Journal of King Saud University – Computer and Information Sciences (2011) 23, 71–78

King Saud University

Journal of King Saud University – Computer and Information Sciences

www.ksu.edu.sa
www.sciencedirect.com

ORIGINAL ARTICLE

Access control using threshold cryptography for ubiquitous computing environments

Jalal Al-Muhtadi [a,*], **Raquel Hill** [b], **Sumayah Al-Rwais** [a]

[a] *College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia*
[b] *School of Informatics and Computing, Indiana University, Bloomington, IN, USA*

Received 10 May 2010; accepted 10 November 2010
Available online 5 May 2011

KEYWORDS

Access control;
Ubiquitous computing;
Pervasive computing;
Threshold cryptography

Abstract Ubiquitous computing is revolutionizing the way humans interact with machines and carry out everyday tasks. It extends everyday computing into the physical world, creating computationally smart environments that feature seamless interactions and automation. As a result of the highly distributed nature of ubiquitous computing, it is essential to develop security mechanisms that lend themselves well to the delicate properties of smart ubiquitous computing environments. In this paper, we introduce a context-aware access control mechanism that utilizes threshold cryptography and multilayer encryption to provide a dynamic and truly distributed method for access control. We simulate our access control scheme and show that access control decisions can be made in a timely manner even as we increase key and file sizes. This mechanism is closely coupled with the context-capturing services and security policy service resulting in a fully context-aware and seamless access control mechanism for typical ubiquitous computing scenarios.

1. Introduction

Ubiquitous computing is revolutionizing the way humans interact with machines and carry out everyday tasks. Through the use of sensors, actuators and context awareness the virtual world is highly intermingled with the physical world, creating profound opportunities for enabling computationally smart spaces with automation, seamless interactions and everywhere anytime services. Information security in such environments that spread across the virtual and physical environments continues to be challenging yet underdeveloped. Many approaches either ignore security issues altogether or try to apply traditional mechanisms that do not lend themselves well to the highly dynamic and truly distributed nature of ubiquitous computing environments. This is particularly true for access control mechanisms, as traditional mechanisms need some

* Corresponding author.
E-mail addresses: jalal@ksu.edu.sa (J. Al-Muhtadi), ralhill@indiana.edu (R. Hill), salrwais@ksu.edu.sa (S. Al-Rwais).

1319-1578

Peer review under responsibility of King Saud University.
doi:10.1016/j.jksuci.2011.05.003

Production and hosting by Elsevier

kind of centralized or semi-centralized authorization server or rely on reference monitors or other primitives to mediate every attempted access by a user. Such solutions do not blend well with the highly distributed, dynamic and context aware nature of ubiquitous computing environments. In this work, we introduce a context-aware access control mechanism that utilizes threshold cryptography and multilayer encryption to provide a dynamic and truly distributed method for access control. This mechanism is closely coupled with the context-capturing services and security policy service resulting in a fully context-aware and seamless access control mechanism for typical ubiquitous computing scenarios.

Researchers have begun to study how context and ubiquitous computing can make hospitals and healthcare networks more efficient work environments (Bardam et al., 2004; Holzinger, 2005). One area where context may help is in defining data access policies. Hospitals and healthcare networks are environments, where information security mechanisms are not effective because policies focus on efficiency instead of data protection. Recent headlines report that 15 hospital workers were fired because they reviewed Nadia Suleman's (aka Octomom) patient record without permission (http://www.foxnews.com; http://www.healthleadersmedia.com). In a similar privacy breach at a UCLA hospital, information related to Farrah Fawcett's cancer treatment was given to the National Enquirer. As a result, 165 employees with positions ranging from doctors to orderlies were fired, suspended or warned (http://www.foxnews.com). The lack of privacy and confidentiality of patient records is not a new problem. In 1995, 24 people in Maryland were indicted for selling patient information from the state's Medicaid database to four HMOs (Woodard, 1995). As stated by Woodard (1995), the biggest threat to digital patient records is confidentiality. ''Even before the introduction of the computer, confidentiality deteriorated as care provided by large groups became more common. But computerized records, particularly if embedded in large networks designed to collect comprehensive lifelong data, can rapidly accelerate that trend'' (Woodard, 1995).

As reported in 1995 (Woodard, 1995) and even today, most hospitals and healthcare networks allow all staff to access digital patient records even when the person does not have direct care responsibility for the patient. While advocates argue that unrestricted access is more efficient, such access limits the effectiveness of security mechanisms like passwords and encryption. As evidenced by previous breaches, healthcare personnel sale information, share passwords and use other means to subvert the system. Passwords and encryption do not restrict the behavior of authorized users. Therefore, we propose to supplement the use of these mechanisms with contextual information that determine when and under what conditions a patient's record can be accessed by individuals who do not have direct care responsibility for the patient. For example, when a patient's doctor or nurse is not available, any staff doctor or nurse should be allowed to view the patient's record to administer care. Additionally, when a medical emergency occurs, any doctor or nurse should be able to access patient information.

In this paper, we offer that security mechanisms used within the proper context can ensure confidentiality policies as well as meet requirements for efficiency. ''Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves'' (Dey, 2001). In this paper, we incorporate threshold cryptography into the context-based encryption mechanism presented in Al-Muhtadi et al. (2006). In this work, we extend the use of context to provide context and location-based security mechanisms for providing confidentiality and restricting access. We envision context being used in conjunction with identity and roles to provide context-based encryption services, which provide finer-grain access control services and efficient key management for group communication. Encrypting files mitigates the need for complicated access control mechanisms or reference monitors that mediate every attempted access by a user. This advantage is even more crucial in a ubiquitous computing environment, where it is common to have different pieces of data stored on a plethora of different devices with various capabilities and processing powers. These can include lightweight devices such as PDAs, smart phones, and smart watches. In such a setting, it is infeasible to implement sophisticated access control checks on these devices.

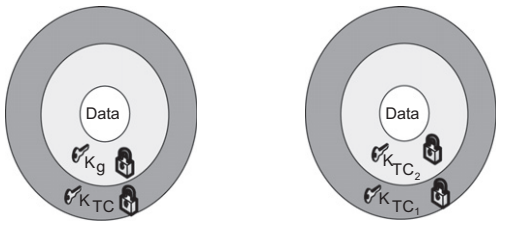
We use threshold cryptography to provide a mechanism for defining how high-level context information is interpreted, and to provide a secure mechanism for enforcing that interpretation. Our novel use of threshold cryptography as a mean for capturing contextual information for access control decisions enables us to leverage the dynamic nature of context to support flexible access control policies and to distribute trust among components within the ubiquitous computing infrastructure.

The remainder of this paper is organized as follows. Section 2 presents our approach and an overview of threshold cryptography. In Section 3 we present a detailed description of our system architecture. Section 4 presents the related work. In Section 5 we present a motivating scenario and access control policy. Section 6 presents the implementation and evaluation and we conclude in Section 7.

2. Approach

In this paper, we use threshold cryptography to enable context-aware access control. Context-aware access control merges data from multiple context sensors and uses this data to determine whether users should be given access to context restricted resources. Our context-aware access control scheme extends the context-based encryption scheme that is presented in Al-Muhtadi et al. (2006). In Al-Muhtadi et al. (2006), encryption is used to restrict access to data resources. Access to resources is limited to users within a specific geographic region. Location data is used to determine whether data should be decrypted or not. The decrypted data is then given to users whose location has been verified.

To enable context-aware access control, we introduce the idea of encrypting sensitive data using the secret-sharing mechanism (threshold cryptography, which we will refer to as TC from now on) which is based on the idea of sharing a secret between different entities. A secret is divided into a number of secret shares. In order to derive the secret, a pre-specified number of entities must collaborate to obtain the secret. Threshold schemes are (k-out-of-n), where n is the total number of all entities and k is the pre-specified number of entities which must join forces to derive a secret. Variants of RSA cryptographic algorithms utilized the idea of threshold schemes by sharing the private key as the secret resulting in



(a) An encryption layer based on secret-sharing which can only be decrypted if the right high-level context is realized

(b) Both encryption layers are based on secret-sharing one is for the context and the other is for the subjects

Figure 1 Multi-layer encryption.

one public key and n private key shares. There are two models for threshold schemes which are used mostly in RSA cryptographic algorithms. They are either single sharing threshold based on Lagrange's interpolation as proposed by Shamir (1979) or threshold sharing functions like geometric based threshold as in Desmedt and Frankel (1990). In this project, we adopt Shamir secret sharing scheme which is based on polynomial interpolation. Assuming that the secret (d) is a number, to divide d into pieces d_i; we pick a random $k-1$ degree polynomial:

$$f(x) = a_0 + a_1 x + \cdots + a_{k-1}x^{k-1} \quad \text{where } a_0 = d.$$

Given any subset k of these $(i, f(i))$, the coefficients of $f(x)$ can be found by interpolation and evaluate $f(0)$ which is d. But knowledge of just $k-1$ is not enough to calculate d. In order to restrict access to data until a higher-level contextual situation is realized, we identify the various low-level sensor data that the higher-level context situation can be derived from, in a similar fashion to what was done in previous work (Ranganathan et al., 2004). For example, if enough sensors in a smart space provide readings that are consistent with a meeting activity in the space, then this should be enough to assume a meeting is taking place, and thus access to data can be granted. It is possible to add multiple layers of encryption to capture richer access control policies. For example, some sensitive data can only be decrypted when a specific number of authorized users (k-out-of-n) are in the right location or under the right contextual situation. Sometimes it might be necessary to accommodate scenarios where several conditions must be present to grant access to the data, in which case, we can select an n-out-of-n secret sharing scheme for the TC layer, assuming the accuracy of all these sensors are sufficient. Alternatively, it is possible to introduce two layers of encryption, where the first layer consists of the required conditions and uses an n-out-of-n scheme, and the other layer consists of these conditions that do not need to be satisfied fully for granting access, and thus, using a k-out-of-n scheme as illustrated in Fig. 1.

3. System architecture

In this section we give a brief overview on how our mechanism works. Our system consists of a general-purpose distributed middleware, made up from distributed components that are developed using Java RMI. These components provide the common core functionality for enabling smart spaces and their applications. The main components include a policy service, context service, and event service. The policy service manages security policies and encryption keys. The context service processes sensor data to derive high level context. The event service provides secure communication among components within the system. Fig. 2 provides an architectural overview of the system. Component details are provided in the following subsections.

3.1. Policy service

The policy service provides primitives for security administrators to create and manage security policies for the smart space environment. Security policies are layered. Each layer of a security policy has at least one corresponding contextual condition. The policy service generates an encryption key for each layer and encrypts the data. The policy service decomposes into n key shares, where n corresponds to the number of contextual conditions that are associated with a layer. The policy service then distributes these key shares to sensor brokers within the smart space environment. Each layer of the security policy is sent to a context manager. Sensor brokers and context managers are components within the context service. Descriptions of these components are provided below.

3.2. Context service

The context service captures and processes contextual information from various sensors. Various contextual information are captured using various sensors, like temperature, lighting levels, sound levels, time and date, schedule, patient vital signs, etc. High-level activities (e.g., a closed meeting taking place in a specific room, etc.) can be implied by fusing sensors or gathering raw data from various sensors, and deriving higher-level contextual information. For example, if the environment is able to detect the presence of several people, who are sitting at a large table in a room and talking in an orderly fashion, then this could imply that a meeting is taking place. The context service supports deducing high-level activities from low-level sensors.

The design of the context service is based on the ideas outlined by Ranganathan et al. (2004) and Ranganathan and Campbell (2003). The context service middleware consists of two components, context managers and sensor brokers. The context managers use first order logic to reason about contextual situations and derive higher-level or abstract contexts from sensor data. Sensor brokers mediate access to data produced by sensors and provide primitives for enabling communication with other components and services in a smart space infrastructure. For lightweight sensors, sensor brokers are simply a dedicated PC on which the sensor's component runs. Sensor brokers store the key shares of the corresponding sensors.

A context manager is responsible for deriving the higher-level context that corresponds to a layer of encryption. When a context manager receives k decryption shares from the sensor brokers, the manager removes one layer of encryption. The process repeats until all layers are removed. The decrypted data is sent back to the user. If an insufficient number of conditions are met, the data cannot be decrypted and the access operation fails. Fig. 3 illustrates this process.

3.3. Event service

Another key component is the event service that allows events to be communicated between distributed objects. The event

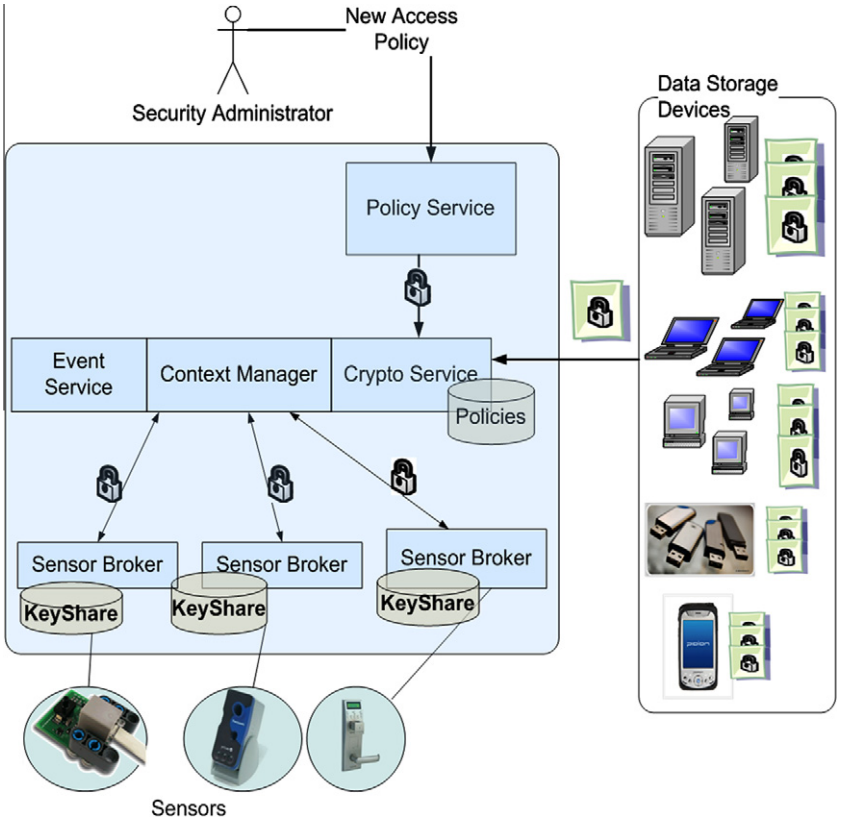


Figure 2 Architectural overview.

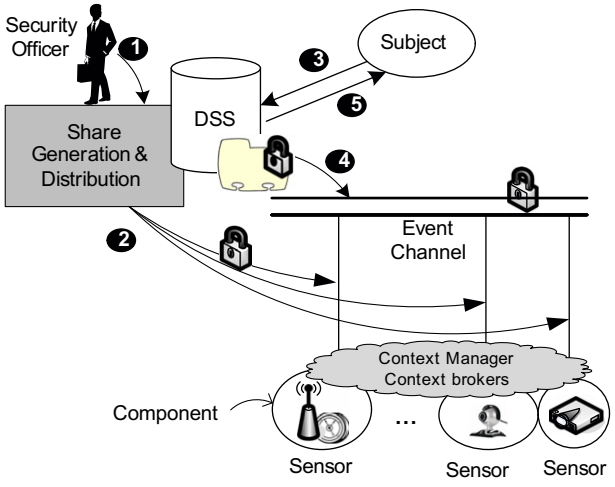


Figure 3 The use of TC for context-based access control.

service utilizes a publish-subscribe model for dispersing events. With the event service, users can create secure event channels where channel participants are restricted to authorized entities and sensitive events are encrypted, as described in Lee et al. (2005). All relevant sensor components within the smart space infrastructure are connected through a special secure event channel as depicted in Fig. 3. A detailed description of Fig. 3 is provided below.

Step 1: Once the security policy for accessing sensitive data is specified, a security officer can specify the necessary conditions that satisfy the high-level context or the identity and/roles of subject(s) that are authorized to access that data. This will depend on the sensitivity of the information and the sensor availability and setup in the smart space.

Step 2: Once the requirements for access are specified, the secret shares are generated according to Shamir's secret sharing scheme. Using a secure end-to-end connection over an event channel, the secret shares are distributed to the components of the relevant sensors, as well as meta-data to identify the condition that needs to be met for a given sensor component (or a context synthesizer component) to apply its share to the data. An encryption "layer" can now be added to the data. The encrypted data is stored in the Distributed Storage Service (DSS) – the DSS implements functionality similar to a file system in a traditional OS, with the addition of context-awareness and support for data distribution across various smart devices and the cloud, i.e., encrypted data can be stored at different locations, including PDAs or the cloud, yet the DSS manages to aggregate the data so that it virtually appears to be stored on a single location (Hess, 2002).

Steps 3 and 4: When a subject requests access to the data, the data is first sent to the event channel of the sensors' component. Each sensor will apply its share to the encrypted data if the appropriate context is realized. If enough sensors participate, that layer of encryption is removed.

Step 5: If the TC layer is successfully removed, then the remaining data is passed to the subject. It is possible to have multiple layers of encryption here, where each layer needs to be decrypted to access the data. The inner layer can be concerned with validating the identity or the role of the requestor.

In scenarios where k-out-of-n subjects must be authenticated to access the data, the inner layer can also be based on a secret sharing scheme, as illustrated in Fig. 1b.

The benefits of this scheme is that trust is distributed rather than trusting a single entity for making decisions on whether the higher-level context is realized or not. In addition, it allows the system to cope with some level of uncertainty if some sensors are unable to give accurate readings.

In addition to applying TC to context, we can employ TC techniques to add an additional layer of encryption to data stored on the DSS or transmitted over event channels. This data can only be decrypted when a specific number of authorized users (k-out-of-n) are in the right location or under the right contextual situation. Sometimes it might be necessary to accommodate scenarios where several conditions must be present to grant access to the data, in which case, we can select an n-out-of-n secret sharing scheme for the TC layer, assuming the accuracy of all these sensors are sufficient. Alternatively, it is possible to introduce two layers of encryption, where the first layer consists of the required conditions and uses an n-out-of-n scheme, and the other layer consists of these conditions that do not need to be satisfied fully for granting access, and thus, using a k-out-of-n scheme.

4. Motivating scenario

To address the problem of ensuring patient privacy, health information systems in a smart hospital environment should require delicate security and privacy policies. Patient records should be kept secure and only accessible under specific circumstances that can be specified at a fine level of granularity. Such fine grain mechanisms are not currently employed because of concerns for timely and efficient access to medical records. Security mechanisms should not delay or prohibit patient care. To this end, we define a complex security policy to test the performance bounds of the threshold cryptography scheme. We do not suggest that the policy is viable for hospital environments, but use it only to illustrate that our approach is efficient even when used to implement complex security policies.

4.1. Access policy

To assess the performance and the practicality of our approach we simulate a smart hospital emergency scenario and apply our approach for dynamic access control. First we assume a three tier policy for data access within our environment. A tier maps to a layer of encryption within our scheme, and the policy that corresponds to the tier defines the context that must be satisfied before the corresponding layer of encryption is removed. We begin my describing the policy at layer 3 which is the outer most layer and conclude with layer 1.

Layer 3 – The policy at layer three specifies when a patient's records may be viewed: The patient's records may be viewed by a doctor when the patient is being admitted to the hospital, or during the hours that the doctor makes his/her rounds, or if the patient is experiencing a medical emergency. Examples of a medical emergency would be: heart rate is over or under a prespecified threshold, high temperature, high blood pressure or any other vital signs. During our simulation, one of the specified conditions must be met before the outermost encryption layer can be removed.

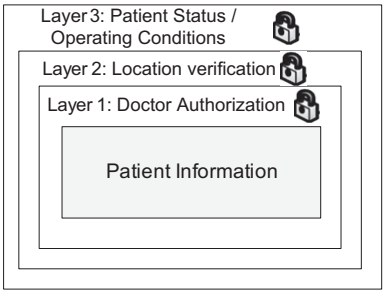


Figure 4 Multi-layer encryption.

Layer 2 – The policy at layer two concerns the location of patient and doctor: The patient's records may be viewed when both the doctor and patient are located within the hospital and the doctor is within the patient's room or in close proximity to the patient. During our simulation, both conditions must be met before the second layer of encryption can be removed.

Layer 1 – The policy at layer one concerns verifying the doctor's identity and credentials: The doctor may view a patient's records if he is the attending physician, or he is the charge physician, who is filling in for the attending physician, and he is affiliated with the hospital. During our simulation at least two of the conditions must be met before the final layer of encryption is removed.

Fig. 4 illustrates the idea of the multi-layer encryption.

Using our mechanism, for the patient's information to be decrypted, each layer should be decrypted (peeled off) only if the right context is realized. This is managed by the policy service which executes the security policy to decrypt the information. Decryption starts with the outermost layer and proceeds if the conditions are met at each successive layer. The patient's records are fully decrypted and made available to the doctor via mobile device if all conditions are met.

5. Implementation and evaluation

Shoup (1999) proposed a practical RSA threshold signature and decryption scheme that is based on Shamir secret sharing scheme. We adopt this algorithm. Our implementation is based on an open source implementation (Weis, 2006), which we improved by adding decryption capabilities and other enhancements. All services and objects are implemented as standalone distributed objects using Java RMI. We use secure event channels for secure communication between the various objects. For our evaluation purposes we run all components on an Intel Core 2 Duo 2.4 GHz machine and we simulate a variety of contextual information to test the system. We focus on simulating and testing the smart hospital scenario that was described in the previous section. The scenario performance is evaluated using four file sizes for patient data; 200, 500, 1000 and 5000 bytes. For each configuration, 20 readings are taken and averaged. We evaluated for three different key sizes. Two time measurements were recorded for the performance of the scenario: (1) "with transmission time," which measures the total time taken from the issuance of a decryption request until the decrypted data is received (includes processing and data transmission, etc.), (2) "without transmission time," where we do not take into account data transfer time, as we are trying to focus on cryptographic

Table 1 Smart hospital scenario with key sizes 128, 256 and 512 bit for layer 1, 2 and 3, respectively.

File size (bytes)	200	500	1000	5000
Without transmission time (ms)	341	369	348	372
With transmission time (ms)	499	533	526	554

Table 2 Smart hospital scenario with key sizes 256, 512 and 1024 bit for layer 1, 2 and 3, respectively.

File size (bytes)	200	500	1000	5000
Without transmission time (ms)	2339	2363	2368	2367
With transmission time (ms)	3189	3215	3232	3233

Table 3 Smart hospital scenario with key sizes 512, 1024 and 2048-bit for layer 1, 2 and 3, respectively.

File size (bytes)	200	500	1000	5000
Without transmission time (ms)	16,757	17,315	15,928	18,337
With transmission time (ms)	22,574	23,125	21,759	24,188

speed evaluation and ignore data transmission between distributed objects as the latter is very dependent on connection type, bandwidth and latency. Results of the evaluation are shown in Tables 1–3 and Figs. 5 and 6.

6. Model security analysis

The primary goal of proposed framework is to control access to information thus ensuring privacy and confidentiality of information. Information will be disclosed only when a certain pre-specified, context-aware, access policy is realized. In this section, we review the security of the model by analyzing the viable threats to the system.

The security policy definition process involves the distribution of policy key shares and creation of event channels. Possible threats include theft of key shares, and thus, impersonating sensor brokers. Another threat could be eavesdropping on the event channel communication, and thus, being able to combine decryption shares returned from the sensor brokers in order to come up with the decrypted data. These threats are addressed as follows. The key shares are encrypted symmetrically using AES-128 (or better) with the pre-specified context conditions' keys. Event channel communication is secured using symmetric encryption with a group key. The group key is exchanged with the event channel's subscribers after authentication is established by asymmetrically encrypting it with the subscribers' public key.

Possible threats during the data decryption phase include the faking of context data at the sensor side when it is requested by the sensor broker and attacking the key shares stor-

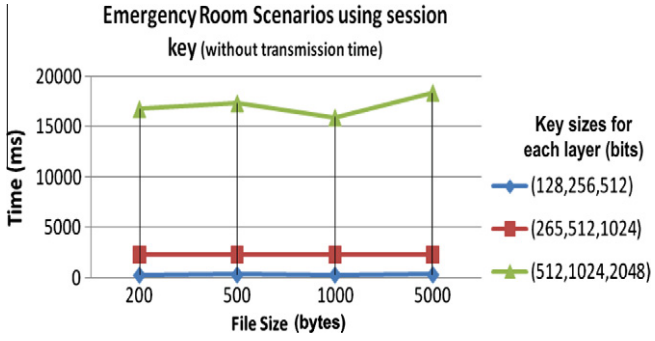


Figure 5 Smart hospital scenario performance without transmission time.

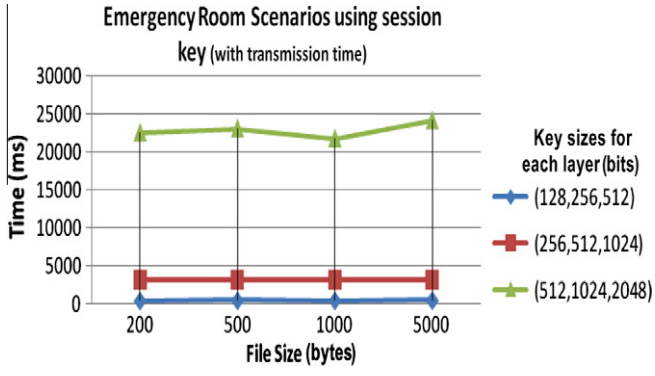


Figure 6 Smart hospital scenario performance with transmission time.

age at the sensor brokers. These threats are addressed by ensuring that communication between sensor brokers and sensors is secured through mutual authentication and the use of encryption, thus making it difficult to send fake sensor data. The key shares stored at the sensor brokers are safely kept because they are already encrypted by the pre-specified context conditions' keys.

Furthermore, in the proposed framework, the level of trust assigned to sensor brokers is distributed and not centralized at one place. If an attacker was able to fake one sensor's data it would have a limited effect on the access control decision, since the model is using threshold cryptography (k-out of-n) and a number of k context conditions will have to be realized for the data to be decrypted. The same applies if an attacker was able to impersonate a certain sensor broker. This also makes the system resilient to some level to a limited number of faulty sensors.

7. Related work

Much recent work on access control systems for ubiquitous and pervasive computing has been based on the Role-based Access Control system (Sandhu et al., 1996) (RBAC). RBAC relies on the principle that access control decisions are based on the roles individuals take on as part of an organization. The key concept in RBAC is a role, which is a placeholder for a set of users. Each role is associated with a set of permissions, which are its rights on objects. These roles may be organized into a hierarchy to reflect the organizational hierarchy among different users in a system. RBAC has been adapted for use in pervasive computing environments (Gill et al., 2001; Viswanatha, 2001; Covington et al., 2000), and the concept of roles is extended to deal with context information. However, RBAC is not sufficiently flexible to handle spontaneous changes in context in an optimized manner. Furthermore, RBAC requires a separate mechanism to enforce the access decisions, in the form of a reference monitor to something similar. The Aware Home project has extended RBAC with object and environment roles (Covington et al., 2000, 2001, 2002) that are used to define context-aware security policies such as those based on temporal authorizations. However, they do not address permissions under specific high-level contextual situations. Kumar (2001) also consider incorporating context into the RBAC model with contexts and context filters. dRBAC (Freudenthal et al., 0000) is a decentralized trust-management and access control mechanism for systems spanning multiple administrative domains.

8. Conclusion and future work

In this paper, we present a novel framework that enables context-aware access control via the use of encryption and threshold cryptography. The approach is novel in that it combines the use of TC and heterogeneous high-level contexts to make an access control decision. In addition, trust is distributed throughout the ubiquitous computing infrastructure. We believe that contextual awareness can enrich traditional security mechanisms with greater flexibility and expressiveness power and enable a variety of security services. Our simulations show that our multilayered access control mechanisms can operate efficiently even for complex scenarios and increasing key sizes.

In this work, we have illustrated how our threshold cryptography enabled access control mechanism can be used to enforce policies that have multiple contextual conditions. We also envision this mechanism being used in situations to increase the confidence in sensor readings by combining the output of multiple sensors via the use of threshold cryptography. We foresee context being used in conjunction with identity and group membership to provide finer-grain access control services, location-based encryption services and efficient key management for group communication.

References

Al-Muhtadi, J., Hill, R., Campbell, R., Mickunas, D., 2006. Context and location-aware encryption for pervasive computing environments. In: Third IEEE International Workshop on Pervasive Computing and Communication Security (PerSec).

Bardam, J., 2004. Applications of context aware computing in hospital work – Examples of design principles. In: ACM Symposium on Applied Computing.

Covington, M.J., Moyer, M.J., Ahamad, M., 2000. Generalized role-based access control for securing future applications. In: 23rd National Information Systems Security Conference.

Covington, M.J., Long, W., Srinivasan, S., Dey, A.K., Ahamad, M., Abowd, G.D., 2001. Securing context-aware applications using environment roles. In: SACMAT, Virginia, USA.

Covington, M.J., Fogla, P., Zhan, Z., Ahamad, M., 2002. A Context-aware security architecture for emerging applications. In: 18th ACSAC, Las Vegas, NV.

Desmedt, Y., Frankel, Y., 1990. Threshold Cryptosystems, Advances in Cryptology – Crypto '89, pp. 307–315.

Dey, A.K., 2001. Understanding and using context. Personal and Ubiquitous Computing 5 (1), 4–7.

Freudenthal, E., Pesin, T., Port, L., Keenan, E., Karamcheti, V., 2002. dRBAC: Distributed role-based access control for dynamic coalition environments. In: 22nd International Conference on Distributed Computing Systems.

Gill, B.S., 2001. Dynamic Policy-Driven Role-Based Access Control for Active Spaces. University of Illinois at Urbana Champaign.

Hess, C.K., 2002. A Context File System for Ubiquitous Computing Environments, University of Illinois at Urbana-Champaign, Urbana-Champaign, CS Technical Report UIUCDCS-R-2002-2285 UILU-ENG-2002-1729.

Holzinger, A, Schwaberger, K., Weitlaner, M, 2005. Ubiquitous computing for hospital applications RFID applications to enable research in real-life environments. In: 29th Annual International Computer Software and Applications Conference.

<http://www.foxnews.com/story/0,2933,511780,00.html>.

<http://www.healthleadersmedia.com/content/230986/topic/WS_HLM2_HR/Octomom-Records-Breach-a-Lesson-in-Patient-Privacy.html>.

Kumar, E.A., 2001. IBM RI 02007.

Lee, A., Boyer, J., Drexelius, C., Naldurg, P., Hill, R., Campbell, R., 2005. Supporting dynamically changing authorizations in pervasive communication systems. In: 2nd International Conference on Security in Pervasive Computing.

Ranganathan, A., Campbell, R.H., 2003. An infrastructure for context-awareness based on first order logic. Personal and Ubiquitous Computing 7, 353–364.

Ranganathan, A., Al-Muhtadi, J., Campbell, R.H., 2004. Reasoning about uncertain contexts in pervasive computing environments. In: IEEE Pervasive Computing Magazine.

Sandhu, R., Coyne, E., Fienstein, H., Youman, C., 1996. Role based access control models. In: IEEE Computer, vol. 29.

Shamir, A., 1979. How to share a secret. Communications of the ACM 22, 612–613.

Shoup, V., 1999. Practical Threshold Signatures. IBM Research Report RZ 3121.

Viswanathan, P., 2001. Security Architecture in Gaia. University of Illinois at Urbana-Champaign.

Weis, S., 2006. Java threshold Signature package. Available from: <http://sourceforge.net/projects/threshsig/>.

Woodard, B., 1995. The Computer-Based Patient Record and Confidentiality, New England Journal of Medicine. Available from: <http://content.nejm.org/cgi/content/full/333/21/1419>.