XXI Autumn Meeting of Polish Information Processing Society
Conference Proceedings, pp.255-265
ISBN 83-922646-0-6

Context-aware security and secure context-awareness in ubiquitous computing environments

Konrad Wrona and Laurent Gomez

SAP Research,
805, Avenue du Docteur Maurice Donat, 06250 Mougins, France
konrad.wrona@sap.com, laurent.gomez@sap.com

Context-awareness is emerging as an important element of future wireless systems. In particular, concepts like ambient intelligence and ubiquitous computing rely on context information in order to personalize services provided to their end users. However, security implications of employing context-awareness in computing systems are not well understood. Security challenges in context-aware systems include integrity, confidentiality and availability of context information, as well as end user's privacy. Another interesting and open question is to what extent availability of additional context information could be used in order to optimise and reconfigure security-related services themselves.

1. Introduction

Ubiquitous computing is referring to scenarios in which computing is omnipresent, and particularly in which devices that are traditionally perceived as dumb are endowed with computing capability [1]. In ubiquitous computing, context information (such as user's location, time, etc.) can have a strong impact on application adaptation. We consider that application adaptation is performed at both application logic and security management levels. At the application logic level, an application can, e.g., modify contrast of the screen depending on the surrounding brightness. At the security management level, an application can, e.g., adapt an encryption mechanism used for communication with remote application depending on the nature of network over which data will be exchanged. We start with presenting the state of the art in context-aware computing in Section II.

Processing of context information introduces two major challenges. The first one is related to a heterogeneous character of context information: context can come from different context information provider and can be of different kinds. A thermometer delivers temperature whereas user's device provides its IP address in a wireless network. The challenge raised by the diversity of context is discussed in Section III, where we present an initial taxonomy of context information. The second challenge is security in context-aware systems. In Section IV, we describe new challenges and opportunities for information security arising in context-aware environments, and in particular in ubiquitous computing environment. In Section V we present possible ways of using context information to influence security services, in particular in order to manage security and trust in context-aware systems. Finally, in Section VI we present one of the ongoing European research projects focusing on security in context-aware systems.

2. Context-aware computing

As more advanced mobile applications begin to emerge, we can observe an increasing interest in use of context information in mobile environment [2], [3], [4]. Several terminologies and classifications for context information have been proposed. Section II-A provides a comprehensive set of definitions related to context-aware computing, which we use in our research. In Section II-B, we describe a life-cycle of a context in a mobile application environment.

2.1. Terminology

As context information we understand any kind of information, which can be used to characterize the state of an entity. An entity might be any kind of asset of a computing system such as user, software, hardware, media storage or data [5]. Moreover, we define as context information provider (CIP) any kind of entity which delivers context information. A context information provider can be for example a thermometer, a GPS receiver or a watch. In this paper, we call a system context-aware if it uses any kind of context information before or during service provisioning, including, e.g., service design, implementation, and delivery. The smart floor system is a good illustration of a context-aware system [6]. The smart floor is a device equipped with force measuring sensors, so that it can detects users walking on it. The purpose of such system can be to identify users. In such case, the context-aware system is the backend system to which users authenticate themselves; the context information is the force pressure measured by the smart floor; and the smart floor acts as the context information provider.

2.2 Life-cycle of Context

We assume that a context information provider delivers context information to a context-aware system following the life-cycle illustrated in Figure 1. The main steps in a life-cycle of context information are:

- **Discovery of context information**: In this step, a context-aware system discovers available context information providers. The discovery can be performed either in a push or a pull mode [7], i.e. the context-aware system can actively look for CIPs or can passively receive information about available CIPs.
- **Acquisition of context information**: In this step, a context-aware system collects context information from the discovered context information providers and stores it in a context information repository for further reasoning. Similar to the process of discovery, the acquisition is performed either in a pull or a push mode. In a pull mode, the context-aware system explicitly requests for context information whereas in a push mode, context information providers push context information to the context-aware system. For example, a GPS receiver can either push every second a geographical position to a context-aware system or the context-aware system can pull the GPS receiver every second for the geographical position.
- **Reasoning about context information**: reasoning mechanisms enable applications to take advantage of the available context information. The reasoning can be performed based on a single piece of context information or on a collection of such information. For example, in the case of a context-aware e-Health application, user's health status can be evaluated based on both his heart rate and blood pressure provided by medical sensors.

Fig. 1. Context life cycle.

2.3. Context Architecture

Several architectures have been developed for supporting discovery, acquisition and reasoning about context information. Two particularly interesting approaches are represented by the Context Toolkit Architecture [8] and the Context Broker Architecture (CoBrA) [9].

The Context Toolkit Architecture aims at facilitating development and deployment of context-aware applications. The basic components of this architecture are context widgets, interpreters, aggregators and a discoverer. Context widgets are software components that provide access to context information. Aggregators collect all the relevant context information for a specific application. Interpreters provide abstraction of context information by reasoning about them. Finally, a discoverer aims at determining what context can be currently sensed from the environment. Application can either get access to aggregator, interpreter or a widget. Moreover,

the Context Toolkit enables basic access control for privacy protection. Figure 2 gives an overview of the Context Toolkit Architecture.

The Context Broker Architecture (CoBrA) proposes architecture for discovery, acquisition and reasoning about context information. It includes also mechanisms for privacy protection of context information. CoBrA assumes that all context information providers have a prior knowledge about the broker's presence and that they communicate with the context broker via a standardized protocol. CoBrA is based on a Web Semantic Ontology, which is described in Section III. CoBrA architecture is depicted in Figure 3.

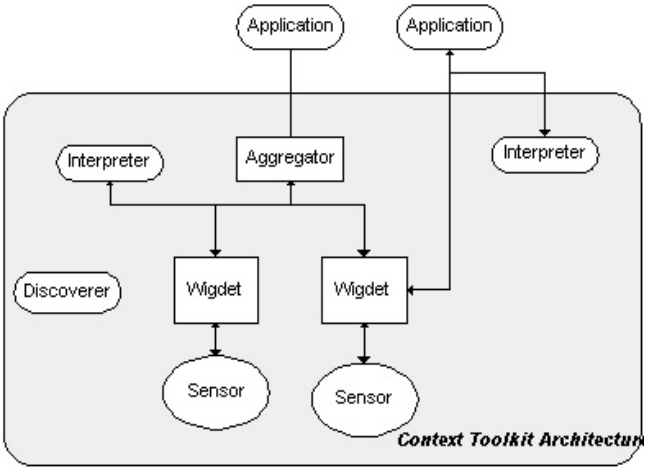

Fig. 2. Context Toolkit Architecture (arrows show the flow of context information).

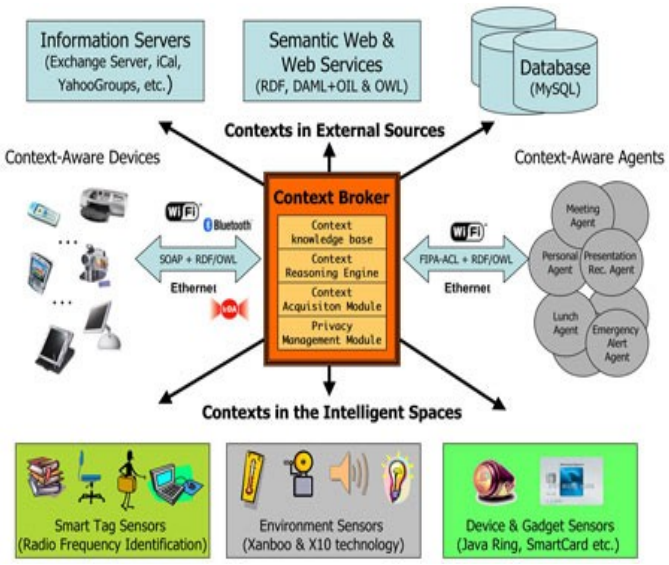

Fig. 3. Context Broker Architecture.

3. Context ontology

Context ontology is needed in order to deal with a heterogeneous character of context information. In particular, and ontology focuses on identifying objects by classifying them and characterizing them with properties [10]. Below we present an early attempt of a classification of context information, including identification of common properties and description of relationships between them. We also briefly introduce the Web Ontology Language [11], which can be used for representing context ontology.

3.1. Context taxonomy

We first aim fat establishing a simple taxonomy for context information, based on various approaches proposed in the literature.

In general, context information can consist of very different information. User's location is one of the most popular kinds of context information [12]. However, other common kinds of context include, e.g., user's identity, activity pattern, IP address and time of transaction.

We can identify the following four basic categories for context information [13], [14]:

System context: A mobile application has to take into account context information related to both the computing system it is running on, e.g. the particular type of mobile device, and to communication system being used, e.g. the particular type of wireless network. System context deals with any kind of context information related to a computing system, e.g. computer CPU, network, IP address, status of a workflow, wireless network, etc. In order to provide a fine-grained classification of system context, we can use model based on context information provided by different layers of Open System Interconnections (OSI) model [15]: application, presentation, session, transport, network, data link, and physical. Figure 4 provides an example of context information available per layer. For example, a workflow status would be provided by the application layer whereas an IP address would be related to the network layer.

User context: refers to any kind of context information related to the user and characterizing him. User context information can be user's age, location, medical history, etc. Biometric information, such as fingerprint, iris or face shape, is a part of user context, too. User context can be also related to his emotions [16], even if it is currently difficult to capture this kind of information in computing systems. User context can also include context information related to user's tasks, social connections, personal state, and spatio-temporal information [17]. The task context captures user's tasks, goals and current activities. The social context focuses on capturing the social relationships, such as family, friends, and colleagues. The personal context describes physical and mental information about the user. The spatio-temporal context refers to the location or movement of the user.

Environmental context: consist of any kind of context information related to the physical environment, which is not covered by system and user context. Environmental context information include, e. g., lighting, temperature, weather, etc.

Temporal context: defines any kind of context information related to time. Time and day are typical temporal context information.

Figure 4 depicts the taxonomy of context information described above.

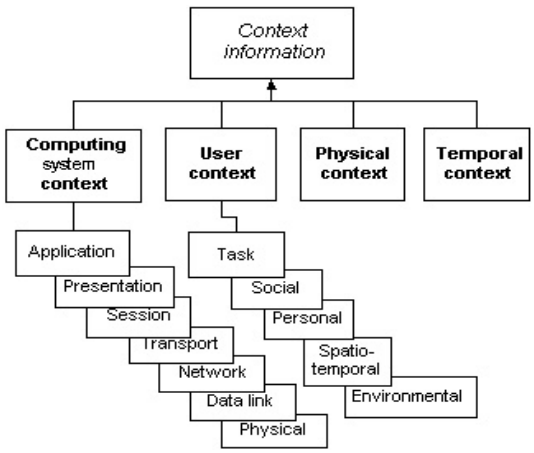

Fig. 4. Classification of context information.

Any context information can be classified as type-of object class as defined in Figure 4. For example, an IP address is considered as type-of Network context information which is a subtype-of System context. Our proposed taxonomy should be considered as a basis for context information classification, and therefore it can be extended. Nevertheless, classification of context information is not sufficient. It is necessary to identify the properties of this context information. In the next section, we describe a set of properties that we believe are common to many kinds of context information.

3.2. Attributes of context information

In this section our goal is to present a non-exhaustive list of attributes which characterize various kinds of context information. Figure 5 illustrates this basic set of attributes of context information. In particular, we have identified the three main properties of context information:

Persistence of context information: We consider persistence of context information as relative to the application life-time. In particular, static context information does not change (or changes very slowly). Static context information can include user's address, information name, room temperature, etc. At the contrary, dynamic context information changes much more often. Dynamic context information includes time, user's location, etc.

Origin of context information: context information can be provided by either an internal or an external source. For example, in the case of an application hosted by a mobile phone, context information coming from the mobile phone itself is considered as internal context information; whereas context information provided, e.g., by a GSM operator (such as user's location) is considered as external context information. We believe that the distinction between internal and external context information is important for the evaluation of, e.g., quality and trustworthiness of context information. In particular, internal context information, such as battery level, can be assumed more reliable than external context information such as network coverage guaranteed by the mobile operator.

Quality of context information: an important attribute of context information is its quality. For example, various approaches have been proposed in order to increase accuracy of GPS location information for outdoor user's localization or to provide user's location indoor based on Wi-Fi networks [18]. Gray [19] introduced criteria for evaluation of quality of the context information. Based on the quality of the context information, the acquisition can be repeated periodically, either in order to maintain the freshness of the context information or when some events in the system occur [7]. The importance of the quality will be further discussed in section IV.
As stated previously, the above list of common property is not exhaustive. Nevertheless, we believe that it provides a good basis for further research.

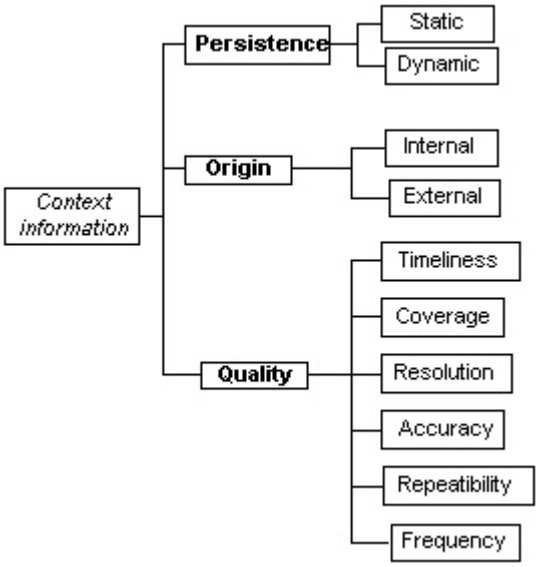

Fig. 5. Attributes of context information.

3.3. Context Information Relationship

Even if context information is clearly classified and if a list of common properties is established, relationships between different kinds context information have to be identified in order to enable further reasoning about context. For example, an e-Health application may acquire context information related to a patient's health status such as a patient's heart rate, pulse and blood pressure. We can assume that an equivalence relation exists between heart rate and pulse [11]. Moreover, we could state, e.g., the following correlation between context information: if pulse lower than 10 and blood pressure lower than 10 then health status is unconscious. During the reasoning phase, if pulse is not available, heart rate can be used for reasoning about the patient's health condition. Identification of relationships between context information permits aggregation of context information at the application level: instead of having different context information from different sources, correlation compiles sets of context information in single and comprehensive information, which can be used by the application. The goal of aggregation and reasoning about context information is to avoid flooding an application with too much context information and to provide a global context understanding. Nevertheless the quality of the reasoning about context information is based on the properties of the individual context information. Depending on the individual quality of a set of context information, reasoning about correlated imperfect context information determines the quality of this new context information [20]. The complexity of reasoning about imperfect information may constitute a significant obstacle for the success of context-aware system.

3.4. OWL-based Context Ontology

The Web Ontology Language (OWL) has been developed as a part of the Semantic Web initiative sponsored by World Wide Web Consortium (W3C) [11]. Based on the XML standard, OWL is a knowledge representation language for defining and instantiating of ontologies. Standard Ontology for Ubiquitous and Pervasive Applications (SOUPA) [21] is an ontology covering knowledge sharing, context information reasoning and inter-operability in ubiquitous environments. Expressed using OWL, SOUPA ontology consists of two sets of ontologies: SOUPA Core and SOUPA Extension. SOUPA Core defines a typical vocabulary for nine basic concepts: person, agent, belief-desire-intention, action, policy, time, space and event. For example, for the person concept, SOUPA Core defines attributes such as first name, last name or contacts. SOUPA Extension enables the definition of new set of vocabulary. Existing SOUPA extensions include location, image capture, and region connection calculus.

4. Context and security

Context ontology provides a conceptual mechanism for understanding of context at the application-level. This context can be further used at application-level for the adaptation of application's logic and management of security. However, such context-aware systems face important security challenges related to the use of context information: the main issue is to how to get secure and trusted context information. In section IV-A, we focus on the management of trusted and secure context information. The goal is to provide to context-aware system reliable context information. Afterwards, we discuss in section V how trusted context can be integrated as input for trust and security management in context-aware system. We also explain in that section the benefits, which mobile application can gain from using context-aware security.

4.1. Secure context

During each step of the life-cycle of context, see also Figure II-B, context-aware systems face specific security and trust challenges. We consider the three main security needs in computer system: confidentiality, integrity and availability [5].

- Confidentiality: focuses on protecting of computer system assets (Computer system assets include hardware, software, media storage and data) from unauthorized entities; For example, user can choose to disclose his age only to specific authorized entities.
- Integrity: focuses on ensuring that computer system assets can be modified only by authorized entities; For example, temperature measurements delivered by thermometer must be reliable and not modified by any entity. Context information spoofing is

- Availability: focuses on ensuring that computer assets must be available to the authorized entities. A GPS receiver must ensure to provide user's location anywhere and anytime.

Those three security goals are also relevant for a context-aware system, with respect to information exchanged between the context information provider and requestor during the context life-cycle. We have identified the following security issues related to exchange of context information:

Confidentiality of context information: (also called context information privacy). Considering user's context information (see also section III-A), user should be able to protect personal information such as his health status, or medical history. Jason proposes an architecture so-called Confab in order to provide privacy in ubiquitous computing [22]. Confab framework has been designed for protecting a user's location information in ubiquitous systems. It is based on an analysis of privacy needs for end-users and application developers. The main idea is that personal information are captured, stored and processed as much as possible on the user's device. Users can apply security control on their choice on those data. Context views have been introduced by Shankar [23]. Context view is a fraction of an entity's context that is relevant to the application. In order to protect context information confidentiality, delivered context information can be controlled in context view. Bussard et al proposed security mechanisms for protecting privacy of context information. [24] These context information are used for further trust relationship establishment between two entities.

Integrity of context information: focuses on guaranteeing that the provided context information has not been corrupted by a third party. Hash functions or public key digital signatures can provide context integrity.

- Hash function: We consider two kind of hash function: un-keyed and keyed hash functions. In the first case, un-keyed hash functions, such as MD5 or SHA-1, provide a very low level of data integrity with respect to the use of context in application adaptation. With keyed hash function, such as MAC, we have the issue of the establishment of pre-shared key between the context information providers and the context-aware systems.
- Public key digital signature: the PKI-based approach might not be always suitable for context-aware systems, especially in distributed systems including low-cost sensors [25].

Availability of context information: denial of service attacks could be easy to perform on a context information provider. Possible denial-of-service attacks on context information providers include the sleep deprivation torture or CPU exhaustion [25]. However, we do not focus on this security challenges here.

When considering that the application logic and the security configuration can depend on context, we have to ensure that context information is trustworthy. In particular, we have to be able to estimate the level of trust a context information requester can put in the delivered context information. Due to the fact that context information provider and the context-aware system might be members of different administrative domains, it is very important to be able to differentiate trustworthy parties from un-trustworthy ones. Defining trust mechanisms and metrics is not an easy task [26], [27]. In the scope of context-aware system, we consider that trust refers to the reliability and accuracy of context information. In this sense, trust is related to the quality of the delivered context information, but not restricted to it. We can evaluate the trust about delivered context information by computing the distance between it and the real context. The following approaches can be used in order to compute this distance and evaluate the trustworthiness of delivered context information:

Statistical analysis of context information: The idea is to perform statistical analysis of the value of context information in order to detect unreliable ones. A simple example is to take an average of temperature values provided by different thermometer in the same room. Such naïve approach raises the following issue: if the temperature values collected are (10; 10; 11; 50) we get an average of over 20. It is obvious that the last value is a wrong one, and the temperature in the room should be 10. A simple solution is to use median that detects that value 50 is out of the scope, and eliminate the context information provider delivering this temperature value as an unreliable provider. Of course, real-life implementation requires use of much more sophisticated statistical tools.

Distributed reputation network: Trustworthiness is often described as the expectation of cooperative behavior [28]. Usually the trustworthiness is based on previous experience with the same party. The problem is that often there has not been any direct interaction before. In this case, an entity can establish trust in its communication partner based on the latter's reputation [29-31]. Reputation is based on the collection of evidence of good and bad behavior. In the case where we use sensors in order to identify and authenticate users, their reputation depend, e.g., on whether the sensors input led to correct authentication or to a violation of a security policy. Reliability of context information is computed based on the previous experiences with a context information provider. The mathematical foundations for reputation management are rooted in statistics and probability [32]. Trust context views [33] have been introduced by Robinson in order to establish a degree and state of trust within a certain interactive context. This approach doesn't consider the authentication of context information provider.

Confidence value: In the scope of Gaia architecture, Al-Muthadi et al [34] define a notion of confidence values in context-aware authentication. Gaia architecture establishes a confidence value based on the authentication devices and protocol used. For example, iris recognition would have a better confidence value

than a fingerprint authentication device. The use of challenge-response protocol would be considered with a strong confidence value than a user's identity sent in clear text.

Above considerations about security and trust are relevant for discovery and acquisition of context information. Reasoning about context information raises another security challenge. Indeed, reasoning about context information often deals with integrating several pieces of context information into another piece of context information. For example, from a patient's heart rate and blood pressure, we can evaluate his health condition. Assuming that patient's heart rate and blood pressure has different level of trust and security, the question is about the level of trust and security about the patient's health condition.

4.2. Context-aware security

Context information can be also used in order to reconfigure and enhance security of the system and of the applications. We define context-aware security as dynamic adaptation of system security policy according to the context. For example, a context-aware system can exploit user's location for access control: if the user is in a public area, such as airport, he should not be able to display confidential data in a way it could be seen by a third party. Context information could also be used in order to automatically reconfigure security mechanisms in order to provide a predefined level of security, at the same time optimising use of resources. For example, email messages send by a mobile worker using WLAN hotspot as an access point for his PDA can be automatically encrypted, whereas the same massages could be sent in plaintext when connected to an access point belonging, which employs the worker.

As we have already mentioned, one of the main challenge in context-aware security is estimation of trust we put in context information. This is especially true, if context information is to be used in order to enforce security policies in mobile applications or to alter security configuration of a portable device.

Below, we describe trust management and access control as two candidates for security services integrating context information.

4.3. Access control

Access control is a set of mechanisms and processes that aim at protecting assets of a computer system from being accessed by unauthorized entities. Access control determines whether an access request to a resource or data is granted or denied [35]. Access control policy defines the set of controls or permissions in the computing system. Permission is normally associated with a subject, such as a user. It is a set of rights for a specific object. Access control policy is commonly enforced through identification of a subject and a decision of granting or denying access to an object, according to the permissions associated to the subject identity. For example, user A authenticates himself to a computing system. A requests access to file C. As permission B allows user A to read and write in file C, access control enforcement grants access to file C to user A. Access control targets the three main security goals: authentication, authorization and accounting. Authentication is the process whereby one party is assured of the identity of a second party involved in a protocol, and that the second has actually participated (i.e., is active at, or immediately prior to, the time the evidence is acquired). Authorization deals with the rights on an asset associated to a principal. Finally, accounting focuses on logging all requests for access, as well as the corresponding access decisions. Context-aware access control can be easily illustrated by the following example: a doctor should have the rights to book a room for a patient in a hospital or plan for a surgery only if he is physically in the hospital and not on vacation. In our e-Health scenario, emergency team member can get access to all health information about victim, only if the latter is unconscious. Based on context information, proximity-based and encounter-based access control can be defined [36]. Proximity-based access control defines a set of security rules based on the proximity of an entity to other entity or group of entities. Another example of a context-aware system that exploit user's location for access control could be as follow: if the user is in a public area, such as an airport terminal, he should not be able to display confidential data in a way it could be seen by untrusted third parties. In the scope of access control, we can also consider context-aware delegation of rights. For example, if a manager is out of office, he can delegate some of his rights to his secretary until he is back in his office. Depending on the urgency of certain task, context-aware system can decide whether delegation can take place or not with respect to particular tasks.

Several architectures have been developed for context-aware access control:

Environmental Roles: Covington et al [37] propose a uniform access control framework for environmental roles. It is an extension to the Role-Based Access Control (RBAC) model. In RBAC model, permissions are associated with roles. In an administrative domain, a role can be a developer or a manager. A role determines

the user's position or ability in an administrative domain. Likely, an environmental role is a role that captures environmental conditions. Environmental roles are based on General Role Based Access Control [38]. Unlikely in the RBAC model which is only subject-oriented, GRBAC allows defining access control policy based on subject, object or environment. An implementation of the environmental roles is provided by, e.g., CASA.

Gaia: Roman et al [39] defined generic context-based software architecture for physical spaces, so-called Gaia. A physical space is a geographic region with limited and well defined boundaries, containing physical objects, heterogeneous networked devices, and users performing a range of activities. Derived from the physical space, Active Space provides to the user a computing representation of physical space. Active Space helps the user to interact with the physical space. Cerberus is a framework for context-aware identification, authentication and access control and reasoning about context, based on Kerberos [40] authentication and Gaia [34]. Cerberus focuses on user's identification via user's context information such as fingerprint, voice and face recognition.

Authorization mechanisms for Intranet: This context-aware authorization architecture, based on Kerberos authentication, enables to activate or deactivate role assigned to a user depending on his context [41]. For example, if a user is in a un-secure place such as airport terminal, the access to sensitive data is denied whereas in the corporate building of the user, he has access to the confidential data.

4.5. Trust management

Several trust models have been proposed in the literature [26]. In the scope of ubiquitous environment, we target the trust management of mobile application across multiple domains. It means that mobile application involved in collaboration may not trust a priori each other. When considering collaborative mobile applications, we can identify the following history information, which can influence the trust relationship during the collaboration:

Respect of common security policy: Assuming that mobile application negotiate a common security policy at the beginning of the collaboration, if one of the entity involved in the collaboration breaks the common security policy it should decrease the trust that the others entities will have in the latter.

Respect of the common goal: The collaboration aims at achieving a specific goal, e.g., sharing a file in a secure manner, exchange contact, etc. If actions of one the entities is not focused on the goal of the collaboration, it level of trust should decreases. At the contrary, if an entity fulfills its entire obligation in the collaboration, its level of trust should increase.

As we discussed in Section IV-A, trust is a complex concept. Defining a generic model of trust is a very challenging and so far unsolved issue [27]. By exploiting the context information, we aim at managing trust between entities in collaboration on a case by case basis. We also aim at providing a dynamic evaluation of level of trust in context information itself. This issue is directly related to risk management. In particular, mobile applications can accept the risk of using un-trustable context information, either if it is in urge or it doesn't really care of the reliability of the context information. On the other hand, user or application can decide to fix a threshold of trust, below which context information is not taken into account when considering adaptation of application logic or security mechanisms.

4.6. Ongoing and future work

The secure context awareness and context aware security are currently field of active research in both academic and industrial community. Mobile Workers' Secure Mobile Application Collaboration in Ubiquitous Environment (MOSQUITO) is a joint European research project funded under EU FP6 IST Programme [21]. It aims at providing a secure framework for collaborative mobile business application. In particular, within MOSQUITO we aim at developing a framework for context-aware security services in ubiquitous computing environment. In order to deal with security of context information, we introduce in MOSQUITO a concept of Context Information Provider (CIP). Trusted CIP (TCIP) is responsible for providing accurate and secure context information to context-aware applications. We consider several classes of TCIPs:

Third Party TCIP: a stand-alone system operated by a party belonging to a different administrative domain than the entity requesting context information. This one is for instance operated by a mobile network operator.

Intra-Domain TCIP: a service that is operated by an entity belonging to the same administrative domain as the entity requesting context information. Intra-domain TCIP can be a stand-alone service or a service integrated in another server.

Embedded TCIP: a tamper-resistant component of a (mobile) device that ensures integrity and authenticity of provided context information.

5. Conclusions

In this paper we have focused on investigation of two different aspects of security related to context information.

First, we presented challenges related to security of context information. Security challenges in context-aware systems include integrity, confidentiality and availability of context information, as well as end user's privacy. Another important issue is trustworthiness of context information, i. e. the amount of trust, which a context information requester can put in the delivered context information.

Second, we have discussed how availability of additional context information could be used in order to manage and optimize security configuration of the mobile devices and services offered to the user.

6. Disclaimer

IST-Directorate General / Integrating and strengthening the ERA: the project MOSQUITO is supported by the European Community. This document does not represent the opinion of the European Community. It is also the sole responsibility of the author and not the responsibility of the European Community using any data that might appear therein.

References

1. F. Stajano, *Security for Ubiquitous Computing*, John Wiley and Sons, Feb. 2002.
2. A. K. Dey and G. D. Abowd, *Towards a better understanding of context and context-awareness*, in Proceedings of the CHI 2000 Workshop on The What, Who, Where, When, and How of Context-Awareness, The Hague, Netherlands, Apr. 2000. ftp://ftp.cc.gatech.edu/pub/gvu/tr/1999/99-22.pdf
3. *Toward a better understanding of context attributes*, in Proceedings of the second IEEE Annual Conference on Pervasive Computing and Communications Workshops (PERCOMMW 04). IEEE Computer Society, 2004.
4. A. K. Dey, *Understanding and using context*, Personal and Ubiquitous Computing Journal, vol. **5** (1), pp. 4–7, 2001. http://www.cc.gatech.edu/fce/ctk/pubs/PeTe5-1.pdf
5. C. P. Pfleeger, *Security in Computing*, Prentice-Hall, Inc., 1997.
6. R. J. Orr and G. D. Abowd, *The Smart Floor: A Mechanism for Natural User Identification and Tracking*, in Conference on Human Factors in Computing Systems (CHI 2000), The Hague, Netherlands, April 2000, pp. 1–6.
7. N. J. Siljee B. I. J, Bosloper I. E, *A classification framework for storage and retreival of context*, in Proceedings of First International Workshop on Modeling and Retrieval of Context (MRC2004), 2004.
8. A. K. Dey, *Providing architectural support for building context-aware applications*, Ph.D. dissertation, Georgia Institute of Technology, Nov. 2000. http://www.cc.gatech.edu/fce/ctk/pubs/deythesis.pdf
9. H. L. Chen, *An intelligent broker architecture for pervasive contextaware systems*, 2004.
10. *What are ontologies, and why do we need them?* 1999.
11. W3C, *Owl - web ontology language*, http://www.w3.org/2004/OWL/
12. U. Hengartner and P. Steenkiste, *Implementing access control to people location information*, in SACMAT '04: Proceedings of the 9th ACM symp. on Access control models and technologies. ACM Press, 2004, pp. 11–20.
13. B. Schilit, N. Adams, and R. Want, *Context-aware computer applications*, Proceedings of the Workshop on Mobile Computing Systems and Applications, pp. 85–90, Dec. 1994.
14. G. Chen and D. Kotz, *A survey of context-aware mobile computing research*, Tech. Rep. Dartmouth Computer Science Technical Report TR2000-381, 2000.
15. ISO, *Osi - open system interconnection.* http://www.iso.org/
16. R. W. Picard, *Affective computing: challenges*, Int. J. Hum.-Comput. Stud., vol. 59, no. 1-2, pp. 55–64, 2003.
17. A. K.-P. Marius Mikalsen, *Representing and reasoning about context in a mobile environment*, 2004.
18. J. Small, A. Smailagic, and D. Siewiorek, *Determining user location for context-aware computing through the use of a wireless lan infrastructure*, Carnegie-Mellon University, Tech. Rep. Project Aura report. http://www.cs.cmu.edu/ aura/docdir/small00.pdf
19. P. Gray and D. Salber, *Modelling and using sensed context information in the design of interactive applications*, in Engineering for Human-Computer Interaction: 8th IFIP International Conference, EHCI 2001, (LNCS), vol. 2254, 2001. http://www.springerlink.com
20. K. Henricksen and J. Indulska., *Modelling and using imperfect context information*, in In Second IEEE Annual Conference on Pervasive Computing and Communications Workshops, 2004.
21. H. C. F. P. T. Finin and A. Joshi, *Soupa: Standard ontology for ubiquitous and pervasive applications*, 2004. http://ebiquity.umbc.edu/v2.1/get/a/publication/105.pdf

22. J. I. Hong and J. A. Landay, *An architecture for privacy-sensitive ubiquitous computing*, in MobiSYS '04: Proceedings of the 2nd international conference on Mobile systems, applications, and services. ACM Press, 2004, pp. 177–189.
23. N. Shankar and D. Bafanz, *Enabling secure ad-hoc communication using context-aware security services*, in UNBICOMP 02: Workshop on Security in Ubiquitous Computing, 2002. http://www.teco.edu/philip/ubicomp2002ws/organize/palo.pdf
24. M. R. Bussard L., Roudier Y., *Untraceable secret credentials: Trust establishment with privacy*, in PERCOMMW'04. Second IEEE Annual Conference on Pervasive Computing and Communications Workshops, 2004.
25. F. Stajano and R. Anderson, *The Resurrecting Duckling: Security issues for ad-hoc wireless networks*, in Proceedings of the 7th International Workshop on Security Protocols, ser. Lecture Notes in Computer Science, M. R. B. Crispo, Ed. Springer, 1999.
26. K. Wrona, *Cooperative communication systems*, Ph.D. dissertation, Department of Wireless Communications, University of Aachen, Germany, 2005.
27. M. A. P. S. T. M. C. Michael, *Clifford Information Security Analyst*,The Aerospace Corporation Matt Bishop Associate Professor The University of California at Davis "Defining, computing and interpreting trust," 2000. http://csdl.computer.org/comp/proceedings/acsac/2000/0859/00/08590088.pdf
28. E. I. Covington M. Ahamd M. and V. H., *Parameterized authentication*, 2004. http://www.springerlink.com
29. S. Ganeriwal and M. B. Srivastava, *Reputation-based framework for high integrity sensor networks*, in SASN '04: Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks. ACM Press, 2004, pp. 66–77.
30. B. Yu and M. P. Singh, *An evidential model of distributed reputation management*, in AAMAS '02. Bologna, Italy: First International Conference on Autonomous Agents and MAS, July 2002.
31. S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, *The eigentrust algorithm for reputation management in p2p networks*, in In Proceedings of the Twelfth International World Wide Web Conference, 2003.
32. G. Shafer, *A mathematical theory of evidence*, 1976.
33. P. Robinson, *Trust context spaces an infrastructure for pervasive security in context-aware environments*, 2003.
34. R. C. Jalal Al-Muhtadi Anand Ranganathan and M. D. Mickunas, *Cerberus: A context-aware security scheme for smart spaces*. 2003. http://gaia.cs.uiuc.edu/papers/cerberus.pdf
35. P. Samarati and S. de Capitani di Vimercati, *Access control: Policies, models, and mechanisms*, 2001.
36. R. S. Roshan K.T, *Models, protocols and architecture for secure pervasive computing: Challenges and research directions*, in PERCOMMW' 04. Second IEEE Annual Conference on Pervasive Computing and Communications Workshops, 2004.
37. Z. Z. Michael J. Covington Prahlad Fogla and M. Ahamad, *A contextaware security architecture for emerging applications*, in Proceedings of the Annual Computer Security Applications Conference (ACSAC), Las Vegas Nevada USA, 2002.
38. M. J. C. M. J. Moyer and M. Ahamad., *Generalized role-based access control for securing future applications*, 2000.
39. M. R. C. K. H. R. C. A. R. R. H. Campbell and K. Nahrstedt, *Gaia: A middleware infrastructure to enable active-spaces*, 2002. http://gaia.cs.uiuc.edu/papers/GaiaSubmitted3.pdf
40. B. C. Neuman and T. Ts'o., *Kerberos: An authentication service for computer networks*, 1994. http://nii.isi.edu/publications/kerberos-neuman-tso.html
41. M. L. Wullems Chris and A. Clark, *Toward context-aware security: an authorization architecture for intranet environments*, 2004.