security infrastructure for distributed healthcare applications. In *Proceedings of the first workshop on pervasive privacy security, privacy, and trust*, Boston, Massachusetts, USA, 2004.

[87] J. Broekstra, A. Kampman, and F. Van Harmelen. Sesame: A generic architecture for storing and querying rdf and rdf schema. volume 2342 of *Lecture Notes in Computer Science*, pages 54–68. Springer Berlin Heidelberg, 2002.

[88] Jena A Semantic Web Framework for Java: http://jena.apache.org/. Verified 2014-02-01.

[89] F. Abel, J. L. De Coi, N. Henze, A. W. Koesling, D. Krause, and D. Olmedilla. Enabling advanced and context-dependent access control in RDF stores. volume 4825 of *Lecture Notes in Computer Science*, pages 1–14. Springer, 2007.

[90] N. D. Rodríguez, M. Pegalajar Cuéllar, J. Lilius, and M. Delgado Calvo-Flores. A survey on ontologies for human behaviour recognition. *ACM Computing Surveys*, 46:132, 2013.

[91] M. Baldauf, S. Dustdar, and F. Rosenberg. A survey on context-aware systems. *International Journal of Ad Hoc and Ubiquitous Computing*, 2(4):263–277, 2007.

[92] Protégé: http://protege.stanford.edu/. Verified 2013-10-28.

[93] M. Horridge, H. Knublauch, A. Rector, R. Stevens, and C. Wroe. A Practical Guide To Building OWL Ontologies Using The Protégé-OWL Plugin and CO-ODE Tools Edition 1.0. *University of Manchester, United Kingdom*, 2004.

[94] G. Flouris, I. Fundulaki, M. Michou, and G. Antoniou. Controlling access to RDF graphs. In *Future Internet-FIS 2010*, pages 107–117. Springer, Berlin, Germany, 2010.

[95] Charles L. Forgy. Rete: A fast algorithm for the many pattern/many object pattern match problem. *Artificial Intelligence*, 19(1):17 – 37, 1982.

[96] J. C. Giarratano and G. Riley. CLIPS Reference Manual: Volume II, Advanced Programming Guide. 1998.

[97] F. Wikström. Getting Started with Smart-M3 Using Python. Technical Report 1071. TUCS, Finland.

[98] Health Care Data Security: HIPAA Draft Security Rules and California Law: http://michaelhcohen.com/healthcare-compliance/hippa/. Verified 2014-02-11.

[99] U.S. Department of Health & Human Services: http://www.hhs.gov/. Verified 2014-02-11.